A UNIFIED GAME-THEORETIC METHODOLOGY FOR THE JOINT LOAD SHARING, ROUTING AND CONGESTION CONTROL PROBLEM

Volume I

by

Anastasios A. Economides

A Dissertation Presented to the

FACULTY OF THE GRADUATE SCHOOL

UNIVERSITY OF SOUTHERN CALIFORNIA

In Partial Fulfillment of the

Requirements for the Degree

DOCTOR OF PHILOSOPHY

(Computer Engineering)

December 1990

UNIVERSITY OF SOUTHERN CALIFORNIA
THE GRADUATE SCHOOL
UNIVERSITY PARK
LOS ANGELES, CALIFORNIA 90089-4015

This dissertation, written by

ANASTASIOS A. ECONOMIDES

under the direction of h.is..... Dissertation Committee, and approved by all its members, has been presented to and accepted by The Graduate School, in partial fulfillment of requirements for the degree of

DOCTOR OF PHILOSOPHY

Barbara Solomon
Dean of Graduate Studies

Date August 23, 1990

DISSERTATION COMMITTEE

Chairperson

Peter H. Baxendale

Στους γονείς μου,

Αχιλλέα και Μαρία.

Acknowledgments

I would like to express my sincere gratitude to my advisor and Chairman of my dissertation committee Professor J. A. Silvester, for his valuable guidance, constant encouragement and sound advice on the area of computer network performance evaluation. Despite his busy schedule as Chairman of the Computer Engineering Division, he always had the time for raising questions and making useful suggestions for the improvement of this research work.

I am deeply indebted to my co-advisor Professor P. A. Ioannou, for his support, constructive criticism, real life advice and enriching conversations on the area of adaptive control.

My thanks to Professor P. Baxendale, for his advice and comments on the area of stochastic processes. I would also like to thank Professor M. Dubois, for his time and enlightening discussions on the area of multiprocessor performance evaluation. I am grateful to Professor G. Papavassilopoulos, for his influence and inspiring teaching on the area of optimization and game theory.

Special thanks to Dr. Vikram Saksena of AT&T, who suggested to me the virtual circuit routing problem, to my officemates: Thomas Papavassiliou and Syu-Je Wang for the cooperative and friendly environment, and to Mazmur Ongkowid for his programming advice. Thanks also to all other colleagues of our research group in the Computer Networks and Distributed Systems Laboratory, namely: Dr. J. Dill, N. Fonseca, A. Lin, Dr. H. Liu, K. Ramakrishnan, Dr. E. Sousa, Dr. J. L. Wang, S.-M.S. Wang, J.S. Yang and Dr. Ch. Yan. Thanks to my numerous Greek friends at U.S.C. with whom I have spent hours on philosophical and scientific discussions.

And most importantly, I am grateful to my parents Achillea and Maria and my sister Helen for their absolute love, devotion and support in everything. Words are not enough to show my appreciation.

Finally, I would like to acknowledge the financial support of the I.K.Y., I.I.E. and AT&T.

Contents

List Of Tables

List Of Figures

Abstract

In this dissertation, we introduce a unified game-theoretic methodology for the multi-objective joint load sharing, routing and congestion control problem in distributed systems. We further propose stochastic learning automata algorithms for decentralized asynchronous dynamic computation of the solution.

First, we define and model the problem on the path flow space using queueing and state space models. Then we develop three methodologies for both the quasi-static and the dynamic cases of the problem: i) Team optimization methodology, when the classes of jobs cooperate for the socially optimum, ii) Nash game methodology, when the classes of jobs compete among themselves and each class tries to operate optimally for its own jobs and iii) Stackelberg game methodology, when some classes of jobs have more power than others, for example priority classes.

For each methodology, we formulate the problem as a Nonlinear Programming, an Optimal Control, a Dynamic Programming, a Nonlinear Complementarity and a Variational Inequality problem. We state conditions for existence/uniqueness of the solution, and derive the optimality conditions for the quasi-static problem using the Karush-Kuhn-Tucker theorem, and for the dynamic problem using Pontryagin's maximum principle.

We apply the proposed methodologies to Datagram, Virtual Circuit and Integrated Services Networks and develop several new queueing models and performance measures, for each network type. We explicitly solve several examples and evaluate the system performance via simulation.

Finally, we propose decentralized asynchronous dynamic load sharing, routing and congestion control using Stochastic Learning Automata. We introduce new classes of Stochastic Learning Automata algorithms and use them as decision makers in the distributed system. We explicitly illustrate their operation with an example for dynamic virtual circuit routing and demonstrate via simulation improved system performance.

Chapter 1

Introduction

1.1 Problem Statement

Jobs arrive at the source node of a distributed system requiring processing and communication (Figure 1.1). However, not all of the jobs are admitted into the system, because this may cause saturation of some system resources. The problem of controlling these external arriving jobs is the *congestion control* problem. The processing of jobs can be done at the source nodes or at some other destination nodes. The problem of selecting the node for processing a job is the *load sharing* problem. After selecting the destination node for processing, the job should be transferred there. Also, jobs arrive at the source nodes simply requiring transfer to a specific destination node. A job that requires transfer to a destination node may arrive there through one of several paths. The problem of selecting the best path for transferring them to their destination is the *routing* problem. All these externally arriving jobs are assigned to the distributed system resources in an efficient way, such that some cost function is minimized. While previous research have concentrate on each problem in isolation or at most at two problem combined, its is important to analyze all problems simultaneously, since the decisions for each problem affects that of another. In this dissertation, we analyze the joint load sharing, routing and congestion control problem.

Load sharing, routing and congestion control algorithms can be classified according to how dynamic the decisions are in : 1) static, 2) quasi-static, and

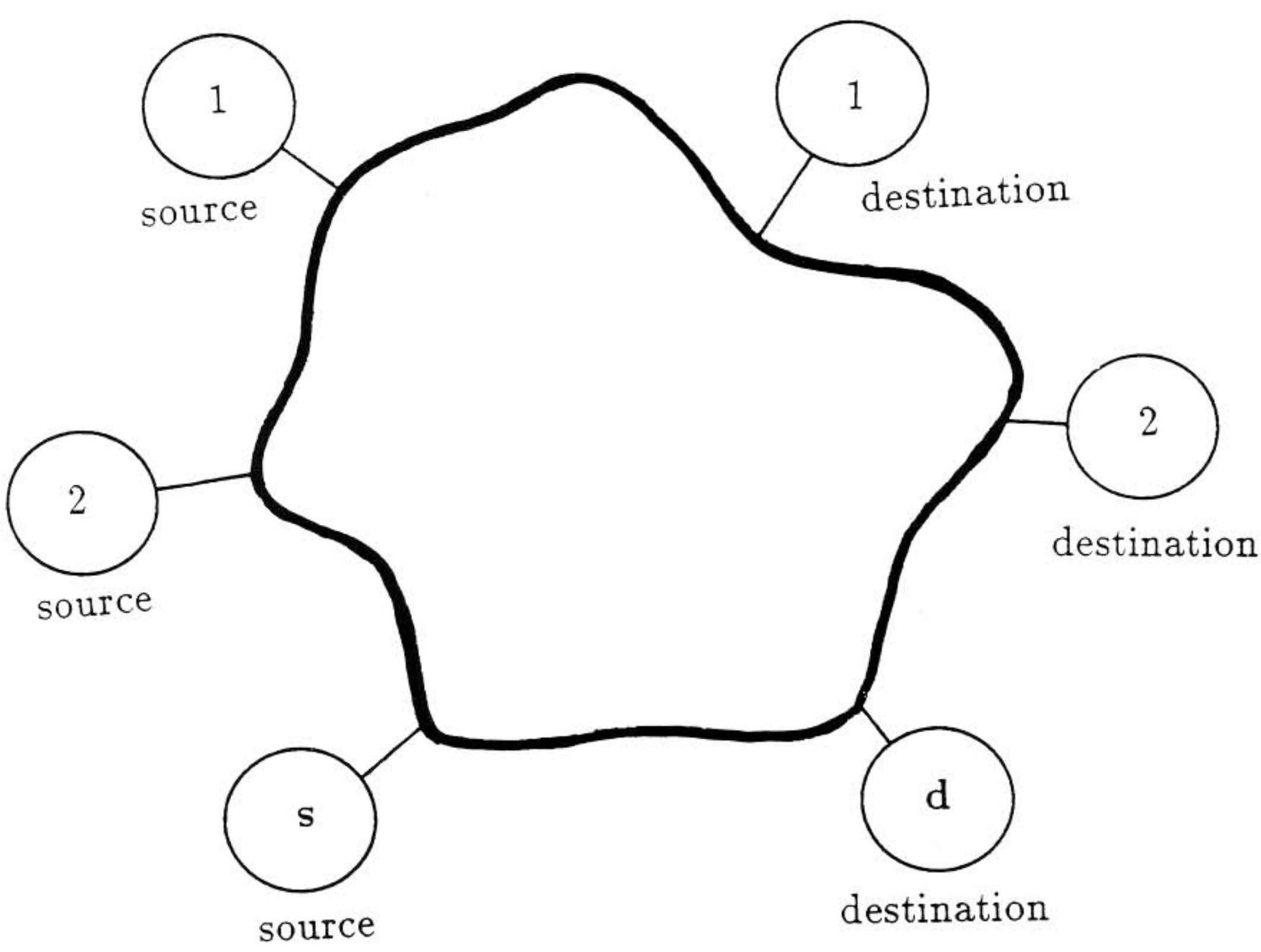


Figure 1.1: A Distributed System.

3) dynamic or adaptive. In *static algorithms*, the decisions are independent of the current system conditions. Static algorithms can be : a) fixed, when there is a predetermined set of alternative actions; b) random, where the traffic is split according to fixed fractions. In *quasi-static algorithms*, the decisions depend partially on the current network conditions but some network parameters are assumed to be stationary over time. In *dynamic algorithms*, the decisions depend on the current network conditions (topology changes, traffic conditions). Since in reality, network conditions change over time the decisions in dynamic algorithms may achieve better performance.

The dynamic problem in any distributed system is a difficult problem even under Markovian assumptions. The resulting Markov Chain does not have product form solution because the transition probabilities depend on the network state. Also, since a dynamic algorithm should depend on the current network state we must find the transient solution of the corresponding Markov Chain with time dependent external arrival and service rates (recall the nasty expressions for the transient analysis of a simple $M/M/1$ queue [477, 251, 255]). Finally, even if we solve the corresponding Markov Chain and find state dependent controls that depend on the current network state, it is impossible to know the current network state at all network resources. The needed time for the network state observations to be transferred from one network point to another is a random variable that also depends on the network state and during this time the network state has already changed. Furthermore, it is also difficult to obtain accurate estimates of the instantaneous rates. So, the network state information is always obsolete and inaccurate. Therefore attacking the stochastic problem directly would be difficult.

Also, in real network control implementations, the average rather than the instantaneous measures of the network state are used due to the following reasons: 1) wide variability of the instantaneous network state values, 2) obsolete network state information, due to transfer delay, 3) periodic implementation of the network control, 4) communication overhead in transferring the instantaneous network state information, and 5) computation overhead in calculation for an exact network optimization.

Most previous research uses simulations and heuristic algorithms to solve the dynamic problem. Another common approach is to devise an ad hoc strategy and then evaluate its performance for different values of some parameters. In this dissertation, we formulate the quasi-static problem as a nonlinear programming problem and the dynamic problem as an optimal control problem.

1.2 Methodology

1.2.1 Objective Criteria

The usual approach to distributed system design and control is the minimization of a single cost function. Most previous research on each of the above problems assume the optimization of a single objective function. In a few papers, multiple objectives are considered and then the usual approach is to combine the objectives as seen by the system administrator into a single function. Thus, it is assumed that all customers in the system are treated similarly and they cooperate for the socially optimum, such as optimizing the average customer performance. Furthermore, previous research is primarily concentrated on systems with a single class of jobs. In this dissertation (sections 4.1, 5.1), we formulate and optimize the performance of multiple cooperative classes of jobs as a *team optimization* problem.

However, in a real distributed environment there is a diversity of customer classes, each with possibly different objectives and different service and accounting requirements. The different classes of customers compete for the limited common resources of the distributed system in order to optimize their own objectives, ignoring the inconvenience that they cause to the other customer classes. For example, different telecommunication companies may share the same communication links and one of them may want to maximize the throughput of its customers, another may want to minimize its average customer delay and a third may want to minimize the blocking probability of its customers. Another example is when different users share a multiprocessor system and one group of users wants to maximize its throughput, similarly another group of users wants to maximize its own throughput, another group of users wants to minimize its average response time and finally

another group of users wants to minimize the variance of its response time. In this dissertation (sections 4.2, 5.2), we formulate and optimize the performance of different non-cooperative classes as a *Nash game*. Nash games are frequently used in optimization problems in economy [11, 361, 174, 16, 443, 445, 369, 336, 213].

Finally, it is quite common to require differentiated service among different classes by assigning different priorities to different classes, for example interactive jobs have higher priority than batch jobs. A high priority class may acquire most of the resources that it needs, while a low priority class should wait for the high priority class to complete service. Since the reason for having priorities is to give preferential treatment to the high priority jobs, it is *not* meaningful to define a *single* multi-objective function (e.g. a convex combination of the objective functions of the different priority classes) for global optimization *across all the priority classes simultaneously*. However, we can still optimize the behavior of jobs within each priority class. Therefore a different approach should be taken for performance optimization of multipriority systems. In this dissertation (sections 4.3, 5.3), we formulate and optimize the performance of different priority classes as a *Stackelberg game*. Also, it is desirable that the system administrator has more power than the system users and frequently he/she wants to impose its decisions on the users. This scenario can also be modeled in a leader-followers framework, that is a Stackelberg game. So, the administrator (leader) directs the users (followers) according to his/her objectives.

1.2.2 Decentralized Dynamic Control

In this section, we show how the load sharing, routing and congestion control decisions can be implemented in a dynamic and decentralized fashion. First, we can implement the dynamic controls according to two ways:

1) In *probabilistic decisions*, the acceptance of a job into the network, the selection of the destination computer site for processing the job and the selection of the routing path to this destination are done probabilistically. So, an action is selected with very high probability if the cost of this action is less than the cost of

all other possible actions. In chapter 6, we propose a methodology for probabilistic decisions based on stochastic learning automata.

2) In *deterministic decisions*, the acceptance of a job into the network, the selection of the destination computer site for processing the job and the selection of the routing path to this destination are done deterministically. One possible rule is to achieve the optimum controls in a weighted round-robin fashion. Another possible rule is to select an action if the cost of this action plus a threshold is less than the cost of all other possible actions.

We also want the load sharing, routing and congestion control decisions to be done in a decentralized fashion. There are two decentralized implementation options:

1) In *source decisions*, each source node accepts an externally arriving job, selects the destination computer site for processing the job and chooses the routing path to this destination.

2) In *node-by-node decisions*, each node (source node and intermediate network nodes) accepts a job (either externally arriving or transferred from another node), selects the destination computer site for processing the job and chooses the routing path to this destination.

Correspondingly to the *source* or *sender-initiated decisions*, we may have *destination* or receiver-initiated decisions, where the destination node decides about the jobs.

Although node-by-node congestion control provides complete decentralization, it may also waste a lot of network resources. If a job is rejected at an intermediate node, then all the network resources used by this job are wasted.

Also, node-by-node load sharing provides complete decentralization, but it requires more overhead, since we need to keep the information about all source-destination pairs. In source load sharing, we only need to keep information about all destinations from this source. An important advantage of source load sharing is that the source defines the destinations where its traffic may be processed avoiding (for example, for security reasons) some other nodes. In node-by-node load sharing the source node has no control over the destination selection. Also, in source

load sharing it is trivial to guarantee no looping. In node-by-node load sharing, there must exist coordination between the source node and an intermediate node to avoid looping.

Similarly, node-by-node routing provides complete decentralization, but it requires more overhead, since we need to keep the information about all paths between all sources to all destinations. In source routing, we only need to keep information about all paths from this source node to all destinations. An important advantage of source routing is that the source defines the paths that its traffic may follow avoiding (for example, for security reasons) some other nodes. In node-by-node routing the source node has no control over the path that its traffic may follow. Also, in source routing it is trivial to guarantee no looping. In node-by-node routing, there must exist coordination between the source node and an intermediate node to avoid looping.

Finally, in future high speed networks, the bottleneck will be on the computation rather on the communication delays. Therefore, it is preferable that all processing intensive functions to be transferred outside of the network to the edges. Source-based protocols satisfy this requirement.

In this dissertation, we formulate the joint load sharing, routing and congestion control problem on the path flow space, so that the decisions may be done at the source nodes.

1.2.3 Stochastic Learning Automata

In the previous section, we stated that the dynamic controls may be implemented in a probabilistic way using stochastic learning automata. These are adaptive control algorithms for highly uncertain systems. They have their origin in the area of mathematical psychology. They select probabilistically an action and then update their action probabilities according to the outcome of the selected action. If the outcome is favorable, then the probability of the selected action increases, otherwise it decreases.

In this dissertation, we propose using stochastic learning automata as decentralized dynamic decision makers at the source nodes. For each class of jobs, at

each source node there are i) an automaton for selecting the destination node for processing, and ii) an automaton for each destination, that selects the routing path to this destination or rejects new jobs.

1.3 Datagram Networks

Most previous research on the load sharing, routing and congestion control problems consider a datagram network. In datagram networks, each job is decomposed in small groups of bits called packets [166, 171, 256, 45, 478, 432]. Each packet is treated as a separate entity independently from other packets belonging to the same job. So, in datagram networks, packets independently acquire and release the required bandwidth for their transmission. In this way, the network resources are efficiently utilized, since packets can be distributed over underutilized resources.

1.4 Virtual Circuit Networks

Most existing networks (Codex, Euronet, SNA, Telenet, Transpac, Tymnet, etc.) as well as proposals for future high speed network architectures employ virtual circuit switching [45, 160, 186, 227, 432]. For each call (virtual circuit, or virtual channel, or virtual connection, or virtual route, or session, or transaction, etc.), a single path is set up from source to destination and all entities (bursts, packets, cells, etc.) that belong to this call follow this path. Virtual circuit switching provides the following advantages:

1) Flexible resource management, since packets of each connection are on a specific path and not all over the network.

2) Easier and fairer access, service, accounting and billing control.

3) No packet resequencing at the destination (due to different delays of packets that arrive at the destination through different network paths), since packets belonging to a specific virtual circuit follow a single path from source to destination (hop-by-hop resequencing due to transmission errors may still needed).

4) Less packet header overhead, since the header carries only the virtual circuit number in which the packet belongs, and not the source and destination addresses.

5) No packet looping, since packets follow an already established path.

6) Less routing update overhead, since the routing is done on a per virtual circuit basis and not on a per packet basis.

7) Easier congestion and flow control for each connection by accepting a new virtual circuit only if it will not congest the network and by controlling the packet rate and resource usage of each admitted virtual circuit.

1.5 Integrated Services Networks

Traditionally, there were separate networks (circuit/packet switched) for carrying different traffic types (voice/data). Integrated Services Networks (ISN's) have been proposed as the future network architecture that will support multimedia traffic (voice, data, video etc.) simultaneously [388, 389]. These multiple classes of traffic will share the same network resources (buffers, switches, transmission lines, etc.) for flexible and efficient resource sharing. However, each class has different and conflicting performance requirements and objectives to those of other classes. Hence new methodologies are needed for network design and control problems. These networks will operate similarly to virtual circuit networks, where all packets belonging to a specific virtual circuit follow the same route.

1.6 Outline of the Dissertation

In this dissertation, we solve the joint problem of accepting new arriving jobs in a distributed system (congestion control), of selecting the destination computer site for processing these jobs (load sharing) and of selecting the path for transferring the jobs to this destination (routing).

In *chapter 2*, we present a survey of historically important studies on each one of these three problems. The only relevance of these studies to this dissertation is that they consider either the load sharing, or the routing, or the congestion

control problem. In this dissertation, we develop a novel and unified methodology for solving the joint problem both for the quasi-static and the dynamic cases.

In *chapter 3*, we develop a generic framework on which we shall based our methodology. We introduce the analytical model that integrates the load sharing, routing and congestion control problems. We formulate the problem on the path flow space, such that the decisions are done at the source nodes. We define the key system variables and the structure of the cost functions and the state space model, for which more details are given at the application sections of following chapters.

In *chapter 4*, we develop three novel methodologies for the quasi-static problem: i) in the team optimization methodology, the classes of jobs cooperate in using the resources of the system for the socially optimum, ii) in the Nash game methodology, the classes of jobs compete among themselves and each class tries to operate optimally for its own jobs, and iii) in the Stackelberg game methodology, some classes of jobs have more power than others, for example priorities. For each methodology, we develop three alternative formulations of the joint problem, namely a Nonlinear Programming, a Nonlinear Complementarity Problem and a Variational inequality formulation. For each formulation, we state the necessary and sufficient conditions for existence and uniqueness of the solution. From the Karush-Kuhn-Tucker conditions, we also derive the abstract form of the solution, that there should be flow only on minimum length paths, to minimum length destinations. The length at each system resource is appropriately defined for each case. Then we apply these three methodologies to datagram, virtual circuit and integrated services networks. For each of these networks, we introduce cost functions for multiple classes and for priority classes of jobs. We also explicitly solve three examples: i) In the first example, two classes of jobs share two processors. The objectives are the minimize the expected job delays. We give in closed form the policies that achieve the team optimum solution and the Nash equilibrium solution and we further investigate them numerically. ii) In the second example, two preemptive priority classes of jobs share two processors. The objectives are the minimize the expected job delays. We give in closed form the policy that achieves the Stackelberg equilibrium solution and we further investigate it numerically.

iii) In the third example, two classes of jobs share two servers. Packets from the first class may be queued waiting service in front of server, while packets from the other class are dropped when there are more than a threshold jobs into the system. Then, the first class wants to minimize its expected delay, while the other class wants to minimize its blocking probability. We find the policy that achieves the Nash equilibrium solution and we further investigate it numerically.

In *chapter 5*, we develop three novel methodologies for the dynamic problem: i) the dynamic team optimization methodology, ii) the dynamic Nash game methodology, and iii) the dynamic Stackelberg game methodology. For each methodology, we develop three alternative formulations of the joint problem, namely an Optimal Control, a Nonlinear Complementarity Problem and a Variational Inequality formulation. For each formulation, we state the necessary and sufficient conditions for existence and uniqueness of the solution. From Pontryagin's maximum principle, we also derive the form of the solution, that there should be flow only on minimum length paths, to minimum length destinations, The length at each system resource is appropriately defined for each case. Then we apply these three methodologies to datagram, virtual circuit and integrated services networks. We develop new dynamic queueing models for multiple classes and priority classes of jobs, as well as linearized approximate dynamic queueing models and Wiener process models. We introduce several new cost functions and state constraints. We explicitly solve an example for virtual circuit networks. We consider a virtual circuit network with Poisson arrivals of virtual circuits and packets, and exponential service requirements. We want to minimize the expected cost of servicing or rejecting virtual circuits, minimize the expected cost of packet delay and maximize the expected profit from packet throughput. We find the dynamic team optimality conditions and we propose a state dependent routing and congestion control algorithm. We investigate and compare (via simulation) this state dependent routing algorithm to the optimal quasi-static algorithm. We find that the more often that we update the state dependent algorithm and the more recent information that we use

the better. When the updating period is not much larger than the mean interarrival time of virtual circuits, then this state dependent algorithm achieves smaller average packet delay than the optimal quasi-static algorithm.

In *chapter 6*, we introduce another novel methodology for decentralized dynamic load sharing, routing and congestion control. We propose stochastic learning automata at the source nodes of the system for admitting or rejecting jobs, for selecting the destination node for job processing, and for selecting the routing path to the destination node. These decisions will be done probabilistically by learning automata algorithms that will update their action probabilities according to measurements of the path and source-destination lengths. The path and source-destination lengths are those derived in the dynamic optimality conditions of chapter 4. We also introduce novel classes of stochastic learning automata:
i) state dependent learning automata, whose adaptation rates are functions of the system state, ii) two-step learning automata, that use larger adaptation rates when the selected action repeatedly gives good performance, iii) multiple response learning automata, that use different adaptation rates for different system response (not just the favorable/unfavorable response of previous learning automata). We prove that these learning automata are feasible at each step, non-absorbing, strictly distance diminishing, ergodic and expedient. We apply this methodology to datagram, virtual circuits and integrated services networks. We give an example, where we make virtual circuit routing decisions using stochastic learning automata algorithms. We show (via simulation) that by suitable tuning the adaptation rates of the algorithms, the learning automata achieve smaller average packet delay. We also show that a path length proposed in chapter 5, is superior to a shortest-queue-type routing, usually used in real networks.

Finally, in *chapter 7*, we conclude on our proposed unified game-theoretic methodology for the multi-objective joint load sharing, routing and congestion control problem in distributed systems and suggest directions for further research.

1.7 Contributions of the Dissertation

The major contribution of this dissertation is that it develops a novel approach for the decentralized joint load sharing, routing and congestion control problem, based on game theory and stochastic learning automata theory.

More specifically, our contributions are :

1) We solve *joint load sharing, routing and congestion control problem.* Solving each problem separately and then combining the derived policies results in only suboptimal solution, because there is strong dependency among these policies. Therefore, we consider the joint problem and optimize the system performance with respect to load sharing, routing and congestion control decisions simultaneously.

2) We introduce a novel *queueing model* for the joint problem on the path flow space. We also introduce the cost functions for each class of jobs and present the generic form of the state space evolution and state constraints. Having a mathematical model always help in formulating and solving a problem. We use this generic form of the model in the optimization formulations. Subsequently, when we specialize to specific network types, we develop more detailed cost functions, state space models and state constraints.

3) We develop three novel *methodologies* for both the quasi-static and the dynamic joint problem. Previous studies on either of these problems assume the optimization of a global objective function. However, different classes of jobs may have different objectives, sometimes even conflicting. Furthermore, in almost all distributed systems, not all jobs are treated the same way. Some jobs (e.g., interactive) have higher priority than others (e.g. batch) jobs. Therefore we introduce a *team optimization methodology* for classes of jobs that cooperate for optimizing a combination of their objective functions. We introduce a *Nash game methodology*, for classes of jobs that compete among themselves for the system resources, and each class tries to optimize its own objective. Finally, we introduce a *Stackelberg game methodology*, for classes of jobs, with some classes having more power than others.

4) We present three novel *formulations* for each methodology. We formulate the joint problem as a *Nonlinear Programming* (quasi-static case), as an *Optimal Control or Dynamic Programming* (dynamic case), as a *Nonlinear Complementarity* and as a *Variational Inequality* Problem. We derive the conditions for *existence, uniqueness and optimality* of the solution. Having three alternative formulations for the joint problem provides larger flexibility in solving it. In order to solve a specific instant of the joint problem, we can use algorithms used to solve Nonlinear Programming, Optimal Control, Nonlinear Complementarity or Variational Inequalities Problems. Also, for specific network type, the cost functions may satisfy the conditions of a formulation and not those of another formulation.

5) We derive the *optimality conditions* for each methodology. Traffic flow should be non-negative only for source-destination pairs of minimum length and on paths of minimum lengths, where the lengths are appropriately defined in each case. It is not only important to solve the joint problem, but also gain insight on the structure of its solution. Then we may also use more suitable algorithms in solving it, or even be satisfied with a suboptimal solution that is close enough to the optimal one. We can also devise heuristic algorithms that will be based on the optimality conditions.

6) We *apply* the proposed methodologies to *Datagram, Virtual Circuit and Integrated Services Networks.* We specialize our cost functions to specific cost functions suitable for each network type.

7) We explicitly solve several *examples* and derive the team optimum, the Nash equilibrium and the Stackelberg equilibrium solution. Investigation of specific examples provides us with insight on the structure of the strategies for each class, as well as their performance. For example, in a two-class two-processor load sharing problem, we have a multiplicity of team optimum solutions, there is also an optimum solution that is independent of the arrival rate of one class. In another load sharing example of two preemptive priority classes that share two processors, we found a conservation law, and that the low priority class decisions are independent of the arrival rates in some cases.

8) We develop novel *dynamic queueing models*. We develop state space models for the expected number of packets in a resource as well in the queue for multiple classes and priority classes. We also develop linearized state space models. These dynamic models can be used in the proposed methodologies for the specific network type under consideration.

9) We develop novel *performance measures* for specific network types. We use one class of these measures to estimate the length (load) of a resource as a convex combination of its current length and its future expected length. We show (via simulation) that this length is superior to some others used in real networks.

10) We propose new *state dependent virtual circuit routing algorithms* and show (via simulation) to be superior to the optimal quasi-static ones. We also show that it is very important to use the most recent information about the network state.

11) We propose *decentralized asynchronous dynamic load sharing, routing and congestion control decisions using stochastic learning automata*. In an uncertain environment, our knowledge about the system state is inaccurate and obsolete. So, instead of taking a definite control decision, we do not overreact but move slowly and steadily towards the best decision. A learning automaton has also more flexibility than a deterministic decision (which is a special case of the learning automaton), since by tuning its parameters we can control its adaptation rate and its stability.

12) We introduce three novel classes of stochastic learning automata algorithms: i) *state-dependent learning automata*, whose adaptation rates are functions of the system state and therefore may follow it more closely. ii) *two-step learning automata*, that use larger adaptation rates when the selected action repeatedly gives good performance and therefore we have more confidence that it will achieve good performance at the next step, ii) *multiple response learning automata*, that use different adaptation rates for different system responses and therefore reward more a very good response, reward less a fair response, penalize a bad response and penalize heavily a very bad response.

13) We *apply the learning automata to Datagram, Virtual Circuit and Integrated Services Networks*. At each source node, a learning automaton selects the best

destination node for processing the job and another one rejects the job or routes it through a path to its destination.

14) We implement and compare (via simulation) *virtual circuit routing using several learning automata algorithms* and different performance measures. We show that all proposed learning automata algorithms perform well and frequent updating with the more recent information provides better performance. For virtual circuit routing, its is also important to include in the link length function both the number of packets and virtual circuits.

Chapter 2

Previous Studies

In this chapter, we present a survey of historically important studies on each one of these three problems. The only relevance of these studies to this dissertation is that they consider either the load sharing, or the routing, or the congestion control problem. In this dissertation, we develop a novel and unified methodology for solving the joint problem both for the quasi-static and the dynamic cases.

We classify previous studies in the load sharing or the routing section according to the intention of their authors to give the corresponding flavor to their work. Note that if the communication network is ignored in the model, then the load sharing and the routing problems are mathematically equivalent. In the load sharing problem, the jobs share some processors for processing, while in the routing problem, the jobs share some link for transmission.

2.1 Load Sharing (or Load Balancing)

In this section, we present previous studies on the quasi-static and dynamic load sharing problem. We do not consider the static allocation problem, where the set of jobs is given and it is asked to allocated the jobs onto the set of processors.

1) *Quasi-static load sharing.*

Here, the load sharing decisions depend on the arrival and service distributions. The usual approach is to formulate the problem as a nonlinear programming problem to minimize the expected job response time with respect to the link flows.

Buzen & Chen [81] and Ni & Hwang [352, 353] study the problem for a multiprocessor, where there is no communication network. Gao, Liu & Railey [181] minimize the expected job response time and the root mean square difference of the load among the sites.

de Souza e Silva & Gerla [124] model the system as a product form queueing network with fixed closed chain routing. They minimize a measure of the average delay with respect to the open chains flows. Tantawi & Towsley [479] model each computer site as well as the network as single nodes. They minimize the expected job response time. Lee & Towsley [293, 294] give priority to either local or remote jobs. They use matrix geometric techniques and integer programming to find the thresholds. Lee, Towsley & Choi [295] formulate the problem with reliability constraints.

Kurose & Simha [279, 277, 278] use first and second derivative algorithms, as well as [450] stochastic approximation algorithms for load sharing. Lin, Yee & Raghavendra [303] consider the joint load sharing and routing problem and minimize a linear combination of the expected job response time at the computer sites and the expected message delay in the network.

In contrast to these studies, that consider a single global objective, Economides & Silvester [138] formulate the multi-objective problem for two preemptive priority classes of jobs as a Stackelberg game. Each class of jobs wants to minimize its own expected response time. They derive the algorithm that achieves the Stackelberg equilibrium. In this dissertation, we further extend our work to the joint quasi-static load sharing, routing and congestion control problem in distributed systems. We formulate and solve it as a static team optimization, as a static Nash game and as a static Stackelberg game problem.

2) *Dynamic load sharing*

Here, the load sharing decisions depend on the current network state. Most studies on dynamic load sharing first propose a policy, and then evaluate its performance using queueing models or simulation.

Glorioso & Colon [194] consider a process oriented system. A process requesting service from another process is routed through the network until it finds it or fails. They use learning automata to update the routing probabilities. Glorioso & Osorio [197] also use learning automata for updating the probabilities for a multiprocessor system.

Chow & Kohler [103] analyze the performance of homogeneous two-processor systems under several algorithms. They also [102] analyze heterogeneous systems. Bryan & Finkel [78] propose a bidding algorithm that sends jobs from a high loaded site to a light loaded one. Livny & Melman [308] show that when no load sharing exists in a broadcast network, then there is a large probability that at least one node is idle while jobs are queued at some other node.

Stankovic [460, 462] applies Bayesian decision theory to the problem and [461] investigates three algorithm via simulation. He also [463] uses a stochastic learning automaton and a bidding algorithm. Mirchandaney & Stankovic [329] show via simulation that learning automata perform better than Bayesian decision theory and double bias load sharing in moderate loads and in heavy loads that cause significant delays in the subnet.

Wang & Morris [506] compare several sender and receiver- initiated algorithms with the same level of information available. They conclude that receiver-initiated algorithms have the potential to outperform sender-initiated algorithms.

Eager, Lazowska & Zahorjan [132, 134] conclude that sender-initiated algorithms are preferable at light to moderate loads, while receiver-initiated algorithms are preferable at high loads. They also [133] suggest that limited performance improvement can be achieved if processes migrate more than once.

Yum & Lin [528] investigate four algorithms via simulation. Ni, Xu & Gendreau [354] propose a bidding algorithm to maximize processor utilization and minimize

communication overhead. Chou & Abraham [99, 100] use a linear mean-square estimator to predict the load at the processors.

Kurose, Singh & Chipalkatti [280] and Kurose & Chipalkatti [276] consider soft real-time systems in which jobs arriving to the system must complete execution within a specified time constraint, otherwise they are lost. The objective is to minimize the percentage of jobs lost.

Yu, Balsamo & Lee [520, 521, 522] evaluate different algorithms to achieve a compromise between balancing the load and reducing communication overhead. They propose an algorithm which takes into account load sharing decisions. Ciciani, Dias & Yu [108] compare several algorithms to find that the decisions should be based on the effect on all transactions in the system, rather than on incoming transactions alone.

Lu & Carey [309] minimize the load unbalance. Hsu & Liu [226] investigate three algorithms based on the estimated unfinished work or the queue length. Hac & Johnson [207, 206] suggest that the parameters of the algorithms should be properly tuned. Hac & Jin [205, 206] consider sender-initiated algorithms with thresholds either on the queue length or the amount of CPU time of active processes.

Chang & Livny [94] consider jobs with deadlines. They conclude that sender-initiated algorithms outperform receiver-initiated algorithms when the load is light. Mutka & Livny [340] investigate via simulation three algorithms. They achieve to protect light users when a few heavy users try to monopolize all free resources. Krueger & Livny [271] examine the means and standard deviations of wait times and wait ratios, and the correlations between wait ratio and service demand as well as between wait time and service demand. They conclude that different algorithms optimize different objectives.

Ferrari & Zhou [151, 148] and Ferrari [151, 150] propose a load index for load sharing decisions. Zhou & Ferrari [532] conclude that algorithms that use periodic load information exchanges and algorithms that acquire such information on demand provide comparable performance. Zhou [531] uses traces from production systems in a simulation study and compare seven load sharing algorithms. He

concludes that algorithms with periodic or nonperiodic load information exchange perform similarly. Source-initiative algorithms were found to perform better than server-initiative ones.

Leland & Ott [300, 299] analyze traces and conclude that the job service requirement is not exponentially distributed. They also conclude that job migration can improve the response ratio of jobs with long service requirement. Iqbal, Saltz & Bokhari [230] compare static and dynamic algorithms. Ferguson, Yemini & Nikolaou [146] describe a microeconomic approach to the load sharing problem based on auctions. Processors use auction models to allocate their CPU to jobs that compete by issuing bids. Thomasian [483] proposes several algorithms based on queueing network models to minimize the expected response time or optimize the job mix.

Liu & Silvester [307] propose a receiver-initiated algorithm to balance the workload. They use thresholds for transferring jobs from heavily loaded sites to lightly loaded ones. They also [306] propose a two-mode algorithm that works as receiver-initiated in heavy loads and sender-initiated in light loads. Pulidas, Towsley & Stankovic [390] consider a sender-initiated algorithm, where an external arriving job, that finds more than a threshold jobs in its local site, is transferred to another site. The objective is to minimize the expected response time.

Bonomi [60, 61, 62] proposes the use of queue lengths instead of processor utilization for load sharing decisions. Bonomi & Kumar [63, 64] show that minimizing the expected job response time is equivalent to balancing the server idle times in a weighted least squares sense. They use processor utilization to allocate jobs on processors. Kumar & Bonomi [272] use a stochastic approximation algorithm to balance the load for a two processor system. Kumar [273] uses stochastic approximation techniques based on the processor idle time. Bonomi [62] analyzes the join-the- shortest-queue algorithm when the service discipline is processor sharing.

Cassandras & Lee [86] minimize the loss probability in systems where a job is lost if its waiting time exceeds its deadline. Weinrib & Shenker [510, 441] consider a finite number of fast servers and an infinite number of slow servers and show via

simulation that for heavy load it is better never to queue at a fast server, but use a slow server.

Towsley & Mirchandaney [487] investigate the effect of the communication delay in the performance. They conclude that algorithms that use only local state information achieve comparable performance to those that use global state information, when the communication delay is high. Mirchandaney, Towsley & Stankovic [331, 330] analyze three algorithms using matrix-geometric solution techniques and recommend load sharing for highly loaded systems.

Casavant & Kuhl [85] investigate the role of state information on the performance achieved by load sharing. They conclude that it is better to use a small amount than to gather a large amount of information, since this will cause a huge overhead. They also provide a taxonomy of load sharing algorithms [86]. Banawan & Zahorjan [21] use semi-Markov decision processes to show that no optimal policy exists that uses only the instantaneous queue length independent of system utilization.

Kleinrock & Korfhage [257] use a Brownian-motion-with-drift model to analyze the transient system behavior. Efe & Groselj [142] conclude that under heavy loads, a load sharing policy may perform worse than no load sharing case, due to the overhead that it produces. Leff, Yu & Lee [296] use regression analysis to estimate the response time. Boel & Schuppen [57] discuss several dynamic algorithms. Avritzer, Gerla, Ribeiro, Carlyle & Karplus [18] conclude that load sharing is effective even in the presence of high overheads, and careful tuning of the algorithms parameters should be done.

In contrast to these studies, that consider a single global objective or ad hoc techniques, in this dissertation, we further extend our work to the joint dynamic load sharing, routing and congestion control problem in distributed systems. We formulate and solve it as a dynamic team optimization, as a dynamic Nash game and as a dynamic Stackelberg game problem.

2.2 Routing

In this section, we present previous studies on the quasi-static and dynamic routing problem.

2.2.1 Datagram Networks

1) *Quasi-static datagram routing*

Here, the routing decisions depend on the arrival and service distributions. The usual approach is to formulate the problem as a nonlinear programming problem to minimize the expected job delay with respect to the link or path flows.

Wardrop [507] calculates the traffic patterns according to the user optimization criterion and the system optimization criterion, for a network consisting of two nodes connected by n independent paths Dafermos & Sparrow [120] develop the optimality conditions for system and user optimum. Fratta, Gerla & Kleinrock [172] use a steepest descent method, called flow deviation, to find the flow assignment. Cantor & Gerla [83] propose a method, based on decomposition techniques, for finding the flow assignment. Schwartz & Cheung [433] propose the gradient projection algorithm to find the flow assignment, for networks with a relative small number of commodities, since it needs less computation time than the flow deviation method. Stern [470] uses relaxation methods to find the flow assignment.

Gallager [179, 178] proposes a distributed loop-free algorithm for finding the flow assignment. Ephremides [144] extends Gallager's algorithm to mixed network composed by ground and satellite links. Bertsekas, Gafni & Vastola [36] present computational results using gradient projection algorithms for finding the flow assignment. Bertsekas [41, 40, 39] propose a gradient projection algorithm. Bertsekas, Gafni & Gallager [43, 44] also propose second derivative algorithms, that may be viewed as approximations to a constrained version of Newton's method. Bertsekas, Hosein & Tseng [45] study the convergence of a dual Gauss-Seidel type relaxation method for network flow problems. Tsai, Huang, Antonio and Tsai [491] present iterative algorithms for box-constrained minimization problems.

Kamoun & Kleinrock [241] demonstrate the efficiency of hierarchical routing for large networks. Kobayashi & Gerla [264] consider the single objective multiple class routing problem in closed queueing networks. Each closed chain corresponds to a different class of customers. They minimize the average delay, which is not convex, for closed chains routing, and therefore local minima exist. Bovopoulos & Lazar [73, 72, 74, 68] investigate the routing and flow control problem for queueing networks with nonzero acknowledgment delay. They maximize the throughput such that the end-to-end expected delay is bounded.

Chen & Meditch [97] propose two interactive algorithms, where the first algorithm generates a flow assignment, and the second algorithm generates a loop-free flow assignment. Saksena [416] analyzes routing with constraints on the number of hops for each path, on the number of paths, and on the end-to-end average delay. Chang & Wu [94] propose an algorithm that at each iteration updates both the estimate of external traffic input and the flow assignment. Pavlidou [377] uses the variable reduction method for finding the flow assignment.

Economides & Silvester [141] formulate the datagram routing in networks with variable quality links. Some links have high error rate and therefore packets fail and must be retransmitted. They formulate the minimization of the expected packet delay as a nonlinear programming problem and they solve it. They also introduce learning automata as routers. According to the objective, they use as feedback information the average packet delay, or the success/failure of a transmitted packet.

In contrast to these studies, that consider a single global objective, Economides & Silvester [139, 140] solve the routing problem as a team optimization problem, when the classes of packets cooperate and as a Nash Game, when the classes of packets compete among themselves. In this dissertation, we further extend our work to the joint quasi-static datagram load sharing, routing and congestion control problem in distributed systems. We formulate and solve it as a static team optimization, as a static Nash game and as a static Stackelberg game problem.

2) *Dynamic datagram routing*

Here, the routing decisions depend on the current network state.

Frank & Taylor [169, 170] describe the network state using discrete state space equations and formulate the routing problem as a linear program.

Boehm & Mobley [56] recommend the biadaptive (for large networks and for small networks with security constraints), a dynamic-programming-based (for small networks without constraints) and a graph-theory-based algorithm (for large networks). Glorioso, Grueneich & Dunn [195] and Glorioso, Grueneich & McElroy [196] propose learning automata for self organization and dynamic routing.

Agnew [2, 3] proposes an adaptive algorithm extracted from the quasi-static algorithm. He also [4] introduces a nonlinear dynamic model for queueing systems. He uses this model to study the stability and transient behavior of a congestion-prone system and to analyze policies for congestion control.

Winston [515] and Weber [508] show that join-the-shortest-queue policy maximizes the discounted number of customers to complete service in any time t. Flatto & McKean [161] also analyze the join-the-shortest-queue policy. Foschini [168] and Foschini and Salz [167] investigate the dynamic routing problem in heavy traffic using diffusion processes.

Segall [437] formulates the problem as a linear optimal control problem with linear state and control variable inequality constraints. He also suggests methods for incremental delay estimation. Moss & Segall [335] present the conceptual form of an algorithm to find a feedback solution when the inputs are constant over time. Meditch [327] uses a state space approach and investigate controllability and observability in the presence of delayed information patterns. He proposes feedback control policies.

Chu & Shen [106, 107] present a policy for hierarchical routing and flow control. When all channels along the primary route do not exceed a threshold, then the primary route is used, otherwise alternative routes are used. Rudin [409] presents a taxonomy of routing strategies and then he proposes the delta routing and compares it to other algorithms. The delta routing combines a centralized and a decentralized policy. The central coordinator prepares the overall strategy

and sends it to the nodes. Then the nodes use this information together with their local state to decide the routing. Yum & Schwartz [525, 526, 527, 436, 436] superimpose local adaptivity on top of a static flow assignment. They propose *the join-biased-queue rule that is the join-shortest-queue with a threshold term* on the queue lengths. Yum [523, 527] proposes a deterministic routing sequence to achieve the flow assignment. He also proposes a semidynamic version of it for a time varying traffic.

Bertsekas [41, 38, 42] investigates the stability, convergence and speed of convergence of routing algorithms. Schwartz & Stern [435] survey studies on the routing problem up to 1980. McQuillan, Richer & Rosen [326] describe the algorithm for the ARPANET. They use the link delays as link length.

Ephremides, Varaiya & Walrand [145] show that the join-the-shortest-queue policy minimizes the expected total time for the completion of service of all jobs which arrive during a time interval, when the queue lengths are observed. When the queue lengths are not observed, it is best to alternate between queues. Olsder & Suri [362] use dynamic programming to derive optimality conditions for a dynamic routing problem, where one of the two servers may fail. Hahne [208] computes dynamic routing strategies for a system of two unreliable finite storage servers.

Boorstyn & Livne [65] propose a two-level adaptive routing, where the first level is the quasi-static solution of the problem, while the second level is dynamic. Chu, Boorstyn & Kershenbaum [105] investigate by simulation several algorithms. Agrawala, Tripathi & Ricart [6] introduce the virtual waiting time technique which does not depend on any arrival and service time distribution. Agrawala & Tripathi [5] show that the deterministic techniques based on the best stochastic rules are not necessarily optimal.

Srikanta Kumar [454] use learning automata at every network node to route a message over one of its outgoing links to its destination according to a measure of the expected delay on this outgoing link and from the next node to the destination. Chrystall & Mars [104] use learning automata to route messages over the outgoing links at every network node. They use the message delay in order to update the probabilities of selecting each outgoing link. Simulation results show that at light

traffic , random, delta and learning routing give comparable average message delay, while at high traffic, learning routing performs better. Similarly, Mars, Narendra & Chrystal [370] use learning automata to route packets over the outgoing links at every network node. Simulation studies of packet routing in datagram networks show that learning routing performs better than fixed rule routing especially at high traffic.

Chou, Bragg & Nilsson [101] conclude that a deterministic strategy is better for balanced traffic, while an adaptive strategy is better for unbalanced and chaotic conditions. They also propose a second-order metric as queue length. Sarachik & Ozguner [423] consider the problem of clearing congested single destination networks for constant inputs. They formulate an optimal control problem and propose an algorithm. Similarly, Sarachik [422, 424] considers the multi-destination problem. He also [421] suggests a routing strategy based on thresholds.

Hajek & Ogier [210, 211] extend Segall's work. They introduce a flow relaxation concept to transform the optimal control problem into an initial flow optimization problem. They first propose three algorithms: i) a gradient descent with bending algorithm, ii) a series of max-flow problems, and iii) a search for successive bottlenecks. Sasaki & Hajek [426] present iterative methods for finding state-dependent strategies. They use flow relaxation to transform the problem and then a projected descent direction method to solve it. Ogier [358] presents an algorithm to compute the flow in a network with piecewise-constant capacities.

Filipiak [158, 156, 159, 155, 153] models the routing and congestion control problem using nonlinear state space models and formulates the problem as an optimal control problem. Then he shows that the optimal flow pattern is given by the steady-state solution of the costate equations.

Muralidhar & Sundareshan [337, 338] present a two-level scheme for the combined routing and flow control problem. At the lower level, the flow assignment is calculated, while at the higher level a set of parameters are calculated. Casalino, Davoli, Minciardi & Zoppoli [84] and Aicardi, Davoli & Minciardi [7] examine the information structure of the dynamic routing problem by considering every node as an agent in a team control problem setting. They define a nested information

structure with a common past and suggest dynamic programming decomposition. Ramamoorthy & Tsai [393] and Tsai, Ramamoorthy, Tsai & Nishiguchi [493] propose a hierarchical routing algorithm and show the tradeoff between performance improvement and communication overhead.

Mason [321] describes several applications of learning automata to networks. He also [322] applies learning automata to datagram routing. Nedzelnitsky [349] introduces a learning automaton model for datagram routing. When a link is used, its penalty probability increases according to its routing probability. Queue delay (total bit length in queue) is used to update the link routing probabilities. Simulation show that learning routing performs better than fixed rule and random routing, especially in heavy traffic. Nedzelnitsky & Narendra [350] also propose using stochastic learning automata for datagram routing.

Tsitsiklis [496] formulates an infinite horizon dynamic routing problem to two servers, that may become unoperational. He shows that optimal policies are switching curves, each corresponding to a particular state of the servers. Maglaris [319] considers a two-level routing policy, where at the high level the quasi-static flows have been found, while at the lower level adaptive decisions are made locally. He formulates the local problem as a Markov decision process and shows that a deterministic threshold policy minimizes the delay. Zhou & Maglaris [530] extend Yum's two-level adaptive routing algorithm. The local decisions are based on thresholds and priorities.

Kumar & Walrand [275] show that if there is a socially optimal policy for a system with no arrivals, which can be implemented for each job following a policy in such a way that no job ever utilizes a previously declined route, then such a policy is an individually optimal policy for each job. Moreover this policy continues to be individually optimal even if the system has an arbitrary arrival process, but with past arrivals independent of future route-traversal times. Halfin [212] calculates bounds for the probability distribution of the number of customers for the join-the-shortest-queue policy.

Whitt [511] shows that there are service time distributions for which the join-the-shortest-queue is not optimal. He also shows that if, in addition, the elapsed

service times of customers in service are known. the long-run average delay is not always minimized by customers joining the queue that minimizes their individual expected delays. Rosberg [401] presents a policy based on a deterministic routing sequence. Rosberg & Makowski [400, 402] show that every routing policy which minimizes the long-run expected holding cost is contained in the set of optimal policies which minimize the total expected cost for a system with fixed initial population and no new arrivals.

Stassinopoulos & Protonotarios [468, 469] suggest methods for congestion clearing in minimum time for single destination networks. Stassinopoulos [465] solves the minimum evacuation time problem for multiple destination networks. Stassinopoulos & Kukutos [467] present an optimal control approach and an algorithm for double ring networks.

Knessl, Matkowsky, Schuss & Tier [260, 261] consider the join-the-shortest-queue policy and find the steady-state probability distribution of the number of customers. Blanc [53] uses power series expansions to calculate state probabilities for the join-the-shortest-queue policy in multiserver systems. Chang & Chang [93] study routing of batch Poisson arrival to multiple synchronous servers. Beutler & Teneketzis [47, 48] use dynamic programming for routing under imperfect information. They show that certain inequalities involving stochastic ordering of information measures can be propagated inductively from one epoch to the next. They show that the optimal policy is of threshold type.

Cassandras, Abidi & Towsley [86, 89] use perturbation analysis to estimate on-line the marginal packet delays through links with respect to link flows. Zinky, Vichniac & Khanna [533, 250] investigate the revised routing metric in the ARPANET and MILNET. Under light loads, the metric behaves as a delay-based metric. Under heavy loads, the metric is based on available capacity. Daneshrad & Morgera [121] show via simulation that learning automata routing performs better than shortest path and proportional routing. Glazer & Tropper [193] propose as metric a combination of link and buffer utilization in a routing algorithm that controls the congestion. Krishnan [268, 269] uses Markov decision processes and propose a join-the-shortest-queue type scheme.

Economides & Silvester [141] introduce learning automata for datagram routing in networks with variable quality links. According to the objective, they use as feedback information the average packet delay, or the success/failure of a transmitted packet.

In contrast to these studies, that consider the system (or social) optimum, and the user (or individual) optimum, in this dissertation, we consider the multi-objective joint dynamic datagram load sharing, routing and congestion control problem, where the classes of jobs play a game. The system (correspondingly user) optimization approach is a special case of this game-theoretic approach- just consider all classes as one (correspondingly, each class as composed by a single job). We formulate and solve the joint problem as a dynamic team optimization, as a dynamic Nash game and as a dynamic Stackelberg game problem.

2.2.2 Virtual Circuit Networks

In virtual circuit networks, we have extra difficulties due to the complex system dynamics. The load sharing, routing and congestion control decisions are made at virtual circuit arrival instants. Subsequently, no control is exercised over the packet routing process, i.e. the network state is affected for longer periods than in datagram networks. The virtual circuit network dynamics occur at two different time scales. The fundamental point is that although the control is exerted at the slower time scale, where the virtual circuit establishment / termination process occurs, the network performance is measured at the faster time scale, where the packet transport process occurs. Previous studies assume either independence of the virtual circuit establishment / termination process and packet transport process or limiting process rates. In this dissertation, we explicitly consider both levels of traffic processes and their interaction.

Depending on the assumptions of how fast the network dynamics change, three main approaches have been investigated for virtual circuit routing:

1) In *Static virtual circuit routing* the number of virtual circuits is given (i.e. no virtual circuit arrivals) and the packet arrival rate per virtual circuit is also given. There exist two main formulations for the static virtual circuit routing problem:

1.1) In the *continuous nonlinear programming* formulation, the decision variables are the flows on the links.

Gerla & Nilsson [188], Gerla,Chan & Boisson De Marca [187], Lam & Lien [285, 286], Kobayashi & Gerla [264], De Souza & Gerla [124] model a virtual circuit network as a closed queueing network where each closed chain corresponds to a flow controlled virtual circuit. Packet arrivals belonging to a specific virtual circuit that find its virtual circuit window full are lost. They use *flow deviation* - type algorithms and report numerical results.

1.2) In the *0-1 nonlinear programming* formulation the decision variables are the assignment or not of a given virtual circuit to a path.

Courtois & Semal [114] modify the *flow deviation* method for the non bifurcated flow case. Gavish & Hantler [182], Gavish & Neuman [183], Narasimhan, Pirkul & De [341], Lin & Yee [304], use *Lagrangian relaxation & subgradient optimization* techniques and report numerical results.

2) In *Quasi-static virtual circuit routing*, it is assumed that the externally arriving traffic changes very slowly over time and individual offered traffic sample functions do not exhibit frequently large and persistent deviations from their averages [39, 497, 45]. Therefore the virtual circuit routing decisions may depend only on the input and link flows.

For the case of "many small users" [176, 177, 497, 45], where there is a very large number of virtual circuits between each source-destination pair and each virtual circuit has a very small packet arrival rate, Gafni & Bertsekas [177] show that the routing updating period should be much smaller than the average virtual circuit duration. In this case, Bertsekas [39], Tsitsiklis & Bertsekas [497], Tsai, Tsitsiklis & Bertsekas [492], Tsai [489, 490] formulate the virtual circuit routing problem as a *continuous nonlinear programming* problem with decision variables the flows on the links or the paths. They use *gradient projection* - type algorithms to precompute

the optimum flows. Then they assign each new arriving virtual circuit on the paths to achieve these optimum flows. Tsai [489] shows that deterministic assignment of the new arriving virtual circuits is better than random.

Humblet, Soloway & Sleinka [228] define as link length in the Codex network [228, 45], the difference of the link cost function when a new virtual circuit is routed through this link and when not. Then a new virtual circuit is routed along the minimum length path. Gopal, Kadaba & Wieber [201] define as link length the first derivative of the average number on this link with respect to the flow through this link. The link length is updated only when the link utilization crosses a threshold. Then a new virtual circuit is also routed along the minimum length path. Jaffe & Segall [234] minimize the average delay over a range of arrival rates with respect to link utilization thresholds and link lengths. Gersht [189] assumes that the average virtual circuit duration is much larger than the average packet delay in the network for the Telenet network. At every node, a new arriving virtual circuit is routed through the outgoing link with the minimum number of virtual circuits.

In contrast to these studies, in this dissertation, we further extend our work to the joint quasi-static virtual circuit load sharing, routing and congestion control problem in distributed systems. We formulate and solve it as a static team optimization, as a static Nash game and as a static Stackelberg game problem.

3) In *Dynamic* or *Adaptive virtual circuit routing*, it is assumed that the network state is continuously changing due to real time traffic fluctuations. Therefore, the routing decisions are taken for each virtual circuit independently and depend on the current network state, for example the current network topology, the current number of packets & virtual circuits, the current virtual circuit & packet arrival rates, the current service requirements, the current link error rates etc.

Two main formulations exist for the dynamic virtual circuit routing problem:

3.1) In the *stochastic learning automata* formulation, the fact that our information about the network state is inaccurate is incorporated into the routing decisions.

Economides, Ioannou & Silvester [136] introduce *state-dependent multiple reward / penalty* stochastic learning automata updating schemes and use them for virtual circuit routing. The idea here is not to overreact by completely trusting the information we have about the network state. Instead of using a definitive decision as to where to send a new arriving virtual circuit, we vary the routing probabilities strongly favoring the minimum length path, i.e. we have a "probabilistic selection of the minimum length path". Note that, the routing along the minimum length path (with probability 1) is a special case of the stochastic learning automata routing. They define as link length the unfinished work due to packets and virtual circuits currently on this link (for user optimization) or the increase in the current number of packets due to the addition of a new virtual circuit on this link (for system optimization). In this dissertation, we further extend our work to the joint dynamic virtual circuit load sharing, routing and congestion control problem in distributed systems. We formulate and solve it as a dynamic team optimization, as a dynamic Nash game and as a dynamic Stackelberg game problem. Then we introduce learning automata to find the optimum decisions in a decentralized and asynchronous way.

3.2) In the *optimal control* formulation, the network state dynamic evolution is described by a state space model of difference or differential equations.

Gafni & Bertsekas [176] consider a simple stochastic problem that turns out to be an NP-complete problem. Then they transform it to the corresponding deterministic problem that is still a hard problem. Finally, they suggest a discrete time heuristic - routing along the minimum first derivative length path.

Tipper & Sundareshan [485] consider a single source-destination virtual circuit routing problem. They develop a nonlinear dynamic queueing model for the average number of packets of a virtual circuit and of interfering traffic at each link along the paths from its source to its destination. Pontryagin's maximum principle provides the necessary conditions for optimal virtual circuit routing. They propose routing of this virtual circuit along the minimum length path, where the length of a link is a quadratic function of the average number of packets on it.

Economides & Silvester [136] consider the routing and congestion control problem for window flow controlled single source-destination virtual circuit networks with error prone links. They develop a queueing model for each error controlled and window flow controlled link. They express the link capacity as a function of the link error rate, link transmission rate, link propagation delay, ACK delay, time-out delay and other flow control parameters. They introduce dynamic non-linear queueing models that describe the dynamic interaction of the virtual circuit and packet processes. Then they define a multi-objective function of the average number of virtual circuits and packets on every link, the virtual circuit rejection flow and the packet throughput on every link. Pontryagin's maximum principle provides the necessary optimality conditions that are also sufficient for this convex virtual circuit routing and congestion control problem. A new virtual circuit is routed along the minimum length path. The length of a link is a quadratic function of the average number of packets on it and a linear function of the average number of packets per virtual circuit, of the cost per virtual circuit and of the packet throughput profit.

In contrast to these studies, that consider a single global objective, in this dissertation, we also consider the non-cooperative multi-objective joint dynamic virtual circuit load sharing, routing and congestion control problem in distributed systems. We formulate and solve it as a dynamic team optimization, as a dynamic Nash game and as a dynamic Stackelberg game problem.

2.2.3 Integrated Services Networks

1) *Quasi-static multi-class routing*

Here, the routing decisions depend on the arrival and service distributions.

Maglaris, Boorstyn, Panwar and Spirtos [317, 318] consider routing in burst-switched networks. They linearize the function of the frozen voice traffic and formulate a linear programming problem with constraints for the data traffic. Yum & Schwartz [524] investigate the routing overhead in integrated circuit/packet-switched networks. Wei, Youwei and Zheng [509] suggest an algorithm that chooses the shortest path weighted by available capacity factors for circuit-switched traffic

and chooses the path which has the minimum product of queueing length and available capacity factors for packet-switched traffic.

Okazaki & Schwartz [360] describe routing problems in networks with a movable boundary scheme. They propose an algorithm to minimize the weighted sum of blocking probability and average delay. They conclude that voice and data should be routed on the same paths. Ibrahim & Elhakeem [229] consider movable-boundary-type networks and suggest a combined routing/capacity assignment algorithm. Cassandras, Kallmes & Towsley [90] consider real-time traffic that has deadline constraints. Maxemchuk & Zarki [324] survey routing and flow control techniques developed for wide area, local area and metropolitan area networks and show the desirable characteristics for high speed wide area networks.

In contrast to these studies, in this dissertation, we further extend our work to the joint quasi-static multi-class load sharing, routing and congestion control problem in distributed systems. We formulate and solve it as a static team optimization, as a static Nash game and as a static Stackelberg game problem.

2) *Dynamic multi-class routing*

Here, the routing decisions depend on the current network state.

Reed & Kim [397] introduce a dynamic routing algorithm that uses both local and global information about the delay across the paths. Bernabei, Calabro & Listanti [35] suggest a flooding routing scheme in an ATM switch. Szymanski [475] proposes a hot-potato or deflection routing in a fiber optic packet switched hypercube. Borgonovo & Cadorin [67] investigate a locally-optimal deflection routing in the Bidirectional Manhattan Network.

In contrast to these studies, in this dissertation, we further extend our work to the joint dynamic multi-class load sharing, routing and congestion control problem in distributed systems. We formulate and solve it as a dynamic team optimization, as a dynamic Nash game and as a dynamic Stackelberg game problem.

2.3 Congestion Control

In this section, we present studies on congestion control. Traditional packet switching networks have employed typically window-based flow control schemes in order to regulate the traffic. Today's trade towards high speed networks makes such schemes inappropriate. Therefore, we do not consider window-based algorithms.

2.3.1 Datagram Networks

1) *Quasi-static datagram congestion control*

Here, the congestion control decisions depend on the arrival and service distributions.

Gallager & Golestaani [180] use a penalty function approach to model and solve the joint flow control and routing problem. They find the rates that achieve a tradeoff between user cost functions and network congestion cost. Then a dynamic flow control algorithm admits or rejects individual packets.

In contrast to these studies, in this dissertation, we further extend our work to the joint quasi-static datagram load sharing, routing and congestion control problem in distributed systems. We formulate and solve it as a static team optimization, as a static Nash game and as a static Stackelberg game problem.

2) *Dynamic datagram congestion control*

Here, the congestion control decisions depend on the current network state.

Most studies on dynamic congestion control define an ad hoc technique for controlling the traffic and then develop the corresponding queueing model. Subsequently, they evaluate its performance for different values of some adjustable parameters.

Pennotti & Schwartz [378] develop a queueing model for analyzing some congestion control schemes where packets are blocked when the buffer occupancies exceed a limit. Schwartz & Saad [434, 413] suggest a tighter control over new packets with priority allotted to transit packets. They propose blocking of new packets when the total node buffer occupancy exceeds a limit.

Rudin & Muller [410, 411] indicate that it may be dangerous to try to study flow control, routing or scheduling as isolated mechanisms. Lam & Reiser [287] and Lam [284] investigate the use of input buffer limits for congestion control. Wunderlich, Kaufman & Gopinath [519] suggest a buffer reservation and processor capacity allocation scheme.

Kamoun, Belguith and Grange [240] and Kamoun [239] propose a drop and throttle flow control policy based on a nodal buffer management scheme. At a given node if the number of allocated buffers is greater than a limit value, then new traffic is rejected, whereas transit traffic is accepted. If the total buffer area is occupied, transit traffic is also rejected and, furthermore, it is dropped from the network. Kleinrock & Tseng [258] propose two schemes that limit the permit generation rate.

Filipiak [157, 153] formulates the congestion clearance problem as an optimal control problem. Stassinopoulos & Konstantopoulos [466] present an algorithm for minimum delay clearing of congested single-destination networks, in minimum time. They also give conditions for arrival rates, so that the minimal value for the maximal delay encountered by any packet does not increase.

Hahne & Gallager [209] investigate the round robin scheduling for fair flow control. They maximize the minimum packet rate of a virtual circuit. Kim & Towsley [252] propose three packet discarding schemes and develop their queueing models. Thaker & Cain [481] discuss the interaction between routing and flow control and present a new scheme. Jacobson [232] describes the internet's congestion control mechanisms. Ramakrishnan & Jain [392] propose a dynamic congestion avoidance scheme that sets up a congestion-indication bit on regular packets, then the source node adapts to the new situation.

Hirano & Watanabe [216] evaluates a marking method for congestion control, which marks packets exceeding the allocated bandwidth at user/network interfaces, and discards these marked packets only at congested nodes. Kamitake & Suda [238] investigate a congestion control scheme and develop its queueing model. Schulzrinne, Kurose & Towsley [430] investigate the selective discarding of a voice packet based on the virtual work found by the packet on arrival to a queue. Petr

& Frost [380] show how dynamic programming can be used to find an optimal discarding policy for congestion control.

In contrast to these studies, in this dissertation, we further extend our work to the joint dynamic datagram load sharing, routing and congestion control problem in distributed systems. We formulate and solve it as a dynamic team optimization, as a dynamic Nash game and as a dynamic Stackelberg game problem.

2.3.2 Virtual Circuit Networks

1) *Quasi-static virtual circuit congestion control*

Here, the congestion control decisions depend on the arrival and service distributions.

Jaffe [233] suggests that each virtual circuit adjusts its throughput rate to achieve an ideal delay-throughput tradeoff. Bharath-Kumar [49] maximizes the power by selecting the rate at which messages are allowed to enter the message path and the average total number in the system. Bharath-Kumar & Jaffe [50] investigate flow control algorithms to maximize the power. They control the rate of message entry to every virtual circuit.

In contrast to these studies, in this dissertation, we further extend our work to the joint quasi-static virtual circuit load sharing, routing and congestion control problem in distributed systems. We formulate and solve it as a static team optimization, as a static Nash game and as a static Stackelberg game problem.

2) *Dynamic virtual circuit congestion control*

Here, the congestion control decisions depend on the current network state.

Matsumoto and Mori [323] restrict gradually the number of virtual circuits when the queue lengths exceed some limits. Katz & Rubin [248] analyze a virtual circuit admission scheme that limits the total number of virtual circuits allowed to share a channel. Golestani [200, 199] proposes an admission control that requires the packet stream of each virtual circuit to possess a certain smoothness property upon arrival to the network and a queueing control scheme that preserves this property as packets travel inside the network. Bala, Cidon & Sohraby [20] propose a leaky bucket type scheme that operates on a virtual circuit basis that limits the

virtual circuit's average rate and the burstiness. It marks packets into two classes which are treated differently by threshold policies.

Economides & Silvester [136] consider the dynamic routing and congestion control problem for window flow controlled single source-destination virtual circuit networks with error prone links. They define a multi-objective function of the average number of virtual circuits and packets on every link, the virtual circuit rejection flow and the packet throughput on every link. Pontryagin's maximum principle provides the necessary optimality conditions that are also sufficient for this convex virtual circuit routing and congestion control problem. A new virtual circuit is admitted into the network if the cost that it will produce is less then the profit that it will offer.

In contrast to these studies, in this dissertation, we further extend our work to the joint dynamic virtual circuit load sharing, routing and congestion control problem in distributed systems. We formulate and solve it as a dynamic team optimization, as a dynamic Nash game and as a dynamic Stackelberg game problem.

2.3.3 Integrated Services Networks

1) *Quasi-static multi-class congestion control*

Here, the congestion control decisions depend on the arrival and service distributions. There is some ambiguity with respect to where some proposed congestion control schemes belong, i.e. quasi-static or dynamic. Since, most authors choose to call their schemes dynamic, we shall refer to them on the next section.

In this dissertation, we further extend our work to the joint quasi-static multi-class load sharing, routing and congestion control problem in distributed systems. We formulate and solve it as a static team optimization, as a static Nash game and as a static Stackelberg game problem.

2) *Dynamic multi-class congestion control*

Here, the congestion control decisions depend on the current network state.

Hou & Lucantoni [219] propose a video/data transport mechanism with built-in congestion control. Data calls have priority during light overload, while video

calls have preemptive priority over data during heavy overload. Ohnishi, Okada & Noguchi [359] discuss a virtual circuit admission control scheme and different priority assignments to different classes.

Hui [227] and Filipiak [160, 154] suggest several congestion measures at the packet, burst and call level. Gersht & Lee [191] propose a congestion control scheme that reserves bandwidth at the virtual circuit set-up instant and chokes/relieves only first-class packets in case of congestion. They also [190] develop an integer optimization algorithm to find the thresholds for their congestion control scheme. Sriram [457] suggests dropping the less significant bits in voice packets for congestion control. Ramamurthy & Dighe [394, 395] propose a rate-based flow control scheme. The rate of each virtual circuit is negotiated at call set up time.

Woodruff & Kositpaiboon [517] suggest that bursty traffic should be statistically multiplexed only if virtual circuit peak rates are low relative to the link speeds, or burst durations are short. Decina, Toniatti, Vaccari & Verri [126] and Decina & Toniatti [125] propose admitting a new virtual circuit into the network, when the sum of peak rates of virtual circuits on any link composed its route does not exceed the bit rate of that link.

Eckberg, Luan & Lucantoni [135] describe congestion and flow control strategies. Sidi, Liu & Gopal [446] propose a leaky-bucket type scheme for congestion control in order to smoothen the incoming traffic. Berger [32] analyzes a rate control throttle scheme where both token and jobs queue. Sumita [472] describes an output buffer management scheme based on a conservation law for an $M/D/1$ queue with finite capacity. Kroner [270] suggests the use of partial buffer sharing instead of the push-out scheme and the separate route system.

In contrast to these studies, in this dissertation, we further extend our work to the joint dynamic multi-class load sharing, routing and congestion control problem in distributed systems. We formulate and solve it as a dynamic team optimization, as a dynamic Nash game and as a dynamic Stackelberg game problem.

2.4 Game Theory Approaches to Flow Control

In this section, we review previous studies on the quasi-static flow control problem using a game theory approach. Although in this dissertation, we consider a different problem (the quasi-static and dynamic joint load sharing, routing and congestion control problem), we present these studies on the flow control problem, so that the reader may have an integrated view of current research on network optimization problems using a game theory approach.

Sanders [420, 418, 419] presents incentive flow control algorithms that allocate transmission rates to each virtual circuit. Cansever [82] presents a greedy algorithm to compute the Nash equilibrium and another algorithm to compute the Pareto optimum for the flow control problem. Douligeris & Mazumdar [130, 129] consider the flow control problem and suggest an algorithm for finding the Pareto optimum that maximizes the product of individual powers. They also [128] find the flow control Nash and Stackelberg equilibria for specific examples.

Hsiao & Lazar [221, 224] show that the flow controls for team optimum are a set of window-type mechanisms. In [290, 222, 225], they consider the user optimum flow control problem, in which each user maximizes its average throughput subject to a constraint on its average delay. They further [223] investigate the flow control Nash equilibrium solution of Markovian queueing networks. Bovopoulos & Lazar [75, 70, 69, 68] investigate iterative asynchronous algorithms for flow control policies to achieve Nash equilibrium. They maximize either the throughput under an average delay constraint, or the power. They also [71] suggest the Gauss-Seidel algorithm to find the Nash equilibrium that maximizes the power. Ferguson, Nikolaou & Yemini [147] consider the Pareto optimum flow control and present algorithms to find it.

Chapter 3

Queueing Model

In this chapter, we develop a generic framework on which we shall based our methodology. We introduce the analytical model that integrates the load sharing, routing and congestion control problems. We formulate the problem on the path flow space, such that the decisions are done at the source nodes. We also introduce the structure of the cost functions and the state space model, for which more details are given at the application sections of following chapters.

3.1 Path Flow Model

We consider a distributed system as seen by jobs of class c, as a set of links $\mathbf{L}^c$, of nodes $\mathbf{N}^c$, of source-destination pairs $\mathbf{SD}^c$, of paths between a given source-destination pair $[sd]$, $\mathbf{\Pi}^c_{[sd]}$, of destination nodes for a given source node $[s.]$, $\mathbf{D}^c_{[s.]}$, of all destination nodes $\mathbf{D}^c = \bigcup_{[s.]} \mathbf{D}^c_{[s.]}$, of source nodes for a given destination node $[.d]$, $\mathbf{S}^c_{[.d]}$, and of all source nodes $\mathbf{S}^c = \bigcup_{[.d]} \mathbf{S}^c_{[.d]}$, for jobs of class c.

The formulation of the joint load sharing, routing and congestion control problem can be done either on the link flow space or on the path flow space. In future high speed computer communication networks the transmission delay will be extremely low and we will not want to spend extra time in network management decisions inside the network. Therefore, the computationally intensive processes,

such as the network management decisions, will be transferred outside of the network either to the source or to the destination node. With this in mind, we formulate the joint load sharing, routing and congestion control problem on the path flow space, which means that the routing decisions will be done at the source nodes. In this way, we also avoid loops, since the packets will follow a previously determined loop free path.

Class c jobs that require processing arrive at the source node $[s.]$ with external arrival rate $\gamma^c_{[s.]}(t) \geq 0$ at time t (Figure 3.1). A load sharing decision is made as to where these jobs will be processed. Let the fraction of jobs sent to node $[.d]$ for processing be $\psi^c_{[sd]}(t)$. Since only one destination node is selected, the sum of the load sharing fractions from node $[.s]$ to all destination nodes $[.d]$ is equal to one. So, let us define the constraint set for the class c load sharing decision variables to be

$$\mathbf{LS}^c = \Big\{ \psi^c_{[sd]}(t) \quad \forall\, [.d] \in \mathbf{D}^c_{[s.]} \quad \forall\, [s.] \in \mathbf{S}^c \;/\; such\ that$$

$$\sum_{[.d] \in \mathbf{D}^c_{[s.]}} \psi^c_{[sd]}(t) = 1 \quad \forall\, [.s] \in \mathbf{S}^c, \qquad and$$

$$\psi^c_{[sd]}(t) \geq 0 \quad \forall\, [.d] \in \mathbf{D}^c_{[s.]}, \quad [s] \in \mathbf{S}^c \Big\}$$

For a more compact presentation we write the load sharing fractions from source node $[s.]$ to all destination nodes for class c as the vector $\mathbf{\Psi}^c_{[s.]}(t) = [\ldots\ \psi^c_{[sd]}(t)\ \ldots]$, from all source nodes to all destination nodes for class c as the vector $\mathbf{\Psi}^c(t) = [\ldots\ \mathbf{\Psi}^c_{[s.]}(t)\ \ldots]$ and for all classes as the vector $\mathbf{\Psi}(t) = [\ldots\ \mathbf{\Psi}^c(t)\ \ldots]$.

Since jobs arrive at the source node $[s.]$ with external arrival rate $\gamma^c_{[s.]}(t)$, then the flow from source node $[s.]$, transferred for processing to the destination node $[.d]$, will be $\gamma^c_{[s.]}(t) * \psi^c_{[sd]}(t)$.

Therefore, the total class c flow that is transferred for processing to the destination node $[.d]$, due to load sharing is :

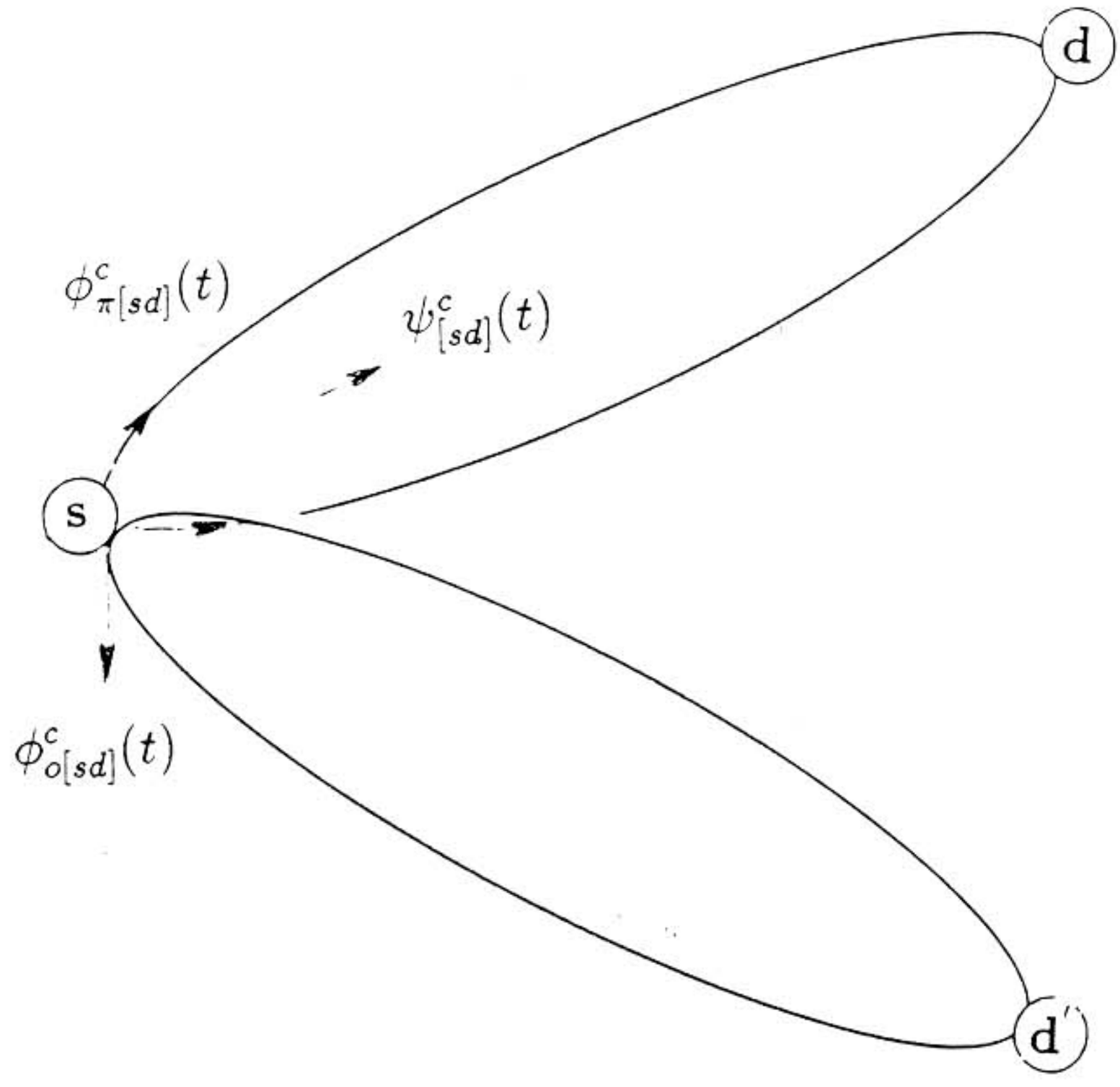


Figure 3.1: Load sharing, routing and congestion control.

$$\lambda^c_{[.d]}(t) = \sum_{[s.]\in \mathbf{S}^c_{[.d]}} \gamma^c_{[s.]}(t) * \psi^c_{[sd]}(t)$$

For a more compact presentation we write the flow transferred to the destination node $[.d]$ due to load sharing reasons for all classes as the vector $\mathbf{\Lambda}_{[.d]}(t) = [... \ \lambda^c_{[.d]}(t) \ ...]$.

After it is decided which fraction of jobs will be processed by destination node $[.d]$, they should be transferred there- this is the routing problem. For routing, we must specify which path between source-destination pair $[sd]$ will be selected. In addition, some jobs may be rejected outside of the network, for congestion control reasons. So, let the fraction of rejected jobs for the source-destination pair $[sd]$ be $\phi^c_{o[sd]}(t)$, and the fraction of jobs routed through path $\pi[sd]$ be $\phi^c_{\pi[sd]}(t)$ (Figure 3.1). Since a job for the source-destination pair may only be rejected or routed through a single path, all these fractions must sum to one. So, let define the constraint set for the class c routing and congestion control decision variables as

$$\begin{aligned}
\mathbf{RC}^c \ = \ \Big\{ & \ \phi^c_{o[sd]}(t), \ \phi^c_{\pi[sd]}(t) \quad \forall \ \pi[sd] \in \mathbf{\Pi}^c_{[sd]} \quad \forall \ [sd] \ / \ such \ that \\
& \phi^c_{o[sd]}(t) + \sum_{\pi[sd]\in \mathbf{\Pi}^c_{[sd]}} \phi^c_{\pi[sd]}(t) = 1 \quad \forall \ [sd] \\
& \phi^c_{o[sd]}(t), \ \phi^c_{\pi[sd]}(t) \geq 0 \quad \forall \ \pi[sd] \in \mathbf{\Pi}^c_{[sd]}, \quad [sd] \ \Big\}
\end{aligned}$$

For a more compact presentation we write the routing and congestion control fractions of all paths between source-destination pair $[sd]$ for class c as the vector $\mathbf{\Phi}^c_{[sd]}(t) = [\phi^c_{o[sd]}(t)... \ \phi^c_{\pi[sd]}(t) \ ...]$, of all paths for class c as the vector $\mathbf{\Phi}^c(t) = [... \ \mathbf{\Phi}^c_{[sd]}(t) \ ...]$ and for all classes as the vector $\mathbf{\Phi}(t) = [... \ \mathbf{\Phi}^c(t) \ ...]$.

Furthermore, jobs arrive at the source node $[s.]$ and require only transfer to the destination node $[.d]$ (i.e. no processing), with external arrival rate $\gamma^c_{[sd]}(t) \geq 0$. Since there is flow $\gamma^c_{[s.]}(t) * \psi^c_{[sd]}(t)$ (due to load sharing decisions) and flow $\gamma^c_{[sd]}(t)$

(due to communication requirements) for the source-destination pair $[sd]$, the total class c flow for the source-destination pair $[sd]$ is $\gamma^c_{[sd]}(t) + \gamma^c_{[s.]}(t) * \psi^c_{[sd]}(t)$.

Due to the congestion control, the fraction of this flow that will be rejected is $\phi^c_{o[sd]}(t)$. So, let the rejected flow for the source-destination pair $[sd]$ be

$$\lambda^c_{o[sd]}(t) = (\gamma^c_{[sd]}(t) + \gamma^c_{[s.]}(t) * \psi^c_{[sd]}(t)) * \phi^c_{o[sd]}(t)$$

For a more compact presentation we write the rejected flow for the source-destination pair $[sd]$ for all classes as the vector $\mathbf{\Lambda}_{o[sd]}(t) = [... \ \lambda^c_{o[sd]}(t) \ ...]$.

Another fraction $\phi^c_{\pi[sd]}(t)$ of this flow will be routed through path $\pi[sd]$ Therefore, the resulting flow on path $\pi[sd]$ will be $(\gamma^c_{[sd]}(t) + \gamma^c_{[s.]}(t) * \psi^c_{[sd]}(t)) * \phi^c_{\pi[sd]}(t)$. This flow will be assigned to the links and nodes that constitute this path. So, the class c flow on link ij will be the sum of all path flows that traverse this link :

$$\lambda^c_{ij}(t) = \sum_{[sd] \in \mathbf{SD}^c} \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} (\gamma^c_{[sd]}(t) + \gamma^c_{[s.]}(t) * \psi^c_{[sd]}(t)) * \phi^c_{\pi[sd]}(t) * 1_{ij \in \pi[sd]}(t)$$

where $1_{ij \in \pi[sd]}(t)$ is the indicator function that link ij is on the path $\pi[sd]$. For a more compact presentation let us write the flow on link ij for all classes as the vector $\mathbf{\Lambda}_{ij}(t) = [... \ \lambda^c_{ij}(t) \ ...]$.

Similarly, the class c flow at node i will be the sum of all path flows that traverse this node :

$$\lambda^c_{i}(t) = \sum_{[sd] \in \mathbf{SD}^c} \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} (\gamma^c_{[sd]}(t) + \gamma^c_{[s.]}(t) * \psi^c_{[sd]}(t)) * \phi^c_{\pi[sd]}(t) * 1_{i \in \pi[sd]}(t)$$

For a more compact presentation let us write the flow at node i for all classes as the vector $\mathbf{\Lambda}_i(t) = [... \ \lambda^c_i(t) \ ...]$ and the flow at all network nodes and links for all classes as the vector $\mathbf{\Lambda}(t) = [... \ \mathbf{\Lambda}_{[.d]}(t) \ ... \ \mathbf{\Lambda}_{o[sd]}(t) \ ... \ \mathbf{\Lambda}_{ij}(t) \ ... \ \mathbf{\Lambda}_i(t) \ ...]$.

3.2 State Space Model

In this section, we describe the dynamic evolution of the system state using a state space model for each system resource. The real system state is a stochastic process (discrete-state continuous time). However, the decision makers cannot have instantaneous knowledge of the global state at every instant. So, even if we solve the stochastic problem, it will be difficult to implement the solution. Therefore, we use the deterministic approximation of this stochastic process by its expected value. We define as state of a system resource, the average work at this resource (continuous-state continuous-time process). This work may be the number of packets, bursts, virtual circuits etc. The average work increases during a time interval by the average work that arrives during this time interval and decreases by the average work that departs (having received service) during this interval. So, if $\mathbf{X}(t)$ is the system state, $\mathbf{\Lambda}(t)$ is the arrival rate and $\mathbf{D}(t)$ is the departure rate, we can write

$$\dot{\mathbf{X}}(t) = \mathbf{\Lambda}(t) - \mathbf{D}(t)$$

Under specific assumptions on the network operation and the traffic distributions, the above abstract form of the state space model reduces to specific differential equations (see sections 5.4, 5.5).

We can write such differential equations for each class c, for each source-destination pair $[sd]$, at each system resource. Let $\mathbf{X}^c_{o[sd]}(t)$ be the state of the rejected class c flow for the source-destination pair $[sd]$. Then the following differential vector equation describes the class c rejected flow for the $[sd]$ pair over time:

$$\dot{\mathbf{X}}^c_{o[sd]}(t) = \mathbf{f}^c_{o[sd]}(t, \mathbf{X}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t))$$

Let $\mathbf{X}^c_{ij[sd]}(t)$ be the state of the class c flow at link ij for the source-destination $[sd]$. Then the following differential vector equation describes the class c flow at link ij for the $[sd]$ pair over time:

$$\dot{\mathbf{X}}^c_{ij[sd]}(t) = \mathbf{f}^c_{ij[sd]}(t, \mathbf{X}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t))$$

Let $\mathbf{X}^c_{i[sd]}(t)$ be the state of the class c flow at node i for the source-destination $[sd]$. Then the following differential vector equation describes the class c flow at node i for the $[sd]$ pair over time:

$$\dot{\mathbf{X}}^c_{i[sd]}(t) = \mathbf{f}^c_{i[sd]}(t, \mathbf{X}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t))$$

Let $\mathbf{X}^c_{[.d][sd]}(t)$ be the state of the class c flow at destination node $[.d]$ from the source node $[s.]$. Then the following differential vector equation describes the class c flow at destination node $[.d]$ originated at node $[s.]$:

$$\dot{\mathbf{X}}^c_{[.d][sd]}(t) = \mathbf{f}^c_{[.d][sd]}(t, \mathbf{X}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t))$$

The state of the network is described by the state of its links, its nodes, its rejected flow and its destination processing sites. So, we define as the network state the vector

$$\mathbf{X}(t) = [... \ \mathbf{X}^c_{ij[sd]}(t) \ ... \ \mathbf{X}^c_{i[sd]}(t) \ ... \ \mathbf{X}^c_{o[sd]}(t) \ ... \ \mathbf{X}^c_{[.d][sd]}(t) \ ...]^T$$

and the differential vector equation that describes the dynamic network evolution

$$\dot{\mathbf{X}}(t) = \mathbf{f}(t, \mathbf{X}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t))$$

In sections 5.4, 5.5 specific versions of these equations are developed.

3.3 Multi-Objective Cost Function

In this section, we describe the multi-objective cost function for the joint load sharing, routing and congestion control problem. First, let us make some useful definitions:

Definition :

Consider the dynamic joint load sharing, routing and congestion control problem in distributed systems with multiple classes. Define the twice continuously differentiable (with respect to $\mathbf{X}$*) vector functions*

$\mathbf{Y}^c(t) = \mathbf{h}^c(t, \mathbf{X}(t))$: *state measurement (observation) of class c and*

$\mathbf{Y}(t) = \mathbf{h}(t, \mathbf{X}(t))$: *global state measurement (observation).*

Define the state observation structure of class c as

1) *perfect state observation iff* $\mathbf{Y}^c(t) = \mathbf{X}(t), \qquad t \in [t_0, t_f]$.

2) *imperfect state observation sharing iff* $\mathbf{Y}^c(t) = \mathbf{Y}(t) \neq \mathbf{X}(t), \; t \in [t_0, t_f]$.

3) *imperfect state observation no sharing iff* $\mathbf{Y}^c(t) \neq \mathbf{Y}(t) \neq \mathbf{X}(t), \; t \in [t_0, t_f]$.

Definition :

Consider the dynamic joint load sharing, routing and congestion control problem in distributed systems with multiple classes. Define

$\mathbf{I}^c(t)$: *information set of class c*

Define the information structure of class c as

1) *open-loop iff* $\mathbf{I}^c(t) = \{\mathbf{X}_0\}, \qquad t \in [t_0, t_f]$,

2) *closed-loop iff* $\mathbf{I}^c(t) = \{\mathbf{Y}^c(s), \;\; \forall \; s \in [t_0, t]\}, \qquad t \in [t_0, t_f]$,

3) *memoryless iff* $\mathbf{I}^c(t) = \{\mathbf{X}_0, \;\; \mathbf{Y}^c(t)\}, \qquad t \in [t_0, t_f]$,

4) *feedback iff* $\mathbf{I}^c(t) = \{\mathbf{Y}^c(t)\}, \qquad t \in [t_0, t_f]$,

5) *τ-delayed closed-loop iff* $\mathbf{I}^c(t) = \{\mathbf{Y}^c(s), \;\; \forall \; s \in [t_0, t - \tau]\}, \; t \in [t_0, t_f]$,

6) *τ-delayed memoryless iff* $\mathbf{I}^c(t) = \{\mathbf{X}_0, \;\; \mathbf{Y}^c(t - \tau)\}, \qquad t \in [t_0, t_f]$,

7) *τ-delayed feedback iff* $\mathbf{I}^c(t) = \{\mathbf{Y}^c(t - \tau)\}, \qquad t \in [t_0, t_f]$,

Let $\mathbf{\Lambda} = [... \; \mathbf{\Lambda}^c \; ...]$, where $\mathbf{\Lambda}^c = \{\mathbf{\Lambda}^c(\mathbf{I}^c(t)), \;\; \forall \; t \in [t_0, t_f]\}$ is the strategy of class c during the whole duration of the problem.

Next, we consider the cost function for class c at time t to be $g^c(t, \mathbf{X}(t), \mathbf{\Lambda}(t))$. a function of the system state and flows and the total cost function $J^c(\mathbf{\Lambda}) = \int_{t_0}^{t_f} g^c(t, \mathbf{X}(t), \mathbf{\Lambda}(t))dt$. We can decompose this cost as the sum of its average cost : i)at every link ij, $g^c_{ij}(t, \mathbf{X}(t), \mathbf{\Lambda}(t))$ (e.g. transmission and propagation cost) and $J^c_{ij}(\mathbf{\Lambda}) = \int_{t_0}^{t_f} g^c_{ij}(t, \mathbf{X}(t), \mathbf{\Lambda}(t))dt$; ii) at every node i, $g^c_i(t, \mathbf{X}(t), \mathbf{\Lambda}(t))$ (e.g. processing cost) and $J^c_i(\mathbf{\Lambda}) = \int_{t_0}^{t_f} g^c_i(t, \mathbf{X}(t), \mathbf{\Lambda}(t))$; iii) at every source-destination pair $[sd]$, $g^c_{[sd]}(t, \mathbf{X}(t), \mathbf{\Lambda}(t))$ (e.g. cost for rejecting jobs) and $J^c_{[sd]}(\mathbf{\Lambda}) = \int_{t_0}^{t_f} g^c_{[sd]}(t, \mathbf{X}(t), \mathbf{\Lambda}(t))dt$; and iv) at every destination node $[.d]$, $g^c_{[.d]}(t, \mathbf{X}(t), \mathbf{\Lambda}(t))$ (e.g. cost for processing jobs) and $J^c_{[.d]}(\mathbf{\Lambda}) = \int_{t_0}^{t_f} g^c_{[.d]}(t, \mathbf{X}(t), \mathbf{\Lambda}(t))dt$. Note that in the above general form of the cost function, the cost at each link or node may depend on the traffic over the whole network, e.g. in packet radio networks, the delay on a radio link depends not only on the traffic on this radio link, but also on the traffic in other interfering radio links from neighboring nodes.

So, the cost function for class c at time t is

$$\begin{aligned} g^c(t, \mathbf{X}(t), \mathbf{\Lambda}(t)) &= \sum_{ij} g^c_{ij}(t, \mathbf{X}(t), \mathbf{\Lambda}(t)) + \sum_{i} g^c_i(t, \mathbf{X}(t), \mathbf{\Lambda}(t)) + \\ &+ \sum_{[sd]} g^c_{[sd]}(t, \mathbf{X}(t), \mathbf{\Lambda}(t)) + \sum_{[.d]} g^c_{[.d]}(t, \mathbf{X}(t), \mathbf{\Lambda}(t)) \end{aligned}$$

The objective function during the whole duration of the problem, from the initial time t_0 to the final time t_f, becomes

$$J^c(\mathbf{\Lambda}) = \int_{t_0}^{t_f} g^c(t, \mathbf{X}(t), \mathbf{\Lambda}(t))dt =$$

$$= \sum_{ij} \int_{t_0}^{t_f} g_{ij}^c(t, \mathbf{X}(t), \mathbf{\Lambda}(t))dt + \sum_{i} \int_{t_0}^{t_f} g_i^c(t, \mathbf{X}(t), \mathbf{\Lambda}(t))dt +$$

$$+ \sum_{[sd]} \int_{t_0}^{t_f} g_{[sd]}^c(t, \mathbf{X}(t), \mathbf{\Lambda}(t))dt + \sum_{[.d]} \int_{t_0}^{t_f} g_{[.d]}^c(t, \mathbf{X}(t), \mathbf{\Lambda}(t))dt$$

For the infinite horizon problem, we consider the following cost function

$$J^c(\mathbf{\Lambda}) = \sum_{ij} \int_{t_0}^{\infty} e^{-c_{ij}t} g_{ij}^c(t, \mathbf{X}(t), \mathbf{\Lambda}(t))dt + \sum_{i} \int_{t_0}^{\infty} e^{-c_i t} g_i^c(t, \mathbf{X}(t), \mathbf{\Lambda}(t))dt +$$

$$+ \sum_{[sd]} \int_{t_0}^{\infty} e^{-c_{[sd]}t} g_{[sd]}^c(t, \mathbf{X}(t), \mathbf{\Lambda}(t))dt + \sum_{[.d]} \int_{t_0}^{\infty} e^{-c_{[.d]}t} g_{[.d]}^c(t, \mathbf{X}(t), \mathbf{\Lambda}(t))dt$$

where $c_{ij}, c_i, c_{[sd]}, c_{[.d]}$ are discount costs.

For networks where the cost of each resource depends only on the traffic on this resource, the cost function for class c at time t becomes

$$g^c(t, \mathbf{X}(t), \mathbf{\Lambda}(t)) = \sum_{ij \in \mathbf{L}^c} g_{ij}^c(t, \mathbf{X}(t), \mathbf{\Lambda}_{ij}(t)) + \sum_{i \in \mathbf{N}^c} g_i^c(t, \mathbf{X}(t), \mathbf{\Lambda}_i(t)) +$$
$$+ \sum_{[sd] \in \mathbf{SD}^c} g_{[sd]}^c(t, \mathbf{X}(t), \mathbf{\Lambda}_{o[sd]}(t)) + \sum_{[.d] \in \mathbf{D}_{[s.]}^c} g_{[.d]}^c(t, \mathbf{X}(t), \mathbf{\Lambda}_{[.d]}(t))$$

and the objective function during the whole duration of the problem, from the initial time t_0 to the final time t_f, becomes

$$J^c(\mathbf{\Lambda}) = \sum_{ij \in \mathbf{L}^c} \int_{t_0}^{t_f} g_{ij}^c(t, \mathbf{X}(t), \mathbf{\Lambda}_{ij}(t))dt \quad + \quad \sum_{i \in \mathbf{N}^c} \int_{t_0}^{t_f} g_i^c(t, \mathbf{X}(t), \mathbf{\Lambda}_i(t))dt \quad +$$
$$+ \sum_{[sd] \in \mathbf{SD}^c} \int_{t_0}^{t_f} g_{[sd]}^c(t, \mathbf{X}(t), \mathbf{\Lambda}_{o[sd]}(t))dt \quad + \quad \sum_{[.d] \in \mathbf{D}_{[s.]}^c} \int_{t_0}^{t_f} g_{[.d]}^c(t, \mathbf{X}(t), \mathbf{\Lambda}_{[.d]}(t))dt$$

and for the infinite horizon problem, the cost function becomes

$$J^c(\mathbf{\Lambda}) = \sum_{ij \in \mathbf{L}^c} \int_{t_0}^{\infty} e^{-c_{ij}t} g_{ij}^c(t, \mathbf{X}(t), \mathbf{\Lambda}_{ij}(t))dt \quad + \quad \sum_{i \in \mathbf{N}^c} \int_{t_0}^{\infty} e^{-c_i t} g_i^c(t, \mathbf{X}(t), \mathbf{\Lambda}_i(t))dt \quad +$$
$$\sum_{[sd] \in \mathbf{SD}^c} \int_{t_0}^{\infty} e^{-c_{od}t} g_{[sd]}^c(t, \mathbf{X}(t), \mathbf{\Lambda}_{o[sd]}(t))dt \quad + \quad \sum_{[.d] \in \mathbf{D}_{[s.]}^c} \int_{t_0}^{\infty} e^{-c_{[.d]}t} g_{[.d]}^c(t, \mathbf{X}(t), \mathbf{\Lambda}_{[.d]}(t))dt$$

The cost function for class c may also be written as a function $J^c(\mathbf{\Phi}, \mathbf{\Psi}) = \int_{t_0}^{t_f} g^c(t, \mathbf{X}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t))dt$ of the routing, congestion control and load sharing fractions on the path flow space, where the strategies of each class c during the whole duration of the game are

$$(\mathbf{\Phi}^c, \mathbf{\Psi}^c) = \left\{ (\hat{\mathbf{\Phi}}^c(\mathbf{I}^c(t)), \hat{\mathbf{\Psi}}^c(\mathbf{I}^c(t))), \ \forall\, t \in [t_0, t_f] \right\}, \quad \forall\, c$$

and for all classes

$$(\mathbf{\Phi}, \mathbf{\Psi}) = [..., \ (\mathbf{\Phi}^c, \mathbf{\Psi}^c), \ ...]$$

Throughout this dissertation, all arrival rates and cost functions are considered to be nonnegative. Also, the feasibility set is considered to be nonempty.

In the next chapters, we formulate and solve the joint load sharing, routing and congestion control problem as a Pareto, Nash and Stackelberg game. For each case, we further give three alternative formulations, namely nonlinear programming, nonlinear complementarity and variational inequality problem formulation.

In the optimization problem, we shall use the Hamiltonian and Lagrangian functions, which we define next:

Define the Hamiltonian for each class c as

$$H^c(t, \mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi}, \mathbf{P}^c) = g^c(t, \mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi}) + \mathbf{P}^c * \mathbf{f}(t, \mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi})$$

where $\mathbf{P}^c = [... \; P^{c,k}_{ij[sd]} \; ... \; P^{c,k}_{i[sd]} \; ... \; P^{c,k}_{o[sd]} \; ... \; P^{c,k}_{[.d][sd]} \; ...]$: vector of class c costate variables.

Define also the derivatives of H^c with respect to the congestion, routing and load sharing fractions at $(t, \mathbf{X}^*(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t), \mathbf{P}^c(t))$ as

$$\frac{\partial H^{c*}}{\phi^c_{o[sd]}} = \frac{\partial H^c(t, \mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi}, \mathbf{P}^c)}{\phi^c_{o[sd]}} \Big|_{(t, \mathbf{X}^*(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t), \mathbf{P}^c(t))}$$

$$\frac{\partial H^{c*}}{\phi^c_{\pi[sd]}} = \frac{\partial H^c(t, \mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi}, \mathbf{P}^c)}{\phi^c_{\pi[sd]}} \Big|_{(t, \mathbf{X}^*(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t), \mathbf{P}^c(t))}$$

$$\frac{\partial H^{c*}}{\psi^c_{[sd]}} = \frac{\partial H^c(t, \mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi}, \mathbf{P}^c)}{\psi^c_{[sd]}} \Big|_{(t, \mathbf{X}^*(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t), \mathbf{P}^c(t))}$$

Define also the Lagrangian for each class c as

$$\begin{aligned} L^c(t, \mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi}, \mathbf{P}^c, \mathbf{Q}^c) \; &= H^c(t, \mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi}, \mathbf{P}^c) + \\ &+ \sum_{[sd] \in \mathbf{SD}^c} Q^c_{[sd]} * \left[1 - \phi^c_{o[sd]} - \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \phi^c_{\pi[sd]} \right] + \\ &+ \sum_{[s.] \in \mathbf{S}^c} Q^c_{[s.]} * \left[1 - \sum_{[.d] \in \mathbf{D}^c_{[s.]}} \psi^c_{[sd]} \right] \end{aligned}$$

with $\phi^c_{o[sd]}, \; \phi^c_{\pi[sd]}, \; \psi^c_{[sd]} \geq 0 \quad \forall \; \pi[sd] \in \mathbf{\Pi}^c_{[sd]}, \; [sd] \in \mathbf{SD}^c, \; c.$

where $\mathbf{Q}^c = [... \; Q^c_{[sd]} \; ... \; Q^c_{[s.]} \; ...]$: vector of class c multipliers for the constraints of the congestion control, routing and load sharing fractions.

Define also the derivatives of L^c with respect to the congestion, routing and load sharing fractions at $(t, \mathbf{X}^*(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t), \mathbf{P}^c(t))$ as

$$\frac{\partial L^{c*}}{\phi^c_{o[sd]}} = \frac{\partial L^c(t, \mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi}, \mathbf{P}^c, \mathbf{Q}^c)}{\phi^c_{o[sd]}} \Big|_{(t, \mathbf{X}^*(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t), \mathbf{P}^c(t), \mathbf{Q}^c(t))}$$

$$\frac{\partial L^{c*}}{\phi^c_{\pi[sd]}} = \frac{\partial L^c(t, \mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi}, \mathbf{P}^c, \mathbf{Q}^c)}{\phi^c_{\pi[sd]}} \Big|_{(t, \mathbf{X}^*(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t), \mathbf{P}^c(t), \mathbf{Q}^c(t))}$$

$$\frac{\partial L^{c*}}{\psi^c_{[sd]}} = \frac{\partial L^c(t, \mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi}, \mathbf{P}^c, \mathbf{Q}^c)}{\psi^c_{[sd]}} \Big|_{(t, \mathbf{X}^*(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t), \mathbf{P}^c(t), \mathbf{Q}^c(t))}$$

3.4 State Constraints

In this section, we describe constraints on the values that the system state (e.g. number of virtual circuits, number of packets) may take. One such constraint is the non-negativity of the state

$$\mathbf{X}(t) \geq 0$$

For flow control as well as implementation reasons, there may be restrictions on the network state

$$\mathbf{W}(\mathbf{X}(t)) \leq 0$$

For example, the expected number of packets at each link should be less than a window size for this link.

We may also have a set of initial constraints, e.g. $\mathbf{I}(\mathbf{X}(t_0)) \leq 0$ for the initial network state $\mathbf{X}(t_0)$, and a set of final constraints $\mathbf{F}(\mathbf{X}(t_f)) \leq 0$ for the final network state $\mathbf{X}(t_f)$. For example, the expected number of packets at the final time should be zero.

In the next chapters, we shall further describe particular constraints on the state for different kinds of networks. Note, that if we consider state constraints in our optimization problem, then the Lagrangian should be suitably modified.

Chapter 4

Quasi-Static Formulation

In this chapter, we develop three novel methodologies for the quasi-static problem: i) in the team optimization methodology, the classes of jobs cooperate in using the resources of the system for the socially optimum, ii) in the Nash game methodology, the classes of jobs compete among themselves and each class tries to operate optimally for its own jobs, and iii) in the Stackelberg game methodology, some classes of jobs have more power than others, for example priorities. For each methodology, we develop three alternative formulations of the joint problem, namely a nonlinear programming, a nonlinear complementarity problem and a variational inequality formulation. For each formulation, we state the necessary and sufficient conditions for existence and uniqueness of the solution. From the Karush-Kuhn-Tucker conditions, we also derive the abstract form of the solution, that there should be flow only on minimum length paths, to minimum length destinations, The length at each system resource is appropriately defined for each case. Then we apply these three methodologies to datagram, virtual circuit and integrated services networks. For each of these networks, we introduce cost functions for multiple classes and for priority classes of jobs. We also explicitly solve three examples: i) In the first example, two classes of jobs share two processors. The objectives are the minimize the expected job delays. We give in closed form the policies that achieve the team optimum solution and the Nash equilibrium solution and we further investigate them numerically. ii) In the second example, two preemptive priority classes of

jobs share two processors. The objectives are to minimize the expected job delays. We give in closed form the policy that achieves the Stackelberg equilibrium solution and we further investigate it numerically. iii) In the third example, two classes of jobs share two servers. Packets from the first class may be queued waiting service in front of server, while packets from the other class are dropped when there are more than a threshold jobs into the system. Then, the first class wants to minimize its expected delay, while the other class wants to minimize its blocking probability. We find the policy that achieves the Nash equilibrium solution and we further investigate it numerically.

4.1 Team Optimal Solution

In this section, we formulate the joint load sharing, routing and congestion control problem in distributed systems as a Pareto game among cooperative classes.

Customers of different classes cooperate in using the resources of the distributed system for the social welfare. The behavior of each class is similar to that of any other class, i.e. to operate optimally for the average job. Stadler [458] and Dauer & Stadler [123] survey research on vector optimization.

Next, we give the definition for a Pareto optimal solution [27], for the joint load sharing, routing and congestion control problem on the path flows.

Definition:

A vector $(\mathbf{\Phi^*}, \mathbf{\Psi^*}) \in (\mathbf{RC}, \mathbf{LS})$ *is called a* Pareto optimal solution *for a* C*-class joint load sharing, routing and congestion control problem if and only if there exists no other vector* $(\mathbf{\Phi}, \mathbf{\Psi}) \in (\mathbf{RC}, \mathbf{LS})$ *such that*

$$J^c(\mathbf{\Phi}, \mathbf{\Psi}) \leq J^c(\mathbf{\Phi^*}, \mathbf{\Psi^*}) \quad \forall\, (\mathbf{\Phi}, \mathbf{\Psi}) \in (\mathbf{RC}, \mathbf{LS})$$

with strict inequality holding for at least one class c.

Define a global cost function

$$J(\mathbf{\Phi}, \mathbf{\Psi}) = \left[\sum_c w^c * [J^c(\mathbf{\Phi}, \mathbf{\Psi})]^p \right]^{1/p}$$

where $1 \leq p < \infty$, $\quad \sum_{c=1}^{C} w^c = 1, \quad w^c \geq 0 \quad \forall\, c.$

For $p \rightarrow \infty$, we have a *minimax* problem [122], since the cost function becomes

$$J(\mathbf{\Phi}, \mathbf{\Psi}) = \max_c \{w^c * J^c(\mathbf{\Phi}, \mathbf{\Psi})\}$$

Another problem formulation is

$$\min_{\epsilon, \mathbf{\Phi}, \mathbf{\Psi}} \ \epsilon$$

such that

$$w^c * J^c(\mathbf{\Phi}, \mathbf{\Psi}) \leq \epsilon \quad \forall\, c$$

Furthermore, another problem formulation is

$$\min_{\mathbf{\Phi}, \mathbf{\Psi}} \ J(\mathbf{\Phi}, \mathbf{\Psi})$$

such that

$$J^c(\mathbf{\Phi}, \mathbf{\Psi}) \leq \hat{J}^c(\mathbf{\Phi}, \mathbf{\Psi}) \quad \forall\, c$$

where $\hat{J}^c$ is the maximum acceptable value for the cost function J^c.

Next, we give the definition for a team optimal solution [27], for the joint load sharing, routing and congestion control problem on the path flows.

Definition:

A vector $(\mathbf{\Phi}^*, \mathbf{\Psi}^*) \in (\mathbf{RC}, \mathbf{LS})$ *is called a team-optimal solution for a* C*-class joint load sharing, routing and congestion control problem if and only if*

$$J(\mathbf{\Phi}^*, \mathbf{\Psi}^*) \ \leq J(\mathbf{\Phi}, \mathbf{\Psi}) \quad \forall\, (\mathbf{\Phi}, \mathbf{\Psi}) \in (\mathbf{RC}, \mathbf{LS})$$

In the next sections, we develop three alternative formulations for the team optimization problem.

4.1.1 Nonlinear Programming Formulation

In this section, we formulate the cooperative load sharing, routing and congestion control problem as a Nonlinear Programming Problem (NPP). Algorithms for solving NPPs is a thoroughly investigated research area and popular algorithms may be found in books by Fiacco & McCormick [152] Zangwill [529], Murray [339], Gill & Murray [192], Bazara & Shetty [30], Fletcher [164, 165], Luenberger [311], Bertsekas & Tsitsiklis [46] among others.

Theorem :

Consider the quasi-static joint load sharing, routing and congestion control problem in distributed systems with multiple cooperative classes.
$(\mathbf{\Phi^*}, \mathbf{\Psi^*}) \in (\mathbf{RC}, \mathbf{LS})$ *is a team-optimal solution if and only if it solves the following* Nonlinear Programming Problem:

$$\begin{aligned} &\textit{minimize} && J(\mathbf{\Phi}, \mathbf{\Psi}) \\ &\textit{with respect to} && (\mathbf{\Phi}, \mathbf{\Psi}) \\ &\textit{such that} && (\mathbf{\Phi}, \mathbf{\Psi}) \in (\mathbf{RC}, \mathbf{LS}) \end{aligned}$$

Proof: It follows from the definition of a team optimal solution.

Theorem :

Consider the quasi-static joint load sharing, routing and congestion control problem in distributed systems with multiple cooperative classes.
If J is differentiable and convex in $(\mathbf{\Phi}, \mathbf{\Psi}) \in (\mathbf{RC}, \mathbf{LS})$,
then $(\mathbf{\Phi^*}, \mathbf{\Psi^*}) \in (\mathbf{RC}, \mathbf{LS})$ *is a team-optimal solution if and only if it satisfies the* Karush-Kuhn-Tucker conditions:

$$\left[\frac{\partial J(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{o[sd]}} - Q^c_{[sd]}\right] * \phi^{c*}_{o[sd]} = 0 \quad \forall\ [sd] \in \mathbf{SD}^c,\ c$$

$$\left[\frac{\partial J(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{\pi[sd]}} - Q^c_{[sd]}\right] * \phi^{c*}_{\pi[sd]} = 0 \quad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c, c$$

$$\left[\frac{\partial J(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \psi^c_{[sd]}} - Q^c_{[s.]}\right] * \psi^{c*}_{[sd]} = 0 \quad \forall\ [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c,\ c$$

$$\frac{\partial J(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{o[sd]}} - Q^c_{[sd]} \geq 0 \qquad \forall\ [sd] \in \mathbf{SD}^c,\ c$$

$$\frac{\partial J(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{\pi[sd]}} - Q^c_{[sd]} \geq 0 \qquad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c,\ c$$

$$\frac{\partial J(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \psi^c_{[sd]}} - Q^c_{[s.]} \geq 0 \qquad \forall\ [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c,\ c$$

$$\phi^{c*}_{o[sd]} + \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \phi^{c*}_{\pi[sd]} = 1 \qquad \forall\ [sd] \in \mathbf{SD}^c,\ c$$

$$\sum_{[.d] \in \mathbf{D}^c_{[s.]}} \psi^{c*}_{[sd]} = 1 \qquad \forall\ [s.] \in \mathbf{S}^c,\ c$$

$$\phi^{c*}_{o[sd]},\ \phi^{c*}_{\pi[sd]} \geq 0 \qquad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c,\ c$$

$$\psi^{c*}_{[sd]} \geq 0 \qquad \forall\ [.d] \in \mathbf{D}^c_s,\ [s.] \in \mathbf{S}^c,\ c$$

Proof: The Lagrangian is

$$L = J + \sum_c \sum_{[sd] \in \mathbf{SD}^c} Q^c_{[sd]} * \left[1 - \phi^c_{o[sd]} - \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \phi^c_{\pi[sd]}\right] +$$

$$+\sum_{c}\sum_{[s.]\in\mathbf{S}^c} Q^c_s * \left[1 - \sum_{d\in\mathbf{D}_{[s.]}} \psi^c_{[sd]}\right]$$

with $\phi^c_{o[sd]},\ \phi^c_{\pi[sd]},\ \psi^c_{[sd]} \geq 0 \quad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c,\ c$

The global cost function J is convex in $(\mathbf{\Phi}, \mathbf{\Psi}) \in (\mathbf{RC}, \mathbf{LS})$, so the Karush-Kuhn-Tucker necessary conditions are also sufficient:

$$\frac{\partial L(\mathbf{\Phi}^*, \mathbf{\Psi}^*, \mathbf{Q})}{\partial \phi^c_{o[sd]}} * \phi^{c*}_{o[sd]} = 0 \ \Rightarrow\ [\frac{\partial J(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{o[sd]}} - Q^c_{[sd]}] * \phi^{c*}_{o[sd]} = 0$$
$$\forall\ [sd] \in \mathbf{SD}^c,\ c$$

$$\frac{\partial L(\mathbf{\Phi}^*, \mathbf{\Psi}^*, \mathbf{Q})}{\partial \phi^c_{\pi[sd]}} * \phi^{c*}_{\pi[sd]} = 0 \ \Rightarrow\ [\frac{\partial J(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{\pi[sd]}} - Q^c_{[sd]}] * \phi^{c*}_{\pi[sd]} = 0$$
$$\forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c, c$$

$$\frac{\partial L(\mathbf{\Phi}^*, \mathbf{\Psi}^*, \mathbf{Q})}{\partial \psi^c_{[sd]}} * \psi^{c*}_{[sd]} = 0 \ \Rightarrow\ [\frac{\partial J(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \psi^c_{[sd]}} - Q^c_{[s.]}] * \psi^{c*}_{[sd]} = 0$$
$$\forall\ [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c,\ c$$

$$\frac{\partial L(\mathbf{\Phi}^*, \mathbf{\Psi}^*, \mathbf{Q})}{\partial \phi^c_{o[sd]}} \geq 0 \ \Rightarrow\ \frac{\partial J(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{o[sd]}} - Q^c_{[sd]} \geq 0$$
$$\forall\ [sd] \in \mathbf{SD}^c,\ c$$

$$\frac{\partial L(\mathbf{\Phi}^*, \mathbf{\Psi}^*, \mathbf{Q})}{\partial \phi^c_{\pi[sd]}} \geq 0 \ \Rightarrow\ \frac{\partial J(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{\pi[sd]}} - Q^c_{[sd]} \geq 0$$
$$\forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c,\ c$$

$$\frac{\partial L(\mathbf{\Phi}^*, \mathbf{\Psi}^*, \mathbf{Q})}{\partial \psi^c_{[sd]}} \geq 0 \ \Rightarrow\ \frac{\partial J(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \psi^c_{[sd]}} - Q^c_{[s.]} \geq 0$$
$$\forall\ [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c,\ c$$

$$\frac{\partial L(\mathbf{\Phi}^*, \mathbf{\Psi}^*, \mathbf{Q})}{\partial Q^c_{[sd]}} \geq 0 \;\Rightarrow\; \phi^{c*}_{o[sd]} + \sum_{\pi[sd]\in \mathbf{\Pi}^c_{[sd]}} \phi^{c*}_{\pi[sd]} = 1 \quad \forall\, [sd] \in \mathbf{SD}^c,\ c$$

$$\frac{\partial L(\mathbf{\Phi}^*, \mathbf{\Psi}^*, \mathbf{Q})}{\partial Q^c_s} = 0 \;\Rightarrow\; \sum_{[.d]\in \mathbf{D}^c_{[s.]}} \psi^{c*}_{[sd]} = 1 \qquad \forall\, [s.] \in \mathbf{S}^c,\ c$$

$$\phi^{c*}_{o[sd]},\ \phi^{c*}_{\pi[sd]} \geq 0 \qquad \forall\, \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c,\ c$$

$$\psi^{c*}_{[sd]} \geq 0 \qquad \forall\, [.d] \in \mathbf{D}^c_s,\ [s.] \in \mathbf{S}^c,\ c$$

□

Theorem :

Consider the quasi-static joint load sharing, routing and congestion control problem in distributed systems with multiple cooperative classes.
If J is differentiable and convex in $(\mathbf{\Phi}, \mathbf{\Psi}) \in (\mathbf{RC}, \mathbf{LS})$,
then $(\mathbf{\Phi}^*, \mathbf{\Psi}^*) \in (\mathbf{RC}, \mathbf{LS})$ is a team-optimal solution if and only if

congestion control

$$\phi^{c*}_{o[sd]} > 0 \ only\ if \quad \frac{\partial J(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{o[sd]}} \leq \frac{\partial J(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{\pi[sd]}}$$

$$\phi^{c*}_{o[sd]} = 0 \qquad o.w.$$

$$\forall\, \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c,\ c$$

routing

$$\phi^{c*}_{\pi[sd]} > 0 \ only \ if \quad \frac{\partial J(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{\pi[sd]}} = \min \left\{ \frac{\partial J(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{o[sd]}}, \min_{p[sd]} \frac{\partial J(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{p[sd]}} \right\}$$

$$\phi^{c*}_{\pi[sd]} = 0 \qquad o.w.$$

$$\forall \ \pi[sd] \in \mathbf{\Pi}^c_{[sd]}, \ [sd] \in \mathbf{SD}^c, \ c$$

load sharing

$$\psi^{c*}_{[sd]} > 0 \ only \ if \quad \frac{\partial J(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \psi^c_{[sd]}} \leq \frac{\partial J(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{[sd']}}$$

$$\psi^{c*}_{[sd]} = 0 \qquad o.w.$$

$$\forall \ [sd'], \ [sd] \in \mathbf{SD}^c, \ c$$

$$\phi^{c*}_{o[sd]} + \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \phi^{c*}_{\pi[sd]} = 1 \quad \forall \ [sd] \in \mathbf{SD}^c, \ c$$

$$\sum_{[.d] \in \mathbf{D}^c_{[s.]}} \psi^{c*}_{[sd]} = 1 \qquad \forall \ [s.] \in \mathbf{S}^c, \ c$$

Proof: These conditions follow directly from the Karush-Kuhn-Tucker conditions. □.

Theorem : *existence*

Consider the quasi-static joint load sharing, routing and congestion control problem in distributed systems with multiple cooperative classes.
If J is continuously differentiable in $(\mathbf{\Phi}, \mathbf{\Psi})$ and convex in $(\mathbf{\Phi}, \mathbf{\Psi}) \in (\mathbf{RC}, \mathbf{LS})$, then there exists a team-optimal solution.

Proof: The constraint sets $\phi^c_{o[sd]} + \sum_{\pi[sd]\in\mathbf{\Pi}^c_{[sd]}} \phi^c_{\pi[sd]} = 1,\ \phi^c_{o[sd]},\ \phi^c_{\pi[sd]} \geq 0 \quad \forall\, \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c,\ c$ and $\sum_{[.d]\in\mathbf{D}_{[s.]}} \psi^c_{[sd]} = 1,\ \psi^c_{[sd]} \geq 0 \quad \forall\ [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c,\ c$ define a convex, closed and bounded set.

The cost function J is jointly continuous in all its arguments and strictly convex in $(\mathbf{\Phi}, \mathbf{\Psi}) \in (\mathbf{RC}, \mathbf{LS})$. Therefore, there exists a team-optimal solution.□

Theorem : *uniqueness*

Consider the quasi-static join load sharing, routing and congestion control problem in distributed systems with multiple cooperative classes. If J is twice continuously differentiable with respect to $\mathbf{\Lambda} \in \mathbf{K}$ and strictly convex in $\mathbf{\Lambda} \in \mathbf{K}$, then there exists a unique team optimum solution on the link flow space.

Proof: The constraint sets $\lambda^c_{io} + \sum_{ij\in\mathbf{O}^c_i} \lambda^c_{ij} = 1,\ \lambda_{io},\ \lambda_{ij} \geq 0 \quad \forall\ ij \in \mathbf{O}^c_i \quad \forall\ i,$ define a convex, closed and bounded set. The cost function J is twice continuously differentiable with respect to $\mathbf{\Lambda} \in \mathbf{K}$ and strictly convex in $\mathbf{\Lambda} \in \mathbf{K}$. Then there exists a unique team optimum solution on the link flow space. □

4.1.2 Nonlinear Complementarity Problem Formulation

In this section, we formulate the cooperative joint load sharing, routing and congestion control problem in distributed systems as a Nonlinear Complementarity Problem (NCP).

Karamardian [244, 245, 243] proves existence and uniqueness for a nonlinear complementarity problem. He also shows [246] the equivalence between a generalized complementarity problem and a variational inequality problem. For transportation networks, Aashtiani & Magnanti [1] formulate the traffic assignment problem as a nonlinear complementarity problem.

Define the vector $\mathbf{Z}$ of class congestion control, routing and load sharing fractions and Lagrange multipliers

$$\mathbf{Z} = \left[\ldots\ \phi^c_{o[sd]}\ \ldots\ \phi^c_{\pi[sd]}\ \ldots\ Q^c_{[sd]}\ \ldots\ \psi^c_{[sd]}\ \ldots\ Q^c_{[s.]}\ \ldots \right]^T$$

Define the vector of class derivatives of the Lagrangian with respect to the congestion control, routing and load sharing fractions and Lagrange multipliers:

$$\nabla L(\mathbf{Z}) = \left[\ldots \left(\frac{\partial J}{\partial \phi^c_{o[sd]}} - Q^c_{o[sd]} \right) \ \ldots \ \left(\frac{\partial J}{\partial \phi^c_{\pi[sd]}} - Q^c_{\pi[sd]} \right) \ \ldots \right.$$

$$\ldots \left(1 - \phi^c_{o[sd]} - \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \phi^c_{\pi[sd]} \right) \ \ldots$$

$$\left. \ldots \left(\frac{\partial J}{\partial \psi^c_{[sd]}} - Q^c_s \right) \ \ldots \left(1 - \sum_{[.d] \in \mathbf{D}^c_{[s.]}} \psi^c_{[sd]} \right) \ \ldots \right]$$

Theorem :

Consider the quasi-static joint load sharing, routing and congestion control problem in distributed systems with multiple cooperative classes.
If J is differentiable and convex in $(\mathbf{\Phi}, \mathbf{\Psi}) \in (\mathbf{RC}, \mathbf{LS})$,
then $(\mathbf{\Phi}^*, \mathbf{\Psi}^*) \in (\mathbf{RC}, \mathbf{LS})$ is a team-optimal solution if and only if it solves the following Nonlinear Complementarity Problem:

$$\nabla L(\mathbf{Z}^*) * \mathbf{Z}^* = 0$$
$$\nabla L(\mathbf{Z}^*) \geq 0$$
$$\mathbf{Z}^* \geq 0$$

Proof: After some algebraic manipulations, we find that the NCP: $\nabla L(\mathbf{Z}) * \mathbf{Z} = 0$; $\nabla L(\mathbf{Z}) \geq 0$; $\mathbf{Z} \geq 0$ with $\mathbf{Z}$ and $\nabla L(\mathbf{Z})$ as defined above, is equivalent to the Karush-Kuhn-Tucker necessary and sufficient conditions. □

Theorem : *existence*

Consider the quasi-static joint load sharing, routing and congestion control problem in distributed systems with multiple cooperative classes.
If $\nabla L(\mathbf{Z})$ is differentiable in $(\mathbf{RC}, \mathbf{LS})$
and its Jacobian matrix is strongly copositive in $(\mathbf{RC}, \mathbf{LS})$,
then there exists a team-optimal solution.

Proof: It is known [243], that the nonlinear complementarity problem $x * f(x) = 0;\ \ f(x) \geq 0;\ \ x \geq 0$ has a solution if f is differentiable in E^n_+ and its Jacobian matrix is strongly copositive in E^n_+. □

Theorem : *uniqueness*

Consider the quasi-static join load sharing, routing and congestion control problem in distributed systems with multiple cooperative classes.

If the gradient vector of the Lagrangian (on the link flow space) with respect to the link, node, computer site and rejected flows, as well as the corresponding Lagrange multipliers is continuous and strongly monotone on the space where these flows are defined,
then there exists a unique team optimal solution on the link flow space.

Proof: It is known [244], that if $f : E^n_+ \rightarrow E^n$ is continuous and strongly monotone on E^n_+, then there exists a unique $\bar{x} \in E^n$ such that $\bar{x} \geq 0;\ \ f(\bar{x}) \geq 0;\ \ \bar{x} f(\bar{x}) = 0$. □

4.1.3 Variational Inequality Formulation

In this section, we formulate the cooperative load sharing, routing and congestion control problem as a Variational Inequality Problem (VIP).

The theory of VIPs has been advanced a lot since the important work of Lions & Stampacchia [305], who prove existence and uniqueness for a variational inequality problem. Kinderlehrer & Stampacchia [253] present a thorough study of VIPs. Hlavacek, Haslinger, Necas & Lovisek [217] and Glowinski, Lions & Tremolieres [198] present algorithms for the solution of VIPs. Cottle, Giannessi & Lions [113] discuss the equivalence between NCP and VIP. For the traffic assignment problem in transportation networks, Dafermos [118] observes that the network equilibrium condition as reformulated by Smith [452] has the form of a variational inequality. She also [119] introduces an iterative scheme for the numerical solution of finite dimension variational inequalities.

Define the vector of class congestion control, routing and load sharing fractions:

$$(\mathbf{\Phi}, \mathbf{\Psi}) = \left[\ldots \phi^c_{o[sd]} \ \ldots \ \phi^c_{\pi[sd]} \ \ldots \ \psi^c_{[sd]} \ \ldots \right]^T$$

Define the vector of class derivatives of the cost function with respect to the congestion control, routing and load sharing fractions:

$$\nabla J(\mathbf{\Phi}, \mathbf{\Psi}) \ = \ \left[\ldots \frac{\partial J}{\partial \phi^c_{o[sd]}} \ldots \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \frac{\partial J}{\partial \phi^c_{\pi[sd]}} \ldots \frac{\partial J}{\partial \psi^c_{[sd]}} \ \ldots \right]$$

Theorem :

Consider the quasi-static joint load sharing, routing and congestion control problem in distributed systems with multiple cooperative classes.
If J is continuously differentiable and convex in $(\mathbf{\Phi}, \mathbf{\Psi}) \in (\mathbf{RC}, \mathbf{LS})$,
then $(\mathbf{\Phi}^*, \mathbf{\Psi}^*) \in (\mathbf{RC}, \mathbf{LS})$ is a team-optimal solution if and only if it solves the following Variational Inequality Problem:

$$\nabla J(\mathbf{\Phi}^*, \mathbf{\Psi}^*) * ((\mathbf{\Phi}, \mathbf{\Psi}) - (\mathbf{\Phi}^*, \mathbf{\Psi}^*)) \geq 0 \qquad \forall \ (\mathbf{\Phi}, \mathbf{\Psi}) \in (\mathbf{RC}, \mathbf{LS})$$

Proof: If $(\mathbf{\Phi}^{c*}, \mathbf{\Psi}^{c*})$ is a local minimum for the following minimization problem

minimize $J(\mathbf{\Phi}, \mathbf{\Psi})$

with respect to $(\mathbf{\Phi}, \mathbf{\Psi})$

such that $(\mathbf{\Phi}, \mathbf{\Psi}) \in (\mathbf{RC}, \mathbf{LS})$

and J is a continuously differentiable convex function over the nonempty convex, closed and bounded set $(\mathbf{RC}, \mathbf{LS})$, then

$$\sum_{c} \sum_{[sd] \in \mathbf{SD}^c} \left\{ \frac{\partial J(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{o[sd]}} * (\phi^c_{o[sd]} - \phi^{c*}_{o[sd]}) + \right.$$

$$+ \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \frac{\partial J(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{\pi[sd]}} * (\phi^c_{\pi[sd]} - \phi^{c*}_{\pi[sd]}) +$$

$$\left. + \frac{\partial J(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \psi^c_{[sd]}} * (\psi^c_{[sd]} - \psi^{c*}_{[sd]}) \right\} \geq 0 \qquad \forall\, (\mathbf{\Phi}, \mathbf{\Psi}) \in (\mathbf{RC}, \mathbf{LS})$$

□

Another equivalent VIP formulation is the following Theorem:

Theorem :

Consider the quasi-static joint load sharing, routing and congestion control problem in distributed systems with multiple cooperative classes.
If J is continuously differentiable and convex in $(\mathbf{\Phi}, \mathbf{\Psi}) \in (\mathbf{RC}, \mathbf{LS})$,
then $(\mathbf{\Phi}^*, \mathbf{\Psi}^*) \in (\mathbf{RC}, \mathbf{LS})$ is a team-optimal solution if and only if it solves the following Variational Inequality Problem:

$$\nabla L(\mathbf{Z}^*) * (\mathbf{Z} - \mathbf{Z}^*) \geq 0 \qquad \forall\ \mathbf{Z} > 0$$

Proof: Karamardian shows, that the NCP: $f(x^*) * x^* = 0; \quad f(x^*) \geq 0; \quad x^* > 0$
and the VIP: find x^* such that $f(x^*) * (x - x^*) \geq 0 \quad \forall x > 0$
are equivalent.□

Theorem : *existence*

Consider the quasi-static joint load sharing, routing and congestion control problem in distributed systems with multiple cooperative classes.
If $\nabla J(\mathbf{\Phi}, \mathbf{\Psi})$ is continuous on $(\mathbf{RC}, \mathbf{LS})$ and bounded,
then there exists a team-optimal solution.

Proof: For a VIP: find x^* such that $f(x^*) * (x - x^*) \geq 0 \quad \forall x \in K$, if f is continuous on K, then there exists a x^* that solves VIP. □

Theorem : *uniqueness*

Consider the quasi-static join load sharing, routing and congestion control problem in distributed systems with multiple cooperative classes.
If the gradient vector $\nabla J(\mathbf{\Lambda})$ of the cost function with respect to the link, node, computer site and rejected flows is continuous and strictly monotone,
then there exists a unique team optimal solution on the link flow space.

Proof: It is known that for the VIP:

find x^* such that $f(x^*) * (x - x^*) \geq 0 \quad \forall x \in K$

if f is continuous and strictly monotone, then there exists a solution. □

4.1.4 K-K-T for Separable Cost Function

In this section, we derive the first order necessary and sufficient conditions for a team-optimal solution on the path flows, when the cost function of each resource depends only on the flow on this resource and is convex with respect to that flow.

$$
\begin{aligned}
&\textit{minimize} \\
&J(\mathbf{\Phi}, \mathbf{\Psi}) \quad = \sum_{ij} J_{ij}(\lambda_{ij}^1, ..., \lambda_{ij}^c, ..., \lambda_{ij}^C) \; + \\
&\qquad + \sum_{i} J_{i}(\lambda_{i}^1, ..., \lambda_{i}^c, ..., \lambda_{i}^C) \; + \\
&\qquad + \sum_{[sd]} J_{[sd]}(\lambda_{o[sd]}^1, ..., \lambda_{o[sd]}^c, ..., \lambda_{o[sd]}^C) \; + \\
&\qquad + \sum_{[.d]} J_{[.d]}(\lambda_{[.d]}^1, ..., \lambda_{[.d]}^c, ..., \lambda_{[.d]}^C) \\
&\textit{with respect to} \quad (\mathbf{\Phi}, \mathbf{\Psi})
\end{aligned}
$$

$$
\textit{such that} \qquad \phi_{o[sd]}^c + \sum_{\pi[sd] \in \mathbf{\Pi}_{[sd]}^c} \phi_{\pi[sd]}^c = 1 \quad \forall \; [sd] \in \mathbf{SD}^c, \; c
$$

$$
\sum_{[.d] \in \mathbf{D}_{[s.]}^c} \psi_{[sd]}^c = 1 \quad \forall \; [s.] \in \mathbf{S}^c, \; c
$$

$$
\phi_{o[sd]}^c, \; \phi_{\pi[sd]}^c \geq 0 \quad \forall \; \pi[sd] \in \mathbf{\Pi}_{[sd]}^c, \; [sd] \in \mathbf{SD}^c, \; c
$$

$$
\psi_{[sd]}^c \geq 0 \quad \forall \; [.d] \in \mathbf{D}_{[s.]}^c, \; [s.] \in \mathbf{S}^c, \; c
$$

The Karush-Kuhn-Tucker necessary and sufficient conditions are:

$$\left[\frac{\partial J_{o[sd]}(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{o[sd]}} - Q^c_{[sd]}\right] * \phi^{c*}_{o[sd]} = 0 \quad \forall\ [sd] \in \mathbf{SD}^c,\ c$$

$$\left[\sum_{ij} \frac{\partial J_{ij}(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{\pi[sd]}} + \sum_{i} \frac{\partial J_{i}(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{\pi[sd]}} - Q^c_{[sd]}\right] * \phi^{c*}_{\pi[sd]} = 0$$

$$\forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c,\ c$$

$$\left[\sum_{ij} \frac{\partial J_{ij}(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \psi^c_{[sd]}} + \sum_{i} \frac{\partial J_{i}(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \psi^c_{[sd]}} + \right.$$

$$\left. + \frac{\partial J_{[sd]}(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \psi^c_{[sd]}} + \frac{\partial J_{[.d]}(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \psi^c_{[sd]}} - Q^c_{[s.]}\right] * \psi^{c*}_{[sd]} = 0$$

$$\forall\ [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c,\ c$$

$$\frac{\partial J_{[sd]}(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{o[sd]}} - Q^c_{[sd]} \geq 0 \quad \forall\ [sd] \in \mathbf{SD}^c,\ c$$

$$\sum_{ij} \frac{\partial J_{ij}(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{\pi[sd]}} + \sum_{i} \frac{\partial J_{i}(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{\pi[sd]}} - Q^c_{[sd]} \geq 0$$

$$\forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c,\ c$$

$$\sum_{ij} \frac{\partial J_{ij}(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \psi^c_{[sd]}} + \sum_{i} \frac{\partial J_{i}(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \psi^c_{[sd]}} +$$

$$+ \frac{\partial J_{[sd]}(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \psi^c_{[sd]}} + \frac{\partial J_{[.d]}(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \psi^c_{[sd]}} - Q^c_{[s.]} \geq 0$$

$$\forall\ [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c,\ c$$

$$\phi^{c*}_{o[sd]} + \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \phi^{c*}_{\pi[sd]} = 1 \quad \forall\ [sd] \in \mathbf{SD}^c,\ c$$

$$\sum_{[.d] \in \mathbf{D}^c_{[s.]}} \psi^{c*}_{[sd]} = 1$$

$$\forall\ [s.] \in \mathbf{S}^c,\ c$$

$$\phi^{c*}_{o[sd]},\ \phi^{c*}_{\pi[sd]} \geq 0 \qquad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c,\ c$$

$$\psi^{c*}_{[sd]} \geq 0 \qquad \forall\ [.d] \in \mathbf{D}^c_{s},\ [s.] \in \mathbf{S}^c,\ c$$

The partial derivatives of the cost function $J(\mathbf{\Phi}, \mathbf{\Psi})$ with respect to the path fractions $\phi^c_{\pi[sd]}$ can be written with respect to the link flows λ^c_{ij} and node flows λ^c_i:

$$\frac{\partial J_{ij}(\mathbf{\Phi}, \mathbf{\Psi})}{\partial \phi^c_{\pi[sd]}} = \frac{\partial J_{ij}(\mathbf{\Lambda}_{ij})}{\partial \lambda^c_{ij}} * \frac{\partial \lambda^c_{ij}}{\partial \phi^c_{\pi[sd]}}$$

$$= \frac{\partial J_{ij}(\mathbf{\Lambda}_{ij})}{\partial \lambda^c_{ij}} * (\gamma^c_{[sd]} + \gamma^c_{[s.]} * \psi^c_{[sd]}) * 1_{ij \in \pi[sd]}$$

$$\frac{\partial J_i(\mathbf{\Phi}, \mathbf{\Psi})}{\partial \phi^c_{\pi[sd]}} = \frac{\partial J_i(\mathbf{\Lambda}_i)}{\partial \lambda^c_i} * \frac{\partial \lambda^c_i}{\partial \phi^c_{\pi[sd]}}$$

$$= \frac{\partial J_i(\mathbf{\Lambda}_i)}{\partial \lambda^c_i} * (\gamma^c_{[sd]} + \gamma^c_{[s.]} * \psi^c_{[sd]}) * 1_{i \in \pi[sd]}$$

$$\frac{\partial J_{[sd]}(\mathbf{\Phi}, \mathbf{\Psi})}{\partial \phi^c_{o[sd]}} = \frac{\partial J_{[sd]}(\mathbf{\Lambda}_{o[sd]})}{\partial \lambda^c_{o[sd]}} * \frac{\partial \lambda^c_{o[sd]}}{\partial \phi^c_{o[sd]}}$$

$$= \frac{\partial J_{[sd]}(\mathbf{\Lambda}_{o[sd]})}{\partial \lambda^c_{o[sd]}} * (\gamma^c_{[sd]} + \gamma^c_{[s.]} * \psi^c_{[sd]})$$

$$\frac{\partial J_{ij}(\mathbf{\Phi},\mathbf{\Psi})}{\partial \psi^c_{[sd]}} = \frac{\partial J_{ij}(\mathbf{\Lambda}_{ij})}{\partial \lambda^c_{ij}} * \frac{\partial \lambda^c_{ij}}{\partial \psi^c_{[sd]}}$$

$$= \sum_{\pi[sd]} \frac{\partial J_{ij}(\mathbf{\Lambda}_{ij})}{\partial \lambda^c_{ij}} * \gamma^c_{[s.]} * \phi^c_{\pi[sd]} * 1_{ij \in \pi[sd]}$$

$$\frac{\partial J_i(\mathbf{\Phi},\mathbf{\Psi})}{\partial \psi^c_{[sd]}} = \frac{\partial J_i(\mathbf{\Lambda}_i)}{\partial \lambda^c_i} * \frac{\partial \lambda^c_i}{\partial \psi^c_{[sd]}}$$

$$= \sum_{\pi[sd]} \frac{\partial J_i(\mathbf{\Lambda}_i)}{\partial \lambda^c_i} * \gamma^c_{[s.]} * \phi^c_{\pi[sd]} * 1_{i \in \pi[sd]}$$

$$\frac{\partial J_{[sd]}(\mathbf{\Phi},\mathbf{\Psi})}{\partial \psi^c_{[sd]}} = \frac{\partial J_{[sd]}(\mathbf{\Lambda}_{o[sd]})}{\partial \lambda^c_{o[sd]}} * \frac{\partial \lambda^c_{o[sd]}}{\partial \psi^c_{[sd]}}$$

$$= \frac{\partial J_{[sd]}(\mathbf{\Lambda}_{o[sd]})}{\partial \lambda^c_{o[sd]}} * \gamma^c_{[s.]} * \phi^c_{o[sd]}$$

$$\frac{\partial J_{[.d]}(\mathbf{\Phi},\mathbf{\Psi})}{\partial \psi^c_{[sd]}} = \frac{\partial J_{[.d]}(\mathbf{\Lambda}_{[.d]})}{\partial \lambda^c_{[.d]}} * \frac{\partial \lambda^c_{[.d]}}{\partial \psi^c_{[sd]}}$$

$$= \frac{\partial J_{[.d]}(\mathbf{\Lambda}_{[.d]})}{\partial \lambda^c_{[.d]}} * \gamma^c_{[s.]}$$

Then the Karush-Kuhn-Tucker conditions become:

$$\left[\frac{\partial J_{o[sd]}(\mathbf{\Lambda}^{*}_{o[sd]})}{\partial \lambda^{c}_{o[sd]}} * (\gamma^{c}_{[sd]} + \gamma^{c}_{[s.]} * \psi^{c*}_{[sd]}) - Q^{c}_{[sd]}\right] * \phi^{c*}_{o[sd]} = 0 \quad \forall\ [sd] \in \mathbf{SD}^{c},\ c$$

$$\left[\sum_{ij} \frac{\partial J_{ij}(\mathbf{\Lambda}^{*}_{ij})}{\partial \lambda^{c}_{ij}} * (\gamma^{c}_{[sd]} + \gamma^{c}_{[s.]} * \psi^{c*}_{[sd]}) * 1_{ij \in \pi[sd]} +\right.$$

$$\left. + \sum_{i} \frac{\partial J_{i}(\mathbf{\Lambda}^{*}_{i})}{\partial \lambda^{c}_{i}} * (\gamma^{c}_{[sd]} + \gamma^{c}_{[s.]} * \psi^{c*}_{[sd]}) * 1_{i \in \pi[sd]} - Q^{c}_{[sd]}\right] * \phi^{c*}_{\pi[sd]} = 0$$

$$\forall\ \pi[sd] \in \mathbf{\Pi}^{c}_{[sd]},\ [sd] \in \mathbf{SD}^{c},\ c$$

$$\left[\sum_{\pi[sd]} \sum_{ij} \frac{\partial J_{ij}(\mathbf{\Lambda}^{*}_{ij})}{\partial \lambda^{c}_{ij}} * \gamma^{c}_{[s.]} * \phi^{c*}_{\pi[sd]} * 1_{ij \in \pi[sd]} +\right.$$

$$+ \sum_{\pi[sd]} \sum_{i} \frac{\partial J_{i}(\mathbf{\Lambda}^{*}_{i})}{\partial \lambda^{c}_{i}} * \gamma^{c}_{[s.]} * \phi^{c*}_{\pi[sd]} * 1_{i \in \pi[sd]} +$$

$$\left. + \frac{\partial J_{[sd]}(\mathbf{\Lambda}^{*}_{[sd]})}{\partial \lambda^{c}_{o[sd]}} * \gamma^{c}_{[s.]} * \phi^{c*}_{o[sd]} + \frac{\partial J_{[.d]}(\mathbf{\Lambda}^{*}_{[.d]})}{\partial \lambda^{c}_{[.d]}} * \gamma^{c}_{[s.]} - Q^{c}_{[s.]}\right] * \psi^{c*}_{[sd]} = 0$$

$$\forall\ [.d] \in \mathbf{D}^{c}_{[s.]},\ [s.] \in \mathbf{S}^{c},\ c$$

$$\frac{\partial J_{[sd]}(\mathbf{\Lambda}^*_{[sd]})}{\partial \lambda^c_{o[sd]}} * (\gamma^c_{[sd]} + \gamma^c_{[s.]} * \psi^{c*}_{[sd]}) - Q^c_{[sd]} \geq 0 \quad \forall\ [sd] \in \mathbf{SD}^c,\ c$$

$$\sum_{ij} \frac{\partial J_{ij}(\mathbf{\Lambda}^*_{ij})}{\partial \lambda^c_{ij}} * (\gamma^c_{[sd]} + \gamma^c_{[s.]} * \psi^{c*}_{[sd]}) * 1_{ij \in \pi[sd]} +$$

$$+ \sum_{i} \frac{\partial J_{i}(\mathbf{\Lambda}^*_{i})}{\partial \lambda^c_{i}} * (\gamma^c_{[sd]} + \gamma^c_{[s.]} * \psi^{c*}_{[sd]}) * 1_{i \in \pi[sd]} - Q^c_{[sd]} \geq 0$$

$$\forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c,\ c$$

$$\sum_{\pi[sd]} \sum_{ij} \frac{\partial J_{ij}(\mathbf{\Lambda}^*_{ij})}{\partial \lambda^c_{ij}} * \gamma^c_{[s.]} * \phi^{c*}_{\pi[sd]} * 1_{ij \in \pi[sd]} +$$

$$+ \sum_{\pi[sd]} \sum_{i} \frac{\partial J_{i}(\mathbf{\Lambda}^*_{i})}{\partial \lambda^c_{i}} * \gamma^c_{[s.]} * \phi^{c*}_{\pi[sd]} * 1_{i \in \pi[sd]} +$$

$$+ \frac{\partial J_{[sd]}(\mathbf{\Lambda}^*_{[sd]})}{\partial \lambda^c_{o[sd]}} * \gamma^c_{[s.]} * \phi^{c*}_{o[sd]} + \frac{\partial J_{[.d]}(\mathbf{\Lambda}^*_{[.d]})}{\partial \lambda^c_{[.d]}} * \gamma^c_{[s.]} - Q^c_{[s.]} \geq 0$$

$$\forall\ [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c,\ c$$

$$\phi^{c*}_{o[sd]} + \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \phi^{c*}_{\pi[sd]} = 1 \quad \forall\ [sd] \in \mathbf{SD}^c,\ c$$

$$\sum_{[.d] \in \mathbf{D}^c_{[s.]}} \psi^{c*}_{[sd]} = 1 \qquad \forall\ [s.] \in \mathbf{S}^c,\ c$$

$$\phi^{c*}_{o[sd]},\ \phi^{c*}_{\pi[sd]} \geq 0 \qquad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c,\ c$$

$$\psi^{c*}_{[sd]} \geq 0 \qquad \forall\ [.d] \in \mathbf{D}^c_{s},\ [s.] \in \mathbf{S}^c,\ c$$

Next, for each class c, we define the length for the rejected flow $[sd]$, the length for the path $\pi[sd]$ and the length for the source-destination pair $[sd]$:

$$l^{c,team}_{o[sd]} = \frac{\partial J_{o[sd]}(\mathbf{\Lambda}_{o[sd]})}{\partial \lambda^c_{o[sd]}} * (\gamma^c_{[sd]} + \gamma^c_{[s.]} * \psi^c_{[sd]}) .$$

$$l^{c,team}_{\pi[sd]} = \sum_{ij} \frac{\partial J_{ij}(\mathbf{\Lambda}_{ij})}{\partial \lambda^c_{ij}} * (\gamma^c_{[sd]} + \gamma^c_{[s.]} * \psi^c_{[sd]}) * 1_{ij \in \pi[sd]} +$$

$$+ \sum_{i} \frac{\partial J_{i}(\mathbf{\Lambda}_{i})}{\partial \lambda^c_{i}} * (\gamma^c_{[sd]} + \gamma^c_{[s.]} * \psi^c_{[sd]}) * 1_{i \in \pi[sd]}$$

$$l^{c,team}_{[sd]} = \sum_{\pi[sd]} \sum_{ij} \frac{\partial J_{ij}(\mathbf{\Lambda}_{ij})}{\partial \lambda^c_{ij}} * \gamma^c_{[s.]} * \phi^c_{\pi[sd]} * 1_{ij \in \pi[sd]} +$$

$$+ \sum_{\pi[sd]} \sum_{i} \frac{\partial J_{i}(\mathbf{\Lambda}_{i})}{\partial \lambda^c_{i}} * \gamma^c_{[s.]} * \phi^c_{\pi[sd]} * 1_{i \in \pi[sd]} +$$

$$+ \frac{\partial J_{[sd]}(\mathbf{\Lambda}_{[sd]})}{\partial \lambda^c_{o[sd]}} * \gamma^c_{[s.]} * \phi^c_{o[sd]} + \frac{\partial J_{[.d]}(\mathbf{\Lambda}_{[.d]})}{\partial \lambda^c_{[.d]}} * \gamma^c_{[s.]}$$

External arriving flow at a source is assigned to the destination that has the minimum length from the source. However, this flow may be rejected if the length of rejecting it is less than the lengths of the paths to its destination. If it is accepted, then it is routed to its destination via the minimum length path.

In the next section, we will derive the same conditions by an alternative way, and we shall state the above ideas more formally.

4.1.5 V.I. for Separable Cost Functions

Equivalently, class c minimizes the global cost function J with respect to its load sharing, routing, and congestion control fractions:

$$
\begin{aligned}
&minimize_{\epsilon} \\
&J(\mathbf{\Phi}, \mathbf{\Psi}) \qquad = \sum_{ij} J_{ij}(\lambda_{ij}^{1}, ..., \lambda_{ij}^{c}, ..., \lambda_{ij}^{C}) + \\
&\qquad + \sum_{i} J_{i}(\lambda_{i}^{1}, ..., \lambda_{i}^{c}, ..., \lambda_{i}^{C}) + \\
&\qquad + \sum_{[sd]} J_{[sd]}(\lambda_{o[sd]}^{1}, ..., \lambda_{o[sd]}^{c}, ..., \lambda_{o[sd]}^{C}) + \\
&\qquad + \sum_{[.d]} J_{[.d]}(\lambda_{[.d]}^{1}, ..., \lambda_{[.d]}^{c}, ..., \lambda_{[.d]}^{C})
\end{aligned}
$$

$$
\begin{aligned}
&with\ respect\ to \quad \mathbf{\Phi} \\
&such\ that \qquad \phi_{o[sd]}^{c} + \sum_{\pi[sd] \in \mathbf{\Pi}_{[sd]}^{c}} \phi_{\pi[sd]}^{c} = 1 \quad \forall\ [sd] \in \mathbf{SD}^{c},\ c \\
&\qquad\qquad \phi_{o[sd]}^{c},\ \phi_{\pi[sd]}^{c} \geq 0 \quad \forall\ \pi[sd] \in \mathbf{\Pi}_{[sd]}^{c},\ [sd] \in \mathbf{SD}^{c},\ c
\end{aligned}
$$

The necessary and sufficient optimality conditions are [311]

$$
\begin{aligned}
&\sum_{c} \sum_{[sd] \in \mathbf{SD}^{c}} \left\{ \frac{\partial J(\mathbf{\Phi}^{*}, \mathbf{\Psi}^{*})}{\partial \phi_{o[sd]}^{c}} * (\phi_{o[sd]}^{c} - \phi_{o[sd]}^{c*}) + \right. \\
&\qquad \left. + \sum_{\pi[sd] \in \mathbf{\Pi}_{[sd]}^{c}} \frac{\partial J(\mathbf{\Phi}^{*}, \mathbf{\Psi}^{*})}{\partial \phi_{\pi[sd]}^{c}} * (\phi_{\pi[sd]}^{c} - \phi_{\pi[sd]}^{c*}) \right\} \geq 0 \quad \forall\ \mathbf{\Phi}^{c} \in \mathbf{RC}^{c} \\
&such\ that \\
&\qquad \phi_{o[sd]}^{c*} + \sum_{\pi[sd] \in \mathbf{\Pi}_{[sd]}^{c}} \phi_{\pi[sd]}^{c*} = 1 \quad \forall\ [sd] \in \mathbf{SD}^{c}, \quad c
\end{aligned}
$$

$$\phi^{c*}_{o[sd]},\ \phi^{c*}_{\pi[sd]} \geq 0 \qquad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]}, \quad [sd] \in \mathbf{SD}^c, \quad c$$

We can decompose these conditions for each class c, for each source-destination pair $[sd] \in \mathbf{SD}^c$

$$\frac{\partial J(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{o[sd]}} * (\phi^c_{o[sd]} - \phi^{c*}_{o[sd]}) + \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \frac{\partial J(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{\pi[sd]}} * (\phi^c_{\pi[sd]} - \phi^{c*}_{\pi[sd]}) \geq 0 \ \ \forall\ \mathbf{\Phi}^c \in \mathbf{RC}^c$$

$$\phi^{c*}_{o[sd]} + \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \phi^{c*}_{\pi[sd]} = 1$$

$$\phi^{c*}_{o[sd]},\ \phi^{c*}_{\pi[sd]} \geq 0 \qquad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]}$$

Therefore, there must be flow only on paths where the first derivative of the cost function $J(\mathbf{\Phi}^*, \mathbf{\Psi}^*)$ with respect to the path routing fractions $\phi^c_{\pi[sd]}$ is minimum

$$\phi^{c*}_{\pi[sd]} > 0 \quad only \quad if \quad \frac{\partial J(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{\pi[sd]}} \leq \frac{\partial J(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{o[sd]}}$$

$$and \qquad \frac{\partial J(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{\pi[sd]}} \leq \frac{\partial J(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{k[sd]}}$$

$$\forall\ k[sd] \neq \pi[sd], \qquad k[sd] \in \mathbf{\Pi}^c_{[sd]}, \qquad \pi[sd] \in \mathbf{\Pi}^c_{[sd]}, \quad c$$

Also, flow is not admitted into the network only if the first derivative of the cost function $J(\mathbf{\Phi}^*, \mathbf{\Psi}^*)$ with respect to the rejection fraction $\phi^c_{o[sd]}$ is minimum:

$$\phi^{c*}_{o[sd]} > 0 \quad only \quad if \quad \frac{\partial J(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{o[sd]}} \leq \frac{\partial J(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{k[sd]}} \qquad \forall\ k[sd] \in \mathbf{\Pi}^c_{[sd]}$$

Theorem : *Routing*

There must be flow only on minimum length paths:

$$\phi^{c*}_{\pi[sd]} > 0 \quad only \quad if \qquad l^{c,team*}_{\pi[sd]} = \min\{ l^{c,team*}_{o[sd]} , \min_{p[sd]}\{ l^{c,team*}_{p[sd]} \}\}$$

$$\phi^{c*}_{\pi[sd]} = 0 \qquad\qquad o.w.$$

$$\phi^{c*}_{o[sd]} + \sum_{\pi[sd]\in \mathbf{\Pi}^c_{[sd]}} \phi^{c*}_{\pi[sd]} = 1$$

$$\forall\ \pi[sd] \in \Pi^c_{[sd]}, \qquad [sd] \in \mathbf{SD}^c, \quad c$$

Theorem : *Congestion Control*

Flow is not admitted into the network only if its rejection length is less than the minimum length path to its destination:

$$\phi^{c*}_{o[sd]} > 0 \quad only \quad if \qquad l^{c,team}_{o[sd]} = \min\{ l^{c,team*}_{o[sd]} , \min_{p[sd]}\{ l^{c,team*}_{p[sd]} \}\}$$

$$\phi^{c*}_{\pi[sd]} = 0 \qquad\qquad o.w.$$

$$\phi^{c*}_{o[sd]} + \sum_{\pi[sd]\in \mathbf{\Pi}^c_{[sd]}} \phi^{c*}_{\pi[sd]} = 1 \quad \forall\ [sd] \in \mathbf{SD}^c, \quad c$$

Having found the optimum routing and congestion control decisions, we proceed to solve the load sharing problem for class c assuming that all other classes act at their optimum decisions. So, the load sharing problem for class c is

$$
\begin{aligned}
&\textit{minimize} \\
&J(\mathbf{\Phi}^*, \mathbf{\Phi}) \qquad = \sum_{ij} J_{ij}(\lambda_{ij}^1, ..., \lambda_{ij}^c, ..., \lambda_{ij}^C) + \\
&\qquad + \sum_{i} J_i(\lambda_i^1, ..., \lambda_i^c, ..., \lambda_i^C) + \\
&\qquad + \sum_{[sd]} J_{[sd]}(\lambda_{o[sd]}^1, ..., \lambda_{o[sd]}^c, ..., \lambda_{o[sd]}^C) + \\
&\qquad + \sum_{[.d]} J_{[.d]}(\lambda_{[.d]}^1, ..., \lambda_{[.d]}^c, ..., \lambda_{[.d]}^C)
\end{aligned}
$$

$$
\textit{with respect to} \quad \mathbf{\Psi}
$$

$$
\textit{such that} \qquad \sum_{[.d] \in \mathbf{D}_{[s.]}} \psi_{[sd]}^c = 1 \quad \forall\, [s.] \in \mathbf{S}^c,\ c
$$

$$
\psi_{[sd]}^c \geq 0 \quad \forall\, [.d] \in \mathbf{D}_{[s.]}^c,\ [s.] \in \mathbf{S}^c,\ c
$$

The necessary and sufficient optimality conditions are:

$$
\sum_{c} \sum_{[sd] \in \mathbf{SD}^c} \frac{\partial J(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \psi_{[sd]}^c} * (\psi_{[sd]}^c - \psi_{[sd]}^{c*}) \quad \geq 0 \ \ \forall\ \mathbf{\Psi}^c \in \mathbf{LS}^c
$$

such that

$$
\sum_{[.d] \in \mathbf{D}_{[s.]}} \psi_{[sd]}^{c*} = 1 \quad \forall\, [s.] \in \mathbf{S}^c, \quad c
$$

$$
\psi_{[sd]}^{c*} \geq 0 \quad \forall\, [.d] \in \mathbf{D}_{[s.]}, \quad [s.] \in \mathbf{S}^c, \quad c
$$

We can decompose these conditions for each class c, for each source node $[s.] \in \mathbf{S}^c$

$$\sum_{[.d] \in \mathbf{D}_{[s.]}} \frac{\partial J(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \psi^c_{[sd]}} * (\psi^c_{[sd]} - \psi^{c*}_{[sd]}) \geq 0 \quad \forall \ \mathbf{\Psi}^c \in \mathbf{LS}^c$$

such that

$$\sum_{[.d] \in \mathbf{D}_{[s.]}} \psi^{c*}_{[sd]} = 1$$

$$\psi^{c*}_{[sd]} \geq 0 \qquad \forall \ [.d] \in \mathbf{D}^c_{[s.]}$$

Therefore, there must be flow only for source-destination pairs for which the first derivative of the cost function $J(\mathbf{\Phi}^*, \mathbf{\Psi}^*)$ with respect to the destination load sharing fractions $\psi^c_{[sd]}$ is minimum:

$$\psi^{c*}_{[sd]} > 0 \quad only \quad if \quad \frac{\partial J(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \psi^c_{[sd]}} \leq \frac{\partial J(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \psi^c_{[sd']}}$$
$$\forall \ [.d'] \neq [.d], \quad [.d'], [.d] \in \mathbf{D}^c_{[s.]}, \quad c$$

Theorem : *Load Sharing*

For each source, there must be flow only to destinations whose length is minimum:

$$\psi^{c*}_{[sd]} > 0 \quad only \quad if \quad l^{c,team*}_{[sd]} = \min_{[sd']} \{ l^{c,team*}_{[sd']} \}$$

$$\psi^{c*}_{[sd]} = 0 \qquad o.w.$$

$$\sum_{[.d] \in \mathbf{S}^c_{[s.]}} \psi^{c*}_{[sd]} = 1 \quad \forall \ [.d] \in \mathbf{D}^c_{[s.]}, \ [s.] \in \mathbf{S}^c, \ c$$

Thus, in this section we have formulated and solved the load sharing, routing and congestion control problem as a team problem among multiple cooperative classes.

4.2 Nash Equilibrium Solution

In this section, we formulate the joint load sharing, routing and congestion control problem on the path flow space as a non-cooperative Nash game among competing classes.

Customers of each class try to use the resources of the distributed system for their own benefit, ignoring the inconvenience that they cause to customers from other classes. Since the behavior of each class is similar to that of any other class, i.e. to operate optimally for its customers, next we consider customers only from class c, and the effect of customers from other classes on them. When the classes are in equilibrium, no class can improve its cost by altering its decision unilaterally.

The study of n-person non-cooperative games was initiated by von Neumann & Morgenstern [504] and Nash [348, 347]. The underline ideas are based on fixed point theorems [486, 66]. Subsequently, many books have appeared by Luce & Raiffa [310], Wuirk & Saposnik [391], Rapoport [396], Arrow & Hahn [11], Shubik [443, 444, 445], Okuguchi [361], Friedman [174], Aubin [16], Ponssard [385], Krass & Hammoudeh [266], Owen [369], Kaplan [242], Moulin [336], Harsanyi & Selten [213], Jianhua [236], Aliprantis, Brown & Burkinshaw [9], Aumann [17], Guth & Kalkofen [204], among others.

Next, we also briefly survey research papers on static Nash game theory:

Rosen [403] proves existence and uniqueness of an equilibrium point for constrained Nash games with nonempty, compact and convex common strategy set and concavity assumptions on their utility functions. then he shows asymptotic stability and uses a gradient method to find the equilibrium point. Karamardian [245] uses the theory of nonlinear complementarity problems to prove existence and uniqueness of the Nash equilibrium.

Williams [512] derives conditions for stability of Nash games, Wilson [514] provides an algorithm for finding the Nash equilibrium. Rosenmuller [405] shows that for nondegenerated Nash games there exists an odd number of equilibrium points. He then provides an algorithm for computing these points.

Luenberger [312] uses l_∞ quasicontraction for sufficiency, uniqueness and stability of the Nash equilibrium. Gabay & Moulin [175] provide alternative sufficient conditions for the existence, uniqueness and stability of the Nash equilibrium. Szidarovszky and Yakowitz [474] prove existence and uniqueness of the Cournot equilibrium. Kreps [267] provides a necessary and sufficient condition for a given completely mixed strategy n-tuple to be the unique point of some finite n-person Nash games. Tesfatsion [480] establishes existence for a class of Nash games with possibly nonacyclic strategies.

Tu & Papavassilopoulos [500, 499] study the two-player linear-quadratic Gaussian Nash and Stackelberg games under explicit control sharing, implicit control sharing and static information. If one player acquires more information, then this extra information is beneficial to him, provided that it is orthogonal to both player's information. They also prove a similar results for the dynamic case, where the strategies are linear functions of the current estimates of the state.

Cohen [110] discusses several variational formulations of the Nash equilibrium problem and examines several algorithms. Li & Basar [302] obtain conditions for existence, uniqueness and stability of Nash equilibrium solutions. They also propose an iterative distributed algorithm. Chenault [98] provides alternative conditions for uniqueness of the Nash equilibrium.

In the following, we shall develop a methodology for the joint quasi-static load sharing, routing and congestion control problem based on the Nash game theory.

Next, we give the definition for a Nash equilibrium [27], for the joint load sharing, routing and congestion control problem on the path flows.

Definition:

A vector $(\mathbf{\Phi}^*, \mathbf{\Psi}^*) \in (\mathbf{RC}.\mathbf{LS})$ *is called a* Nash equilibrium *for a C-class joint load sharing, routing and congestion control problem if and only if*

$$J^1\begin{pmatrix} \mathbf{\Phi}^{1*}, \ldots, \mathbf{\Phi}^{c*}, \ldots \mathbf{\Phi}^{C*} \\ \mathbf{\Psi}^{1*}, \ldots, \mathbf{\Psi}^{c*}, \ldots, \mathbf{\Psi}^{C*} \end{pmatrix} \leq \inf_{\substack{\mathbf{\Phi}^1 \in \mathbf{RC}^1 \\ \mathbf{\Psi}^1 \in \mathbf{LS}^1}} J^1\begin{pmatrix} \mathbf{\Phi}^1, \ldots, \mathbf{\Phi}^{c*}, \ldots \mathbf{\Phi}^{C*} \\ \mathbf{\Psi}^1, \ldots, \mathbf{\Psi}^{c*}, \ldots, \mathbf{\Psi}^{C*} \end{pmatrix}$$

$$\ldots$$

$$J^c\begin{pmatrix} {}^{1*}, \ldots, \mathbf{\Phi}^{c*}, \ldots \mathbf{\Phi}^{C*} \\ \mathbf{\Psi}^{1*}, \ldots, \mathbf{\Psi}^{c*}, \ldots, \mathbf{\Psi}^{C*} \end{pmatrix} \leq \inf_{\substack{\mathbf{\Phi}^c \in \mathbf{RC}^c \\ \mathbf{\Psi}^c \in \mathbf{LS}^c}} J^c\begin{pmatrix} \mathbf{\Phi}^{1*}, \ldots, \mathbf{\Psi}^c, \ldots, \mathbf{\Psi}^{C*} \\ \mathbf{\Psi}^{1*}, \ldots \mathbf{\Phi}^c, \ldots, \mathbf{\Phi}^{C*} \end{pmatrix}$$

$$\ldots$$

$$J^C\begin{pmatrix} \mathbf{\Phi}^{1*}, \ldots, \mathbf{\Phi}^{c*}, \ldots, \mathbf{\Phi}^{C*} \\ \mathbf{\Psi}^{1*}, \ldots, \mathbf{\Psi}^{c*}, \ldots, \mathbf{\Psi}^{C*} \end{pmatrix} \leq \inf_{\substack{\mathbf{\Phi}^C \in \mathbf{RC}^C \\ \mathbf{\Psi}^C \in \mathbf{LS}^C}} J^C\begin{pmatrix} \mathbf{\Phi}^{1*}, \ldots, \mathbf{\Phi}^{c*}, \ldots, \mathbf{\Phi}^C \\ \mathbf{\Psi}^{1*}, \ldots, \mathbf{\Psi}^{c*}, \ldots, \mathbf{\Psi}^C \end{pmatrix}$$

In the next sections, we develop three alternative formulations for the Nash game problem.

4.2.1 Nonlinear Programming Problem

In this section, we formulate the non-cooperative load sharing, routing and congestion control problem as a Nonlinear Programming Problem (NPP). Algorithms for solving Nash equilibrium problems are traditional iterative algorithms that use first and possibly second derivatives. According to the iteration scheme, they also can be classified as Gauss-Seidel, Successive Overrelaxation and Jacobi iteration algorithms. Ortega & Rheinboldt [368], Rosenmuller [405], Wilson [512], Scarf [429], Luenberger [312], Gabay & Moulin [175], Li & Basar [302], Cohen [110] among others describe such iterative algorithms.

Theorem :

Consider the quasi-static joint load sharing, routing and congestion control problem in distributed systems with multiple competing classes.
$(\mathbf{\Phi}^*, \mathbf{\Psi}^*) \in (\mathbf{RC}, \mathbf{LS})$ *is a Nash equilibrium if and only if it solves the following* Nonlinear Programming Problem:

$$\begin{array}{ll} \forall\ c & \\ \textit{minimize} & J^c(\mathbf{\Phi}^{1*}, \mathbf{\Psi}^{1*}, ... \mathbf{\Phi}^c, \mathbf{\Psi}^c, ..., \mathbf{\Phi}^{C*}, \mathbf{\Psi}^{C*}) \\ \textit{with respect to} & (\mathbf{\Phi}^c, \mathbf{\Psi}^c) \\ \textit{such that} & (\mathbf{\Phi}^c, \mathbf{\Psi}^c) \in (\mathbf{RC}^c, \mathbf{LS}^c) \end{array}$$

Proof: It follows directly from the definition of the Nash equilibrium.□

Theorem :

Consider the quasi-static joint load sharing, routing and congestion control problem in distributed systems with multiple competing classes. If for each class c, J^c *is differentiable and convex in* $(\mathbf{\Phi}^c, \mathbf{\Psi}^c) \in (\mathbf{RC}^c, \mathbf{LS}^c)$, *for each fixed value of* $(\mathbf{\Phi}^1, \mathbf{\Psi}^1, ..., \mathbf{\Phi}^{c-1}, \mathbf{\Psi}^{c-1}, \mathbf{\Phi}^{c+1}, \mathbf{\Psi}^{c+1}, ..., \mathbf{\Phi}^C, \mathbf{\Psi}^C)$
$\in (\mathbf{RC}^1, \mathbf{LS}^1, ..., \mathbf{RC}^{c-1}, \mathbf{LS}^{c-1}, \mathbf{RC}^{c+1}, \mathbf{LS}^{c+1}, ..., \mathbf{RC}^C, \mathbf{LS}^C)$,
then $(\mathbf{\Phi}^*, \mathbf{\Psi}^*) \in (\mathbf{RC}, \mathbf{LS})$ *is a Nash equilibrium if and only if it satisfies the* Karush-Kuhn-Tucker conditions:

$$\left[\frac{\partial J^c(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{o[sd]}} - Q^c_{[sd]}\right] * \phi^{c*}_{o[sd]} = 0 \quad \forall\ [sd] \in \mathbf{SD}^c,\ c$$

$$\left[\frac{\partial J^c(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{\pi[sd]}} - Q^c_{[sd]}\right] * \phi^{c*}_{\pi[sd]} = 0 \quad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c, c$$

$$\left[\frac{\partial J^c(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \psi^c_{[sd]}} - Q^c_{[s.]}\right] * \psi^{c*}_{[sd]} = 0 \quad \forall\ [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c,\ c$$

$$\frac{\partial J^c(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{o[sd]}} - Q^c_{[sd]} \geq 0 \quad \forall\ [sd] \in \mathbf{SD}^c,\ c$$

$$\frac{\partial J^c(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{\pi[sd]}} - Q^c_{[sd]} \geq 0 \quad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c,\ c$$

$$\frac{\partial J^c(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \psi^c_{[sd]}} - Q^c_{[s.]} \geq 0 \quad \forall\ [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c,\ c$$

$$\phi^{c*}_{o[sd]} + \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \phi^{c*}_{\pi[sd]} = 1 \quad \forall\ [sd] \in \mathbf{SD}^c,\ c$$

$$\sum_{[.d] \in \mathbf{D}^c_{[s.]}} \psi^{c*}_{[sd]} = 1 \qquad \forall\ [s.] \in \mathbf{S}^c,\ c$$

$$\phi^{c*}_{o[sd]},\ \phi^{c*}_{\pi[sd]} \geq 0 \qquad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c,\ c$$

$$\psi^{c*}_{[sd]} \geq 0 \qquad \forall\ [.d] \in \mathbf{D}^c_s,\ [s.] \in \mathbf{S}^c,\ c$$

Proof: The Lagrangian for each class c is

$$L^c = J^c + \sum_{[sd] \in \mathbf{SD}^c} Q^c_{[sd]} * \left[1 - \phi^c_{o[sd]} - \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \phi^c_{\pi[sd]}\right] + \sum_{[s.] \in \mathbf{S}^c} Q^c_s * \left[1 - \sum_{[.d] \in \mathbf{D}^c_{[s.]}} \psi^c_{[sd]}\right]$$

with $\phi^c_{o[sd]},\ \phi^c_{\pi[sd]},\ \psi^c_{[sd]} \geq 0 \quad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c,\ c$

The cost function for each class c, J^c, is convex in $(\mathbf{\Phi}^c, \mathbf{\Psi}^c) \in (\mathbf{RC}^c, \mathbf{LS}^c)$, so the Karush-Kuhn-Tucker necessary conditions are also sufficient:

$$\frac{\partial L^c(\mathbf{\Phi}^*, \mathbf{\Psi}^*, \mathbf{Q})}{\partial \phi^c_{o[sd]}} * \phi^{c*}_{o[sd]} = 0 \;\Rightarrow\; [\frac{\partial J^c(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{o[sd]}} - Q^c_{[sd]}] * \phi^{c*}_{o[sd]} = 0$$

$$\forall\ [sd] \in \mathbf{SD}^c,\ c$$

$$\frac{\partial L^c(\mathbf{\Phi}^*, \mathbf{\Psi}^*, \mathbf{Q})}{\partial \phi^c_{\pi[sd]}} * \phi^{c*}_{\pi[sd]} = 0 \;\Rightarrow\; [\frac{\partial J^c(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{\pi[sd]}} - Q^c_{[sd]}] * \phi^{c*}_{\pi[sd]} = 0$$

$$\forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c, c$$

$$\frac{\partial L^c(\mathbf{\Phi}^*, \mathbf{\Psi}^*, \mathbf{Q})}{\partial \psi^c_{[sd]}} * \psi^{c*}_{[sd]} = 0 \;\Rightarrow\; [\frac{\partial J^c(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \psi^c_{[sd]}} - Q^c_{[s.]}] * \psi^{c*}_{[sd]} = 0$$

$$\forall\ [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c,\ c$$

$$\frac{\partial L^c(\mathbf{\Phi}^*, \mathbf{\Psi}^*, \mathbf{Q})}{\partial \phi^c_{o[sd]}} \geq 0 \;\Rightarrow\; \frac{\partial J^c(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{o[sd]}} - Q^c_{[sd]} \geq 0$$

$$\forall\ [sd] \in \mathbf{SD}^c,\ c$$

$$\frac{\partial L^c(\mathbf{\Phi}^*, \mathbf{\Psi}^*, \mathbf{Q})}{\partial \phi^c_{\pi[sd]}} \geq 0 \;\Rightarrow\; \frac{\partial J^c(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{\pi[sd]}} - Q^c_{[sd]} \geq 0$$

$$\forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c,\ c$$

$$\frac{\partial L^c(\mathbf{\Phi}^*, \mathbf{\Psi}^*. \mathbf{Q})}{\partial \psi^c_{[sd]}} \geq 0 \;\Rightarrow\; \frac{\partial J^c(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \psi^c_{[sd]}} - Q^c_{[s.]} \geq 0$$

$$\forall\ [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c,\ c$$

$$\frac{\partial L^c(\mathbf{\Phi}^*, \mathbf{\Psi}^*, \mathbf{Q})}{\partial Q^c_{[sd]}} \geq 0 \;\Rightarrow\; \phi^{c*}_{o[sd]} + \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \phi^{c*}_{\pi[sd]} = 1 \quad \forall\ [sd] \in \mathbf{SD}^c,\ c$$

$$\frac{\partial L^c(\mathbf{\Phi}^*, \mathbf{\Psi}^*, \mathbf{Q})}{\partial Q^c_s} = 0 \;\Rightarrow\; \sum_{[.d] \in \mathbf{D}^c_{[s.]}} \psi^{c*}_{[sd]} = 1 \qquad \forall\ [s.] \in \mathbf{S}^c,\ c$$

$$\phi^{c*}_{o[sd]},\ \phi^{c*}_{\pi[sd]} \geq 0 \qquad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c,\ c$$

$$\psi^{c*}_{[sd]} \geq 0 \qquad \forall\ [.d] \in \mathbf{D}^c_s,\ [s.] \in \mathbf{S}^c,\ c$$

□

Theorem :

Consider the quasi-static joint load sharing, routing and congestion control problem in distributed systems with multiple competing classes. If for each class c, J^c is differentiable and convex in $(\mathbf{\Phi}^c, \mathbf{\Psi}^c) \in (\mathbf{RC}^c, \mathbf{LS}^c)$. for each fixed value of

$(\mathbf{\Phi}^1, \mathbf{\Psi}^1, ... \mathbf{\Phi}^{c-1}, \mathbf{\Psi}^{c-1}, \mathbf{\Phi}^{c+1}, \mathbf{\Psi}^{c+1}, ..., \mathbf{\Phi}^C, \mathbf{\Psi}^C)$

$\in (\mathbf{RC}^1, \mathbf{LS}^1, ... \mathbf{RC}^{c-1}, \mathbf{LS}^{c-1}, \mathbf{RC}^{c+1}, \mathbf{LS}^{c+1}, ..., \mathbf{RC}^C, \mathbf{LS}^C)$.

then $(\mathbf{\Phi}^*, \mathbf{\Psi}^*) \in (\mathbf{RC}, \mathbf{LS})$ is a Nash equilibrium solution if and only if

congestion control

$$\phi^{c*}_{o[sd]} > 0 \ only \ if \quad \frac{\partial J^c(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{o[sd]}} \leq \frac{\partial J^c(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{\pi[sd]}}$$

$$\phi^{c*}_{o[sd]} = 0 \qquad o.w.$$

$$\forall \ \pi[sd] \in \mathbf{\Pi}^c_{[sd]}, \ [sd] \in \mathbf{SD}^c, \ c$$

routing

$$\phi^{c*}_{\pi[sd]} > 0 \ only \ if \quad \frac{\partial J^c(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{\pi[sd]}} = \min \left\{ \frac{\partial J^c(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{o[sd]}}, \min_{p[sd]} \frac{\partial J^c(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{p[sd]}} \right\}$$

$$\phi^{c*}_{\pi[sd]} = 0 \qquad o.w.$$

$$\forall \ \pi[sd] \in \mathbf{\Pi}^c_{[sd]}, \ [sd] \in \mathbf{SD}^c, \ c$$

load sharing

$$\psi^{c*}_{[sd]} > 0 \ only \ if \quad \frac{\partial J^c(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \psi^c_{[sd]}} \leq \frac{\partial J^c(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{[sd']}},$$

$$\psi^{c*}_{[sd]} = 0 \qquad o.w.$$

$$\forall \ [sd'], \ [sd] \in \mathbf{SD}^c, \ c$$

$$\phi^{c*}_{o[sd]} + \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \phi^{c*}_{\pi[sd]} = 1 \quad \forall \ [sd] \in \mathbf{SD}^c, \ c$$

$$\sum_{[.d] \in \mathbf{D}^c_{[s.]}} \psi^{c*}_{[sd]} = 1 \qquad \forall \ [s.] \in \mathbf{S}^c, \ c$$

Proof: It follows from the Karush-Kuhn-Tucker conditions.□

Theorem : *existence*

Consider the quasi-static joint load sharing, routing and congestion control problem in distributed systems with multiple competing classes.
If for each class c,
i) J^c *is continuously differentiable in* $(\mathbf{\Phi}, \mathbf{\Psi})$ *and*
ii) J^c *is (strictly) convex in* $(\mathbf{\Phi}^c, \mathbf{\Psi}^c) \in (\mathbf{RC}^C, \mathbf{LS}^C)$, *for each fixed value of*
$(\mathbf{\Phi}^1, \mathbf{\Psi}^1, ..., \mathbf{\Phi}^{c-1}, \mathbf{\Psi}^{c-1}, \mathbf{\Phi}^{c+1}, \mathbf{\Psi}^{c+1}, ..., \mathbf{\Phi}^C, \mathbf{\Psi}^C)$
$\in (\mathbf{RC}^1, \mathbf{LS}^1, ..., \mathbf{RC}^{c-1}, \mathbf{LS}^{c-1}, \mathbf{RC}^{c+1}, \mathbf{LS}^{c+1}, ..., \mathbf{RC}^C, \mathbf{LS}^C)$,
then there exists a Nash equilibrium.

Proof: The constraint sets $\phi^c_{o[sd]} + \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \phi^c_{\pi[sd]} = 1, \ \phi_{o[sd]}, \ \phi_{\pi[sd]} \geq 0 \quad \forall \ \pi[sd] \in \mathbf{\Pi}^c_{[sd]}$, $[sd]$, and $\sum_{[.d] \in \mathbf{D}^c_{[s.]}} \psi^c_{[sd]} = 1, \ \psi_{[sd]} \geq 0 \quad \forall \ [.d] \in \mathbf{D}^c_{[s.]}, \ [s.] \in \mathbf{S}^c$ define a convex, closed and bounded set.

Each cost function J^c is jointly continuous in all its arguments and (strictly) convex in $(\mathbf{\Phi}^c, \mathbf{\Psi}^c) \in (\mathbf{RC}^c, \mathbf{LS}^c)$, for each fixed value of

$(\mathbf{\Phi}^1, \mathbf{\Psi}^1, ..., \mathbf{\Phi}^{c-1}, \mathbf{\Psi}^{c-1}, \mathbf{\Phi}^{c+1}, \mathbf{\Psi}^{c+1}, ... \mathbf{\Phi}^C, \mathbf{\Psi}^C)$
$\in (\mathbf{RC}^1, \mathbf{LS}^1, ... \mathbf{RC}^{c-1}, \mathbf{LS}^{c-1}, \mathbf{RC}^{c+1}, \mathbf{LS}^{c+1}, ... \mathbf{RC}^C, \mathbf{LS}^C)$.
Therefore, there exists a Nash equilibrium.□

Theorem : *uniqueness*

Consider the quasi-static joint load sharing, routing and congestion control problem in distributed systems with multiple competing classes.
If for each class c
i) J^c *is twice continuously differentiable with respect to* $\mathbf{\Lambda}^c \in \mathbf{K}^c$,
ii) J^c *is strictly convex in* $\mathbf{\Lambda}^c \in \mathbf{K}^c$, *for each fixed value of*
$(\mathbf{\Lambda}^1, ..., \mathbf{\Lambda}^{c-1}, \mathbf{\Lambda}^{c+1}, ..., \mathbf{\Lambda}^C) \in (\mathbf{K}^1, ... \mathbf{K}^{c-1}, \mathbf{K}^{c+1}, ..., \mathbf{K}^C)$,
iii) J^c *is concave in* $(\mathbf{\Lambda}^1, ..., \mathbf{\Lambda}^{c-1}, \mathbf{\Lambda}^{c+1}, ..., \mathbf{\Lambda}^C) \in (\mathbf{K}^1, ..., \mathbf{K}^{c-1}, \mathbf{K}^{c+1}, ..., \mathbf{K}^C)$
for each fixed value of $\mathbf{\Lambda}^c \in \mathbf{K}^c$, *and*
iv) $\exists\ \mathbf{r} = [... \ r^c_{io} \ ... \ r^c_{ij} \ ...] > 0$ *such that*
$\left[... \ r^c_{io} \dfrac{\partial J^c(\mathbf{\Lambda})}{\partial \lambda^c_{io}} \ ... \ r^c_{ij} \dfrac{\partial J^c(\mathbf{\Lambda})}{\partial \lambda^c_{ij}} \ ...\right]$ *is strictly monotone in* $\mathbf{\Lambda} \in \mathbf{K}$,
then there exists a unique Nash equilibrium.

Proof: The constraint sets $\lambda^c_{io} + \sum_{ij \in \mathbf{O}^c_i} \lambda^c_{ij} = 1,\ \lambda_{io},\ \lambda_{ij} \geq 0 \quad \forall\ ij \in \mathbf{O}^c_i \quad \forall\ i,$
define a convex, closed and bounded set.

Using ii), iii) and iv) we can show that $[G(\mathbf{\Lambda}, \mathbf{r}) + G^T(\mathbf{\Lambda}, \mathbf{r})]$ is positive definite, where $G(\mathbf{\Lambda}, \mathbf{r})$ is the Jacobian of $[... \ r^c_{io} \dfrac{\partial J^c(\mathbf{\Lambda})}{\partial \lambda^c_{io}} \ ... \ r^c_{ij} \dfrac{\partial J^c(\mathbf{\Lambda})}{\partial \lambda^c_{ij}} \ ...]^T$.

It follows that $\sum_{c=1}^{C} \sum_{i \in \mathbf{N}^c} \{\ r^c_{io} J^c(\mathbf{\Lambda}) + \sum_{ij \in \mathbf{O}^c_i} r^c_{ij} J^c(\mathbf{\Lambda})\}$ is diagonally strictly convex in $\mathbf{\Lambda} \in \mathbf{K}$, for fixed $\mathbf{r} > 0$. Then there exists a unique Nash equilibrium.□

4.2.2 Nonlinear Complementarity Problem

In this section, we formulate the non-cooperative load sharing, routing and congestion control problem as a Nonlinear Complementarity Problem (NCP).

Define the vector of class congestion control, routing and load sharing fractions and Lagrange multipliers:

$$\mathbf{Z} = \left[\ldots \phi^c_{o[sd]} \ \ldots \ \phi^c_{\pi[sd]} \ \ldots \ Q^c_{[sd]} \ \ldots\ \psi^c_{[sd]} \ \ldots \ Q^c_{[s\cdot]} \ \ldots \right]^T$$

and the vector of class derivative of the Lagrangian with respect to the congestion control, routing and load sharing fractions as well as the Lagrange multipliers:

$$\begin{aligned} \nabla L(\mathbf{Z}) \ = \ & \left[\ldots \left(\frac{\partial J^c}{\partial \phi^c_{o[sd]}} - Q^c_{o[sd]} \right) \ \ldots \ \left(\frac{\partial J^c}{\partial \phi^c_{\pi[sd]}} - Q^c_{\pi[sd]} \right) \ \ldots \right. \\ & \ldots \ \left(1 - \phi^c_{o[sd]} - \textstyle\sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \phi^c_{\pi[sd]} \right) \ \ldots \\ & \left. \ldots \ \left(\frac{\partial J^c}{\partial \psi^c_{[sd]}} - Q^c_s \right) \ \ldots \ \left(1 - \sum_{[\cdot d] \in \mathbf{D}^c_{[s\cdot]}} \psi^c_{[sd]} \right) \ \ldots \right] \end{aligned}$$

Theorem :

Consider the quasi-static joint load sharing, routing and congestion control problem in distributed systems with multiple competing classes. If for each class c, J^c is differentiable and convex in $(\mathbf{\Phi}^c, \mathbf{\Psi}^c) \in (\mathbf{RC}^C, \mathbf{LS}^C)$, for each fixed value of
$(\mathbf{\Phi}^1, \mathbf{\Psi}^1, ..., \mathbf{\Phi}^{c-1}, \mathbf{\Psi}^{c-1}, \mathbf{\Phi}^{c+1}, \mathbf{\Psi}^{c+1}, ..., \mathbf{\Phi}^C, \mathbf{\Psi}^C)$
$\in (\mathbf{RC}^1, \mathbf{LS}^1, ..., \mathbf{RC}^{c-1}, \mathbf{LS}^{c-1}, \mathbf{RC}^{c+1}, \mathbf{LS}^{c+1}, ..., \mathbf{RC}^C, \mathbf{LS}^C)$,
then $(\mathbf{\Phi}^*, \mathbf{\Psi}^*) \in (\mathbf{RC}, \mathbf{LS})$ is a Nash equilibrium if and only if it solves the following Nonlinear Complementarity Problem:

$$\nabla L(\mathbf{Z}^*) * \mathbf{Z}^* = 0$$
$$\nabla L(\mathbf{Z}^*) \geq 0$$
$$\mathbf{Z}^* \geq 0$$

Proof: After some algebraic manipulations, we find that the NCP: $\nabla L(\mathbf{Z}) * \mathbf{Z} = 0; \quad \nabla L(\mathbf{Z}) \geq 0; \quad \mathbf{Z} \geq 0$ with $\mathbf{Z}$ and $\nabla L(\mathbf{Z})$ as defined above, is equivalent to the Karush-Kuhn-Tucker necessary and sufficient conditions. □

Theorem : *existence*

Consider the quasi-static joint load sharing, routing and congestion control problem in distributed systems with multiple competing classes.
If $\nabla L(\mathbf{Z})$ *is differentiable in* $(\mathbf{RC}, \mathbf{LS})$
and its Jacobian matrix is strongly copositive in $(\mathbf{RC}, \mathbf{LS})$,
then there exists a Nash equilibrium.

Proof: It is known [243], that the NCP: $x * f(x) = 0; \quad f(x) \geq 0; \quad x \geq 0$ has a solution if f is differentiable in E^n_+ and its Jacobian matrix is strongly copositive in E^n_+. □

Theorem : *uniqueness*

Consider the quasi-static join load sharing, routing and congestion control problem in distributed systems with multiple competing classes.
If the gradient vector of the Lagrangian (on the link flow space) with respect to the link, node, computer site and rejected flows, as well as the corresponding Lagrange multipliers is continuous and strongly monotone on the space where these flows are defined,
then there exists a unique Nash equilibrium on the link flow space.

Proof: It is known [244], that if $f : E^n_+ \rightarrow E^n$ is continuous and strongly monotone on E^n_+, then there exists a unique $\bar{x} \in E^n$ such that $\bar{x} \geq 0, \quad f(\bar{x}) \geq 0, \quad \bar{x} f(\bar{x}) = 0$. □

4.2.3 Variational Inequality Formulation

In this section, we formulate the non-cooperative load sharing, routing and congestion control problem as a Variational Inequality Problem (VIP).

Define the vector of class congestion control, routing and load sharing fractions,

$$(\mathbf{\Phi}, \mathbf{\Psi}) = \left[\ldots \phi^c_{o[sd]} \ldots \phi^c_{\pi[sd]} \ldots \psi^c_{[sd]} \ldots \right]^T$$

and the vector of class derivatives of the cost function with respect to the congestion control, routing and load sharing fractions:

$$\nabla J(\mathbf{\Phi}, \mathbf{\Psi}) = \left[\ldots \frac{\partial J^c}{\partial \phi^c_{o[sd]}} \ldots \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \frac{\partial J^c}{\partial \phi^c_{\pi[sd]}} \ldots \frac{\partial J^c}{\partial \psi^c_{[sd]}} \ldots \right]$$

Theorem :

Consider the quasi-static joint load sharing, routing and congestion control problem in distributed systems with multiple competing classes.

If for each class c,

J^c *is continuously differentiable and convex in* $(\mathbf{\Phi}^c, \mathbf{\Psi}^c) \in (\mathbf{RC}^C, \mathbf{LS}^C)$,

for each fixed value of $(\mathbf{\Phi}^1, \mathbf{\Psi}^1, ..., \mathbf{\Phi}^{c-1}, \mathbf{\Psi}^{c-1}, \mathbf{\Phi}^{c+1}, \mathbf{\Psi}^{c+1}, ..., \mathbf{\Phi}^C, \mathbf{\Psi}^C)$

$\in (\mathbf{RC}^1, \mathbf{LS}^1, ..., \mathbf{RC}^{c-1}, \mathbf{LS}^{c-1}, \mathbf{RC}^{c+1}, \mathbf{LS}^{c+1}, ..., \mathbf{RC}^C, \mathbf{LS}^C)$,

then $(\mathbf{\Phi}^*, \mathbf{\Psi}^*) \in (\mathbf{RC}, \mathbf{LS})$ *is a Nash equilibrium if and only if it solves the following Variational Inequality Problem:*

$$\nabla J(\mathbf{\Phi}^*, \mathbf{\Psi}^*) * ((\mathbf{\Phi}, \mathbf{\Psi}) - (\mathbf{\Phi}^*, \mathbf{\Psi}^*)) \geq 0 \qquad \forall\, (\mathbf{\Phi}, \mathbf{\Psi}) \in (\mathbf{RC}, \mathbf{LS})$$

Proof: If $(\mathbf{\Phi}^{c*}, \mathbf{\Psi}^{c*})$ is a local minimum for the following minimization problem

minimize	$J^c(\mathbf{\Phi}^{1*}, \mathbf{\Psi}^{1*}, ..., \mathbf{\Phi}^c, \mathbf{\Psi}^c, ..., \mathbf{\Phi}^{C*}, \mathbf{\Psi}^{C*})$
with respect to	$(\mathbf{\Phi}^c, \mathbf{\Psi}^c)$
such that	$(\mathbf{\Phi}^c, \mathbf{\Psi}^c) \in (\mathbf{RC}^c, \mathbf{LS}^c)$

and J^c is a continuously differentiable convex function over the nonempty convex, closed and bounded set $(\mathbf{RC}^c, \mathbf{LS}^c)$, then

$$\sum_{[sd]\in \mathbf{SD}^c} \left\{ \frac{\partial J^c(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{o[sd]}} * (\phi^c_{o[sd]} - \phi^{c*}_{o[sd]}) + \right.$$

$$+ \sum_{\pi[sd]\in \mathbf{\Pi}^c_{[sd]}} \frac{\partial J^c(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{\pi[sd]}} * (\phi^c_{\pi[sd]} - \phi^{c*}_{\pi[sd]}) +$$

$$\left. + \frac{\partial J^c(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \psi^c_{[sd]}} * (\psi^c_{[sd]} - \psi^{c*}_{[sd]}) \right\} \geq 0 \quad \forall (\mathbf{\Phi}^c, \mathbf{\Psi}^c) \in (\mathbf{RC}^c, \mathbf{LS}^c),\ c$$

Summing over all classes

$$\sum_{c} \sum_{[sd] \in \mathbf{SD}^c} \left\{ \frac{\partial J^c(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{o[sd]}} * (\phi^c_{o[sd]} - \phi^{c*}_{o[sd]}) + \right.$$

$$+ \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \frac{\partial J^c(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{\pi[sd]}} * (\phi^c_{\pi[sd]} - \phi^{c*}_{\pi[sd]}) +$$

$$\left. + \frac{\partial J^c(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \psi^c_{[sd]}} * (\psi^c_{[sd]} - \psi^{c*}_{[sd]}) \right\} \geq 0 \qquad \forall\, (\mathbf{\Phi}^c, \mathbf{\Psi}^c) \in (\mathbf{RC}^c, \mathbf{LS}^c)$$

□

Another equivalent VIP formulation is the following Theorem:

Theorem :

Consider the quasi-static joint load sharing, routing and congestion control problem in distributed systems with multiple competing classes. If for each class c, J^c is continuously differentiable and convex in $(\mathbf{\Phi}^c, \mathbf{\Psi}^c) \in (\mathbf{RC}^C, \mathbf{LS}^C)$,
for each fixed value of $(\mathbf{\Phi}^1, \mathbf{\Psi}^1, ..., \mathbf{\Phi}^{c-1}, \mathbf{\Psi}^{c-1}, \mathbf{\Phi}^{c+1}, \mathbf{\Psi}^{c+1}, ..., \mathbf{\Phi}^C, \mathbf{\Psi}^C)$
$\in (\mathbf{RC}^1, \mathbf{LS}^1, ..., \mathbf{RC}^{c-1}, \mathbf{LS}^{c-1}, \mathbf{RC}^{c+1}, \mathbf{LS}^{c+1}, ..., \mathbf{RC}^C, \mathbf{LS}^C)$,
then $(\mathbf{\Phi}^*, \mathbf{\Psi}^*) \in (\mathbf{RC}, \mathbf{LS})$ *is a Nash equilibrium if and only if it solves the following Variational Inequality Problem:*

$$\nabla L(\mathbf{Z}^*) * (\mathbf{Z} - \mathbf{Z}^*) \geq 0 \qquad \forall\, \mathbf{Z} > 0$$

Proof: Karamardian shows that the NCP: $f(x^*) * x^* = 0; \quad f(x^*) \geq 0; \quad x^* > 0$
and the VIP: find x^* such that $f(x^*) * (x - x^*) \geq 0 \quad \forall x > 0$
are equivalent.□

Theorem : *existence*

Consider the quasi-static joint load sharing, routing and congestion control problem in distributed systems with multiple competing classes.
If $\nabla J(\mathbf{\Phi}, \mathbf{\Psi})$ *is continuous on* $(\mathbf{RC}, \mathbf{LS})$ *and bounded,*
then there exists a Nash equilibrium.

Proof: For a VIP: find x^* such that $f(x^*) * (x - x^*) \geq 0 \quad \forall x \in K$, if f is continuous on K, then there exists a x^* that solves VIP. □

Theorem : *uniqueness*

Consider the quasi-static join load sharing, routing and congestion control problem in distributed systems with multiple competing classes.
If the gradient vector $\nabla J(\mathbf{\Lambda})$ *of the cost function with respect to the link, node, computer site and rejected flows is continuous and strictly monotone,*
then there exists a unique Nash equilibrium.

Proof: For the VIP:
find x^* such that $f(x^*) * (x - x^*) \geq 0 \quad \forall x \in K$
if f is continuous and strictly monotone, then there exists a solution. □

4.2.4 K-K-T for Separable Cost Function

In this section, we derive the first order necessary and sufficient conditions for a Nash equilibrium on the path flows, when the cost function of each networks resource depends only on the flow on this resource and is convex with respect to that flow.

According to the Nash equilibrium definition, each class c minimizes its cost function J^c given the optimum decisions of all other classes.

$$
\begin{aligned}
&minimize \\
&J^c\left(\begin{matrix}\mathbf{\Phi}^{1*},...,\mathbf{\Phi}^{c},...,\mathbf{\Phi}^{C*}\\ \mathbf{\Psi}^{1*},...,\mathbf{\Psi}^{c},...,\mathbf{\Psi}^{C*}\end{matrix}\right) &= \sum_{ij} J^c_{ij}(\lambda^{1*}_{ij},...,\lambda^{c}_{ij},...,\lambda^{C*}_{ij}) \;+ \\
&&+\sum_{i} J^c_{i}(\lambda^{1*}_{i},...,\lambda^{c}_{i},...,\lambda^{C*}_{i}) \;+ \\
&&+\sum_{[sd]} J^c_{[sd]}(\lambda^{1*}_{o[sd]},...,\lambda^{c}_{o[sd]},...,\lambda^{C*}_{o[sd]}) \;+ \\
&&+\sum_{d} J^c_{[.d]}(\lambda^{1*}_{[.d]},...,\lambda^{c}_{[.d]},...,\lambda^{C*}_{[.d]})
\end{aligned}
$$

$$
\textit{with respect to} \qquad (\mathbf{\Phi}^c, \mathbf{\Psi}^c)
$$

$$
\textit{such that} \qquad \phi^c_{o[sd]} + \sum_{\pi[sd]\in\mathbf{\Pi}^c_{[sd]}} \phi^c_{\pi[sd]} = 1 \quad \forall\, [sd] \in \mathbf{SD}^c
$$

$$
\sum_{[.d]\in\mathbf{D}^c_{[s.]}} \psi^c_{[sd]} = 1 \quad \forall\, [s.] \in \mathbf{S}^c
$$

$$
\phi^c_{o[sd]},\ \phi^c_{\pi[sd]} \geq 0 \quad \forall \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c
$$

$$
\psi^c_{[sd]} \geq 0 \quad \forall [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c
$$

The Karush-Kuhn-Tucker necessary and sufficient conditions are:

$$\left[\frac{\partial J^c_{[sd]}(,\mathbf{\Phi}^*,\mathbf{\Psi}^*)}{\partial \phi^c_{o[sd]}} - Q^c_{[sd]}\right] * \phi^{c*}_{o[sd]} = 0 \quad \forall\ [sd] \in \mathbf{SD}^c,\ c$$

$$\left[\sum_{ij} \frac{\partial J^c_{ij}(\mathbf{\Phi}^*,\mathbf{\Psi}^*)}{\partial \phi^c_{\pi[sd]}} + \sum_{i} \frac{\partial J^c_{i}(\mathbf{\Phi}^*,\mathbf{\Psi}^*)}{\partial \phi^c_{\pi[sd]}} - Q^c_{[sd]}\right] * \phi^{c*}_{\pi[sd]} = 0$$

$$\forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c, c$$

$$\left[\sum_{ij} \frac{\partial J^c_{ij}(\mathbf{\Phi}^*,\mathbf{\Psi}^*)}{\partial \psi^c_{[sd]}} + \sum_{i} \frac{\partial J^c_{i}(\mathbf{\Phi}^*,\mathbf{\Psi}^*)}{\partial \psi^c_{[sd]}} + \right.$$

$$\left. + \frac{\partial J^c_{[sd]}(\mathbf{\Phi}^*,\mathbf{\Psi}^*)}{\partial \psi^c_{[sd]}} + \frac{\partial J^c_{[.d]}(\mathbf{\Phi}^*,\mathbf{\Psi}^*)}{\partial \psi^c_{[sd]}} - Q^c_{[s.]}\right] * \psi^{c*}_{[sd]} = 0$$

$$\forall\ [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c,\ c$$

$$\frac{\partial J^c_{[sd]}(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{o[sd]}} - Q^c_{[sd]} \geq 0 \quad \forall\ [sd] \in \mathbf{SD}^c,\ c$$

$$\sum_{ij} \frac{\partial J^c_{ij}(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{\pi[sd]}} + \sum_{i} \frac{\partial J^c_{i}(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{\pi[sd]}} - Q^c_{[sd]} \geq 0$$

$$\forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c, c$$

$$\sum_{ij} \frac{\partial J^c_{ij}(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \psi^c_{[sd]}} + \sum_{i} \frac{\partial J^c_{i}(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \psi^c_{[sd]}} +$$

$$+ \frac{\partial J^c_{[sd]}(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \psi^c_{[sd]}} + \frac{\partial J^c_{[.d]}(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \psi^c_{[sd]}} - Q^c_{[s.]} \geq 0$$

$$\forall\ [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c,\ c$$

$$\phi^{c*}_{o[sd]} + \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \phi^{c*}_{\pi[sd]} = 1 \quad \forall\ [sd] \in \mathbf{SD}^c,\ c$$

$$\sum_{[.d] \in \mathbf{D}^c_{[s.]}} \psi^{c*}_{[sd]} = 1 \qquad \forall\ [s.] \in \mathbf{S}^c,\ c$$

$$\phi^{c*}_{o[sd]},\ \phi^{c*}_{\pi[sd]} \geq 0 \qquad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c,\ c$$

$$\psi^{c*}_{[sd]} \geq 0 \qquad \forall\ [.d] \in \mathbf{D}^c_{s},\ [s.] \in \mathbf{S}^c,\ c$$

The partial derivatives of the cost function $J^c(\mathbf{\Phi}, \mathbf{\Psi})$ with respect to the path fractions $\phi^c_{\pi[sd]}$ can be written with respect to the link flows λ^c_{ij} and node flows λ^c_i:

$$\frac{\partial J^c_{ij}(\mathbf{\Phi}, \mathbf{\Psi})}{\partial \phi^c_{\pi[sd]}} = \frac{\partial J^c_{ij}(\mathbf{\Lambda}_{ij})}{\partial \lambda^c_{ij}} * \frac{\partial \lambda^c_{ij}}{\partial \phi^c_{\pi[sd]}}$$

$$= \frac{\partial J^c_{ij}(\mathbf{\Lambda}_{ij})}{\partial \lambda^c_{ij}} * (\gamma^c_{[sd]} + \gamma^c_{[s\cdot]} * \psi^c_{[sd]}) * 1_{ij \in \pi[sd]}$$

$$\frac{\partial J^c_i(\mathbf{\Phi}, \mathbf{\Psi})}{\partial \phi^c_{\pi[sd]}} = \frac{\partial J^c_i(\mathbf{\Lambda}_i)}{\partial \lambda^c_i} * \frac{\partial \lambda^c_i}{\partial \phi^c_{\pi[sd]}}$$

$$= \frac{\partial J^c_i(\mathbf{\Lambda}_i)}{\partial \lambda^c_i} * (\gamma^c_{[sd]} + \gamma^c_{[s\cdot]} * \psi^c_{[sd]}) * 1_{i \in \pi[sd]}$$

$$\frac{\partial J^c_{[sd]}(\mathbf{\Phi}, \mathbf{\Psi})}{\partial \phi^c_{o[sd]}} = \frac{\partial J^c_{[sd]}(\mathbf{\Lambda}_{o[sd]})}{\partial \lambda^c_{o[sd]}} * \frac{\partial \lambda^c_{o[sd]}}{\partial \phi^c_{o[sd]}}$$

$$= \frac{\partial J^c_{[sd]}(\mathbf{\Lambda}_{o[sd]})}{\partial \lambda^c_{o[sd]}} * (\gamma^c_{[sd]} + \gamma^c_{[s\cdot]} * \psi^c_{[sd]})$$

$$\frac{\partial J_{ij}^{c}(\mathbf{\Phi},\mathbf{\Psi})}{\partial \psi'^{c}_{[sd]}} = \frac{\partial J_{ij}^{c}(\mathbf{\Lambda}_{ij})}{\partial \lambda_{ij}^{c}} * \frac{\partial \lambda_{ij}^{c}}{\partial \psi'^{c}_{[sd]}}$$

$$= \sum_{\pi[sd]\in\mathbf{\Pi}^{c}_{[sd]}} \frac{\partial J_{ij}^{c}(\mathbf{\Lambda}_{ij})}{\partial \lambda_{ij}^{c}} * \gamma^{c}_{[s.]} * \phi^{c}_{\pi[sd]} * 1_{ij\in\pi[sd]}$$

$$\frac{\partial J_{i}^{c}(\mathbf{\Phi},\mathbf{\Psi})}{\partial \psi'^{c}_{[sd]}} = \frac{\partial J_{i}^{c}(\mathbf{\Lambda}_{i})}{\partial \lambda_{i}^{c}} * \frac{\partial \lambda_{i}^{c}}{\partial \psi'^{c}_{[sd]}}$$

$$= \sum_{\pi[sd]\in\mathbf{\Pi}^{c}_{[sd]}} \frac{\partial J_{i}^{c}(\mathbf{\Lambda}_{i})}{\partial \lambda_{i}^{c}} * \gamma^{c}_{[s.]} * \phi^{c}_{\pi[sd]} * 1_{i\in\pi[sd]}$$

$$\frac{\partial J^{c}_{[sd]}(\mathbf{\Phi},\mathbf{\Psi})}{\partial \psi'^{c}_{[sd]}} = \frac{\partial J^{c}_{[sd]}(\mathbf{\Lambda}_{o[sd]})}{\partial \lambda^{c}_{o[sd]}} * \frac{\partial \lambda^{c}_{o[sd]}}{\partial \psi'^{c}_{[sd]}} = \frac{\partial J^{c}_{[sd]}(\mathbf{\Lambda}_{o[sd]})}{\partial \lambda^{c}_{o[sd]}} * \gamma^{c}_{[s.]} * \phi^{c}_{o[sd]}$$

$$\frac{\partial J^{c}_{[.d]}(\mathbf{\Phi},\mathbf{\Psi})}{\partial \psi'^{c}_{[sd]}} = \frac{\partial J^{c}_{[.d]}(\mathbf{\Lambda}_{[.d]})}{\partial \lambda^{c}_{[.d]}} * \frac{\partial \lambda^{c}_{[.d]}}{\partial \psi'^{c}_{[sd]}} = \frac{\partial J^{c}_{[.d]}(\mathbf{\Lambda}_{[.d]})}{\partial \lambda^{c}_{[.d]}} * \gamma^{c}_{[s.]}$$

Then the Karush-Kuhn-Tucker conditions become:

$$\left[\frac{\partial J^c_{[sd]}(\mathbf{\Lambda}^*_{o[sd]})}{\partial \lambda^c_{o[sd]}} * (\gamma^c_{[sd]} + \gamma^c_{[s.]} * \psi^{c*}_{[sd]}) - Q^c_{[sd]}\right] * \phi^{c*}_{o[sd]} = 0 \quad \forall\ [sd] \in \mathbf{SD}^c,\ c$$

$$\left[\sum_{ij} \frac{\partial J^c_{ij}(\mathbf{\Lambda}^*_{ij})}{\partial \lambda^c_{ij}} * (\gamma^c_{[sd]} + \gamma^c_{[s.]} * \psi^{c*}_{[sd]}) * 1_{ij \in \pi[sd]} +\right.$$

$$\left. + \sum_{i} \frac{\partial J^c_{i}(\mathbf{\Lambda}^*_{i})}{\partial \lambda^c_{i}} * (\gamma^c_{[sd]} + \gamma^c_{[s.]} * \psi^{c*}_{[sd]}) * 1_{i \in \pi[sd]} - Q^c_{[sd]}\right] * \phi^{c*}_{\pi[sd]} = 0$$

$$\forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c, c$$

$$\left[\sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \sum_{ij} \frac{\partial J^c_{ij}(\mathbf{\Lambda}^*_{ij})}{\partial \lambda^c_{ij}} * \gamma^c_{[s.]} * \phi^{c*}_{\pi[sd]} * 1_{ij \in \pi[sd]} +\right.$$

$$+ \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \sum_{i} \frac{\partial J^c_{i}(\mathbf{\Lambda}^*_{i})}{\partial \lambda^c_{i}} * \gamma^c_{[s.]} * \phi^{c*}_{\pi[sd]} * 1_{i \in \pi[sd]} +$$

$$\left. + \frac{\partial J^c_{[sd]}(\mathbf{\Lambda}^*_{[sd]})}{\partial \lambda^c_{o[sd]}} * \gamma^c_{[s.]} * \phi^{c*}_{o[sd]} + \frac{\partial J^c_{[.d]}(\mathbf{\Lambda}^*_{[.d]})}{\partial \lambda^c_{[.d]}} * \gamma^c_{[s.]} - Q^c_{[s.]}\right] * \psi^{c*}_{[sd]} = 0$$

$$\forall\ [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c,\ c$$

$$\frac{\partial J^c_{[sd]}(\mathbf{\Lambda}^*_{[sd]})}{\partial \lambda^c_{o[sd]}} * (\gamma^c_{[sd]} + \gamma^c_{[s.]} * \psi^{c*}_{[sd]}) - Q^c_{[sd]} \geq 0 \quad \forall\ [sd] \in \mathbf{SD}^c,\ c$$

$$\sum_{ij} \frac{\partial J^c_{ij}(\mathbf{\Lambda}^*_{ij})}{\partial \lambda^c_{ij}} * (\gamma^c_{[sd]} + \gamma^c_{[s.]} * \psi^{c*}_{[sd]}) * 1_{ij \in \pi[sd]} +$$

$$+ \sum_{i} \frac{\partial J^c_{i}(\mathbf{\Lambda}^*_{i})}{\partial \lambda^c_{i}} * (\gamma^c_{[sd]} + \gamma^c_{[s.]} * \psi^{c*}_{[sd]}) * 1_{i \in \pi[sd]} + -Q^c_{[sd]} \geq 0$$

$$\forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c, c$$

$$\sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \sum_{ij} \frac{\partial J^c_{ij}(\mathbf{\Lambda}^*_{ij})}{\partial \lambda^c_{ij}} * \gamma^c_{[s.]} * \phi^{c*}_{\pi[sd]} * 1_{ij \in \pi[sd]} +$$

$$+ \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \sum_{i} \frac{\partial J^c_{i}(\mathbf{\Lambda}^*_{i})}{\partial \lambda^c_{i}} * \gamma^c_{[s.]} * \phi^{c*}_{\pi[sd]} * 1_{i \in \pi[sd]} +$$

$$+ \frac{\partial J^c_{[sd]}(\mathbf{\Lambda}^*_{[sd]})}{\partial \lambda^c_{o[sd]}} * \gamma^c_{[s.]} * \phi^{c*}_{o[sd]} + \frac{\partial J^c_{[.d]}(\mathbf{\Lambda}^*_{[.d]})}{\partial \lambda^c_{[.d]}} * \gamma^c_{[s.]} - Q^c_{[s.]} \geq 0$$

$$\forall\ [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c,\ c$$

$$\phi^{c*}_{o[sd]} + \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \phi^{c*}_{\pi[sd]} = 1 \quad \forall\ [sd] \in \mathbf{SD}^c,\ c$$

$$\sum_{[.d] \in \mathbf{D}^c_{[s.]}} \psi^{c*}_{[sd]} = 1 \qquad \forall\ [s.] \in \mathbf{S}^c,\ c$$

$$\phi^{c*}_{o[sd]},\ \phi^{c*}_{\pi[sd]} \geq 0 \qquad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c,\ c$$

$$\psi^{c*}_{[sd]} \geq 0 \qquad \forall\ [.d] \in \mathbf{D}^c_s,\ [s.] \in \mathbf{S}^c,\ c$$

Next, for each class c, we define the length for the rejected flow $[sd]$, the length for the path $\pi[sd]$ and the length for the source-destination pair $[sd]$:

$$l^{c,Nash}_{o[sd]} = \frac{\partial J^c_{o[sd]}(\mathbf{\Lambda}_{o[sd]})}{\partial \lambda^c_{o[sd]}} * (\gamma^c_{[sd]} + \gamma^c_{[s.]} * \psi^c_{[sd]})$$

$$l^{c,Nash}_{\pi[sd]} = \sum_{ij} \frac{\partial J^c_{ij}(\mathbf{\Lambda}_{ij})}{\partial \lambda^c_{ij}} * (\gamma^c_{[sd]} + \gamma^c_{[s.]} * \psi^c_{[sd]}) * 1_{ij \in \pi[sd]} +$$

$$+ \sum_{i} \frac{\partial J^c_{i}(\mathbf{\Lambda}_{i})}{\partial \lambda^c_{i}} * (\gamma^c_{[sd]} + \gamma^c_{[s.]} * \psi^c_{[sd]}) * 1_{i \in \pi[sd]}$$

$$l^{c,Nash}_{[sd]} = \sum_{\pi[sd]} \sum_{ij} \frac{\partial J^c_{ij}(\mathbf{\Lambda}_{ij})}{\partial \lambda^c_{ij}} * \gamma^c_{[s.]} * \phi^c_{\pi[sd]} * 1_{ij \in \pi[sd]} +$$

$$+ \sum_{\pi[sd]} \sum_{i} \frac{\partial J^c_{i}(\mathbf{\Lambda}_{i})}{\partial \lambda^c_{i}} * \gamma^c_{[s.]} * \phi^c_{\pi[sd]} * 1_{i \in \pi[sd]} +$$

$$+ \frac{\partial J^c_{[sd]}(\mathbf{\Lambda}_{[sd]})}{\partial \lambda^c_{o[sd]}} * \gamma^c_{[s.]} * \phi^c_{o[sd]} + \frac{\partial J^c_{[.d]}(\mathbf{\Lambda}_{[.d]})}{\partial \lambda^c_{[.d]}} * \gamma^c_{[s.]}$$

External arriving flow at a source is assigned to the destination that has the minimum length from the source. However, this flow may be rejected if the length of rejecting it is less than the lengths of the paths to its destination. If it is accepted, then it is routed to its destination via the minimum length path.

In the next section, we will derive the same conditions by an alternative way, and we shall state the above ideas more formally.

4.2.5 V.I. for Separable Cost Functions

Equivalently, the Nash equilibrium definition, each class c minimizes its cost function J^c given the optimum decisions of all other classes. We first solve the routing and congestion control problem assuming that all other classes act optimally for themselves. So, class c first solves the routing and congestion control problems

$$
\begin{aligned}
&\textit{minimize} \\
&J^c\left(\begin{array}{c}\boldsymbol{\Phi}^{1*},\ldots.\,\boldsymbol{\Phi}^{c},\ldots,\boldsymbol{\Phi}^{C*}\\ \boldsymbol{\Psi}^{1*},\ldots.\,\boldsymbol{\Psi}^{c},\ldots,\boldsymbol{\Psi}^{C*}\end{array}\right) &= \sum_{ij} J^c_{ij}(\lambda^{1*}_{ij},\ldots,\lambda^{c}_{ij},\ldots,\lambda^{C*}_{ij}) \;+ \\
&&+\;\sum_{i} J^c_{i}(\lambda^{1*}_{i},\ldots,\lambda^{c}_{i},\ldots,\lambda^{C*}_{i}) \;+ \\
&&+\;\sum_{[sd]} J^c_{[sd]}(\lambda^{1*}_{o[sd]},\ldots,\lambda^{c}_{o[sd]},\ldots,\lambda^{C*}_{o[sd]}) \;+ \\
&&+\;\sum_{[.d]} J^c_{[.d]}(\lambda^{1*}_{[.d]},\ldots,\lambda^{c}_{[.d]},\ldots,\lambda^{C*}_{[.d]})
\end{aligned}
$$

with respect to $\boldsymbol{\Phi}^c$

such that

$$\phi^c_{o[sd]} + \sum_{\pi[sd]\in\boldsymbol{\Pi}^c_{[sd]}} \phi^c_{\pi[sd]} = 1 \quad \forall\, [sd] \in \mathbf{SD}^c$$

$$\phi^c_{o[sd]},\ \phi^c_{\pi[sd]} \geq 0 \quad \forall \pi[sd] \in \boldsymbol{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c$$

The necessary and sufficient optimality conditions for class c are [311]

$$
\sum_{[sd]\in\mathbf{SD}^c}\left\{\frac{\partial J^c(\boldsymbol{\Phi}^*,\boldsymbol{\Psi}^*)}{\partial \phi^c_{o[sd]}} * (\phi^c_{o[sd]} - \phi^{c*}_{o[sd]}) \;+\right.
$$

$$
\left.+\sum_{\pi[sd]\in\boldsymbol{\Pi}^c_{[sd]}} \frac{\partial J^c(\boldsymbol{\Phi}^*,\boldsymbol{\Psi}^*)}{\partial \phi^c_{\pi[sd]}} * (\phi^c_{\pi[sd]} - \phi^{c*}_{\pi[sd]})\right\} \geq 0 \quad \forall\, \boldsymbol{\Phi}^c \in \mathbf{RC}^c
$$

such that

$$\phi^{c*}_{o[sd]} + \sum_{\pi[sd]\in\boldsymbol{\Pi}^c_{[sd]}} \phi^{c*}_{\pi[sd]} = 1 \quad \forall\, [sd] \in \mathbf{SD}^c$$

$$\phi^{c*}_{o[sd]},\ \phi^{c*}_{\pi[sd]} \geq 0 \quad \forall \pi[sd] \in \boldsymbol{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c$$

We can decompose these conditions for each source-destination pair $[sd] \in \mathbf{SD}^c$

$$\frac{\partial J^c(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{o[sd]}} * (\phi^c_{o[sd]} - \phi^{c*}_{o[sd]}) + \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \frac{\partial J^c(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{\pi[sd]}} * (\phi^c_{\pi[sd]} - \phi^{c*}_{\pi[sd]}) \geq 0 \quad \forall \mathbf{\Phi}^c \in \mathbf{RC}^c$$

$$\phi^{c*}_{o[sd]} + \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \phi^{c*}_{\pi[sd]} = 1$$

$$\phi^{c*}_{o[sd]},\ \phi^{c*}_{\pi[sd]} \geq 0 \qquad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]}$$

Therefore, there must be flow only on paths where the first derivative of the cost function $J^c(\mathbf{\Phi}^*, \mathbf{\Psi}^*)$ with respect to the path routing fractions $\phi^c_{\pi[sd]}$ is minimum

$$\phi^{c*}_{\pi[sd]} > 0 \quad only\ \ if \quad \frac{\partial J^c(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{\pi[sd]}} \leq \frac{\partial J^c(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{o[sd]}}$$

$$and \qquad \frac{\partial J^c(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{\pi[sd]}} \leq \frac{\partial J^c(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{k[sd]}}$$

$$\forall\ k[sd] \neq \pi[sd] \qquad k[sd] \in \mathbf{\Pi}^c_{[sd]} \qquad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]}$$

Also, flow is not admitted into the network only if the first derivative of the cost function $J^c(\mathbf{\Phi}^*, \mathbf{\Psi}^*)$ with respect to the rejection fraction $\phi^c_{o[sd]}$ is minimum:

$$\phi^{c*}_{o[sd]} > 0 \quad only\ \ if \quad \frac{\partial J^c(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{o[sd]}} \leq \frac{\partial J^c(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \phi^c_{k[sd]}} \qquad \forall\ k[sd] \in \mathbf{\Pi}^c_{[sd]}$$

Theorem : *Routing*

There must be flow only on minimum length paths:

$$\phi^{c*}_{\pi[sd]} > 0 \quad only \quad if \quad l^{c,Nash*}_{\pi[sd]} = \min\{l^{c,Nash*}_{o[sd]}, \min_{p[sd]}\{l^{c,Nash*}_{p[sd]}\}\}$$

$$\phi^{c*}_{\pi[sd]} = 0 \qquad o.w.$$

$$\phi^{c*}_{o[sd]} + \sum_{\pi[sd]\in\mathbf{\Pi}^c_{[sd]}} \phi^{c*}_{\pi[sd]} = 1$$

$$\forall\ \pi[sd] \in \Pi^c_{[sd]}, \quad [sd] \in \mathbf{SD}^c, \quad c$$

Theorem : *Congestion Control*

Flow is not admitted into the network only if its rejection length is less than the minimum length path to its destination:

$$\phi^{c*}_{o[sd]} > 0 \quad only \quad if \quad l^{c,Nash*}_{o[sd]} = \min\{l^{c,Nash*}_{o[sd]}, \min_{p[sd]}\{l^{c,Nash*}_{p[sd]}\}\}$$

$$\phi^{c*}_{[sd]} = 0 \qquad o.w.$$

$$\phi^{c*}_{o[sd]} + \sum_{\pi[sd]\in\mathbf{\Pi}^c_{[sd]}} \phi^{c*}_{\pi[sd]} = 1 \quad \forall\ [sd] \in \mathbf{SD}^c, \quad c$$

Having found the optimum routing and congestion control decisions, we proceed to solve the load sharing problem for class c assuming that all other classes act at their optimum decisions. So, the load sharing problem for class c is

minimize

$$J^c\left(\begin{array}{c}\mathbf{\Phi}^{1*},...,\mathbf{\Phi}^{c*},...,\mathbf{\Phi}^{C*}\\ \mathbf{\Psi}^{1*},...,\mathbf{\Psi}^{c},...,\mathbf{\Psi}^{C*}\end{array}\right) = \sum_{ij} J^c_{ij}(\lambda^{1*}_{ij},...,\lambda^{c}_{ij},...,\lambda^{C*}_{ij}) +$$

$$+ \sum_{i} J^c_{i}(\lambda^{1*}_{i},...,\lambda^{c}_{i},...,\lambda^{C*}_{i}) +$$

$$+ \sum_{[sd]} J^c_{[sd]}(\lambda^{1*}_{o[sd]},...,\lambda^{c}_{o[sd]},...,\lambda^{C*}_{o[sd]}) +$$

$$+ \sum_{[.d]} J^c_{[.d]}(\lambda^{1*}_{[.d]},...,\lambda^{c}_{[.d]},...,\lambda^{C*}_{[.d]})$$

with respect to $\mathbf{\Psi}^c$

such that

$$\sum_{[.d]\in\mathbf{D}^c_{[s.]}} \psi^c_{[sd]} = 1 \quad \forall\ [s.] \in \mathbf{S}^c$$

$$\psi^c_{[sd]} \geq 0 \quad \forall\ [.d] \in \mathbf{D}^c_{[s.]},\ \ [s.] \in \mathbf{S}^c$$

The necessary and sufficient optimality conditions for class c are:

$$\sum_{[sd]\in\mathbf{SD}^c} \frac{\partial J^c(\mathbf{\Phi}^*,\mathbf{\Psi}^*)}{\partial \psi^c_{[sd]}} * (\psi^c_{[sd]} - \psi^{c*}_{[sd]}) \quad \geq 0 \quad \forall\ \mathbf{\Psi}^c \in \mathbf{LS}^c$$

such that

$$\sum_{[.d]\in\mathbf{D}^c_{[s.]}} \psi^{c*}_{[sd]} = 1 \quad \forall\ [s.] \in \mathbf{S}^c$$

$$\psi^{c*}_{[sd]} \geq 0 \quad \forall\ [.d] \in \mathbf{D}^c_{[s.]},\ \ [s.] \in \mathbf{S}^c$$

We can decompose these conditions for each source node $[s.] \in \mathbf{S}^c$

$$\sum_{[.d] \in \mathbf{D}^c_{[s.]}} \frac{\partial J^c(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \psi^c_{[sd]}} * (\psi^c_{[sd]} - \psi^{c*}_{[sd]}) \quad \geq 0 \qquad \forall \ \mathbf{\Psi}^c \in \mathbf{LS}^c$$

such that

$$\sum_{[.d] \in \mathbf{D}^c_{[s.]}} \psi^{c*}_{[sd]} = 1$$

$$\psi^{c*}_{[sd]} \geq 0 \qquad \forall \ [.d] \in \mathbf{D}^c_{[s.]}$$

Therefore, there must be flow only for source-destination pairs for which the first derivative of the cost function $J^c(\mathbf{\Phi}^*, \mathbf{\Psi}^*)$ with respect to the destination load sharing fractions $\psi^c_{[sd]}$ is minimum:

$$\psi^{c*}_{[sd]} > 0 \quad only \quad if \quad \frac{\partial J^c(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \psi^c_{[sd]}} \leq \frac{\partial J^c(\mathbf{\Phi}^*, \mathbf{\Psi}^*)}{\partial \psi^c_{[sd']}} \qquad \forall \ [.d'] \neq [.d], \quad [.d'], [.d] \in \mathbf{D}^c_{[s.]}$$

Theorem : *Load Sharing*

For each source, there must be flow only to destinations whose length is minimum:

$$\psi^{c*}_{[sd]} > 0 \quad only \quad if \qquad l^{c,Nash*}_{[sd]} = \min_{[sd']} \{ l^{c,Nash*}_{[sd']} \}$$

$$\psi^{c*}_{[sd]} = 0 \qquad\qquad o.w.$$

$$\sum_{[.d] \in \mathbf{D}^c_{[s.]}} \psi^{c*}_{[sd]} = 1 \ \ \forall \ [.d] \in \mathbf{D}^c_{[s.]}, [s.] \in \mathbf{S}^c, \ c$$

Thus, in this section we have formulated and solved the load sharing, routing and congestion control problem as a Nash game among multiple competing classes.

4.3 Stackelberg Equilibrium Solution

In this section, we formulate the joint load sharing, routing and congestion control problem in distributed systems with two classes of jobs, one more powerful than the other, as a non-cooperative Stackelberg game. A simple example of such classes of jobs is when they have different priorities.

Customers of the most powerful class use the resources of the distributed system for their own benefit, ignoring the inconvenience that they cause to customers from the less powerful class.

After the pioneering work of von Stackelberg (1938), these games have been named Stackelberg games. Next, we briefly survey research on static Stackelberg games:

Bracken & McGill [76] formulate mathematical programs with optimization problems in the constraints, i.e. Stackelberg games. Polak & Mayne [382] present an algorithm for minimizing a function subject to functional inequality constraints. Blankeship & Falk [54] describe a generalized cutting-plane algorithm for constrained optimization and minimax problems.

Simaan [447] considers Stackelberg games for two-level optimization problems. Castanon & Sandell [91] illustrate the non-convexity of Stackelberg games and suggest that the leader's cost function must be selected carefully.

Papavassilopoulos [372, 371] describes algorithms for leader-follower games. He notice that difficulties arise due to the nonconvex character of the follower's reaction set. He also [373] solves the linear quadratic Gaussian static Nash and Stackelberg games. He presents necessary and sufficient conditions for existence and uniqueness of the solution as well as procedures for finding all the solutions.

Bialas & Karwan [51, 52] present techniques for Stackelberg games. Shimizu & Aiyoshi [442] apply the penalty method to solve the Stackelberg game. Chang & Luh [95] and Luh, Chang & Chang [313] derive necessary and sufficient conditions for Stackelberg games via the inducible region concept.

In the following, we shall develop a methodology for the joint quasi-static load sharing, routing and congestion control problem based on the Stackelberg game theory.

Next, we give some definitions for a two hierarchical class game similar to those in [27] for Stackelberg games:

Definition :

In a two hierarchical class joint load sharing, routing and congestion control problem, with the most powerful (e.g. high priority) class α as the leader and the less powerful (e.g. low priority) class β as the follower, the set $\mathcal{R}^{\beta}(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha})$. defined for the class α strategy $(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}) \in (\mathbf{RC}^{\alpha}, \mathbf{LS}^{\alpha})$, by:

$$\mathcal{R}^{\beta}(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}) = \{ \quad (\mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta}) \in (\mathbf{RC}^{\beta}, \mathbf{\Psi}^{\beta}) \text{ such that :}$$
$$J^{\beta}(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}, \mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta}) \leq J^{\beta}(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}, \hat{\mathbf{\Phi}}^{\beta}, \hat{\mathbf{\Psi}}^{\beta}),$$
$$\forall \ (\hat{\mathbf{\Phi}}^{\beta}, \hat{\mathbf{\Psi}}^{\beta}), \ \text{such that} \ (\hat{\mathbf{\Phi}}^{\beta}, \hat{\mathbf{\Psi}}^{\beta}) \in (\mathbf{RC}^{\beta}, \mathbf{LS}^{\beta}) \}$$

is the optimal response (rational reaction) set of the less powerful class β to the strategy of the most powerful class α.

What the above definition says is that the less powerful class β chooses its decision vector $(\mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta})$, that minimizes its cost function $J^{\beta}(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}, \mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta})$, for given strategy $(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha})$ of the most powerful class α.

Definition :

In a two hierarchical class joint load sharing, routing and congestion control problem with the most powerful (e.g. high priority) class α as the leader, a strategy $(\mathbf{\Phi}^{\alpha *}, \mathbf{\Psi}^{\alpha *}) \in (\mathbf{RC}^{\alpha}, \mathbf{LS}^{\alpha})$ is called a Stackelberg equilibrium strategy for the most powerful class α if and only if

$$\inf_{(\mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta}) \in \mathcal{R}^{\beta}(\mathbf{\Phi}^{\alpha *}, \mathbf{\Psi}^{\alpha *})} J^{\alpha}(\mathbf{\Phi}^{\alpha *}, \mathbf{\Psi}^{\alpha *}, \mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta}) \leq \inf_{(\mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta}) \in \mathcal{R}^{\beta}(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha})} J^{\alpha}(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}, \mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta})$$

$$\forall \ (\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}) \in (\mathbf{RC}^{\alpha}, \mathbf{LS}^{\alpha})$$

This means that the most powerful class α chooses its strategy $(\mathbf{\Phi}^{\alpha *}, \mathbf{\Psi}^{\alpha *})$ that minimizes its cost function $J^{\alpha}(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}, \mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta})$, given the optimal response set $\mathcal{R}^{\beta}(\mathbf{\Phi}^{\alpha *}, \mathbf{\Psi}^{\alpha *})$ of the less powerful class β to its strategy $(\mathbf{\Phi}^{\alpha *}, \mathbf{\Psi}^{\alpha *})$.

Definition :

Let $(\mathbf{\Phi}^{\alpha*}, \mathbf{\Psi}^{\alpha*}) \in (\mathbf{RC}^{\alpha}, \mathbf{LS}^{\alpha})$ *be a Stackelberg strategy for the most powerful (e.g. high priority) class* α. *Then any element* $(\mathbf{\Phi}^{\beta*}, \mathbf{\Psi}^{\beta*}) \in \mathcal{R}^{\beta}(\mathbf{\Phi}^{\alpha*}, \mathbf{\Psi}^{\alpha*})$ *is an optimal strategy for the less powerful (e.g. low priority) class* β *that is in equilibrium with* $(\mathbf{\Phi}^{\alpha*}, \mathbf{\Psi}^{\alpha*})$. *The strategy* $(\mathbf{\Phi}^{\alpha*}, \mathbf{\Psi}^{\alpha*}, \mathbf{\Phi}^{\beta*}, \mathbf{\Psi}^{\beta*})$ *is a* Stackelberg solution *for the game with the most powerful class* α *as the leader and the cost pair*

$J^{\alpha}(\mathbf{\Phi}^{\alpha*}, \mathbf{\Psi}^{\alpha*}, \mathbf{\Phi}^{\beta*}, \mathbf{\Psi}^{\beta*})$, $J^{\beta}(\mathbf{\Phi}^{\alpha*}, \mathbf{\Psi}^{\alpha*}, \mathbf{\Phi}^{\beta*}, \mathbf{\Psi}^{\beta*})$ *is the corresponding Stackelberg equilibrium outcome.*

4.3.1 Nonlinear Programming Formulation

In this section, we formulate the two hierarchical class joint load sharing, routing and congestion control problem as a Nonlinear Programming Problem (NPP).

Theorem :

Consider the quasi-static two hierarchical class joint load sharing, routing and congestion control problem in distributed systems.

$(\mathbf{\Phi}^{*}, \mathbf{\Psi}^{*}) \in (\mathbf{RC}, \mathbf{LS})$ *is a Stackelberg equilibrium if and only if it solves the following Nonlinear Programming Problem:*

$$\textit{minimize} \qquad J^{\alpha}(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}, \mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta})$$

$$\textit{with respect to} \quad (\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}, \mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta}, \mathbf{Q}^{\beta})$$

$$\textit{such that} \qquad (\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}) \in (\mathbf{RC}^{\alpha}, \mathbf{LS}^{\alpha}) \qquad (\mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta}) \in (\mathbf{RC}^{\beta}, \mathbf{LS}^{\beta})$$

$$J^{\beta}(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}, \mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta}) = \min_{(\hat{\mathbf{\Phi}}^{\beta}, \hat{\mathbf{\Psi}}^{\beta}) \in (\mathbf{RC}^{\beta}, \mathbf{LS}^{\beta})} J^{\beta}(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}, \hat{\mathbf{\Phi}}^{\beta}, \hat{\mathbf{\Psi}}^{\beta})$$

Proof: It follows directly from the definition of the Stackelberg equilibrium. □

Theorem :

Consider the quasi-static two hierarchical class joint load sharing, routing and congestion control problem in distributed systems.

If the cost function of the most powerful class α, J^{α}, *is differentiable and convex in* $(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}) \in (\mathbf{RC}^{\alpha}, \mathbf{LS}^{\alpha})$, *for each fixed value of* $(\mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta}) \in (\mathbf{RC}^{\beta}, \mathbf{LS}^{\beta})$ *and the cost function of the less powerful class* β, J^{β}, *is differentiable and convex in* $(\mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta}) \in (\mathbf{RC}^{\beta}, \mathbf{LS}^{\beta})$, *for each fixed value of* $(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}) \in (\mathbf{RC}^{\alpha}, \mathbf{LS}^{\alpha})$,

then $(\mathbf{\Phi}^{*}, \mathbf{\Psi}^{*}) \in (\mathbf{RC}, \mathbf{LS})$ *is a Stackelberg equilibrium if and only if it solves the following Nonlinear Programming Problem:*

$$\textit{minimize} \qquad J^{\alpha}(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}, \mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta})$$

$$\textit{with respect to} \quad (\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}, \mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta}, \mathbf{Q}^{\beta})$$

$$\textit{such that} \qquad \left[\frac{\partial J^{\beta}(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}, \mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta})}{\partial \phi^{\beta}_{o[sd]}} - Q^{\beta}_{[sd]}\right] * \phi^{\beta}_{o[sd]} = 0$$

$$\forall\ [sd] \in \mathbf{SD}^{\beta}$$

$$\left[\frac{\partial J^{\beta}(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}, \mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta})}{\partial \phi^{\beta}_{\pi[sd]}} - Q^{\beta}_{[sd]}\right] * \phi^{\beta}_{\pi[sd]} = 0$$

$$\forall\ \pi[sd] \in \mathbf{\Pi}^{\beta}_{[sd]},\ [sd] \in \mathbf{SD}^{\beta}$$

$$\left[\frac{\partial J^{\beta}(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}, \mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta})}{\partial \psi^{\beta}_{[sd]}} - Q^{\beta}_{[s.]}\right] * \psi^{\beta}_{[sd]} = 0$$

$$\forall\ [.d] \in \mathbf{D}^{\beta}_{[s.]},\ [s.] \in \mathbf{S}^{\beta}$$

$$\frac{\partial J^{\beta}(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}, \mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta})}{\partial \phi^{\beta}_{o[sd]}} - Q^{\beta}_{[sd]} \geq 0 \quad \forall\ [sd] \in \mathbf{SD}^{\beta}$$

$$\frac{\partial J^{\beta}(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}, \mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta})}{\partial \phi^{\beta}_{\pi[sd]}} - Q^{\beta}_{[sd]} \geq 0$$

$$\forall\ \pi[sd] \in \mathbf{\Pi}^{\beta}_{[sd]},\ [sd] \in \mathbf{SD}^{\beta}$$

$$\frac{\partial J^{\beta}(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}, \mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta})}{\partial \psi^{\beta}_{[sd]}} - Q^{\beta}_{[s.]} \geq 0$$

$$\forall\ [.d] \in \mathbf{D}^{\beta}_{[s.]},\ [s.] \in \mathbf{S}^{\beta}$$

$$\phi^{\alpha}_{o[sd]} + \sum_{\pi[sd] \in \mathbf{\Pi}^{\alpha}_{[sd]}} \phi^{\alpha}_{\pi[sd]} = 1 \quad \forall\ [sd] \in \mathbf{SD}^{\alpha}$$

$$\sum_{[.d] \in \mathbf{D}^{\alpha}_{[s.]}} \psi^{\alpha}_{[sd]} = 1 \qquad \forall\ [s.] \in \mathbf{S}^{\alpha}$$

$$\phi^{\alpha}_{o[sd]},\ \phi^{\alpha}_{\pi[sd]} \geq 0 \qquad \forall\ \pi[sd] \in \mathbf{\Pi}^{\alpha}_{[sd]},\ [sd] \in \mathbf{SD}^{\alpha}$$

$$\psi^{\alpha}_{[sd]} \geq 0 \qquad \forall\ [.d] \in \mathbf{D}^{\alpha}_{[s.]},\ [s.] \in \mathbf{S}^{\alpha}$$

$$\phi^{\beta}_{o[sd]} + \sum_{\pi[sd]\in\mathbf{\Pi}^{\beta}_{[sd]}} \phi^{\beta}_{\pi[sd]} = 1 \quad \forall\ [sd] \in \mathbf{SD}^{\beta}$$

$$\sum_{[.d]\in\mathbf{D}^{\beta}_{[s.]}} \psi^{\beta}_{[sd]} = 1 \qquad \forall\ [s.] \in \mathbf{S}^{\beta}$$

$$\phi^{\beta}_{o[sd]},\ \phi^{\beta}_{\pi[sd]} \geq 0 \qquad \forall\ \pi[sd] \in \mathbf{\Pi}^{\beta}_{[sd]},\ [sd] \in \mathbf{SD}^{\beta}$$

$$\psi^{\beta}_{[sd]} \geq 0 \qquad \forall\ [.d] \in \mathbf{D}^{\beta}_{[s.]},\ [s.] \in \mathbf{S}^{\beta}$$

Proof: The Lagrangian for the less powerful class β is

$$L^{\beta} = J^{\beta} + \sum_{[sd]\in\mathbf{SD}^{\beta}} Q^{\beta}_{[sd]} * \left[1 - \phi^{\beta}_{o[sd]} - \sum_{\pi[sd]\in\mathbf{\Pi}^{\beta}_{[sd]}} \phi^{\beta}_{\pi[sd]}\right] +$$

$$\sum_{[s.]\in\mathbf{S}^{\beta}} Q^{\beta}_{[s.]} * \left[1 - \sum_{[.d]\in\mathbf{D}^{\beta}_{[s.]}} \psi^{\beta}_{[sd]}\right]$$

with $\phi^{\beta}_{o[sd]},\ \phi^{\beta}_{\pi[sd]},\ \psi^{\beta}_{[sd]} \geq 0 \quad \forall\ \pi[sd] \in \mathbf{\Pi}^{\beta}_{[sd]},\ [sd] \in \mathbf{SD}^{\beta}$

The cost function for the less powerful class β, J^{β}, is convex in $(\mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta}) \in (\mathbf{RC}^{\beta}, \mathbf{LS}^{\beta})$, so the Karush-Kuhn-Tucker necessary conditions are also sufficient :

$$\frac{\partial L^{\beta}(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}, \mathbf{\Phi}^{\beta*}, \mathbf{\Psi}^{\beta*}, \mathbf{Q}^{\beta})}{\partial \phi^{\beta}_{o[sd]}} * \phi^{\beta*}_{o[sd]} = 0 \Rightarrow$$

$$[\frac{\partial J^{\beta}(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}, \mathbf{\Phi}^{\beta*}, \mathbf{\Psi}^{\beta*})}{\partial \phi^{\beta}_{o[sd]}} - Q^{\beta}_{[sd]}] * \phi^{\beta*}_{o[sd]} = 0 \quad \forall\, [sd] \in \mathbf{SD}^{\beta}$$

$$\frac{\partial L^{\beta}(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}, \mathbf{\Phi}^{\beta*}\mathbf{\Psi}^{\beta*}, \mathbf{Q}^{\beta})}{\partial \phi^{\beta}_{\pi[sd]}} * \phi^{\beta*}_{\pi[sd]} = 0 \Rightarrow$$

$$[\frac{\partial J^{\beta}(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}, \mathbf{\Phi}^{\beta*}, \mathbf{\Psi}^{\beta*})}{\partial \phi^{\beta}_{\pi[sd]}} - Q^{\beta}_{[sd]}] * \phi^{\beta*}_{\pi[sd]} = 0 \quad \forall\, \pi[sd] \in \mathbf{\Pi}^{\beta}_{[sd]},\ [sd] \in \mathbf{SD}^{\beta}$$

$$\frac{\partial L^{\beta}(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}, \mathbf{\Phi}^{\beta*}, \mathbf{\Psi}^{\beta*}, \mathbf{Q}^{\beta})}{\partial \psi^{\beta}_{[sd]}} * \psi^{\beta*}_{[sd]} = 0 \Rightarrow$$

$$[\frac{\partial J^{\beta}(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}, \mathbf{\Phi}^{\beta*}, \mathbf{\Psi}^{\beta*})}{\partial \psi^{\beta}_{[sd]}} - Q^{\beta}_{[s.]}] * \psi^{\beta*}_{[sd]} = 0 \quad \forall\, [.d] \in \mathbf{D}^{\beta}_{[s.]},\ [s.] \in \mathbf{S}^{\beta}$$

$$\frac{\partial L^{\beta}(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}, \mathbf{\Phi}^{\beta*}, \mathbf{\Psi}^{\beta*}, \mathbf{Q}^{\beta})}{\partial \phi^{\beta}_{o[sd]}} \geq 0 \;\; \Rightarrow$$

$$\frac{\partial J^{\beta}(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}, \mathbf{\Phi}^{\beta*}, \mathbf{\Psi}^{\beta*})}{\partial \phi^{\beta}_{o[sd]}} - Q^{\beta}_{[sd]} \geq 0 \quad \forall\, [sd] \in \mathbf{SD}^{\beta}$$

$$\frac{\partial L^{\beta}(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}, \mathbf{\Phi}^{\beta*}, \mathbf{\Psi}^{\beta*}, \mathbf{Q}^{\beta})}{\partial \phi^{\beta}_{\pi[sd]}} \geq 0 \;\; \Rightarrow$$

$$\frac{\partial J^{\beta}(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}, \mathbf{\Phi}^{\beta*}, \mathbf{\Psi}^{\beta*})}{\partial \phi^{\beta}_{\pi[sd]}} - Q^{\beta}_{[sd]} \geq 0 \quad \forall\, \pi[sd] \in \mathbf{\Pi}^{\beta}_{[sd]},\; [sd] \in \mathbf{SD}^{\beta}$$

$$\frac{\partial L^{\beta}(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}, \mathbf{\Phi}^{\beta*}, \mathbf{\Psi}^{\beta*}, \mathbf{Q}^{\beta})}{\partial \psi^{\beta}_{[sd]}} \geq 0 \;\; \Rightarrow$$

$$\frac{\partial J^{\beta}(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}, \mathbf{\Phi}^{\beta*}, \mathbf{\Psi}^{\beta*})}{\partial \psi^{\beta}_{[sd]}} - Q^{\beta}_{[s.]} \geq 0 \quad \forall\, [.d] \in \mathbf{D}^{\beta}_{[s.]},\; [s.] \in \mathbf{S}^{\beta}$$

$$\frac{\partial L^{\beta}(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}, \mathbf{\Phi}^{\beta*}, \mathbf{\Psi}^{\beta*}, \mathbf{Q}^{\beta})}{\partial Q^{\beta}_{[sd]}} \geq 0 \Rightarrow$$

$$\phi^{\beta*}_{o[sd]} + \sum_{\pi[sd] \in \mathbf{\Pi}^{\beta}_{[sd]}} \phi^{\beta*}_{\pi[sd]} = 1 \qquad \forall\ [sd] \in \mathbf{SD}^{\beta}$$

$$\frac{\partial L^{\beta}(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}, \mathbf{\Phi}^{\beta*}, \mathbf{\Psi}^{\beta*}, \mathbf{Q}^{\beta})}{\partial Q^{\beta}_{[s.]}} = 0 \Rightarrow$$

$$\sum_{[.d] \in \mathbf{D}^{c}_{[s.]}} \psi^{\beta*}_{[sd]} = 1 \qquad \forall\ [s.] \in \mathbf{S}^{\beta}$$

$$\phi^{\beta*}_{o[sd]},\ \phi^{\beta*}_{\pi[sd]} \geq 0 \qquad \forall\ \pi[sd] \in \mathbf{\Pi}^{\beta}_{[sd]},\ [sd] \in \mathbf{SD}^{\beta}$$

$$\psi^{\beta*}_{[sd]} \geq 0 \qquad \forall\ [.d] \in \mathbf{D}^{\beta}_{[s.]},\ [s.] \in \mathbf{S}^{\beta}$$

The above conditions result in a convex set of flows $\phi^{\beta}_{o[sd]},\ \phi^{\beta}_{\pi[sd]},\ \psi^{\beta}_{[sd]} \quad \forall\ \pi[sd] \in \mathbf{\Pi}^{\beta}_{[sd]},\ [sd] \in \mathbf{SD}^{\beta}$. □

Theorem :

Consider the quasi-static two hierarchical class joint load sharing, routing and congestion control problem in distributed systems.

If the cost function of the most powerful class α, J^{α}, *is differentiable and convex in* $(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}) \in (\mathbf{RC}^{\alpha}, \mathbf{LS}^{\alpha})$, *for each fixed value of* $(\mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta}) \in (\mathbf{RC}^{\beta}, \mathbf{LS}^{\beta})$ *and the cost function of the less powerful class* β, J^{β}, *is differentiable and convex in* $(\mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta}) \in (\mathbf{RC}^{\beta}, \mathbf{LS}^{\beta})$, *for each fixed value of* $(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}) \in (\mathbf{RC}^{\alpha}, \mathbf{LS}^{\alpha})$,
then $(\mathbf{\Phi}^{*}, \mathbf{\Psi}^{*}) \in (\mathbf{RC}, \mathbf{LS})$ *is a Stackelberg equilibrium if and only if it solves the following the Karush-Kuhn-Tucker conditions:*

$$\left[\frac{\partial J^{\alpha *}}{\partial \phi^{\alpha}_{o[sd]}} + \sum_{[sd] \in \mathbf{SD}^{\beta}} Q^{\alpha\beta}_{o[sd]} * \frac{\partial^2 J^{\beta *}}{\partial \phi^{\alpha}_{o[sd]} \phi^{\beta}_{o[sd]}} * \phi^{\beta *}_{o[sd]} + \right.$$

$$+ \sum_{[sd] \in \mathbf{SD}^{\beta}} \sum_{\pi[sd] \in \mathbf{\Pi}^{\beta}_{[sd]}} Q^{\alpha\beta}_{\pi[sd]} * \frac{\partial^2 J^{\beta *}}{\partial \phi^{\alpha}_{o[sd]} \phi^{\beta}_{\pi[sd]}} * \phi^{\beta *}_{\pi[sd]} +$$

$$+ \sum_{[sd] \in \mathbf{SD}^{\beta}} Q^{\alpha\beta}_{[sd]} * \frac{\partial^2 J^{\beta *}}{\partial \phi^{\alpha}_{o[sd]} \psi^{\beta}_{[sd]}} * \psi^{\beta *}_{[sd]} +$$

$$+ \sum_{[sd] \in \mathbf{SD}^{\beta}} \bar{Q}^{\alpha\beta}_{o[sd]} * \left(- \frac{\partial^2 J^{\beta *}}{\partial \phi^{\alpha}_{o[sd]} \phi^{\beta}_{o[sd]}} \right) +$$

$$+ \sum_{[sd] \in \mathbf{SD}^{\beta}} \sum_{\pi[sd] \in \mathbf{\Pi}^{\beta}_{[sd]}} \bar{Q}^{\alpha\beta}_{\pi[sd]} * \left(- \frac{\partial^2 J^{\beta *}}{\partial \phi^{\alpha}_{o[sd]} \phi^{\beta}_{\pi[sd]}} \right) +$$

$$\left. + \sum_{[sd] \in \mathbf{SD}^{\beta}} \bar{Q}^{\alpha\beta}_{[sd]} * \left(- \frac{\partial^2 J^{\beta *}}{\partial \phi^{\alpha}_{o[sd]} \psi^{\beta}_{[sd]}} \right) - Q^{\alpha}_{[sd]} \right] * \phi^{\alpha *}_{o[sd]} = 0$$

$$\forall \ [sd] \in \mathbf{SD}^{\alpha}$$

$$\left[\frac{\partial J^{\alpha*}}{\partial \phi^{\alpha}_{\pi[sd]}} + \sum_{[sd]\in \mathbf{SD}^{\beta}} Q^{\alpha\beta}_{\pi[sd]} * \frac{\partial^2 J^{\beta*}}{\partial \phi^{\alpha}_{\pi[sd]} \phi^{\beta}_{o[sd]}} * \phi^{\beta*}_{o[sd]} + \right.$$

$$+ \sum_{[sd]\in \mathbf{SD}^{\beta}} \sum_{\pi[sd]\in \mathbf{\Pi}^{\beta}_{[sd]}} Q^{\alpha\beta}_{\pi[sd]} * \frac{\partial^2 J^{\beta*}}{\partial \phi^{\alpha}_{\pi[sd]} \phi^{\beta}_{\pi[sd]}} * \phi^{\beta*}_{\pi[sd]} +$$

$$+ \sum_{[sd]\in \mathbf{SD}^{\beta}} Q^{\alpha\beta}_{[sd]} * \frac{\partial^2 J^{\beta*}}{\partial \phi^{\alpha}_{\pi[sd]} \psi^{\beta}_{[sd]}} * \psi^{\beta*}_{[sd]} +$$

$$+ \sum_{[sd]\in \mathbf{SD}^{\beta}} \bar{Q}^{\alpha\beta}_{o[sd]} * \left(-\frac{\partial^2 J^{\beta*}}{\partial \phi^{\alpha}_{\pi[sd]} \phi^{\beta}_{o[sd]}}\right) +$$

$$+ \sum_{[sd]\in \mathbf{SD}^{\beta}} \sum_{\pi[sd]\in \mathbf{\Pi}^{\beta}_{[sd]}} \bar{Q}^{\alpha\beta}_{\pi[sd]} * \left(-\frac{\partial^2 J^{\beta*}}{\partial \phi^{\alpha}_{\pi[sd]} \phi^{\beta}_{\pi[sd]}}\right) +$$

$$\left. + \sum_{[sd]\in \mathbf{SD}^{\beta}} \bar{Q}^{\alpha\beta}_{[sd]} * \left(-\frac{\partial^2 J^{\beta*}}{\partial \phi^{\alpha}_{\pi[sd]} \psi^{\beta}_{[sd]}}\right) - Q^{\alpha}_{[sd]}\right] * \phi^{\alpha*}_{\pi[sd]} = 0$$

$$\forall\ \pi[sd] \in \mathbf{\Pi}^{\alpha}_{[sd]},\ [sd] \in \mathbf{SD}^{\alpha}$$

$$\left[\frac{\partial J^{\alpha *}}{\partial \psi^{\alpha}_{[sd]}} + \sum_{[sd]\in \mathbf{SD}^{\beta}} Q^{\alpha\beta}_{[sd]} * \frac{\partial^2 J^{\beta *}}{\partial \psi^{,\alpha}_{[sd]} \phi^{,\beta}_{o[sd]}} * \phi^{\beta *}_{o[sd]} + \right.$$

$$+ \sum_{[sd]\in \mathbf{SD}^{\beta}} \sum_{\pi[sd]\in \mathbf{\Pi}^{\beta}_{[sd]}} Q^{\alpha\beta}_{\pi[sd]} * \frac{\partial^2 J^{\beta *}}{\partial \psi^{,\alpha}_{[sd]} \phi^{,\beta}_{\pi[sd]}} * \phi^{\beta *}_{\pi[sd]} +$$

$$+ \sum_{[sd]\in \mathbf{SD}^{\beta}} Q^{\alpha\beta}_{[sd]} * \frac{\partial^2 J^{\beta *}}{\partial \psi^{,\alpha}_{[sd]} \psi^{,\beta}_{[sd]}} * \psi^{\beta *}_{[sd]} +$$

$$+ \sum_{[sd]\in \mathbf{SD}^{\beta}} \bar{Q}^{\alpha\beta}_{o[sd]} * \left(- \frac{\partial^2 J^{\beta *}}{\partial \psi^{\alpha}_{[sd]} \phi^{\beta}_{o[sd]}} \right) +$$

$$+ \sum_{[sd]\in \mathbf{SD}^{\beta}} \sum_{\pi[sd]\in \mathbf{\Pi}^{\beta}_{[sd]}} \bar{Q}^{\alpha\beta}_{\pi[sd]} * \left(- \frac{\partial^2 J^{\beta *}}{\partial \psi^{\alpha}_{[sd]} \phi^{\beta}_{\pi[sd]}} \right) +$$

$$\left. + \sum_{[sd]\in \mathbf{SD}^{\beta}} \bar{Q}^{\alpha\beta}_{[sd]} * \left(- \frac{\partial^2 J^{\beta *}}{\partial \psi^{\alpha}_{[sd]} \psi^{\beta}_{[sd]}} \right) - Q^{\alpha}_{[s.]} \right] * \psi^{\alpha *}_{[sd]} = 0$$

$$\forall\, [sd] \in \mathbf{SD}^{\alpha}$$

$$\frac{\partial J^{\alpha *}}{\partial \phi^{\alpha}_{o[sd]}} + \sum_{[sd] \in \mathbf{SD}^{\beta}} Q^{\alpha\beta}_{o[sd]} * \frac{\partial^2 J^{\beta *}}{\partial \phi^{\alpha}_{o[sd]} \phi^{\beta}_{o[sd]}} * \phi^{\beta *}_{o[sd]} +$$

$$+ \sum_{[sd] \in \mathbf{SD}^{\beta}} \sum_{\pi[sd] \in \mathbf{\Pi}^{\beta}_{[sd]}} Q^{\alpha\beta}_{\pi[sd]} * \frac{\partial^2 J^{\beta *}}{\partial \phi^{\alpha}_{o[sd]} \phi^{\beta}_{\pi[sd]}} * \phi^{\beta *}_{\pi[sd]} +$$

$$+ \sum_{[sd] \in \mathbf{SD}^{\beta}} Q^{\alpha\beta}_{[sd]} * \frac{\partial^2 J^{\beta *}}{\partial \phi^{\alpha}_{o[sd]} \psi^{\beta}_{[sd]}} * \psi^{\beta *}_{[sd]} +$$

$$+ \sum_{[sd] \in \mathbf{SD}^{\beta}} \bar{Q}^{\alpha\beta}_{o[sd]} * \left(- \frac{\partial^2 J^{\beta *}}{\partial \phi^{\alpha}_{o[sd]} \phi^{\beta}_{o[sd]}} \right) +$$

$$+ \sum_{[sd] \in \mathbf{SD}^{\beta}} \sum_{\pi[sd] \in \mathbf{\Pi}^{\beta}_{[sd]}} \bar{Q}^{\alpha\beta}_{\pi[sd]} * \left(- \frac{\partial^2 J^{\beta *}}{\partial \phi^{\alpha}_{o[sd]} \phi^{\beta}_{\pi[sd]}} \right) +$$

$$+ \sum_{[sd] \in \mathbf{SD}^{\beta}} \bar{Q}^{\alpha\beta}_{[sd]} * \left(- \frac{\partial^2 J^{\beta *}}{\partial \phi^{\alpha}_{o[sd]} \psi^{\beta}_{[sd]}} \right) - Q^{\alpha}_{[sd]} \geq 0$$

$$\forall \, [sd] \in \mathbf{SD}^{\alpha}$$

$$\frac{\partial J^{\alpha*}}{\partial \phi^{\alpha}_{\pi[sd]}} + \sum_{[sd]\in \mathbf{SD}^{\beta}} Q^{\alpha\beta}_{\pi[sd]} * \frac{\partial^2 J^{\beta*}}{\partial \phi^{\alpha}_{\pi[sd]} \phi^{\beta}_{o[sd]}} * \phi^{\beta*}_{\pi[sd]} +$$

$$+ \sum_{[sd]\in \mathbf{SD}^{\beta}} \sum_{\pi[sd]\in \mathbf{\Pi}^{\beta}_{[sd]}} Q^{\alpha\beta}_{\pi[sd]} * \frac{\partial^2 J^{\beta*}}{\partial \phi^{\alpha}_{\pi[sd]} \phi^{\beta}_{\pi[sd]}} * \phi^{\beta*}_{\pi[sd]} +$$

$$+ \sum_{[sd]\in \mathbf{SD}^{\beta}} Q^{\alpha\beta}_{[sd]} * \frac{\partial^2 J^{\beta*}}{\partial \phi^{\alpha}_{\pi[sd]} \psi^{\beta}_{[sd]}} * \psi^{\beta*}_{[sd]} +$$

$$+ \sum_{[sd]\in \mathbf{SD}^{\beta}} \bar{Q}^{\alpha\beta}_{o[sd]} * \left(- \frac{\partial^2 J^{\beta*}}{\partial \phi^{\alpha}_{\pi[sd]} \phi^{\beta}_{o[sd]}} \right) +$$

$$+ \sum_{[sd]\in \mathbf{SD}^{\beta}} \sum_{\pi[sd]\in \mathbf{\Pi}^{\beta}_{[sd]}} \bar{Q}^{\alpha\beta}_{\pi[sd]} * \left(- \frac{\partial^2 J^{\beta*}}{\partial \phi^{\alpha}_{\pi[sd]} \phi^{\beta}_{\pi[sd]}} \right) +$$

$$+ \sum_{[sd]\in \mathbf{SD}^{\beta}} \bar{Q}^{\alpha\beta}_{[sd]} * \left(- \frac{\partial^2 J^{\beta*}}{\partial \phi^{\alpha}_{\pi[sd]} \psi^{\beta}_{[sd]}} \right) - Q^{\alpha}_{[sd]} \geq 0$$

$$\forall\ \pi[sd] \in \mathbf{\Pi}^{\alpha}_{[sd]},\ [sd] \in \mathbf{SD}^{\alpha}$$

$$\frac{\partial J^{\alpha *}}{\partial \psi^{\alpha}_{[sd]}} + \sum_{[sd] \in \mathbf{SD}^{\beta}} Q^{\alpha\beta}_{[sd]} * \frac{\partial^2 J^{\beta *}}{\partial \psi^{\alpha}_{[sd]} \phi^{\beta}_{o[sd]}} * \phi^{\beta *}_{o[sd]} +$$

$$+ \sum_{[sd] \in \mathbf{SD}^{\beta}} \sum_{\pi[sd] \in \mathbf{\Pi}^{\beta}_{[sd]}} Q^{\alpha\beta}_{\pi[sd]} * \frac{\partial^2 J^{\beta *}}{\partial \psi^{\alpha}_{[sd]} \phi^{\beta}_{\pi[sd]}} * \phi^{\beta *}_{\pi[sd]} +$$

$$+ \sum_{[sd] \in \mathbf{SD}^{\beta}} Q^{\alpha\beta}_{[sd]} * \frac{\partial^2 J^{\beta *}}{\partial \psi^{\alpha}_{[sd]} \psi^{\beta}_{[sd]}} * \psi^{\beta *}_{[sd]} +$$

$$+ \sum_{[sd] \in \mathbf{SD}^{\beta}} \bar{Q}^{\alpha\beta}_{o[sd]} * \left(- \frac{\partial^2 J^{\beta *}}{\partial \psi^{\alpha}_{[sd]} \phi^{\beta}_{o[sd]}} \right) +$$

$$+ \sum_{[sd] \in \mathbf{SD}^{\beta}} \sum_{\pi[sd] \in \mathbf{\Pi}^{\beta}_{[sd]}} \bar{Q}^{\alpha\beta}_{\pi[sd]} * \left(- \frac{\partial^2 J^{\beta *}}{\partial \psi^{\alpha}_{[sd]} \phi^{\beta}_{\pi[sd]}} \right) +$$

$$+ \sum_{[sd] \in \mathbf{SD}^{\beta}} \bar{Q}^{\alpha\beta}_{[sd]} * \left(- \frac{\partial^2 J^{\beta *}}{\partial \psi^{\alpha}_{[sd]} \psi^{\beta}_{[sd]}} \right) - Q^{\alpha}_{[s.]} \geq 0$$

$$\forall \, [sd] \in \mathbf{SD}^{\alpha}$$

$$\left[\frac{\partial J^{\alpha *}}{\partial \phi^{\beta}_{o[sd]}} + \sum_{[sd] \in \mathbf{SD}^{\beta}} Q^{\alpha\beta}_{o[sd]} * \frac{\partial^2 J^{\beta *}}{\partial \phi^{\beta}_{o[sd]} \phi^{\beta}_{o[sd]}} * \phi^{\beta *}_{o[sd]} + \right.$$

$$+ \sum_{[sd] \in \mathbf{SD}^{\beta}} \sum_{\pi[sd] \in \mathbf{\Pi}^{\beta}_{[sd]}} Q^{\alpha\beta}_{\pi[sd]} * \frac{\partial^2 J^{\beta *}}{\partial \phi^{\beta}_{o[sd]} \phi^{\beta}_{\pi[sd]}} * \phi^{\beta *}_{\pi[sd]} +$$

$$+ \sum_{[sd] \in \mathbf{SD}^{\beta}} Q^{\alpha\beta}_{[sd]} * \frac{\partial^2 J^{\beta *}}{\partial \phi^{\beta}_{o[sd]} \psi^{\beta}_{[sd]}} * \psi^{\beta *}_{[sd]} +$$

$$+ \sum_{[sd] \in \mathbf{SD}^{\beta}} \bar{Q}^{\alpha\beta}_{o[sd]} * \left(- \frac{\partial^2 J^{\beta *}}{\partial \phi^{\beta}_{o[sd]} \phi^{\beta}_{o[sd]}} \right) +$$

$$+ \sum_{[sd] \in \mathbf{SD}^{\beta}} \sum_{\pi[sd] \in \mathbf{\Pi}^{\beta}_{[sd]}} \bar{Q}^{\alpha\beta}_{\pi[sd]} * \left(- \frac{\partial^2 J^{\beta *}}{\partial \phi^{\beta}_{o[sd]} \phi^{\beta}_{\pi[sd]}} \right) +$$

$$\left. + \sum_{[sd] \in \mathbf{SD}^{\beta}} \bar{Q}^{\alpha\beta}_{[sd]} * \left(- \frac{\partial^2 J^{\beta *}}{\partial \phi^{\beta}_{o[sd]} \psi^{\beta}_{[sd]}} \right) - Q^{\beta}_{[sd]} \right] * \phi^{\beta *}_{o[sd]} = 0$$

$$\forall \ [sd] \in \mathbf{SD}^{\beta}$$

$$\left[\frac{\partial J^{\alpha *}}{\partial \phi^{\beta}_{\pi[sd]}} + \sum_{[sd]\in \mathbf{SD}^{\beta}} Q^{\alpha\beta}_{\pi[sd]} * \frac{\partial^2 J^{\beta *}}{\partial \phi^{\beta}_{\pi[sd]} \phi^{\beta}_{o[sd]}} * \phi^{\beta *}_{\pi[sd]} + \right.$$

$$+ \sum_{[sd]\in \mathbf{SD}^{\beta}} \sum_{\pi[sd]\in \mathbf{\Pi}^{\beta}_{[sd]}} Q^{\alpha\beta}_{\pi[sd]} * \frac{\partial^2 J^{\beta *}}{\partial \phi^{\beta}_{\pi[sd]} \phi^{\beta}_{\pi[sd]}} * \phi^{\beta *}_{\pi[sd]} +$$

$$+ \sum_{[sd]\in \mathbf{SD}^{\beta}} Q^{\alpha\beta}_{[sd]} * \frac{\partial^2 J^{\beta *}}{\partial \phi^{\beta}_{\pi[sd]} \psi^{\beta}_{[sd]}} * \psi^{\beta *}_{[sd]} +$$

$$+ \sum_{[sd]\in \mathbf{SD}^{\beta}} \bar{Q}^{\alpha\beta}_{o[sd]} * \left(-\frac{\partial^2 J^{\beta *}}{\partial \phi^{\beta}_{\pi[sd]} \phi^{\beta}_{o[sd]}}\right) +$$

$$+ \sum_{[sd]\in \mathbf{SD}^{\beta}} \sum_{\pi[sd]\in \mathbf{\Pi}^{\beta}_{[sd]}} \bar{Q}^{\alpha\beta}_{\pi[sd]} * \left(-\frac{\partial^2 J^{\beta *}}{\partial \phi^{\beta}_{\pi[sd]} \phi^{\beta}_{\pi[sd]}}\right) +$$

$$\left. + \sum_{[sd]\in \mathbf{SD}^{\beta}} \bar{Q}^{\alpha\beta}_{[sd]} * \left(-\frac{\partial^2 J^{\beta *}}{\partial \phi^{\beta}_{\pi[sd]} \psi^{\beta}_{[sd]}}\right) - Q^{\beta}_{[sd]}\right] * \phi^{\beta *}_{\pi[sd]} = 0$$

$$\forall\ \pi[sd] \in \mathbf{\Pi}^{\beta}_{[sd]},\ [sd] \in \mathbf{SD}^{\beta}$$

$$\left[\frac{\partial J^{\alpha *}}{\partial \psi^{\beta}_{[sd]}} + \sum_{[sd]\in \mathbf{SD}^{\beta}} Q^{\alpha\beta}_{[sd]} * \frac{\partial^2 J^{\beta *}}{\partial \psi^{\beta}_{[sd]} \phi^{\beta}_{o[sd]}} * \phi^{\beta *}_{o[sd]} + \right.$$

$$+ \sum_{[sd]\in \mathbf{SD}^{\beta}} \sum_{\pi[sd]\in \mathbf{\Pi}^{\beta}_{[sd]}} Q^{\alpha\beta}_{\pi[sd]} * \frac{\partial^2 J^{\beta *}}{\partial \psi^{\beta}_{[sd]} \phi^{\beta}_{\pi[sd]}} * \phi^{\beta *}_{\pi[sd]} +$$

$$+ \sum_{[sd]\in \mathbf{SD}^{\beta}} Q^{\alpha\beta}_{[sd]} * \frac{\partial^2 J^{\beta *}}{\partial \psi^{\beta}_{[sd]} \psi^{\beta}_{[sd]}} * \psi^{\beta *}_{[sd]} +$$

$$+ \sum_{[sd]\in \mathbf{SD}^{\beta}} \bar{Q}^{\alpha\beta}_{o[sd]} * \left(-\frac{\partial^2 J^{\beta *}}{\partial \psi^{\beta}_{[sd]} \phi^{\beta}_{o[sd]}}\right) +$$

$$+ \sum_{[sd]\in \mathbf{SD}^{\beta}} \sum_{\pi[sd]\in \mathbf{\Pi}^{\beta}_{[sd]}} \bar{Q}^{\alpha\beta}_{\pi[sd]} * \left(-\frac{\partial^2 J^{\beta *}}{\partial \psi^{\beta}_{[sd]} \phi^{\beta}_{\pi[sd]}}\right) +$$

$$\left. + \sum_{[sd]\in \mathbf{SD}^{\beta}} \bar{Q}^{\alpha\beta}_{[sd]} * \left(-\frac{\partial^2 J^{\beta *}}{\partial \psi^{\beta}_{[sd]} \psi^{\beta}_{[sd]}}\right) - Q^{\beta}_{[s.]}\right] * \psi^{\beta *}_{[sd]} = 0$$

$$\forall\ [sd] \in \mathbf{SD}^{\beta}$$

$$\frac{\partial J^{\alpha *}}{\partial \phi^{\beta}_{o[sd]}} + \sum_{[sd] \in \mathbf{SD}^{\beta}} Q^{\alpha\beta}_{o[sd]} * \frac{\partial^2 J^{\beta *}}{\partial \phi^{\beta}_{o[sd]} \phi^{\beta}_{o[sd]}} * \phi^{\beta *}_{o[sd]} +$$

$$+ \sum_{[sd] \in \mathbf{SD}^{\beta}} \sum_{\pi[sd] \in \mathbf{\Pi}^{\beta}_{[sd]}} Q^{\alpha\beta}_{\pi[sd]} * \frac{\partial^2 J^{\beta *}}{\partial \phi^{\beta}_{o[sd]} \phi^{\beta}_{\pi[sd]}} * \phi^{\beta *}_{\pi[sd]} +$$

$$+ \sum_{[sd] \in \mathbf{SD}^{\beta}} Q^{\alpha\beta}_{[sd]} * \frac{\partial^2 J^{\beta *}}{\partial \phi^{\beta}_{o[sd]} \psi^{\beta}_{[sd]}} * \psi^{\beta *}_{[sd]} +$$

$$+ \sum_{[sd] \in \mathbf{SD}^{\beta}} \bar{Q}^{\alpha\beta}_{o[sd]} * \left(- \frac{\partial^2 J^{\beta *}}{\partial \phi^{\beta}_{o[sd]} \phi^{\beta}_{o[sd]}} \right) +$$

$$+ \sum_{[sd] \in \mathbf{SD}^{\beta}} \sum_{\pi[sd] \in \mathbf{\Pi}^{\beta}_{[sd]}} \bar{Q}^{\alpha\beta}_{\pi[sd]} * \left(- \frac{\partial^2 J^{\beta *}}{\partial \phi^{\beta}_{o[sd]} \phi^{\beta}_{\pi[sd]}} \right) +$$

$$+ \sum_{[sd] \in \mathbf{SD}^{\beta}} \bar{Q}^{\alpha\beta}_{[sd]} * \left(- \frac{\partial^2 J^{\beta *}}{\partial \phi^{\beta}_{o[sd]} \psi^{\beta}_{[sd]}} \right) - Q^{\beta}_{[sd]} \geq 0$$

$$\forall \; [sd] \in \mathbf{SD}^{\beta}$$

$$\frac{\partial J^{\alpha *}}{\partial \phi^{\beta}_{\pi[sd]}} + \sum_{[sd]\in \mathbf{SD}^{\beta}} Q^{\alpha\beta}_{\pi[sd]} * \frac{\partial^2 J^{\beta *}}{\partial \phi^{\beta}_{\pi[sd]} \phi^{\beta}_{o[sd]}} * \phi^{\beta *}_{\pi[sd]} +$$

$$+ \sum_{[sd]\in \mathbf{SD}^{\beta}} \sum_{\pi[sd]\in \mathbf{\Pi}^{\beta}_{[sd]}} Q^{\alpha\beta}_{\pi[sd]} * \frac{\partial^2 J^{\beta *}}{\partial \phi^{\beta}_{\pi[sd]} \phi^{\beta}_{\pi[sd]}} * \phi^{\beta *}_{\pi[sd]} +$$

$$+ \sum_{[sd]\in \mathbf{SD}^{\beta}} Q^{\alpha\beta}_{[sd]} * \frac{\partial^2 J^{\beta *}}{\partial \phi^{\beta}_{\pi[sd]} \psi^{\beta}_{[sd]}} * \psi^{\beta *}_{[sd]} +$$

$$+ \sum_{[sd]\in \mathbf{SD}^{\beta}} \bar{Q}^{\alpha\beta}_{o[sd]} * \left(- \frac{\partial^2 J^{\beta *}}{\partial \phi^{\beta}_{\pi[sd]} \phi^{\beta}_{o[sd]}} \right) +$$

$$+ \sum_{[sd]\in \mathbf{SD}^{\beta}} \sum_{\pi[sd]\in \mathbf{\Pi}^{\beta}_{[sd]}} \bar{Q}^{\alpha\beta}_{\pi[sd]} * \left(- \frac{\partial^2 J^{\beta *}}{\partial \phi^{\beta}_{\pi[sd]} \phi^{\beta}_{\pi[sd]}} \right) +$$

$$+ \sum_{[sd]\in \mathbf{SD}^{\beta}} \bar{Q}^{\alpha\beta}_{[sd]} * \left(- \frac{\partial^2 J^{\beta *}}{\partial \phi^{\beta}_{\pi[sd]} \psi^{\beta}_{[sd]}} \right) - Q^{\beta}_{[sd]} \geq 0$$

$$\forall\ \pi[sd] \in \mathbf{\Pi}^{\beta}_{[sd]},\ [sd] \in \mathbf{SD}^{\beta}$$

$$\frac{\partial J^{\alpha *}}{\partial \psi^{\beta}_{[sd]}} + \sum_{[sd] \in \mathbf{SD}^{\beta}} Q^{\alpha\beta}_{[sd]} * \frac{\partial^2 J^{\beta *}}{\partial \psi'^{\beta}_{[sd]} \phi^{\beta}_{o[sd]}} * \phi^{\beta *}_{o[sd]} +$$

$$+ \sum_{[sd] \in \mathbf{SD}^{\beta}} \sum_{\pi[sd] \in \mathbf{\Pi}^{\beta}_{[sd]}} Q^{\alpha\beta}_{\pi[sd]} * \frac{\partial^2 J^{\beta *}}{\partial \psi'^{\beta}_{[sd]} \phi^{\beta}_{\pi[sd]}} * \phi^{\beta *}_{\pi[sd]} +$$

$$+ \sum_{[sd] \in \mathbf{SD}^{\beta}} Q^{\alpha\beta}_{[sd]} * \frac{\partial^2 J^{\beta *}}{\partial \psi'^{\beta}_{[sd]} \psi'^{\beta}_{[sd]}} * \psi^{\beta *}_{[sd]} +$$

$$+ \sum_{[sd] \in \mathbf{SD}^{\beta}} \bar{Q}^{\alpha\beta}_{o[sd]} * \left(- \frac{\partial^2 J^{\beta *}}{\partial \psi'^{\beta}_{[sd]} \phi^{\beta}_{o[sd]}} \right) +$$

$$+ \sum_{[sd] \in \mathbf{SD}^{\beta}} \sum_{\pi[sd] \in \mathbf{\Pi}^{\beta}_{[sd]}} \bar{Q}^{\alpha\beta}_{\pi[sd]} * \left(- \frac{\partial^2 J^{\beta *}}{\partial \psi'^{\beta}_{[sd]} \phi^{\beta}_{\pi[sd]}} \right) +$$

$$+ \sum_{[sd] \in \mathbf{SD}^{\beta}} \bar{Q}^{\alpha\beta}_{[sd]} * \left(- \frac{\partial^2 J^{\beta *}}{\partial \psi'^{\beta}_{[sd]} \psi'^{\beta}_{[sd]}} \right) - Q^{\beta}_{[s.]} \geq 0$$

$$\forall \; [sd] \in \mathbf{SD}^{\beta}$$

$$\phi^{\alpha*}_{o[sd]} + \sum_{\pi[sd]\in\mathbf{\Pi}^{\alpha}_{[sd]}} \phi^{\alpha*}_{\pi[sd]} = 1 \quad \forall\ [sd] \in \mathbf{SD}^{\alpha}$$

$$\sum_{[.d]\in\mathbf{D}^{c}_{[s.]}} \psi^{\alpha*}_{[sd]} = 1 \qquad \forall\ [s.] \in \mathbf{S}^{\alpha}$$

$$\phi^{\alpha*}_{o[sd]},\ \phi^{\alpha*}_{\pi[sd]} \geq 0 \qquad \forall\ \pi[sd] \in \mathbf{\Pi}^{\alpha}_{[sd]},\ [sd] \in \mathbf{SD}^{\alpha}$$

$$\psi^{\alpha*}_{[sd]} \geq 0 \qquad \forall\ [.d] \in \mathbf{D}^{\alpha}_{[s.]},\ [s.] \in \mathbf{S}^{\alpha}$$

$$\phi^{\beta*}_{o[sd]} + \sum_{\pi[sd]\in\mathbf{\Pi}^{\beta}_{[sd]}} \phi^{\beta*}_{\pi[sd]} = 1 \quad \forall\ [sd] \in \mathbf{SD}^{\beta}$$

$$\sum_{[.d]\in\mathbf{D}^{c}_{[s.]}} \psi^{\beta*}_{[sd]} = 1 \qquad \forall\ [s.] \in \mathbf{S}^{\beta}$$

$$\phi^{\beta*}_{o[sd]},\ \phi^{\beta*}_{\pi[sd]} \geq 0 \qquad \forall\ \pi[sd] \in \mathbf{\Pi}^{\beta}_{[sd]},\ [sd] \in \mathbf{SD}^{\beta}$$

$$\psi^{\beta*}_{[sd]} \geq 0 \qquad \forall\ [.d] \in \mathbf{D}^{\beta}_{[s.]},\ [s.] \in \mathbf{S}^{\beta}$$

$$\bar{Q}^{\alpha\beta}_{o[sd]} * \left[Q^{\beta}_{[sd]} - \frac{\partial J^{\beta*}}{\partial \phi^{\beta}_{o[sd]}} \right] = 0 \quad \forall\ [sd] \in \mathbf{SD}^{\beta}$$

$$\bar{Q}^{\alpha\beta}_{\pi[sd]} * \left[Q^{\beta}_{[sd]} - \frac{\partial J^{\beta*}}{\partial \phi^{\beta}_{\pi[sd]}} \right] = 0 \quad \forall\ \pi[sd] \in \mathbf{\Pi}^{\beta}_{[sd]},\ [sd] \in \mathbf{SD}^{\beta}$$

$$\bar{Q}^{\alpha\beta}_{[sd]} * \left[Q^{\beta}_{[s.]} - \frac{\partial J^{\beta*}}{\partial \psi^{\beta}_{[sd]}} \right] = 0 \qquad \forall\ [sd] \in \mathbf{SD}^{\beta}$$

Proof: The Lagrangian for the most powerful class α, when the less powerful class β has achieved its optimum decisions, is

$$L^{\alpha\beta} = J^{\alpha} + \sum_{[sd]\in\mathbf{SD}^{\beta}} Q^{\alpha\beta}_{o[sd]} * \left[\frac{\partial J^{\beta}}{\partial \phi^{\beta}_{o[sd]}} - Q^{\beta}_{[sd]}\right] * \phi^{\beta}_{o[sd]} +$$

$$+ \sum_{[sd]\in\mathbf{SD}^{\beta}} \sum_{\pi[sd]\in\mathbf{\Pi}^{\beta}_{[sd]}} Q^{\alpha\beta}_{\pi[sd]} * \left[\frac{\partial J^{\beta}}{\partial \phi^{\beta}_{\pi[sd]}} - Q^{\beta}_{[sd]}\right] * \phi^{\beta}_{\pi[sd]} +$$

$$+ \sum_{[s.]\in\mathbf{S}^{\beta}} \sum_{[.d]\in\mathbf{D}^{\beta}_{[s.]}} Q^{\alpha\beta}_{[sd]} * \left[\frac{\partial J^{\beta}}{\partial \psi^{\beta}_{[sd]}} - Q^{\beta}_{[s.]}\right] * \psi^{\beta}_{[sd]} +$$

$$+ \sum_{[sd]\in\mathbf{SD}^{\beta}} \bar{Q}^{\alpha\beta}_{o[sd]} * \left[Q^{\beta}_{[sd]} - \frac{\partial J^{\beta}}{\partial \phi^{\beta}_{o[sd]}}\right] +$$

$$+ \sum_{[sd]\in\mathbf{SD}^{\beta}} \sum_{\pi[sd]\in\mathbf{\Pi}^{\beta}_{[sd]}} \bar{Q}^{\alpha\beta}_{\pi[sd]} * \left[Q^{\beta}_{[sd]} - \frac{\partial J^{\beta}}{\partial \phi^{\beta}_{\pi[sd]}}\right] +$$

$$+ \sum_{[s.]\in\mathbf{S}^{\beta}} \sum_{[.d]\in\mathbf{D}^{\beta}_{[s.]}} \bar{Q}^{\alpha\beta}_{[sd]} * \left[Q^{\beta}_{[s.]} - \frac{\partial J^{\beta}}{\partial \psi^{\beta}_{[sd]}}\right] +$$

$$+ \sum_{[sd]\in\mathbf{SD}^{\alpha}} Q^{\alpha}_{[sd]} * \left[1 - \phi^{\alpha}_{o[sd]} - \sum_{\pi[sd]\in\mathbf{\Pi}^{\alpha}_{[sd]}} \phi^{\alpha}_{\pi[sd]}\right] +$$

$$+ \sum_{[s.]\in\mathbf{S}^{\alpha}} Q^{\alpha}_{[s.]} * \left[1 - \sum_{[.d]\in\mathbf{D}^{\alpha}_{[s.]}} \psi^{\alpha}_{[sd]}\right] +$$

$$+ \sum_{[sd]\in\mathbf{SD}^{\beta}} Q^{\beta}_{[sd]} * \left[1 - \phi^{\beta}_{o[sd]} - \sum_{\pi[sd]\in\mathbf{\Pi}^{\beta}_{[sd]}} \phi^{\beta}_{\pi[sd]}\right] +$$

$$+ \sum_{[s.]\in\mathbf{S}^{\beta}} Q^{\beta}_{[s.]} * \left[1 - \sum_{[.d]\in\mathbf{D}^{\beta}_{[s.]}} \psi^{\beta}_{[sd]}\right]$$

$$with \quad \phi^{\alpha}_{o[sd]},\ \phi^{\alpha}_{\pi[sd]},\ \psi^{\alpha}_{[sd]} \geq 0 \qquad \forall\ \pi[sd] \in \mathbf{\Pi}^{\alpha}_{[sd]},\ [sd] \in \mathbf{SD}^{\alpha}$$

$$\phi^{\beta}_{o[sd]},\ \phi^{\beta}_{\pi[sd]},\ \psi^{\beta}_{[sd]} \geq 0 \qquad \forall\ \pi[sd] \in \mathbf{\Pi}^{\beta}_{[sd]},\ [sd] \in \mathbf{SD}^{\beta}$$

$$\bar{Q}^{\alpha\beta}_{o[sd]},\ \bar{Q}^{\alpha\beta}_{\pi[sd]},\ \bar{Q}^{\alpha\beta}_{[sd]} \geq 0 \qquad \forall\ \pi[sd] \in \mathbf{\Pi}^{\beta}_{[sd]},\ [sd] \in \mathbf{SD}^{\beta}$$

$$\bar{Q}^{\alpha\beta}_{o[sd]} * \left[Q^{\beta}_{[sd]} - \frac{\partial J^{\beta}}{\partial \phi^{\beta}_{o[sd]}} \right] = 0 \quad \forall\ [sd] \in \mathbf{SD}^{\beta}$$

$$\bar{Q}^{\alpha\beta}_{\pi[sd]} * \left[Q^{\beta}_{[sd]} - \frac{\partial J^{\beta}}{\partial \phi^{\beta}_{\pi[sd]}} \right] = 0 \quad \forall\ \pi[sd] \in \mathbf{\Pi}^{\beta}_{[sd]},\ [sd] \in \mathbf{SD}^{\beta}$$

$$\bar{Q}^{\alpha\beta}_{[sd]} * \left[Q^{\beta}_{[s.]} - \frac{\partial J^{\beta}}{\partial \psi^{\beta}_{[sd]}} \right] = 0 \qquad \forall\ [sd] \in \mathbf{SD}^{\beta}$$

the Karush-Kuhn-Tucker necessary conditions are

$$\frac{\partial L^{\alpha\beta *}}{\partial \phi^{\alpha}_{o[sd]}} * \phi^{\alpha *}_{o[sd]} = 0, \quad \frac{\partial L^{\alpha\beta *}}{\partial \phi^{\alpha}_{\pi[sd]}} * \phi^{\alpha *}_{\pi[sd]} = 0, \quad \frac{\partial L^{\alpha\beta *}}{\partial \psi^{\alpha}_{[sd]}} * \psi^{\alpha *}_{[sd]} = 0$$

$$\frac{\partial L^{\alpha\beta *}}{\partial \phi^{\alpha}_{o[sd]}} \geq 0, \quad \frac{\partial L^{\alpha\beta *}}{\partial \phi^{\alpha}_{\pi[sd]}} \geq 0, \quad \frac{\partial L^{\alpha\beta *}}{\partial \psi^{\alpha}_{[sd]}} \geq 0$$

$$\frac{\partial L^{\alpha\beta *}}{\partial \phi^{\beta}_{o[sd]}} * \phi^{\beta *}_{o[sd]} = 0, \quad \frac{\partial L^{\alpha\beta *}}{\partial \phi^{\beta}_{\pi[sd]}} * \phi^{\beta *}_{\pi[sd]} = 0, \quad \frac{\partial L^{\alpha\beta *}}{\partial \psi^{\beta}_{[sd]}} * \psi^{\beta *}_{[sd]} = 0$$

$$\frac{\partial L^{\alpha\beta *}}{\partial \phi^{\beta}_{o[sd]}} \geq 0, \quad \frac{\partial L^{\alpha\beta *}}{\partial \phi^{\beta}_{\pi[sd]}} \geq 0 \quad \frac{\partial L^{\alpha\beta *}}{\partial \psi^{\beta}_{[sd]}} \geq 0$$

$$\frac{\partial L^{\alpha\beta}}{\partial Q^{\alpha}_{[sd]}} = 0 \quad \frac{\partial L^{\alpha\beta}}{\partial Q^{\alpha}_{[s.]}} = 0 \quad \phi^{\alpha*}_{o[sd]},\ \phi^{\alpha*}_{\pi[sd]},\ \psi^{\alpha*}_{[sd]} \geq 0$$

$$\frac{\partial L^{\alpha\beta}}{\partial Q^{\beta}_{[sd]}} = 0 \quad \frac{\partial L^{\alpha\beta}}{\partial Q^{\beta}_{[s.]}} = 0 \quad \phi^{\beta*}_{o[sd]},\ \phi^{\beta*}_{\pi[sd]},\ \psi^{\beta*}_{[sd]} \geq 0$$

$$\bar{Q}^{\alpha\beta}_{o[sd]} * \left[Q^{\beta}_{[sd]} - \frac{\partial J^{\beta*}}{\partial \phi^{\beta}_{o[sd]}} \right] = 0$$

$$\bar{Q}^{\alpha\beta}_{\pi[sd]} * \left[Q^{\beta}_{[sd]} - \frac{\partial J^{\beta*}}{\partial \phi^{\beta}_{\pi[sd]}} \right] = 0$$

$$\bar{Q}^{\alpha\beta}_{[sd]} * \left[Q^{\beta}_{[s.]} - \frac{\partial J^{\beta*}}{\partial \psi^{\beta*}_{[sd]}} \right] = 0$$

Evaluating the above expressions, we have the theorem.□

Next, for each class c, we define the length for the rejected flow $[sd]$, the length for the path $\pi[sd]$ and the length for the source-destination pair $[sd]$:

$$l^{\alpha,Stack}_{o[sd]} = \frac{\partial J^{\alpha}}{\partial \phi^{\alpha}_{o[sd]}} + \sum_{[sd]\in \mathbf{SD}^{\beta}} Q^{\alpha\beta}_{o[sd]} * \frac{\partial^2 J^{\beta}}{\partial \phi^{\alpha}_{o[sd]} \phi^{\beta}_{o[sd]}} * \phi^{\beta}_{o[sd]} +$$

$$+ \sum_{[sd]\in \mathbf{SD}^{\beta}} \sum_{\pi[sd]\in \mathbf{\Pi}^{\beta}_{[sd]}} Q^{\alpha\beta}_{\pi[sd]} * \frac{\partial^2 J^{\beta}}{\partial \phi^{\alpha}_{o[sd]} \phi^{\beta}_{\pi[sd]}} * \phi^{\beta}_{\pi[sd]} +$$

$$+ \sum_{[sd]\in \mathbf{SD}^{\beta}} Q^{\alpha\beta}_{[sd]} * \frac{\partial^2 J^{\beta}}{\partial \phi^{\alpha}_{o[sd]} \psi^{\beta}_{[sd]}} * \psi^{\beta}_{[sd]} +$$

$$+ \sum_{[sd]\in \mathbf{SD}^{\beta}} \bar{Q}^{\alpha\beta}_{o[sd]} * \left(- \frac{\partial^2 J^{\beta}}{\partial \phi^{\alpha}_{o[sd]} \phi^{\beta}_{o[sd]}} \right) +$$

$$+ \sum_{[sd]\in \mathbf{SD}^{\beta}} \sum_{\pi[sd]\in \mathbf{\Pi}^{\beta}_{[sd]}} \bar{Q}^{\alpha\beta}_{\pi[sd]} * \left(- \frac{\partial^2 J^{\beta}}{\partial \phi^{\alpha}_{o[sd]} \phi^{\beta}_{\pi[sd]}} \right) +$$

$$+ \sum_{[sd]\in \mathbf{SD}^{\beta}} \bar{Q}^{\alpha\beta}_{[sd]} * \left(- \frac{\partial^2 J^{\beta}}{\partial \phi^{\alpha}_{o[sd]} \psi^{\beta}_{[sd]}} \right)$$

$$
\begin{aligned}
l^{\alpha,Stack}_{\pi[sd]} = & \frac{\partial J^{\alpha}}{\partial \phi^{\alpha}_{\pi[sd]}} + \sum_{[sd]\in \mathbf{SD}^{\beta}} Q^{\alpha\beta}_{\pi[sd]} * \frac{\partial^2 J^{\beta}}{\partial \phi^{\alpha}_{\pi[sd]} \phi^{\beta}_{o[sd]}} * \phi^{\beta}_{o[sd]} + \\
& + \sum_{[sd]\in \mathbf{SD}^{\beta}} \sum_{\pi[sd]\in \mathbf{\Pi}^{\beta}_{[sd]}} Q^{\alpha\beta}_{\pi[sd]} * \frac{\partial^2 J^{\beta}}{\partial \phi^{\alpha}_{\pi[sd]} \phi^{\beta}_{\pi[sd]}} * \phi^{\beta}_{\pi[sd]} + \\
& + \sum_{[sd]\in \mathbf{SD}^{\beta}} Q^{\alpha\beta}_{[sd]} * \frac{\partial^2 J^{\beta}}{\partial \phi^{\alpha}_{\pi[sd]} \psi^{\beta}_{[sd]}} * \psi^{\beta}_{[sd]} + \\
& + \sum_{[sd]\in \mathbf{SD}^{\beta}} \bar{Q}^{\alpha\beta}_{o[sd]} * \left(- \frac{\partial^2 J^{\beta}}{\partial \phi^{\alpha}_{\pi[sd]} \phi^{\beta}_{o[sd]}} \right) + \\
& + \sum_{[sd]\in \mathbf{SD}^{\beta}} \sum_{\pi[sd]\in \mathbf{\Pi}^{\beta}_{[sd]}} \bar{Q}^{\alpha\beta}_{\pi[sd]} * \left(- \frac{\partial^2 J^{\beta}}{\partial \phi^{\alpha}_{\pi[sd]} \phi^{\beta}_{\pi[sd]}} \right) + \\
& + \sum_{[sd]\in \mathbf{SD}^{\beta}} \bar{Q}^{\alpha\beta}_{[sd]} * \left(- \frac{\partial^2 J^{\beta}}{\partial \phi^{\alpha}_{\pi[sd]} \psi^{\beta}_{[sd]}} \right)
\end{aligned}
$$

$$
\begin{aligned}
l^{\alpha,Stack}_{[sd]} = \quad & \frac{\partial J^{\alpha}}{\partial \psi^{\alpha}_{[sd]}} + \sum_{[sd]\in \mathbf{SD}^{\beta}} Q^{\alpha\beta}_{[sd]} * \frac{\partial^2 J^{\beta}}{\partial \psi^{\alpha}_{[sd]} \phi^{\beta}_{o[sd]}} * \phi^{\beta}_{o[sd]} + \\
& + \sum_{[sd]\in \mathbf{SD}^{\beta}} \sum_{\pi[sd]\in \mathbf{\Pi}^{\beta}_{[sd]}} Q^{\alpha\beta}_{\pi[sd]} * \frac{\partial^2 J^{\beta}}{\partial \psi^{\alpha}_{[sd]} \phi^{\beta}_{\pi[sd]}} * \phi^{\beta}_{\pi[sd]} + \\
& + \sum_{[sd]\in \mathbf{SD}^{\beta}} Q^{\alpha\beta}_{[sd]} * \frac{\partial^2 J^{\beta}}{\partial \psi^{\alpha}_{[sd]} \partial \phi^{\beta}_{\pi[sd]}} * \psi^{\beta}_{[sd]} + \\
& + \sum_{[sd]\in \mathbf{SD}^{\beta}} \bar{Q}^{\alpha\beta}_{o[sd]} * \left(- \frac{\partial^2 J^{\beta}}{\partial \psi^{\alpha}_{[sd]} \partial \phi^{\beta}_{o[sd]}} \right) + \\
& + \sum_{[sd]\in \mathbf{SD}^{\beta}} \sum_{\pi[sd]\in \mathbf{\Pi}^{\beta}_{[sd]}} \bar{Q}^{\alpha\beta}_{\pi[sd]} * \left(- \frac{\partial^2 J^{\beta}}{\partial \psi^{\alpha}_{[sd]} \partial \phi^{\beta}_{\pi[sd]}} \right) + \\
& + \sum_{[sd]\in \mathbf{SD}^{\beta}} \bar{Q}^{\alpha\beta}_{[sd]} * \left(- \frac{\partial^2 J^{\beta}}{\partial \psi^{\alpha}_{[sd]} \psi^{\beta}_{[sd]}} \right)
\end{aligned}
$$

$$l^{\beta,Stack}_{o[sd]} = \frac{\partial J^{\alpha}}{\partial \phi^{\beta}_{o[sd]}} + \sum_{[sd]\in \mathbf{SD}^{\beta}} Q^{\alpha\beta}_{o[sd]} * \frac{\partial^2 J^{\beta}}{\partial \phi^{\beta}_{o[sd]} \phi^{\beta}_{o[sd]}} +$$

$$+ \sum_{[sd]\in \mathbf{SD}^{\beta}} \sum_{\pi[sd]\in \mathbf{\Pi}^{\beta}_{[sd]}} Q^{\alpha\beta}_{\pi[sd]} * \frac{\partial^2 J^{\beta}}{\partial \phi^{\beta}_{o[sd]} \partial \phi^{\beta}_{\pi[sd]}} * \phi^{\beta}_{\pi[sd]} +$$

$$+ \sum_{[sd]\in \mathbf{SD}^{\beta}} Q^{\alpha\beta}_{[sd]} * \frac{\partial^2 J^{\beta}}{\partial \phi^{\beta}_{o[sd]} \psi^{\beta}_{[sd]}} * \psi^{\beta}_{[sd]} +$$

$$+ \sum_{[sd]\in \mathbf{SD}^{\beta}} \bar{Q}^{\alpha\beta}_{o[sd]} * \left(- \frac{\partial^2 J^{\beta}}{\partial \phi^{\beta}_{o[sd]} \phi^{\beta}_{o[sd]}} \right) +$$

$$+ \sum_{[sd]\in \mathbf{SD}^{\beta}} \sum_{\pi[sd]\in \mathbf{\Pi}^{\beta}_{[sd]}} \bar{Q}^{\alpha\beta}_{\pi[sd]} * \left(- \frac{\partial^2 J^{\beta}}{\partial \phi^{\beta}_{o[sd]} \phi^{\beta}_{\pi[sd]}} \right) +$$

$$+ \sum_{[sd]\in \mathbf{SD}^{\beta}} \bar{Q}^{\alpha\beta}_{[sd]} * \left(- \frac{\partial^2 J^{\beta}}{\partial \phi^{\beta}_{o[sd]} \psi^{\beta}_{[sd]}} \right)$$

$$
\begin{aligned}
l^{\beta,Stack}_{\pi[sd]} = & \frac{\partial J^{\alpha}}{\partial \phi^{\beta}_{\pi[sd]}} + \sum_{[sd]\in \mathbf{SD}^{\beta}} Q^{\alpha\beta}_{\pi[sd]} * \frac{\partial^2 J^{\beta}}{\partial \phi^{\beta}_{\pi[sd]} \phi^{\beta}_{o[sd]}} * \phi^{\beta}_{\pi[sd]} + \\
& + \sum_{[sd]\in \mathbf{SD}^{\beta}} \sum_{\pi[sd]\in \mathbf{\Pi}^{\beta}_{[sd]}} Q^{\alpha\beta}_{\pi[sd]} * \frac{\partial^2 J^{\beta}}{\partial \phi^{\beta}_{\pi[sd]} \phi^{\beta}_{\pi[sd]}} * \phi^{\beta}_{\pi[sd]} + \\
& + \sum_{[sd]\in \mathbf{SD}^{\beta}} Q^{\alpha\beta}_{[sd]} * \frac{\partial^2 J^{\beta}}{\partial \phi^{\beta}_{\pi[sd]} \psi^{\beta}_{[sd]}} * \psi^{\beta}_{[sd]} + \\
& + \sum_{[sd]\in \mathbf{SD}^{\beta}} \bar{Q}^{\alpha\beta}_{o[sd]} * \left(- \frac{\partial^2 J^{\beta}}{\partial \phi^{\beta}_{\pi[sd]} \phi^{\beta}_{o[sd]}} \right) + \\
& + \sum_{[sd]\in \mathbf{SD}^{\beta}} \sum_{\pi[sd]\in \mathbf{\Pi}^{\beta}_{[sd]}} \bar{Q}^{\alpha\beta}_{\pi[sd]} * \left(- \frac{\partial^2 J^{\beta}}{\partial \phi^{\beta}_{\pi[sd]} \phi^{\beta}_{\pi[sd]}} \right) + \\
& + \sum_{[sd]\in \mathbf{SD}^{\beta}} \bar{Q}^{\alpha\beta}_{[sd]} * \left(- \frac{\partial^2 J^{\beta}}{\partial \phi^{\beta}_{\pi[sd]} \psi^{\beta}_{[sd]}} \right)
\end{aligned}
$$

$$l^{\beta,Stack}_{[sd]} = \frac{\partial J^{\alpha}}{\partial \psi'^{\beta}_{[sd]}} + \sum_{[sd]\in \mathbf{SD}^{\beta}} Q^{\alpha\beta}_{[sd]} * \frac{\partial^2 J^{\beta}}{\partial \psi'^{\beta}_{[sd]} \phi^{\beta}_{o[sd]}} * \phi^{\beta}_{o[sd]} +$$

$$+ \sum_{[sd]\in \mathbf{SD}^{\beta}} \sum_{\pi[sd]\in \mathbf{\Pi}^{\beta}_{[sd]}} Q^{\alpha\beta}_{\pi[sd]} * \frac{\partial^2 J^{\beta}}{\partial \psi'^{\beta}_{[sd]} \phi^{\beta}_{\pi[sd]}} * \phi^{\beta}_{\pi[sd]} +$$

$$+ \sum_{[sd]\in \mathbf{SD}^{\beta}} Q^{\alpha\beta}_{[sd]} * \frac{\partial^2 J^{\beta}}{\partial \psi'^{\beta}_{[sd]} \psi'^{\beta}_{[sd]}} * \psi'^{\beta}_{[sd]} +$$

$$+ \sum_{[sd]\in \mathbf{SD}^{\beta}} \bar{Q}^{\alpha\beta}_{o[sd]} * \left(- \frac{\partial^2 J^{\beta}}{\partial \psi^{\beta}_{[sd]} \phi^{\beta}_{o[sd]}} \right) +$$

$$+ \sum_{[sd]\in \mathbf{SD}^{\beta}} \sum_{\pi[sd]\in \mathbf{\Pi}^{\beta}_{[sd]}} \bar{Q}^{\alpha\beta}_{\pi[sd]} * \left(- \frac{\partial^2 J^{\beta}}{\partial \psi^{\beta}_{[sd]} \phi^{\beta}_{\pi[sd]}} \right) +$$

$$+ \sum_{[sd]\in \mathbf{SD}^{\beta}} \bar{Q}^{\alpha\beta}_{[sd]} * \left(- \frac{\partial^2 J^{\beta}}{\partial \psi^{\beta}_{[sd]} \psi^{\beta}_{[sd]}} \right)$$

External arriving flow at a source is assigned to the destination that has the minimum length from the source. However, this flow may be rejected if the length of rejecting it is less than the lengths of the paths to its destination. If it is accepted, then it is routed to its destination via the minimum length path.

Theorem :

Consider the quasi-static two hierarchical class joint load sharing, routing and congestion control problem in distributed systems.

If the cost function of the most powerful class α, J^{α}, *is differentiable and convex in* $(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}) \in (\mathbf{RC}^{\alpha}, \mathbf{LS}^{\alpha})$, *for each fixed value of* $(\mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta}) \in (\mathbf{RC}^{\beta}, \mathbf{LS}^{\beta})$ *and the cost function of the less powerful class* β, J^{β}, *is differentiable and convex in* $(\mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta}) \in (\mathbf{RC}^{\beta}, \mathbf{LS}^{\beta})$, *for each fixed value of* $(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}) \in (\mathbf{RC}^{\alpha}, \mathbf{LS}^{\alpha})$,

then $(\mathbf{\Phi}^*, \mathbf{\Psi}^*) \in (\mathbf{RC}, \mathbf{LS})$ *is a Stackelberg equilibrium if and only if it satisfies the following conditions:*

congestion control

$$\phi^{\alpha*}_{o[sd]} > 0 \ only \ if \quad l^{\alpha,Stack*}_{o[sd]} = min\{l^{\alpha,Stack*}_{o[sd]}, \min_{p[sd]}\{l^{\alpha,Stack*}_{p[sd]}\}\}$$

$$\phi^{\alpha*}_{o[sd]} = 0 \qquad o.w.$$

$$\forall \ [sd] \in SD^{\alpha}$$

routing

$$\phi^{\alpha*}_{\pi[sd]} > 0 \ only \ if \quad l^{\alpha,Stack*}_{\pi[sd]} = min\{l^{\alpha,Stack*}_{o[sd]}, \min_{p[sd]}\{l^{\alpha,Stack*}_{p[sd]}\}\}$$

$$\phi^{\alpha*}_{\pi[sd]} = 0 \qquad o.w.$$

$$\forall \ \pi[sd] \in \mathbf{\Pi}^{\alpha}_{[sd]}, \ [sd] \in SD^{\alpha}$$

load sharing

$$\psi^{\alpha*}_{[sd]} > 0 \ only \ if \quad l^{\alpha,Stack*}_{[sd]} = \min_{[sd']}\{l^{\alpha,Stack*}_{[sd']}\}$$

$$\psi^{\alpha*}_{[sd']} = 0 \qquad o.w.$$

$$\forall \ [.d] \in \mathbf{D}^{\alpha}_{[s.]}, \ [sd] \in SD^{\alpha}$$

congestion control

$$\Phi^{\beta*}_{o[sd]} > 0 \ only \ if \quad l^{\beta,Stack*}_{o[sd]} = min\{l^{\beta,Stack*}_{o[sd]}, \min_{p[sd]}\{l^{\beta,Stack*}_{p[sd]}\}\}$$

$$\Phi^{\beta*}_{o[sd]} = 0 \qquad o.w.$$

$$\forall \ [sd] \in SD^{\beta}$$

routing

$$\phi^{\beta*}_{\pi[sd]} > 0 \ only \ if \quad l^{\beta,Stack*}_{\pi[sd]} = min\{l^{\beta,Stack*}_{o[sd]}, \min_{p[sd]}\{l^{\beta,Stack*}_{p[sd]}\}\}$$

$$\phi^{\beta*}_{\pi[sd]} = 0 \qquad o.w.$$

$$\forall \ \pi[sd] \in \Pi^{\beta}_{[sd]}, \ [sd] \in SD^{\beta}$$

load sharing

$$\psi^{\beta*}_{[sd]} > 0 \ only \ if \quad l^{\beta,Stack*}_{[sd]} = \min_{[sd']}\{l^{\beta,Stack*}_{[sd']}\}$$

$$\psi^{\beta*}_{[sd']} = 0 \qquad o.w.$$

$$\forall \ [.d] \in \mathbf{D}^{\beta}_{[s.]}, \ [sd] \in SD^{\beta}$$

$$\phi^{\alpha*}_{o[sd]} + \sum_{\pi[sd]\in\mathbf{\Pi}^{\alpha}_{[sd]}} \phi^{\alpha*}_{\pi[sd]} = 1 \quad \forall\ [sd] \in \mathbf{SD}^{\alpha}$$

$$\sum_{[.d]\in\mathbf{D}^{c}_{[s.]}} \psi^{\alpha*}_{[sd]} = 1 \qquad \forall\ [s.] \in \mathbf{S}^{\alpha}$$

$$\phi^{\beta*}_{o[sd]} + \sum_{\pi[sd]\in\mathbf{\Pi}^{\beta}_{[sd]}} \phi^{\beta*}_{\pi[sd]} = 1 \quad \forall\ [sd] \in \mathbf{SD}^{\beta}$$

$$\sum_{[.d]\in\mathbf{D}^{c}_{[s.]}} \psi^{\beta*}_{[sd]} = 1 \qquad \forall\ [s.] \in \mathbf{S}^{\beta}$$

$$\bar{Q}^{\alpha\beta}_{o[sd]} * \left[Q^{\beta}_{[sd]} - \frac{\partial J^{\beta*}}{\partial \phi^{\beta}_{o[sd]}} \right] = 0 \quad \forall\ [sd] \in \mathbf{SD}^{\beta}$$

$$\bar{Q}^{\alpha\beta}_{\pi[sd]} * \left[Q^{\beta}_{[sd]} - \frac{\partial J^{\beta*}}{\partial \phi^{\beta}_{\pi[sd]}} \right] = 0 \quad \forall\ \pi[sd] \in \mathbf{\Pi}^{\beta}_{[sd]},\ [sd] \in \mathbf{SD}^{\beta}$$

$$\bar{Q}^{\alpha\beta}_{[sd]} * \left[Q^{\beta}_{[s.]} - \frac{\partial J^{\beta*}}{\partial \psi^{\beta}_{[sd]}} \right] = 0 \qquad \forall\ [sd] \in \mathbf{SD}^{\beta}$$

Proof: It follows from the Karush-Kuhn-Tucker conditions.□

Theorem :

Consider the quasi-static two hierarchical class joint load sharing, routing and congestion control problem in distributed systems. If the cost function of the less powerful class β, J^{β}, is strictly convex in $(\mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta}) \in (\mathbf{RC}^{\beta}, \mathbf{LS}^{\beta})$, for each fixed value of $(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}) \in (\mathbf{RC}^{\alpha}, \mathbf{LS}^{\alpha})$,

then the most powerful class achieves smaller cost if it acts as a leader in the Stackelberg game formulation than if it participates in the Nash game formulation.

Proof: This is a well known result for Stackelberg games [27].□

4.3.2 Nonlinear Complementarity Problem Formulation

In this section, we formulate the class β optimization problem as a Nonlinear Complementarity Problem (NCP).

Define the vector for class β of the congestion control, routing and load sharing fractions as well as Lagrange multipliers:

$$\mathbf{Z}^{\beta} = \left[\ldots \phi^{\beta}_{o[sd]} \ldots \phi^{\beta}_{\pi[sd]} \ldots Q^{\beta}_{[sd]} \ldots\ldots \psi^{\beta}_{[sd]} \ldots Q^{\beta}_{[s.]} \ldots \right]^{T}$$

and the vector for class β of the first derivatives of the Lagrangian with respect to the congestion control, routing and load sharing fractions as well as Lagrange multipliers:

$$\nabla L^{\beta}(\mathbf{Z}^{\beta}) = \left[\ldots \left(\frac{\partial J^{\beta}}{\partial \phi^{\beta}_{o[sd]}} - Q^{\beta}_{o[sd]} \right) \ldots \left(\frac{\partial J^{\beta}}{\partial \phi^{\beta}_{\pi[sd]}} - Q^{\beta}_{\pi[sd]} \right) \ldots \right\}$$

$$\ldots \left(1 - \phi^{\beta}_{o[sd]} - \sum_{\pi[sd] \in \mathbf{\Pi}^{\beta}_{[sd]}} \phi^{\beta}_{\pi[sd]} \right) \ldots$$

$$\ldots \left(\frac{\partial J^{\beta}}{\partial \psi^{\beta}_{[sd]}} - Q^{\beta}_{[s.]} \right) \ldots \left(1 - \sum_{[.d] \in \mathbf{D}^{\beta}_{[s.]}} \psi^{\beta}_{[sd]} \right) \ldots \right]$$

Theorem :

Consider the quasi-static two hierarchical class joint load sharing, routing and congestion control problem in distributed systems. If J^β *is continuously differentiable and convex with respect to* $(\mathbf{\Phi}^\beta, \mathbf{\Psi}^\beta) \in (\mathbf{RC}^\beta, \mathbf{LS}^\beta)$.

then $(\mathbf{\Phi}^*, \mathbf{\Psi}^*) \in (\mathbf{RC}, \mathbf{LS})$ *is a Stackelberg equilibrium if and only if it solves the following problem*

$$\begin{aligned}
&\textit{minimize} && J^\alpha(\mathbf{\Phi}^\alpha, \mathbf{\Psi}^\alpha, \mathbf{\Phi}^\beta, \mathbf{\Psi}^\beta) \\
&\textit{with respect to} && (\mathbf{\Phi}^\alpha, \mathbf{\Psi}^\alpha, \mathbf{\Phi}^\beta, \mathbf{\Psi}^\beta, \mathbf{Q}^\beta) \\
&\textit{such that} && (\mathbf{\Phi}^\alpha, \mathbf{\Psi}^\alpha) \in (\mathbf{RC}^\alpha, \mathbf{LS}^\alpha) \qquad (\mathbf{\Phi}^\beta, \mathbf{\Psi}^\beta) \in (\mathbf{RC}^\beta, \mathbf{LS}^\beta) \\
& && \nabla L^\beta(\mathbf{Z}^\beta) * \mathbf{Z}^\beta = 0 \\
& && \nabla L^\beta(\mathbf{Z}^\beta) \geq 0 \\
& && \mathbf{Z}^\beta \geq 0
\end{aligned}$$

Proof: After some algebraic manipulations, we find that the NCP: $\nabla L(\mathbf{Z}) * \mathbf{Z} = 0$; $\nabla L(\mathbf{Z}) \geq 0$; $\mathbf{Z} \geq 0$ with $\mathbf{Z}$ and $\nabla L(\mathbf{Z})$ as defined above, is equivalent to the Karush-Kuhn-Tucker necessary and sufficient conditions for the follower's minimization problem. □

4.3.3 Variational Inequality Formulation

In this section, we formulate the class β optimization problem as a Variational Inequality Problem (VIP).

Define the vector for class β congestion control, routing and load sharing fractions:

$$(\mathbf{\Phi}^\beta, \mathbf{\Psi}^\beta) = \left[\ldots \; \phi^\beta_{o[sd]} \; \cdots \; \phi^\beta_{\pi[sd]} \; \cdots \; \psi^\beta_{[sd]} \; \ldots \right]^T$$

and the vector for class β derivatives of its cost function with respect to the congestion control, routing and load sharing fractions:

$$\nabla J^{\beta}(\mathbf{\Phi}, \mathbf{\Psi}) = \left[\cdots \frac{\partial J^{\beta}}{\partial \phi^{\beta}_{o[sd]}} \cdots \sum_{\pi[sd] \in \mathbf{\Pi}^{\beta}_{[sd]}} \frac{\partial J^{\beta}}{\partial \phi^{\beta}_{\pi[sd]}} \cdots \frac{\partial J^{\beta}}{\partial \psi^{\beta}_{[sd]}} \cdots \right]$$

Theorem :

Consider the quasi-static two hierarchical class joint load sharing, routing and congestion control problem in distributed systems.
If J^{β} *is continuously differentiable and convex in* $(\mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta}) \in (\mathbf{RC}^{\beta}, \mathbf{LS}^{\beta})$.
then $(\mathbf{\Phi}^{*}, \mathbf{\Psi}^{*}) \in (\mathbf{RC}, \mathbf{LS})$ *is a Stackelberg equilibrium if and only if it solves the following problem:*

$$\begin{aligned}
&\textit{minimize} && J^{\alpha}(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}, \mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta}) \\
&\textit{with respect to} && (\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}, \mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta}, \mathbf{Q}^{\beta}) \\
&\textit{such that} && (\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}) \in (\mathbf{RC}^{\alpha}, \mathbf{LS}^{\alpha}) \qquad (\mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta}) \in (\mathbf{RC}^{\beta}, \mathbf{LS}^{\beta}) \\
& && \nabla J^{\beta}(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}, \mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta}) * ((\hat{\mathbf{\Phi}}^{\beta}, \hat{\mathbf{\Psi}}^{\beta}) - (\mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta})) \geq 0 \\
& && \forall\ (\hat{\mathbf{\Phi}}^{\beta}, \hat{\mathbf{\Psi}}^{\beta}) \in (\mathbf{RC}^{\beta}, \mathbf{LS}^{\beta})
\end{aligned}$$

Proof: If $(\mathbf{\Phi}^{\beta *}, \mathbf{\Psi}^{\beta *})$ is a local minimum for the follower's minimization problem

$$\begin{aligned}
&\textit{minimize} && J^{\beta}(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}, \mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta}) \\
&\textit{with respect to} && (\mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta}) \\
&\textit{such that} && (\mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta}) \in (\mathbf{RC}^{\beta}, \mathbf{LS}^{\beta})
\end{aligned}$$

and J^{β} is a continuously differentiable convex function over the nonempty convex, closed and bounded set $(\mathbf{RC}^{\beta}, \mathbf{LS}^{\beta})$, then

$$\sum_{[sd]\in \mathbf{SD}^{\beta}} \left\{ \frac{\partial J^{\beta}(\mathbf{\Phi}^{*}, \mathbf{\Psi}^{*})}{\partial \phi^{\beta}_{o[sd]}} * (\phi^{\beta}_{o[sd]} - \phi^{\beta *}_{o[sd]}) + \right.$$

$$+ \sum_{\pi[sd]\in \mathbf{\Pi}^{\beta}_{[sd]}} \frac{\partial J^{\beta}(\mathbf{\Phi}^{*}, \mathbf{\Psi}^{*})}{\partial \phi^{\beta}_{\pi[sd]}} * (\phi^{\beta}_{\pi[sd]} - \phi^{\beta *}_{\pi[sd]}) +$$

$$\left. + \frac{\partial J^{\beta}(\mathbf{\Phi}^{*}, \mathbf{\Psi}^{*})}{\partial \psi^{\beta}_{[sd]}} * (\psi^{\beta}_{[sd]} - \psi^{\beta *}_{[sd]}) \right\} \geq 0 \qquad \forall\, (\mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta}) \in (\mathbf{RC}^{\beta}, \mathbf{LS}^{\beta})$$

□

Another equivalent formulation is given in the following Theorem:

Theorem :

Consider the quasi-static two hierarchical class joint load sharing, routing and congestion control problem in distributed systems. If J^{β} *is continuously differentiable and convex with respect to* $(\mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta}) \in (\mathbf{RC}^{\beta}, \mathbf{LS}^{\beta})$, *then* $(\mathbf{\Phi}^{*}, \mathbf{\Psi}^{*}) \in (\mathbf{RC}, \mathbf{LS})$ *is a Stackelberg equilibrium if and only if it solves the following* Variational Inequality Problem*:*

$$\begin{array}{ll} \textit{minimize} & J^{\alpha}(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}, \mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta}) \\ \textit{with respect to} & (\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}, \mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta}) \\ \textit{such that} & (\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}) \in (\mathbf{RC}^{\alpha}, \mathbf{LS}^{\alpha}) \qquad (\mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta}) \in (\mathbf{RC}^{\beta}, \mathbf{LS}^{\beta}) \\ & \nabla L^{\beta}(\mathbf{Z}^{\beta}) * (\hat{\mathbf{Z}}^{\beta} - \mathbf{Z}^{\beta}) \geq 0 \qquad \forall\ \hat{\mathbf{Z}}^{\beta} > 0 \end{array}$$

Proof: Karamardian shows that the NCP: $f(x^{*}) * x^{*} = 0; \quad f(x^{*}) \geq 0; \quad x^{*} > 0$ and the VIP: find x^{*} such that $f(x^{*}) * (x - x^{*}) \geq 0 \quad \forall x > 0$ are equivalent.□

4.3.4 K-K-T for Separable Cost Functions

In this section, we derive the first order necessary and sufficient conditions for a Stackelberg equilibrium on the path flows, when the cost function at each network resource depends only on the flow on this and is convex with respect to that flow.

The partial derivatives of the cost function $J^{\alpha}(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}, \mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta})$ with respect to the path fractions $\phi^{\alpha}_{\pi[sd]}$ can be written as the with respect to the link flows λ^{α}_{ij} and node flows λ^{α}_{i}:

$$\frac{\partial J^{\alpha}_{ij}(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}, \mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta})}{\partial \phi^{\alpha}_{\pi[sd]}} = \frac{\partial J^{\alpha}_{ij}(\mathbf{\Lambda}_{ij})}{\partial \lambda^{\alpha}_{ij}} * \frac{\partial \lambda^{\alpha}_{ij}}{\partial \phi^{\alpha}_{\pi[sd]}}$$

$$= \frac{\partial J^{\alpha}_{ij}(\mathbf{\Lambda}_{ij})}{\partial \lambda^{\alpha}_{ij}} * (\gamma^{\alpha}_{[sd]} + \gamma^{\alpha}_{[s.]} * \psi^{\alpha}_{[sd]}) * 1_{ij \in \pi[sd]}$$

$$\frac{\partial J^{\alpha}_{i}(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}, \mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta})}{\partial \phi^{\alpha}_{\pi[sd]}} = \frac{\partial J^{\alpha}_{i}(\mathbf{\Lambda}_{i})}{\partial \lambda^{\alpha}_{i}} * \frac{\partial \lambda^{\alpha}_{i}}{\partial \phi^{\alpha}_{\pi[sd]}}$$

$$= \frac{\partial J^{\alpha}_{i}(\mathbf{\Lambda}_{i})}{\partial \lambda^{\alpha}_{i}} * (\gamma^{\alpha}_{[sd]} + \gamma^{\alpha}_{[s.]} * \psi^{\alpha}_{[sd]}) * 1_{i \in \pi[sd]}$$

$$\frac{\partial J^{\alpha}(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}, \mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta})}{\partial \phi^{\alpha}_{o[sd]}} = \frac{\partial J^{\alpha}_{[sd]}(\mathbf{\Lambda}_{o[sd]})}{\partial \lambda^{\alpha}_{o[sd]}} * \frac{\partial \lambda^{\alpha}_{o[sd]}}{\partial \phi^{\alpha}_{o[sd]}}$$

$$= \frac{\partial J^{\alpha}_{[sd]}(\mathbf{\Lambda}_{o[sd]})}{\partial \lambda^{\alpha}_{o[sd]}} * (\gamma^{\alpha}_{[sd]} + \gamma^{\alpha}_{[s.]} * \psi^{\alpha}_{[sd]})$$

$$\frac{\partial J_{ij}^{\alpha}(\mathbf{\Phi}^{\alpha},\mathbf{\Psi}^{\alpha},\mathbf{\Phi}^{\beta},\mathbf{\Psi}^{\beta})}{\partial \psi_{[sd]}^{\alpha}} = \frac{\partial J_{ij}^{\alpha}(\mathbf{\Lambda}_{ij})}{\partial \lambda_{ij}^{\alpha}} * \frac{\partial \lambda_{ij}^{\alpha}}{\partial \psi_{[sd]}^{\alpha}}$$

$$= \sum_{\pi[sd]\in\mathbf{\Pi}_{[sd]}^{\alpha}} \frac{\partial J_{ij}^{\alpha}(\mathbf{\Lambda}_{ij})}{\partial \lambda_{ij}^{\alpha}} * \gamma_{[s.]}^{\alpha} * \phi_{\pi[sd]}^{\alpha} * 1_{ij\in\pi[sd]}$$

$$\frac{\partial J_{i}^{\alpha}(\mathbf{\Phi}^{\alpha},\mathbf{\Psi}^{\alpha},\mathbf{\Phi}^{\beta},\mathbf{\Psi}^{\beta})}{\partial \psi_{[sd]}^{\alpha}} = \frac{\partial J_{i}^{\alpha}(\mathbf{\Lambda}_{i})}{\partial \lambda_{i}^{\alpha}} * \frac{\partial \lambda_{i}^{\alpha}}{\partial \psi_{[sd]}^{\alpha}}$$

$$= \sum_{\pi[sd]\in\mathbf{\Pi}_{[sd]}^{\alpha}} \frac{\partial J_{i}^{\alpha}(\mathbf{\Lambda}_{i})}{\partial \lambda_{i}^{\alpha}} * \gamma_{[s.]}^{\alpha} * \phi_{\pi[sd]}^{\alpha} * 1_{i\in\pi[sd]}$$

$$\frac{\partial J_{[sd]}^{\alpha}(\mathbf{\Phi}^{\alpha},\mathbf{\Psi}^{\alpha},\mathbf{\Phi}^{\beta},\mathbf{\Psi}^{\beta})}{\partial \psi_{[sd]}^{\alpha}} = \frac{\partial J_{[sd]}^{\alpha}(\mathbf{\Lambda}_{o[sd]})}{\partial \lambda_{o[sd]}^{\alpha}} * \frac{\partial \lambda_{o[sd]}^{\alpha}}{\partial \psi_{[sd]}^{\alpha}}$$

$$= \frac{\partial J_{[sd]}^{\alpha}(\mathbf{\Lambda}_{o[sd]})}{\partial \lambda_{o[sd]}^{\alpha}} * \gamma_{[s.]}^{\alpha} * \phi_{o[sd]}^{\alpha} =$$

$$\frac{\partial J_{[.d]}^{\alpha}(\mathbf{\Phi}^{\alpha},\mathbf{\Psi}^{\alpha},\mathbf{\Phi}^{\beta},\mathbf{\Psi}^{\beta})}{\partial \psi_{[sd]}^{\alpha}} = \frac{\partial J_{[.d]}^{\alpha}(\mathbf{\Lambda}_{[.d]})}{\partial \lambda_{[.d]}^{\alpha}} * \frac{\partial \lambda_{[.d]}^{\alpha}}{\partial \psi_{[sd]}^{\alpha}}$$

$$= \frac{\partial J_{[.d]}^{\alpha}(\mathbf{\Lambda}_{[.d]})}{\partial \lambda_{[.d]}^{\alpha}} * \gamma_{[s.]}^{\alpha}$$

The partial derivatives of the cost function $J^{\beta}(\mathbf{\Phi}, \mathbf{\Psi})$ with respect to the path fractions $\phi^{\beta}_{\pi[sd]}$ can be written with respect to the link flows λ^{β}_{ij} and node flows λ^{β}_{i}:

$$\frac{\partial J^{\beta}_{ij}(\mathbf{\Phi}, \mathbf{\Psi})}{\partial \phi^{\beta}_{\pi[sd]}} = \frac{\partial J^{\beta}_{ij}(\mathbf{\Lambda}_{ij})}{\partial \lambda^{\beta}_{ij}} * \frac{\partial \lambda^{\beta}_{ij}}{\partial \phi^{\beta}_{\pi[sd]}}$$

$$= \frac{\partial J^{\beta}_{ij}(\mathbf{\Lambda}_{ij})}{\partial \lambda^{\beta}_{ij}} * (\gamma^{\beta}_{[sd]} + \gamma^{\beta}_{[s.]} * \psi^{\beta}_{[sd]}) * 1_{ij \in \pi[sd]}$$

$$\frac{\partial J^{\beta}_{i}(\mathbf{\Phi}, \mathbf{\Psi})}{\partial \phi^{\beta}_{\pi[sd]}} = \frac{\partial J^{\beta}_{i}(\mathbf{\Lambda}_{i})}{\partial \lambda^{\beta}_{i}} * \frac{\partial \lambda^{\beta}_{i}}{\partial \phi^{\beta}_{\pi[sd]}}$$

$$= \frac{\partial J^{\beta}_{i}(\mathbf{\Lambda}_{i})}{\partial \lambda^{\beta}_{i}} * (\gamma^{\beta}_{[sd]} + \gamma^{\beta}_{[s.]} * \psi^{\beta}_{[sd]}) * 1_{i \in \pi[sd]}$$

$$\frac{\partial J^{\beta}_{[sd](\mathbf{\Phi}, \mathbf{\Psi})}}{\partial \phi^{\beta}_{o[sd]}} = \frac{\partial J^{\beta}_{[sd]}(\mathbf{\Lambda}_{o[sd]})}{\partial \lambda^{\beta}_{o[sd]}} * \frac{\partial \lambda^{\beta}_{o[sd]}}{\partial \phi^{\beta}_{o[sd]}}$$

$$= \frac{\partial J^{\beta}_{[sd]}(\mathbf{\Lambda}_{o[sd]})}{\partial \lambda^{\beta}_{o[sd]}} * (\gamma^{\beta}_{[sd]} + \gamma^{\beta}_{[s.]} * \psi^{\beta}_{[sd]})$$

$$\frac{\partial J_{ij}^{\beta}(\mathbf{\Phi},\mathbf{\Psi})}{\partial \psi_{[sd]}^{\beta}} = \frac{\partial J_{ij}^{\beta}(\mathbf{\Lambda}_{ij})}{\partial \lambda_{ij}^{\beta}} * \frac{\partial \lambda_{ij}^{\beta}}{\partial \psi_{[sd]}^{\beta}}$$

$$= \sum_{\pi[sd]\in\mathbf{\Pi}_{[sd]}^{\beta}} \frac{\partial J_{ij}^{\beta}(\mathbf{\Lambda}_{ij})}{\partial \lambda_{ij}^{\beta}} * \gamma_{[s.]}^{\beta} * \phi_{\pi[sd]}^{\beta} * 1_{ij\in\pi[sd]}$$

$$\frac{\partial J_{i}^{\beta}(\mathbf{\Phi},\mathbf{\Psi})}{\partial \psi_{[sd]}^{\beta}} = \frac{\partial J_{i}^{\beta}(\mathbf{\Lambda}_{i})}{\partial \lambda_{i}^{\beta}} * \frac{\partial \lambda_{i}^{\beta}}{\partial \psi_{[sd]}^{\beta}}$$

$$= \sum_{\pi[sd]\in\mathbf{\Pi}_{[sd]}^{\beta}} \frac{\partial J_{i}^{\beta}(\mathbf{\Lambda}_{i})}{\partial \lambda_{i}^{\beta}} * \gamma_{[s.]}^{\beta} * \phi_{\pi[sd]}^{\beta} * 1_{i\in\pi[sd]}$$

$$\frac{\partial J_{[sd]}^{\beta}(\mathbf{\Phi},\mathbf{\Psi})}{\partial \psi_{[sd]}^{\beta}} = \frac{\partial J_{[sd]}^{\beta}(\mathbf{\Lambda}_{o[sd]})}{\partial \lambda_{o[sd]}^{\beta}} * \frac{\partial \lambda_{o[sd]}^{\beta}}{\partial \psi_{[sd]}^{\beta}} = \frac{\partial J_{[sd]}^{\beta}(\mathbf{\Lambda}_{o[sd]})}{\partial \lambda_{o[sd]}^{\beta}} * \gamma_{[s.]}^{\beta} * \phi_{o[sd]}^{\beta}$$

$$\frac{\partial J_{[.d]}^{\beta}(\mathbf{\Phi},\mathbf{\Psi})}{\partial \psi_{[sd]}^{\beta}} = \frac{\partial J_{[.d]}^{\beta}(\mathbf{\Lambda}_{[.d]})}{\partial \lambda_{[.d]}^{\beta}} * \frac{\partial \lambda_{[.d]}^{\beta}}{\partial \psi_{[sd]}^{\beta}} = \frac{\partial J_{[.d]}^{\beta}(\mathbf{\Lambda}_{[.d]})}{\partial \lambda_{[.d]}^{\beta}} * \gamma_{[s.]}^{\beta}$$

Substituting into the lengths defined in section 4.3.1, we have the length for the rejected flow $[sd]$, the length for the path $\pi[sd]$ and the length for the source-destination pair $[sd]$.

4.4 Application to Datagram Networks

In this section, we apply the methodologies developed in the previous sections to datagram networks.

4.4.1 Cost Functions for Multiple Classes

Numerous books have appeared on queueing models [334, 116, 414, 477, 399, 235, 251, 255, 115, 10, 427, 112, 202, 332, 185, 127, 516], on queueing networks [249, 77, 288, 291, 185, 332, 111] on their use in analysis and synthesis of computer networks and systems [184, 263, 488, 438, 149, 214, 471] and on network optimization problems [256, 431, 484, 45, 432, 505]. In this section, we propose using performance measures derived by modeling the system using such queueing models. Next, we give an example for $M/G/1$ queues. Let the total arrival rate to the system be λ. Consider a system resource (node, link, computing site, etc.) ij with service rate C_{ij}. Let also τ_{ij} be a flow independent constant delay (e.g. propagation delay) at resource ij. Packets from class c arrive at rate λ_{ij}^c (Poisson) and require service (general distribution) with mean $1/\mu$ and second moment $\overline{x^2}$. We may take as cost function for class c at resource ij the weighted average packet delay [255, 256, 45]

$$J_{ij}^c = \frac{\lambda_{ij}^c}{\lambda} * \left[\frac{1}{\mu C_{ij}} + \frac{\sum_k \lambda_{ij}^k * \overline{x^2} * \mu C_{ij}}{2 * (\mu C_{ij} - \sum_k \lambda_{ij}^k)} + \tau_{ij} \right]$$

The first and second derivatives of J_{ij}^c with respect to λ_{ij}^c are

$$\frac{\partial J_{ij}^c}{\partial \lambda_{ij}^c} = \frac{1}{\lambda * \mu C_{ij}} + \frac{\overline{x^2} * \mu C_{ij}}{2\lambda} * \frac{\lambda_{ij}^c * \mu C_{ij} + \sum_k \lambda_{ij}^k * (\mu C_{ij} - \sum_k \lambda_{ij}^k)}{(\mu C_{ij} - \sum \lambda_{ij}^k)^2} + \frac{\tau_{ij}}{\lambda}$$

$$\frac{\partial^2 J_{ij}^c}{\partial (\lambda_{ij}^c)^2} = \frac{\overline{x^2} * \mu C_{ij}}{2\lambda} * \frac{2\lambda_{ij}^c * \mu C_{ij} + \mu C_{ij} * (\mu C_{ij} - \sum_k \lambda_{ij}^k)}{(\mu C_{ij} - \sum_k \lambda_{ij}^k)^3}$$

Consider a system resource (node, link, computing site, etc.) ij with m_{ij} servers, each at service rate C_{ij}. Let also τ_{ij} be a flow independent constant delay (e.g. propagation delay) at resource ij. Packets from class c arrive at rate λ_{ij}^c (Poisson) and require service (exponential distribution) with mean $1/\mu$. We may take as cost function for class c at resource ij the weighted average packet delay [255, 256, 45]

$$J_{ij}^c = \frac{\lambda_{ij}^c}{\lambda} * \left[\frac{1}{\mu C_{ij}} + \frac{P_{Q,ij}}{m_{ij} * \mu C_{ij} - \sum_k \lambda_{ij}^k} + \tau_{ij} \right]$$

where $P_{Q,ij}$ is Erlang's C formula (probability of queueing) (see section 4.8.1).

Of course, considering other queueing models, we can also define other cost functions (see section 4.9.2)

4.4.2 Cost Functions for Priority Classes

Consider a system resource (node, link, computing site, etc.) ij with service rate C_{ij}. Let also τ_{ij} be a flow independent constant delay (e.g. propagation delay) at resource ij. Packets from class c arrive at rate λ_{ij}^c (Poisson) and require service (general distribution) with mean $1/\mu^c$ and second moment $\overline{(x^c)^2}$. Classes $1, 2, ..., c-1$ have non-preemptive priority over class c, while class c has non-preemptive priority over classes $c+1, c+2, ...$ We may take as cost function for class c at resource ij its weighted average packet delay [255, 256, 45]

$$J_{ij}^c = \frac{\lambda_{ij}^c}{\lambda} * \left[\frac{1}{\mu^c C_{ij}} + \frac{\sum_{k=1}^{C} \lambda_{ij}^k * \overline{(x^k)^2}}{2 * \left(1 - \sum_{k=1}^{c-1} \frac{\lambda_{ij}^k}{\mu^k C_{ij}}\right) * \left(1 - \sum_{k=1}^{c} \frac{\lambda_{ij}^k}{\mu^k C_{ij}}\right)} + \tau_{ij} \right]$$

Its first and second derivatives with respect to λ_{ij}^c are

$$\frac{\partial J_{ij}^c}{\partial \lambda_{ij}^c} = \frac{1}{\lambda * \mu^c C_{ij}} + \frac{\left[\sum_{k=1}^{C} \lambda_{ij}^k * \overline{(x^k)^2} + \lambda_{ij}^c * \overline{(x^c)^2}\right]}{2\lambda * \left(1 - \sum_{k=1}^{c-1} \frac{\lambda_{ij}^k}{\mu^k C_{ij}}\right) * \left(1 - \sum_{k=1}^{c} \frac{\lambda_{ij}^k}{\mu^k C_{ij}}\right)} +$$

$$+ \frac{\frac{\lambda_{ij}^c}{\mu^c C_{ij}} * \sum_{k=1}^{C} \lambda_{ij}^k * \overline{(x^k)^2}}{2\lambda * \left(1 - \sum_{k=1}^{c-1} \frac{\lambda_{ij}^k}{\mu^k C_{ij}}\right) * \left(1 - \sum_{k=1}^{c} \frac{\lambda_{ij}^k}{\mu^k C_{ij}}\right)^2} + \frac{\tau_{ij}}{\lambda}$$

$$\frac{\partial^2 J_{ij}^c}{\partial (\lambda_{ij}^c)^2} = \frac{1}{\lambda * \left(1 - \sum_{k=1}^{c-1} \frac{\lambda_{ij}^k}{\mu^k C_{ij}}\right)} * \left\{ \frac{\overline{(x^c)^2}}{1 - \sum_{k=1}^{c} \frac{\lambda_{ij}^k}{\mu^k C_{ij}}} + \right.$$

$$\left. + \frac{\sum_{k=1}^{C} \lambda_{ij}^k * \overline{(x^k)^2} + \lambda_{ij}^c * \overline{(x^c)^2}}{\mu^c C_{ij} * \left(1 - \sum_{k=1}^{c} \frac{\lambda_{ij}^k}{\mu^k C_{ij}}\right)^2} + \frac{\lambda_{ij}^c * \sum_{k=1}^{C} \lambda_{ij}^k * \overline{(x^k)^2}}{(\mu^c C_{ij})^2 * \left(1 - \sum_{k=1}^{c} \frac{\lambda_{ij}^k}{\mu^k C_{ij}}\right)^3} \right\}$$

Consider a system resource (node, link, computing site, etc.) ij with service rate C_{ij}. Let also τ_{ij} be a flow independent constant delay (e.g. propagation delay) at resource ij. Packets from class c arrive at rate λ_{ij}^c (Poisson) and require service (general distribution) with mean $1/\mu^c$ and second moment $\overline{(x^c)^2}$. Classes $1, 2, ..., c-1$ have preemptive priority over class c, while class c has preemptive priority over classes $c+1, c+2, ...$ We may take as cost function for class c at resource ij its weighted average packet delay [255, 256, 45]

$$g_{ij}^c = \frac{\lambda_{ij}^c}{\lambda} * \left[\frac{\frac{1}{\mu^c C_{ij}} * \left(1 - \sum_{k=1}^{c} \frac{\lambda_{ij}^k}{\mu^k C_{ij}}\right) + \frac{1}{2} * \sum_{k=1}^{c} \lambda_{ij}^k * \overline{(x^k)^2}}{\left(1 - \sum_{k=1}^{c-1} \frac{\lambda_{ij}^k}{\mu^k C_{ij}}\right) * \left(1 - \sum_{k=1}^{c} \frac{\lambda_{ij}^k}{\mu^k C_{ij}}\right)} + \tau_{ij} \right]$$

Its first and second derivatives with respect to λ_{ij}^c are

$$\frac{\partial g_{ij}^c}{\partial \lambda_{ij}^c} = \frac{\left[\frac{1}{\mu^c C_{ij}} * \left(1 - \sum_{k=1}^{c} \frac{\lambda_{ij}^k}{\mu^k C_{ij}}\right) + \frac{1}{2} * \sum_{k=1}^{c} \lambda_{ij}^k * \overline{(x^k)^2} + \frac{1}{2} * \lambda_{ij}^c * \overline{(x^c)^2}\right]}{\lambda * \left(1 - \sum_{k=1}^{c-1} \frac{\lambda_{ij}^k}{\mu^k C_{ij}}\right) * \left(1 - \sum_{k=1}^{c} \frac{\lambda_{ij}^k}{\mu^k C_{ij}}\right)}$$

$$+ \frac{\frac{\lambda_{ij}^c}{2\mu^c C_{ij}} * \sum_{k=1}^{c} \lambda_{ij}^k * \overline{(x^k)^2}}{\lambda * \left(1 - \sum_{k=1}^{c-1} \frac{\lambda_{ij}^k}{\mu^k C_{ij}}\right) * \left(1 - \sum_{k=1}^{c} \frac{\lambda_{ij}^k}{\mu^k C_{ij}}\right)^2} + \frac{\tau_{ij}}{\lambda}\Bigg]$$

$$\frac{\partial^2 g_{ij}^c}{\partial (\lambda_{ij}^c)^2} = \frac{1}{\lambda * \left(1 - \sum_{k=1}^{c-1} \frac{\lambda_{ij}^k}{\mu^k C_{ij}}\right)} * \Bigg\{ \frac{\overline{(x^c)^2}}{1 - \sum_{k=1}^{c} \frac{\lambda_{ij}^k}{\mu^k C_{ij}}} +$$

$$+ \frac{\sum_{k=1}^{c} \lambda_{ij}^k * \overline{(x^k)^2} + \lambda_{ij}^c * \overline{(x^c)^2}}{\mu^c C_{ij} * \left(1 - \sum_{k=1}^{c} \frac{\lambda_{ij}^k}{\mu^k C_{ij}}\right)^2} + \frac{\lambda_{ij}^c * \sum_{k=1}^{c} \lambda_{ij}^k * \overline{(x^k)^2}}{(\mu^c C_{ij})^2 * \left(1 - \sum_{k=1}^{c} \frac{\lambda_{ij}^k}{\mu^k C_{ij}}\right)^3} \Bigg\}$$

We can use the above defined functions to evaluate the length to destinations as well as the path lengths. Then using the Theorems of the previous sections we send jobs to minimum length destinations via minimum length paths. If the rejection length is the minimum, then we rejected them.

We further illustrate the proposed methodologies and the performance of the proposed algorithms by explicitly solving some examples.

4.5 Example 1: Two-classes Two-processors

4.5.1 Introduction

In this section, we introduce a simple queueing model for two classes of jobs that share two servers (Figure 4.1).

The problem is to assign these jobs to the two servers so as to minimize a delay objective. An application is routing, where two classes of packets may use two different links for transmission between the source and destination. Another application is load sharing for a multiprocessor system, where two classes of jobs may use two processors for execution.

Let class α jobs arrive to the system with rate λ^α (Poisson arrivals) and class β jobs arrive to the system with rate λ^β (Poisson arrivals). So, the total arrival rate is $\lambda = \lambda^\alpha + \lambda^\beta$. Jobs of both classes may be served at either of the two processors, which have service rates C_1 and C_2. So, the total system capacity is $C = C_1 + C_2$. Without loss of generality, let the service requirement of each job be exponentially distributed with mean 1. The fraction of class α jobs assigned to processor 1 is ϕ_1^α and to processor 2 is ϕ_2^α; and the fraction of class β jobs assigned to processor 1 is ϕ_1^β and to processor 2 is ϕ_2^β.

Furthermore, for stability reasons it is assumed that the total arrival rate is less than the total service rate: $\lambda^\alpha + \lambda^\beta \leq C_1 + C_2$ *or* $\lambda \leq C$.

In the following sections, we consider three formulations and solutions for sharing the two processors among jobs of the two classes.

4.5.2 Traffic Aggregation

In this section, we find the optimal load sharing policy, when the two classes are aggregated into a single class. Therefore, the fraction of class α jobs assigned to a processor is equal to the fraction of class β jobs assigned to that processor, i.e. $\phi_1^\alpha = \phi_1^\beta = \phi_1$ and $\phi_2^\alpha = \phi_2^\beta = \phi_2$. If both classes want to minimize the average job delay in the system [256], then we have the following optimization problem:

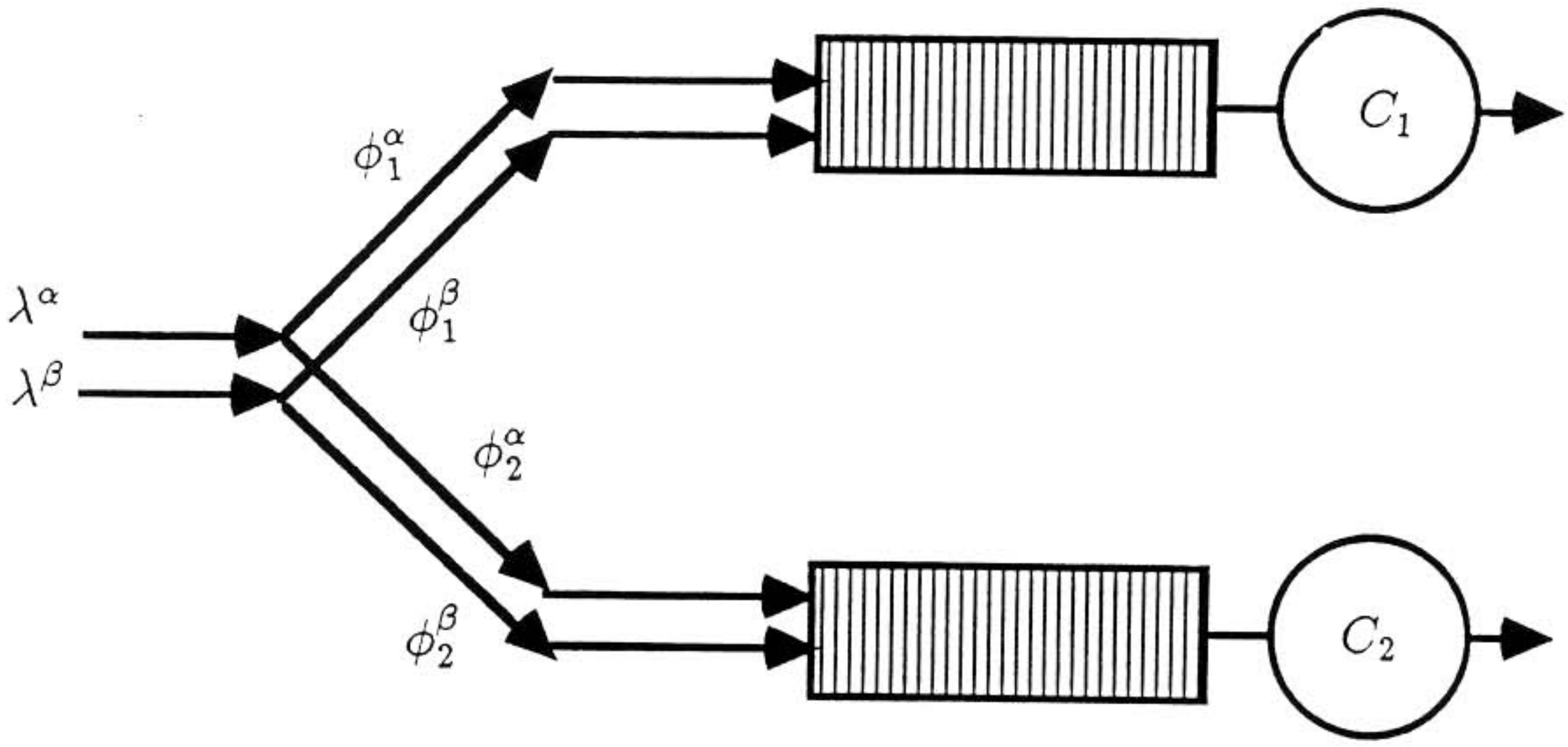


Figure 4.1: Two-class two-processor load sharing.

minimize

$$J(\phi_1, \phi_2) = \frac{\phi_1}{C_1 - (\lambda^\alpha + \lambda^\beta) * \phi_1} + \frac{\phi_2}{C_2 - (\lambda^\alpha + \lambda^\beta) * \phi_2}$$

with respect to $\phi_1, \ \phi_2$

such that $\phi_1 + \phi_2 = 1, \ \ \phi_1 \geq 0, \ \ \phi_2 \geq 0.$

The average delay objective function $J(\phi_1, \phi_2)$ is convex with respect to (ϕ_1, ϕ_2) over the convex space $\phi_1 + \phi_2 = 1$, $\phi_1, \phi_2 \geq 0$, for $C_1 - (\lambda^\alpha + \lambda^\beta) * \phi_1 > 0$ and $C_2 - (\lambda^\alpha + \lambda^\beta) * \phi_2 > 0$. This is a simple problem and can easily be solved [45, 141]. First define the auxiliary variable

$$K_1 = \frac{C_1 + C_2 - \lambda^\alpha - \lambda^\beta}{\lambda^\alpha + \lambda^\beta} * \frac{\sqrt{C_1}}{\sqrt{C_1} + \sqrt{C_2}}$$

Then, the following policy optimally assigns the arriving jobs to the two processors:

If $C_1 - \sqrt{C_1 C_2} \leq \lambda^\alpha + \lambda^\beta$ *and* $C_2 - \sqrt{C_1 C_2} \leq \lambda^\alpha + \lambda^\beta \leq C_1 + C_2$

then $\phi_1^* = \frac{C_1}{\lambda^\alpha + \lambda^\beta} - K_1$

If $0 \leq \lambda^\alpha + \lambda^\beta \leq C_1 - \sqrt{C_1 C_2}$

then $\phi_1^* = 1$

If $0 \leq \lambda^\alpha + \lambda^\beta \leq C_2 - \sqrt{C_1 C_2}$

then $\phi_1^* = 0$

Of course, the optimum load sharing fraction to the other processor is $\phi_2^* = 1 - \phi_1^*$.

In Figure 4.2 we show the optimum routing fraction to processor 1, ϕ_1^*, versus the system utilization, λ/C, for fixed processor 2 capacity, $C_2 = 1$, and different processor 1 capacities, $C_1 = 1, 2, 3, 5, 7$ and 10. When the two processors have equal capacity, $C_1 = C_2 = 1$, then the flow is split half to each processor ($\phi_1^* = \phi_2^* = 0.5$). As we increase the processor 1 capacity, then this processor tends to be exclusively used ($\phi_1^* = 1$) for a larger range of system utilization.

4.5.3 Team Optimum Solution

In this section, we find the optimum load sharing decisions, when each class is treated independently from the other. The fraction of class α jobs assigned to a processor may be different than the fraction of class β jobs assigned to that processor. However, both classes minimize the same objective - the average job delay. This problem can be considered as a cooperative team game between the two classes, where each class solves the following problem:

$$\begin{aligned} &\textit{minimize} \\ &J(\phi_1^\alpha, \phi_2^\alpha, \phi_1^\beta, \phi_2^\beta) = \frac{\lambda^\alpha * \phi_1^\alpha + \lambda^\beta * \phi_1^\beta}{\lambda^\alpha + \lambda^\beta} * \frac{1}{C_1 - \lambda^\alpha * \phi_1^\alpha - \lambda^\beta * \phi_1^\beta} + \\ &\qquad + \frac{\lambda^\alpha * \phi_2^\alpha + \lambda^\beta * \phi_2^\beta}{\lambda^\alpha + \lambda^\beta} * \frac{1}{C_2 - \lambda^\alpha * \phi_2^\alpha - \lambda^\beta * \phi_2^\beta} \\ &\textit{with respect to} \quad \phi_1^\alpha,\ \phi_2^\alpha,\ \phi_1^\beta,\ \phi_2^\beta \\ &\textit{such that} \qquad \phi_1^\alpha + \phi_2^\alpha = 1,\ \ \phi_1^\beta + \phi_2^\beta = 1,\quad \phi_1^\alpha,\ \phi_2^\alpha,\ \phi_1^\beta,\ \phi_2^\beta \geq 0. \end{aligned}$$

The objective function $J(\phi_1^\alpha, \phi_2^\alpha, \phi_1^\beta, \phi_2^\beta)$ is convex with respect to $(\phi_1^\alpha, \phi_2^\alpha, \phi_1^\beta, \phi_2^\beta)$ over the convex space $\phi_1^\alpha + \phi_2^\alpha = 1$, $\phi_1^\beta + \phi_2^\beta = 1$, $\phi_1^\alpha, \phi_2^\alpha, \phi_1^\beta, \phi_2^\beta \geq 0$, for $C_1 - \lambda^\alpha * \phi_1^\alpha - \lambda^\beta * \phi_1^\beta > 0$ and for $C_2 - \lambda^\alpha * \phi_2^\alpha - \lambda^\beta * \phi_2^\beta > 0$. Define the auxiliary variables

$$K_1^\alpha = \frac{C_1 + C_2 - \lambda^\alpha - \lambda^\beta}{\lambda^\alpha} * \frac{\sqrt{C_1}}{\sqrt{C_1} + \sqrt{C_2}}$$

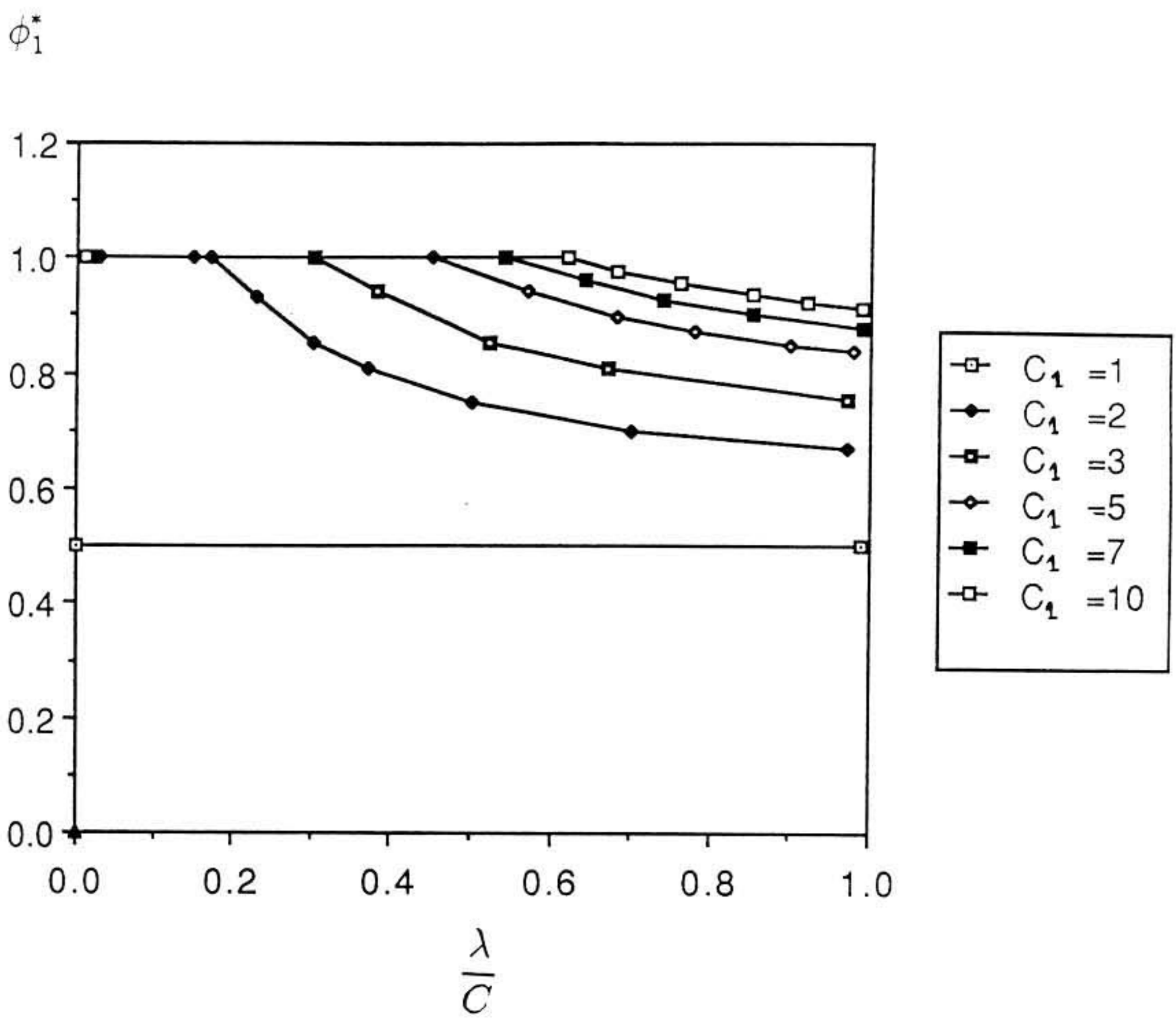


Figure 4.2: Optimum routing fraction to processor 1, ϕ_1^*, versus the system utilization, λ/C, for fixed processor 2 capacity, $C_2 = 1$, and different processor 1 capacities, $C_1 = 1$, 2, 3, 5, 7 and 10.

$$K_1^{\beta} = \frac{C_1 + C_2 - \lambda^{\alpha} - \lambda^{\beta}}{\lambda^{\beta}} * \frac{\sqrt{C_1}}{\sqrt{C_1} + \sqrt{C_2}}$$

Then, the following policy will optimally assign the arriving jobs to the two processors:

$$If \quad \lambda^{\alpha} + \lambda^{\beta} \leq C_1 + C_2,$$

$$then \qquad \phi_1^{\alpha *} = \frac{C_1 - \lambda^{\beta}\phi_1^{\beta *}}{\lambda^{\alpha}} - K_1^{\alpha}$$

$$\phi_1^{\beta *} = \frac{C_1 - \lambda^{\alpha}\phi_1^{\alpha *}}{\lambda^{\beta}} - K_1^{\beta}$$

accept the solution only if

$$C_1 - \lambda^{\beta}\phi_1^{\beta *} - \sqrt{\frac{C_1}{C_2}}(C_2 - \lambda^{\beta}\phi_2^{\beta *}) \leq \lambda^{\alpha}$$

$$C_2 - \lambda^{\beta}\phi_2^{\beta *} - \sqrt{\frac{C_2}{C_1}}(C_1 - \lambda^{\beta}\phi_1^{\beta *}) \leq \lambda^{\alpha}$$

$$C_1 - \lambda^{\alpha}\phi_1^{\alpha *} - \sqrt{\frac{C_1}{C_2}}(C_2 - \lambda^{\alpha}\phi_2^{\alpha *}) \leq \lambda^{\beta}$$

$$C_2 - \lambda^{\alpha}\phi_2^{\alpha *} - \sqrt{\frac{C_2}{C_1}}(C_1 - \lambda^{\alpha}\phi_1^{\alpha *}) \leq \lambda^{\beta}$$

$$If \quad \lambda^{\alpha} + \lambda^{\beta} \leq C_1 - \sqrt{C_1 C_2},$$

$$then \quad \phi_1^{\alpha *} = 1, \quad \phi_1^{\beta *} = 1$$

$$If \quad \lambda^{\alpha}\sqrt{C_2} - \lambda^{\beta}\sqrt{C_1} = \sqrt{C_1 C_2}(\sqrt{C_1} - \sqrt{C_2}),$$

$$then \quad \phi_1^{\alpha *} = 1, \quad \phi_1^{\beta *} = 0$$

$$If \quad \lambda^{\alpha}\sqrt{C_1} - \lambda^{\beta}\sqrt{C_2} = \sqrt{C_1 C_2}(\sqrt{C_2} - \sqrt{C_1}),$$

$$then \quad \phi_1^{\alpha *} = 0, \quad \phi_1^{\beta *} = 1$$

$$If \quad \lambda^{\alpha} + \lambda^{\beta} \leq C_2 - \sqrt{C_1 C_2},$$

$$then \quad \phi_1^{\alpha *} = 0, \quad \phi_1^{\beta *} = 0$$

$$If \quad \lambda^{\alpha} + \lambda^{\beta} \geq C_1 - \sqrt{C_1 C_2} \quad and \quad \lambda^{\alpha}\sqrt{C_2} - \lambda^{\beta}\sqrt{C_1} \leq \sqrt{C_1 C_2}(\sqrt{C_1} - \sqrt{C_2}),$$

$$then \quad \phi_1^{\alpha *} = 1$$

$$\phi_1^{\beta *} = \frac{C_1 - \lambda^{\alpha}}{\lambda^{\beta}} - K_1^{\beta}$$

$$If \quad \lambda^{\alpha} + \lambda^{\beta} \geq C_2 - \sqrt{C_1 C_2} \quad and \quad \lambda^{\alpha}\sqrt{C_1} - \lambda^{\beta}\sqrt{C_2} \leq \sqrt{C_1 C_2}(\sqrt{C_2} - \sqrt{C_1}),$$

$$then \quad \phi_1^{\alpha *} = 0$$

$$\phi_1^{\beta *} = \frac{C_1}{\lambda^{\beta}} - K_1^{\beta}$$

$$If \quad \lambda^{\alpha} + \lambda^{\beta} \geq C_1 - \sqrt{C_1 C_2} \quad and \quad \lambda^{\beta}\sqrt{C_2} - \lambda^{\alpha}\sqrt{C_1} \leq \sqrt{C_1 C_2}(\sqrt{C_1} - \sqrt{C_2}),$$

$$then \quad \phi_1^{\beta *} = 1$$

$$\phi_1^{\alpha *} = \frac{C_1 - \lambda^{\beta}}{\lambda^{\alpha}} - K_1^{\alpha}$$

$$If \quad \lambda^{\alpha} + \lambda^{\beta} \geq C_2 - \sqrt{C_1 C_2} \quad and \quad \lambda^{\beta}\sqrt{C_1} - \lambda^{\alpha}\sqrt{C_2} \leq \sqrt{C_1 C_2}(\sqrt{C_2} - \sqrt{C_1}),$$

$$then \quad \phi_1^{\beta *} = 0$$

$$\phi_1^{\alpha *} = \frac{C_1}{\lambda^{\alpha}} - K_1^{\alpha}$$

Of course, the optimum load sharing fractions to the other processor are $\phi_2^{\alpha *} = 1 - \phi_1^{\alpha *}$ and $\phi_2^{\beta *} = 1 - \phi_1^{\beta *}$.

In order to find the optimum load sharing fractions $(\phi_1^{\alpha *}, \phi_1^{\beta *})$ for the first case of the team load sharing policy, we give all possible values to $\phi_1^{\beta} \in [0, 1]$ and calculate the corresponding values for the ϕ_1^{α}

$$\phi_1^{\alpha} = \frac{C_1 - \lambda^{\beta}\phi_1^{\beta}}{\lambda^{\alpha}} - K_1^{\alpha}$$

Then we check if the conditions for the resulting load sharing fractions $\phi_1^{\alpha}, \phi_2^{\alpha}$, $\phi_1^{\beta}, \phi_2^{\beta}$ are satisfied and accept them as optimum load sharing fractions, if so.

In Fig 4.3, we show the optimum load sharing fractions $(\phi_1^{\alpha *}, \phi_1^{\beta *})$ for a simple homogeneous case $C_1 = C_2 = 1$ and $\lambda^{\alpha} = \lambda^{\beta} = 0.1, ..., 0.9$. We note that the solution pairs form a straight line which is stated in the Proposition 1. Thus we have a multiplicity of optimum load sharing fractions and we can choose any pair of them with some other criterion. For example if we want $\phi_1^{\alpha} = \phi_1^{\beta}$, then the solution set reduces to a single point that is also the solution of the previous section, where we treat the two classes as one.

Proposition 1:

The set of the optimum load sharing fractions $(\phi_1^{\alpha *}, \phi_1^{\beta *})$ *for fixed arrival rate* λ^{α} *and* λ^{β} *and fixed processor capacities* C_1 *and* C_2 *forms a straight line.*

Proof: The general equation that gives the optimum load sharing fractions is

$$\phi_1^{\alpha *} = \frac{C_1 - \lambda^{\beta}\phi_1^{\beta *}}{\lambda^{\alpha}} - K_1^{\alpha}$$

Obviously, this equation describes a straight line. □

In Figure 4.4, we show the optimum load sharing fractions $(\phi_1^{\alpha *}, \phi_1^{\beta *})$ for fixed processor capacities, $C_1 = 2, \quad C_2 = 1$, fixed class β arrival rate, $\lambda^{\beta} = 1$, and

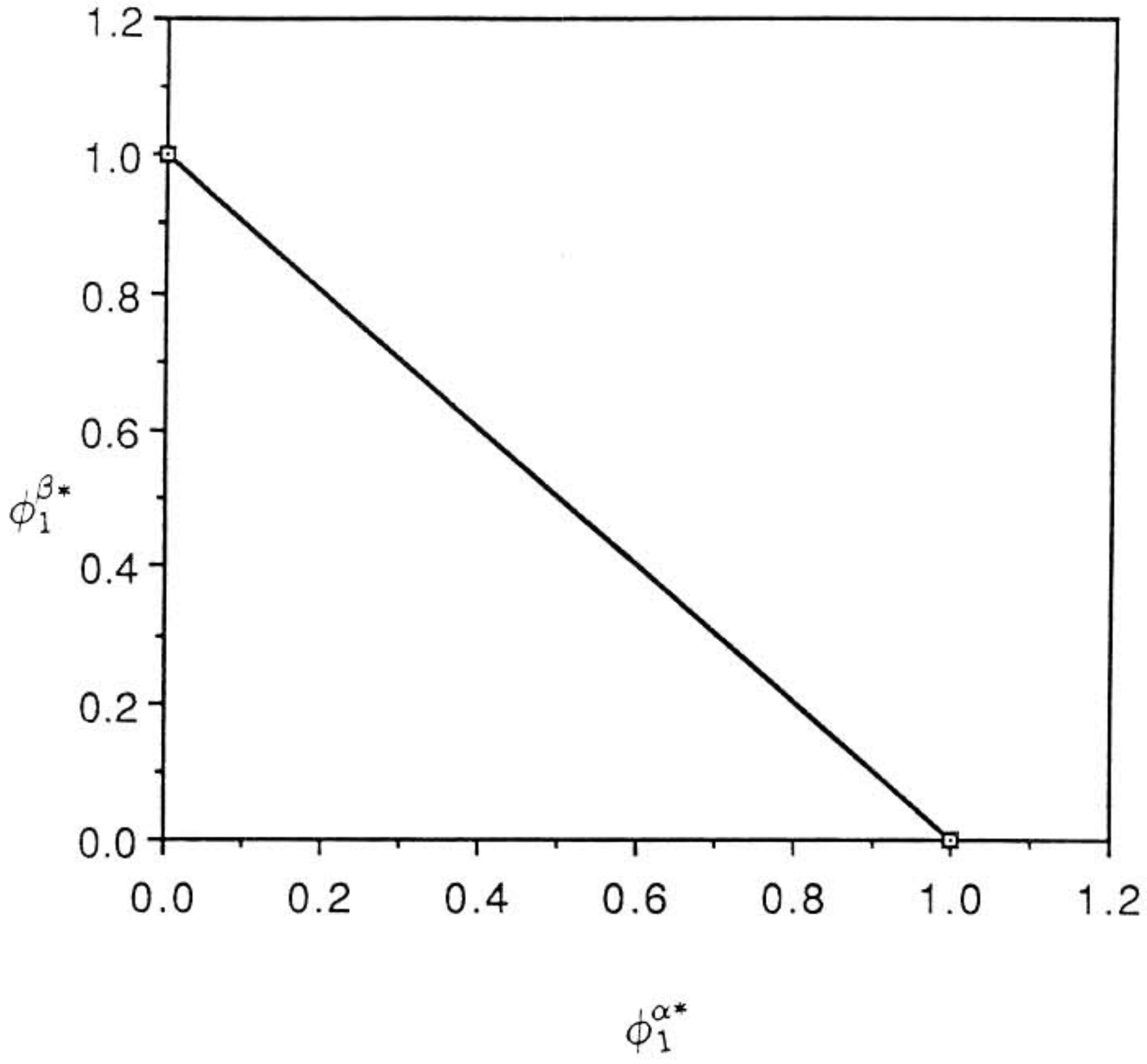


Figure 4.3: Team optimum load sharing fractions $(\phi_1^{\alpha*}, \phi_1^{\beta*})$ for a simple homogeneous case $C_1 = C_2 = 1$ and $\lambda^\alpha = \lambda^\beta = 0.1, ..., 0.9$.

different class α arrival rates, $\lambda^{\alpha} = 0.1, ... 1.9$. We notice something remarkable. The straight line solutions for different class α arrival rates intersect at a single intersection point. This means that there is a common pair of optimum load sharing fractions $(\phi_1^{\alpha*}, \phi_1^{\beta*})$, where we can optimally operate for different class α arrival rates. So, we can use the optimum load sharing fractions of the intersection point and operate optimally even if the class α arrival rate varies. Proposition 2 describes this result more formally.

Proposition 2:

For a given system $C_1 > C_2$, with fixed class β arrival rate λ^{β},

If

$$0 \leq C_1 - (C_1 + C_2 - \lambda^{\beta}) * \frac{\sqrt{C_1}}{\sqrt{C_1} + \sqrt{C_2}} \leq \lambda^{\beta}$$

then the straight lines of the team optimum fractions $(\phi_1^{\alpha*}, \phi_1^{\beta*})$, for different class α arrival rates λ^{α} ($\lambda^{\alpha} + \lambda^{\beta} \leq C_1 + C_2$), intersect at a single point

$$\phi_1^{\alpha*} = \frac{\sqrt{C_1}}{\sqrt{C_1} + \sqrt{C_2}}$$

$$\phi_1^{\beta*} = \frac{C_1}{\lambda^{\beta}} - \frac{C_1 + C_2 - \lambda^{\beta}}{\lambda^{\beta}} * \frac{\sqrt{C_1}}{\sqrt{C_1} + \sqrt{C_2}}$$

i.e. this intersection point is independent of the class α arrival rate λ^{α}.

Proof: Let the straight line of the optimum load sharing fractions for a given class α arrival rate λ_1^{α} be

$$\phi_1^{\alpha*} = \frac{C_1 - \lambda^{\beta}\phi_1^{\beta*}}{\lambda_1^{\alpha}} - \frac{C_1 + C_2 - \lambda_1^{\alpha} - \lambda^{\beta}}{\lambda_1^{\alpha}} * \frac{\sqrt{C_1}}{\sqrt{C_1} + \sqrt{C_2}}$$

Let also the straight line of the optimum load sharing fractions for another given class α arrival rate λ_2^{α} be

$$\phi_1^{\alpha**} = \frac{C_1 - \lambda^{\beta}\phi_1^{\beta**}}{\lambda_2^{\alpha}} - \frac{C_1 + C_2 - \lambda_2^{\alpha} - \lambda^{\beta}}{\lambda_2^{\alpha}} * \frac{\sqrt{C_1}}{\sqrt{C_1} + \sqrt{C_2}}$$

Since the slope of each line depends on the class α arrival rate, these lines will intersect each other. Now in order to prove that all lines intersect at a single

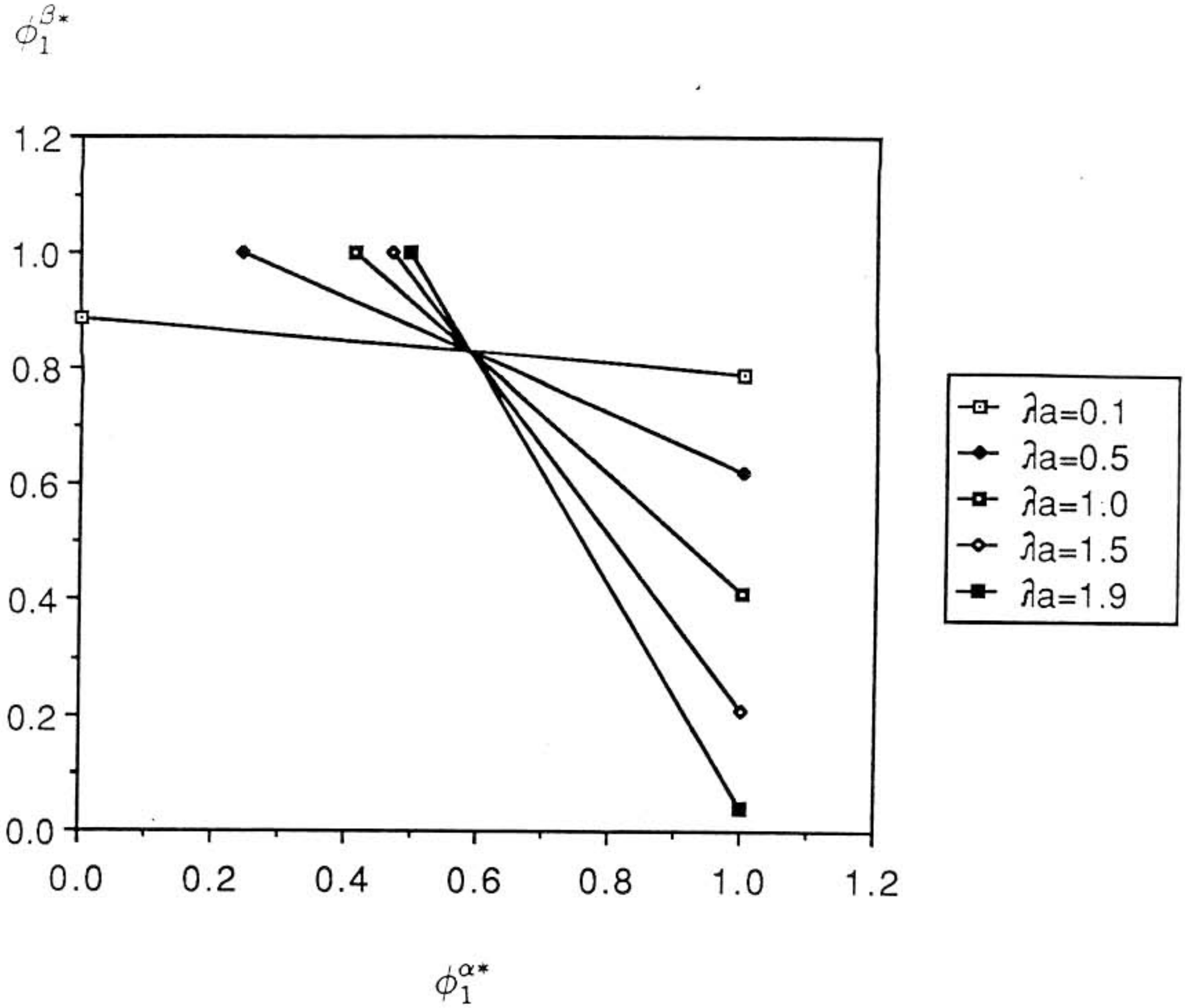


Figure 4.4: Team optimum load sharing fractions $(\phi_1^{\alpha*}, \phi_1^{\beta*})$ for fixed processor capacities, $C_1 = 2$, $C_2 = 1$, fixed class β arrival rate, $\lambda^\beta = 1$, and different class α arrival rates, $\lambda^\alpha = 0.1, ..., 1.9$.

point, it is enough to prove that the intersection point is independent of the class α arrival rate λ^α.

Let $(\phi_1^{\alpha\#}, \phi_1^{\beta\#})$ be the intersection point of these two lines, i.e. $\phi_1^{\alpha*} = \phi_1^{\alpha**} = \phi_1^{\alpha\#}$ and $\phi_1^{\beta*} = \phi_1^{\beta**} = \phi_1^{\beta\#}$. Then

$$\frac{C_1 - \lambda^\beta \phi_1^{\beta\#}}{\lambda_1^\alpha} - \frac{C_1 + C_2 - \lambda_1^\alpha - \lambda^\beta}{\lambda_1^\alpha} * \frac{\sqrt{C_1}}{\sqrt{C_1} + \sqrt{C_2}} =$$

$$= \frac{C_1 - \lambda^\beta \phi_1^{\beta\#}}{\lambda_2^\alpha} - \frac{C_1 + C_2 - \lambda_2^\alpha - \lambda^\beta}{\lambda_2^\alpha} * \frac{\sqrt{C_1}}{\sqrt{C_1} + \sqrt{C_2}}$$

and finally

$$\phi_1^{\beta\#} = \frac{C_1}{\lambda^\beta} - \frac{C_1 + C_2 - \lambda^\beta}{\lambda^\beta} * \frac{\sqrt{C_1}}{\sqrt{C_1} + \sqrt{C_2}}$$

$$\phi_1^{\alpha\#} = \frac{\sqrt{C_1}}{\sqrt{C_1} + \sqrt{C_2}}$$

Thus the intersection point $(\phi_1^{\alpha\#}, \phi_1^{\beta\#})$ is independent from the class α arrival rate λ^α. In order for the intersection point to be also a solution, it must be in the range $0 \leq \phi_1^{\beta\#} \leq 1$. Thus the result. □

In Figure 4.5, we show the optimum load sharing fractions $(\phi_1^{\alpha*}, \phi_1^{\beta*})$ for fixed arrival rates $\lambda^\alpha = 2, \quad \lambda^\beta = 1$, fixed processor 2 capacity $C_2 = 1$ and different processor 1 capacities $C_1 = 2.1, ..., 3.8$. We see that the solution lines are parallel.

Proposition 3:

The straight lines of the optimum load sharing fractions for fixed arrival rates $\lambda^\alpha, \lambda^\beta$, fixed processor 2 capacity C_2 and different processor 1 capacities C_1 are parallel.

Proof: The optimum load sharing fractions are described by the following straight line

$$\phi_1^{\alpha*} = \frac{C_1 - \lambda^\beta \phi_1^*}{\lambda^\alpha} - \frac{C_1 + C_2 - \lambda^\alpha - \lambda^\beta}{\lambda^\alpha} * \frac{\sqrt{C_1}}{\sqrt{C_1} + \sqrt{C_2}}$$

that has slope independent of the capacity of the processors. □

As we have seen we have a set of optimum load sharing fraction pairs $(\phi_1^{\alpha*}, \phi_1^{\beta*})$ that all achieve the same global minimum delay. However, these optimum load

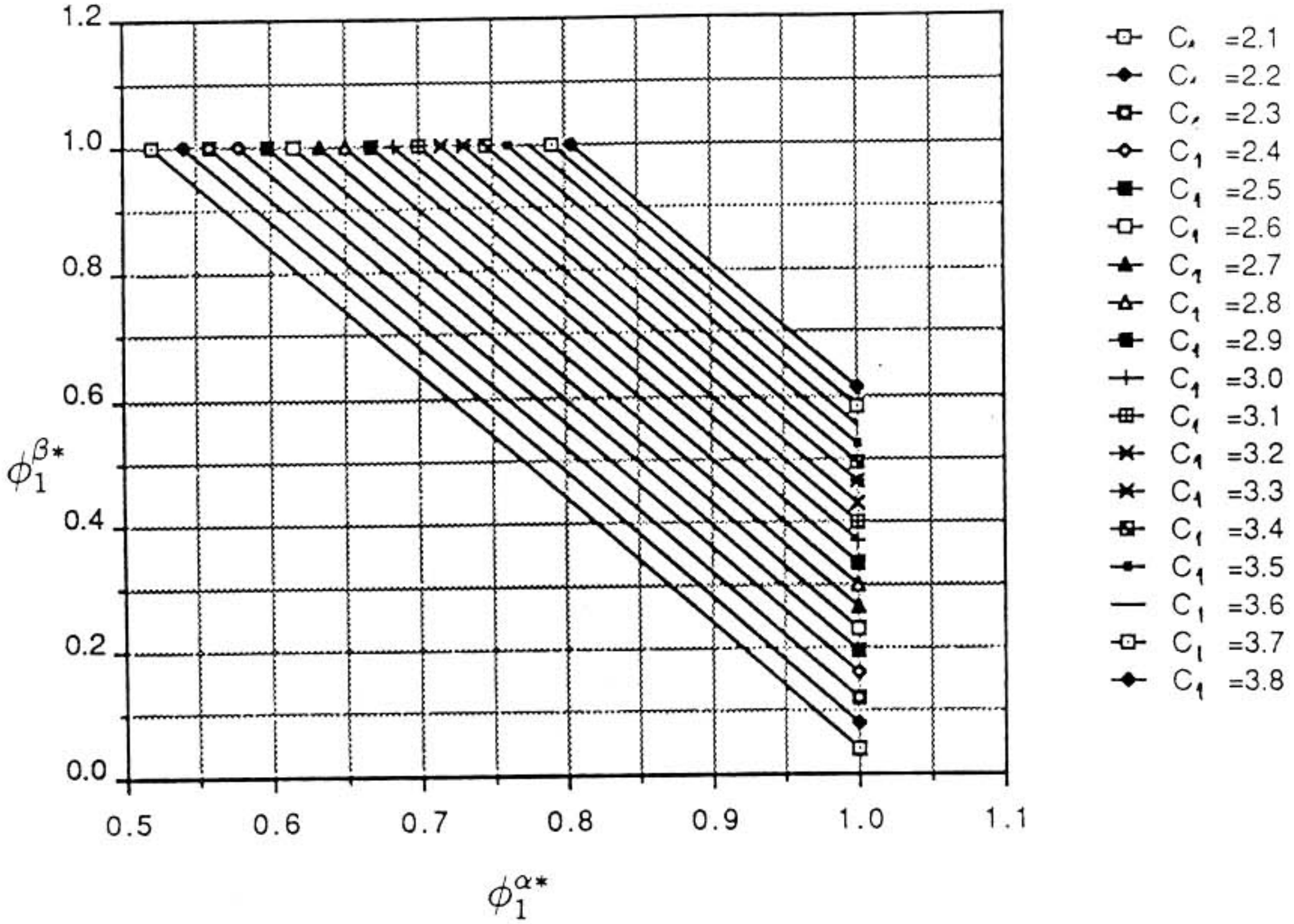


Figure 4.5: Team optimum load sharing fractions $(\phi_1^{\alpha*}, \phi_1^{\beta*})$ for fixed arrival rates $\lambda^{\alpha} = 2, \quad \lambda^{\beta} = 1$, fixed processor 2 capacity $C_2 = 1$ and different processor 1 capacities $C_1 = 2.1, ..., 3.8$.

sharing fractions will give different average delays for each class. So, we can choose the operating point using another delay objective. In Figure 4.6, we show the difference in the average delay of class α and class β jobs, $J^{\alpha*} - J^{\beta*}$, versus the class α optimum load sharing fraction, $\phi_1^{\alpha*}$, for fixed processor capacities, $C_1 = 2$, $C_2 = 1$, fixed class β arrival rate, $\lambda^\beta = 1$, and different class α arrival rates, $\lambda^\alpha = 0.1, ... 1.9$. An example is when it is desired that both classes have the same average delay. Then this point will be the intersection of the delay difference line and the zero delay difference line. The operating point for this case is the same as the case where we aggregate the two classes into a single class and therefore we treat them similarly. Another example is when there is a secondary objective that class α should receive better treatment than class β. Then the lowest point of the delay difference line $J^{\alpha*} - J^{\beta*}$ is chosen.

4.5.4 Nash Equilibrium Solution

In this section, we find the optimum load sharing decisions, when each class chooses the best strategy for its jobs given the decision of the other class. Class α assigns its jobs to the two processors such that the average delay of its jobs is minimized. Similarly, class β assigns its jobs to the two processors such that the average delay of its jobs is minimized. Therefore jobs of different classes do not have the same objective and they compete for sharing the two processors. We formulate and solve the above multi-objective optimization problem as a non cooperative Nash game between the two classes. After reaching a Nash equilibrium, no class of jobs will have a rational motive to unilaterally deviate from its equilibrium strategy.

Class α solves the following problem:

minimize

$$J^\alpha(\phi_1^\alpha, \phi_2^\alpha, \phi_1^{\beta*}, \phi_2^{\beta*}) = \frac{\phi_1^\alpha}{C_1 - \lambda^\alpha * \phi_1^\alpha - \lambda^\beta * \phi_1^{\beta*}} + \frac{\phi_2^\alpha}{C_2 - \lambda^\alpha * \phi_2^\alpha - \lambda^\beta * \phi_2^{\beta*}}$$

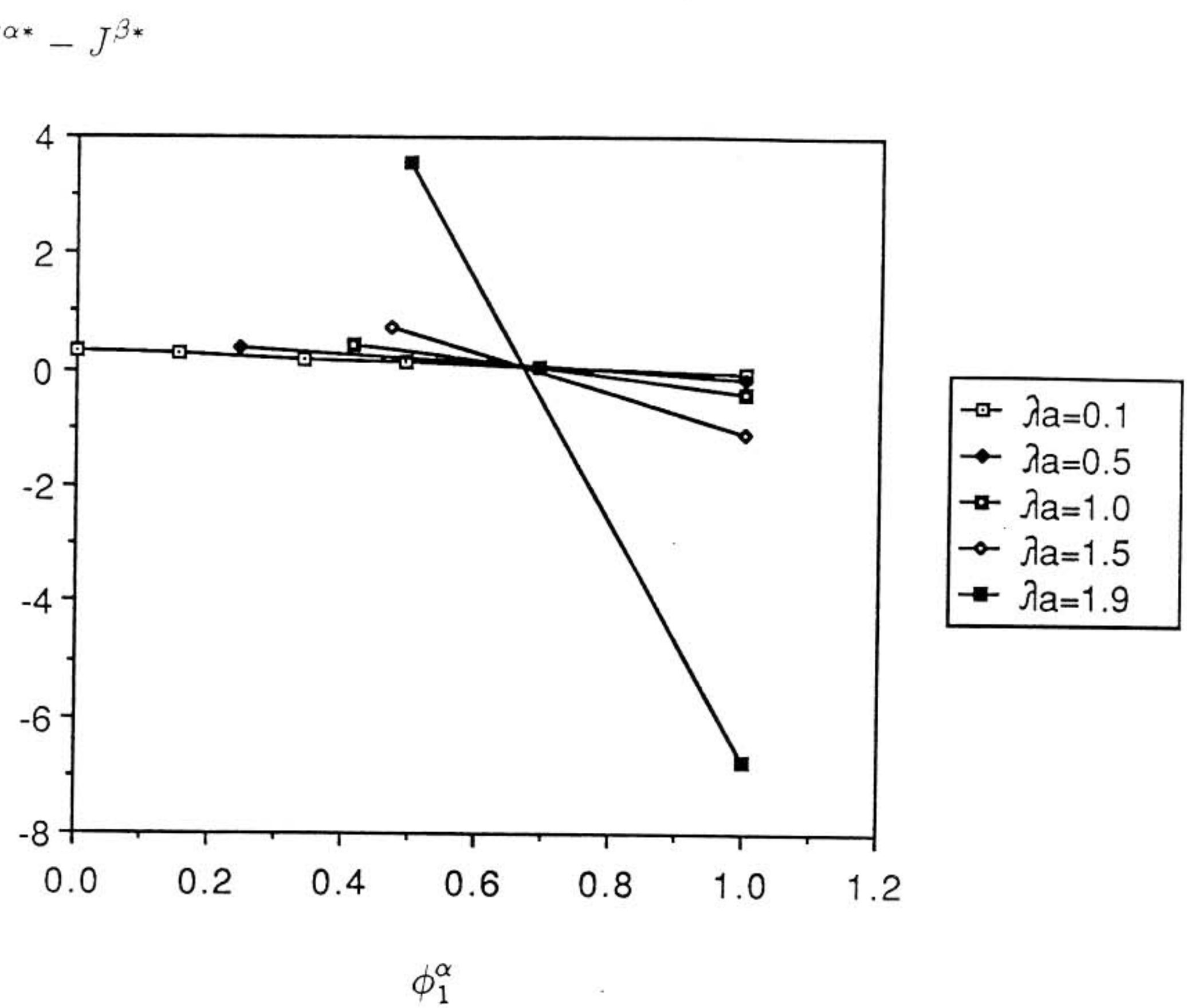


Figure 4.6: For Team optimal solution, the difference in the average delay of class α and class β jobs, $J^{\alpha*} - J^{\beta*}$, versus the class α optimum load sharing fraction, $\phi_1^{\alpha*}$, for fixed processor capacities, $C_1 = 2, \quad C_2 = 1$, fixed class β arrival rate, $\lambda^\beta = 1$, and different class α arrival rates, $\lambda^\alpha = 0.1, ..., 1.9$.

with respect to $\phi_1^\alpha,\ \phi_2^\alpha$

such that $\phi_1^\alpha + \phi_2^\alpha = 1,\quad \phi_1^\alpha,\ \phi_2^\alpha \geq 0$

The objective function $J^\alpha(\phi_1^\alpha, \phi_2^\alpha, \phi_1^{\beta*}, \phi_2^{\beta*})$ is convex with respect to $(\phi_1^\alpha, \phi_2^\alpha)$ over the convex space $\phi_1^\alpha + \phi_2^\alpha = 1, \phi_1^\alpha, \phi_2^\alpha \geq 0$, for $C_1 - \lambda^\alpha * \phi_1^\alpha - \lambda^\beta * \phi_1^{\beta*} > 0$ and $C_2 - \lambda^\alpha * \phi_2^\alpha - \lambda^\beta * \phi_2^{\beta*} > 0$.

Class β solves a similar problem:

minimize

$$J^\beta(\phi_1^{\alpha*}, \phi_2^{\alpha*}, \phi_1^\beta, \phi_2^\beta) = \frac{\phi_1^\beta}{C_1 - \lambda^\alpha * \phi_1^{\alpha*} - \lambda^\beta * \phi_1^\beta} + \frac{\phi_2^\beta}{C_2 - \lambda^\alpha * \phi_2^{\alpha*} - \lambda^\beta * \phi_2^\beta}$$

with respect to $\phi_1^\beta,\ \phi_2^\beta$

such that $\phi_1^\beta + \phi_2^\beta = 1,\quad \phi_1^\beta,\ \phi_2^\beta \geq 0.$

The objective function $J^\beta(\phi_1^{\alpha*}, \phi_2^{\alpha*}, \phi_1^\beta, \phi_2^\beta)$ is convex $(\phi_1^\beta, \phi_2^\beta)$ over the convex space $\phi_1^\beta + \phi_2^\beta = 1, \phi_1^\beta, \phi_2^\beta \geq 0$, for $C_1 - \lambda^\alpha * \phi_1^{\alpha*} - \lambda^\beta * \phi_1^\beta > 0$ and $C_2 - \lambda^\alpha * \phi_2^{\alpha*} - \lambda^\beta * \phi_2^\beta > 0$.

When the players are in a Nash equilibrium, no player can improve his cost by altering his decision unilaterally. Next, we give the definition of a Nash equilibrium in our context:

Definition :

A vector $[\phi_1^{\alpha*}, \phi_2^{\alpha*}, \phi_1^{\beta*}, \phi_2^{\beta*}]$ *with* $\phi_1^{\alpha*} + \phi_2^{\alpha*} = 1,\ \ \phi_1^{\beta*} + \phi_2^{\beta*} = 1$, *and* $\phi_1^{\alpha*},\ \phi_2^{\alpha*},\ \phi_1^{\beta*},\ \phi_2^{\beta*} \geq 0$ *is called a* Nash equilibrium *for the two-class two-processor load sharing problem if*

$$J^{\alpha}(\phi_1^{\alpha*}, \phi_2^{\alpha*}, \phi_1^{\beta*}, \phi_2^{\beta*}) \leq \inf_{\substack{\phi_1^{\alpha} + \phi_2^{\alpha} = 1, \\ \phi_1^{\alpha}, \phi_2^{\alpha} \geq 0}} J^{\alpha}(\phi_1^{\alpha}, \phi_2^{\alpha}, \phi_1^{\beta*}, \phi_2^{\beta*})$$

$$J^{\beta}(\phi_1^{\alpha*}, \phi_2^{\alpha*}, \phi_1^{\beta*}, \phi_2^{\beta*}) \leq \inf_{\substack{\phi_1^{\beta} + \phi_2^{\beta} = 1, \\ \phi_1^{\beta}, \phi_2^{\beta} \geq 0}} J^{\beta}(\phi_1^{\alpha*}, \phi_2^{\alpha*}, \phi_1^{\beta}, \phi_2^{\beta})$$

Therefore each class will minimize its average job delay given that the other class has minimized the average delay of its jobs.

Theorem 1: *uniqueness*

Let two classes of jobs α and β compete for two processors. Jobs from each class arrive according to Poisson distribution and require service according to exponential distribution. Each class tries to minimize the average delay of its own jobs. There exists a unique Nash equilibrium for the above problem.

Proof: The action spaces $\phi_1^{\alpha}+\phi_2^{\alpha} = 1,\ \phi_1^{\alpha},\ \phi_2^{\alpha} \geq 0$ and $\phi_1^{\beta}+\phi_2^{\beta} = 1,\ \phi_1^{\beta},\ \phi_2^{\beta} \geq 0$ define a convex, closed and compact set. The cost function J^{α} is jointly continuous in all its arguments and convex in $(\phi_1^{\alpha}, \phi_2^{\alpha})$ for each fixed value of $(\phi_1^{\beta}, \phi_2^{\beta})$. The cost function J^{β} is jointly continuous in all its arguments and convex in $(\phi_1^{\beta}, \phi_2^{\beta})$ for each fixed value of $(\phi_1^{\alpha}, \phi_2^{\alpha})$. The function $J^{\alpha} + J^{\beta}$ is continuous and convex in $(\phi_1^{\alpha}, \phi_2^{\alpha})$ for each fixed value of $(\phi_1^{\beta}, \phi_2^{\beta})$ as well as is continuous and convex in $(\phi_1^{\beta}, \phi_2^{\beta})$ for each fixed value of $(\phi_1^{\alpha}, \phi_2^{\alpha})$. Therefore the above routing game admits a Nash equilibrium.

The Jacobian matrix with elements $\dfrac{\partial^2 J^c}{\partial \phi_j^c \partial \phi_i^k}$, $c, k = \alpha, \beta,\ \ i, j = 1, 2$ is strictly diagonally dominant for all $(\phi_1^{\alpha}, \phi_2^{\alpha}, \phi_1^{\beta}, \phi_2^{\beta})$ such that $\phi_1^{\alpha} + \phi_2^{\alpha} = 1,\ \phi_1^{\beta} + \phi_2^{\beta} = 1,$ $\phi_1^{\alpha},\ \phi_2^{\alpha},\ \phi_1^{\beta},\ \phi_2^{\beta} \geq 0,\ C_1 - \lambda^{\alpha} * \phi_1^{\alpha} - \lambda^{\beta} * \phi_1^{\beta} > 0$ and $C_2 - \lambda^{\alpha} * \phi_2^{\alpha} - \lambda^{\beta} * \phi_2^{\beta} > 0$. □

Next, we find this unique Nash equilibrium for the above load sharing game.

Define the auxiliary variables

$$N_1^{\alpha*}(\phi_1^{\beta*}) = \frac{C_1 + C_2 - \lambda^{\alpha} - \lambda^{\beta}}{\lambda^{\alpha}} * \frac{\sqrt{C_1 - \lambda^{\beta} * \phi_1^{\beta*}}}{\sqrt{C_1 - \lambda^{\beta} * \phi_1^{\beta*}} + \sqrt{C_2 - \lambda^{\beta} * \phi_2^{\beta*}}}$$

$$N_1^{\beta*}(\phi_1^{\alpha*}) = \frac{C_1 + C_2 - \lambda^\alpha - \lambda^\beta}{\lambda^\beta} * \frac{\sqrt{C_1 - \lambda^\alpha * \phi_1^{\alpha*}}}{\sqrt{C_1 - \lambda^\alpha * \phi_1^{\alpha*}} + \sqrt{C_2 - \lambda^\alpha * \phi_2^{\alpha*}}}$$

Then, the following policy will route the arriving jobs to the two processors such that a Nash equilibrium is achieved:

$If \quad \lambda^\alpha + \lambda^\beta \leq C_1 + C_2,$

$$then \quad \phi_1^{\alpha*} = \frac{C_1 - \lambda^\beta \phi_1^{\beta*}}{\lambda^\alpha} - N_1^\alpha(\phi_1^{\beta*})$$

$$\phi_1^{\beta*} = \frac{C_1 - \lambda^\alpha \phi_1^{\alpha*}}{\lambda^\beta} - N_1^\beta(\phi_1^{\alpha*})$$

accept the solution only if

$$C_1 - \lambda^\beta \phi_1^{\beta*} - \sqrt{(C_1 - \lambda^\beta \phi_1^{\beta*})(C_2 - \lambda^\beta \phi_2^{\beta*})} \leq \lambda^\alpha$$

$$C_2 - \lambda^\beta \phi_2^{\beta*} - \sqrt{(C_1 - \lambda^\beta \phi_1^{\beta*})(C_2 - \lambda^\beta \phi_2^{\beta*})} \leq \lambda^\alpha$$

$$C_1 - \lambda^\alpha \phi_1^{\alpha*} - \sqrt{(C_1 - \lambda^\alpha \phi_1^{\alpha*})(C_2 - \lambda^\alpha \phi_2^{\alpha*})} \leq \lambda^\beta$$

$$C_2 - \lambda^\alpha \phi_2^{\alpha*} - \sqrt{(C_1 - \lambda^\alpha \phi_1^{\alpha*})(C_2 - \lambda^\alpha \phi_2^{\alpha*})} \leq \lambda^\beta$$

$If \quad \lambda^\alpha + \lambda^\beta \leq C_1 - \sqrt{(C_1 - \lambda^\alpha)C_2} \quad and \quad \lambda^\alpha + \lambda^\beta \leq C_1 - \sqrt{(C_1 - \lambda^\beta)C_2},$

$$then \quad \phi_1^{\alpha*} = 1, \quad \phi_1^{\beta*} = 1$$

$If \quad 0 \leq \lambda^\alpha \leq C_1 - \sqrt{(C_1(C_2 - \lambda^\beta)} \quad and \quad 0 \leq \lambda^\beta \leq C_2 - \sqrt{(C_1 - \lambda^\alpha)C_2},$

$$then \quad \phi_1^{\alpha*} = 1, \quad \phi_1^{\beta*} = 0$$

$$If \quad 0 \le \lambda^{\alpha} \le C_2 - \sqrt{(C_1 - \lambda^{\beta})C_2} \quad and \quad 0 \le \lambda^{\beta} \le C_1 - \sqrt{C_1(C_2 - \lambda^{\alpha})},$$

$$then \quad \phi_1^{\alpha *} = 0, \quad \phi_1^{\beta *} = 1$$

$$If \quad \lambda^{\alpha} + \lambda^{\beta} \le C_2 - \sqrt{C_1(C_2 - \lambda^{\alpha})} \quad and \quad \lambda^{\alpha} + \lambda^{\beta} \le C_2 - \sqrt{C_1(C_2 - \lambda^{\beta})},$$

$$then \quad \phi_1^{\alpha *} = 0, \quad \phi_1^{\beta *} = 0$$

$$If \quad \lambda^{\alpha} + \lambda^{\beta} \ge C_1 - \sqrt{(C_1 - \lambda^{\alpha})C_2} \quad and \quad \lambda^{\beta} \ge C_2 - \sqrt{(C_1 - \lambda^{\alpha})C_2},$$

$$then \quad \phi_1^{\alpha *} = 1$$

$$\phi_1^{\beta *} = \frac{C_1 - \lambda^{\alpha}}{\lambda^{\beta}} - N_1^{\beta}(1)$$

accept the solution only if

$$\lambda^{\alpha} \le C_1 - \lambda^{\beta}\phi_1^{\beta *} - \sqrt{(C_1 - \lambda^{\beta}\phi_1^{\beta *})(C_2 - \lambda^{\beta}\phi_2^{\beta *})}$$

$$If \quad \lambda^{\alpha} + \lambda^{\beta} \ge C_2 - \sqrt{C_1(C_2 - \lambda^{\alpha})} \quad and \quad \lambda^{\beta} \ge C_1 - \sqrt{C_1(C_2 - \lambda^{\alpha})},$$

$$then \quad \phi_1^{\alpha *} = 0$$

$$\phi_1^{\beta *} = \frac{C_1}{\lambda^{\beta}} - N_1^{\beta}(0)$$

accept the solution only if

$$\lambda^{\alpha} \le C_2 - \lambda^{\beta}\phi_2^{\beta *} - \sqrt{(C_1 - \lambda^{\beta}\phi_1^{\beta *})(C_2 - \lambda^{\beta}\phi_2^{\beta *})}$$

$$If \quad \lambda^{\alpha} + \lambda^{\beta} \geq C_1 - \sqrt{(C_1 - \lambda^{\beta})C_2} \quad and \quad \lambda^{\alpha} \geq C_2 - \sqrt{(C_1 - \lambda^{\beta})C_2},$$

$$then \quad \phi_1^{\beta *} = 1$$

$$\phi_1^{\alpha *} = \frac{C_1 - \lambda^{\beta}}{\lambda^{\alpha}} - N_1^{\alpha}(1)$$

accept the solution only if

$$\lambda^{\beta} \leq C_1 - \lambda^{\alpha}\phi_1^{\alpha *} - \sqrt{(C_1 - \lambda^{\alpha}\phi_1^{\alpha *})(C_2 - \lambda^{\alpha}\phi_2^{\alpha *})}$$

$$If \quad \lambda^{\alpha} + \lambda^{\beta} \geq C_2 - \sqrt{C_1(C_2 - \lambda^{\beta})} \quad and \quad \lambda^{\alpha} \geq C_1 - \sqrt{C_1(C_2 - \lambda^{\beta})},$$

$$then \quad \phi_1^{\beta *} = 0$$

$$\phi_1^{\alpha *} = \frac{C_1}{\lambda^{\alpha}} - N_1^{\alpha}(0)$$

accept the solution only if

$$\lambda^{\beta} \leq C_2 - \lambda^{\alpha}\phi_2^{\alpha *} - \sqrt{(C_1 - \lambda^{\alpha}\phi_1^{\alpha *})(C_2 - \lambda^{\alpha}\phi_2^{\alpha *})}$$

Of course, the Nash equilibrium load sharing fractions to the other processor are $\phi_2^{\alpha *} = 1 - \phi_1^{\alpha *}$ and $\phi_2^{\beta *} = 1 - \phi_1^{\beta *}$.

In order to find the Nash equilibrium load sharing fractions $(\phi_1^{\alpha *}, \phi_1^{\beta *})$ for the first case of the Nash load sharing, we need an iterative algorithm to calculate them. So, starting with $\phi_1^{\alpha *}(0) = \phi_1^{\beta *}(0) = 0$, we iterate according to the following algorithm:

$$\phi_1^{\alpha}(k+1) = \frac{C_1 - \lambda^{\beta}\phi_1^{\beta}(k)}{\lambda^{\alpha}} - N_1^{\alpha}(\phi_1^{\beta}(k))$$

$$\phi_1^{\beta}(k+1) = \frac{C_1 - \lambda^{\alpha}\phi_1^{\alpha}(k)}{\lambda^{\beta}} - N_1^{\beta}(\phi_1^{\alpha}(k))$$

In Figure 4.7, we show the average delay difference between the two classes $J^{\alpha*} - J^{\beta*}$ for fixed processor capacities $C_1 = 2, C_2 = 1$, fixed class β arrival rate $\lambda^\beta = 1$ and different class α arrival rates λ^α. When the class α arrival rate is equal to the class β arrival rate $\lambda^\alpha = \lambda^\beta = 1$, then both classes have the same average delay. When a class has larger arrival rate then it also has larger average delay. For a very small class α arrival rate λ^α, we notice something peculiar: the average delay difference curve is not monotonic with the arrival rate. This happens because for these values we hit the boundary ($\phi_1^{\alpha*} = 1$), as we see in Figure 4.8.

In Figure 4.8, we show the Nash equilibrium load sharing fractions of the two classes $\phi_1^{\alpha*}$ and $\phi_1^{\beta*}$, for fixed processor capacities, $C_1 = 2, C_2 = 1$, fixed class β arrival rate, $\lambda^\beta = 1$ and different class α arrival rates, λ^α. We see that for very small class α arrival rate λ^α, class α uses exclusively the faster processor 1 ($\phi_1^{\alpha*} = 1$). For equal arrival rates $\lambda^\alpha = \lambda^\beta = 1$, the Nash equilibrium load sharing fractions intersect at the point $\phi_1^{\alpha*} = \phi_1^{\beta*}$. As we increase the arrival rate they depart each other to meet again when the arrival rate becomes large.

For comparison, we also show in Figure 4.9 the team optimum load sharing fractions of the two classes, $\phi_1^{\alpha*}$ and $\phi_1^{\beta*}$, for fixed processor capacities, $C_1 = 2, C_2 = 1$, fixed class β arrival rate, $\lambda^\beta = 1$ and different class α arrival rates, λ^α. In this case, for a specific λ^α, we have a multiplicity of solutions $(\phi_1^{\alpha*}, \phi_1^{\beta*})$. Also, in the team optimum solution, the set of $\phi_1^{\alpha*}$ intersect with the set of $\phi_1^{\beta*}$ not in just one point, but over a large range of values. Finally, the curve $\phi_1^\alpha = \phi_1^\beta$ is the optimum load sharing solution when the two classes are treated as a single class.

Thus, we have presented 3 approaches for multi-class load sharing. In the first approach, which is the usual approach [45] for the load sharing problem, the two classes are treated as a single class and we give the load sharing policy that minimizes the average job delay. In the second approach, which is the team optimization approach for the multi-class cooperative load sharing problem, the two classes are treated differently, however both classes cooperate to minimize the average job delay. We give the load sharing policy that minimizes the average job delay and further investigate the achieved team optimum solution. Note that the first approach can be considered as a special case of the second approach. Finally,

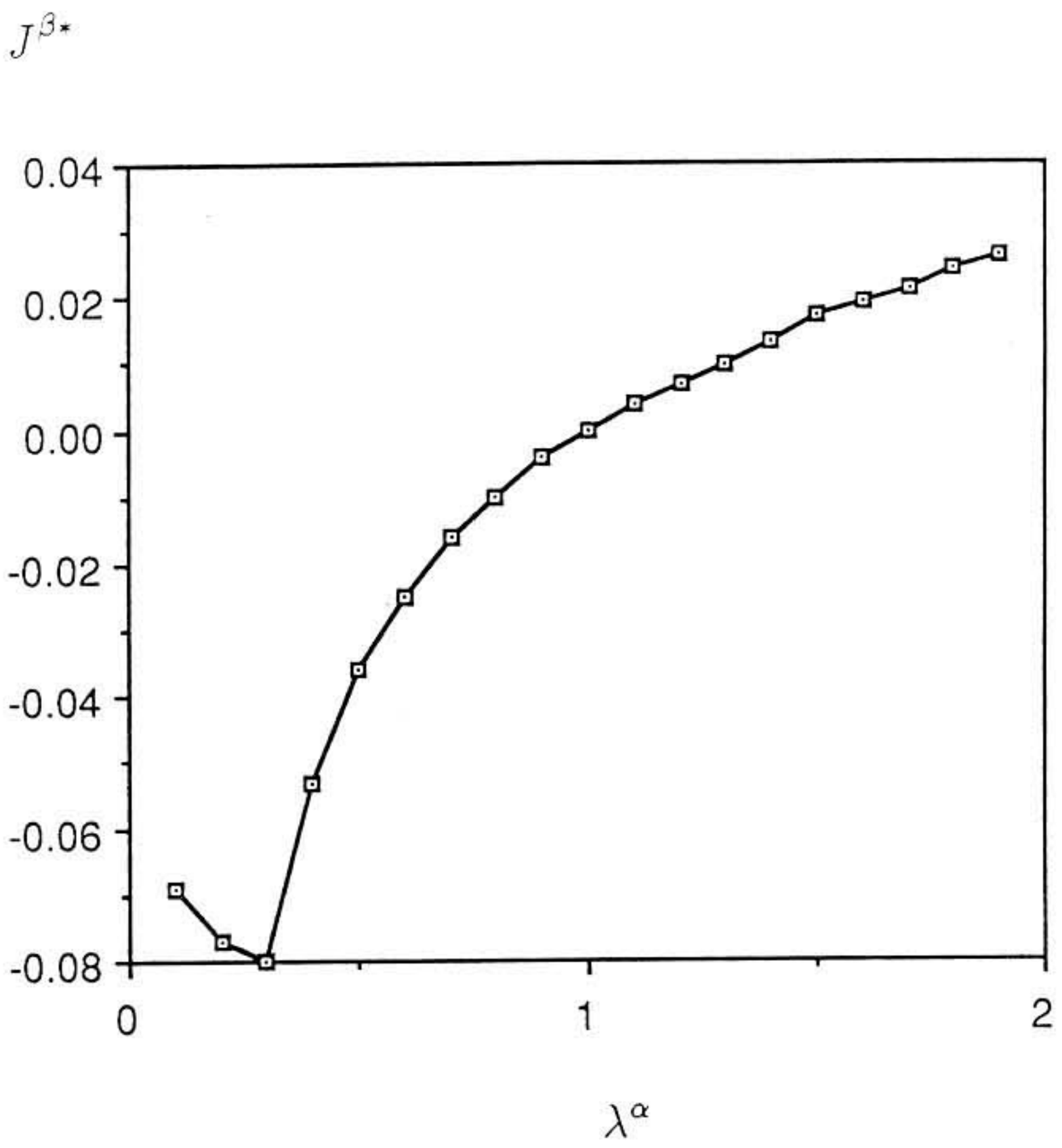


Figure 4.7: For the Nash equilibrium solution, the average delay difference between the two classes $J^{\alpha*} - J^{\beta*}$ for fixed processor capacities $C_1 = 2, C_2 = 1$, fixed class β arrival rate $\lambda^{\beta} = 1$ and different class α arrival rates λ^{α}.

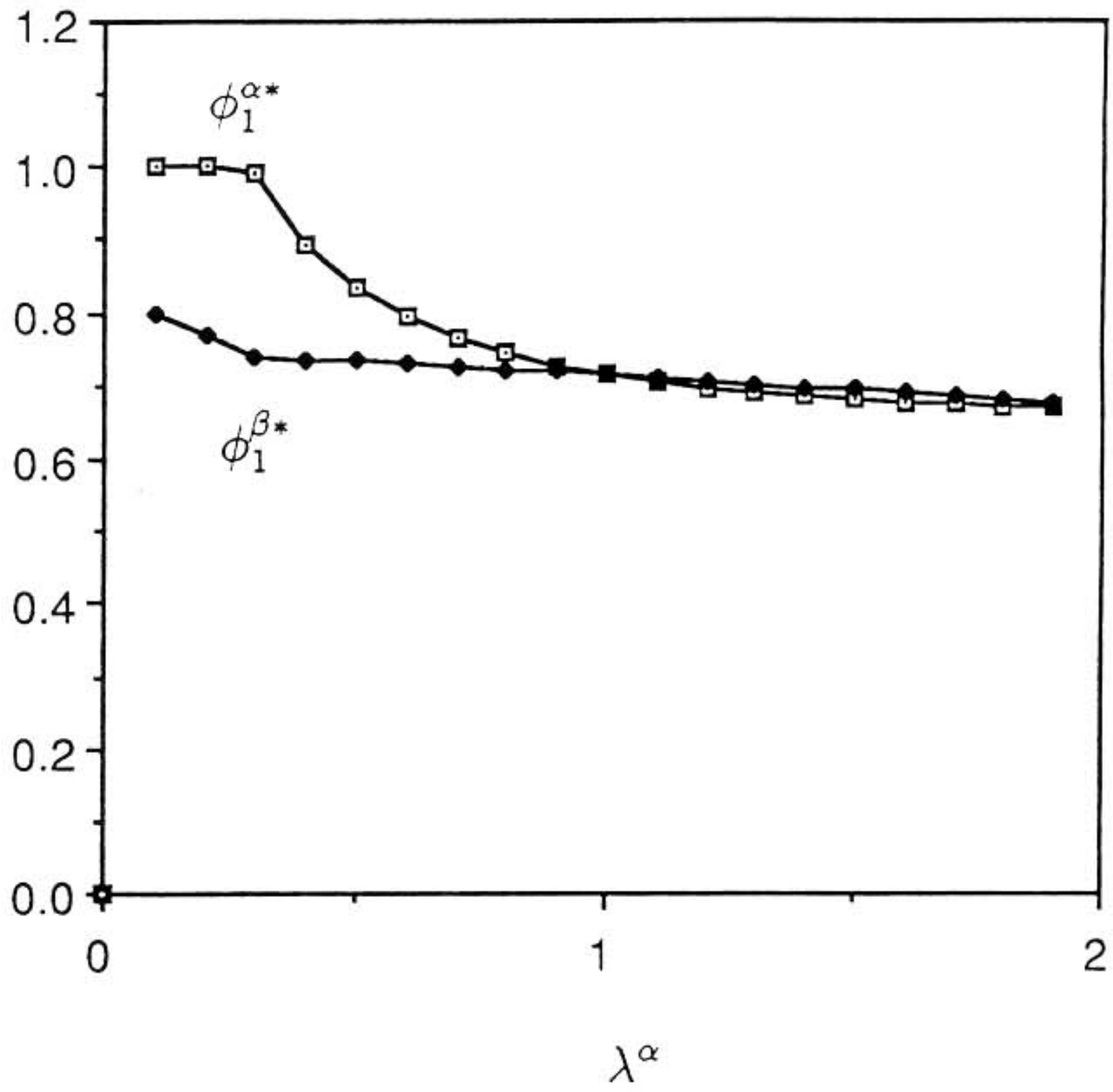


Figure 4.8: Nash equilibrium load sharing fractions of the two classes $\phi_1^{\alpha*}$ and $\phi_1^{\beta*}$, for fixed processor capacities, $C_1 = 2, C_2 = 1$, fixed class β arrival rate, $\lambda^\beta = 1$ and different class α arrival rates, λ^α.

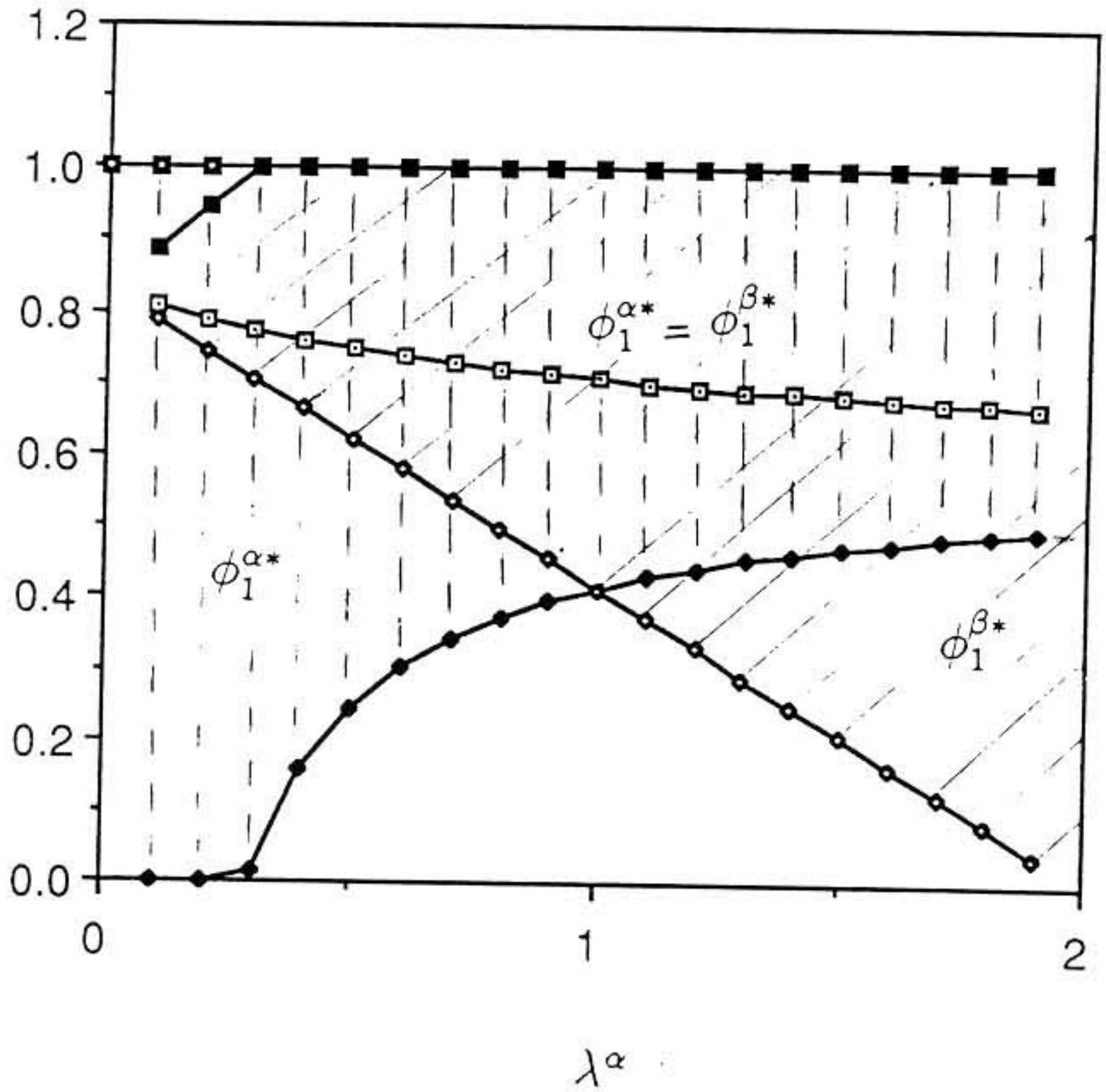


Figure 4.9: Team optimum load sharing fractions of the two classes, $\phi_1^{\alpha*}$ and $\phi_1^{\beta*}$, for fixed processor capacities, $C_1 = 2, C_2 = 1$, fixed class β arrival rate, $\lambda^\beta = 1$ and different class α arrival rates, λ^α.

in the third approach, which is the Nash game approach for the multi-class non-cooperative load sharing problem, the two classes compete for the two processors and each class tries to minimize the average delay of its own jobs. We give the load sharing policy that is the result of this competition and further investigate the achieved Nash equilibrium solution.

4.6 Example 2: Two-priority classes Two- processors

4.6.1 Introduction

In this section, we consider a two preemptive resume priority class two-processor load sharing (or routing) problem. An application is load sharing for a multiprocessor system, where interactive jobs (high priority) and batch jobs (low priority) may use two processors for execution. Another application is routing, where voice packets (high priority) and data packets (low priority) may use two different links for transmission between source-destination. We formulate this priority multi-objective optimization problem as a non cooperative Stackelberg game between the two priority classes.

When a high priority job is assigned to a processor, if there is another high priority job there, then it is put in the queue. If there are only low priority jobs there, then the low priority job is preempted and the high priority one starts been executing immediately. When all the high priority jobs have finished receiving service, then the low priority job that was preempted resumes and continues receiving service [256, 45]. The high priority class α (leader) assigns its jobs to the two processors, such that the average delay of its jobs is minimized. On the other hand, the low priority class β (follower) assigns its jobs to the two processors, such that the average delay of its jobs is minimized, after the high priority class α has optimally assigned its jobs. Thus a Stackelberg equilibrium is achieved.

4.6.2 Stackelberg Equilibrium Solution

The cost function that we use for the high preemptive resume priority class α is its average job delay [256]:

$$J^{\alpha}(\phi_1^{\alpha}, \phi_2^{\alpha}) = \frac{\phi_1^{\alpha} * \frac{1}{\mu^{\alpha} C_1}}{1 - \frac{\lambda^{\alpha} \phi_1^{\alpha}}{\mu^{\alpha} C_1}} + \frac{\phi_2^{\alpha} * \frac{1}{\mu^{\alpha} C_2}}{1 - \frac{\lambda^{\alpha} \phi_2^{\alpha}}{\mu^{\alpha} C_2}}$$

This is a strictly convex function with respect to $(\phi_1^\alpha, \phi_2^\alpha)$ over the convex space $\phi_1^\alpha + \phi_2^\alpha = 1, \quad \phi_1^\alpha, \quad \phi_2^\alpha \geq 0$, for $\mu^\alpha C_1 - \lambda^\alpha \phi_1^\alpha > 0$ and $\mu^\alpha C_2 - \lambda^\alpha \phi_2^\alpha > 0$.

Similarly, the cost function for the low preemptive resume priority class β is its average job delay [256]:

$$J^\beta(\phi_1^\alpha, \phi_2^\alpha, \phi_1^\beta, \phi_2^\beta) = \frac{\phi_1^\beta * [\frac{1}{\mu^\beta C_1} - \frac{\lambda^\alpha \phi_1^\alpha}{\mu^\alpha C_1 * \mu^\beta C_1} + \frac{\lambda^\alpha \phi_1^\alpha}{(\mu^\alpha C_1)^2}]}{(1 - \frac{\lambda^\alpha \phi_1^\alpha}{\mu^\alpha C_1}) * (1 - \frac{\lambda^\alpha \phi_1^\alpha}{\mu^\alpha C_1} - \frac{\lambda^\beta \phi_1^\beta}{\mu^\beta C_1})} +$$

$$+ \frac{\phi_2^\beta * [\frac{1}{\mu^\beta C_2} - \frac{\lambda^\alpha \phi_2^\alpha}{\mu^\alpha C_2 * \mu^\beta C_2} + \frac{\lambda^\alpha \phi_2^\alpha}{(\mu^\alpha C_2)^2}]}{(1 - \frac{\lambda^\alpha \phi_2^\alpha}{\mu^\alpha C_2}) * (1 - \frac{\lambda^\alpha \phi_2^\alpha}{\mu^\alpha C_2} - \frac{\lambda^\beta \phi_2^\beta}{\mu^\beta C_2})}$$

This is a strictly convex function with respect to $(\phi_1^\beta + \phi_2^\beta)$ over the convex space $\phi_1^\beta + \phi_2^\beta = 1, \quad \phi_1^\beta, \quad \phi_2^\beta \geq 0$, for $1 - \frac{\lambda^\alpha \phi_1^\alpha}{\mu^\alpha C_1} > 0$, $1 - \frac{\lambda^\alpha \phi_1^\alpha}{\mu^\alpha C_1} - \frac{\lambda^\beta \phi_1^\beta}{\mu^\beta C_1} > 0$, $1 - \frac{\lambda^\alpha \phi_2^\alpha}{\mu^\alpha C_2} > 0$, and $1 - \frac{\lambda^\alpha \phi_2^\alpha}{\mu^\alpha C_2} - \frac{\lambda^\beta \phi_2^\beta}{\mu^\beta C_2} > 0$.

It is also a strictly convex function with respect to $(\phi_1^\alpha + \phi_2^\alpha)$ over the convex space $\phi_1^\alpha + \phi_2^\alpha = 1, \quad \phi_1^\alpha, \quad \phi_2^\alpha \geq 0$, $1 - \frac{\lambda^\alpha \phi_1^\alpha}{\mu^\alpha C_1} > 0$, $1 - \frac{\lambda^\alpha \phi_1^\alpha}{\mu^\alpha C_1} - \frac{\lambda^\beta \phi_1^\beta}{\mu^\beta C_1} > 0$, $1 - \frac{\lambda^\alpha \phi_2^\alpha}{\mu^\alpha C_2} > 0$, and $1 - \frac{\lambda^\alpha \phi_2^\alpha}{\mu^\alpha C_2} - \frac{\lambda^\beta \phi_2^\beta}{\mu^\beta C_2} > 0$.

The overall average job delay is:

$$J(\phi_1^\alpha, \phi_2^\alpha, \phi_1, \phi_2^\beta) = \frac{\lambda^\alpha}{\lambda^\alpha + \lambda^\beta} * J^\alpha(\phi_1^\alpha, \phi_2^\alpha) + \frac{\lambda^\beta}{\lambda^\alpha + \lambda^\beta} * J^\beta(\phi_1^\alpha, \phi_2^\alpha, \phi_1^\beta, \phi_2^\beta)$$

Theorem 1: *existence*

Let two preemptive resume priority classes of jobs compete for two processors. Jobs from each class arrive according to Poisson distribution and require service according to exponential distribution. Each class tries to minimize the average delay of its own jobs. For the above load sharing problem, there exists a Stackelberg equilibrium.

Proof: This is a two player non zero-sum continuous kernel game on the square, for which the follower's cost functional $J^{\beta}(\phi_1^{\alpha}, \phi_2^{\alpha}, \phi_1^{\beta}, \phi_2^{\beta})$ is strictly convex with respect to the leader's strategy over the convex space $\phi_1^{\alpha} + \phi_2^{\alpha} = 1, \quad \phi_1^{\alpha}, \ \phi_2^{\alpha} \geq 0$. Therefore, it admits a Stackelberg equilibrium solution. □

The high preemptive resume priority class α solves the following problem:

minimize

$$J^{\alpha}(\phi_1^{\alpha}, \phi_2^{\alpha}) = \frac{\phi_1^{\alpha}}{\mu^{\alpha} C_1 - \lambda^{\alpha} \phi_1^{\alpha}} + \frac{\phi_2^{\alpha}}{\mu^{\alpha} C_2 - \lambda^{\alpha} \phi_2^{\alpha}}$$

with respect to $\phi_1^{\alpha}, \ \phi_2^{\alpha}$

such that $\phi_1^{\alpha} + \phi_2^{\alpha} = 1, \quad \phi_1^{\alpha}, \ \phi_2^{\alpha} \geq 0.$

On the other hand, the low preemptive resume priority class β solves the following problem:

minimize

$$J^{\beta}(\phi_1^{\alpha*}, \phi_2^{\alpha*}, \phi_1^{\beta}, \phi_2^{\beta}) = \frac{\phi_1^{\beta} * [\frac{C_1}{\mu^{\beta}} - \frac{\lambda^{\alpha}\phi_1^{\alpha*}}{\mu^{\alpha}\mu^{\beta}} + \frac{\lambda^{\alpha}\phi_1^{\alpha*}}{(\mu^{\alpha})^2}]}{(C_1 - \frac{\lambda^{\alpha}\phi_1^{\alpha*}}{\mu^{\alpha}}) * (C_1 - \frac{\lambda^{\alpha}\phi_1^{\alpha*}}{\mu^{\alpha}} - \frac{\lambda^{\beta}\phi_1^{\beta}}{\mu^{\beta}})} +$$

$$+ \frac{\phi_2^{\beta} * [\frac{C_2}{\mu^{\beta}} - \frac{\lambda^{\alpha}\phi_2^{\alpha*}}{\mu^{\alpha}\mu^{\beta}} + \frac{\lambda^{\alpha}\phi_2^{\alpha*}}{(\mu^{\alpha})^2}]}{(C_2 - \frac{\lambda^{\alpha}\phi_2^{\alpha*}}{\mu^{\alpha}}) * (C_2 - \frac{\lambda^{\alpha}\phi_2^{\alpha*}}{\mu^{\alpha}} - \frac{\lambda^{\beta}\phi_2^{\beta}}{\mu^{\beta}})}$$

with respect to $\phi_1^\beta,\ \phi_2^\beta$

such that $\phi_1^\beta + \phi_2^\beta = 1, \quad \phi_1^\beta,\ \phi_2^\beta \geq 0.$

Let define the auxiliary variables:

$$C_1^\alpha = C_1 - \frac{\lambda^\alpha \phi_1^{\alpha*}}{\mu^\alpha}$$

$$C_2^\alpha = C_2 - \frac{\lambda^\alpha \phi_2^{\alpha*}}{\mu^\alpha}$$

$$C_1^\beta = \frac{C_1}{\mu^\beta} - \frac{\lambda^\alpha \phi_1^{\alpha*}}{\mu^\alpha \mu^\beta} + \frac{\lambda^\alpha \phi_1^{\alpha*}}{(\mu^\alpha)^2}$$

$$C_2^\beta = \frac{C_2}{\mu^\beta} - \frac{\lambda^\alpha \phi_2^{\alpha*}}{\mu^\alpha \mu^\beta} + \frac{\lambda^\alpha \phi_2^{\alpha*}}{(\mu^\alpha)^2}$$

Then, the following policy allocates the arriving jobs to the two processors such that a Stackelberg equilibrium is achieved:

$$If \qquad \frac{\lambda^{\alpha}}{\mu^{\alpha}} \leq C_1' + C_2',$$

$then$

$$If \quad C_1 - \sqrt{C_1 C_2} \leq \frac{\lambda^{\alpha}}{\mu^{\alpha}} \quad and \quad C_2 - \sqrt{C_1 C_2} \leq \frac{\lambda^{\alpha}}{\mu^{\alpha}}$$

$$then \quad \phi_1^{\alpha *} = \frac{C_1}{\frac{\lambda^{\alpha}}{\mu^{\alpha}}} - \frac{C_1 + C_2 - \frac{\lambda^{\alpha}}{\mu^{\alpha}}}{\frac{\lambda^{\alpha}}{\mu^{\alpha}}} * \frac{\sqrt{C_1}}{\sqrt{C_1} + \sqrt{C_2}}$$

$$If \quad 0 \leq \frac{\lambda^{\alpha}}{\mu^{\alpha}} \leq C_1 - \sqrt{C_1 C_2}$$

$$then \quad \phi_1^{\alpha *} = 1$$

$$If \quad 0 \leq \frac{\lambda^{\alpha}}{\mu^{\alpha}} \leq C_2 - \sqrt{C_1 C_2}$$

$$then \quad \phi_1^{\alpha *} = 0$$

$$If \qquad \frac{\lambda^{\beta}}{\mu^{\beta}} \leq C_1^{\alpha} + C_2^{\alpha}$$

$then$

$$If \quad C_1^{\alpha} - C_2^{\alpha} * \sqrt{\frac{C_1^{\beta}}{C_2^{\beta}}} \leq \frac{\lambda^{\beta}}{\mu^{\beta}} \quad and \quad C_2^{\alpha} - C_1^{\alpha} * \sqrt{\frac{C_2^{\beta}}{C_1^{\beta}}} \leq \frac{\lambda^{\beta}}{\mu^{\beta}}$$

$$then \quad \phi_1^{\beta *} = \frac{\frac{C_1^{\alpha}}{\mu^{\beta}}}{\frac{\lambda^{\beta}}{\mu^{\beta}}} - \frac{C_1^{\alpha} + C_2^{\alpha} - \frac{\lambda^{\beta}}{\mu^{\beta}}}{\frac{\lambda^{\beta}}{\mu^{\beta}}} * \frac{\sqrt{C_1^{\beta}}}{\sqrt{C_1^{\beta}} + \sqrt{C_2^{\beta}}}$$

$$If \quad \frac{\lambda^{\beta}}{\mu^{\beta}} \leq C_1^{\alpha} - C_2^{\alpha} * \sqrt{\frac{C_1^{\beta}}{C_2^{\beta}}}$$

$$then \quad \phi_1^{\beta *} = 1$$

$$If \quad \frac{\lambda^{\beta}}{\mu^{\beta}} \leq C_2^{\alpha} - C_2^{\alpha} * \sqrt{\frac{C_2^{\beta}}{C_1^{\beta}}}$$

$$then \quad \phi_1^{\beta *} = 0$$

Of course, the Stackelberg equilibrium load sharing fractions to the other processor are $\phi_2^{\alpha *} = 1 - \phi_1^{\alpha *}$ and $\phi_2^{\beta *} = 1 - \phi_1^{\beta *}$.

Substituting the Stackelberg equilibrium fractions into the average delay functions, we have the Stackelberg equilibrium outcome of the game. Let a two processor system $C_1 \geq C_2$. Then we consider several cases:

Case 1: If $C_1 - \sqrt{C_1 C_2} \leq \frac{\lambda^{\alpha}}{\mu^{\alpha}}$, then the Stackelberg equilibrium load sharing decisions for the high priority class are given by:

$$\phi_1^{\alpha*} = \frac{C_1}{\frac{\lambda^\alpha}{\mu^\alpha}} - \frac{C_1 + C_2 - \frac{\lambda^\alpha}{\mu^\alpha}}{\frac{\lambda^\alpha}{\mu^\alpha}} * \frac{\sqrt{C_1}}{\sqrt{C_1} + \sqrt{C_2}}$$

$$\phi_2^{\alpha*} = \frac{C_2}{\frac{\lambda^\alpha}{\mu^\alpha}} - \frac{C_1 + C_2 - \frac{\lambda^\alpha}{\mu^\alpha}}{\frac{\lambda^\alpha}{\mu^\alpha}} * \frac{\sqrt{C_2}}{\sqrt{C_1} + \sqrt{C_2}}$$

and the average delay of the high priority jobs is:

$$J^\alpha(\phi_1^{\alpha*}, \phi_2^{\alpha*}) = \frac{(\sqrt{C_1} + \sqrt{C_2})^2}{\lambda^\alpha * (C_1 + C_2 - \frac{\lambda^\alpha}{\mu^\alpha})} - \frac{2}{\lambda^\alpha}$$

The auxiliary variables become

$$\mathcal{C}_1^\alpha = (C_1 + C_2 - \frac{\lambda^\alpha}{\mu^\alpha}) * \frac{\sqrt{C_1}}{\sqrt{C_1} + \sqrt{C_2}}$$

$$\mathcal{C}_2^\alpha = (C_1 + C_2 - \frac{\lambda^\alpha}{\mu^\alpha}) * \frac{\sqrt{C_2}}{\sqrt{C_2} + \sqrt{C_2}}$$

$$\mathcal{C}_1^\beta = \frac{C_1}{\mu^\alpha} + (C_1 + C_2 - \lambda^\alpha/\mu^\alpha) * (\frac{1}{\mu^\beta} - \frac{1}{\mu^\alpha}) * \frac{\sqrt{C_1}}{\sqrt{C_1} + \sqrt{C_2}}$$

$$\mathcal{C}_2^\beta = \frac{C_2}{\mu^\alpha} + (C_1 + C_2 - \lambda^\alpha/\mu^\alpha) * (\frac{1}{\mu^\beta} - \frac{1}{\mu^\alpha}) * \frac{\sqrt{C_2}}{\sqrt{C_1} + \sqrt{C_2}}$$

Then the load sharing decisions for the low priority class β are:

$$\phi_1^{\beta*} = \frac{\mathcal{C}_1^\alpha}{\frac{\lambda^\beta}{\mu^\beta}} - \frac{C_1 + C_2 - \frac{\lambda^\alpha}{\mu^\alpha} - \frac{\lambda^\beta}{\mu^\beta}}{\frac{\lambda^\beta}{\mu^\beta}} * \frac{\sqrt{\mathcal{C}_1^\beta}}{\sqrt{\mathcal{C}_1^\beta} + \sqrt{\mathcal{C}_2^\beta}}$$

$$\phi_2^{\beta*} = \frac{\mathcal{C}_2^\alpha}{\frac{\lambda^\beta}{\mu^\beta}} - \frac{\mathcal{C}_1^\alpha + \mathcal{C}_2^\alpha - \frac{\lambda^\alpha}{\mu^\alpha} - \frac{\lambda^\beta}{\mu^\beta}}{\frac{\lambda^\beta}{\mu^\beta}} * \frac{\sqrt{\mathcal{C}_2^\beta}}{\sqrt{\mathcal{C}_1^\beta} + \sqrt{\mathcal{C}_2^\beta}}$$

and the Stackelberg equilibrium average delay of the low priority jobs is:

$$J^{\beta}(\phi_1^{\alpha*}, \phi_2^{\alpha*}, \phi_1^{\beta*}, \phi_2^{\beta*}) = \frac{(\sqrt{C_1^{\beta}} - \sqrt{C_2^{\beta}})^2}{\frac{\lambda^{\beta}}{\mu^{\beta}} * (C_1 + C_2 - \frac{\lambda^{\alpha}}{\mu} - \frac{\lambda^{\beta}}{\mu^{\beta}})} -$$

$$- \frac{(\sqrt{C_1} + \sqrt{C_2}) * (C_1^{\beta}\sqrt{C_2} + C_2^{\beta}\sqrt{C_1})}{\frac{\lambda^{\beta}}{\mu^{\beta}} * \sqrt{C_1 C_2} * (C_1 + C_2 - \frac{\lambda^{\alpha}}{\mu^{\alpha}})}$$

Case 2: If $C_1 - \sqrt{C_1 C_2} \geq \frac{\lambda^{\alpha}}{\mu^{\alpha}}$,

then the Stackelberg equilibrium load sharing decisions for the high priority class α are given by: $\phi_1^{\alpha*} = 1, \quad \phi_2^{\alpha*} = 0$

and the average delay of the high priority jobs is:

$$J^{\alpha}(\phi_1^{\alpha*}, \phi_2^{\alpha*}) = \frac{1}{\mu^{\alpha} C_1 - \lambda^{\alpha}}$$

The auxiliary variables become

$$C_1^{\alpha} = C_1 - \frac{\lambda^{\alpha}}{\mu^{\alpha}}$$

$$C_2^{\alpha} = C_2$$

$$C_1^{\beta} = \frac{C_1}{\mu^{\beta}} - \frac{\lambda^{\alpha}}{\mu^{\alpha}\mu^{\beta}} + \frac{\lambda^{\alpha}}{(\mu_a)^2}$$

$$C_2^{\beta} = \frac{C_2}{\mu^{\beta}}$$

Then we consider two cases:

Case 2.1: If $C_1^{\alpha} - C_2^{\alpha} * \sqrt{\frac{C_1^{\beta}}{C_2^{\beta}}} \leq \frac{\lambda^{\beta}}{\mu} \quad and \quad C_2^{\alpha} - C_1^{\alpha} * \sqrt{\frac{C_2^{\beta}}{C_1^{\beta}}} \leq \frac{\lambda^{\beta}}{\mu}$

then the Stackelberg equilibrium load sharing decisions for the low priority class β are given by:

$$\phi_1^{\beta*} = \frac{C_1 - \frac{\lambda^\alpha}{\mu^\alpha}}{\frac{\lambda^\beta}{\mu^\beta}} - \frac{C_1 + C_2 - \frac{\lambda^\alpha}{\mu^\alpha} - \frac{\lambda^\beta}{\mu^\beta}}{\frac{\lambda^\beta}{\mu^\beta}} * \frac{\sqrt{C_1^\beta}}{\sqrt{C_1^\beta} + \sqrt{C_2^\beta}}$$

$$\phi_2^{\beta*} = \frac{C_2}{\frac{\lambda^\beta}{\mu^\beta}} - \frac{C_1 + C_2 - \frac{\lambda^\alpha}{\mu^\alpha} - \frac{\lambda^\beta}{\mu^\beta}}{\frac{\lambda^\beta}{\mu^\beta}} * \frac{\sqrt{C_2^\beta}}{\sqrt{C_1^\beta} + \sqrt{C_2^\beta}}$$

$$J^\beta(\phi_1^{\alpha*}, \phi_2^{\alpha*}, \phi_1^\beta, \phi_2^\beta) = \frac{(\sqrt{C_1^\beta} + \sqrt{C^\beta}_2)^2}{\frac{\lambda^\beta}{\mu^\beta} * (C_1 + C_2 - \frac{\lambda^\alpha}{\mu^\alpha} - \frac{\lambda^\beta}{\mu^\beta})} -$$

$$- \frac{C_1^\beta C_2 + C_2^\beta (C_1 - \frac{\lambda^\alpha}{\mu^\alpha})}{\frac{\lambda^\beta}{\mu^\beta} * (C_1 - \frac{\lambda^\alpha}{\mu^\alpha}) * C_2}$$

Case 2.2: If $C_1^\alpha - C_2^\alpha * \sqrt{\frac{C_1^\beta}{C_2^\beta}} \geq \frac{\lambda^\beta}{\mu}$,

then the Stackelberg equilibrium load sharing decisions for the low priority class β are given by: $\phi_1^{\beta*} = 1 \quad \phi_2^{\beta*} = 0$,

and the Stackelberg equilibrium average delay of the low priority jobs is:

$$J^\beta(\phi_1^{\alpha*}, \phi_2^{\alpha*}, \phi_1^{\beta*}, \phi_2^{\beta*}) = \frac{C_1^\beta}{(C_1 - \frac{\lambda^\alpha}{\mu^\alpha}) * (C_1 - \frac{\lambda^\alpha}{\mu^\alpha} - \frac{\lambda^\beta}{\mu^\beta})}$$

4.6.3 Interesting Results

From the above Stackelberg equilibrium solution and outcome of the game, we have some interesting results:

Proposition 1: *Conservation law*

Let a two-processor system $C_1 \geq C_2$ that is shared by jobs from two preemptive resume priority classes. Jobs from the high priority class α arrive according to

Poisson distribution with rate λ^α and require service according to exponential distribution with mean μ. Jobs from the low priority class β arrive according to Poisson distribution with rate λ^β and require service according to exponential distribution with mean μ. Let also the total arrival rate is constant $\lambda^\alpha + \lambda^\beta = \lambda = constant$ and less than the mean service rate $\lambda^\alpha + \lambda^\beta \leq \mu(C_1 + C_2)$. If each class tries to minimize the average delay of its own jobs, then the average job delay is constant, i.e.

$$J(\phi_1^{\alpha*}, \phi_2^{\alpha*}, \phi_1^{\beta*}, \phi_2^{\beta*}) = constant$$

Proof: Case 1: If $C_1 - \sqrt{C_1 C_2} \leq \frac{\lambda^\alpha}{\mu}$, then the average delay of the high priority jobs is:

$$J^\alpha(\phi_1^{\alpha*}, \phi_2^{\alpha*}) = \frac{(\sqrt{C_1} + \sqrt{C_2})^2}{\lambda^\alpha * (C_1 + C_2 - \frac{\lambda^\alpha}{\mu})} - \frac{2}{\lambda^\alpha}$$

and the average delay of the low priority jobs is:

$$J^\beta(\phi_1^{\alpha*}, \phi_2^{\alpha*}, \phi_1^{\beta*}, \phi_2^{\beta*}) = \frac{(\sqrt{C_1} + \sqrt{C_2})^2}{\mu * (C_1 + C_2 - \frac{\lambda^\alpha}{\mu}) * (C_1 + C_2 - \frac{\lambda^\alpha}{\mu} - \frac{\lambda^\beta}{\mu})}$$

Finally, the overall average job delay is:

$$J(\phi_1^{\alpha*}, \phi_2^{\alpha*}, \phi_1^{\beta*}, \phi_2^{\beta*}) = \frac{(\sqrt{C_1} + \sqrt{C_2})^2}{\mu * \lambda * (C_1 + C_2 - \lambda)} - \frac{2}{\lambda}$$

Case 2.1: If $\frac{\lambda^\alpha}{\mu} \leq C_1 - \sqrt{C_1 C_2} \leq \frac{\lambda^\alpha}{\mu} + \frac{\lambda^\beta}{\mu}$ *and* $C_2 - (C_1 - \frac{\lambda^\alpha}{\mu}) * \sqrt{\frac{C_1}{C_2}} \leq \frac{\lambda^\beta}{\mu}$, then the average delay of the high priority jobs is:

$$J^\alpha(\phi_1^{\alpha*}, \phi_2^{\alpha*}) = \frac{1}{\mu C_1 - \lambda^\alpha}$$

and the average delay of the low priority jobs is:

$$J^{\beta}(\phi_1^{\alpha*}, \phi_2^{\alpha*}, \phi_1^{\beta*}, \phi_2^{\beta*}) = \frac{(\sqrt{C_1} + \sqrt{C_2})^2}{\lambda^{\beta} * (C_1 + C_2 - \frac{\lambda}{\mu})} - \frac{2C_1 - \frac{\lambda^{\alpha}}{\mu}}{\lambda^{\beta} * (C_1 - \frac{\lambda^{\alpha}}{\mu})}$$

Finally, the overall average job delay is:

$$J(\phi_1^{\alpha*}, \phi_2^{\alpha*}, \phi_1^{\beta*}, \phi_2^{\beta*}) = \frac{(\sqrt{C_1} + \sqrt{C_2})^2}{(\lambda^{\alpha} + \lambda^{\beta}) * (C_1 + C_2 - \frac{\lambda^{\alpha}}{\mu} - \frac{\lambda^{\beta}}{\mu})} - \frac{2}{\lambda^{\alpha} + \lambda^{\beta}}$$

Case 2.2: If $C_1 - \sqrt{C_1 C_2} \geq \frac{\lambda^{\alpha}}{\mu} + \frac{\lambda^{\beta}}{\mu}$,

and the Stackelberg equilibrium average delay of the low priority jobs is:

$$J^{\beta}(\phi_1^{\alpha*}, \phi_2^{\alpha*}, \phi_1^{\beta*}, \phi_2^{\beta*}) = \frac{C_1}{\mu * (C_1 - \frac{\lambda^{\alpha}}{\mu}) * (C_1 - \frac{\lambda^{\alpha}}{\mu} - \frac{\lambda^{\beta}}{\mu})}$$

Finally, the overall average job delay is:

$$J(\phi_1^{\alpha*}, \phi_2^{\alpha*}, \phi_1^{\beta*}, \phi_2^{\beta*}) = \frac{1}{\mu C_1 - \lambda}$$

Therefore for constant λ, the $J(\phi_1^{\alpha*}, \phi_2^{\alpha*}, \phi_1^{\beta*}, \phi_2^{\beta*})$ is also constant. □

The above proposition says that for equal mean service requirements for both priority classes and constant total arrival rate the overall average job delay is constant, i.e. it does not depend on the mix of high and low priority jobs.

In Figure 4.10, we show the Stackelberg equilibrium average delay of the high priority class α, $J^{\alpha}(\phi_1^{\alpha*}, \phi_2^{\alpha*})$, of the low priority class β, $J^{\beta}(\phi_1^{\alpha*}, \phi_2^{\alpha*}, \phi_1^{\beta*}, \phi_2^{\beta*})$, and of the system $J(\phi_1^{\alpha*}, \phi_2^{\alpha*}, \phi_1^{\beta*}, \phi_2^{\beta*})$ versus different mixes of the high and low priority arrival rates $\frac{\lambda^{\alpha}}{\lambda^{\beta}}$, for fixed processor capacities $C_1 = 2, C_2 = 1$, fixed total arrival rate $\lambda^{\alpha} + \lambda^{\beta} = 2.5$, and equal mean service requirement of the high and the low priority jobs $1/\mu^{\alpha} = 1/\mu^{\beta} = 1$. We note that the overall average job delay is constant and independent from the mix of the high and low priority jobs.

Proposition 2:

Let a two-processor system $C_1 \geq C_2$ that is shared by jobs from two preemptive resume priority classes. Jobs from the high priority class α arrive according

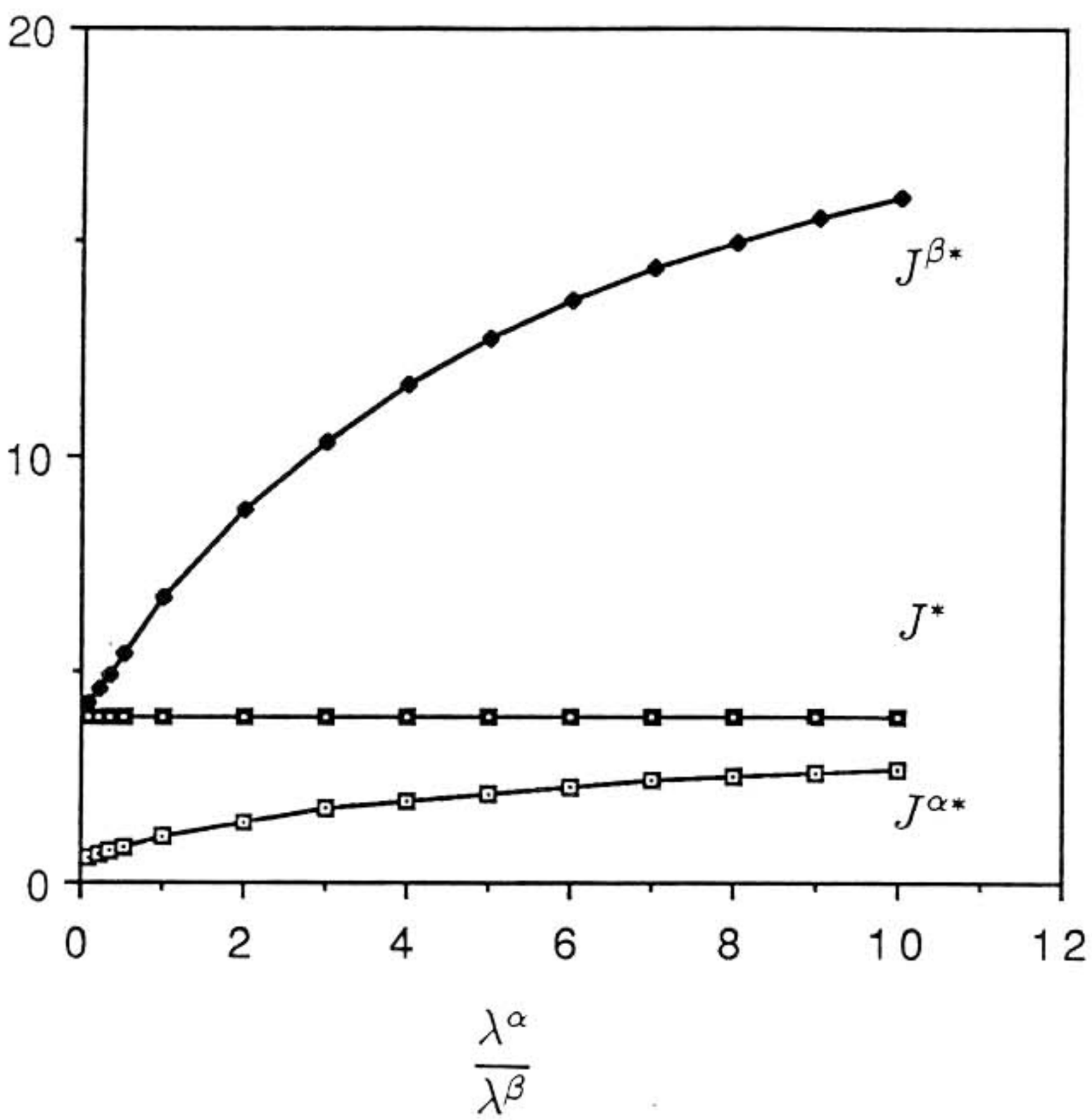


Figure 4.10: Stackelberg equilibrium average delay J^{α}, J^{β} and J versus different mixes of the high and low priority arrival rates $\frac{\lambda^{\alpha}}{\lambda^{\beta}}$, for $C_1 = 2, C_2 = 1$, $\lambda^{\alpha} + \lambda^{\beta} = 2.5$, and $\mu^{\alpha} = \mu^{\beta} = 1$.

to Poisson distribution with rate λ^α and require service according to exponential distribution with mean μ. Jobs from the low priority class β arrive according to Poisson distribution with rate λ^β and require service according to exponential distribution with mean μ. Let also the total arrival rate is less than the mean service rate $\lambda^\alpha + \lambda^\beta \leq \mu(C_1 + C_2)$. Let also that each class tries to minimize the average delay of its own jobs.

If $C_1 - \sqrt{C_1 C_2} \leq \lambda^\alpha / \mu$,

then $\phi_1^{\beta *} = \dfrac{\sqrt{C_1}}{\sqrt{C_1} + \sqrt{C_2}}$

i.e. the load sharing decisions of the low priority class is independent of the arrival rates λ^α and λ^β.

Proof: When $\mu^\alpha = \mu^\beta = \mu$, for Case 1, we have $C_1^\beta = \dfrac{C_1}{\mu}$, $C_2^\beta = \dfrac{C_2}{\mu}$. The proof follows immediately. □

What the above proposition says is that for equal mean service requirements for both priority classes, when the high priority class α uses both processors, then the Stackelberg equilibrium decisions of the low priority class β are constant and independent of the arrival rates $(\lambda^\alpha, \lambda^\beta)$. This result is not intuitive, because we might expect that the load sharing decisions for the low priority class β should also depend on the arrival rates (as it is the case for the high priority class α). It is also very important, because even when the arrival rates vary over time, the load sharing policy for the low priority jobs remains the same (Figure 4.11).

Proposition 3:

Let a two-processor system $C_1 \geq C_2$ that is shared by jobs from two preemptive resume priority classes. Jobs from the high priority class α arrive according to Poisson distribution with rate λ^α and require service according to exponential distribution with mean μ. Jobs from the low priority class β arrive according to Poisson distribution with rate λ^β and require service according to exponential distribution with mean μ. Let also the total arrival rate is less than the mean service rate $\lambda^\alpha + \lambda^\beta \leq \mu(C_1 + C_2)$. Let also that each class tries to minimize the average delay of its own jobs.

If $\dfrac{\lambda^\alpha}{\mu^\alpha} + \dfrac{\lambda^\beta}{\mu^\beta} \rightarrow C_1 + C_2$, *then* $\phi_1^{\beta *} \rightarrow \dfrac{\sqrt{C_1}}{\sqrt{C_1} + \sqrt{C_2}}$.

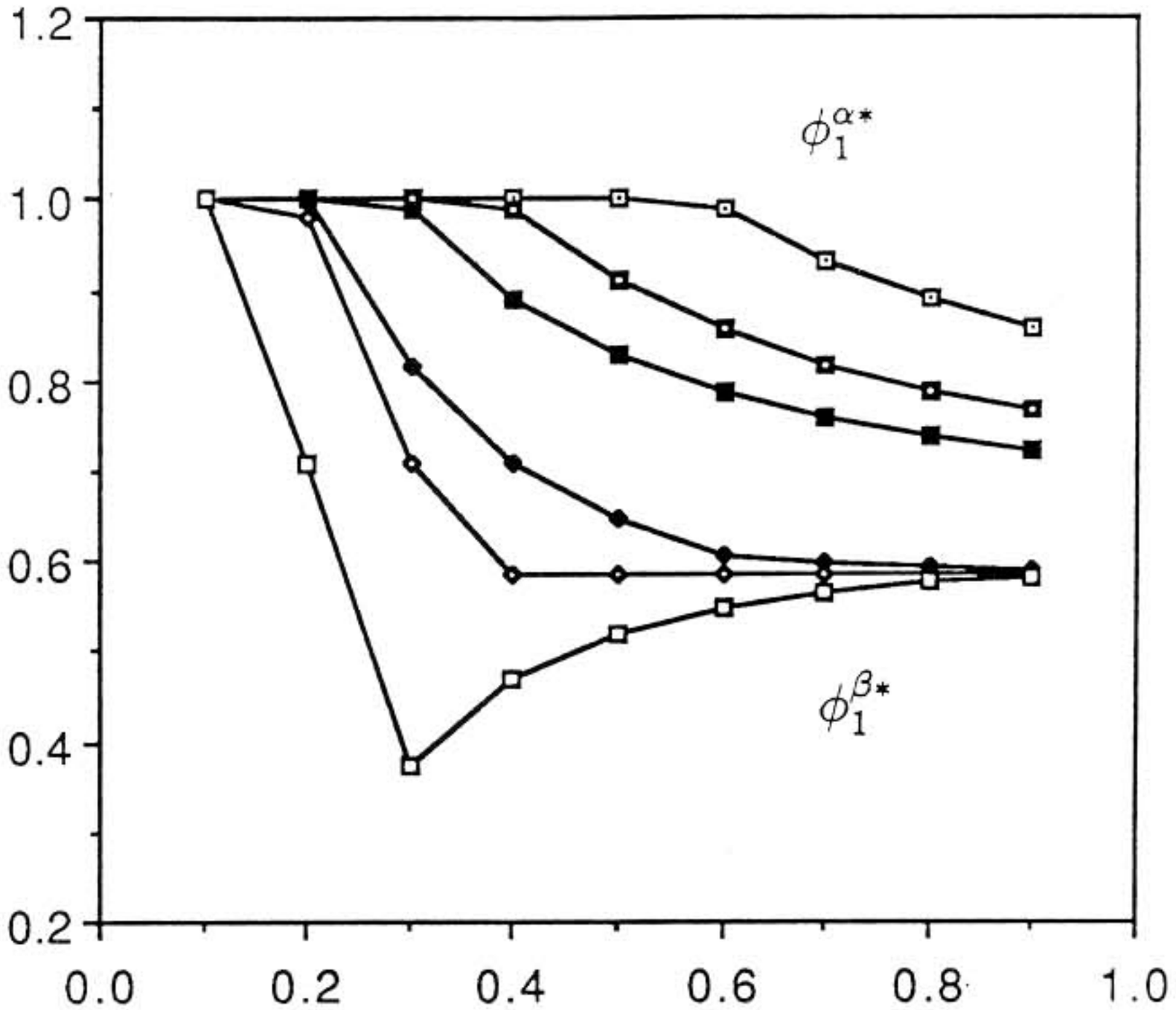


Figure 4.11: Stackelberg equilibrium fractions $\phi_1^{\alpha*}$ and $\phi_1^{\beta*}$ versus $\frac{\lambda^\alpha/\mu^\alpha + \lambda^\beta/\mu^\beta}{C_1 + C_2}$, for $C_1 = 2, C_2 = 1, \lambda^\alpha = \lambda^\beta$ and $\mu^\alpha = 2 * \mu^\beta = 1$, $\mu^\alpha = \mu^\beta = 1$ and $2 * \mu^\alpha = \mu^\beta = 1$.

Proof:

Case 1: If $C_1 - \sqrt{C_1 C_2} \leq \frac{\lambda^\alpha}{\mu^\alpha}$ *and* $C_2 - \sqrt{C_1 C_2} \leq \frac{\lambda^\alpha}{\mu^\alpha}$,

When the arriving service requirement $\frac{\lambda^\alpha}{\mu^\alpha} + \frac{\lambda^\beta}{\mu^\beta}$ approaches the total service capacity $C_1 + C_2$, then the low priority class β uses both processors. The Stackelberg equilibrium load sharing decisions for the low priority class β approach

$$\phi_1^{\beta *} \rightarrow \frac{\sqrt{C_1}}{\sqrt{C_1} + \sqrt{C_2}}$$

$$\phi_2^{\beta *} \rightarrow \frac{\sqrt{C_2}}{\sqrt{C_1} + \sqrt{C_2}}$$

Case 2: If $C_1 - \sqrt{C_1 C_2} \geq \frac{\lambda^\alpha}{\mu}$,

Then we consider two cases:

Case 2.1: If $C_1^\alpha - C_2^\alpha * \sqrt{\frac{C_1^\beta}{C_2^\beta}} \leq \frac{\lambda^\beta}{\mu}$ *and* $C_2^\alpha - C_1^\alpha * \sqrt{\frac{C_2^\beta}{C_1^\beta}} \leq \frac{\lambda^\beta}{\mu}$,

then the Stackelberg equilibrium load sharing decisions for the low priority class β approach the following constant values:

$$\phi_1^{\beta *} \rightarrow \frac{\sqrt{C_1}}{\sqrt{C_1} + \sqrt{C_2}}$$

$$\phi_2^{\beta *} \rightarrow \frac{\sqrt{C_2}}{\sqrt{C_1} + \sqrt{C_2}}$$

□

The above proposition says that even for different service requirements for the two priority classes, when the total arriving service requirement $\frac{\lambda^\alpha}{\mu^\alpha} + \frac{\lambda^\beta}{\mu^\beta}$ approaches the total service capacity $C_1 + C_2$, the Stackelberg load sharing decisions for the low priority class β become constant and independent from the arrival rates.

In **Figure** 4.11, we show the Stackelberg equilibrium load sharing fractions of both the high priority class α, $\phi_1^{\alpha *}$, and of the low priority class β, $\phi_1^{\beta *}$, versus the system load $\frac{\lambda^\alpha/\mu^\alpha + \lambda^\beta/\mu^\beta}{C_1 + C_2}$, for fixed processor capacities $C_1 = 2, C_2 = 1$, equal arrival rates $\lambda^\alpha = \lambda^\beta$ and different ratio of the mean service requirement of the high and the low priority jobs $1/\mu^\alpha = 2/\mu^\beta = 1$, $1/\mu^\alpha = 1/\mu^\beta = 1$, $1/\mu^\beta = 2/\mu^\alpha = 1$.

For equal mean service requirements of the high and low priority jobs, we note that when the high priority class α uses both processors ($0 < \phi_1^{\alpha *} < 1$), then the load sharing decisions $\phi_1^{\beta *}$ for the low priority class β are constant and independent of the arrival rates $\lambda^\alpha, \lambda^\beta$. For different mean service requirements of the high and low priority jobs, we note that when the system load $\frac{\lambda^\alpha/\mu^\alpha + \lambda^\beta/\mu^\beta}{C_1 + C_2}$ approaches 1, then the load sharing decisions $\phi_1^{\beta *}$ for the low priority class β approach the same constant value as for the case of equal mean service requirements.

In this section, we have explicitly solved a two processor load sharing problem, when two preemptive resume priority classes of jobs share the two processors. For equal mean service requirements of jobs from both classes, when the high priority class uses both processors, then the Stackelberg equilibrium decisions of the low priority jobs do not depend on the arrival rates of the jobs. That means that even if the arrival rates change during operation, our load sharing algorithm will still perform "optimally" for the low priority class. Also, for constant total arrival rate, even if we change the mix of high and low priority classes, then the overall average job delay remains constant.

4.6.4 Discussion

Next, we discuss some other results that can be derived from the Stackelberg solution of the two processor load sharing problem among jobs from two preemptive resume priority classes.

i) Constant λ^α

Consider a two processor system $C_1 \leq C_2$ with fixed arrival rate of interactive (high priority) jobs $\lambda^\alpha = constant$. If this multiprocessor is also to be used by batch (low priority) jobs and we want to secure an upper bound on the average delay of batch jobs $J^\beta \leq J_0^\beta$, then we should restrict the arrival rate of the batch jobs up to an upper limit. For example, if $\mu^\alpha = \mu^\beta = \mu$, Case 1, then

$$\frac{(\sqrt{C_1}+\sqrt{C_2})^2}{\mu*(C_1+C_2-\frac{\lambda^\alpha}{\mu})*(C_1+C_2-\frac{\lambda^\alpha}{\mu}-\frac{\lambda^\beta}{\mu})} \leq J_0^\beta \quad =\cdot$$

$$\lambda^\beta \leq \mu*(C_1+C_2)-\lambda^\alpha-\frac{(\sqrt{C_1}+\sqrt{C_2})^2}{J_0^\beta*(C_1+C_2-\frac{\lambda^\alpha}{\mu})}$$

In Figure 4.12 we show the Stackelberg equilibrium average delay of both the higher priority class α, $J^\alpha(\phi_1^{\alpha*},\phi_2^{\alpha*})$, of the lower priority class β, $J^\beta(\phi_1^{\alpha*},\phi_2^{\alpha*},\phi_1^*,\phi_2^{\beta*})$, and of the system $J(\phi_1^{\alpha*},\phi_2^{\alpha*},\phi_1^*,\phi_2^{\beta*})$ versus the low priority class β arrival rate, λ^β, for fixed processor capacities $C_1=2, C_2=1$, fixed mean service requirements $1/\mu^\alpha = 1/\mu^\beta = 1$ and fixed high priority class α arrival rate $\lambda^\alpha = 1.0$. So, for example, if the average delay of batch jobs J^β should be less than 10, then the arrival rate of batch jobs should be $\lambda^\beta < 1.71$.

ii) Constant λ^β

Next, consider a two processor system $C_1 \leq C_2$ with fixed arrival rate of batch jobs (low priority) $\lambda^\beta = constant$. If this multiprocessor is also to be used by interactive jobs (high priority) and we want to secure an upper bound on the average delay of batch jobs $J^\beta \leq J_0^\beta$, then we should restrict the arrival rate of the interactive jobs up to an upper limit. For example, if $\mu^\alpha = \mu^\beta = \mu$, Case 1, then

$$\frac{(\sqrt{C_1}+\sqrt{C_2})^2}{\mu*(C_1+C_2-\frac{\lambda^\alpha}{\mu})*(C_1+C_2-\frac{\lambda^\alpha}{\mu}-\frac{\lambda^\beta}{\mu})} \leq J_0^\beta \quad \Rightarrow$$

$$\lambda^\alpha \leq \mu(C_1+C_2)-\frac{\lambda^\beta}{2}-\sqrt{(\frac{\lambda^\beta}{2})^2+\frac{\mu(\sqrt{C_1}+\sqrt{C_2})^2}{J_0^\beta}}$$

In Figure 4.13, we show the Stackelberg equilibrium average delay of both the higher priority class α, $J^\alpha(\phi_1^{\alpha*},\phi_2^{\alpha*})$, of the lower priority class β, $J^\beta(\phi_1^{\alpha*},\phi_2^{\alpha*},\phi_1^*,\phi_2^{\beta*})$, and of the system $J(\phi_1^{\alpha*},\phi_2^{\alpha*},\phi_1^*,\phi_2^{\beta*})$ versus the high priority class α arrival rate, λ^α, for fixed processor capacities $C_1=2, C_2=1$, fixed mean service requirements $1/\mu^\alpha = 1/\mu^\beta = 1$. and fixed low priority class β arrival rate $\lambda^\beta = 1.0$.

So, for example, if the average delay of batch jobs J^β should be less than 10, then the arrival rate of interactive jobs should be $\lambda^\beta < 1.59$.

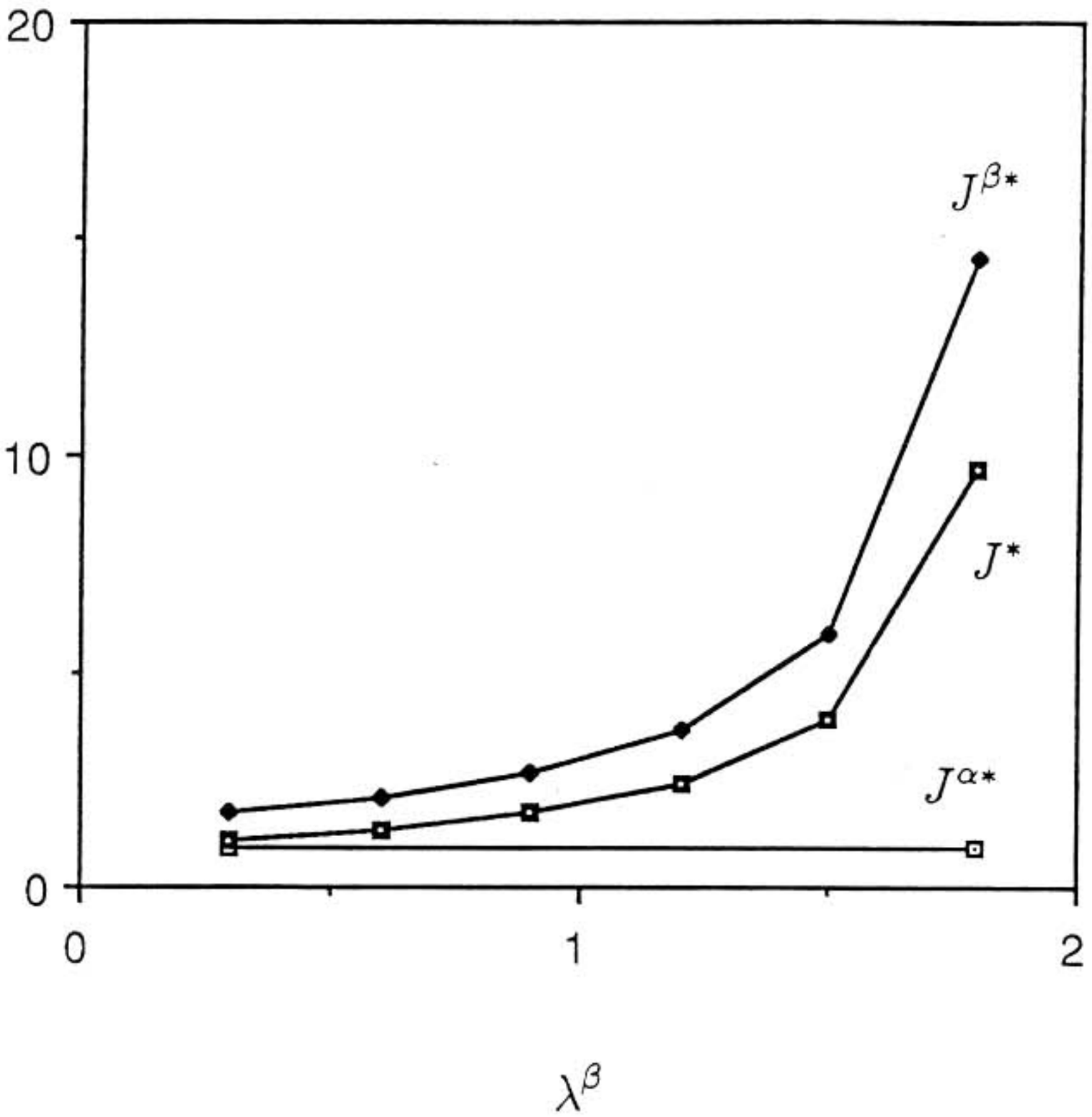


Figure 4.12: Stackelberg equilibrium average delay J^α, J^β and J versus λ^β, for $C_1 = 2, C_2 = 1$, $\mu^\alpha = \mu^\beta = 1$ $\lambda^\alpha = 1.0$.

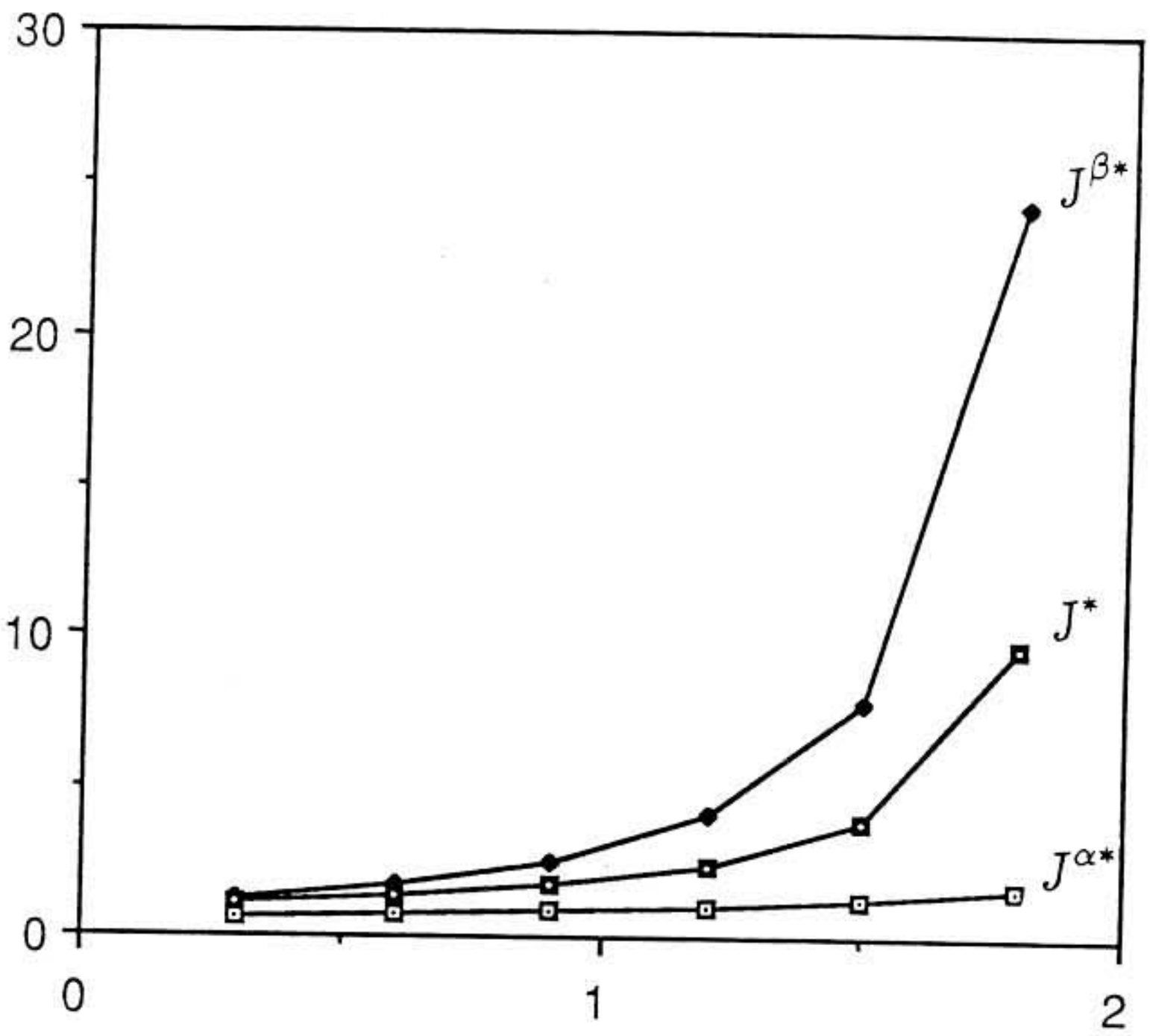


Figure 4.13: Stackelberg equilibrium average delay J^α, J^β and J versus λ^α, for $C_1 = 2, C_2 = 1$, $\mu^\alpha = \mu^\beta = 1$, and $\lambda^\beta = 1.0$.

iii) Constant λ

Thirdly, consider a two processor system $C_1 \leq C_2$ with fixed total arrival rate of both interactive and batch jobs $\lambda^\alpha + \lambda^\beta = \lambda = constant$. We want to determine what mix of interactive and batch jobs will secure an upper bound on the average delay of interactive jobs $J^\alpha \leq J_0^\alpha$ as well as on batch jobs $J^\beta \leq J_0^\beta$. Let $k = \frac{\lambda^\alpha}{\lambda^\beta}$ be the mix of interactive over batch jobs. Then we can write the arrival rate of interactive jobs as $\lambda^\alpha = \frac{k}{k+1}\lambda$ and the arrival rate of batch jobs as $\lambda^\beta = \frac{1}{k+1}\lambda$. For example, if $\mu^\alpha = \mu^\beta = 1$, Case 2.2, then

$$\frac{1}{\mu C_1 - \frac{k}{k+1}\lambda} \leq J_0^\alpha \Rightarrow \qquad k \leq \frac{J_0^\alpha * \mu C_1}{1 - J_0^\alpha * (\mu C_1 - \lambda)}$$

$$\frac{\mu C_1}{(\mu C_1 - \frac{k}{k+1}\lambda) * (\mu C_1 - \lambda)} \leq J_0^\beta \Rightarrow k \leq \frac{\mu C_1 * [J_0^\beta * (\mu C_1 - \lambda) - 1]}{\mu C_1 - J_0^\beta * (\mu C_1 - \lambda)^2}$$

In Figure 4.10, we show the Stackelberg equilibrium average delay of both the high priority class α, $J^\alpha(\phi_1^{\alpha*}, \phi_2^{\alpha*})$, of the low priority class β, $J^\beta(\phi_1^{\alpha*}, \phi_2^{\alpha*}, \phi_1^*, \phi_2^{\beta*})$, and of the system $J(\phi_1^{\alpha*}, \phi_2^{\alpha*}, \phi_1^*, \phi_2^{\beta*})$ versus different mix of the high and low priority arrival rates $k = \frac{\lambda^\alpha}{\lambda^\beta}$, for fixed processor capacities $C_1 = 2, C_2 = 1$, fixed total arrival rate $\lambda = \lambda^\alpha + \lambda^\beta = 2.5$, and equal mean service requirement of the high and the low priority jobs $1/\mu^\alpha = 1/\mu^\beta = 1$.

So, for example, if the average delay of interactive jobs should be less than 2 and the average delay of batch jobs should be less than 10, then mix of interactive and batch jobs should be $\frac{\lambda^\alpha}{\lambda^\beta} < 2.8$.

iv) Different Processor Rate Ratios

Finally, in Figure 4.14, we show the Stackelberg equilibrium fraction to processor 1, of both the high and low priority classes versus the system load $\frac{\lambda^\alpha/\mu^\alpha + \lambda^\beta/\mu^\beta}{C_1 + C_2}$ for equal arrival rates $\lambda^\alpha = \lambda^\beta$, equal mean service requirements $1/\mu^\alpha = 1/\mu^\beta = 1$ and different ration of the service rates of the two processors $\frac{C_1}{C_2} = 5, 4, 3, 2, 1, 1/2, 1/3, 1/4, 1/5$.

When the service rate of processor 1 is substantially larger than the service rate of processor 2, $(C_1 = 5 * C_2)$, then processor 1 is used exclusively for almost all

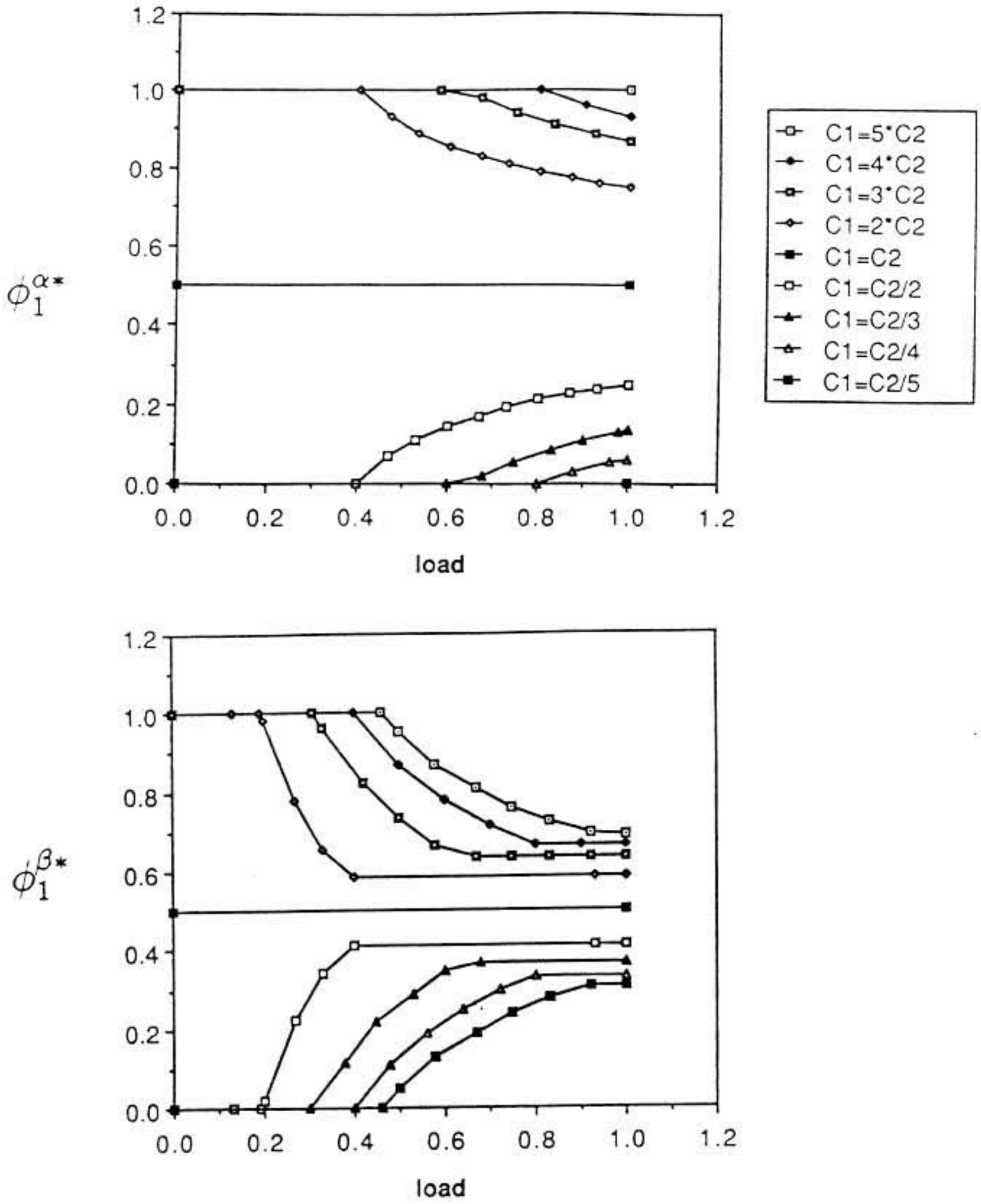


Figure 4.14: Stackelberg equilibrium fraction to processor 1, of both the high and low priority classes versus the system load $\frac{\lambda^\alpha/\mu^\alpha + \lambda^\beta/\mu^\beta}{C_1 + C_2}$ for equal arrival rates $\lambda^\alpha = \lambda^\beta$, equal mean service requirements $1/\mu^\alpha = 1/\mu^\beta = 1$ and different ration of the service rates of the two processors $\frac{C_1}{C_2} = 5, 4, 3, 2, 1, 1/2, 1/3, 1/4, 1/5$.

arrival rates. When the service rates are $C_1 = 4*C_2$, then for low and medium load, processor 1 is exclusively used, but for heavy system load, processor 2 also is used. When the service rates are $C_1 = 3*C_2$, then the slow processor starts been used for lower system load. When the service rates are $C_1 = 2*C_2$, then the slow processor starts been used for even lower system load. When the service rates are equal $C_1 = C_2$, then both processors are used equally ($\phi_1^{\alpha*} = \phi_2^{\alpha*} = \phi_1^{\beta*} = \phi_2^{\beta*} = 0.5$). Now, when processor 2 is faster, a similar scenario happens, i.e. the faster processor 2 is, the more it is exclusively used.

In this section, we formulated and solved a *priority load sharing* problem. Real distributed systems assign different priorities to different classes of jobs, in order to give preferential treatment to some classes of jobs. Therefore, it is not meaningful to optimize a single function over all different priority classes simultaneously. We formulated a two-priority class load sharing problem as a Stackelberg game with leader the high priority class and follower the low priority class. Furthermore, we gave the explicit solution when two preemptive resume priority classes want to minimize their average job delay. We found that for equal mean service requirements of jobs from both classes, when both processors are used, then the decisions of the low priority class do not depend on the arrival rates, i.e. even if the arrival rates vary, the same routing probabilities can be used for the low priority jobs. Also, for equal mean service requirements of jobs from both classes, when the total arrival rate of jobs is constant but the mix of high and low priority jobs varies, then the overall average job delay remains constant.

4.7 Application to Virtual Circuit Networks

In this section, we apply the methodologies developed in the previous sections to virtual circuit networks.

4.7.1 Cost Functions for Multiple Classes

In this section, we introduce several cost functions for multiple class virtual circuit networks. Consider a system resource (node. link, computer site, etc.). Without loss of generality, let its service rate be 1. Packets from class c arrive at rate λ_{ij}^c (Poisson) and require service (in FIF0 order) with mean $\overline{x^c}$ and second moment $\overline{(x^c)^2}$. Let also τ_{ij} be a flow independent constant delay (e.g. propagation delay) at resource ij.

Let a new packet finds N_Q^c packets in the queue from each class c and x_0 the residual service time of the packet in service. Let also $x^{c,n}$ be the service requirement of the n^{th} packet from class c at the queue. If we know all $x^{c,n}$, then the waiting time of a new packet is

$$W = x_0 + \sum_c \sum_{n=1}^{N_Q^c} x^{c,n}$$

Let a new packet finds N_Q^c packet in the queue from each class c and x_0 the residual service time of the packet in service. Let also $\overline{x^c}$ be the mean service requirement of packets from class c. Then the average waiting time of a new packet is

$$W = x_0 + \sum_c N_Q^c * \overline{x^c}$$

Let a new packet finds V^c virtual circuits from each class c and x_0 the residual service time of the packet in service. Let also for each class c, packets from each virtual circuit arrive at rate r^c with mean service requirement $\overline{x^c}$. Then the average waiting time of a new packet is

$$W = x_0 + \sum_c r^c * V^c * W * \overline{x^c} \Rightarrow W = \frac{x_0}{1 - \sum_c r^c * V^c * \overline{x^c}}$$

Let x_0 the residual service time of the packet in service. Let also for each class c, packets from each virtual circuit arrive at rate r^c with mean service requirement $\overline{x^c}$ and virtual circuits arrive at rate γ^c (Poisson) and stay for a duration with mean $1/\delta^c$ (general distribution). Then the average waiting time of a new packet is

$$W = \frac{x_0}{1 - \sum_c r^c * \frac{\gamma^c}{\delta^c} * \overline{x^c}}$$

The residual service time x_0 can be either measured (if we know the service requirement of the packet in service) or estimated.

Let the packet in service belongs to class c, then the estimated residual service time is $x_0 = \frac{\overline{(x^c)^2}}{2 * \overline{x^c}}$.

Let for each class c, there are N^c packets in the resource belonging to class c. The probability that there is a class c packet in service is $N^c/(1 + \sum_k N^k)$ and therefore the estimated residual service time is $x_0 = \sum_c \left\{ \frac{N^c}{1 + \sum_k N^k} * \frac{\overline{(x^c)^2}}{2 * \overline{x^c}} \right\}$.

Let for each class c, there are V^c virtual circuits in the resource belonging to class c. The packet arrival rate per virtual circuit for class c is r^c. Then the estimated residual service time is $x_0 = \frac{1}{2} * \sum_c r^c * V^c * \overline{(x^c)^2}$.

Let for each class c, the virtual circuit arrival rate (Poisson) is γ^c, its duration (general distribution) has mean $1/\delta^c$ and its packet rate is r^c. Then the estimated residual service time is $x_0 = \frac{1}{2} * \sum_c r^c * \frac{\gamma^c}{\delta^c} * \overline{(x^c)^2}$.

Then we may define as cost function for class c at resource ij the weighted average packet delay

$$g_{ij}^c = \frac{\lambda_{ij}^c}{\lambda} * [\overline{x^c} + W_{ij} + \tau_{ij}]$$

4.7.2 Cost Functions for Priority Classes

Let class α has non-preemptive priority over class β. By a similar development as in the previous section, we have several cases for the average waiting time for the high priority class α depending on our information:

i) $W^{\alpha} = x_o + \sum_{n=1}^{N_Q^{\alpha}} x^{\alpha,n}$.

ii) $W^{\alpha} = x_0 + N_Q^{\alpha} * \overline{x^{\alpha}}$.

iii) $W^{\alpha} = x_0 + r^{\alpha} * V^{\alpha} * W^{\alpha} * \overline{x^{\alpha}} \Rightarrow W^{\alpha} = \dfrac{x_0}{1 - r^{\alpha} * V^{\alpha} * \overline{x^{\alpha}}}$.

iv) $W^{\alpha} = \dfrac{x_0}{1 - r^{\alpha} * \dfrac{\gamma^{\alpha}}{\delta^{\alpha}} * \overline{x^{\alpha}}}$

Then we may define as cost function for class c at resource ij the weighted average packet delay

$$J_{ij}^{\alpha} = \frac{\lambda_{ij}^{\alpha}}{\lambda} * \left[\overline{x^{\alpha}} + W_{ij}^{\alpha} + \tau_{ij}\right]$$

and the average waiting time for the low priority class β is

i)

$$W^{\beta} = x_0 + \sum_{n=1}^{N_Q^{\alpha}} x^{\alpha,n} + \sum_{n=1}^{N_Q^{\beta}} x^{\beta,n} + r^{\alpha} * V^{\alpha} * W^{\beta} * \overline{x^{\alpha}} \Rightarrow$$

$$W^{\beta} = \frac{x_0 + \sum_{n=1}^{N_Q^{\alpha}} x^{\alpha,n} + \sum_{n=1}^{N_Q^{\beta}} x^{\beta,n}}{1 - r^{\alpha} * V^{\alpha} * \overline{x^{\alpha}}}$$

ii)

$$W^{\beta} = x_0 + N_Q^{\alpha} * \overline{x^{\alpha}} + N_Q^{\beta} * \overline{x^{\beta}} + r^{\alpha} * V^{\alpha} * W^{\beta} * \overline{x^{\alpha}} \Rightarrow$$

$$W^{\beta} = \frac{x_0 + N_Q^{\alpha} * \overline{x^{\alpha}} + N_Q^{\beta} * \overline{x^{\beta}}}{1 - r^{\alpha} * V^{\alpha} * \overline{x^{\alpha}}}$$

iii)

$$W^{\beta} = x_0 + r^{\alpha} * V^{\alpha} * W^{\alpha} * \overline{x^{\alpha}} + r^{\beta} * V^{\beta} * W^{\beta} * \overline{x^{\beta}} + r^{\alpha} * V^{\alpha} * W^{\beta} * \overline{x^{\alpha}} \Rightarrow$$

$$W^\beta = \frac{x_0 + r^\alpha * V^\alpha * W^\alpha * \overline{x^\alpha}}{1 - r^\beta * V^\beta * \overline{x^\beta} - r^\alpha * V^\alpha * \overline{x^\alpha}}$$

iv)

$$W^\beta = x_0 + r^\alpha * \frac{\gamma^\alpha}{\delta^\alpha} * W^\alpha * \overline{x^\alpha} + r^\beta * \frac{\gamma^\beta}{\delta^\beta} * W^\beta * \overline{x^\beta} + r^\alpha * \frac{\gamma^\alpha}{\delta^\alpha} * W^\beta * \overline{x^\alpha} \Rightarrow$$

$$W^\beta = \frac{x_0 + r^\alpha * \frac{\gamma^\alpha}{\delta^\alpha} * W^\alpha * \overline{x^\alpha}}{1 - r^\beta * \frac{\gamma^\beta}{\delta^\beta} * \overline{x^\beta} - r^\alpha * \frac{\gamma^\alpha}{\delta^\alpha} * \overline{x^\alpha}}$$

Then we may define as cost function for class β at resource ij the weighted average packet delay

$$J_{ij}^\beta = \frac{\lambda_{ij}^\beta}{\lambda} * \left[\overline{x^\beta} + W_{ij}^\beta + \tau_{ij}\right]$$

Similar arguments can be used for multiple non-preemptive priorities.

Let class α has preemptive priority over class β. By a similar development as in the previous section, we have the average packet delay for the high priority class α

i)

$$U^\alpha = x_0^\alpha + \sum_{n=1}^{N_Q^\alpha} x^{\alpha,n}$$
$$T^\alpha = x^\alpha + U^\alpha$$

ii)

$$U^\alpha = x_0^\alpha + N_Q^\alpha * \overline{x^\alpha}$$
$$T^\alpha = \overline{x^\alpha} + U^\alpha$$

iii)

$$U^\alpha = x_0^\alpha + r^\alpha * V^\alpha * U^\alpha * \overline{x^\alpha} \Rightarrow U^\alpha = \frac{x_0^\alpha}{1 - r^\alpha * V^\alpha * \overline{x^\alpha}}$$
$$T^\alpha = \overline{x^\alpha} + U^\alpha$$

iv)

$$U^\alpha = \frac{x_0^\alpha}{1 - r^\alpha * \frac{\gamma^\alpha}{\delta^\alpha} * \overline{x^\alpha}}$$

$$T^{\alpha} = \overline{x^{\alpha}} + U^{\alpha}$$

where $x_0^{\alpha} = \lambda^{\alpha} * \overline{(x^{\alpha})^2}/2$ is the residual service time for class α.

Then we may define as cost function for class α at resource ij the weighted average packet delay

$$J_{ij}^{\alpha} = \frac{\lambda_{ij}^{\alpha}}{\lambda} * \left[T_{ij}^{\alpha} + \tau_{ij}\right]$$

Similarly, we have the average packet delay for the low priority class β

i)

$$U^{\beta} = x_0^{\beta} + \sum_{n=1}^{N_Q^{\alpha}} x^{\alpha,n} + \sum_{n=1}^{N_Q^{\beta}} x^{\beta,n}$$

$$T^{\beta} = x^{\beta} + U^{\beta} + r^{\alpha} * V^{\alpha} * T^{\beta} * \overline{x^{\alpha}} \Rightarrow T^{\beta} = \frac{x^{\beta} + U^{\beta}}{1 - r^{\alpha} * V^{\alpha} * \overline{x^{\alpha}}}$$

ii)

$$U^{\beta} = x_0^{\beta} + N_Q^{\alpha} * \overline{x^{\alpha}} + N_Q^{\beta} * \overline{x^{\beta}}$$

$$T^{\beta} = \frac{\overline{x^{\beta}} + U^{\beta}}{1 - r^{\alpha} * V^{\alpha} * \overline{x^{\alpha}}}$$

iii)

$$U^{\beta} = x_0^{\beta} + r^{\alpha} * V^{\alpha} * U^{\beta} * \overline{x^{\alpha}} + r^{\beta} * V^{\beta} * U^{\beta} * \overline{x^{\beta}} \Rightarrow$$

$$U^{\beta} = \frac{x_0^{\beta}}{1 - r^{\alpha} * V^{\alpha} * \overline{x^{\alpha}} - r^{\beta} * V^{\beta} * \overline{x^{\beta}}}$$

$$T^{\beta} = \frac{\overline{x^{\beta}} + U^{\beta}}{1 - r^{\alpha} * V^{\alpha} * \overline{x^{\alpha}}}$$

iv)

$$U^{\beta} = \frac{x_0^{\beta}}{1 - r^{\alpha} * \frac{\gamma^{\alpha}}{\delta^{\alpha}} * \overline{x^{\alpha}} - r^{\beta} * \frac{\gamma^{\beta}}{\delta^{\beta}} * \overline{x^{\beta}}}$$

$$T^{\beta} = \frac{\overline{x^{\beta}} + U^{\beta}}{1 - r^{\alpha} * \frac{\gamma^{\alpha}}{\alpha} * \overline{x^{\alpha}}}$$

where $x_0^{\beta} = (\lambda^{\alpha} * \overline{(x^{\alpha})^2} + \lambda^{\beta} * \overline{(x^{\beta})^2})/2$ is the residual service time for class β.

Similar arguments can be used for multiple preemptive priorities.

Then we may define as cost function for class β at resource ij the weighted average packet delay

$$J_{ij}^{\beta} = \frac{\lambda_{ij}^{\beta}}{\lambda} * \left[T_{ij}^{\beta} + \tau_{ij} \right]$$

4.8 Application to Integrated Services Networks

In this section, we apply the methodologies developed in the previous sections to integrated services networks.

4.8.1 Cost Functions for Multiple Classes

In this section, we introduce several cost functions for multiple class integrated services networks. Consider a system resource (node, link, computing site, etc.) ij with m_{ij} servers, each at service rate C_{ij}. Let also τ_{ij} be a flow independent constant delay (e.g. propagation delay) at resource ij. Packets from class c arrive at rate λ_{ij}^c (Poisson) and require service (general distribution) with mean $1/\mu$. We may take as cost function for class c at resource ij the weighted Erlang's C formula (probability of queueing)

$$J_{ij}^c = \frac{\lambda_{ij}^c}{\lambda} * P_{Q,ij}$$

where

$$P_{Q,ij} = \frac{\frac{\left(\frac{\sum_k \lambda_{ij}^k}{\mu C_{ij}}\right)^{m_{ij}}}{m_{ij}!} * \frac{m_{ij} * \mu C_{ij}}{m_{ij} * \mu C_{ij} - \sum_k \lambda_{ij}^k}}{\sum_{n=0}^{m-1} \frac{\left(\frac{\sum_k \lambda_{ij}^k}{\mu C_{ij}}\right)^n}{n!} + \frac{\left(\frac{\sum_k \lambda_{ij}^k}{\mu C_{ij}}\right)^{m_{ij}}}{m_{ij}!} * \frac{m_{ij} * \mu C_{ij}}{m_{ij} * \mu C_{ij} - \sum_k \lambda_{ij}^k}}$$

Consider a system resource (node, link, computing site, etc.) ij with m_{ij} servers, each at service rate C_{ij} and no buffers. Let also τ_{ij} be a flow independent constant delay (e.g. propagation delay) at resource ij. Packets from class c arrive at rate λ_{ij}^c (Poisson) and require service (general distribution) with mean $1/\mu$. We may take as cost function for class c at resource ij the weighted Erlang's B formula (or Erlang's loss formula)

$$J_{ij}^c = \frac{\lambda_{ij}^c}{\lambda} * B_{ij}(m_{ij})$$

and as cost function for class c along path π

$$J_\pi^c = \frac{\lambda_\pi^c}{\lambda} * \prod_{ij \in \pi} B_{ij}(m_{ij})$$

where

$$B_{ij}(m_{ij}) = \frac{\frac{\left(\frac{\sum_k \lambda_{ij}^k}{\mu C_{ij}}\right)^{m_{ij}}}{m_{ij}!}}{\sum_{n=1}^{m_{ij}} \frac{\left(\frac{\sum_k \lambda_{ij}^k}{\mu C_{ij}}\right)^n}{n!}}$$

Considering other queueing models, we can also define other cost functions (see next section).

4.9 Example

In this section, we consider the multi-objective routing problem in multiple class Integrated Services Networks. Packets from the first class can be queued, while packets from the other class are blocked. Therefore, the first class wants to minimize its average packet delay, while the other class wants to minimize its blocking probability.

We formulate the problem as a Nash game, where each class tries to minimize its own cost function in competition with the other class. We derive the routing policy for a two server parallel system and show the strategy and performance of each class (Figure 4.15).

4.9.1 Introduction

Traditionally, there were separate networks (circuit/packet switched) for carrying different traffic types (voice/data). In design and control problems of such networks, the goal is optimization of a single performance objective for the traffic type carried by the network under consideration. For example, in circuit switched networks, a voice call should be guaranteed an acceptable delay or else it should

be blocked. The objective is then to minimize the voice blocking probability. In packet switched networks. a data packet may be queued in buffers on intermediate nodes, thus the objective is to minimize the packet delay.

Today's trend is toward a single high speed packet switched network, called Broadband Integrated Services Digital Network (B-ISDN), that will support multimedia traffic (voice, data, video, etc.) simultaneously. These multiple classes of traffic will share the same network resources (buffers, switches, transmission lines, etc.) for flexible and efficient resource sharing. However, each class has different and conflicting performance requirements and objectives to those of other classes. Hence new methodologies are needed for network design and control problems.

In this section, we present an approach for the multiple class routing problem based on game-theory and we explicitly solve the routing problem for two classes of packets that share two links

One class of packets may be queued at the link buffers, while the second class of packets are blocked when there is not enough space. The objective for the first class is to minimize the delay for its packets, while the objective for the second class is to minimize its blocking probability. An application of this problem is to data/voice packet switched networks, where data packets may be queued at the links, while voice packets are dropped when they estimate that they will experience unacceptable delay. Another application is to ATM networks, where regular packets may be queued at the link buffers, while marked packets are dropped when there is congestion. Another application is for networks shared by different vendors, where the first vendor has unlimited access to the links, while the second vendor may use the links only if the congestion level is below a threshold.

4.9.2 Multi-server Queues with Blocking

We consider a queueing system of m servers that are shared by packets from two classes α and β. Class α packets arrive at rate λ^{α} (Poisson) and if all servers are busy, then they queue in the single queue of the system. Class β packets arrive at rate λ^{β} (Poisson) and if there are more than K packets (in queue and in service),

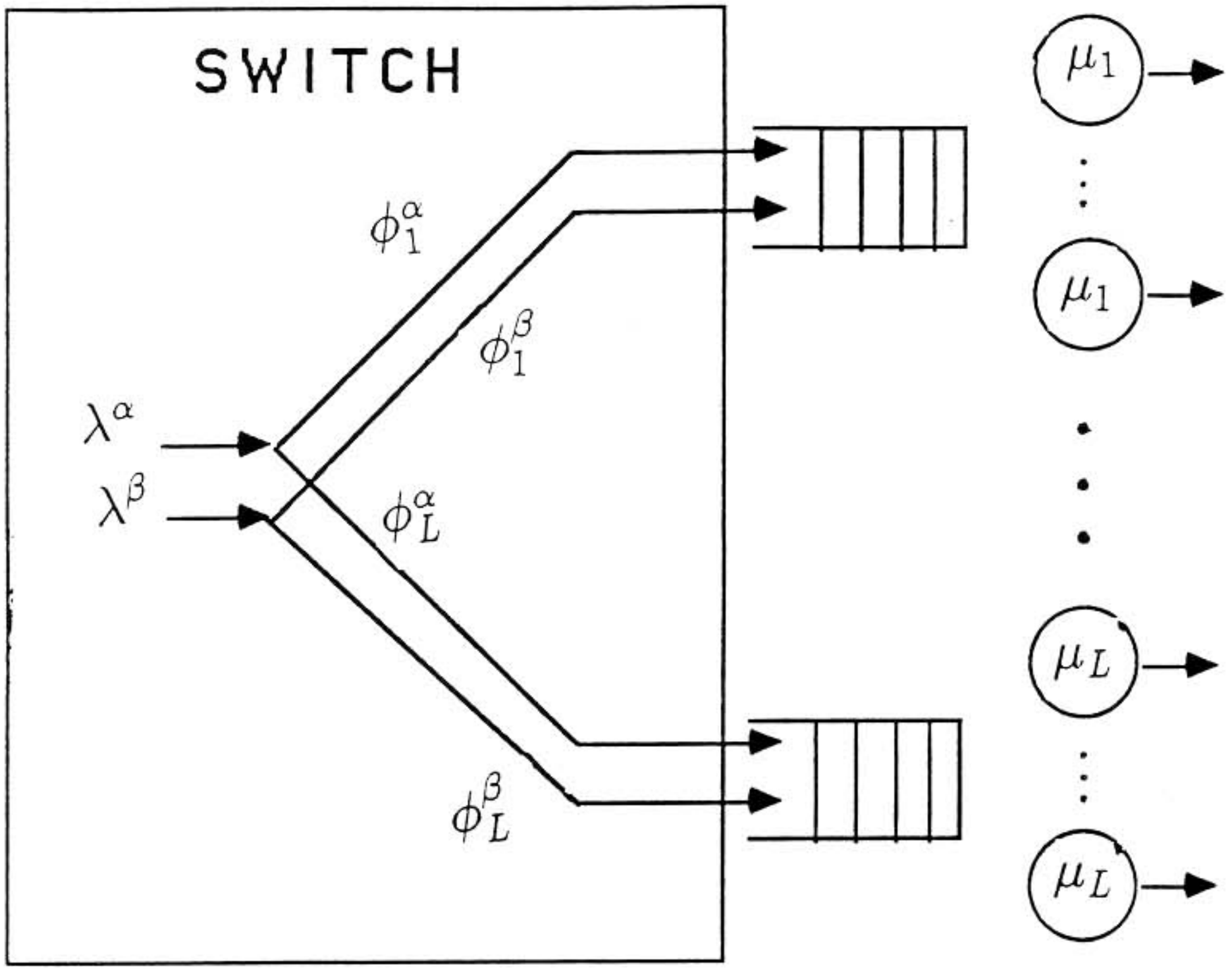


Figure 4.15: Routing in a switch.

then they are rejected. Without loss of generality, let the service requirement of each packet be exponentially distributed with mean $1/\mu$ and the rate of each server be 1. A packet receives service from a server chosen randomly among the free servers. Furthermore, for stability reasons it is assumed that the total arrival rate is less than the total service rate: $\lambda^{\alpha} + \lambda^{\beta} * P[n < K] \leq m * \mu$.

So, class α packets may be queued, while class β packets are blocked. Therefore, a reasonable objective is for class α to minimize its average packet delay, and for class β to minimize its blocking probability.

For the above queueing system, we consider two cases:

i) $K \geq m$, i.e. the blocking threshold for class β packets is greater than or equal to the number of servers.

ii) $K \leq m$, i.e. the blocking threshold for class β packets is less than or equal to the number of servers.

i) $\mathbf{K \leq m}$

Let first consider the case where the blocking threshold K for class β packets is greater or equal to the number of servers m (Fig. 4.16). If upon arrival a class β packet finds all m servers busy it may be queued if there are less than $K - m$ packets in the queue, otherwise it is lost. In this case, the threshold K is selected such that the maximum (expected) delay of a class β packet is less than an acceptable threshold T^{β}_{max}. The maximum delay of a class β packet is when upon arrival it finds $K - 1$ packets. It waits until all $K - m - 1$ in front of it in the queue plus 1 packet in service are served at rate $m\mu$ and then it enters service. So, its waiting plus service time is $\frac{K-m}{m\mu} + \frac{1}{\mu} \leq T^{\beta}_{max} \Rightarrow K \leq m\mu * T^{\beta}_{max}$.

The steady state probability that there are n packets in the system is

$$\pi_n = \begin{cases} \left(\frac{\lambda^{\alpha}+\lambda^{\beta}}{\mu}\right)^n * \frac{1}{n!} * \pi_0 & n \leq m \\ \left(\frac{\lambda^{\alpha}+\lambda^{\beta}}{m\mu}\right)^{n-m} * \left(\frac{\lambda^{\alpha}+\lambda^{\beta}}{\mu}\right)^m * \frac{1}{m!} * \pi_0 & m \leq n \leq K \\ \left(\frac{\lambda^{\alpha}}{m\mu}\right)^{n-K} * \left(\frac{\lambda^{\alpha}+\lambda^{\beta}}{m\mu}\right)^{K-m} * \left(\frac{\lambda^{\alpha}+\lambda^{\beta}}{\mu}\right)^m * \frac{1}{m!} * \pi_0 & K \leq n \end{cases}$$

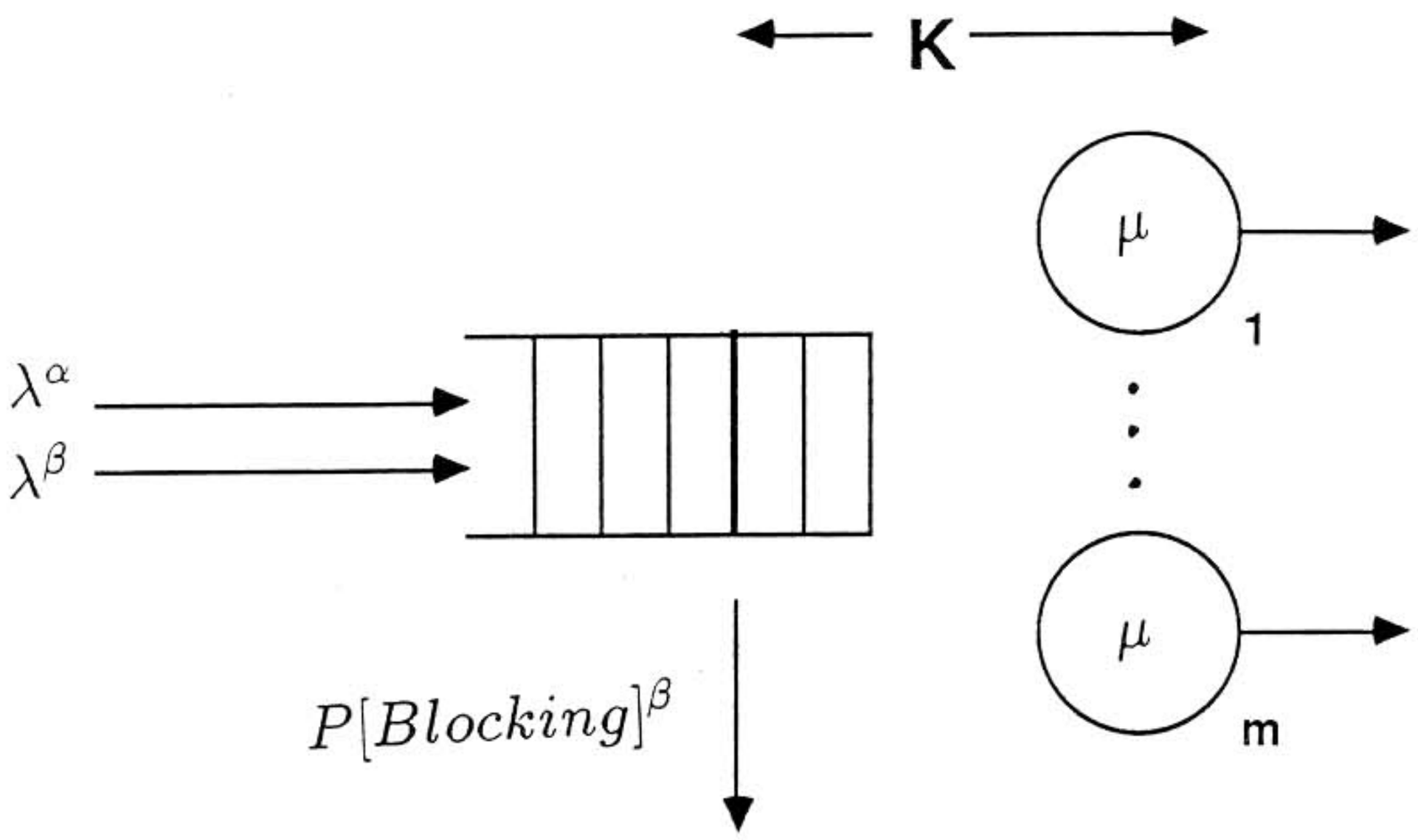


Figure 4.16: A multi-server queue, where class α packets may be queued, while class β packets are dropped when the number of packets in the system is greater than or equal to K, where $K \geq m$.

Then the probability that the system is idle is given by:

$$\pi_0 = \left[\sum_{n=0}^{m-1} \left(\frac{\lambda^\alpha + \lambda^\beta}{\mu} \right)^n * \frac{1}{n!} + \right.$$

$$\left. + \left(\frac{\lambda^\alpha + \lambda^\beta}{\mu} \right)^m * \frac{1}{m!} * \frac{1 - \frac{\lambda^\alpha}{m\mu} - \frac{\lambda^\beta}{m\mu} * \left(\frac{\lambda^\alpha + \lambda^\beta}{m\mu} \right)^{K-m}}{(1 - \frac{\lambda^\alpha}{m\mu}) * (1 - \frac{\lambda^\alpha + \lambda^\beta}{m\mu})} \right]^{-1}$$

The probability that a class β packet is lost is the probability that there are at least K packets in the system:

$$P[n \geq K] = \sum_{n=K}^{\infty} \pi_n = \left(\frac{\lambda^\alpha + \lambda^\beta}{m\mu} \right)^{K-m} * \left(\frac{\lambda^\alpha + \lambda^\beta}{\mu} \right)^m * \frac{1}{m!} * \frac{\pi_0}{1 - \frac{\lambda^\alpha}{m\mu}}$$

The average number of packets in the system is:

$$\bar{N} = \sum_{n=0}^{\infty} n * \pi_n = \pi_0 * \left\{ \sum_{n=1}^{m-1} \left(\frac{\lambda^\alpha + \lambda^\beta}{\mu} \right)^n * \frac{1}{(n-1)!} + \right.$$

$$+ \left(\frac{\lambda^\alpha + \lambda^\beta}{\mu} \right)^m * \frac{1}{m!} * \left[m * \frac{1 - \left(\frac{\lambda^\alpha + \lambda^\beta}{m\mu} \right)^{K-m+1}}{1 - \frac{\lambda^\alpha + \lambda^\beta}{m\mu}} + \right.$$

$$\left. + \frac{\lambda^\alpha + \lambda^\beta}{m\mu} * \frac{1 - (K - m + 1) * \left(\frac{\lambda^\alpha + \lambda^\beta}{m\mu} \right)^{K-m} + (K - m) * \left(\frac{\lambda^\alpha + \lambda^\beta}{m\mu} \right)^{K-m+1}}{\left(1 - \frac{\lambda^\alpha + \lambda^\beta}{m\mu} \right)^2} \right] +$$

$$\left. + \left(\frac{\lambda^\alpha + \lambda^\beta}{m\mu} \right)^{K-m} * \left(\frac{\lambda^\alpha + \lambda^\beta}{\mu} \right)^m * \frac{1}{m!} * \left[K * \frac{\frac{\lambda^\alpha}{m\mu}}{1 - \frac{\lambda^\alpha}{m\mu}} + \frac{\frac{\lambda^\alpha}{m\mu}}{\left(1 - \frac{\lambda^\alpha}{m\mu} \right)^2} \right] \right\}$$

The overall average packet delay is:

$$\bar{T} = \frac{\bar{N}}{\lambda^{\alpha} + \lambda^{\beta} * (1 - P[n \geq K])}$$

The average packet delay for class α is:

$$\bar{T}^{\alpha} = \sum_{n=0}^{m-1} \frac{1}{\mu} * \pi_n + \sum_{n=m}^{\infty} \left(\frac{n-m+1}{m\mu} + \frac{1}{\mu} \right) * \pi_n \Rightarrow$$

$$\bar{T}^{\alpha} = \frac{\pi_0}{\mu} * \left\{ \sum_{n=0}^{m-1} \left(\frac{\lambda^{\alpha} + \lambda^{\beta}}{\mu} \right)^n * \frac{1}{n!} + \right.$$

$$+ \left(\frac{\lambda^{\alpha} + \lambda^{\beta}}{\mu} \right)^m * \frac{1}{m!} * \left[\frac{1 - \left(\frac{\lambda^{\alpha} + \lambda^{\beta}}{m\mu} \right)^{K-m}}{1 - \frac{\lambda^{\alpha} + \lambda^{\beta}}{m\mu}} + \right.$$

$$\left. + \frac{1}{m} * \frac{1 - (K - m + 1) * \left(\frac{\lambda^{\alpha} + \lambda^{\beta}}{m\mu} \right)^{K-m} + (K - m) * \left(\frac{\lambda^{\alpha} + \lambda^{\beta}}{m\mu} \right)^{K-m+1}}{\left(1 - \frac{\lambda^{\alpha} + \lambda^{\beta}}{m\mu} \right)^2} \right] +$$

$$\left. + \left(\frac{\lambda^{\alpha} + \lambda^{\beta}}{m\mu} \right)^{K-m} * \left(\frac{\lambda^{\alpha} + \lambda^{\beta}}{\mu} \right)^m * \frac{1}{m!} * \left[\frac{\frac{K}{m}}{1 - \frac{\lambda^{\alpha}}{m\mu}} + \frac{\frac{1}{m}}{\left(1 - \frac{\lambda^{\alpha}}{m\mu} \right)^2} \right] \right\}$$

Next, we also present the above performance measures for the special cases $K = m$ and $m = 1$.

Special Cases:

i) $K = m$

In this case class β packets can not be queued. So, if upon arrival a class β packet finds all m servers busy it is lost. Then the previously defined performance measures become:

$$\pi_0 = \left[\sum_{n=0}^{m-1} \left(\frac{\lambda^\alpha + \lambda^\beta}{\mu} \right)^n * \frac{1}{n!} + \left(\frac{\lambda^\alpha + \lambda^\beta}{\mu} \right)^m * \frac{1}{m!} * \frac{1}{1 - \frac{\lambda^\alpha}{m\mu}} \right]^{-1}$$

$$P[n \geq m] = \left(\frac{\lambda^\alpha + \lambda^\beta}{\mu} \right)^m * \frac{1}{m!} * \frac{\pi_0}{1 - \frac{\lambda^\alpha}{m\mu}}$$

$$\bar{N} = \pi_0 * \left\{ \sum_{n=1}^{m-1} \left(\frac{\lambda^\alpha + \lambda^\beta}{\mu} \right)^n * \frac{1}{(n-1)!} + \right.$$

$$\left. + \left(\frac{\lambda^\alpha + \lambda^\beta}{\mu} \right)^m * \frac{1}{m!} * \left[\frac{m}{1 - \frac{\lambda^\alpha}{m\mu}} + \frac{\frac{\lambda^\alpha}{m\mu}}{\left(1 - \frac{\lambda^\alpha}{m\mu} \right)^2} \right] \right\}$$

$$\bar{T} = \frac{\bar{N}}{\lambda^\alpha + \lambda^\beta * (1 - P[n \geq m])}$$

$$\bar{T}^\alpha = \frac{\pi_0}{\mu} * \left\{ \sum_{n=0}^{m-1} \left(\frac{\lambda^\alpha + \lambda^\beta}{\mu} \right)^n * \frac{1}{n!} + \left(\frac{\lambda^\alpha + \lambda^\beta}{\mu} \right)^m * \frac{1}{m!} * \left[\frac{1}{1 - \frac{\lambda^\alpha}{m\mu}} + \frac{\frac{1}{m}}{\left(1 - \frac{\lambda^\alpha}{m\mu} \right)^2} \right] \right\}$$

ii) $m = 1$

In this case, there is only one server. Then the previously defined performance measures become:

$$\pi_0 = \frac{1 - \frac{\lambda^\alpha}{\mu} - \frac{\lambda^\beta}{\mu} * \left(\frac{\lambda^\alpha + \lambda^\beta}{\mu} \right)^K}{\left(1 - \frac{\lambda^\alpha}{\mu} \right) * \left(1 - \frac{\lambda^\alpha + \lambda^\beta}{\mu} \right)}$$

$$P[n \geq K] = \frac{\left(\frac{\lambda^{\alpha} + \lambda^{\beta}}{\mu}\right)^{K} * \left(1 - \frac{\lambda^{\alpha} + \lambda^{\beta}}{\mu}\right)}{1 - \frac{\lambda^{\alpha}}{\mu} - \frac{\lambda^{\beta}}{\mu} * \left(\frac{\lambda^{\alpha} + \lambda^{\beta}}{\mu}\right)^{K}}$$

$$\bar{N} = \pi_0 * \left\{ \frac{\lambda^{\alpha} + \lambda^{\beta}}{\mu} * \frac{1 - (K+1) * \left(\frac{\lambda^{\alpha} + \lambda^{\beta}}{\mu}\right)^{K} + K * \left(\frac{\lambda^{\alpha} + \lambda^{\beta}}{\mu}\right)^{K+1}}{\left(1 - \frac{\lambda^{\alpha} + \lambda^{\beta}}{\mu}\right)^{2}} + \right.$$

$$\left. + \left(\frac{\lambda^{\alpha} + \lambda^{\beta}}{\mu}\right)^{K} * \left[K * \frac{\frac{\lambda^{\alpha}}{\mu}}{1 - \frac{\lambda^{\alpha}}{\mu}} + \frac{\frac{\lambda^{\alpha}}{\mu}}{\left(1 - \frac{\lambda^{\alpha}}{\mu}\right)^{2}} \right] \right\}$$

$$\bar{T} = \frac{\bar{N}}{\lambda^{\alpha} + \lambda^{\beta} * (1 - P[n \geq K])}$$

$$\bar{T}^{\alpha} = \frac{\pi_0}{\mu} * \left\{ 1 + \frac{\lambda^{\alpha} + \lambda^{\beta}}{\mu} * \left[\frac{1 - \left(\frac{\lambda^{\alpha} + \lambda^{\beta}}{\mu}\right)^{K-1}}{1 - \frac{\lambda^{\alpha} + \lambda^{\beta}}{\mu}} + \right.\right.$$

$$\left.\left. + \frac{1 - K * \left(\frac{\lambda^{\alpha} + \lambda^{\beta}}{\mu}\right)^{K-1} + (K-1) * \left(\frac{\lambda^{\alpha} + \lambda^{\beta}}{\mu}\right)^{K}}{\left(1 - \frac{\lambda^{\alpha} + \lambda^{\beta}}{\mu}\right)^{2}} \right] + \right.$$

$$\left. + \left(\frac{\lambda^{\alpha} + \lambda^{\beta}}{\mu}\right)^{K} * \left[\frac{K}{1 - \frac{\lambda^{\alpha}}{\mu}} + \frac{1}{\left(1 - \frac{\lambda^{\alpha}}{\mu}\right)^{2}} \right] \right\}$$

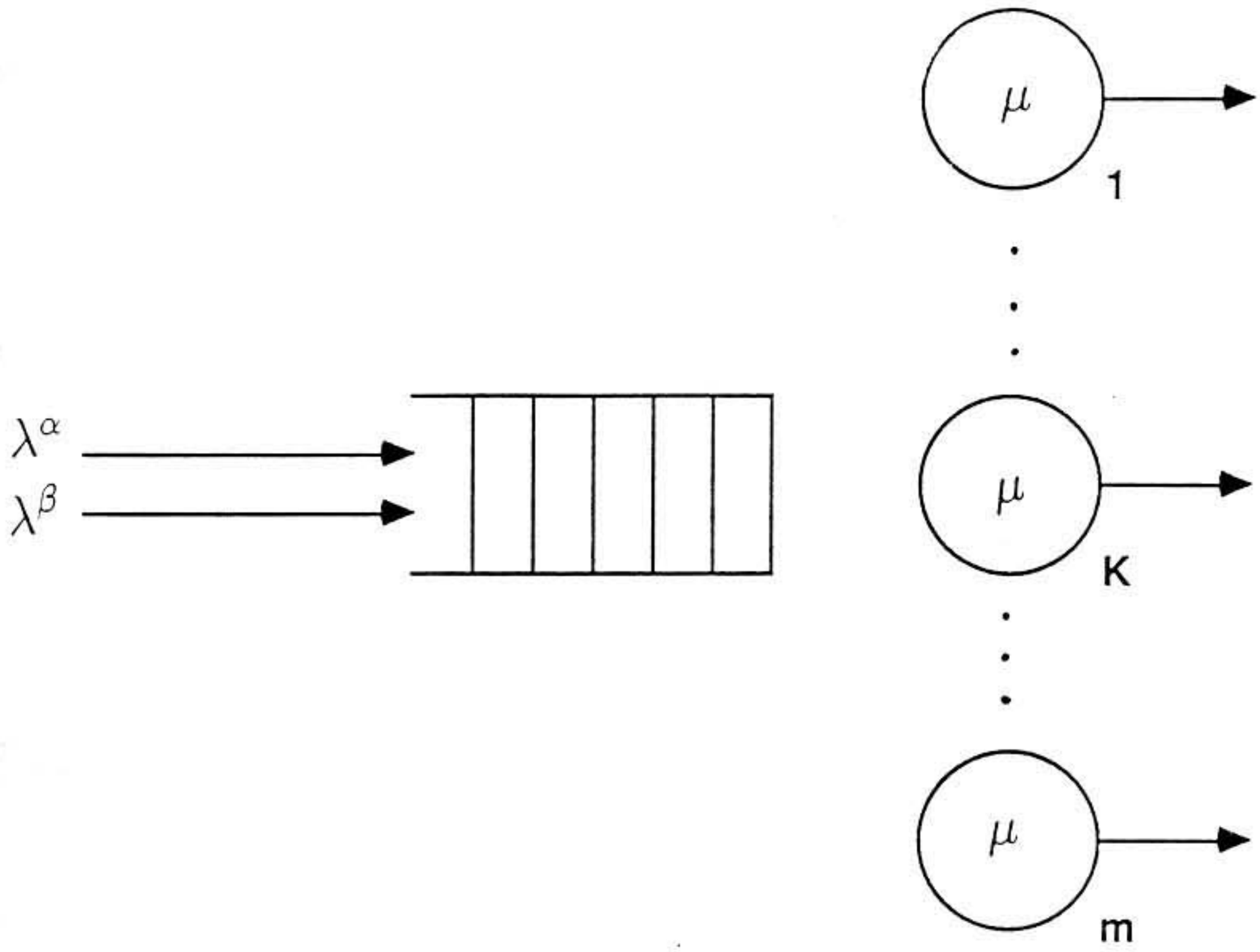


Figure 4.17: A multi-server queue, where class α packets may be queued, while class β packets are dropped when the number of packets in the system is greater than or equal to K, where $K \leq m$.

ii) $\mathbf{K \leq m}$

Let now consider the case where the blocking threshold K for class β packet is less or equal to the number of servers m (Fig. 4.17). If upon arrival a class β packet finds less than K servers busy, then it is starts been serviced, otherwise it is lost. In this case, the threshold K can be selected such that the maximum blocking of a class β packet is less than an acceptable threshold B^{β}_{max}, i.e. $P[n \geq K] \leq B^{\beta}_{max}$. Also, the service rate of each server must be large enough to guarantee acceptable packet delay $\frac{1}{\mu} \leq T^{\beta}_{max}$.

Then the steady state probability that there are n packets in the system is

$$\pi_n = \begin{cases} \left(\frac{\lambda^{\alpha}+\lambda^{\beta}}{\mu}\right)^n * \frac{1}{n!} * \pi_0 & n \leq K \\ \left(\frac{\lambda^{\alpha}}{\mu}\right)^{n-K} * \left(\frac{\lambda^{\alpha}+\lambda^{\beta}}{\mu}\right)^K * \frac{1}{n!} * \pi_0 & K \leq n \leq m \\ \left(\frac{\lambda^{\alpha}}{m\mu}\right)^{n-m} * \left(\frac{\lambda^{\alpha}}{\mu}\right)^{m-K} * \left(\frac{\lambda^{\alpha}+\lambda^{\beta}}{\mu}\right)^K * \frac{1}{m!} * \pi_0 & m \leq n \end{cases}$$

Then the probability that the system is idle is given by

$$\pi_0 = \left[\sum_{n=0}^{K}\left(\frac{\lambda^{\alpha}+\lambda^{\beta}}{\mu}\right)^n * \frac{1}{n!} + \left(\frac{\lambda^{\alpha}+\lambda^{\beta}}{\mu}\right)^K * \sum_{n=1}^{m-K}\left(\frac{\lambda^{\alpha}}{\mu}\right)^n * \frac{1}{(K+n)!} + \right.$$

$$\left. + \left(\frac{\lambda^{\alpha}}{\mu}\right)^{m-K} * \left(\frac{\lambda^{\alpha}+\lambda^{\beta}}{\mu}\right)^K * \frac{1}{m!} * \frac{\frac{\lambda^{\alpha}}{m\mu}}{1-\frac{\lambda^{\alpha}}{m\mu}}\right]^{-1}$$

The probability that class β packets are lost is given by

$$P[n \geq K] = \sum_{n=K}^{\infty} \pi_n = \pi_0 * \left\{ \left(\frac{\lambda^\alpha + \lambda^\beta}{\mu} \right)^K * \sum_{n=0}^{m-K} \left(\frac{\lambda^\alpha}{\mu} \right)^n * \frac{1}{(K+n)!} + \right.$$

$$\left. + \left(\frac{\lambda^\alpha}{\mu} \right)^{m-K} * \left(\frac{\lambda^\alpha + \lambda^\beta}{\mu} \right)^K * \frac{1}{m!} * \frac{\frac{\lambda^\alpha}{m\mu}}{1 - \frac{\lambda^\alpha}{m\mu}} \right\}$$

The average number of packets in the system is

$$\bar{N} = \sum_{n=0}^{\infty} n * \pi_n = \pi_0 * \left\{ \sum_{n=1}^{K} \left(\frac{\lambda^\alpha + \lambda^\beta}{\mu} \right)^n * \frac{1}{(n-1)!} + \right.$$

$$+ \left(\frac{\lambda^\alpha + \lambda^\beta}{\mu} \right)^K * \sum_{n=1}^{m-K} \left(\frac{\lambda^\alpha}{\mu} \right)^n * \frac{1}{(K+n-1)!} +$$

$$\left. + \left(\frac{\lambda^\alpha}{\mu} \right)^{m-K} * \left(\frac{\lambda^\alpha + \lambda^\beta}{\mu} \right)^K * \frac{1}{m!} * \left[m * \frac{\frac{\lambda^\alpha}{m\mu}}{1 - \frac{\lambda^\alpha}{m\mu}} + \frac{\frac{\lambda^\alpha}{m\mu}}{\left(1 - \frac{\lambda^\alpha}{m\mu}\right)^2} \right] \right\}$$

The overall average packet delay is

$$\bar{T} = \frac{\bar{N}}{\lambda^\alpha + \lambda^\beta * (1 - P[n \geq K])}$$

and the average class α packet delay is

$$\bar{T}^{\alpha} = \sum_{n=0}^{m-1} \frac{1}{\mu} * \pi_n + \sum_{n=m}^{\infty} \left(\frac{n-m+1}{m\mu} + \frac{1}{\mu} \right) * \pi_n =$$

$$= \frac{\pi_0}{\mu} * \left\{ \sum_{n=0}^{K-1} \left(\frac{\lambda^{\alpha} + \lambda^{\beta}}{\mu} \right)^n * \frac{1}{n!} + \right.$$

$$+ \left(\frac{\lambda^{\alpha} + \lambda^{\beta}}{\mu} \right)^K * \sum_{n=0}^{m-K-1} \left(\frac{\lambda^{\alpha}}{\mu} \right)^n * \frac{1}{(K+n)!} +$$

$$\left. + \left(\frac{\lambda^{\alpha}}{\mu} \right)^{m-K} * \left(\frac{\lambda^{\alpha} + \lambda^{\beta}}{\mu} \right)^K * \frac{1}{m!} * \left[\frac{1}{1 - \frac{\lambda^{\alpha}}{m\mu}} + \frac{\frac{1}{m}}{\left(1 - \frac{\lambda^{\alpha}}{m\mu} \right)^2} \right] \right\}$$

Next,, we also present the above performance measures for the special cases $K = m$ and $K = 1$.

Special Cases:

i) $K = m$ same as previously.

ii) $K = 1$

In this case, class β packets may use only one server. A class β packet arrival that finds another class β packet in service is lost. Then the previously defined performance measures become:

$$\pi_0 = \left[1 + \frac{\lambda^{\alpha} + \lambda^{\beta}}{\mu} * \sum_{n=0}^{m-1} \left(\frac{\lambda^{\alpha}}{\mu} \right)^n * \frac{1}{(n+1)!} + \left(\frac{\lambda^{\alpha}}{\mu} \right)^{m-1} * \frac{\lambda^{\alpha} + \lambda^{\beta}}{\mu} * \frac{1}{m!} * \frac{\frac{\lambda^{\alpha}}{m\mu}}{1 - \frac{\lambda^{\alpha}}{m\mu}} \right]^{-1}$$

$$P[n \geq 1] = 1 - \pi_0$$

$$\bar{N} = \pi_0 * \left\{ \frac{\lambda^\alpha + \lambda^\beta}{\mu} * \sum_{n=0}^{m-1} \left(\frac{\lambda^\alpha}{\mu} \right)^n * \frac{1}{n!} + \right.$$

$$\left. + \left(\frac{\lambda^\alpha}{\mu} \right)^{m-1} * \frac{\lambda^\alpha + \lambda^\beta}{\mu} * \frac{1}{m!} * \left[m * \frac{\frac{\lambda^\alpha}{m\mu}}{1 - \frac{\lambda^\alpha}{m\mu}} + \frac{\frac{\lambda^\alpha}{m\mu}}{\left(1 - \frac{\lambda^\alpha}{m\mu}\right)^2} \right] \right\}$$

$$\bar{T} = \frac{\bar{N}}{\lambda^\alpha + \lambda^\beta * \pi_0}$$

$$\bar{T}^\alpha = \frac{\pi_0}{\mu} * \left\{ 1 + \frac{\lambda^\alpha + \lambda^\beta}{\mu} * \sum_{n=0}^{m-2} \left(\frac{\lambda^\alpha}{\mu} \right)^n * \frac{1}{(n+1)!} + \right.$$

$$\left. + \left(\frac{\lambda^\alpha}{\mu} \right)^{m-1} * \frac{\lambda^\alpha + \lambda^\beta}{\mu} * \frac{1}{m!} * \left[\frac{1}{1 - \frac{\lambda^\alpha}{m\mu}} + \frac{\frac{1}{m}}{\left(1 - \frac{\lambda^\alpha}{m\mu}\right)^2} \right] \right\}$$

4.9.3 A Game Theory Formulation

In this section, we consider a routing problem in a parallel system that is composed of L multi-server queueing systems, each one similar to the one analyzed in the previous section.

Let class c packets arrive to this parallel system with rate λ^c (Poisson arrivals) and may use any one of these L multi-server queueing systems in order to reach a destination node. The fraction of class c packets assigned to multi-server queueing system i is ϕ_i^c and the vector of these fractions for each class c is $\mathbf{\Phi}^c = [... \ \phi_i^c \ ...]$. There are m_i servers at the queueing system i, each one with rate μ_i and the blocking threshold is K_i.

In this section, for simplicity, we consider two classes of packets, i.e. $c \in \{\alpha, \beta\}$. Class α packets can be queued and therefore class α wants to minimize its average packet delay, while class β packets are blocked and therefore class β wants to minimize its blocking probability.

First, we formulate the problem as a *Pareto* game between the two classes, where both classes minimize a single common cost function which is a combination of their individual cost functions. The Pareto problem is

$$\textit{minimize} \qquad \left| \epsilon * |J^{\alpha}(\mathbf{\Phi}^{\alpha}, \mathbf{\Phi}^{\beta})|^{p} + (1-\epsilon) * |J^{\alpha}(\mathbf{\Phi}^{\alpha}, \mathbf{\Phi}^{\beta})|^{p} \right|^{1/p}$$

$$\textit{with respect to} \quad (\mathbf{\Phi}^{\alpha}, \mathbf{\Phi}^{\beta})$$

$$\textit{such that} \qquad 0 < \epsilon < 1, \quad p \geq 1$$

$$\sum_{i=1}^{L} \phi_i^{\alpha} = 1, \quad \phi_i^{\alpha} \geq 0 \ \ \forall \ i$$

$$\sum_{i=1}^{L} \phi_i^{\beta} = 1, \quad \phi_i^{\beta} \geq 0 \ \ \forall \ i$$

For $p \rightarrow \infty$, we have a *minimax* problem, where we minimize the maximum cost function. This leads to a conservative strategy, since we are not interested in directly minimizing the cost functions of both classes. The minimax problem is

$$\textit{minimize} \qquad max \left\{ J^{\alpha}(\mathbf{\Phi}^{\alpha}, \mathbf{\Phi}^{\beta}), J^{\beta}(\mathbf{\Phi}^{\alpha}, \mathbf{\Phi}^{\beta}) \right\}$$

$$\textit{with respect to} \quad (\mathbf{\Phi}^{\alpha}, \mathbf{\Phi}^{\beta})$$

$$\textit{such that} \qquad \sum_{i=1}^{L} \phi_i^{\alpha} = 1, \quad \phi_i^{\alpha} \geq 0 \ \ \forall \ i$$

$$\sum_{i=1}^{L} \phi_i^{\beta} = 1, \quad \phi_i^{\beta} \geq 0 \ \ \forall \ i$$

Finally, we formulate the problem as a *Nash* game between the two classes, where each class knows the cost functions and constraints for both classes. After reaching a Nash equilibrium, no class will have a rational motive to unilaterally deviate from its equilibrium strategy.

Class α solves the following problem:

$$\textit{minimize} \qquad J^{\alpha}(\mathbf{\Phi}^{\alpha}, \mathbf{\Phi}^{\beta *}) = \sum_{i=1}^{L} \phi_i^{\alpha} * \bar{T}_i^{\alpha}$$

$$\textit{with respect to} \quad \mathbf{\Phi}^{\alpha}$$

$$\textit{such that} \qquad \sum_{i=1}^{L} \phi_i^{\alpha} = 1, \quad \phi_i^{\alpha} \geq 0 \ \ \forall\, i$$

and class β solves the following problem:

$$\textit{minimize} \qquad J^{\beta}(\mathbf{\Phi}^{\alpha *}, \mathbf{\Phi}^{\beta}) = \sum_{i=1}^{L} \phi_i^{\beta} * P[n_i \geq K_i]$$

$$\textit{with respect to} \quad \mathbf{\Phi}^{\beta}$$

$$\textit{such that} \qquad \sum_{i=1}^{L} \phi_i^{\beta} = 1, \quad \phi_i^{\beta} \geq 0 \ \ \forall\, i$$

4.9.4 Numerical Results

Here, we consider a parallel system composed of two single-server queueing systems, and class β packets are blocked when the server is busy, i.e. $L = 2, \quad K = m = 1$.

Then the previously defined performance measures become:

$$\pi_0 = \frac{\mu - \lambda^{\alpha}}{\mu + \lambda^{\beta}}$$

$$P[Blocking]^{\beta} = P[n \geq 1] = \frac{\lambda^{\alpha} + \lambda^{\beta}}{\mu + \lambda^{\beta}}$$

$$\bar{N} = \frac{(\lambda^{\alpha} + \lambda^{\beta}) * \mu}{(\mu - \lambda^{\alpha}) * (\mu + \lambda^{\beta})}$$

$$\bar{T} = \frac{1}{\mu - \lambda^{\alpha}}$$

$$\bar{T}^{\alpha} = \frac{1}{\mu - \lambda^{\alpha}} + \frac{\lambda^{\beta}}{\mu * (\mu + \lambda^{\beta})}$$

If ϕ_i^c is the fraction of class c packets routed to server i, then we can write $\lambda_i^c = \lambda^c * \phi_i^c$, where $c \in \{\alpha, \beta\}$, $i \in \{1, 2\}$.

Formulating the problem as a Pareto game with $\mathbf{\Phi}^{\alpha} = \mathbf{\Phi}^{\beta} = \mathbf{\Phi}$, both classes solve the following problem:

minimize

$$J(\mathbf{\Phi}) = \sum_{i=1}^{2} \phi_i * \left\{ \epsilon * \left[\frac{}{\mu_i - \lambda^{\alpha} * \phi_i} + \frac{\lambda^{\beta} * \phi_i}{\mu_i * (\mu_i + \lambda^{\beta} * \phi_i)} \right] + (1 - \epsilon) * \frac{(\lambda^{\alpha} + \lambda^{\beta}) *_i}{\mu_i + \lambda^{\beta} * \phi_i} \right\}$$

with respect to $\phi_1,\ \phi_2$

such that $$\sum_{i=1}^{2} \phi_i = 1, \quad \phi_1,\ \phi_2 \geq 0$$

The first and second derivatives of J with respect to ϕ_1 are

$$\frac{\partial J}{\partial \phi_1} = \epsilon * \left[\frac{\mu_1}{(\mu_1 - \lambda^{\alpha} * \phi_1)^2} + \frac{2 * \lambda^{\beta} * \phi_1 * \mu_1 + (\lambda^{\beta} * \phi_1)^2}{\mu_1 * (\mu_1 + \lambda^{\beta} * \phi_1)} \right] +$$

$$+ \quad (1 - \epsilon) * (\lambda^{\alpha} + \lambda^{\beta}) * \frac{2 * \mu_1 * \phi_1 + \lambda^{\beta} * (\phi_1)^2}{(\mu_1 + \lambda^{\beta} * \phi_1)^2}$$

$$\frac{\partial^2 J}{\partial (\phi_1)^2} = \epsilon * \left[\frac{2 * \lambda^{\alpha} * \mu_1}{(\mu_1 - \lambda^{\alpha} * \phi_1)^3} + \frac{2 * \lambda^{\beta} * \mu_1}{(\mu_1 + \lambda^{\beta} * \phi_1)^3} \right] +$$

$$+ \quad (1 - \epsilon) * (\lambda^{\alpha} + \lambda^{\beta}) * \frac{2 * (\mu_1)^2}{(\mu_1 + \lambda^{\beta} * \phi_1)^3}$$

Formulating the problem as a Nash game, class α solves the following problem:

$$minimize \qquad J^{\alpha}(\mathbf{\Phi}^{\alpha}, \mathbf{\Phi}^{\beta *}) = \sum_{i=1}^{2} \phi_i^{\alpha} * \left[\frac{1}{\mu_i - \lambda^{\alpha} * \phi_i^{\alpha}} + \frac{\lambda^{\beta} * \phi_i^{\beta *}}{\mu_i * (\mu_i + \lambda^{\beta} * \phi_i^{\beta *})} \right]$$

$$with\ respect\ to \quad \phi_1^{\alpha},\ \phi_2^{\alpha}$$

$$such\ that \qquad \sum_{i=1}^{2} \phi_i^{\alpha} = 1, \quad \phi_1^{\alpha},\ \phi_2^{\alpha} \geq 0$$

The first and second order derivatives of J^{α} with respect to ϕ_1^{α} are

$$\frac{\partial J^{\alpha}}{\partial \phi_1^{\alpha}} = \frac{\mu_1}{(\mu_1 - \lambda^{\alpha} * \phi_1^{\alpha})^2} + \frac{\lambda^{\beta} * \phi_1^{\beta}}{\mu_1 * (\mu_1 + \lambda^{\beta} * \phi_1^{\beta})}$$

$$\frac{\partial^2 J^{\alpha}}{\partial (\phi_1^{\alpha})^2} = \frac{2 * \mu_1 * \lambda^{\alpha}}{(\mu_1 - \lambda^{\alpha} * \phi_1^{\alpha})^3}$$

and the cross derivative of J^{α} with respect to $\phi_1^{\beta}, \phi_1^{\alpha}$ is

$$\frac{\partial^2 J^{\alpha}}{\partial \phi_1^{\beta} \partial \phi_1^{\alpha}} = \frac{\lambda^{\beta}}{(\mu_1 + \lambda^{\beta} * \phi_1^{\beta})^2}$$

Similarly for the derivatives of J^{α} with respect to ϕ_2^{α} and ϕ_2^{β}.

Also, class β solves the following problem:

$$minimize \qquad J^{\beta}(\mathbf{\Phi}^{\alpha *}, \mathbf{\Phi}^{\beta}) = \sum_{i=1}^{2} \phi_i^{\beta} * \frac{(\lambda^{\alpha} * \phi_i^{\alpha *} + \lambda^{\beta} * \phi_i^{\beta})}{\mu_i + \lambda^{\beta} * \phi_i^{\beta}}$$

$$with\ respect\ to \quad \phi_1^{\beta},\ \phi_2^{\beta}$$

$$such\ that \qquad \sum_{i=1}^{2} \phi_i^{\beta} = 1, \quad \phi_1^{\beta},\ \phi_2^{\beta} \geq 0$$

The first and second order derivatives of J^{β} with respect to ϕ_1^{β} are

$$\frac{\partial J^{\beta}}{\partial \phi_1^{\beta}} = 1 - \frac{\mu_1 * (\mu_1 - \lambda^{\alpha} * \phi_1^{\alpha})}{(\mu_1 + \lambda^{\beta} * \phi_1^{\beta})^3}$$

$$\frac{\partial^2 J^\beta}{\partial(\phi_1^\beta)^2} = \frac{2 * \lambda^\beta * \mu_1 * (\mu_1 - \lambda^\alpha * \phi_1^\alpha)}{(\mu_1 + \lambda^\beta * \phi_1^\beta)^3}$$

and the cross derivative of J^β with respect to $\phi_1^\alpha, \phi_1^\beta$ is

$$\frac{\partial^2 J^\beta}{\partial\phi_1^\alpha \partial\phi_1^\beta} = \frac{\lambda^\alpha * \mu_1}{(\mu_1 + \lambda^\beta * \phi_1^\beta)^2}$$

Theorem: *existence & uniqueness*

Let packets from two competing classes α and β arrive according to Poisson distribution. They may be transmitted through multiple links with exponential service time distribution. Class α minimizes its average packet delay, while class β minimizes its blocking probability.

For the above routing game, there exists a unique Nash equilibrium solution.

Proof: The action spaces $\phi_1^\alpha + \phi_2^\alpha = 1,\ \phi_1^\alpha,\ \phi_2^\alpha \geq 0$ and $\phi_1^\beta + \phi_2^\beta = 1,\ \phi_1^\beta,\ \phi_2^\beta \geq 0$ define a convex, closed and compact set. The cost function J^α is jointly continuous in all its arguments and convex in $(\phi_1^\alpha, \phi_2^\alpha)$ for each fixed value of $(\phi_1^\beta, \phi_2^\beta)$. The cost function J^β is jointly continuous in all its arguments and convex in $(\phi_1^\beta, \phi_2^\beta)$ for each fixed value of $(\phi_1^\alpha, \phi_2^\alpha)$. The function $J^\alpha + J^\beta$ is continuous and convex in $(\phi_1^\alpha, \phi_2^\alpha)$ for each fixed value of $(\phi_1^\beta, \phi_2^\beta)$ as well as is continuous and convex in $(\phi_1^\beta, \phi_2^\beta)$ for each fixed value of $(\phi_1^\alpha, \phi_2^\alpha)$. Therefore the above routing game admits a Nash equilibrium.

The Jacobian matrix with elements $\frac{\partial^2 J^c}{\partial\phi_j^c \partial\phi_i^k}, \quad c,k = \alpha,\beta, \quad i,j = 1,2$ is strictly diagonally dominant for all $(\phi_1^\alpha, \phi_2^\alpha, \phi_1^\beta, \phi_2^\beta)$ such that $\phi_1^\alpha + \phi_2^\alpha = 1,\ \phi_1^\beta + \phi_2^\beta = 1,$ $\phi_1^\alpha,\ \phi_2^\alpha,\ \phi_1^\beta,\ \phi_2^\beta \geq 0,\ C_1 - \lambda^\alpha * \phi_1^\alpha - \lambda^\beta * \phi_1^\beta > 0$ and $C_2 - \lambda^\alpha * \phi_2^\alpha - \lambda^\beta * \phi_2^\beta > 0$. □

The following policy gives the Nash equilibrium solution where class α minimizes its average packet delay, while class β minimizes its blocking probability:

If $\lambda^\alpha < \mu_1 - \sqrt{\frac{\mu_1 * \mu_2 * (\mu_2 + \lambda^\beta)}{\mu_2 + 2 * \lambda^\beta}}$ and $\lambda^\beta \leq \mu_2 * \sqrt{\frac{\mu_1}{\mu_1 - \lambda^\alpha}} - \mu_2,$

then

$$\phi_1^{\alpha *} = 1, \quad \phi^{\beta *} = 0$$

If $\lambda^\alpha < \mu_2 - \sqrt{\frac{\mu_1 * \mu_2 * (\mu_1 + \lambda^\beta)}{\mu_1 + 2 * \lambda^\beta}}$ and $\lambda^\beta \leq \mu_1 * \sqrt{\frac{\mu_2}{\mu_2 - \lambda^\alpha}} - \mu_1$,
then

$$\phi_1^{\alpha *} = 0, \quad \phi^{\beta *} = 1$$

If $\lambda^\alpha < \mu_1$, $\lambda^\beta \geq \sqrt{\mu_1 * (\mu_1 - \lambda^\alpha)} - \mu_1$ and $\lambda^\beta \geq \mu_2 * \sqrt{\frac{\mu_1}{\mu_1 - \lambda^\alpha}} - \mu_2$,
then

$$\phi_1^{\alpha *} = 1, \quad \phi_1^{\beta *} = -\frac{\mu_1}{\lambda^\beta} + \frac{\mu_1 + \mu_2 + \lambda^\beta}{\lambda^\beta} * \frac{\sqrt{\mu_1 * (\mu_1 - \lambda^\alpha)}}{\sqrt{\mu_1 * (\mu_1 - \lambda^\alpha)} + \mu_2}$$

accept the solution only if

$$\lambda^\alpha \leq \mu_1 - \sqrt{\frac{\mu_1}{\frac{1}{\mu_2} + \frac{\lambda^\beta * \phi_2^{\beta *}}{\mu_2 * (\mu_2 + \lambda^\beta * \phi_2^{\beta *})} - \frac{\lambda^\beta * \phi_1^{\beta *}}{\mu_1 * (\mu_1 + \lambda^\beta * \phi_1^{\beta *})}}}$$

If $\lambda^\alpha < \mu_2$, $\lambda^\beta \geq \sqrt{\mu_2 * (\mu_2 - \lambda^\alpha)} - \mu_2$ and $\lambda^\beta \geq \mu_1 * \sqrt{\frac{\mu_2}{\mu_2 - \lambda^\alpha}} - \mu_1$,
then

$$\phi_1^{\alpha *} = 0, \quad \phi_1^{\beta *} = -\frac{\mu_1}{\lambda^\beta} + \frac{\mu_1 + \mu_2 + \lambda^\beta}{\lambda^\beta} * \frac{\mu_1}{\mu_1 + \sqrt{\mu_2 * (\mu_2 - \lambda^\alpha)}}$$

accept the solution only if

$$\lambda^\alpha \leq \mu_2 - \sqrt{\frac{\mu_2}{\frac{1}{\mu_1} + \frac{\lambda^\beta * \phi_1^{\beta *}}{\mu_1 * (\mu_1 + \lambda^\beta * \phi_1^{\beta *})} - \frac{\lambda^\beta * \phi_2^{\beta *}}{\mu_2 * (\mu_2 + \lambda^\beta * \phi_2^{\beta *})}}}$$

If $\mu_1 * (\mu_1 + \lambda^\beta) > \mu_2 * \lambda^\beta$,
$\lambda^\alpha \geq \mu_1 - \mu_1 * \sqrt{\frac{\mu_2 * (\mu_1 + \lambda^\beta)}{\mu_1 * (\mu_1 + \lambda^\beta) - \mu_2 * \lambda^\beta}}$ and
$\lambda^\alpha \geq \mu_2 - \sqrt{\frac{\mu_1 * \mu_2 * (\mu_1 + \lambda^\beta)}{\mu_1 + 2 * \lambda^\beta}}$,
then

$$\phi_1^{\alpha *} = arg \min_{\phi_1^\alpha \geq 0} J^\alpha(\phi_1^\alpha, 1)$$

$$\phi_1^{\beta *} = 1$$

accept the solution only if

$$\mu_1 - \lambda^{\alpha} * \phi_1^{\alpha *} > 0, \quad \mu_2 - \lambda^{\alpha} * \phi_2^{\alpha *} > 0,$$

$$\lambda^{\beta} \leq \sqrt{\frac{\mu_1 * \mu_2 * (\mu_1 - \lambda^{\alpha} * \phi_1^{\alpha *})}{\mu_2 - \lambda^{\alpha} * \phi_2^{\alpha *}}} - \mu_1$$

If $\mu_2 * (\mu_2 + \lambda^{\beta}) > \mu_1 * \lambda^{\beta}$,

$\lambda^{\alpha} \geq \mu_1 - \sqrt{\frac{\mu_1 * \mu_2 * (\mu_2 + \lambda^{\beta})}{\mu_2 + 2 * \lambda^{\beta}}}$ and

$\lambda^{\alpha} \geq \mu_2 - \mu_2 * \sqrt{\frac{\mu_1 * (\mu_2 + \lambda^{\beta})}{\mu_2 * (\mu_2 + \lambda^{\beta}) - \mu_1 * \lambda^{\beta}}}$,

then

$$\phi_1^{\alpha *} = arg \min_{\phi_1^{\alpha} \geq 0} J^{\alpha}(\phi_1^{\alpha}, 0)$$

$$\phi_1^{\beta *} = 0$$

accept the solution only if

$$\mu_1 - \lambda^{\alpha} * \phi_1^{\alpha *} > 0, \quad \mu_2 - \lambda^{\alpha} * \phi_2^{\alpha *} > 0,$$

$$\lambda^{\beta} \leq \sqrt{\frac{\mu_1 * \mu_2 * (\mu_2 - \lambda^{\alpha} * \phi_2^{\alpha *})}{\mu_1 - \lambda^{\alpha} * \phi_1^{\alpha *}}} - \mu_2$$

If all other cases

then

$$\phi_1^{\alpha *} = arg \min_{\phi_1^{\alpha} \geq 0} J^{\alpha}(\phi_1^{\alpha}, \phi_1^{\beta *})$$

$$\phi_1^{\beta *} = arg \min_{\phi_1^{\beta} \geq 0} J^{\beta}(\phi_1^{\alpha *}, \phi_1^{\beta})$$

accept the solution only if

$$\frac{1}{\mu_2} + \frac{\lambda^{\beta} * \phi_2^{\beta *}}{\mu_2 * (\mu_2 + \lambda^{\beta} * \phi_2^{\beta *})} - \frac{\lambda^{\beta} * \phi_1^{\beta *}}{\mu_1 * (\mu_1 + \lambda^{\beta} * \phi_1^{\beta *})} > 0$$

$$\frac{1}{\mu_1} + \frac{\lambda^\beta * \phi_1^{\beta*}}{\mu_1 * (\mu_1 + \lambda^\beta * \phi_1^{\beta*})} - \frac{\lambda^\beta * \phi_2^{\beta*}}{\mu_2 * (\mu_2 + \lambda^\beta * \phi_2^{\beta*})} > 0$$

$$\lambda^\alpha \geq \mu_1 - \sqrt{\frac{\mu_1}{\frac{1}{\mu_2} + \frac{\lambda^\beta * \phi_2^{\beta*}}{\mu_2 * (\mu_2 + \lambda^\beta * \phi_2^{\beta*})} - \frac{\lambda^\beta * \phi_1^{\beta*}}{\mu_1 * (\mu_1 + \lambda^\beta * \phi_1^{\beta*})}}}$$

$$\lambda^\alpha \geq \mu_2 - \sqrt{\frac{\mu_2}{\frac{1}{\mu_1} + \frac{\lambda^\beta * \phi_1^{\beta*}}{\mu_1 * (\mu_1 + \lambda^\beta * \phi_1^{\beta*})} - \frac{\lambda^\beta * \phi_2^{\beta*}}{\mu_2 * (\mu_2 + \lambda^\beta * \phi_2^{\beta*})}}}$$

$$\mu_1 - \lambda^\alpha * \phi_1^{\alpha*} > 0$$

$$\mu_2 - \lambda^\alpha * \phi_2^{\alpha*} > 0$$

$$\lambda^\beta \geq \sqrt{\frac{\mu_1 * \mu_2 * (\mu_1 - \lambda^\alpha * \phi_1^{\alpha*})}{\mu_2 - \lambda^\alpha * \phi_2^{\alpha*}}} - \mu_1$$

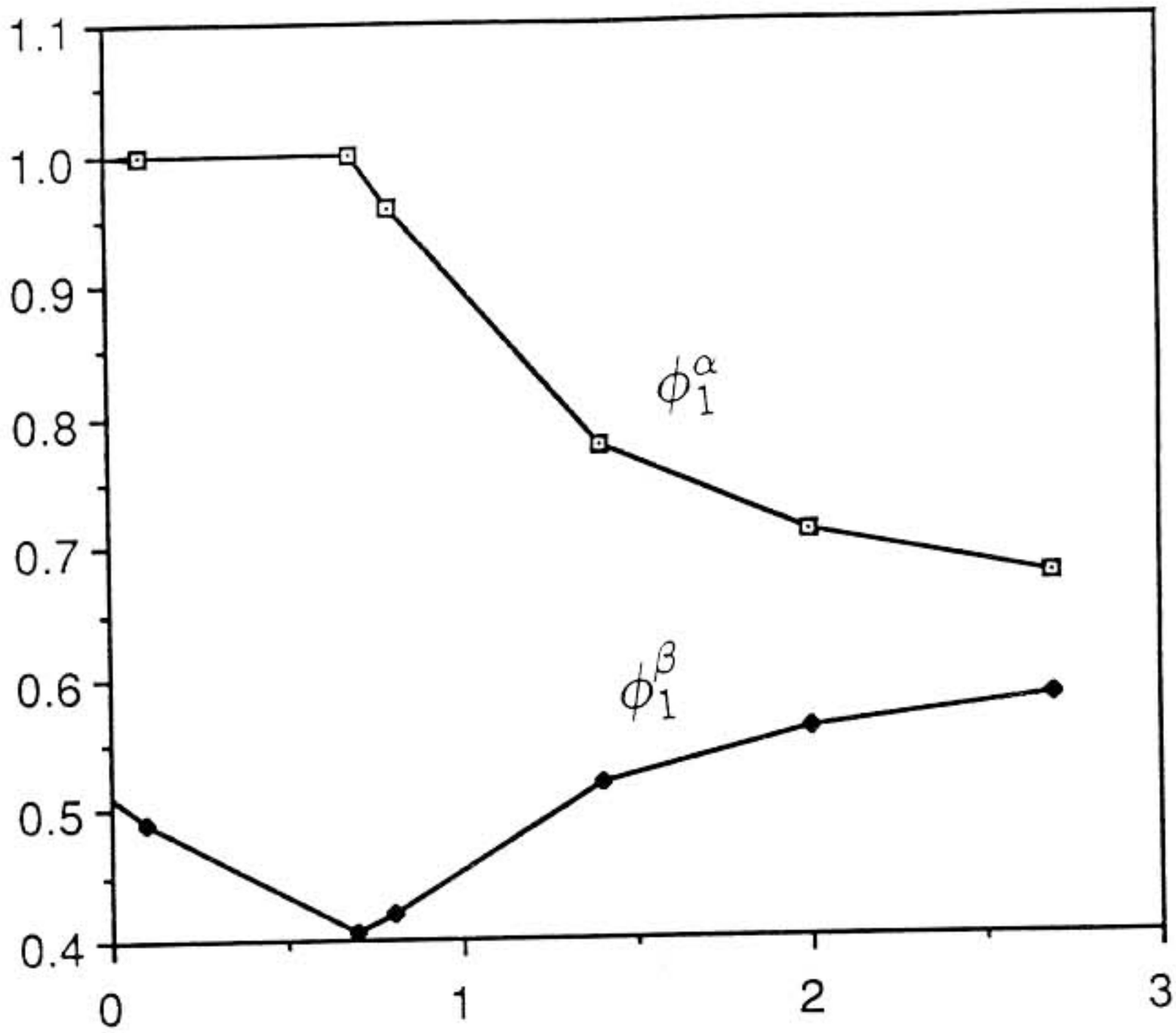
$$\lambda^\beta \geq \sqrt{\frac{\mu_1 * \mu_2 * (\mu_2 - \lambda^\alpha * \phi_2^{\alpha*})}{\mu_1 - \lambda^\alpha * \phi_1^{\alpha*}}} - \mu_2$$

Note that in the above policy the cases $\phi_1^{\alpha*} = \phi_1^{\beta*} = 1$ and $\phi_1^{\alpha*} = \phi_1^{\beta*} = 0$ do not appear. The following Corollary follows.

Corollary:

Let packets from two competing classes α and β arrive according to Poisson distribution. They may be transmitted through multiple links with exponential service time distribution. Class α minimizes its average packet delay, while class β minimizes its blocking probability. For the above routing game, it is never the case that both classes use only the same server simultaneously.

In Figure 4.18, we show the Nash equilibrium fractions $\phi_1^{\alpha*}$, $\phi_1^{\beta*}$, for equal arrival rates $\lambda^\alpha = \lambda^\beta$ and server rates $\mu_1 = 2, \mu_2 = 1$. We notice that for light load, class



$$\lambda^\alpha = \lambda^\beta$$

Figure 4.18: Nash equilibrium fractions $\phi_1^{\alpha*}, \phi_1^{\beta*}$, for equal arrival rates $\lambda^\alpha = \lambda^\beta$ and server rates $\mu_1 = 2, \mu_2 = 1$.

α uses the faster server, while class β uses both servers. As the load increases, both classes use both servers.

In Fig. 4.19, we show the Nash equilibrium fractions $\phi_1^{\alpha*}, \phi_1^{\beta*}$, for fixed class β arrival rate $\lambda^\beta = 1$ and server rates $\mu_1 = 2, \mu_2 = 1$. We notice a behavior similar to that of the previous Figure.

In Fig 4.20, we show the Nash equilibrium fractions $\phi_1^{\alpha*}, \phi_1^{\beta*}$, for fixed class β arrival rate $\lambda^\beta = 0.1$ and server rates $\mu_1 = 2, \mu_2 = 1$. We notice that for light load, class α uses the faster server, while class β uses both servers. As the load increases, class α starts using both servers, while class β starts using only the faster server.

We further explore this last case where the class β arrival rate is very small. We also make the first server much faster than the second server. In Fig. 4.21, we show the Nash equilibrium fractions $\phi_1^{\alpha*}, \phi_1^{\beta*}$, for fixed class β arrival rate $\lambda^\beta = 0.1$ and server rates $\mu_1 = 10, \mu_2 = 1$. We notice a behavior similar to that of the previous Figure. For light traffic load, class α uses exclusively the faster server, while class β uses both servers but with preference to the faster server. As the load increases, class α continues using the faster server, while class β turns to the slow server and it uses it without any interference from class α. However, for heavy load, class α starts using both servers and this has an immediate effect on the blocking probability of class β.

In Fig 4.22, we show the Nash equilibrium class β blocking probability $P[Blocking]^\beta$, for fixed class β arrival rate $\lambda^\beta = 0.1$ and server rates $\mu_1 = 10, \mu_2 = 1$. For intermediate load, class α uses only the faster server and class β uses only the slower server. So, there is no interference between the two classes and the blocking probability of class β becomes independent of the class α arrival rate λ^α. However, as λ^α increases, class α starts using both servers and the class β blocking probability starts increasing.

In Figure 4.23, we show the Nash equilibrium class α average packet delay, T^α, for fixed class β arrival rate $\lambda^\beta = 0.1$ and server rates $\mu_1 = 10, \mu_2 = 1$.

In this section, we have considered the routing problem in integrated services networks, where one class of packets wants to minimize its average packet delay, while another class of packets wants to minimize its blocking probability. We mo-

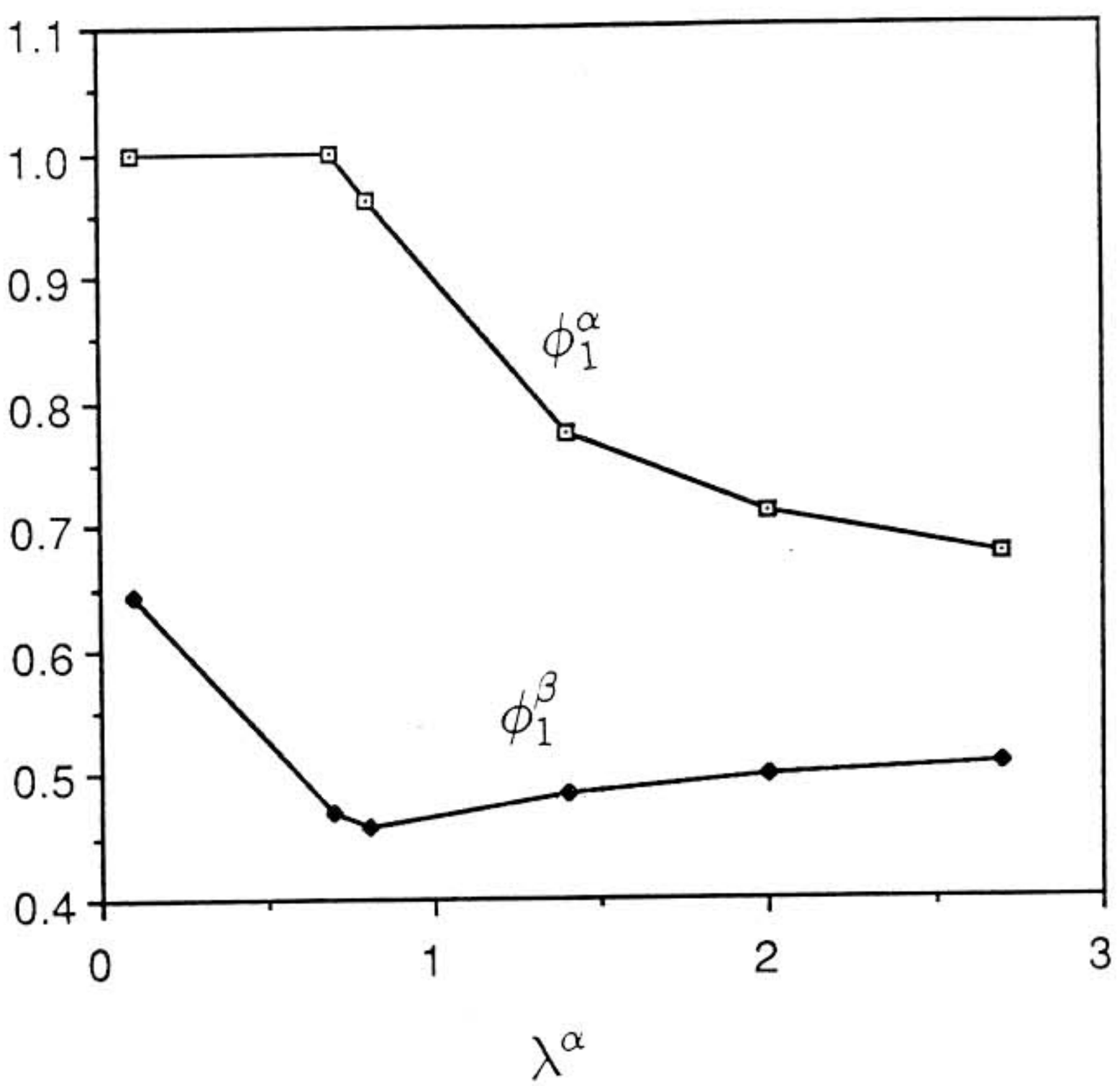


Figure 4.19: Nash equilibrium fractions $\phi_1^{\alpha *}, \phi_1^{\beta *}$, for fixed class β arrival rate $\lambda^\beta = 1$ and server rates $\mu_1 = 2, \mu_2 = 1$.

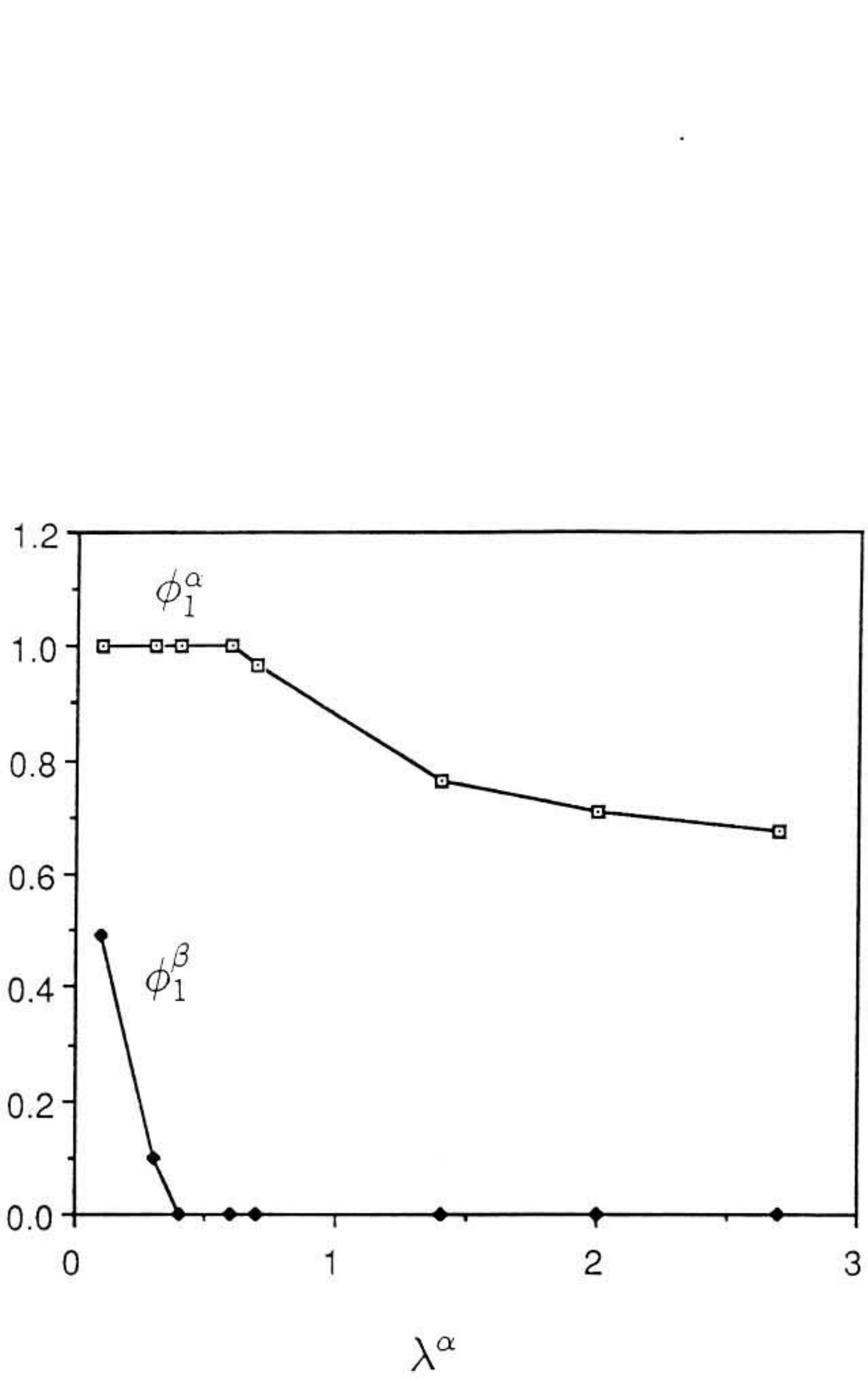


Figure 4.20: Nash equilibrium fractions $\phi_1^{\alpha*}, \phi_1^{\beta*}$, for fixed class β arrival rate $\lambda^\beta = 0.1$ and server rates $\mu_1 = 2, \mu_2 = 1$.

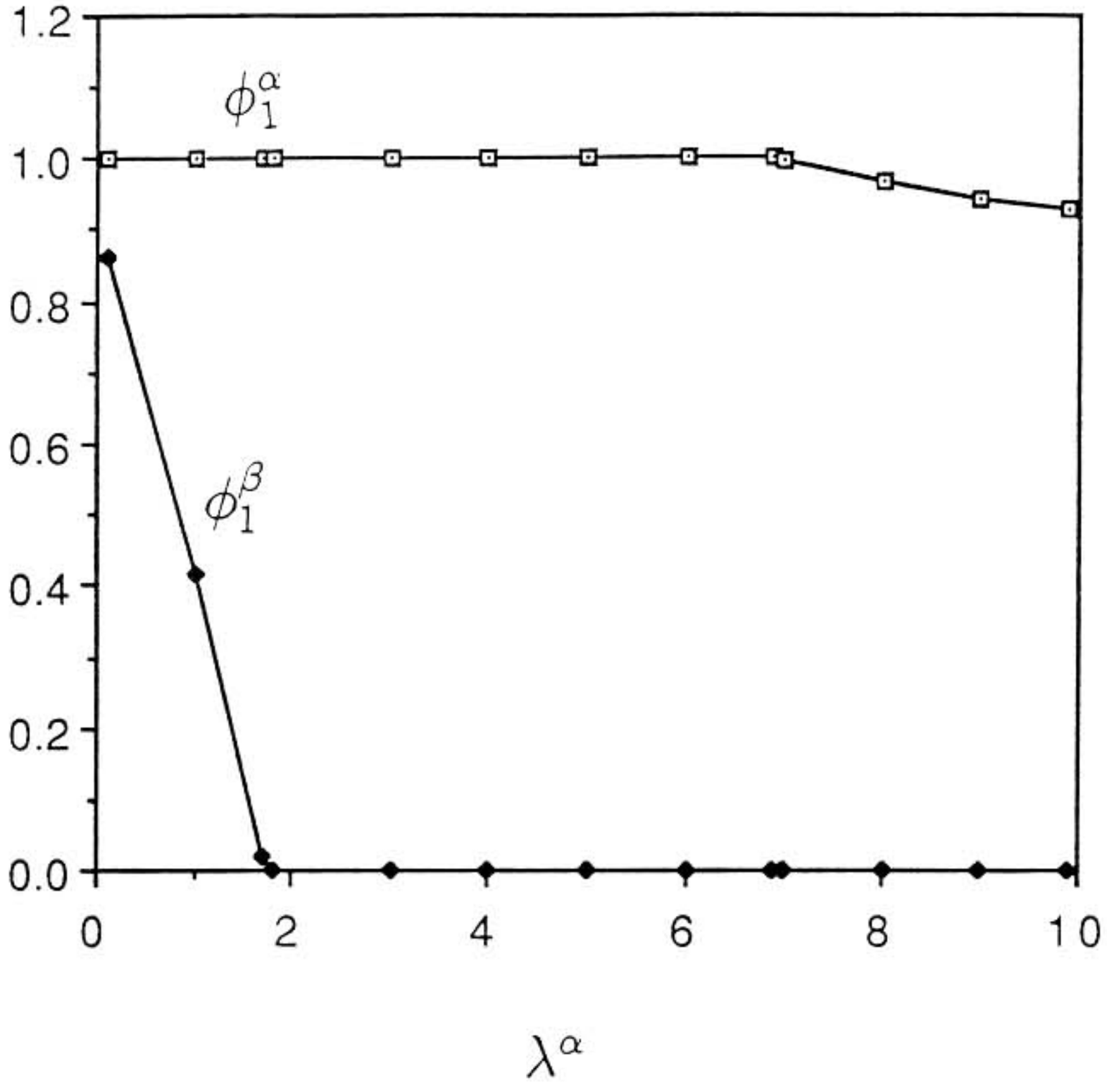


Figure 4.21: Nash equilibrium fractions $\phi_1^{\alpha*}, \phi_1^{\beta*}$, for fixed class β arrival rate $\lambda^\beta = 0.1$ and server rates $\mu_1 = 10, \mu_2 = 1$.

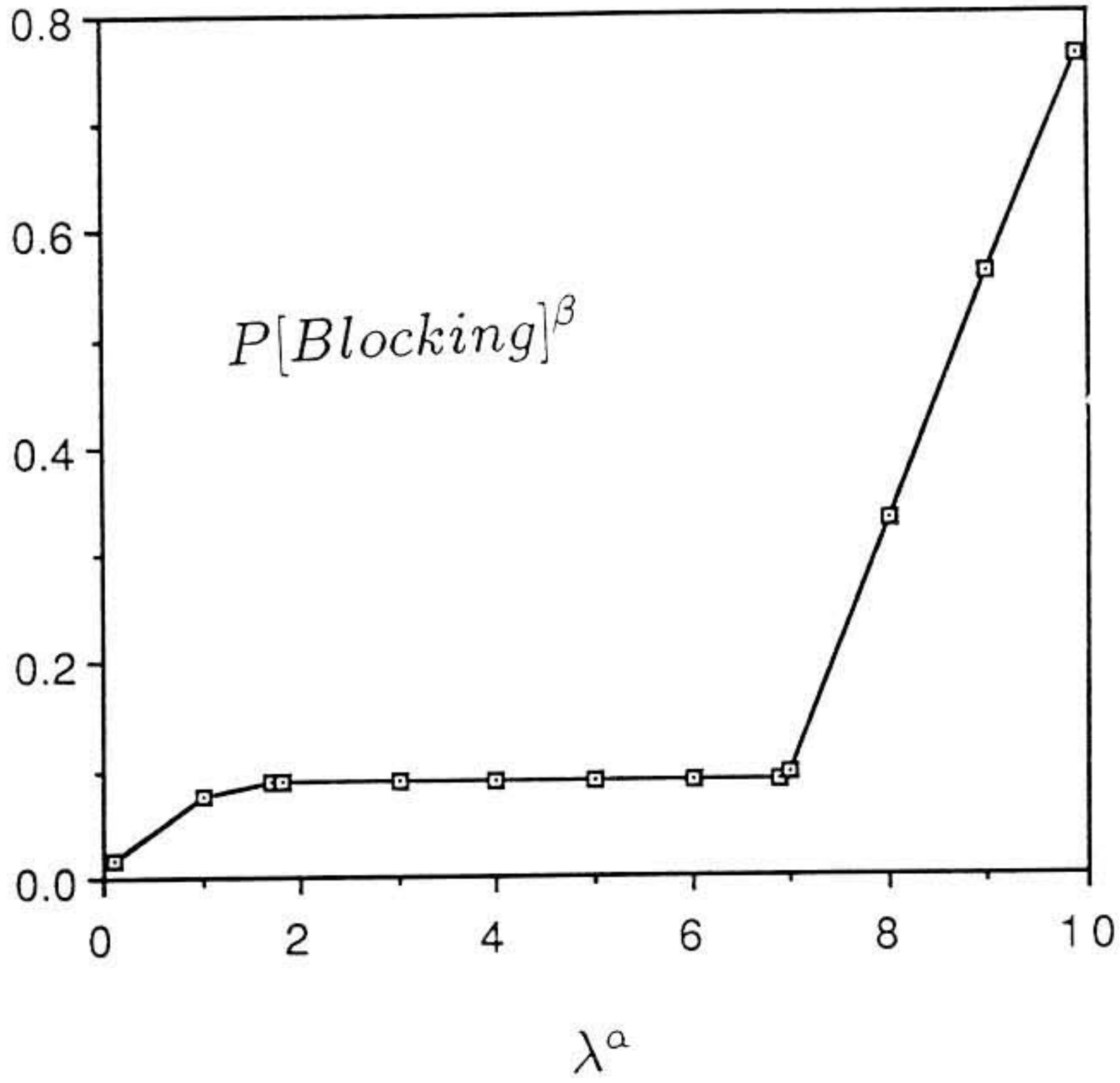


Figure 4.22: Nash equilibrium class β blocking probability $P[Blocking]^{\beta}$, for fixed class β arrival rate $\lambda^{\beta} = 0.1$ and server rates $\mu_1 = 10, \mu_2 = 1$.

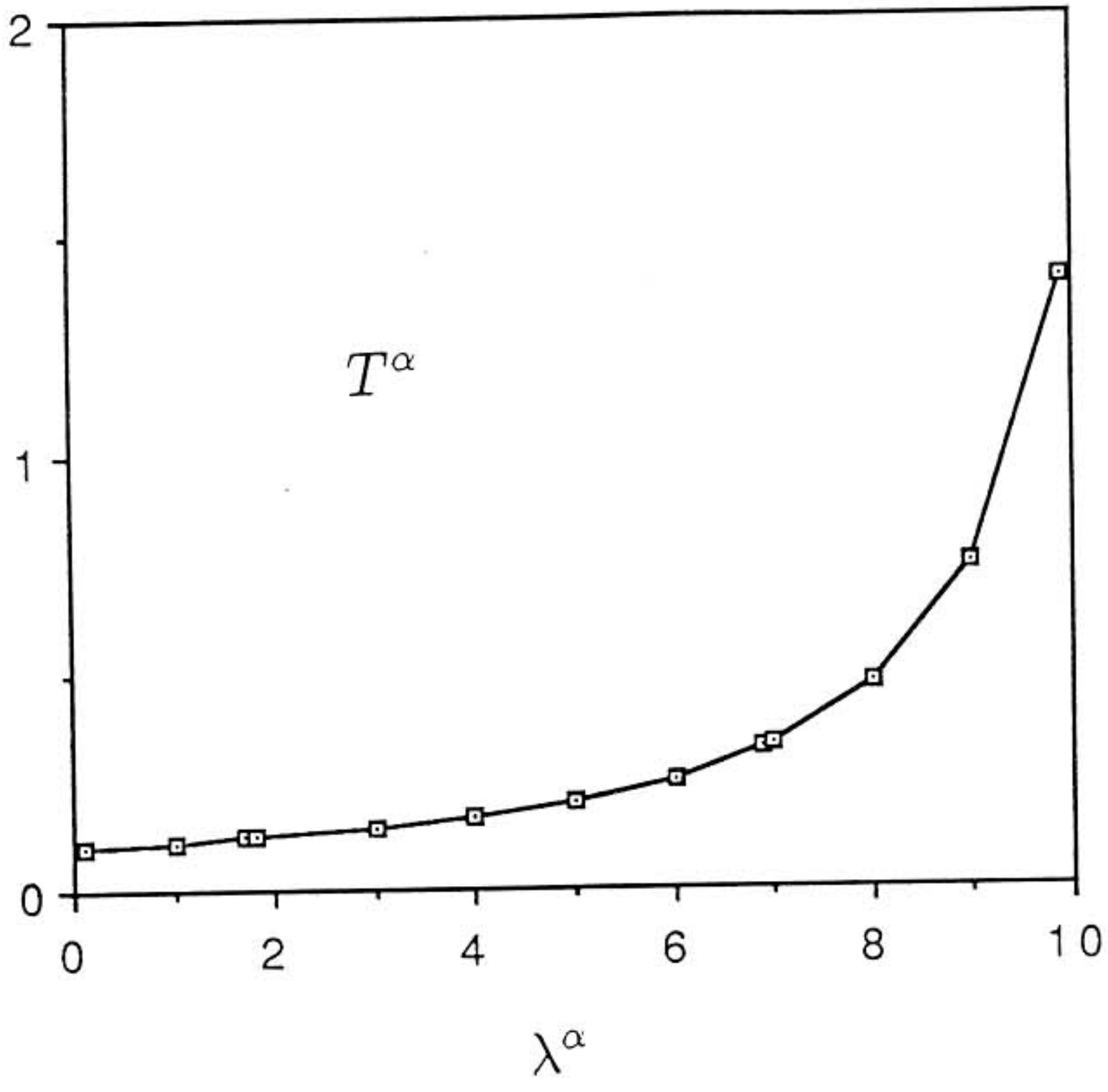


Figure 4.23: Nash equilibrium class α average packet delay, T^α, for fixed class β arrival rate $\lambda^\beta = 0.1$ and server rates $\mu_1 = 10, \mu_2 = 1$.

deled and found several performance measures for a multi-server queueing system, where the packets of one class may be queued, while the packets of another class are blocked when there are more than K packets into the system.

Then we considered the routing problem through a parallel system of such multi-server queues with blocking. We formulated the problem as a Nash game between the two classes and found the Nash equilibrium solution.

Extensions of this work may be to consider an arbitrary network with multiple classes, each class having different blocking threshold and mean service requirement. Classes whose packets may be queued will want to minimize their average packet delay, while those that are blocked will want to minimize their blocking probability.

A UNIFIED GAME-THEORETIC METHODOLOGY FOR THE JOINT LOAD SHARING, ROUTING AND CONGESTION CONTROL PROBLEM

Volume II

by

Anastasios A. Economides

A Dissertation Presented to the

FACULTY OF THE GRADUATE SCHOOL

UNIVERSITY OF SOUTHERN CALIFORNIA

In Partial Fulfillment of the

Requirements for the Degree

DOCTOR OF PHILOSOPHY

(Computer Engineering)

December 1990

Chapter 5

Dynamic Formulation

In this chapter, we develop three novel methodologies for the dynamic problem: i) the dynamic team optimization methodology, ii) the dynamic Nash game methodology, and iii) the dynamic Stackelberg game methodology. For each methodolodology, we develop three alternative formulations of the joint problem, namely an optimal control, a nonlinear complementarity problem and a variational inequality formulation. For each formulation, we state the necessary and sufficient conditions for existence and uniqueness of the solution. From Pontryagin's maximum principle, we also derive the form of the solution, that there should be flow only on minimum length paths, to minimum length destinations, The length at each system resource is appropriately defined for each case. Then we apply these three methodologies to datagram, virtual circuit and integrated services networks. We develop new dynamic queueing models for multiple classes and priority classes of jobs, as well as linearized approximate dynamic queueing models and Wiener process models. We introduce several new cost functions and state constraints. We explicitly solve an example for virtual circuit networks. We consider a virtual circuit network with Poisson arrivals of virtual circuits and packets, and exponential service requirements. We want to minimize the expected cost of servicing or rejecting virtual circuits, minimize the expected cost of packet delay and maximize the expected profit from packet throughput. We find the dynamic team optimality conditions and we propose a state dependent routing and congestion control algorithm. We investigate and compare (via simulation) this state dependent routing

algorithm to the optimal quasi-static algorithm. We find that the more often that we update the state dependent algorithm and the more recent information that we use the better. When the updating period is not much larger than the mean interarrival time of virtual circuits, then this state dependent algorithm achieves smaller average packet delay than the optimal quasi-static algorithm.

5.1 Team Optimal Solution

In this section, we formulate the dynamic joint load sharing, routing and congestion control problem on the path flow space as a cooperative dynamic team game among cooperative classes.

Customers of each class cooperate in using the resources of the distributed system for the social welfare. The behavior of each class is similar to that of any other class, that is to operate optimally for the average job. Ho [218] presents a tutorial on team theory where the decision makers have access to different information concerning the underline uncertainties. Leitmann [297] provides a rigorous analysis of cooperative and zero-sum non-cooperative games.

Next, we give the definition for a Pareto optimal solution, for the joint load sharing, routing and congestion control problem on the path flows.

Definition:

A vector $(\mathbf{\Phi}^*, \mathbf{\Psi}^*) \in (\mathbf{RC}, \mathbf{LS})$ *is called a* Pareto optimal solution *for a* C*-class joint load sharing, routing and congestion control problem if and only if there exists no other vector* $(\mathbf{\Phi}, \mathbf{\Psi}) \in (\mathbf{RC}, \mathbf{LS})$ *such that*

$$J^c(\mathbf{\Phi}, \mathbf{\Psi}) \quad \leq J^c(\mathbf{\Phi}^*, \mathbf{\Psi}^*) \quad \forall\, (\mathbf{\Phi}, \mathbf{\Psi}) \in (\mathbf{RC}, \mathbf{LS})$$

with strict inequality holding for at least one class c.

Define a global cost function

$$J(\mathbf{\Phi}, \mathbf{\Psi}) = \left[\sum_c [w^c * J^c(\mathbf{\Phi}, \mathbf{\Psi})]^p\right]^{1/p}$$

where $1 \leq p < \infty$, $\quad \sum_{c=1}^{C} w^c = 1, \quad w^c \geq 0 \quad \forall\, c.$

For $p \rightarrow \infty$, we have a *minimax* problem [122], since the cost function becomes

$$J(\mathbf{\Phi}, \mathbf{\Psi}) = \max_{c} \{w^c * J^c(\mathbf{\Phi}, \mathbf{\Psi})\}$$

Another problem formulation is

$$\min_{\epsilon, \mathbf{\Phi}, \mathbf{\Psi}} \ \epsilon$$

such that

$$w^c * J^c(\mathbf{\Phi}, \mathbf{\Psi}) \leq \epsilon \quad \forall\, c$$

Furthermore, another problem formulation is

$$\min_{\mathbf{\Phi}, \mathbf{\Psi}} \ J(\mathbf{\Phi}, \mathbf{\Psi})$$

such that

$$J^c(\mathbf{\Phi}, \mathbf{\Psi}) \leq \hat{J}^c(\mathbf{\Phi}, \mathbf{\Psi}) \quad \forall\, c$$

where $\hat{J}^c$ is the maximum acceptable value for the cost function J^c.

Next, we give the definition for a team optimal solution [27], for the joint load sharing, routing and congestion control problem on the path flows.

Definition:

A vector $(\mathbf{\Phi}^*, \mathbf{\Psi}^*) \in (\mathbf{RC}, \mathbf{LS})$ *is called a team optimal solution for a C-class joint load sharing, routing and congestion control problem if and only if*

$$\mathrm{J}\,(\Phi^*, \mathbf{\Psi}^*) \quad \leq J(\mathbf{\Phi}, \mathbf{\Psi}) \quad \forall\, (\mathbf{\Phi}, \mathbf{\Psi}) \in (\mathbf{RC}, \mathbf{LS})$$

In the next sections, we develop three alternative formulations for the joint load sharing, routing and congestion control problem.

5.1.1 Optimal Control Formulation

In this section, we formulate the dynamic cooperative joint load sharing, routing and congestion control problem as an Optimal Control Problem (OCP). Algorithms for solving OCPs is a thoroughly investigated research area and popular algorithms

may be found in books by Athans & Falb [14], Lee & Markus [292], Plant [381], Sage [415], McCausland [325], Dyer & McReynolds [131], Kirk [254], Russell [412], Gruver & Sachs [203], Sethi & Thompson [440], Knowles [262], Lewis [301] among others.

Define the Hamiltonian as

$$H(t, \mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi}, \mathbf{P}) = g(t, \mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi}) + \mathbf{P} * \mathbf{f}(t, \mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi})$$

where $\mathbf{P} = [... \; P^{c,k}_{ij[sd]} \; ... \; P^{c,k}_{i[sd]} \; ... \; P^{c,k}_{o[sd]} \; ... \; P^{c,k}_{[.d][sd]} \; ...]$: vector of costate variables.

Define also the derivatives of H with respect to the congestion, routing and load sharing fractions at $(t, \mathbf{X}^*(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t), \mathbf{P}(t))$ as

$$\frac{\partial H^*}{\phi^c_{o[sd]}} = \frac{\partial H(t, \mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi}, \mathbf{P})}{\phi^c_{o[sd]}} \Big|_{(t, \mathbf{X}^*(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t), \mathbf{P}(t))}$$

$$\frac{\partial H^*}{\phi^c_{\pi[sd]}} = \frac{\partial H(t, \mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi}, \mathbf{P})}{\phi^c_{\pi[sd]}} \Big|_{(t, \mathbf{X}^*(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t), \mathbf{P}(t))}$$

$$\frac{\partial H^*}{\psi^c_{[sd]}} = \frac{\partial H(t, \mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi}, \mathbf{P})}{\psi^c_{[sd]}} \Big|_{(t, \mathbf{X}^*(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t), \mathbf{P}(t))}$$

Define also the Lagrangian as

$$\begin{aligned} \mathrm{L}\,(\mathrm{t}, \mathbf{X}, \Phi, \mathbf{\Psi}, \mathbf{P}, \mathbf{Q}) \quad &= \mathrm{H}\,(\mathrm{t}, \mathbf{X}, \Phi, \mathbf{\Psi}, \mathbf{P}) + \\ &+ \sum_c \sum_{[sd] \in \mathbf{SD}^c} Q^c_{[sd]} * \left[1 - \phi^c_{o[sd]} - \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \phi^c_{\pi[sd]} \right] + \\ &+ \sum_c \sum_{[s.] \in \mathbf{S}^c} Q^c_{[s.]} * \left[1 - \sum_{[.d] \in \mathbf{D}^c_{[s.]}} \psi^c_{[sd]} \right] \end{aligned}$$

$$with \qquad \phi^c_{o[sd]}, \; \phi^c_{\pi[sd]}, \; \psi^c_{[sd]} \geq 0 \quad \forall \; \pi[sd] \in \mathbf{\Pi}^c_{[sd]}, \; [sd] \in \mathbf{SD}^c, \; c$$

where $\mathbf{Q} = [... \; Q^c_{[sd]} \; ... \; Q^c_{[s.]} \; ...]$: vector of multipliers for the constraints of the congestion control, routing and load sharing fractions.

Define also the derivatives of L with respect to the congestion, routing and load sharing fractions at $(t, \mathbf{X}^*(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t), \mathbf{P}(t))$ as

$$\frac{\partial L^*}{\phi^c_{o[sd]}} = \frac{\partial L(t, \mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi}, \mathbf{P}, \mathbf{Q})}{\phi^c_{o[sd]}} \mid_{(t, \mathbf{X}^*(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t), \mathbf{P}(t), \mathbf{Q}(t))}$$

$$\frac{\partial L^*}{\phi^c_{\pi[sd]}} = \frac{\partial L(t, \mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi}, \mathbf{P}, \mathbf{Q})}{\phi^c_{\pi[sd]}} \mid_{(t, \mathbf{X}^*(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t), \mathbf{P}(t), \mathbf{Q}(t))}$$

$$\frac{\partial L^*}{\psi^c_{[sd]}} = \frac{\partial L(t, \mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi}, \mathbf{P}, \mathbf{Q})}{\psi^c_{[sd]}} \mid_{(t, \mathbf{X}^*(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t), \mathbf{P}(t), \mathbf{Q}(t))}$$

Theorem :

Consider the dynamic joint load sharing, routing and congestion control problem in distributed systems with multiple cooperative classes, with fixed initial time t_0 *and final time* t_f.

$(\mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) \in (\mathbf{RC}, \mathbf{LS})$ *is a team optimal solution if and only if it solves the following Optimal Control Problem:*

$$\textit{minimize} \qquad \int_{t_0}^{t_f} g(t, \mathbf{X}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t))dt$$

$$\textit{with respect to} \quad (\Phi(t), \mathbf{\Psi}(t))$$

$$\textit{such that} \qquad \dot{\mathbf{X}}(t) = \mathbf{f}(t, \mathbf{X}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t))$$

$$\mathbf{X}(t_0) = \mathbf{X}_0$$

$$(\Phi(t), \mathbf{\Psi}(t)) \in (\mathbf{RC}, \mathbf{LS})$$

Proof: It follows from the definition of the team optimal solution.□

Necessary conditions for optimality are provided by Pontryagin's Maximum Principle. Besides the previously referred books on optimal control theory, some other books that contain material on Pontryagin's maximum principle are the following: Hestenes [215], Arrow & Kurz [12], Tabak & Kuo [476], Boltyanskii

[58], Berkovitz [33], Bryson & Ho [79], Fleming & Rishel [162], Boltyanskii [59], Leitmann [298], Macki & Strauss [315], Alekseev, Tikhomirov & Fomin [8].

Theorem : *necessary conditions*

Consider the dynamic joint load sharing, routing and congestion control problem in distributed systems with multiple cooperative classes, with fixed initial time t_0 *and final time* t_f.

Let $g(t, \mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi})$, $\mathbf{f}(t, \mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi})$, *be continuously differentiable with respect to* $(\mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi}) \in (\mathbf{R}^n, \mathbf{\Phi}, \mathbf{\Psi}) \quad \forall\, t \in [t_0, t_f]$.

If $(\hat{\mathbf{\Phi}}^*(t, \mathbf{X}_0), \hat{\mathbf{\Psi}}^*(t, \mathbf{X}_0)) = (\mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) \in (\mathbf{RC}, \mathbf{LS})$ *is a piecewise continuous open-loop team optimal solution and* $\{\mathbf{X}^*(t),\ t \in [t_0, t_f]\}$ *is the corresponding state trajectory, then* $\exists\ \mathbf{P}(t) : [t_0, t_f] \rightarrow \mathbf{R}^n$ *continuous and piecewise continuously differentiable vector function, such that* $\forall\ t \in [t_0, t_f]$:

$$\dot{\mathbf{X}}^*(t) = \mathbf{f}(t, \mathbf{X}^*(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\mathbf{X}^*(t_0) = \mathbf{X}_0$$

$$\left[\frac{\partial H^*}{\partial \phi^c_{o[sd]}} - Q^c_{[sd]}(t)\right] * \phi^{c*}_{o[sd]}(t) = 0 \qquad \forall\ [sd] \in \mathbf{SD}^c,\ c$$

$$\left[\frac{\partial H^*}{\partial \phi^c_{\pi[sd]}} - Q^c_{[sd]}(t)\right] * \phi^{c*}_{\pi[sd]}(t) = 0 \quad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c, c$$

$$\left[\frac{\partial H^*}{\partial \psi^c_{[sd]}} - Q^c_{[s.]}(t)\right] * \psi^{c*}_{[sd]}(t) = 0 \qquad \forall\ [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c,\ c$$

$$\frac{\partial H^*}{\partial \phi^c_{o[sd]}} - Q^c_{[sd]}(t) \geq 0 \qquad \forall\ [sd] \in \mathbf{SD}^c,\ c$$

$$\frac{\partial H^*}{\partial \phi^c_{\pi[sd]}} - Q^c_{[sd]}(t) \geq 0 \qquad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c,\ c$$

$$\frac{\partial H^*}{\partial \psi^c_{[sd]}} - Q^c_{[s.]}(t) \geq 0 \qquad \forall\ [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c,\ c$$

$$\dot{\mathbf{P}}(t) = -\nabla_{\mathbf{X}} H(t, \mathbf{X}^*, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t), \mathbf{P}(t))$$

$$\mathbf{P}(\mathrm{t}_f) = \mathbf{0}$$

$$\phi^{c*}_{o[sd]}(t) + \sum_{\pi[sd]\in\mathbf{\Pi}^c_{[sd]}} \phi^{c*}_{\pi[sd]}(t) = 1 \qquad \forall\ [sd] \in \mathbf{SD}^c,\ c$$

$$\sum_{[.d]\in\mathbf{D}^c_{[s.]}} \psi^{c*}_{[sd]}(t) = 1 \qquad \forall\ [s.] \in \mathbf{S}^c,\ c$$

$$\phi^{c*}_{o[sd]}(t),\ \phi^{c*}_{\pi[sd]}(t) \geq 0 \qquad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c,\ c$$

$$\psi^{c*}_{[sd]}(t) \geq 0 \qquad \forall\ [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c,\ c$$

Proof: The Lagrangian is

$$L = H + \sum_c \sum_{[sd]\in\mathbf{SD}^c} Q^c_{[sd]} * \left[1 - \phi^c_{o[sd]} - \sum_{\pi[sd]\in\mathbf{\Pi}^c_{[sd]}} \phi^c_{\pi[sd]}\right] +$$

$$+ \sum_c \sum_{[s.]\in\mathbf{S}^c} Q^c_{[s.]} * \left[1 - \sum_{[.d]\in\mathbf{D}^c_{[s.]}} \psi^c_{[sd]}\right]$$

with $\phi^c_{o[sd]},\ \phi^c_{\pi[sd]},\ \psi^c_{[sd]} \geq 0 \quad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c,\ c$

Pontryagin's maximum principle necessary conditions are:

$$\dot{\mathbf{X}}^*(t) = \mathbf{f}(t, \mathbf{X}^*(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\mathbf{X}^*(t_0) = \underset{\sim}{\mathbf{X}}_0$$

$$\frac{\partial L^*}{\partial \phi^c_{o[sd]}} * \phi^{c*}_{o[sd]}(t) = 0 \;\Rightarrow\; \left[\frac{\partial H^*}{\partial \phi^c_{o[sd]}} - Q^c_{[sd]}(t)\right] * \phi^{c*}_{o[sd]}(t) = 0$$

$$\forall\ [sd] \in \mathbf{SD}^c,\ c$$

$$\frac{\partial L^*}{\partial \phi^c_{\pi[sd]}} * \phi^{c*}_{\pi[sd]}(t) = 0 \;\Rightarrow\; \left[\frac{\partial H^*}{\partial \phi^c_{\pi[sd]}} - Q^c_{[sd]}(t)\right] * \phi^{c*}_{\pi[sd]}(t) = 0$$

$$\forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c, c$$

$$\frac{\partial L^*}{\partial \psi^c_{[sd]}} * \psi^{c*}_{[sd]}(t) = 0 \;\Rightarrow\; \left[\frac{\partial H^*}{\partial \psi^c_{[sd]}} - Q^c_{[s.]}(t)\right] * \psi^{c*}_{[sd]}(t) = 0$$

$$\forall\ [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c,\ c$$

$$\frac{\partial L^*}{\partial \phi^c_{o[sd]}} \geq 0 \;\Rightarrow\; \frac{\partial H^*}{\partial \phi^c_{o[sd]}} - Q^c_{[sd]}(t) \geq 0 \qquad \forall\ [sd] \in \mathbf{SD}^c,\ c$$

$$\frac{\partial L^*}{\partial \phi^c_{\pi[sd]}} \geq 0 \;\Rightarrow\; \frac{\partial H^*}{\partial \phi^c_{\pi[sd]}} - Q^c_{[sd]}(t) \geq 0 \qquad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c,\ c$$

$$\frac{\partial L^*}{\partial \psi^c_{[sd]}} \geq 0 \;\Rightarrow\; \frac{\partial H^*}{\partial \psi^c_{[sd]}} - Q^c_{[s.]}(t) \geq 0 \qquad \forall\ [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c,\ c$$

$$\dot{\mathbf{P}}(t) = -\nabla_{\mathbf{X}} H(t, \mathbf{X}^*, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t), \mathbf{P}(t))$$

$$\mathbf{P}(t_f) = \mathbf{0}$$

$$\frac{\partial L^*}{\partial Q^c_{[sd]}} = 0 \;\Rightarrow \phi^{c*}_{o[sd]}(t) + \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \phi^{c*}_{\pi[sd]}(t) = 1 \quad \forall\, [sd] \in \mathbf{SD}^c,\; c$$

$$\frac{\partial L^*}{\partial Q^c_{[s.]}} = 0 \;\Rightarrow \sum_{[.d] \in \mathbf{D}^c_{[s.]}} \psi^{c*}_{[sd]}(t) = 1 \qquad \forall\, [s.] \in \mathbf{S}^c,\; c$$

$$\phi^{c*}_{o[sd]}(t),\; \phi^{c*}_{\pi[sd]}(t) \geq 0 \qquad \forall\, \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\; [sd] \in \mathbf{SD}^c,\; c$$

$$\psi^{c*}_{[sd]}(t) \geq 0 \qquad \forall\, [.d] \in \mathbf{D}^c_{[s.]},\; [s.] \in \mathbf{S}^c,\; c4$$

□

Theorem : *sufficient conditions*

Consider the dynamic joint load sharing, routing and congestion control problem in distributed systems with multiple cooperative classes, with fixed initial time t_0 *and final time* t_f.

Let $g(t, \mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi})$, $\mathbf{f}(t, \mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi})$, *be continuously differentiable with respect to* $(\mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi}) \in (\mathbf{R}^n, \mathbf{\Phi}, \mathbf{\Psi})\ \ \forall\, t \in [t_0, t_f]$.

Let $(\bar{\mathbf{X}}(t), \bar{\mathbf{\Phi}}(t), \bar{\mathbf{\Psi}}(t)) \in (\mathbf{R}^n, \mathbf{RC}, \mathbf{LS})$ *is an admissible pair for the Optimal Control Problem and* $H(t, \mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi}, \mathbf{P}(t))$ *is convex in* $(\mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi}) \in (\mathbf{R}^n, \mathbf{RC}, \mathbf{LS})$, $\forall t \in [t_0, t_f]$. *If* $\exists\ \mathbf{P}(t) : [t_0, t_f] \rightarrow \mathbf{R}^n$ *continuous and piecewise continuously differentiable vector function, such that* $\forall\ t \in [t_0, t_f]$:

$$\dot{\bar{\mathbf{X}}}(t) = \mathbf{f}(t, \bar{\mathbf{X}}(t), \bar{\mathbf{\Phi}}(t), \bar{\mathbf{\Psi}}(t))$$

$$\bar{\mathbf{X}}(t_0) = \mathbf{X}_0$$

$$\left[\frac{\partial H}{\partial \phi^c_{o[sd]}} - Q^c_{[sd]}(t)\right] * \bar{\phi}^c_{o[sd]}(t) = 0 \qquad \forall\ [sd] \in \mathbf{SD}^c,\ c$$

$$\left[\frac{\partial H}{\partial \phi^c_{\pi[sd]}} - Q^c_{[sd]}(t)\right] * \bar{\phi}^c_{\pi[sd]}(t) = 0 \qquad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c, c$$

$$\left[\frac{\partial H}{\partial \psi^c_{[sd]}} - Q^c_{[s.]}(t)\right] * \bar{\psi}^c_{[sd]}(t) = 0 \qquad \forall\ [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c,\ c$$

$$\frac{\partial H}{\partial \phi^c_{o[sd]}} - Q^c_{[sd]}(t) \geq 0 \qquad \forall\ [sd] \in \mathbf{SD}^c,\ c$$

$$\frac{\partial H}{\partial \phi^c_{\pi[sd]}} - Q^c_{[sd]}(t) \geq 0 \qquad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c,\ c$$

$$\frac{\partial H}{\partial \psi^c_{[sd]}} - Q^c_{[s.]}(t) \geq 0 \qquad \forall\ [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c,\ c$$

$$\dot{\mathbf{P}}(t) = -\nabla_{\mathbf{X}} H(t, \bar{\mathbf{X}}, \bar{\mathbf{\Phi}}(t), \bar{\mathbf{\Psi}}(t), \mathbf{P}(t))$$

$$\mathbf{P}(\mathrm{t}_f) = \mathbf{0}$$

$$\bar{\phi}^c_{o[sd]}(t) + \sum_{\pi[sd]\in\mathbf{\Pi}^c_{[sd]}} \bar{\phi}^c_{\pi[sd]}(t) = 1 \qquad \forall\ [sd] \in \mathbf{SD}^c,\ c$$

$$\sum_{[.d]\in\mathbf{D}^c_{[s.]}} \bar{\psi}^c_{[sd]}(t) = 1 \qquad \forall\ [s.] \in \mathbf{S}^c,\ c$$

$$\bar{\phi}^c_{o[sd]}(t),\ \bar{\phi}^c_{\pi[sd]}(t) \geq 0 \qquad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c,\ c$$

$$\bar{\psi}^c_{[sd]}(t) \geq 0 \qquad \forall\ [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c,\ c$$

then $(\bar{\mathbf{X}}(t), \bar{\mathbf{\Phi}}(t), \bar{\mathbf{\Psi}}(t))$ *is optimal.*

Proof: The proof is similar to that of the necessary conditions. In addition, we use the convexity of the Hamiltonian with respect to the state and controls.□

Theorem :

Consider the dynamic joint load sharing, routing and congestion control problem in distributed systems with multiple cooperative classes, with fixed initial time t_0 *and final time* t_f.

Let $g(t, \mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi})$, $\mathbf{f}(t, \mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi})$, *be continuously differentiable with respect to* $(\mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi}) \in (\mathbf{R}^n, \mathbf{RC}, \mathbf{LS})$, $\forall\ t \in [t_0, t_f]$.

If $(\hat{\mathbf{\Phi}}^*(t, \mathbf{X}, \mathbf{X}_0), \hat{\mathbf{\Psi}}^*(t, \mathbf{X}, \mathbf{X}_0)) = (\mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) \in (\mathbf{RC}, \mathbf{LS})$ *is a* <u>*closed-loop memoryless team optimal solution*</u> *such that* $(\hat{\mathbf{\Phi}}^*(t, \mathbf{X}, \mathbf{X}_0), \hat{\mathbf{\Phi}}^*(t, \mathbf{X}, \mathbf{X}_0))$ *is continuously differentiable with respect to* $\mathbf{X} \in \mathbf{R}^n$, $\forall\ c$, $t \in [t_0, t_f]$ *and* $\{\mathbf{X}^*(t),\ t \in [t_0, t_f]\}$ *is the corresponding state trajectory, then* $\exists\ \mathbf{P}(t) : [t_0, t_f] \rightarrow \mathbf{R}^n$, *continuous and piecewise continuously differentiable vector functions, such that* $\forall\ t \in [t_0, t_f]$:

$$\dot{\mathbf{X}}^*(t) = \mathbf{f}(t, \mathbf{X}^*(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\mathbf{X}^*(t_0) = \mathbf{X}_0$$

$$\left[\frac{\partial H^*}{\partial \phi^c_{o[sd]}} - Q^c_{[sd]}(t)\right] * \phi^{c*}_{o[sd]}(t) = 0 \qquad \forall\ [sd] \in \mathbf{SD}^c,\ c$$

$$\left[\frac{\partial H^*}{\partial \phi^c_{\pi[sd]}} - Q^c_{[sd]}(t)\right] * \phi^{c*}_{\pi[sd]}(t) = 0 \qquad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c, c$$

$$\left[\frac{\partial H^*}{\partial \psi^c_{[sd]}} - Q^c_{[s.]}(t)\right] * \psi^{c*}_{[sd]}(t) = 0 \qquad \forall\ [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c,\ c$$

$$\frac{\partial H^*}{\partial \phi^c_{o[sd]}} - Q^c_{[sd]}(t) \geq 0 \qquad \forall\ [sd] \in \mathbf{SD}^c,\ c$$

$$\frac{\partial H^*}{\partial \phi^c_{\pi[sd]}} - Q^c_{[sd]}(t) \geq 0 \qquad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c,\ c$$

$$\frac{\partial H^*}{\partial \psi^c_{[sd]}} - Q^c_{[s.]}(t) \geq 0 \qquad \forall\ [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c,\ c$$

$$\dot{\mathbf{P}}(t) = -\nabla_{\mathbf{X}} H(t, \mathbf{X}^*, \hat{\mathbf{\Phi}}^*(t, \mathbf{X}^*, \mathbf{X}_0), \hat{\mathbf{\Psi}}^*(t, \mathbf{X}^*, \mathbf{X}_0), \mathbf{P}(t))$$

$$\mathbf{P}\ (\mathrm{t}_f) = \mathbf{0}$$

$$\phi^{c*}_{o[sd]}(t) + \sum_{\pi[sd]\in\mathbf{\Pi}^c_{[sd]}} \phi^{c*}_{\pi[sd]}(t) = 1 \qquad \forall\ [sd] \in \mathbf{SD}^c,\ c$$

$$\sum_{[.d]\in\mathbf{D}^c_{[s.]}} \psi^{c*}_{[sd]}(t) = 1 \qquad \forall\ [s.] \in \mathbf{S}^c,\ c$$

$$\phi^{c*}_{o[sd]}(t),\ \phi^{c*}_{\pi[sd]}(t) \geq 0 \qquad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c,\ c$$

$$\psi^{c*}_{[sd]}(t) \geq 0 \qquad \forall\ [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c,\ c$$

Proof: The proof is similar to that of the open-loop solution.□

5.1.2 Dynamic Programming Formulation

In this section, we formulate the dynamic cooperative joint load sharing, routing and congestion control problem as a Dynamic Programming Problem (DPP). Algorithms for solving DPP's may be found in books by Bellman [31], Howard [220], Kumar & Varaiya [274] Bertsekas [37], Ross [406] among others.

Theorem :

Consider the dynamic joint load sharing, routing and congestion control problem in distributed systems with multiple cooperative classes, with fixed initial time t_0 *and final time* t_f.

$(\mathbf{\Phi^*}, \mathbf{\Psi^*}) \in (\mathbf{RC}, \mathbf{LS})$ *is a team optimal solution if and only if the following conditions are satisfied:*

i) $\int_{t_0}^{t_f} g(t, \mathbf{X^*}(s), \hat{\mathbf{\Phi}}^*(\mathbf{X^*}(s))), \hat{\mathbf{\Psi}}^*(\mathbf{X^*}(s))ds = constant$

ii) $\exists\ \mathbf{X^*},\ \mathbf{P}$ absolutely continuous such that:

$$\mathrm{H}\ (\mathrm{t}, \mathbf{X^*}(t), \mathbf{\Phi^*}(\mathbf{X}(t)), \mathbf{\Psi^*}(\mathbf{X}(t)), \mathbf{P}(t)) \ - \ H^c(t, \mathbf{X}(t), \mathbf{\Phi}(\mathbf{X}(t)), \mathbf{\Psi}(\mathbf{X}(t)), \mathbf{P}(t)) +$$

$$+\ \dot{\mathbf{P}}(t) * (\mathbf{X^*}(t) - \mathbf{X}) \ \leq 0 \qquad a.e.\ \ t \in [t_0, t_f],\ \forall\ \mathbf{X} \in \mathbf{R}^n,\ (\mathbf{\Phi}, \mathbf{\Psi}) \in (\mathbf{RC}, \mathbf{LS})$$

$$\mathbf{P}\ (\mathrm{t}_f) * (\mathbf{X^*}(t_f) - \mathbf{X}) \leq 0 \quad \forall\ \mathbf{X} \in \mathbf{R}^n$$

Proof: Substituting the state equation in ii) and integrating it, we get the definition of the team-optimal solution. □

Definition :

Consider the dynamic joint load sharing, routing and congestion control problem in distributed systems with multiple cooperative classes, with fixed initial time t_0 *and final time* t_f.

Under the memoryless perfect state or closed-loop perfect state information structure, $(\hat{\mathbf{\Phi}}, \hat{\mathbf{\Psi}}) \in (\mathbf{RC}, \mathbf{LS})$ *constitutes a feedback team optimal solution solution if and only if* $\exists\ V : [t_0, t_f] * \mathbf{R}^n \rightarrow R$ *satisfying the following relations:*

$$
\begin{aligned}
\mathrm{V(t,X)} &= \int_t^{t_f} g(s, \mathbf{X}^*(s), \hat{\mathbf{\Phi}}^*(s, \mathbf{I}(s)), \hat{\mathbf{\Psi}}^*(s, \mathbf{I}(s)))ds \leq \\
&\leq \int_t^{t_f} g(s, \mathbf{X}^*(s), \hat{\mathbf{\Phi}}(s, \mathbf{I}(s)), \hat{\mathbf{\Psi}}(s, \mathbf{I}(s)))ds \\
&\quad \forall\ (\hat{\mathbf{\Phi}}(s, \mathbf{I}(s)), \hat{\mathbf{\Psi}}(s, \mathbf{I}(s))) \in (\mathbf{RC}, \mathbf{LS}),\ \mathbf{X} \in \mathbf{R}^n \\
&\quad \text{such that} \quad \forall\ s \in [t, t_f] \\
\dot{\mathrm{X}}(s) &= \mathbf{f}(\mathrm{s}, \mathbf{X}\ (\mathrm{s}), \hat{\mathbf{\Phi}}(s, \mathbf{I}(s)), \hat{\mathbf{\Psi}}(s, \mathbf{I}(s))) \\
\mathbf{X}\ (\mathrm{t}) &= \mathbf{X} \\
\dot{\mathrm{X}}^*(s) &= \mathbf{f}(\mathrm{s}, \mathbf{X}^*(s), \hat{\mathbf{\Phi}}^*(s, \mathbf{I}(s)), \hat{\mathbf{\Psi}}^*(s, \mathbf{I}(s))) \\
\mathrm{X}^*(s) &= \mathbf{X}
\end{aligned}
$$

where $\mathbf{I}(s) = \{\mathbf{X}(s), \mathbf{X}_0\}$ or $\mathbf{I}(s) = \{\mathbf{X}(\tau),\ \tau \leq s\}$.

$V(t, \mathbf{X})$ is the value function associated with the optimal control problem of minimizing J over $(\bar{\mathbf{\Phi}}, \bar{\mathbf{\Psi}}) \in \mathbf{LS}, \mathbf{RC})$.

The concept of feedback team optimal solution means that if $(\mathbf{\Phi}(s), \mathbf{\Psi}(s))$ is a feedback team optimal solution to the problem during $[t_0, t_f]$, is also a feedback team optimal solution to the problem during $[t, t_f]$, with the initial state taken as $\mathbf{X}(t)$. So, feedback team optimal solution strategies will depend only on the time variable and the current value of the state, but not on memory.

Proposition :

Every open-loop team optimal solution for the dynamic joint load sharing, routing and congestion control problem among cooperative classes is also closed-loop team optimal solution.

Proposition :

Under the memoryless (respectively, closed-loop) perfect state information structure, every feedback team optimal solution for the dynamic joint load sharing, routing and congestion control problem among cooperative classes is a closed-loop no memory (respectively, closed-loop) team optimal solution.

Theorem :

Consider the dynamic joint load sharing, routing and congestion control problem in distributed systems with multiple cooperative classes, with fixed initial time t_0 *and final time* t_f.

Under the memory perfect state or closed loop perfect state information structure, $(\hat{\mathbf{\Phi}}, \hat{\mathbf{\Psi}}) \in (\mathbf{RC}, \mathbf{LS})$ *provides a feedback team optimal solution if* $\exists V : [t_0, t_f] * \mathbf{R}^n \rightarrow \mathbf{R}$ *continuously differentiable satisfying the partial differential equations*

$$-\frac{\partial V(t,\mathbf{X})}{\partial t} = \min_{(\mathbf{\Phi},\mathbf{\Psi})\in(\mathbf{RC},\mathbf{LS})} \left\{ \frac{\partial V(t,\mathbf{X})}{\partial \mathbf{X}} * \mathbf{f}(t,\mathbf{X},\mathbf{\Phi},\mathbf{\Psi}) + g(t,\mathbf{X},\mathbf{\Phi},\mathbf{\Psi}) \right\} =$$

$$= \frac{\partial V(t,\mathbf{X})}{\partial \mathbf{X}} * \mathbf{f}(t,\mathbf{X},\hat{\mathbf{\Phi}}^*(t,\mathbf{X}),\hat{\mathbf{\Psi}}^*(t,\mathbf{X})) + g(t,\mathbf{X},\hat{\mathbf{\Phi}}^*(t,\mathbf{X}),\hat{\mathbf{\Psi}}^*(t,\mathbf{X}))$$

The above equation is called Hamilton-Jacobi-Bellman (H-J-B) equation.

5.1.3 Nonlinear Complementarity Problem Formulation

In this section, we formulate the dynamic cooperative load sharing, routing and congestion control problem as a Nonlinear Complementarity Problem (NCP).

Define the vector of class load sharing, routing and load sharing fractions as well as Lagrange multipliers:

$$\mathbf{Z}(t) = [... \ \phi^c_{o[sd]}(t) \ ... \ \phi^c_{\pi[sd]} \ ... \ Q^c_{[sd]}(t) \ .... \ \psi^c_{[sd]}(t) \ ... \ Q^c_{[s.]}(t) \ ...]^T$$

and the vector of class derivative of the Lagrangian with respect to the congestion control, routing and load sharing fractions as well as Lagrange multipliers:

$$\nabla L(t,\mathbf{X}(t),\mathbf{Z}(t)) = \left[\ldots \left(\frac{\partial H}{\partial \phi^c_{o[sd]}} - Q^c_{o[sd]}(t)\right) \ldots \left(\frac{\partial H}{\partial \phi^c_{\pi[sd]}} - Q^c_{\pi[sd]}(t)\right) \ldots \right.$$

$$\ldots \left(1 - \phi^c_{o[sd]}(t) - \sum_{\pi[sd]\in\mathbf{\Pi}^c_{[sd]}} \phi^c_{\pi[sd]}(t)\right) \ldots$$

$$\left. \ldots \left(\frac{\partial H}{\partial \psi^c_{[sd]}} - Q^c_{[s.]}(t)\right) \ldots \left(1 - \sum_{[.d]\in\mathbf{D}^c_{[s.]}} \psi^c_{[sd]}(t)\right) \ldots \right]$$

Theorem :

Consider the dynamic joint load sharing, routing and congestion control problem in distributed systems with multiple cooperative classes, with fixed initial time t_0 *and final time* t_f.

Let $g(t,\mathbf{X},\mathbf{\Phi},\mathbf{\Psi})$, $\mathbf{f}(t,\mathbf{X},\mathbf{\Phi},\mathbf{\Psi})$, *be continuously differentiable with respect to* $(\mathbf{X},\mathbf{\Phi},\mathbf{\Psi}) \in (\mathbf{R}^n,\mathbf{\Phi},\mathbf{\Psi}) \quad \forall\ t \in [t_0,t_f]$. *If* H *is differentiable and convex in* $(\mathbf{X},\mathbf{\Phi},\mathbf{\Psi}) \in (\mathbf{RC},\mathbf{LS}), \quad \forall\ t \in [t_0,t_f]$,

then $(\mathbf{\Phi}^*(t),\mathbf{\Psi}^*(t)) \in (\mathbf{RC},\mathbf{LS})$ *is a team optimal solution if and only if it solves the following* Nonlinear Complementarity Problem $\forall\ t \in [t_0,t_f]$:

$$\nabla L(t,\mathbf{X}^*(t),\mathbf{Z}^*(t)) * \mathbf{Z}^*(t) = 0$$

$$\nabla L(t,\mathbf{X}^*(t),\mathbf{Z}^*(t)) \geq 0$$

$$\mathbf{Z}^*(t) \geq 0$$

$$\dot{\mathbf{X}}^*(t) = \mathbf{f}(t,\mathbf{X}^*(t),\mathbf{\Phi}^*(t),\mathbf{\Psi}^*(t))$$

$$\mathbf{X}^*(t_0) = \mathbf{X}_0$$

$$\dot{\mathbf{P}}(t) = -\nabla_{\mathbf{X}} H(t,\mathbf{X}^*,\mathbf{\Phi}^*(t),\mathbf{\Psi}^*(t),\mathbf{P}(t))$$

$$\mathbf{P}(t_f) = \mathbf{0}$$

Proof: After some algebraic manipulations, we find that the NCP: $\nabla L(\mathbf{Z}(t)) * \mathbf{Z}(t) = 0 \quad \nabla L(\mathbf{Z}(t)) \geq 0 \quad \mathbf{Z}(t) \geq 0$ with $\mathbf{Z}(t)$ and $\nabla L(\mathbf{Z}(t))$ as defined above, is equivalent to the Pontryagin's maximum principle necessary conditions. □

5.1.4 Variational Inequality Formulation

In this section, we formulate the dynamic cooperative load sharing, routing and congestion control problem as a Variational Inequality Problem (VIP).

Define the vector of class congestion control, routing and load sharing fractions:

$$(\mathbf{\Phi}(t), \mathbf{\Psi}(t)) = \left[\ldots \phi^c_{o[sd]}(t) \ldots \phi^c_{\pi[sd]}(t) \ldots \psi^c_{[sd]}(t) \ldots \right]^T$$

as well the vector of class derivatives of the cost function with respect to the congestion control, routing and load sharing fractions:

$$\nabla H(t, \mathbf{X}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t), \mathbf{P}(t)) \quad = \quad \left[\ldots \frac{\partial H}{\partial \phi^c_{o[sd]}} \ldots \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \frac{\partial H}{\partial \phi^c_{\pi[sd]}} \ldots \frac{\partial H}{\partial \psi^c_{[sd]}} \ldots \right]$$

Theorem :

Consider the dynamic joint load sharing, routing and congestion control problem in distributed systems with multiple cooperative classes, with fixed initial time t_0 *and final time* t_f.

Let $g(t, \mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi})$, $\mathbf{f}(t, \mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi})$, *be continuously differentiable with respect to* $(\mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi}) \in (\mathbf{R}^n, \mathbf{\Phi}, \mathbf{\Psi}) \quad \forall\ t \in [t_0, t_f]$. *If* H *is continuously differentiable and convex in* $(\mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi}) \in (\mathbf{R}^n, \mathbf{RC}, \mathbf{LS})$, $\quad \forall\ t \in [t_0, t_f]$,

then $(\mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) \in (\mathbf{RC}, \mathbf{LS})$ *is a team optimal solution if and only if it solves the following* Variational Inequality Problem $\forall\ t \in [t_0, t_f]$:

$$\nabla H(t, \mathbf{X}^*(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t), \mathbf{P}(t)) * ((\mathbf{\Phi}, \mathbf{\Psi}) - (\mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))) \geq 0$$
$$\forall\ (\mathbf{\Phi}, \mathbf{\Psi}) \in (\mathbf{RC}, \mathbf{LS})$$
$$\dot{\mathbf{X}}^*(t) = \mathbf{f}(t, \mathbf{X}^*(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$
$$\mathbf{X}^*(t_0) = \mathbf{X}_0$$
$$\dot{\mathbf{P}}(t) = -\nabla_{\mathbf{X}} H(t, \mathbf{X}^*, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t), \mathbf{P}(t))$$
$$\mathbf{P}(t_f) = \mathbf{0}$$

Proof: If $(\mathbf{\Phi}(t), \mathbf{\Psi}(t))$ is a local minimum for the following minimization problem

$$\begin{aligned} &\textit{minimize} && \int_{t_0}^{t_f} g(t, \mathbf{X}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t))dt \\ &\textit{with respect to} && (\Phi(t), \mathbf{\Psi}(t)) \\ &\textit{such that} && \dot{\mathbf{X}}(t) = \mathbf{f}(t, \mathbf{X}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t)) \\ & && \mathbf{X}(t_0) = \mathbf{X}_0 \end{aligned}$$

$$(\Phi(t), \mathbf{\Psi}(t)) \in (\mathbf{RC}, \mathbf{LS})$$

and g is a continuously differentiable convex function over the nonempty convex, closed and bounded set $(\mathbf{RC}, \mathbf{LS})$, then $\forall\, t \in [t_0, t_f]$:

$$\sum_{c} \sum_{[sd] \in \mathbf{SD}^c} \left\{ \frac{\partial H^*}{\partial \phi^c_{o[sd]}} * (\phi^c_{o[sd]} - \phi^{c*}_{o[sd]}(t)) + \right.$$

$$+ \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \frac{\partial H^*}{\partial \phi^c_{\pi[sd]}} * (\phi^c_{\pi[sd]} - \phi^{c*}_{\pi[sd]}(t)) +$$

$$\left. + \frac{\partial H^*}{\partial \psi^c_{[sd]}} * (\psi^c_{[sd]} - \psi^{c*}_{[sd]}(t)) \right\} \geq 0 \quad \forall\, (\mathbf{\Phi}, \mathbf{\Psi}) \in (\mathbf{RC}, \mathbf{LS})$$

□

Another equivalent VIP formulation is the following Theorem:

Theorem :

Consider the dynamic joint load sharing, routing and congestion control problem in distributed systems with multiple cooperative classes, with fixed initial time t_0 *and final time* t_f.

Let $g(t, \mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi})$, $\mathbf{f}(t, \mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi})$, *be continuously differentiable with respect to* $(\mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi}) \in (\mathbf{R}^n, \mathbf{\Phi}, \mathbf{\Psi}) \quad \forall\ t \in [t_0, t_f]$. *If* H *is continuously differentiable and convex in* $(\mathbf{H}, \mathbf{\Phi}, \mathbf{\Psi}) \in (\mathbf{R}^n, \mathbf{RC}, \mathbf{LS}), \quad \forall\ t \in [t_0, t_f]$,

then $(\mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) \in (\mathbf{RC}, \mathbf{LS})$ *is a team optimal solution if and only if it solves the following* Variational Inequality Problem $\forall\ t \in [t_0, t_f]$:

$$\nabla L(t, \mathbf{X}^*(t), \mathbf{Z}^*(t)) * (\mathbf{Z} - \mathbf{Z}^*(t)) \geq 0 \qquad \forall\ \mathbf{Z} > 0$$

$$\dot{\mathbf{X}}^*(t) = \mathbf{f}(t, \mathbf{X}^*(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\mathbf{X}^*(t_0) = \mathbf{X}_0$$

$$\dot{\mathbf{P}}(t) = -\nabla_{\mathbf{X}} H(t, \mathbf{X}^*, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t), \mathbf{P}(t))$$

$$\mathbf{P}(t_f) = \mathbf{0}$$

The NCP: $f(x^*) * x^* = 0 \quad f(x^*) \geq 0 \quad x^* > 0$
and the VIP: find x^* such that $f(x^*) * (x - x^*) \geq 0 \quad \forall x > 0$
are equivalent.□

5.1.5 Maximum Principle for Separable Cost Functions

In this section, we derive the first order necessary conditions for a team optimal solution on the path flows, when the cost function of each resource depends only on the flow on this resource.

According to the team optimal solution definition, each class c minimizes its cost function g given the optimum decisions of all other classes.

$$
\begin{aligned}
\textit{minimize} \quad & \int_{t_0}^{t_f} g(t, \mathbf{X}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t))dt = \\
& = \sum_{ij} \int_{t_0}^{t_f} g_{ij}(t, \mathbf{X}_{ij}(t), \mathbf{\Lambda}_{ij}(t)_d t \quad + \\
& + \quad \sum_{i} \int_{t_0}^{t_f} g_i(t, \mathbf{X}_i(t), \mathbf{\Lambda}_i(t))dt \quad + \\
& + \quad \sum_{[sd]} \int_{t_0}^{t_f} g_{o[sd]}(t, \mathbf{X}_{o[sd]}(t), \mathbf{\Lambda}_{o[sd]}(t))dt \quad + \\
& + \quad \sum_{[.d]} \int_{t_0}^{t_f} g_{[.d]}(t, \mathbf{X}_{[.d]}(t), \mathbf{\Lambda}_{[.d]}(t))dt \\
\textit{with respect to} \quad & (\Phi(t), \mathbf{\Psi}(t))
\end{aligned}
$$

such that

$$\dot{\mathbf{X}}^k_{ij[sd]}(t) = \mathbf{f}^k_{ij[sd]}(t, \mathbf{X}_{ij}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t)) \qquad \forall\ ij,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{X}}^k_{i[sd]}(t) = \mathbf{f}^k_{i[sd]}(t, \mathbf{X}_{i}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t)) \qquad \forall\ i,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{X}}^k_{o[sd]}(t) = \mathbf{f}^k_{o[sd]}(t, \mathbf{X}_{o[sd]}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t)) \qquad \forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{X}}^k_{[.d][sd]}(t) = \mathbf{f}^k_{[.d][sd]}(t, \mathbf{X}_{[.d]}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t)) \qquad \forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$E\mathbf{X}^k_{ij[sd]}(t_0) = \mathbf{X}^k_{ij[sd],0} \qquad \forall\ ij,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\mathbf{X}^k_{i[sd]}(t_0) = \mathbf{X}^k_{i[sd],0} \qquad \forall\ i,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\mathbf{X}^k_{o[sd]}(t_0) = \mathbf{X}^k_{o[sd],0} \qquad \forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\mathbf{X}^k_{[.d][sd]}(t_0) = \mathbf{X}^k_{[.d][sd],0} \qquad \forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\phi^c_{o[sd]}(t) + \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \phi^c_{\pi[sd]}(t) = 1 \qquad \forall\ [sd] \in \mathbf{SD}^c$$

$$\sum_{[.d] \in \mathbf{D}^c_{[s.]}} \psi^c_{[sd]}(t) = 1 \qquad \forall\ [s.] \in \mathbf{S}^c$$

$$\phi^c_{o[sd]}(t),\ \phi^c_{\pi[sd]}(t) \geq 0 \qquad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c$$

$$\psi^c_{[sd]}(t) \geq 0 \qquad \forall\ [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c$$

Pontryagin's maximum principle necessary conditions are $\forall\, t \in [t_0, t_f]$:

$$\dot{\mathbf{X}}^{k*}_{ij[sd]}(t) = \mathbf{f}^{k}_{ij[sd]}(t, \mathbf{X}^{*}_{ij}(t), \mathbf{\Phi}^{*}(t), \mathbf{\Psi}^{*}(t)) \qquad \forall\, ij,\ [sd] \in \mathbf{SD}^{k},\ k$$

$$\dot{\mathbf{X}}^{k*}_{i[sd]}(t) = \mathbf{f}^{k}_{i[sd]}(t, \mathbf{X}^{*}_{i}(t), \mathbf{\Phi}^{*}(t), \mathbf{\Psi}^{*}(t)) \qquad \forall\, i,\ [sd] \in \mathbf{SD}^{k},\ k$$

$$\dot{\mathbf{X}}^{k*}_{o[sd]}(t) = \mathbf{f}^{k}_{o[sd]}(t, \mathbf{X}^{*}_{o[sd]}(t), \mathbf{\Phi}^{*}(t), \mathbf{\Psi}^{*}(t)) \qquad \forall\, [sd] \in \mathbf{SD}^{k},\ k$$

$$\dot{\mathbf{X}}^{k*}_{[.d][sd]}(t) = \mathbf{f}^{k}_{[.d][sd]}(t, \mathbf{X}^{*}_{[.d]}(t), \mathbf{\Phi}^{*}(t), \mathbf{\Psi}^{*}(t)) \qquad \forall\, [sd] \in \mathbf{SD}^{k},\ k$$

$$\mathbf{X}^{k*}_{ij[sd]}(t_0) = \mathbf{X}^{k}_{ij[sd],0} \qquad \forall\, ij,\ [sd] \in \mathbf{SD}^{k},\ k$$

$$\mathbf{X}^{k*}_{i[sd]}(t_0) = \mathbf{X}^{k}_{i[sd],0} \qquad \forall\, i,\ [sd] \in \mathbf{SD}^{k},\ k$$

$$\mathbf{X}^{k*}_{o[sd]}(t_0) = \mathbf{X}^{k}_{o[sd],0} \qquad \forall\, [sd] \in \mathbf{SD}^{k},\ k$$

$$\mathbf{X}^{k*}_{[.d][sd]}(t_0) = \mathbf{X}^{k}_{[.d][sd],0} \qquad \forall\, [sd] \in \mathbf{SD}^{k},\ k$$

$$\left[\frac{\partial g_{[sd]}(t,\mathbf{X}^{*}_{o[sd]}(t),\mathbf{\Phi}^{*}(t),\mathbf{\Psi}^{*}(t))}{\partial \phi^{c}_{o[sd]}}+\right.$$

$$+\sum_{k}\mathbf{P}^{k,c}_{o[sd]} * \nabla_{\phi^{c}_{o[sd]}}\mathbf{f}^{k}_{o[sd]}(t,\mathbf{X}^{*}_{o[sd]}(t),\mathbf{\Phi}^{*}(t),\mathbf{\Psi}^{*}(t))-$$

$$\left.-Q^{c}_{[sd]}(t)\right] * \phi^{c*}_{o[sd]}(t)=0 \qquad \forall\ [sd]\in\mathbf{SD}^{c},\ c$$

$$\left[\sum_{ij}\frac{\partial g_{ij}(t,\mathbf{X}^{*}_{ij}(t),\mathbf{\Phi}^{*}(t),\mathbf{\Psi}^{*}(t))}{\partial \phi^{c}_{\pi[sd]}}+\sum_{i}\frac{\partial g_{i}(t,\mathbf{X}^{*}_{i}(t),\mathbf{\Phi}^{*}(t),\mathbf{\Psi}^{*}(t))}{\partial \phi^{c}_{\pi[sd]}}+\right.$$

$$+\sum_{k}\sum_{[s'd']}\sum_{ij}\mathbf{P}^{k,c}_{ij[s'd']}(t) * \nabla_{\phi^{c}_{\pi[sd]}}\mathbf{f}^{k}_{ij[s'd']}(t,\mathbf{X}^{*}_{ij}(t),\mathbf{\Phi}^{*}(t),\mathbf{\Psi}^{*}(t))+$$

$$+\sum_{k}\sum_{[s'd']}\sum_{i}\mathbf{P}^{k,c}_{i[s'd']}(t) * \nabla_{\phi^{c}_{\pi[sd]}}\mathbf{f}^{c}_{i[s'd']}(t,\mathbf{X}^{*}_{i}(t),\mathbf{\Phi}^{*}(t),\mathbf{\Psi}^{*}(t))-$$

$$\left.-Q^{c}_{[sd]}(t)\right] * \phi^{c*}_{\pi[sd]}(t)=0 \qquad \forall\ \pi[sd]\in\mathbf{\Pi}^{c}_{[sd]},\ [sd]\in\mathbf{SD}^{c},\ c$$

$$\Bigg[\sum_{ij}\frac{\partial g_{ij}(t,\mathbf{X}^*_{ij}(t),\mathbf{\Phi}^*(t),\mathbf{\Psi}^*(t))}{\partial\psi^c_{[sd]}}+\sum_{i}\frac{\partial g_i(t,\mathbf{X}^*_i(t),\mathbf{\Phi}^*(t),\mathbf{\Psi}^*(t))}{\partial\psi^c_{[sd]}}+$$

$$+\frac{\partial g_{[sd]}(t,\mathbf{X}^*_{o[sd]}(t),\mathbf{\Phi}^*(t),\mathbf{\Psi}^*(t))}{\partial\psi^c_{[sd]}}+\frac{\partial g_{[.d]}(t,\mathbf{X}^*_{[.d]}(t),\mathbf{\Phi}^*(t),\mathbf{\Psi}^*(t))}{\partial\psi^c_{[sd]}}+$$

$$+\sum_k\sum_{[s'd']}\sum_{ij}\mathbf{P}^{k,c}_{ij[s'd']}(t)*\nabla_{\psi^c_{[sd]}}\mathbf{f}^k_{ij[s'd']}(t,\mathbf{X}^*_{ij}(t),\mathbf{\Phi}^*(t),\mathbf{\Psi}^*(t))+$$

$$+\sum_k\sum_{[s'd']}\sum_{i}\mathbf{P}^{k,c}_{i[s'd']}(t)*\nabla_{\psi^c_{[sd]}}\mathbf{f}^k_{i[s'd']}(t,\mathbf{X}^*_{i}(t),\mathbf{\Phi}^*(t),\mathbf{\Psi}^*(t))+$$

$$+\sum_k\mathbf{P}^{k,c}_{o[sd]}(t)*\nabla_{\psi^c_{[sd]}}\mathbf{f}^k_{o[sd]}(t,\mathbf{X}^*_{o[sd]}(t),\mathbf{\Phi}^*(t),\mathbf{\Psi}^*(t))+$$

$$+\sum_k\sum_{[s'.]}\mathbf{P}^{k,c}_{[.d][s'd]}(t)*\nabla_{\psi^c_{[sd]}}\mathbf{f}^k_{[.d][s'd]}(t,\mathbf{X}^*_{[.d]}(t),\mathbf{\Phi}^*(t),\mathbf{\Psi}^*(t))-$$

$$-Q^c_{[s.]}(t)\Bigg]*\psi^{c*}_{[sd]}(t)=0 \qquad \forall\ [.d]\in\mathbf{D}^c_{[s.]},\ [s.]\in\mathbf{S}^c,\ c$$

$$\frac{\partial g_{[sd]}(t, \mathbf{X}^{*}_{o[sd]}(t), \mathbf{\Phi}^{*}(t), \mathbf{\Psi}^{*}(t))}{\partial \phi^{c}_{o[sd]}} +$$

$$\sum_{k} \mathbf{P}^{k,c}_{o[sd]} * \nabla_{\phi^{c}_{o[sd]}} \mathbf{f}^{k}_{o[sd]}(t, \mathbf{X}^{*}_{o[sd]}(t), \mathbf{\Phi}^{*}(t), \mathbf{\Psi}^{*}(t)) -$$

$$-Q^{c}_{[sd]}(t) \geq 0 \qquad \forall\, [sd] \in \mathbf{SD}^{c},\ c$$

$$\sum_{ij} \frac{\partial g_{ij}(t, \mathbf{X}^{*}_{ij}(t), \mathbf{\Phi}^{*}(t), \mathbf{\Psi}^{*}(t))}{\partial \phi^{c}_{\pi[sd]}} + \sum_{i} \frac{\partial g_{i}(t, \mathbf{X}^{*}_{i}(t), \mathbf{\Phi}^{*}(t), \mathbf{\Psi}^{*}(t))}{\partial \phi^{c}_{\pi[sd]}} +$$

$$+ \sum_{k} \sum_{[s'd']} \sum_{ij} \mathbf{P}^{k,c}_{ij[s'd']}(t) * \nabla_{\phi^{c}_{\pi[sd]}} \mathbf{f}^{k}_{ij[s'd']}(t, \mathbf{X}^{*}_{ij}(t), \mathbf{\Phi}^{*}(t), \mathbf{\Psi}^{*}(t)) +$$

$$+ \sum_{k} \sum_{[s'd']} \sum_{i} \mathbf{P}^{k,c}_{i[s'd']}(t) * \nabla_{\phi^{c}_{\pi[sd]}} \mathbf{f}^{c}_{i[s'd']}(t, \mathbf{X}^{*}_{i}(t), \mathbf{\Phi}^{*}(t), \mathbf{\Psi}^{*}(t)) -$$

$$-Q^{c}_{[sd]}(t) \geq 0 \qquad \forall\, \pi[sd] \in \mathbf{\Pi}^{c}_{[sd]},\ [sd] \in \mathbf{SD}^{c},\ c$$

$$\sum_{ij} \frac{\partial g_{ij}(t, \mathbf{X}^*_{ij}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))}{\partial \psi^c_{[sd]}} + \sum_{i} \frac{\partial g_i(t, \mathbf{X}^*_{i}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))}{\partial \psi^c_{[sd]}} +$$

$$+ \frac{\partial g_{[sd]}(t, \mathbf{X}^*_{o[sd]}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))}{\partial \psi^c_{[sd]}} + \frac{\partial g_{[.d]}(t, \mathbf{X}^*_{[.d]}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))}{\partial \psi^c_{[sd]}} +$$

$$+ \sum_{k} \sum_{[s'd']} \sum_{ij} \mathbf{P}^{k,c}_{ij[s'd']}(t) * \nabla_{\psi^c_{[sd]}} \mathbf{f}^k_{ij[s'd']}(t, \mathbf{X}^*_{ij}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) +$$

$$+ \sum_{k} \sum_{[s'd']} \sum_{i} \mathbf{P}^{k,c}_{i[s'd']}(t) * \nabla_{\psi^c_{[sd]}} \mathbf{f}^k_{i[s'd']}(t, \mathbf{X}^*_{i}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) +$$

$$+ \sum_{k} \mathbf{P}^{k,c}_{o[sd]}(t) * \nabla_{\psi^c_{[sd]}} \mathbf{f}^k_{o[sd]}(t, \mathbf{X}^*_{o[sd]}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) +$$

$$+ \sum_{k} \sum_{[s'd]} \mathbf{P}^{k,c}_{[.d][s'd]}(t) * \nabla_{\psi^c_{[sd]}} \mathbf{f}^k_{[.d][s'd]}(t, \mathbf{X}^*_{[.d]}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-Q^c_{[s.]}(t) \geq 0 \qquad \forall\, [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c,\ c$$

$$\dot{\mathbf{P}}^{c,k}_{ij[sd]}(t) = -\nabla_{\mathbf{X}^k_{ij[sd]}} g_{ij}(t, \mathbf{X}^*_{ij}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-\sum_n \sum_{[s'd']} \mathbf{P}^{c,n}_{ij[s'd']}(t) * \nabla_{\mathbf{X}^k_{ij[sd]}} \mathbf{f}^n_{ij[s'd']}(t, \mathbf{X}^*_{ij}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\forall\ ij,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{P}}^{c,k}_{i[sd]}(t) = -\nabla_{\mathbf{X}^k_{i[sd]}} g_i(t, \mathbf{X}^*_i, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-\sum_n \sum_{[s'd']} \mathbf{P}^{c,n}_{i[s'd']}(t) * \nabla_{\mathbf{X}^k_{i[sd]}} \mathbf{f}^n_{i[s'd']}(t, \mathbf{X}^*_i, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\forall\ i,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{P}}^{c,k}_{o[sd]}(t) = -\nabla_{\mathbf{X}^k_{o[sd]}} g_{[sd]}(t, \mathbf{X}^*_{o[sd]}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-\sum \mathbf{P}^{c,n}_{o[sd]}(t) * \nabla_{\mathbf{X}^k_{o[sd]}} \mathbf{f}^n_{o[sd]}(t, \mathbf{X}^*_{o[sd]}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{P}}^{c,k}_{[.d][sd]}(t) = -\nabla_{\mathbf{X}^k_{[.d][sd]}} g_{[.d]}(t, \mathbf{X}^*_{[.d]}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-\sum_n \sum_{[s'.]} \mathbf{P}^{c,n}_{[.d][sd]}(t) * \nabla_{\mathbf{X}^k_{[.d][sd]}} \mathbf{f}^n_{[.d][s'd]}(t, \mathbf{X}^*_{[.d]}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\phi^{c*}_{o[sd]}(t) + \sum_{\pi[sd]\in\mathbf{\Pi}^c_{[sd]}} \phi^{c*}_{\pi[sd]}(t) = 1 \quad \forall\ [sd] \in \mathbf{SD}^c,\ c$$

$$\sum_{[.d]\in\mathbf{D}^c_{[s.]}} \psi^{c*}_{[sd]}(t) = 1 \qquad \forall\ [s.] \in \mathbf{S}^c,\ c$$

$$\phi^{c*}_{o[sd]}(t),\ \phi^{c*}_{\pi[sd]}(t) \geq 0 \qquad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c,\ c$$

$$\psi^{c*}_{[sd]}(t) \geq 0 \qquad \forall\ [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c,\ c$$

The partial derivatives of the cost function $g(t, \mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi})$ with respect to the path tractions $\phi^c_{\pi[sd]}$ can be written with respect to the link flows λ^c_{ij} and node flows λ^c_i:

$$\frac{\partial g_{ij}(t, \mathbf{X}_{ij}, \mathbf{\Phi}, \mathbf{\Psi})}{\partial \phi^c_{\pi[sd]}} = \frac{\partial g_{ij}(t, \mathbf{X}_{ij}, \mathbf{\Lambda}_{ij})}{\partial \lambda^c_{ij}} * \frac{\partial \lambda^c_{ij}}{\partial \phi^c_{\pi[sd]}} =$$

$$= \frac{\partial g_{ij}(t, \mathbf{X}_{ij}, \mathbf{\Lambda}_{ij})}{\partial \lambda^c_{ij}} * (\gamma^c_{[sd]}(t) + \gamma^c_{[s.]}(t) * \psi^c_{[sd]}) * 1_{ij\in\pi[sd]}(t)$$

$$\frac{\partial g_i(t, \mathbf{X}_i, \mathbf{\Phi}, \mathbf{\Psi})}{\partial \phi^c_{\pi[sd]}} = \frac{\partial g_i(t, \mathbf{X}_i, \mathbf{\Lambda}_i)}{\partial \lambda^c_i} * \frac{\partial \lambda^c_i}{\partial \phi^c_{\pi[sd]}} =$$

$$= \frac{\partial g_i(t, \mathbf{X}_i, \mathbf{\Lambda}_i)}{\partial \lambda^c_i} * (\gamma^c_{[sd]}(t) + \gamma^c_{[s.]}(t) * \psi^c_{[sd]}) * 1_{i\in\pi[sd]}(t)$$

$$\frac{\partial g_{[sd]}(t, \mathbf{X}_{o[sd]}, \mathbf{\Phi}, \mathbf{\Psi})}{\partial \phi^c_{o[sd]}} = \frac{\partial g_{[sd]}(t, \mathbf{X}_{o[sd]}, \mathbf{\Lambda}_{o[sd]})}{\partial \lambda^c_{o[sd]}} * \frac{\partial \lambda^c_{o[sd]}}{\partial \phi^c_{o[sd]}} =$$

$$= \frac{\partial g_{[sd]}(t, \mathbf{X}_{o[sd]}, \mathbf{\Lambda}_{o[sd]})}{\partial \lambda^c_{o[sd]}} * (\gamma^c_{[sd]}(t) + \gamma^c_{[s.]}(t) * \psi^c_{[sd]})$$

$$\frac{\partial g_{ij}(t,\mathbf{X}_{ij},\mathbf{\Phi},\mathbf{\Psi})}{\partial \psi^c_{[sd]}} = \frac{\partial g_{ij}(t,\mathbf{X}_{ij},\mathbf{\Lambda}_{ij})}{\partial \lambda^c_{ij}} * \frac{\partial \lambda^c_{ij}}{\partial \psi^c_{[sd]}} =$$

$$= \sum_{\pi[sd]\in\mathbf{\Pi}^c_{[sd]}} \frac{\partial g_{ij}(t,\mathbf{X}_{ij},\mathbf{\Lambda}_{ij})}{\partial \lambda^c_{ij}} * \gamma^c_{[s.]}(t) * \phi^{c*}_{\pi[sd]} * 1_{ij\in\pi[sd]}(t)$$

$$\frac{\partial g_i(t,\mathbf{X}_i,\mathbf{\Phi},\mathbf{\Psi})}{\partial \psi^c_{[sd]}} = \frac{\partial g_i(t,\mathbf{X}_i,\mathbf{\Lambda}_i)}{\partial \lambda^c_i} * \frac{\partial \lambda^c_i}{\partial \psi^c_{[sd]}} =$$

$$= \sum_{\pi[sd]\in\mathbf{\Pi}^c_{[sd]}} \frac{\partial g_i(t,\mathbf{X}_i,\mathbf{\Lambda}_i)}{\partial \lambda^c_i} * \gamma^c_{[s.]}(t) * \phi^{c*}_{\pi[sd]} * 1_{i\in\pi[sd]}(t)$$

$$\frac{\partial g_{[sd]}(t,\mathbf{X}_{o[sd]},\mathbf{\Phi},\mathbf{\Psi})}{\partial \psi^c_{[sd]}} = \frac{\partial g_{[sd]}(t,\mathbf{X}_{o[sd]},\mathbf{\Lambda}_{o[sd]})}{\partial \lambda^c_{o[sd]}} * \frac{\partial \lambda^c_{o[sd]}}{\partial \psi^c_{[sd]}} =$$

$$= \frac{\partial g_{[sd]}(t,\mathbf{X}_{o[sd]},\mathbf{\Lambda}_{o[sd]})}{\partial \lambda^c_{o[sd]}} * \gamma^c_{[s.]}(t) * \phi^{c*}_{o[sd]}$$

$$\frac{\partial g_{[.d]}(t,\mathbf{X}_{[.d]},\mathbf{\Phi},\mathbf{\Psi})}{\partial \psi^c_{[sd]}} = \frac{\partial g_{[.d]}(t,\mathbf{X}_{[.d]},\mathbf{\Lambda}_{[.d]})}{\partial \lambda^c_{[.d]}} * \frac{\partial \lambda^c_{[.d]}}{\partial \psi^c_{[sd]}} =$$

$$= \frac{\partial g_{[.d]}(t,\mathbf{X}_{[.d]},\mathbf{\Lambda}_{[.d]})}{\partial \lambda^c_{[.d]}} * \gamma^c_{[s.]}(t)$$

Then Pontryagin's maximum principle becomes $\forall\ t \in [t_0, t_f]$:

$$\dot{\mathbf{X}}^{k*}_{ij[sd]}(t) = \mathbf{f}^{k}_{ij[sd]}(t, \mathbf{X}^{*}_{ij}(t), \mathbf{\Phi}^{*}(t), \mathbf{\Psi}^{*}(t)) \quad \forall\ ij,\ [sd] \in \mathbf{SD}^{k},\ k$$

$$\dot{\mathbf{X}}^{k*}_{i[sd]}(t) = \mathbf{f}^{k}_{i[sd]}(t, \mathbf{X}^{*}_{i}(t), \mathbf{\Phi}^{*}(t), \mathbf{\Psi}^{*}(t)) \quad \forall\ \subset,\ [sd] \in \mathbf{SD}^{k},\ k$$

$$\dot{\mathbf{X}}^{k*}_{o[sd]}(t) = \mathbf{f}^{k}_{o[sd]}(t, \mathbf{X}^{*}_{o[sd]}(t), \mathbf{\Phi}^{*}(t), \mathbf{\Psi}^{*}(t)) \quad \forall\ [sd] \in \mathbf{SD}^{k},\ k$$

$$\dot{\mathbf{X}}^{k*}_{[.d][sd]}(t) = \mathbf{f}^{k}_{[.d][sd]}(t, \mathbf{X}^{*}_{[.d]}(t), \mathbf{\Phi}^{*}(t), \mathbf{\Psi}^{*}(t)) \quad \forall\ [sd] \in \mathbf{SD}^{k},\ k$$

$$\mathbf{X}^{k*}_{ij[sd]}(t_0) = \mathbf{X}^{k}_{ij[sd],0} \quad \forall\ ij,\ [sd] \in \mathbf{SD}^{k},\ k$$

$$\mathbf{X}^{k*}_{i[sd]}(t_0) = \mathbf{X}^{k}_{i[sd],0} \quad \forall\ i,\ [sd] \in \mathbf{SD}^{k},\ k$$

$$\mathbf{X}^{k*}_{o[sd]}(t_0) = \mathbf{X}^{k}_{o[sd],0} \quad \forall\ [sd] \in \mathbf{SD}^{k},\ k$$

$$\mathbf{X}^{k*}_{[.d][sd]}(t_0) = \mathbf{X}^{k}_{[.d][sd],0} \quad \forall\ [sd] \in \mathbf{SD}^{k},\ k$$

$$\left[\frac{\partial g_{[sd]}(t,\mathbf{X}^{*}_{o[sd]}(t),\mathbf{\Lambda}^{*}_{o[sd]}(t))}{\partial \lambda^{c}_{o[sd]}} * (\gamma^{c}_{[sd]}(t)+\gamma^{c}_{[s.]}(t) * \psi^{c*}_{[sd]}(t))+\right.$$

$$+\sum_{k}\mathbf{P}^{k,c}_{o[sd]} * \nabla_{\phi^{c}_{o[sd]}}\mathbf{f}^{k}_{o[sd]}(t,\mathbf{X}^{*}_{o[sd]}(t),\mathbf{\Phi}^{*}(t),\mathbf{\Psi}^{*}(t))-$$

$$\left.-Q^{c}_{[sd]}(t)\right] * \phi^{c*}_{o[sd]}(t)=0 \qquad \forall\ [sd]\in\mathbf{SD}^{c},\ c$$

$$\left[\sum_{ij}\frac{\partial g_{ij}(t,\mathbf{X}^{*}_{ij}(t),\mathbf{\Lambda}^{*}_{ij}(t))}{\partial \lambda^{c}_{ij}} * (\gamma^{c}_{[sd]}(t)+\gamma^{c}_{[s.]}(t) * \psi^{c*}_{[sd]}(t)) * 1_{ij\in\pi[sd]}(t)+\right.$$

$$+\sum_{i}\frac{\partial g_{i}(t,\mathbf{X}^{*}_{i}(t),\mathbf{\Lambda}^{*}_{i}(t))}{\partial \lambda^{c}_{i}} * (\gamma^{c}_{[sd]}(t)+\gamma^{c}_{[s.]}(t) * \psi^{c*}_{[sd]}(t)) * 1_{i\in\pi[sd]}(t)+$$

$$+\sum_{k}\sum_{[s'd']}\sum_{ij}\mathbf{P}^{k,c}_{ij[s'd']}(t) * \nabla_{\phi^{c}_{\pi[sd]}}\mathbf{f}^{k}_{ij[s'd']}(t,\mathbf{X}^{*}_{ij}(t),\mathbf{\Phi}^{*}(t),\mathbf{\Psi}^{*}(t))+$$

$$+\sum_{k}\sum_{[s'd']}\sum_{i}\mathbf{P}^{k,c}_{i[s'd']}(t) * \nabla_{\phi^{c}_{\pi[sd]}}\mathbf{f}^{c}_{i[s'd']}(t,\mathbf{X}^{*}_{i}(t),\mathbf{\Phi}^{*}(t),\mathbf{\Psi}^{*}(t))-$$

$$\left.-Q^{c}_{[sd]}(t)\right] * \phi^{c*}_{\pi[sd]}(t)=0 \qquad \forall\ \pi[sd]\in\mathbf{\Pi}^{c}_{[sd]},\ [sd]\in\mathbf{SD}^{c},\ c$$

$$\Bigg[\sum_{ij}\sum_{\pi[sd]\in\mathbf{\Pi}^c_{[sd]}}\frac{\partial g_{ij}(t,\mathbf{X}^*_{ij}(t),\mathbf{\Lambda}^*_{ij}(t))}{\partial\lambda^c_{ij}}*\gamma^c_{[s.]}(t)*\phi^{c*}_{\pi[sd]}(t))*1_{ij\in\pi[sd]}(t)+$$

$$+\sum_{i}\sum_{\pi[sd]\in\mathbf{\Pi}^c_{[sd]}}\frac{\partial g_{i}(t,\mathbf{X}^*_{i}(t),\mathbf{\Lambda}^*_{i}(t))}{\partial\lambda^c_{i}}*\gamma^c_{[s.]}(t)*\phi^{c*}_{\pi[sd]}(t))*1_{i\in\pi[sd]}(t)+$$

$$+\frac{\partial g_{[sd]}(t,\mathbf{X}^*_{o[sd]}(t),\mathbf{\Lambda}^*_{o[sd]}(t))}{\partial\lambda^c_{o[sd]}}*\gamma^c_{[s.]}(t)*\phi^{c*}_{o[sd]}(t))+$$

$$+\frac{\partial g_{[.d]}(t,\mathbf{X}^*_{[.d]}(t),\mathbf{\Lambda}^*_{[.d]}(t))}{\partial\lambda^c_{[.d]}}*\gamma^c_{[s.]}(t)+$$

$$+\sum_{k}\sum_{[s'd']}\sum_{ij}\mathbf{P}^{k,c}_{ij[s'd']}(t)*\nabla_{\psi^c_{[sd]}}\mathbf{f}^k_{ij[s'd']}(t,\mathbf{X}^*_{ij}(t),\mathbf{\Phi}^*(t),\mathbf{\Psi}^*(t))+$$

$$+\sum_{k}\sum_{[s'd']}\sum_{i}\mathbf{P}^{k,c}_{i[s'd']}(t)*\nabla_{\psi^c_{[sd]}}\mathbf{f}^k_{i[s'd']}(t,\mathbf{X}^*_{i}(t),\mathbf{\Phi}^*(t),\mathbf{\Psi}^*(t))+$$

$$+\sum_{k}\mathbf{P}^{k,c}_{o[sd]}(t)*\nabla_{\psi^c_{[sd]}}\mathbf{f}^k_{o[sd]}(t,\mathbf{X}^*_{o[sd]}(t),\mathbf{\Phi}^*(t),\mathbf{\Psi}^*(t))+$$

$$+\sum_{k}\sum_{[s'.]}\mathbf{P}^{k,c}_{[.d][s'd]}(t)*\nabla_{\psi^c_{[sd]}}\mathbf{f}^k_{[.d][s'd]}(t,\mathbf{X}^*_{[.d]}(t),\mathbf{\Phi}^*(t),\mathbf{\Psi}^*(t))-$$

$$-Q^c_{[s.]}(t)\Bigg]*\psi^{c*}_{[sd]}(t)=0\qquad\qquad\forall\ [.d]\in\mathbf{D}^c_{[s.]},\ [s.]\in\mathbf{S}^c,\ c$$

$$\frac{\partial g_{[sd]}(t, \mathbf{X}^*_{o[sd]}(t), \mathbf{\Lambda}^*_{o[sd]}(t))}{\partial \lambda^c_{o[sd]}} * (\gamma^c_{[sd]}(t) + \gamma^c_{[s.]}(t) * \psi^{c*}_{[sd]}(t)) +$$

$$+ \sum_k \mathbf{P}^{k,c}_{o[sd]} * \nabla_{\phi^c_{o[sd]}} \mathbf{f}^k_{o[sd]}(t, \mathbf{X}^*_{o[sd]}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-Q^c_{[sd]}(t) \geq 0 \qquad \forall\ [sd] \in \mathbf{SD}^c,\ c$$

$$\sum_{ij} \frac{\partial g_{ij}(t, \mathbf{X}^*_{ij}(t), \mathbf{\Lambda}^*_{ij}(t))}{\partial \lambda^c_{ij}} * (\gamma^c_{[sd]}(t) + \gamma^c_{[s.]}(t) * \psi^{c*}_{[sd]}(t)) * 1_{ij \in \pi[sd]}(t) +$$

$$+ \sum_i \frac{\partial g_i(t, \mathbf{X}^*_i(t), \mathbf{\Lambda}^*_i(t))}{\partial \lambda^c_i} * (\gamma^c_{[sd]}(t) + \gamma^c_{[s.]}(t) * \psi^{c*}_{[sd]}(t)) * 1_{i \in \pi[sd]}(t) +$$

$$+ \sum_k \sum_{[s'd']} \sum_{ij} \mathbf{P}^{k,c}_{ij[s'd']}(t) * \nabla_{\phi^c_{\pi[sd]}} \mathbf{f}^k_{ij[s'd']}(t, \mathbf{X}^*_{ij}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) +$$

$$+ \sum_k \sum_{[s'd']} \sum_i \mathbf{P}^{k,c}_{i[s'd']}(t) * \nabla_{\phi^c_{\pi[sd]}} \mathbf{f}^c_{i[s'd']}(t, \mathbf{X}^*_i(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-Q^c_{[sd]}(t) \geq 0 \qquad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c,\ c$$

$$\sum_{ij} \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \frac{\partial g_{ij}(t, \mathbf{X}^*_{ij}(t), \mathbf{\Lambda}^*_{ij}(t))}{\partial \lambda^c_{ij}} * \gamma^c_{[s.]}(t) * \phi^{c*}_{\pi[sd]}(t)) * 1_{ij \in \pi[sd]}(t) +$$

$$+ \sum_{i} \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \frac{\partial g_i(t, \mathbf{X}^*_i(t), \mathbf{\Lambda}^*_i(t))}{\partial \lambda^c_i} * \gamma^c_{[s.]}(t) * \phi^{c*}_{\pi[sd]}(t)) * 1_{i \in \pi[sd]}(t) +$$

$$+ \frac{\partial g_{[sd]}(t, \mathbf{X}^*_{o[sd]}(t), \mathbf{\Lambda}^*_{o[sd]}(t))}{\partial \lambda^c_{o[sd]}} * \gamma^c_{[s.]}(t) * \phi^{c*}_{o[sd]}(t)) +$$

$$+ \frac{\partial g_{[.d]}(t, \mathbf{X}^*_{[.d]}(t), \mathbf{\Lambda}^*_{[.d]}(t))}{\partial \lambda^c_{[.d]}} * \gamma^c_{[s.]}(t) +$$

$$+ \sum_{k} \sum_{[s'd']} \sum_{ij} \mathbf{P}^{k,c}_{ij[s'd']}(t) * \nabla_{\psi^c_{[sd]}} \mathbf{f}^k_{ij[s'd']}(t, \mathbf{X}^*_{ij}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) +$$

$$+ \sum_{k} \sum_{[s'd']} \sum_{i} \mathbf{P}^{k,c}_{i[s'd']}(t) * \nabla_{\psi^c_{[sd]}} \mathbf{f}^k_{i[s'd']}(t, \mathbf{X}^*_{i}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) +$$

$$+ \sum_{k} \mathbf{P}^{k,c}_{o[sd]}(t) * \nabla_{\psi^c_{[sd]}} \mathbf{f}^k_{o[sd]}(t, \mathbf{X}^*_{o[sd]}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) +$$

$$+ \sum_{k} \sum_{[s'.]} \mathbf{P}^{k,c}_{[.d][s'd]}(t) * \nabla_{\psi^c_{[sd]}} \mathbf{f}^k_{[.d][s'd]}(t, \mathbf{X}^*_{[.d]}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-Q^c_{[s.]}(t) \geq 0 \qquad \forall\ [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c,\ c$$

$$\dot{\mathbf{P}}^{c,k}_{ij[sd]}(t) = -\nabla_{\mathbf{X}^k_{ij[sd]}} g_{ij}(t, \mathbf{X}^*_{ij}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-\sum_n \sum_{[s'd']} \mathbf{P}^{c,n}_{ij[s'd']}(t) * \nabla_{\mathbf{X}^k_{ij[sd]}} \mathbf{f}^n_{ij[s'd']}(t, \mathbf{X}^*_{ij}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\forall\ ij,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{P}}^{c,k}_{i[sd]}(t) = -\nabla_{\mathbf{X}^k_{i[sd]}} g_i(t, \mathbf{X}^*_i, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-\sum_n \sum_{[s'd']} \mathbf{P}^{c,n}_{i[s'd']}(t) * \nabla_{\mathbf{X}^k_{i[sd]}} \mathbf{f}^n_{i[s'd']}(t, \mathbf{X}^*_i, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\forall\ i,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{P}}^{c,k}_{o[sd]}(t) = -\nabla_{\mathbf{X}^k_{o[sd]}} g_{[sd]}(t, \mathbf{X}^*_{o[sd]}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-\sum_n \mathbf{P}^{c,n}_{o[sd]}(t) * \nabla_{\mathbf{X}^k_{o[sd]}} \mathbf{f}^n_{o[sd]}(t, \mathbf{X}^*_{o[sd]}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{P}}^{c,k}_{[.d][sd]}(t) = -\nabla_{\mathbf{X}^k_{[.d][sd]}} g_{[.d]}(t, \mathbf{X}^*_{[.d]}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-\sum_n \sum_{[s'.]} \mathbf{P}^{c,n}_{[.d][sd]}(t) * \nabla_{\mathbf{X}^k_{[.d][sd]}} \mathbf{f}^n_{[.d][s'd]}(t, \mathbf{X}^*_{[.d]}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\phi^{c*}_{o[sd]}(t) + \sum_{\pi[sd]\in\mathbf{\Pi}^c_{[sd]}} \phi^{c*}_{\pi[sd]}(t) = 1 \quad \forall\ [sd] \in \mathbf{SD}^c,\ c$$

$$\sum_{[.d]\in\mathbf{D}^c_{[s.]}} \psi^{c*}_{[sd]}(t) = 1 \qquad \forall\ [s.] \in \mathbf{S}^c,\ c$$

$$\phi^{c*}_{o[sd]}(t),\ \phi^{c*}_{\pi[sd]}(t) \geq 0 \qquad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c,\ c$$

$$\psi^{c*}_{[sd]}(t) \geq 0 \qquad \forall\ [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c,\ c$$

Next, for each class c, we define the length for the rejected flow $[sd]$, the length for the path $\pi[sd]$ and the length for the source-destination pair $[sd]$:

$$l^{c,team}_{o[sd]}(t) = \frac{\partial g_{[sd]}(t, \mathbf{X}_{o[sd]}(t), \mathbf{\Lambda}_{o[sd]}(t))}{\partial \lambda^c_{o[sd]}} * (\gamma^c_{[sd]}(t) + \gamma^c_{[s.]}(t) * \psi^c_{[sd]}(t)) +$$

$$+ \sum_k \mathbf{P}^{k,c}_{o[sd]} * \nabla_{\phi^c_{o[sd]}} \mathbf{f}^k_{o[sd]}(t, \mathbf{X}_{o[sd]}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t))$$

$$\forall\ [sd] \in \mathbf{SD}^c,\ c$$

$$l^{c,team}_{\pi[sd]}(t) = \sum_{ij} \frac{\partial g_{ij}(t, \mathbf{X}_{ij}(t), \mathbf{\Lambda}_{ij}(t))}{\partial \lambda^c_{ij}} * (\gamma^c_{[sd]}(t) + \gamma^c_{[s.]}(t) * \psi^c_{[sd]}(t)) * 1_{ij \in \pi[sd]}(t) +$$

$$+ \sum_{i} \frac{\partial g_{i}(t, \mathbf{X}_{i}(t), \mathbf{\Lambda}_{i}(t))}{\partial \lambda^c_{i}} * (\gamma^c_{[sd]}(t) + \gamma^c_{[s.]}(t) * \psi^c_{[sd]}(t)) * 1_{i \in \pi[sd]}(t) +$$

$$+ \sum_{k} \sum_{[s'd']} \sum_{ij} \mathbf{P}^{k,c}_{ij[s'd']}(t) * \nabla_{\phi^c_{\pi[sd]}} \mathbf{f}^k_{ij[s'd']}(t, \mathbf{X}_{ij}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t)) +$$

$$+ \sum_{k} \sum_{[s'd']} \sum_{i} \mathbf{P}^{k,c}_{i[s'd']}(t) * \nabla_{\phi^c_{\pi[sd]}} \mathbf{f}^c_{i[s'd']}(t, \mathbf{X}_{i}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t))$$

$$\forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c,\ c$$

$$l^{c,team}_{[sd]}(t) = \sum_{ij} \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \frac{\partial g_{ij}(t, \mathbf{X}_{ij}(t), \mathbf{\Lambda}_{ij}(t))}{\partial \lambda^c_{ij}} * \gamma^c_{[s.]}(t) * \phi^c_{\pi[sd]}(t)) * 1_{ij \in \pi[sd]}(t) +$$

$$+ \sum_{i} \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \frac{\partial g_{i}(t, \mathbf{X}_{i}(t), \mathbf{\Lambda}_{i}(t))}{\partial \lambda^c_{i}} * \gamma^c_{[s.]}(t) * \phi^c_{\pi[sd]}(t)) * 1_{i \in \pi[sd]}(t) +$$

$$+ \frac{\partial g_{[sd]}(t, \mathbf{X}_{o[sd]}(t), \mathbf{\Lambda}_{o[sd]}(t))}{\partial \lambda^c_{o[sd]}} * \gamma^c_{[s.]}(t) * \phi^c_{o[sd]}(t)) +$$

$$+ \frac{\partial g_{[.d]}(t, \mathbf{X}_{[.d]}(t), \mathbf{\Lambda}_{[.d]}(t))}{\partial \lambda^c_{[.d]}} * \gamma^c_{[s.]}(t) +$$

$$+ \sum_{k} \sum_{[s'd']} \sum_{ij} \mathbf{P}^{k,c}_{ij[s'd']}(t) * \nabla_{\psi^c_{[sd]}} \mathbf{f}^k_{ij[s'd']}(t, \mathbf{X}_{ij}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t)) +$$

$$+ \sum_{k} \sum_{[s'd']} \sum_{i} \mathbf{P}^{k,c}_{i[s'd']}(t) * \nabla_{\psi^c_{[sd]}} \mathbf{f}^k_{i[s'd']}(t, \mathbf{X}_{i}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t)) +$$

$$+ \sum_{k} \mathbf{P}^{k,c}_{o[sd]}(t) * \nabla_{\psi^c_{[sd]}} \mathbf{f}^k_{o[sd]}(t, \mathbf{X}_{o[sd]}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t)) +$$

$$+ \sum_{k} \sum_{[s'.]} \mathbf{P}^{k,c}_{[.d][s'd]}(t) * \nabla_{\psi^c_{[sd]}} \mathbf{f}^k_{[.d][s'd]}(t, \mathbf{X}_{[.d]}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t)) -$$

$$\forall\ [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c,\ c$$

External arriving flow at a source is assigned to the destination that has the minimum length from the source. However, this flow may be rejected if the length of rejecting it is less than the lengths of the paths to its destination. If it is accepted, then it is routed to its destination via the minimum length path.

In the next section, we will derive the same conditions by an alternative way, and we shall state the above ideas more formally.

5.1.6 V.I. for Separable Cost Functions

Equivalently, the team optimal solution definition, each class c minimizes its cost function g given the optimum decisions of all other classes. We first solve the routing and congestion control problem assuming that all other classes act optimally for themselves. So, class c first solves the routing and congestion control problems

$$
\begin{aligned}
\textit{minimize} \qquad & \int_{t_0}^{t_f} g(t, \mathbf{X}, \mathbf{\Phi}(t), \mathbf{\Psi}(t))dt = \\
& = \sum_{ij} \int_{t_0}^{t_f} g_{ij}(t, \mathbf{X}_{ij}(t), \mathbf{\Lambda}_{ij}(t))dt \quad + \\
& + \sum_{i} \int_{t_0}^{t_f} g_{i}(t, \mathbf{X}_{i}(t), \mathbf{\Lambda}_{i}(t))dt \quad + \\
& + \sum_{[sd]} \int_{t_0}^{t_f} g_{[sd]}(t, \mathbf{X}_{o[sd]}(t), \mathbf{\Lambda}_{o[sd]}(t))dt \quad + \\
& + \sum_{[.d]} \int_{t_0}^{t_f} g_{[.d]}(t, \mathbf{X}_{[.d]}(t), \mathbf{\Lambda}_{[.d]}(t))dt
\end{aligned}
$$

$$
\textit{with respect to} \qquad \mathbf{\Phi}(t)
$$

such that

$$\dot{\mathbf{X}}^k_{ij[sd]}(t) = \mathbf{f}^k_{ij[sd]}(t, \mathbf{X}_{ij}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t)) \quad \forall\ ij,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{X}}^k_{i[sd]}(t) = \mathbf{f}^k_{i[sd]}(t, \mathbf{X}_{i}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t)) \quad \forall\ i,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{X}}^k_{o[sd]}(t) = \mathbf{f}^k_{o[sd]}(t, \mathbf{X}_{o[sd]}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t)) \quad \forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{X}}^k_{[.d][sd]}(t) = \mathbf{f}^k_{[.d][sd]}(t, \mathbf{X}_{[.d]}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t)) \quad \forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\mathbf{X}^k_{ij[sd]}(t_0) = \mathbf{X}^k_{ij[sd],0} \quad \forall\ ij,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\mathbf{X}^k_{i[sd]}(t_0) = \mathbf{X}^k_{i[sd],0} \quad \forall\ i,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\mathbf{X}^k_{o[sd]}(t_0) = \mathbf{X}^k_{o[sd],0} \quad \forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\mathbf{X}^k_{[.d][sd]}(t_0) = \mathbf{X}^k_{[.d][sd],0} \quad \forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\phi^c_{o[sd]}(t) + \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \phi^c_{\pi[sd]}(t) = 1 \quad \forall\ [sd] \in \mathbf{SD}^c$$

$$\phi^c_{o[sd]}(t),\ \phi^c_{\pi[sd]}(t) \geq 0 \quad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c$$

The necessary optimality conditions are $\forall\ t \in [t_0, t_f]$:

$$\sum \sum_{[sd] \in \mathbf{SD}^c} \left\{ \frac{\partial g(t, \mathbf{X}^*(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))}{\partial \phi^c_{o[sd]} * (\phi^c_{o[sd]} - \phi^{c*}_{o[sd]}(t)) \ +} \right.$$

$$\left. + \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \frac{\partial g(t, \mathbf{X}^*(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))}{\partial \phi^c_{\pi[sd]} * (\phi^c_{\pi[sd]} - \phi^{c*}_{\pi[sd]}(t))} \right\} \geq 0 \qquad \forall\ \mathbf{\Phi}^c \in \mathbf{RC}^c$$

such that

$$\dot{\mathbf{X}}^{k*}_{ij[sd]}(t) = \mathbf{f}^k_{ij[sd]}(t, \mathbf{X}^*_{ij}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) \qquad \forall\ ij,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{X}}^{k*}_{i[sd]}(t) = \mathbf{f}^k_{i[sd]}(t, \mathbf{X}^*_{i}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) \qquad \forall\ i,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{X}}^{k*}_{o[sd]}(t) = \mathbf{f}^k_{o[sd]}(t, \mathbf{X}^*_{o[sd]}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) \qquad \forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{X}}^{k*}_{[.d][sd]}(t) = \mathbf{f}^k_{[.d][sd]}(t, \mathbf{X}^*_{[.d]}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) \qquad \forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\mathbf{X}^{k*}_{ij[sd]}(t_0) = \mathbf{X}^k_{ij[sd],0} \qquad \forall\ ij,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\mathbf{X}^{k*}_{i[sd]}(t_0) = \mathbf{X}^k_{i[sd],0} \qquad \forall\ i,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\mathbf{X}^{k*}_{o[sd]}(t_0) = \mathbf{X}^k_{o[sd],0} \qquad \forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\mathbf{X}^{k*}_{[.d][sd]}(t_0) = \mathbf{X}^k_{[.d][sd],0} \qquad \forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{P}}^{c,k}_{ij[sd]}(t) = -\nabla_{\mathbf{X}^k_{ij[sd]}} g_{ij}(t, \mathbf{X}^*_{ij}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-\sum_n \sum_{[s'd']} \nabla_{\mathbf{X}^k_{ij[sd]}} \mathbf{P}^{c,n}_{ij[s'd']}(t) * \mathbf{f}^n_{ij[s'd']}(t, \mathbf{X}^*_{ij}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\forall\ ij,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{P}}^{c,k}_{i[sd]}(t) = -\nabla_{\mathbf{X}^k_{i[sd]}} g_i(t, \mathbf{X}^*_i, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-\sum_n \sum_{[s'd']} \nabla_{\mathbf{X}^k_{i[sd]}} \mathbf{P}^{c,n}_{i[s'd']}(t) * \mathbf{f}^n_{i[s'd']}(t, \mathbf{X}^*_i, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\forall\ i,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{P}}^{c,k}_{o[sd]}(t) = -\nabla_{\mathbf{X}^k_{o[sd]}} g_{[sd]}(t, \mathbf{X}^*_{o[sd]}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-\sum_n \nabla_{\mathbf{X}^k_{o[sd]}} \mathbf{P}^{c,n}_{o[sd]}(t) * \mathbf{f}^n_{o[sd]}(t, \mathbf{X}^*_o, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{P}}^{c,k}_{[.d][sd]}(t) = -\nabla_{\mathbf{X}^k_{[.d][sd]}} g_{[.d]}(t, \mathbf{X}^*_{[.d]}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-\sum_n \sum_{[s'.]} \nabla_{\mathbf{X}^k_{[.d][sd]}} \mathbf{P}^{c,n}_{[.d][sd]}(t) * \mathbf{f}^n_{[.d][s'd]}(t, \mathbf{X}^*_{[.d]}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\phi^{c*}_{o[sd]}(t) + \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \phi^{c*}_{\pi[sd]}(t) = 1 \quad \forall\ [sd] \in \mathbf{SD}^c$$

$$\phi^{c*}_{o[sd]}(t),\ \phi^{c*}_{\pi[sd]}(t) \geq 0 \qquad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c$$

We can decompose these conditions for each source-destination pair $[sd] \in \mathbf{SD}^c$ $\forall\, t \in [t_0, t_f]$:

$$\frac{\partial g(t, \mathbf{X}^*(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))}{\partial \phi^c_{o[sd]}} * (\phi^c_{o[sd]} - \phi^{c*}_{o[sd]}(t)) +$$

$$+ \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \frac{\partial g(t, \mathbf{X}^*(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))}{\partial \phi^c_{\pi[sd]}} * (\phi^c_{\pi[sd]} - \phi^{c*}_{\pi[sd]}(t)) \geq 0 \qquad \forall\, \mathbf{\Phi}^c \in \mathbf{RC}^c$$

such that

$$\begin{aligned}
\dot{\mathbf{X}}^{k*}_{ij[sd]}(t) &= \mathbf{f}^k_{ij[sd]}(t, \mathbf{X}^*_{ij}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) && \forall\, ij,\ [sd] \in \mathbf{SD}^k,\ k \\
\dot{\mathbf{X}}^{k*}_{i[sd]}(t) &= \mathbf{f}^k_{i[sd]}(t, \mathbf{X}^*_{i}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) && \forall\, i,\ [sd] \in \mathbf{SD}^k,\ k \\
\dot{\mathbf{X}}^{k*}_{o[sd]}(t) &= \mathbf{f}^k_{o[sd]}(t, \mathbf{X}^*_{o[sd]}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) && \forall\, [sd] \in \mathbf{SD}^k,\ k \\
\dot{\mathbf{X}}^{k*}_{[.d][sd]}(t) &= \mathbf{f}^k_{[.d][sd]}(t, \mathbf{X}^*_{[.d]}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) && \forall\, [sd] \in \mathbf{SD}^k,\ k \\
\mathbf{X}^{k*}_{ij[sd]}(t_0) &= \mathbf{X}^k_{ij[sd],0} && \forall\, ij,\ [sd] \in \mathbf{SD}^k,\ k \\
\mathbf{X}^{k*}_{i[sd]}(t_0) &= \mathbf{X}^k_{i[sd],0} && \forall\, i,\ [sd] \in \mathbf{SD}^k,\ k \\
\mathbf{X}^{k*}_{o[sd]}(t_0) &= \mathbf{X}^k_{o[sd],0} && \forall\, [sd] \in \mathbf{SD}^k,\ k \\
\mathbf{X}^{k*}_{[.d][sd]}(t_0) &= \mathbf{X}^k_{[.d][sd],0} && \forall\, [sd] \in \mathbf{SD}^k,\ k
\end{aligned}$$

$$\dot{\mathbf{P}}^{c,k}_{ij[sd]}(t) = -\nabla_{\mathbf{X}^k_{ij[sd]}} g_{ij}(t, \mathbf{X}^*_{ij}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-\sum_n \sum_{[s'd']} \nabla_{\mathbf{X}^k_{ij[sd]}} \mathbf{P}^{c,n}_{ij[s'd']}(t) * \mathbf{f}^n_{ij[s'd']}(t, \mathbf{X}^*_{ij}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\forall\ [sd] \in \mathbf{SD}^k,\ ij,\ k$$

$$\dot{\mathbf{P}}^{c,k}_{i[sd]}(t) = -\nabla_{\mathbf{X}^k_{i[sd]}} g_i(t, \mathbf{X}^*_i, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-\sum_n \sum_{[s'd']} \nabla_{\mathbf{X}^k_{i[sd]}} \mathbf{P}^{c,n}_{i[s'd']}(t) * \mathbf{f}^n_{i[s'd']}(t, \mathbf{X}^*_i, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\forall\ i,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{P}}^{c,k}_{o[sd]}(t) = -\nabla_{\mathbf{X}^k_{o[sd]}} g_{[sd]}(t, \mathbf{X}^*_{o[sd]}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-\sum_n \nabla_{\mathbf{X}^k_{o[sd]}} \mathbf{P}^{c,n}_{o[sd]}(t) * \mathbf{f}^n_{o[sd]}(t, \mathbf{X}^*_o, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{P}}^{c,k}_{[.d][sd]}(t) = -\nabla_{\mathbf{X}^k_{[.d][sd]}} g_{[.d]}(t, \mathbf{X}^*_{[.d]}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-\sum_n \sum_{[s'.]} \nabla_{\mathbf{X}^k_{[.d][sd]}} \mathbf{P}^{c,n}_{[.d][sd]}(t) * \mathbf{f}^n_{[.d][s'd]}(t, \mathbf{X}^*_{[.d]}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\phi^{c*}_{o[sd]}(t) + \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \phi^{c*}_{\pi[sd]} = 1$$

$$\phi^c_{o[sd]}(t),\ \phi^c_{\pi[sd]}(t) \geq 0 \qquad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]}$$

Theorem : *Routing*

$$\phi^{c*}_{\pi[sd]}(t) > 0 \quad only \quad if \qquad l^{c,team*}_{\pi[sd]}(t) = \min\{l^{c,team*}_{o[sd]}(t), \min_{p[sd]}\{l^{c,team*}_{p[sd]}(t)\}\}$$

$$\phi^{c*}_{\pi[sd]}(t) = 0 \qquad o.w.$$

$$\phi^{c*}_{o[sd]}(t) + \sum_{\pi[sd]\in\mathbf{\Pi}^c_{[sd]}} \phi^{c*}_{\pi[sd]}(t) = 1$$

$$\forall\ \pi[sd] \in \Pi^c_{[sd]}, \qquad [sd] \in \mathbf{SD}^c, \quad c$$

and satisfies the partial differential vectors for the state and the costate variables.

Theorem : *Congestion Control*

Flow is not admitted into the network only if its rejection length is less than the minimum length path to its destination:

$$\phi^{c*}_{o[sd]}(t) > 0 \quad only \quad if \qquad l^{c,team*}_{o[sd]}(t) = \min\{l^{c,team*}_{o[sd]}(t), \min_{p[sd]}\{l^{c,team*}_{p[sd]}(t)\}\}$$

$$\phi^{c*}_{\pi[sd]} = 0 \qquad o.w.$$

$$\phi^{c*}_{o[sd]}(t) + \sum_{\pi[sd]\in\mathbf{\Pi}^c_{[sd]}} \phi^{c*}_{\pi[sd]}(t) = 1 \quad \forall\ [sd] \in \mathbf{SD}^c, \quad c$$

and satisfies the partial differential vectors for the state and the costate variables.

Having found the optimum routing and congestion control decisions, we proceed to solve the load sharing problem for class c assuming also that all other classes act at their optimum decisions. So, the load sharing problem for class c is

$$minimize \qquad \int_{t_0}^{tf} g(t, \mathbf{X}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}(t))dt =$$

$$= \sum_{ij} \int_{t_0}^{tf} g_{ij}(t, \mathbf{X}_{ij}(t), \mathbf{\Lambda}_{ij}(t))dt \; +$$

$$+ \sum_{i} \int_{t_0}^{tf} g_{i}(t, \mathbf{X}_{i}(t), \mathbf{\Lambda}_{i}(t))dt \; +$$

$$+ \sum_{[sd]} \int_{t_0}^{tf} g_{[sd]}(t, \mathbf{X}_{o[sd]}(t), \mathbf{\Lambda}_{o[sd]}(t))dt \; +$$

$$+ \sum_{[.d]} \int_{t_0}^{tf} g_{[.d]}(t, \mathbf{X}_{[.d]}(t), \mathbf{\Lambda}_{[.d]}(t))dt$$

with respect to $\mathbf{\Psi}(t)$

such that

$$\dot{\mathbf{X}}^k_{ij[sd]}(t) = \mathbf{f}^k_{ij[sd]}(t, \mathbf{X}_{ij}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t)) \quad \forall\, ij,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{X}}^k_{i[sd]}(t) = \mathbf{f}^k_{i[sd]}(t, \mathbf{X}_{i}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t)) \quad \forall\, i,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{X}}^k_{o[sd]}(t) = \mathbf{f}^k_{o[sd]}(t, \mathbf{X}_{o[sd]}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t)) \quad \forall\, [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{X}}^k_{[.d][sd]}(t) = \mathbf{f}^k_{[.d][sd]}(t, \mathbf{X}_{[.d]}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t)) \quad \forall\, [sd] \in \mathbf{SD}^k,\ k$$

$$\mathbf{X}^k_{ij[sd]}(t_0) = \mathbf{X}^k_{ij[sd],0} \quad \forall\, ij,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\mathbf{X}^k_{i[sd]}(t_0) = \mathbf{X}^k_{i[sd],0} \quad \forall\, i,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\mathbf{X}^k_{o[sd]}(t_0) = \mathbf{X}^k_{o[sd],0} \quad \forall\, [sd] \in \mathbf{SD}^k,\ k$$

$$\mathbf{X}^k_{[.d][sd]}(t_0) = \mathbf{X}^k_{[.d][sd],0} \quad \forall\, [sd] \in \mathbf{SD}^k,\ k$$

$$\sum_{[.d] \in \mathbf{D}^c_{[s.]}} \psi^c_{[sd]}(t) = 1 \quad \forall\, [s.] \in \mathbf{S}^c$$

$$\psi^c_{[sd]}(t) \geq 0 \quad \forall\, [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c$$

The necessary and sufficient optimality conditions are $\forall\ t \in [t_0, t_f]$:

$$\sum_c \frac{\partial g(t, \mathbf{X}^*(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))}{\partial \psi^c_{[sd]}} * (\psi^c_{[sd]} - \psi^{c*}_{[sd]}(t)) \quad \geq 0 \qquad \forall\ \mathbf{\Psi} \in \mathbf{LS}$$

such

$$\dot{\mathbf{X}}^{k*}_{ij[sd]}(t) = \mathbf{f}^k_{ij[sd]}(t, \mathbf{X}^*_{ij}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) \qquad \forall\ ij,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{X}}^{k*}_{i[sd]}(t) = \mathbf{f}^k_{i[sd]}(t, \mathbf{X}^*_{i}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) \qquad \forall\ i,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{X}}^{k*}_{o[sd]}(t) = \mathbf{f}^k_{o[sd]}(t, \mathbf{X}^*_{o[sd]}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) \qquad \forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{X}}^{k*}_{[.d][sd]}(t) = \mathbf{f}^k_{[.d][sd]}(t, \mathbf{X}^*_{[.d]}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) \qquad \forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\mathbf{X}^{k*}_{ij[sd]}(t_0) = \mathbf{X}^k_{ij[sd],0} \qquad \forall\ ij,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\mathbf{X}^{k*}_{i[sd]}(t_0) = \mathbf{X}^k_{i[sd],0} \qquad \forall\ i,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\mathbf{X}^{k*}_{o[sd]}(t_0) = \mathbf{X}^k_{o[sd],0} \qquad \forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\mathbf{X}^{k*}_{[.d][sd]}(t_0) = \mathbf{X}^k_{[.d][sd],0} \qquad \forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{P}}^{c,k}_{ij[sd]}(t) = -\nabla_{\mathbf{X}^k_{ij[sd]}} g_{ij}(t, \mathbf{X}^*_{ij}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-\sum_n \sum_{[s'd']} \nabla_{\mathbf{X}^k_{ij[sd]}} \mathbf{P}^{c,n}_{ij[s'd']}(t) * \mathbf{f}^n_{ij[s'd']}(t, \mathbf{X}^*_{ij}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\forall\ ij,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{P}}^{c,k}_{i[sd]}(t) = -\nabla_{\mathbf{X}^k_{i[sd]}} g_i(t, \mathbf{X}^*_i, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-\sum_n \sum_{[s'd']} \nabla_{\mathbf{X}^k_{i[sd]}} \mathbf{P}^{c,n}_{i[s'd']}(t) * \mathbf{f}^n_{i[s'd']}(t, \mathbf{X}^*_i, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\forall\ i,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{P}}^{c,k}_{o[sd]}(t) = -\nabla_{\mathbf{X}^k_{o[sd]}} g_{[sd]}(t, \mathbf{X}^*_{o[sd]}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-\sum_n \nabla_{\mathbf{X}^k_{o[sd]}} \mathbf{P}^{c,n}_{o[sd]}(t) * \mathbf{f}^n_{o[sd]}(t, \mathbf{X}^*_o, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{P}}^{c,k}_{[.d][sd]}(t) = -\nabla_{\mathbf{X}^k_{[.d][sd]}} g_{[.d]}(t, \mathbf{X}^*_{[.d]}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-\sum_n \sum_{[s'.]} \nabla_{\mathbf{X}^k_{[.d][sd]}} \mathbf{P}^{c,n}_{[.d][sd]}(t) * \mathbf{f}^n_{[.d][s'd]}(t, \mathbf{X}^*_{[.d]}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\sum_{[.d] \in \mathbf{D}^c_{[s.]}} \psi^{c*}_{[sd]}(t) = 1 \quad \forall\ [s.] \in \mathbf{S}^c$$

$$\psi^{c*}_{[sd]}(t) \geq 0 \qquad \forall\ [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c$$

We can decompose these conditions for each source node $[s.] \in \mathbf{S}^c \;\; \forall\, t \in [t_0, t_f]$:

$$\sum_{[.d] \in \mathbf{D}^c_{[s.]}} \frac{\partial g(t, \mathbf{X}^*(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))}{\partial \psi^c_{[sd]}} * (\psi^c_{[sd]} - \psi^{c*}_{[sd]}(t)) \quad \geq 0 \qquad \forall\, \mathbf{\Psi}^c \in \mathbf{LS}^c$$

such that

$$\begin{aligned}
\dot{\mathbf{X}}^{k*}_{ij[sd]}(t) &= \mathbf{f}^k_{ij[sd]}(t, \mathbf{X}^*_{ij}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) && \forall\, ij,\ [sd] \in \mathbf{SD}^k,\ k \\
\dot{\mathbf{X}}^{k*}_{i[sd]}(t) &= \mathbf{f}^k_{i[sd]}(t, \mathbf{X}^*_{i}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) && \forall\, i,\ [sd] \in \mathbf{SD}^k,\ k \\
\dot{\mathbf{X}}^{k*}_{o[sd]}(t) &= \mathbf{f}^k_{o[sd]}(t, \mathbf{X}^*_{o[sd]}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) && \forall\, [sd] \in \mathbf{SD}^k,\ k \\
\dot{\mathbf{X}}^{k*}_{[.d][sd]}(t) &= \mathbf{f}^k_{[.d][sd]}(t, \mathbf{X}^*_{[.d]}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) && \forall\, [sd] \in \mathbf{SD}^k,\ k \\
\mathbf{X}^{k*}_{ij[sd]}(t_0) &= \mathbf{X}^k_{ij[sd],0} && \forall\, ij,\ [sd] \in \mathbf{SD}^k,\ k \\
\mathbf{X}^{k*}_{i[sd]}(t_0) &= \mathbf{X}^k_{i[sd],0} && \forall\, i,\ [sd] \in \mathbf{SD}^k,\ k \\
\mathbf{X}^{k*}_{o[sd]}(t_0) &= \mathbf{X}^k_{o[sd],0} && \forall\, [sd] \in \mathbf{SD}^k,\ k \\
\mathbf{X}^{k*}_{[.d][sd]}(t_0) &= \mathbf{X}^k_{[.d][sd],0} && \forall\, [sd] \in \mathbf{SD}^k,\ k
\end{aligned}$$

$$\dot{\mathbf{P}}^{c,k}_{ij[sd]}(t) = -\nabla_{\mathbf{X}^k_{ij[sd]}} g_{ij}(t, \mathbf{X}^*_{ij}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-\sum_n \sum_{[s'd']} \nabla_{\mathbf{X}^k_{ij[sd]}} \mathbf{P}^{c,n}_{ij[s'd']}(t) * \mathbf{f}^n_{ij[s'd']}(t, \mathbf{X}^*_{ij}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\forall\ ij,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{P}}^{c,k}_{i[sd]}(t) = -\nabla_{\mathbf{X}^k_{i[sd]}} g_i(t, \mathbf{X}^*_i, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-\sum_n \sum_{[s'd']} \nabla_{\mathbf{X}^k_{i[sd]}} \mathbf{P}^{c,n}_{i[s'd']}(t) * \mathbf{f}^n_{i[s'd']}(t, \mathbf{X}^*_i, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\forall\ i,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{P}}^{c,k}_{o[sd]}(t) = -\nabla_{\mathbf{X}^k_{o[sd]}} g_{[sd]}(t, \mathbf{X}^*_{o[sd]}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-\sum_n \nabla_{\mathbf{X}^k_{o[sd]}} \mathbf{P}^{c,n}_{o[sd]}(t) * \mathbf{f}^n_{o[sd]}(t, \mathbf{X}^*_o, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{P}}^{c,k}_{[.d][sd]}(t) = -\nabla_{\mathbf{X}^k_{[.d][sd]}} g_{[.d]}(t, \mathbf{X}^*_{[.d]}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-\sum_n \sum_{[s'.]} \nabla_{\mathbf{X}^k_{[.d][sd]}} \mathbf{P}^{c,n}_{[.d][sd]}(t) * \mathbf{f}^n_{[.d][s'd]}(t, \mathbf{X}^*_{[.d]}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\sum_{.d] \in \mathbf{D}^c_{[s.]}} \psi^{c*}_{[sd]}(t) = 1 \quad \forall\ [s.] \in \mathbf{S}^c$$

$$\psi^{c*}_{[sd]}(t) \geq 0 \qquad \forall\ [.d] \in \mathbf{D}^c_{[s.]}$$

Theorem : *Load Sharing*

For each source, there must be flow only to destinations whose length is minimum:

$$\psi^{c*}_{[sd]}(t) > 0 \quad only \quad if \qquad l^{c,team*}_{[sd]}(t) = \min_{[sd']}\{l^{c,team*}_{[sd']}(t)\}$$

$$\psi^{c*}_{[sd]}(t) = 0 \qquad\qquad o.w.$$

$$\sum_{[.d]\in \mathbf{S}^c_{[s.]}} \psi^{c*}_{[sd]}(t) = 1 \quad \forall\ [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c,\ c$$

and satisfies the partial differential vectors for the state and the costate variables.

Thus, in this section we have formulated and solved the load sharing, routing and congestion control problem as a team problem among multiple cooperative classes.

5.2 Nash Equilibrium Solution

In this section, we formulate the dynamic join load sharing, routing and congestion control problem on the path flow space as a non-cooperative dynamic Nash game among competing classes.

Customers of each class try to use the resources of the distributed system for their own benefit, ignoring the inconvenience that they cause to customers from other classes. Since the behavior of each class is similar to that of any other class, i.e. to operate optimally for its customers, next we consider customers only from class c, and the effect of customers from other classes on them.

After the static non-cooperative games by Nash [347], dynamic non-cooperative games have been investigated and are presented in books by: Isaacs [231] Blaquiete, Gerard & Leitmann [55] Friedman [173] Case [87], Rosenmuller [404], Mehlmann [328], Krasovskii & Subbotin [265] among others.

Next, we briefly review research on dynamic Nash games:

Berkovitz [34] obtains necessary conditions for zero-sum differential games. Sarma, Ragade & Prasad [425] introduce dynamic n-person noncooperative dynamic games and provide necessary conditions. Case [88] provides sufficient conditions and use dynamic programming arguments. Stalford & Leitmann [459] discuss sufficiency conditions for dynamic Nash games.

Sandell [417] proves that for deterministic nonzero-sum games any open-loop Nash strategy is also a closed-loop strategy. Williams [513] obtains sufficient conditions for the existence of Nash equilibrium and proves that a class of linear-quadratic differential games have equilibrium point when the duration of the game is sufficiently small.

Papavassilopoulos [374] proves existence and uniqueness of the solution for discrete-time linear-quadratic Gaussian Nash games with one-step delay observation sharing pattern. The solution is also linear in the information. Tu & Papavassilopoulos [501] consider discrete-time linear-quadratic Gaussian Nash games. They show that better information is beneficial to all players if the number of stages of the game, or the number of players, is larger than some bounds. For two-person

zero-sum games, better information is beneficial to the player who has better maneuverability. Basar & Li [26] derive conditions for existence and uniqueness for stochastic linear-quadratic differential games. They also provide an algorithm for an iterative distributed computation of the solution.

When the classes are in equilibrium, no class can decrease its cost by altering its decision unilaterally. Next, we give the definition for a Nash equilibrium [27], for the join load sharing, routing and congestion control problem on the path flows.

Definition:

A vector $(\mathbf{\Phi^*}, \mathbf{\Psi^*}) \in (\mathbf{RC}, \mathbf{LS})$ *is called a* Nash equilibrium *for a* C*-class join load sharing, routing and congestion control problem if and only if*

$$J^1\begin{pmatrix} \mathbf{\Phi}^{1*}, ..., \mathbf{\Phi}^{c*}, ..., \mathbf{\Phi}^{C*} \\ \mathbf{\Psi}^{1*}, ..., \mathbf{\Psi}^{c*}, ..., \mathbf{\Psi}^{C*} \end{pmatrix} \leq \inf_{\substack{\mathbf{\Phi}^1 \in \mathbf{RC}^1 \\ \mathbf{\Psi}^1 \in \mathbf{LS}^1}} J^1\begin{pmatrix} \mathbf{\Phi}^{1}, ..., \mathbf{\Phi}^{c*}, ..., \mathbf{\Phi}^{C*} \\ \mathbf{\Psi}^{1}, ..., \mathbf{\Psi}^{c*}, ..., \mathbf{\Psi}^{C*} \end{pmatrix}$$

...

$$J^c\begin{pmatrix} \mathbf{\Phi}^{1*}, ..., \mathbf{\Phi}^{c*}, ..., \mathbf{\Phi}^{C*} \\ \mathbf{\Psi}^{1*}, ..., \mathbf{\Psi}^{c*}, ..., \mathbf{\Psi}^{C*} \end{pmatrix} \leq \inf_{\substack{\mathbf{\Phi}^c \in \mathbf{RC}^c \\ \mathbf{\Psi}^c \in \mathbf{LS}^c}} J^c\begin{pmatrix} \mathbf{\Phi}^{1*}, ..., \mathbf{\Phi}^{c}, ..., \mathbf{\Phi}^{C*} \\ \mathbf{\Psi}^{1*}, ..., \mathbf{\Psi}^{c}, ..., \mathbf{\Psi}^{C*} \end{pmatrix}$$

...

$$J^C\begin{pmatrix} \mathbf{\Phi}^{1*}, ..., \mathbf{\Phi}^{c*}, ..., \mathbf{\Phi}^{C*} \\ \mathbf{\Psi}^{1*}, ..., \mathbf{\Psi}^{c*}, ..., \mathbf{\Psi}^{C*} \end{pmatrix} \leq \inf_{\substack{\mathbf{\Phi}^C \in \mathbf{RC}^C \\ \mathbf{\Psi}^C \in \mathbf{LS}^C}} J^C\begin{pmatrix} \mathbf{\Phi}^{1*}, ..., \mathbf{\Phi}^{c*}, ..., \mathbf{\Phi}^{C} \\ \mathbf{\Psi}^{1*}, ..., \mathbf{\Psi}^{c*}, ..., \mathbf{\Psi}^{C} \end{pmatrix}$$

5.2.1 Optimal Control Formulation

In this section, we formulate the dynamic non-cooperative join load sharing, routing and congestion control problem as an Optimal Control Problem (OCP).

Theorem :

Consider the dynamic join load sharing, routing and congestion control problem in distributed systems with multiple competing classes, with fixed initial time t_0 *and final time* t_f*.*

If for each class c, $H^c(t, \mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi}, \mathbf{P}(t))$ *is differentiable and convex in* $(\mathbf{X}, \mathbf{\Phi}^c, \mathbf{\Psi}^c) \in (\mathbf{R}, \mathbf{RC}^c, \mathbf{LS}^c) \quad \forall\, t \in [t_0, t_f]$, *for each fixed value of*

$(\mathbf{\Phi}^1, \mathbf{\Psi}^1, ..., \mathbf{\Phi}^{c-1}, \mathbf{\Psi}^{c-1}, \mathbf{\Phi}^{c+1}, \mathbf{\Psi}^{c+1}, ..., \mathbf{\Phi}^C, \mathbf{\Psi}^C)$

$\in (\mathbf{RC}^1, \mathbf{LS}^1, ..., \mathbf{RC}^{c-1}, \mathbf{LS}^{c-1}, \mathbf{RC}^{c+1}, \mathbf{LS}^{c+1}, ..., \mathbf{RC}^C, \mathbf{LS}^C)$,

then $(\mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) \in (\mathbf{RC}, \mathbf{LS})$ *is a Nash equilibrium if and only if it solves the following* Optimal Control Problem $\forall\, t \in [t_0, t_f]$:

$\forall\, c$

$$\textit{minimize} \qquad \int_{t_0}^{t_f} g^c(t, \mathbf{X}(t), \mathbf{\Phi}^{1*}(t), \mathbf{\Psi}^{1*}(t), ..., \mathbf{\Phi}^c(t), \mathbf{\Psi}^c(t), ..., \mathbf{\Phi}^{C*}(t), \mathbf{\Psi}^{C*}(t))dt$$

$$\textit{with respect to} \quad (\mathbf{\Phi}^c(t), \mathbf{\Psi}^c(t))$$

$$\textit{such that} \qquad \dot{\mathbf{X}}(t) = \mathbf{f}(t, \mathbf{X}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t))$$

$$\mathbf{X}(t_0) = \mathbf{X}_0$$

$$(\mathbf{\Phi}^c(t), \mathbf{\Psi}^c(t)) \in (\mathbf{RC}^c, \mathbf{LS}^c)$$

Proof: It follows from the definition of the Nash equilibrium. □

Theorem :

Consider the dynamic join load sharing, routing and congestion control problem in distributed systems with multiple competing classes, with fixed initial time t_0 *and final time* t_f.

Let for each class c, $g^c(t, \mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi})$, $\mathbf{f}(t, \mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi})$, *are continuously differentiable with respect to* $\mathbf{X} \in \mathbf{R}^n$, $\forall\, t \in [t_0, t_f]$.

If $(\hat{\mathbf{\Phi}}^*(t, \mathbf{X}_0), \hat{\mathbf{\Psi}}^*(t, \mathbf{X}_0)) = (\mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) \in (\mathbf{RC}, \mathbf{LS})$ *is an* open-loop Nash equilibrium *and* $\{\mathbf{X}^*(t),\ t \in [t_0, t_f]\}$ *is the corresponding state trajectory, then* $\exists\, \mathbf{P}^c(t) : [t_0, t_f] \rightarrow \mathbf{R}^n$, $\forall\, c$ *continuous and piecewise continuously differentiable vector functions, such that* $\forall\, t \in [t_0, t_f]$:

$$\dot{\mathbf{X}}^*(t) = \mathbf{f}(t, \mathbf{X}^*(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\mathbf{X}^*(t_0) = \mathbf{X}_0$$

$$\left[\frac{\partial H^{c*}}{\partial \phi^c_{o[sd]}} - Q^c_{[sd]}(t)\right] * \phi^{c*}_{o[sd]}(t) = 0 \qquad \forall\ [sd] \in \mathbf{SD}^c,\ c$$

$$\left[\frac{\partial H^{c*}}{\partial \phi^c_{\pi[sd]}} - Q^c_{[sd]}(t)\right] * \phi^{c*}_{\pi[sd]}(t) = 0 \qquad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c, c$$

$$\left[\frac{\partial H^{c*}}{\partial \psi^c_{[sd]}} - Q^c_{[s.]}(t)\right] * \psi^{c*}_{[sd]}(t) = 0 \qquad \forall\ [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c,\ c$$

$$\frac{\partial H^{c*}}{\partial \phi^c_{o[sd]}} - Q^c_{[sd]}(t) \geq 0 \qquad \forall\ [sd] \in \mathbf{SD}^c,\ c$$

$$\frac{\partial H^{c*}}{\partial \phi^c_{\pi[sd]}} - Q^c_{[sd]}(t) \geq 0 \qquad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c,\ c$$

$$\frac{\partial H^{c*}}{\partial \psi^c_{[sd]}} - Q^c_{[s.]}(t) \geq 0 \qquad \forall\ [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c,\ c$$

$$\dot{\mathbf{P}}^c(t) = -\nabla_{\mathbf{X}} H^c(t, \mathbf{X}^*, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t), \mathbf{P}^c(t)) \qquad \forall\ c$$

$$\mathbf{P}^c(t_f) = \mathbf{0} \qquad \forall\ c$$

$$\phi^{c*}_{o[sd]}(t) + \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \phi^{c*}_{\pi[sd]}(t) = 1 \qquad \forall\ [sd] \in \mathbf{SD}^c,\ c$$

$$\sum_{[.d] \in \mathbf{D}^c_{[s.]}} \psi^{c*}_{[sd]}(t) = 1 \qquad \forall\ [s.] \in \mathbf{S}^c,\ c$$

$$\phi^{c*}_{o[sd]}(t),\ \phi^{c*}_{\pi[sd]}(t) \geq 0 \qquad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c,\ c$$

$$\psi^{c*}_{[sd]}(t) \geq 0 \qquad \forall\ [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c,\ c$$

Proof: The Lagrangian for each class c is

$$L^c = H^c + \sum_{[sd]\in \mathbf{SD}^c} Q^c_{[sd]} * \left[1 - \phi^c_{o[sd]} - \sum_{\pi[sd]\in \mathbf{\Pi}^c_{[sd]}} \phi^c_{\pi[sd]}\right] + \sum_{[s.]\in \mathbf{S}^c} Q^c_{[s.]} * \left[1 - \sum_{[.d]\in \mathbf{D}^c_{[s.]}} \psi^c_{[sd]}\right]$$

with $\phi^c_{o[sd]},\ \phi^c_{\pi[sd]},\ \psi^c_{[sd]} \geq 0 \quad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c,\ c$

Pontryagin's maximum principle necessary conditions are:

$$\dot{\mathbf{X}}^*(t) = \mathbf{f}(t, \mathbf{X}^*(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\mathbf{X}^*(t_0) = \mathbf{X}_0$$

$$\frac{\partial L^{c*}}{\partial \phi^c_{o[sd]}} * \phi^{c*}_{o[sd]}(t) = 0 \ \Rightarrow\ [\frac{\partial H^{c*}}{\partial \phi^c_{o[sd]}} - Q^c_{[sd]}(t)] * \phi^{c*}_{o[sd]}(t) = 0 \quad \forall\ [sd] \in \mathbf{SD}^c,\ c$$

$$\frac{\partial L^{c*}}{\partial \phi^c_{\pi[sd]}} * \phi^{c*}_{\pi[sd]}(t) = 0 \ \Rightarrow\ [\frac{\partial H^{c*}}{\partial \phi^c_{\pi[sd]}} - Q^c_{[sd]}(t)] * \phi^{c*}_{\pi[sd]}(t) = 0$$

$$\forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c, c$$

$$\frac{\partial L^{c*}}{\partial \psi^c_{[sd]}} * \psi^{c*}_{[sd]}(t) = 0 \ \Rightarrow\ [\frac{\partial H^{c*}}{\partial \psi^c_{[sd]}} - Q^c_{[s.]}(t)] * \psi^{c*}_{[sd]}(t) = 0 \quad \forall\ [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c,\ c$$

$$\frac{\partial L^{c*}}{\partial \phi^c_{o[sd]}} \geq 0 \ \Rightarrow\ \frac{\partial H^{c*}}{\partial \phi^c_{o[sd]}} - Q^c_{[sd]}(t) \geq 0 \quad \forall\ [sd] \in \mathbf{SD}^c,\ c$$

$$\frac{\partial L^{c*}}{\partial \phi^c_{\pi[sd]}} \geq 0 \ \Rightarrow\ \frac{\partial H^{c*}}{\partial \phi^c_{\pi[sd]}} - Q^c_{[sd]}(t) \geq 0 \quad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c,\ c$$

$$\frac{\partial L^{c*}}{\partial \psi^c_{[sd]}} \geq 0 \ \Rightarrow\ \frac{\partial H^{c*}}{\partial \psi^c_{[sd]}} - Q^c_{[s.]}(t) \geq 0 \quad \forall\ [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c,\ c$$

$$\dot{\mathbf{P}}^c(t) = -\nabla_{\mathbf{X}} H^c(t, \mathbf{X}^*, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t), \mathbf{P}^c(t)) \qquad \forall\ c$$

$$\mathbf{P}^c(t_f) = \mathbf{0} \qquad \forall\ c$$

$$\frac{\partial L^{c*}}{\partial Q^c_{[sd]}} \geq 0 \ \Rightarrow \phi^{c*}_{o[sd]}(t) + \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \phi^{c*}_{\pi[sd]}(t) = 1 \quad \forall\ [sd] \in \mathbf{SD}^c,\ c$$

$$\frac{\partial L^{c*}}{\partial Q^c_{[s.]}} = 0 \ \Rightarrow \sum_{[.d] \in \mathbf{D}^c_{[s.]}} \psi^{c*}_{[sd]}(t) = 1 \quad \forall\ [s.] \in \mathbf{S}^c,\ c$$

$$\phi^{c*}_{o[sd]}(t),\ \phi^{c*}_{\pi[sd]}(t) \geq 0 \qquad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c,\ c$$

$$\psi^{c*}_{[sd]}(t) \geq 0 \qquad \forall\ [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c,\ c$$

□

Theorem :

Consider the dynamic join load sharing, routing and congestion control problem in distributed systems with multiple competing classes, with fixed initial time t_0 *and final time* t_f.

Let for each class c, $g^c(t, \mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi})$, $\mathbf{f}(t, \mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi})$, *are continuously differentiable with respect to* $(\mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi}) \in (\mathbf{R}^n, \mathbf{RC}, \mathbf{LS})$, $\forall\ t \in [t_0, t_f]$.

If $(\hat{\mathbf{\Phi}}^*(t, \mathbf{X}, \mathbf{X_0}), \hat{\mathbf{\Psi}}^*(t, \mathbf{X}, \mathbf{X_0})) = (\mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) \in (\mathbf{RC}, \mathbf{LS})$ *is a closed-loop memoryless Nash equilibrium such that* $(\hat{\mathbf{\Phi}}^{c*}(t, \mathbf{X}, \mathbf{X_0}), \hat{\mathbf{\Phi}}^{c*}(t, \mathbf{X}, \mathbf{X_0}))$ *is continuously differentiable with respect to* $\mathbf{X} \in \mathbf{R}^n$, $\forall\ c$, $t \in [t_0, t_f]$ *and* $\{\mathbf{X}^*(t),\ t \in [t_0, t_f]\}$ *is the corresponding state trajectory, then* $\exists\ \mathbf{P}^c(t) : [t_0, t_f] \rightarrow \mathbf{R}^n$, $\forall\ c$, *continuous and piecewise continuously differentiable vector functions, such that* $\forall\ t \in [t_0, t_f]$:

$$\dot{\mathbf{X}}^*(t) = \mathbf{f}(t, \mathbf{X}^*(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\mathbf{X}^*(t_0) = \mathbf{X}_0$$

$$\left[\frac{\partial H^{c*}}{\partial \phi^c_{o[sd]}} - Q^c_{[sd]}(t)\right] * \phi^{c*}_{o[sd]}(t) = 0 \quad \forall\ [sd] \in \mathbf{SD}^c,\ c$$

$$\left[\frac{\partial H^{c*}}{\partial \phi^c_{\pi[sd]}} - Q^c_{[sd]}(t)\right] * \phi^{c*}_{\pi[sd]}(t) = 0 \quad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c, c$$

$$\left[\frac{\partial H^{c*}}{\partial \psi^c_{[sd]}} - Q^c_{[s.]}(t)\right] * \psi^{c*}_{[sd]}(t) = 0 \qquad \forall\ [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c,\ c$$

$$\frac{\partial H^{c*}}{\partial \phi^c_{o[sd]}} - Q^c_{[sd]}(t) \geq 0 \qquad \forall\ [sd] \in \mathbf{SD}^c,\ c$$

$$\frac{\partial H^{c*}}{\partial \phi^c_{\pi[sd]}} - Q^c_{[sd]}(t) \geq 0 \qquad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c,\ c$$

$$\frac{\partial H^{c*}}{\partial \psi^c_{[sd]}} - Q^c_{[s.]}(t) \geq 0 \qquad \forall\ [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c,\ c$$

$$\dot{\mathbf{P}}^c(t) = -\nabla_{\mathbf{X}} H^c(t, \mathbf{X}^*, \hat{\mathbf{\Phi}}^*(t, \mathbf{X}^*, \mathbf{X}_0), \hat{\mathbf{\Psi}}^*(t, \mathbf{X}^*, \mathbf{X}_0), \mathbf{P}^c(t)) \quad \forall\ c$$

$$\mathbf{P}^c(t_f) = \mathbf{0} \qquad \forall\ c$$

$$\phi^{c*}_{o[sd]}(t) + \sum_{\pi[sd]\in\mathbf{\Pi}^c_{[sd]}} \phi^{c*}_{\pi[sd]}(t) = 1 \quad \forall\ [sd] \in \mathbf{SD}^c,\ c$$

$$\sum_{[.d]\in\mathbf{D}^c_{[s.]}} \psi^{c*}_{[sd]}(t) = 1 \qquad \forall\ [s.] \in \mathbf{S}^c,\ c$$

$$\phi^{c*}_{o[sd]}(t),\ \phi^{c*}_{\pi[sd]}(t) \geq 0 \qquad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c,\ c$$

$$\psi^{c*}_{[sd]}(t) \geq 0 \qquad \forall\ [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c,\ c$$

Proof: The proof is similar to that for the open-loop solution.□

The above set of equations does not in general admit a single solution. In order to eliminate informational nonuniqueness in the derivation of Nash equilibrium under dynamic information, we constrain the Nash solution concept further (see next section).

5.2.2 Dynamic Programming Formulation

In this section, we formulate the dynamic non-cooperative joint load sharing, routing and congestion control problem as a Dynamic Programming Problem (DPP). Algorithms for solving DPP's may be found in books by Bellman [31], Howard [220], Kumar & Varaiya [274] Bertsekas [37], Ross [406] among others.

Theorem :

Consider the dynamic join load sharing, routing and congestion control problem in distributed systems with multiple competing classes, with fixed initial time t_0 *and final time* t_f.

$(\mathbf{\Phi^*}, \mathbf{\Psi^*}) \in (\mathbf{RC}, \mathbf{LS})$ *is optimal if and only if the following conditions are satisfied:*

$$i) \int_{t_0}^{t_f} g^c(t, \mathbf{X^*}(s), \hat{\mathbf{\Phi}}^*(\mathbf{X^*}(s)), \hat{\mathbf{\Psi}}^*(\mathbf{X^*}(s))ds = constant \quad \forall\ c$$

$ii) \exists\ \mathbf{X^*},\ \mathbf{P}^c,\ \ \forall\ c$ such that :

$$H^c(t, \mathbf{X^*}(t), \begin{matrix}\mathbf{\Phi}^{1*}(\mathbf{X^*}(t)), ..., \mathbf{\Phi}^{c*}(\mathbf{X^*}(t)), ..., \mathbf{\Phi}^{C*}(\mathbf{X^*}(t)) \\ \mathbf{\Psi}^{1*}(\mathbf{X^*}(t)), ..., \mathbf{\Psi}^{c*}(\mathbf{X^*}(t)), ..., \mathbf{\Psi}^{C*}(\mathbf{X^*}(t)),\end{matrix} \mathbf{P}^c(t)) -$$

$$- H^c(t, \mathbf{X}(t), \begin{matrix}\mathbf{\Phi}^{1*}(\mathbf{X}(t)), ..., \mathbf{\Phi}^{c}(\mathbf{X}(t)), ..., \mathbf{\Phi}^{C*}(\mathbf{X}(t)), \\ \mathbf{\Psi}^{1*}(\mathbf{X}(t)), ..., \mathbf{\Psi}^{c}(\mathbf{X}(t)), ..., \mathbf{\Psi}^{C*}(\mathbf{X}(t)),\end{matrix} \mathbf{P}^c(t)) +$$

$$+ \dot{\mathbf{P}}^c(t) * (\mathbf{X^*}(t) - \mathbf{X}) \quad \leq 0 \quad a.e. \quad t \in [t_0, t_f],\ \forall\ \mathbf{X} \in \mathbf{R}^n,\ (\mathbf{\Phi}^c, \mathbf{\Psi}^c) \in (\mathbf{RC}^c, \mathbf{LS}^c),\ c$$

$$\mathbf{P}^c(t_f) * (\mathbf{X^*}(t_f) - \mathbf{X}) \leq 0 \quad \forall\ \mathbf{X} \in \mathbf{R}^n$$

Proof: By integration of ii) and using the state equation, we get the Nash equilibrium conditions.

Definition :

Consider the dynamic join load sharing, routing and congestion control problem in distributed systems with multiple competing classes, with fixed initial time t_0 *and final time* t_f.

Under the memoryless perfect state or closed-loop perfect state information structure, $(\hat{\mathbf{\Phi}}, \hat{\mathbf{\Psi}}) \in (\mathbf{RC}, \mathbf{LS})$ *constitutes a* <u>*feedback Nash equilibrium*</u> *solution if and only if* $\exists\ V^c : [t_0, t_f] * \mathbf{R}^n \rightarrow R$ *satisfying the following relations for each class c:*

$$
\begin{aligned}
V^c(t,\mathbf{X}) &= \\
&= \int_t^{t_f} g^c(s,\mathbf{X}^*(s), \begin{matrix}\hat{\mathbf{\Phi}}^{1*}(s,\mathbf{I}(s)),...,\hat{\mathbf{\Phi}}^{c*}(s,\mathbf{I}(s)),...,\hat{\mathbf{\Phi}}^{C*}(s,\mathbf{I}(s)) \\ \hat{\mathbf{\Psi}}^{1*}(s,\mathbf{I}(s)),...,\hat{\mathbf{\Psi}}^{c*}(s,\mathbf{I}(s)),...,\hat{\mathbf{\Psi}}^{C*}(s,\mathbf{I}(s))\end{matrix})ds \leq \\
&\leq \int_t^{t_f} g^c(s,\mathbf{X}^*(s), \begin{matrix}\hat{\mathbf{\Phi}}^{1*}(s,\mathbf{I}(s)),...,\hat{\mathbf{\Phi}}^{c}(s,\mathbf{I}(s)),...,\hat{\mathbf{\Phi}}^{C*}(s,\mathbf{I}(s)) \\ \hat{\mathbf{\Psi}}^{1*}(s,\mathbf{I}(s)),...,\hat{\mathbf{\Psi}}^{c}(s,\mathbf{I}(s)),...,\hat{\mathbf{\Psi}}^{C*}(s,\mathbf{I}(s))\end{matrix})ds \\
&\forall\ (\hat{\mathbf{\Phi}}^c(s,\mathbf{I}(s)),\hat{\mathbf{\Psi}}^c(s,\mathbf{I}(s))) \in (\mathbf{RC}^c,\mathbf{LS}^c),\ \mathbf{X}\in\mathbf{R}^n \\
&\textit{such that}\ \ \forall\ s\in[t,t_f] \\
\dot{\mathbf{X}}^c(s) &= \mathbf{f}(s,\mathbf{X}^c(s), \begin{matrix}\hat{\mathbf{\Phi}}^{1*}(s,\mathbf{I}(s)),...,\hat{\mathbf{\Phi}}^{c}(s,\mathbf{I}(s)),...,\hat{\mathbf{\Phi}}^{C*}(s,\mathbf{I}(s)) \\ \hat{\mathbf{\Psi}}^{1*}(s,\mathbf{I}(s)),...,\hat{\mathbf{\Psi}}^{c}(s,\mathbf{I}(s)),...,\hat{\mathbf{\Psi}}^{C*}(s,\mathbf{I}(s))\end{matrix}) \\
\mathbf{X}^c(t) &= \mathbf{X} \\
\dot{\mathbf{X}}^*(s) &= \mathbf{f}(s,\mathbf{X}^*(s), \begin{matrix}\hat{\mathbf{\Phi}}^{1*}(s,\mathbf{I}(s)),...,\hat{\mathbf{\Phi}}^{c*}(s,\mathbf{I}(s)),...,\hat{\mathbf{\Phi}}^{C*}(s,\mathbf{I}(s)) \\ \hat{\mathbf{\Psi}}^{1*}(s,\mathbf{I}(s)),...,\hat{\mathbf{\Psi}}^{c*}(s,\mathbf{I}(s)),...,\hat{\mathbf{\Psi}}^{C*}(s,\mathbf{I}(s))\end{matrix}) \\
\mathbf{X}^*(s) &= \mathbf{X}
\end{aligned}
$$

where $\mathbf{I}(s) = \{\mathbf{X}(s), \mathbf{X_0}\}$ or $\mathbf{I}(s) = \{\mathbf{X}(\tau),\ \tau \leq s\}$.

The concept of feedback Nash equilibrium solution means that if $(\mathbf{\Phi}(s), \mathbf{\Psi}(s))$ is a feedback Nash equilibrium solution to the problem during $[t_0, t_f]$, is also a feedback Nash equilibrium solution to the problem during $[t, t_f]$, with the initial state taken as $\mathbf{X}(t)$. So, feedback Nash equilibrium strategies will depend only on the time variable and the current value of the state, but not on memory.

Proposition :

Every open-loop Nash equilibrium solution for the dynamic joint load sharing, routing and congestion control problem among cooperative classes is also closed-loop Nash equilibrium solution.

Proposition :

Under the memoryless (respectively, closed-loop) perfect state information structure, every feedback Nash equilibrium solution of the dynamic join load sharing,

routing and congestion control problem among competing classes is a closed-loop no memory (respectively, closed-loop) Nash equilibrium solution.

Theorem :

Consider the dynamic join load sharing, routing and congestion control problem in distributed systems with multiple competing classes, with fixed initial time t_0 and final time t_f.

Under the memory perfect state or closed loop perfect state information structure, $(\hat{\mathbf{\Phi}}, \hat{\mathbf{\Psi}}) \in (\mathbf{RC}, \mathbf{LS})$ provides a feedback Nash equilibrium solution if $\exists V^c$: $[t_0, t_f] * \mathbf{R}^n \rightarrow \mathbf{R}$, $\forall$ c satisfying the partial differential equations

$$-\frac{\partial V^c(t,\mathbf{X})}{\partial t} =$$

$$= \min_{(\mathbf{\Phi}^c, \mathbf{\Psi}^c) \in (\mathbf{RC}^c, \mathbf{LS}^c)} \{ \frac{\partial V^c(t,\mathbf{X})}{\partial \mathbf{X}} * \mathbf{f}(t, \mathbf{X}, \begin{matrix} \hat{\mathbf{\Phi}}^{1*}(t,\mathbf{X}), ..., \mathbf{\Phi}^c, ..., \hat{\mathbf{\Phi}}^{C*}(t,\mathbf{X}) \\ \hat{\mathbf{\Psi}}^{1*}(t,\mathbf{X}), ..., \mathbf{\Psi}^c, ..., \hat{\mathbf{\Psi}}^{C*}(t,\mathbf{X}) \end{matrix}) +$$

$$+ g(t, \mathbf{X}, \begin{matrix} \hat{\mathbf{\Phi}}^{1*}(t,\mathbf{X}), ..., \mathbf{\Phi}^c, ..., \hat{\mathbf{\Phi}}^{C*}(t,\mathbf{X}) \\ \hat{\mathbf{\Psi}}^{1*}(t,\mathbf{X}), ..., \mathbf{\Psi}^c, ..., \hat{\mathbf{\Psi}}^{C*}(t,\mathbf{X}) \end{matrix}) \}$$

$$= \frac{\partial V^c(t,\mathbf{X})}{\partial \mathbf{X}} * \mathbf{f}(t, \mathbf{X}, \begin{matrix} \hat{\mathbf{\Phi}}^{1*}(t,\mathbf{X}), ..., \hat{\mathbf{\Phi}}^{c*}(t,\mathbf{X}), ..., \hat{\mathbf{\Phi}}^{C*}(t,\mathbf{X}) \\ \hat{\mathbf{\Psi}}^{1*}(t,\mathbf{X}), ..., \hat{\mathbf{\Psi}}^{c*}(t,\mathbf{X}), ..., \hat{\mathbf{\Psi}}^{C*}(t,\mathbf{X}) \end{matrix}) +$$

$$+ g(t, \mathbf{X}, \begin{matrix} \hat{\mathbf{\Phi}}^{1*}(t,\mathbf{X}), ..., \hat{\mathbf{\Phi}}^{c*}(t,\mathbf{X}), ..., \hat{\mathbf{\Phi}}^{C*}(t,\mathbf{X}) \\ \hat{\mathbf{\Psi}}^{1*}(t,\mathbf{X}), ..., \hat{\mathbf{\Psi}}^{c*}(t,\mathbf{X}), ..., \hat{\mathbf{\Psi}}^{C*}(t,\mathbf{X}) \end{matrix})$$

5.2.3 Nonlinear Complementarity Problem Formulation

In this section, we formulate the dynamic non-cooperative load sharing, routing and congestion control problem as a Nonlinear Complementarity Problem (NCP).

Define the vector of class congestion control, routing and load sharing fractions as well as Lagrange multipliers:

$$\mathbf{Z}(t) = \left[... \ \phi^c_{o[sd]}(t) \ ... \ \phi^c_{\pi[sd]} \ ... \ Q^c_{[sd]}(t) \ .... \ \psi^c_{[sd]}(t) \ ... \ Q^c_{[s.]}(t) \ ... \right]^T$$

and the vector of class derivative of the Lagrangian with respect to the congestion control, routing and load sharing fractions as well as Lagrange multipliers:

$$\nabla L(\mathbf{Z}(t)) = \left[\ldots \left(\frac{\partial H^c}{\partial \phi^c_{o[sd]}} - Q^c_{o[sd]}(t) \right) \ \ldots \ \left(\frac{\partial H^c}{\partial \phi^c_{\pi[sd]}} - Q^c_{\pi[sd]}(t) \right) \ \ldots \right.$$

$$\ldots \left(1 - \phi^c_{o[sd]}(t) - \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \phi^c_{\pi[sd]}(t) \right) \ \ldots$$

$$\left. \ldots \left(\frac{\partial H^c}{\partial \psi^c_{[sd]}} - Q^c_{[s.]}(t) \right) \ \ldots \ \left(1 - \sum_{[.d] \in \mathbf{D}^c_{[s.]}} \psi^c_{[sd]}(t) \right) \ \ldots \right]$$

Theorem :

Consider the dynamic join load sharing, routing and congestion control problem in distributed systems with multiple competing classes, with fixed initial time t_0 *and final time* t_f.

If for each class c,

g^c *is differentiable and convex in* $(\mathbf{\Phi}^c(t), \mathbf{\Psi}^c(t)) \in (\mathbf{RC}^C, \mathbf{LS}^C)$, *for each fixed value of* $(\mathbf{\Phi}^1(t), \mathbf{\Psi}^1(t), ..., \mathbf{\Phi}^{c-1}(t), \mathbf{\Psi}^{c-1}(t), ..., \mathbf{\Psi}^{c+1}(t), \mathbf{\Phi}^{c+1}(t), ..., \mathbf{\Phi}^c(t), \mathbf{\Psi}^c(t))$

$\in (\mathbf{RC}^1, \mathbf{LS}^1, ..., \mathbf{RC}^{c-1}, \mathbf{LS}^{c-1}, ..., \mathbf{RC}^{c+1}, \mathbf{LS}^{c+1}, ..., \mathbf{RC}^C, \mathbf{LS}^C)$

then $(\mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) \in (\mathbf{RC}, \mathbf{LS})$ *is a Nash equilibrium if and only if it solves the following* Nonlinear Complementarity Problem $\forall\ t \in [t_0, t_f]$:

$$\nabla L(\mathbf{Z}^*(t)) * \mathbf{Z}^*(t) = 0$$

$$\nabla L(\mathbf{Z}^*(t)) \geq 0$$

$$\mathbf{Z}^*(t) \geq 0$$

$$\dot{\mathbf{X}}^*(t) = \mathbf{f}(t, \mathbf{X}^*(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\mathbf{X}^*(t_0) = \mathbf{X}_0$$

$$\dot{\mathbf{P}}^c(t) = -\nabla_{\mathbf{X}} H^c(t, \mathbf{X}^*, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t), \mathbf{P}^c(t)) \quad \forall\ c$$

$$\mathbf{P}^c(t_f) = \mathbf{0} \quad \forall\ c$$

Proof: After some algebraic manipulations, we find that the NCP: $\nabla L(\mathbf{Z}(t)) * \mathbf{Z}(t) = 0; \quad \nabla L(\mathbf{Z}(t)) \geq 0; \quad \mathbf{Z}(t) \geq 0$ with $\mathbf{Z}(t)$ and $\nabla L(\mathbf{Z}(t))$ as defined above, is equivalent to the Pontryagin's maximum principle necessary conditions. □

5.2.4 Variational Inequality Formulation

In this section, we formulate the dynamic non-cooperative load sharing, routing and congestion control problem as a Variational Inequality Problem (VIP).

Define the vector of class congestion control, routing and load sharing fractions:

$$(\mathbf{\Phi}(t), \mathbf{\Psi}(t)) = \left[\ldots \phi^c_{o[sd]}(t) \ \ldots \ \phi^c_{\pi[sd]}(t) \ \ldots \ \psi^c_{[sd]}(t) \ \ldots \right]^T$$

as well the vector of class derivatives of the cost function with respect to the congestion control, routing and load sharing fractions:

$$\nabla H(t, \mathbf{X}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t), \mathbf{P}(t)) \quad = \quad \left[\ldots \frac{\partial H^c}{\partial \phi^c_{o[sd]}} \ \ldots \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \frac{\partial H^c}{\partial \phi^c_{\pi[sd]}} \ \ldots \ \frac{\partial H^c}{\partial \psi^c_{[sd]}} \ \ldots \right]$$

Theorem :

Consider the dynamic join load sharing, routing and congestion control problem in distributed systems with multiple competing classes, with fixed initial time t_0 and final time t_f.

Let for each class c, $g^c(t, \mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi})$, $\mathbf{f}(t, \mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi})$, be continuously differentiable with respect to $(\mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi}) \in (\mathbf{R}^n, \mathbf{\Phi}, \mathbf{\Psi}) \quad \forall\, t \in [t_0, t_f]$. If H^c is continuously differentiable and convex in $(\mathbf{X}, \mathbf{\Phi}^c, \mathbf{\Psi}^c) \in (\mathbf{R}^n, \mathbf{RC}^c, \mathbf{LS}^c), \quad \forall\, t \in [t_0, t_f]$, for each fixed value of

$(\mathbf{\Phi}^1(t), \mathbf{\Psi}^1(t), ..., \mathbf{\Phi}^{c-1}(t), \mathbf{\Psi}^{c-1}(t), \mathbf{\Phi}^{c+1}(t), \mathbf{\Psi}^{c+1}(t), ..., \mathbf{\Phi}^c(t), \mathbf{\Psi}^c(t))$
$\in (\mathbf{RC}^1, \mathbf{LS}^1, ..., \mathbf{RC}^{c-1}, \mathbf{LS}^{c-1}, \mathbf{RC}^{c+1}, \mathbf{LS}^{c+1}, ..., \mathbf{RC}^C, \mathbf{LS}^C)$,

then $(\mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) \in (\mathbf{RC}, \mathbf{LS})$ is a Nash equilibrium if and only if it solves the following Variational Inequality Problem $\forall\, t \in [t_0, t_f]$:

$$\nabla H(t, \mathbf{X}^*(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t), \mathbf{P}(t)) * ((\mathbf{\Phi}, \mathbf{\Psi}) - (\mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))) \geq 0$$
$$\forall\ (\mathbf{\Phi}, \mathbf{\Psi}) \in (\mathbf{RC}, \mathbf{LS})$$
$$\dot{\mathbf{X}}^*(t) = \mathbf{f}(t, \mathbf{X}^*(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$
$$\mathbf{X}^*(t_0) = \mathbf{X}_0$$
$$\dot{\mathbf{P}}^c(t) = -\nabla_{\mathbf{X}} H^c(t, \mathbf{X}^*, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t), \mathbf{P}^c(t)) \quad \forall\ c$$
$$\mathbf{P}^c(t_f) = \mathbf{0} \quad \forall\ c$$

Proof: If $(\mathbf{\Phi}^{c*}(t), \mathbf{\Psi}^{c*}(t))$ is a local minimum for the following minimization problem

$$\begin{array}{ll} \textit{minimize} & \displaystyle\int_{t_0}^{t_f} g^c(t, \mathbf{X}(t), \begin{array}{c}\mathbf{\Phi}^{1*}(t), ..., \mathbf{\Phi}^c(t), ..., \mathbf{\Phi}^{C*}(t) \\ \mathbf{\Psi}^{1*}(t), ..., \mathbf{\Psi}^c(t), ..., \mathbf{\Psi}^{C*}(t)\end{array})dt \\ \textit{with respect to} & (\mathbf{\Phi}^c(t), \mathbf{\Psi}^c(t)) \\ \textit{such that} & \dot{\mathbf{X}}(t) = \mathbf{f}(t, \mathbf{X}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t)) \\ & \mathbf{X}(t_0) = \mathbf{X}_0 \end{array}$$

$$(\mathbf{\Phi}^c(t), \mathbf{\Psi}^c(t)) \in (\mathbf{RC}^c, \mathbf{LS}^c)$$

and g^c is a continuously differentiable convex function over the nonempty convex, closed and bounded set $(\mathbf{RC}^c, \mathbf{LS}^c)$, then $\forall \ddot{\in}[t_0, t_f]$:

$$\sum_{[sd]\in\mathbf{SD}^c} \left\{ \frac{\partial H^{c*}}{\partial \phi^c_{o[sd]}} * (\phi^c_{o[sd]} - \phi^{c*}_{o[sd]}(t)) + \sum_{\pi[sd]\in\mathbf{\Pi}^c_{[sd]}} \frac{\partial H^{c*}}{\partial \phi^c_{\pi[sd]}} * (\phi^c_{\pi[sd]} - \phi^{c*}_{\pi[sd]}(t)) + \right.$$
$$\left. + \frac{\partial H^{c*}}{\partial \psi^c_{[sd]}} * (\psi^c_{[sd]} - \psi^{c*}_{[sd]}(t)) \right\} \geq 0 \quad \forall\ (\mathbf{\Phi}^c, \mathbf{\Psi}^c) \in (\mathbf{RC}^c, \mathbf{LS}^c),\ c$$

Summing over all classes

$$\sum_{c} \sum_{[sd] \in \mathbf{SD}^c} \left\{ \frac{\partial H^{c*}}{\partial \phi^c_{o[sd]}} * (\phi^c_{o[sd]} - \phi^{c*}_{o[sd]}(t)) + \right.$$

$$+ \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \frac{\partial H^{c*}}{\partial \phi^c_{\pi[sd]}} * (\phi^c_{\pi[sd]} - \phi^{c*}_{\pi[sd]}(t)) +$$

$$\left. + \frac{\partial H^{c*}}{\partial \psi^c_{[sd]}} * (\psi^c_{[sd]} - \psi^{c*}_{[sd]}(t)) \right\} \geq 0 \qquad \forall \, (\mathbf{\Phi}^c, \mathbf{\Psi}^c) \in (\mathbf{RC}^c, \mathbf{LS}^c)$$

□

Another equivalent VIP formulation is given in the following Theorem:

Theorem :

Consider the dynamic join load sharing, routing and congestion control problem in distributed systems with multiple competing classes, with fixed initial time t_0 *and final time* t_f.

Let for each class c, $g^c(t, \mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi})$, $\mathbf{f}(t, \mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi})$, *be continuously differentiable with respect to* $(\mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi}) \in (\mathbf{R}^n, \mathbf{\Phi}, \mathbf{\Psi}) \;\; \forall \, t \in [t_0, t_f]$. *If* H^c *is continuously differentiable and convex in* $(\mathbf{X}, \mathbf{\Phi}^c, \mathbf{\Psi}^c) \in (\mathbf{R}^n, \mathbf{RC}^c, \mathbf{LS}^c), \quad \forall \, t \in [t_0, t_f]$, *for each fixed value of*

$$(\mathbf{\Phi}^1(t), \mathbf{\Psi}^1(t), ..., \mathbf{\Phi}^{c-1}(t), \mathbf{\Psi}^{c-1}(t), \mathbf{\Phi}^{c+1}(t), \mathbf{\Psi}^{c+1}(t), ..., \mathbf{\Phi}^c(t), \mathbf{\Psi}^c(t))$$
$$\in (\mathbf{RC}^1, \mathbf{LS}^1, ..., \mathbf{RC}^{c-1}, \mathbf{LS}^{c-1}, \mathbf{RC}^{c+1}, \mathbf{LS}^{c+1}, ..., \mathbf{RC}^C, \mathbf{LS}^C),$$

then $(\mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) \in (\mathbf{RC}, \mathbf{LS})$ *is a Nash equilibrium if and only if it solves the following* Variational Inequality Problem $\forall \, t \in [t_0, t_f]$:

$$\nabla L(\mathbf{Z}(t)^*) * (\mathbf{Z} - \mathbf{Z}(t)^*) \geq 0 \qquad \forall \, \mathbf{Z} \geq 0$$
$$\dot{\mathbf{X}}^*(t) = \mathbf{f}(t, \mathbf{X}^*(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$
$$\mathbf{X}^*(t_0) = \mathbf{X}_0$$
$$\dot{\mathbf{P}}^c(t) = -\nabla_{\mathbf{X}} H^c(t, \mathbf{X}^*, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t), \mathbf{P}^c(t)) \quad \forall \, c$$
$$\mathbf{P}^c(t_f) = \mathbf{0} \quad \forall \, c$$

Proof: The NCP: $f(x^*) * x^* = 0 \;\; f(x^*) \geq 0 \;\; x^* \geq 0$
and the VIP: find x^* such that $f(x^*) * (x - x^*) \geq 0 \;\; \forall x \geq 0$
are equivalent.□

5.2.5 Maximum Principle for Separable Cost Functions

In this section, we derive the first order necessary conditions for a Nash equilibrium on the path flows, when the cost function of each resource depends only on the flow on this resource.

According to the Nash equilibrium definition, each class c minimizes its cost function g^c given the optimum decisions of all other classes.

$$minimize \quad \int_{t_0}^{t_f} g^c(t, \mathbf{X}(t), \begin{matrix} \mathbf{\Phi}^{1*}(t), ..., \mathbf{\Phi}^{c}(t), ..., \mathbf{\Phi}^{C*}(t) \\ \mathbf{\Psi}^{1*}(t), ..., \mathbf{\Psi}^{c}(t), ..., \mathbf{\Psi}^{C*}(t) \end{matrix})dt =$$

$$= \sum_{ij} \int_{t_0}^{t_f} g_{ij}^c(t, \mathbf{X}_{ij}(t), \lambda_{ij}^{1*}(t), ..., \lambda_{ij}^{c}(t), ..., \lambda_{ij}^{C*}(t))dt \; +$$

$$+ \sum_{i} \int_{t_0}^{t_f} g_{i}^c(t, \mathbf{X}_{i}(t), \lambda_{i}^{1*}(t), ..., \lambda_{i}^{c}(t), ..., \lambda_{i}^{C*}(t))dt \; +$$

$$+ \sum_{[sd]} \int_{t_0}^{t_f} g_{[sd]}^c(t, \mathbf{X}_{o[sd]}(t), \lambda_{o[sd]}^{1*}(t), ..., \lambda_{o[sd]}^{c}(t), ..., \lambda_{o[sd]}^{C*}(t))dt \; +$$

$$+ \sum_{[.d]} \int_{t_0}^{t_f} g_{[.d]}^c(t, \mathbf{X}_{[.d]}(t), \lambda_{[.d]}^{1*}(t), ..., \lambda_{[.d]}^{c}(t), ..., \lambda_{[.d]}^{C*}(t))dt$$

$$with \; respect \; to \quad (\mathbf{\Phi}^c(t), \mathbf{\Psi}^c(t))$$

such that

$$\dot{\mathbf{X}}^k_{ij[sd]}(t) = \mathbf{f}^k_{ij[sd]}(t, \mathbf{X}_{ij}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t)) \quad \forall\ ij,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{X}}^k_{i[sd]}(t) = \mathbf{f}^k_{i[sd]}(t, \mathbf{X}_{i}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t)) \quad \forall\ i,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{X}}^k_{o[sd]}(t) = \mathbf{f}^k_{o[sd]}(t, \mathbf{X}_{o[sd]}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t)) \quad \forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{X}}^k_{[.d][sd]}(t) = \mathbf{f}^k_{[.d][sd]}(t, \mathbf{X}_{[.d]}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t)) \quad \forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\mathbf{X}^k_{ij[sd]}(t_0) = \mathbf{X}^k_{ij[sd],0} \quad \forall\ ij,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\mathbf{X}^k_{i[sd]}(t_0) = \mathbf{X}^k_{i[sd],0} \quad \forall\ i,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\mathbf{X}^k_{o[sd]}(t_0) = \mathbf{X}^k_{o[sd],0} \quad \forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\mathbf{X}^k_{[.d][sd]}(t_0) = \mathbf{X}^k_{[.d][sd],0} \quad \forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\phi^c_{o[sd]}(t) + \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \phi^c_{\pi[sd]}(t) = 1 \quad \forall\ [sd] \in \mathbf{SD}^c$$

$$\sum_{[.d] \in \mathbf{D}^c_{[s.]}} \psi^c_{[sd]}(t) = 1 \quad \forall\ [s.] \in \mathbf{S}^c$$

$$\phi^c_{o[sd]}(t),\ \phi^c_{\pi[sd]}(t) \geq 0 \quad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c$$

$$\psi^c_{[sd]}(t) \geq 0 \quad \forall\ [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c$$

Pontryagin's maximum principle necessary conditions are:

$$\dot{\mathbf{X}}^{k*}_{ij[sd]}(t) = \mathbf{f}^{k}_{ij[sd]}(t, \mathbf{X}^{*}_{ij}(t), \mathbf{\Phi}^{*}(t), \mathbf{\Psi}^{*}(t)) \quad \forall\ ij,\ [sd] \in \mathbf{SD}^{k},\ k$$

$$\dot{\mathbf{X}}^{k*}_{i[sd]}(t) = \mathbf{f}^{k}_{i[sd]}(t, \mathbf{X}^{*}_{i}(t), \mathbf{\Phi}^{*}(t), \mathbf{\Psi}^{*}(t)) \quad \forall\ i,\ [sd] \in \mathbf{SD}^{k},\ k$$

$$\dot{\mathbf{X}}^{k*}_{o[sd]}(t) = \mathbf{f}^{k}_{o[sd]}(t, \mathbf{X}^{*}_{o[sd]}(t), \mathbf{\Phi}^{*}(t), \mathbf{\Psi}^{*}(t)) \quad \forall\ [sd] \in \mathbf{SD}^{k},\ k$$

$$\dot{\mathbf{X}}^{k*}_{[.d][sd]}(t) = \mathbf{f}^{k}_{[.d][sd]}(t, \mathbf{X}^{*}_{[.d]}(t), \mathbf{\Phi}^{*}(t), \mathbf{\Psi}^{*}(t)) \quad \forall\ [sd] \in \mathbf{SD}^{k},\ k$$

$$\mathbf{X}^{k*}_{ij[sd]}(t_0) = \mathbf{X}^{k}_{ij[sd],0} \quad \forall\ ij,\ [sd] \in \mathbf{SD}^{k},\ k$$

$$\mathbf{X}^{k*}_{i[sd]}(t_0) = \mathbf{X}^{k}_{i[sd],0} \quad \forall\ i,\ [sd] \in \mathbf{SD}^{k},\ k$$

$$\mathbf{X}^{k*}_{o[sd]}(t_0) = \mathbf{X}^{k}_{o[sd],0} \quad \forall\ [sd] \in \mathbf{SD}^{k},\ k$$

$$\mathbf{X}^{k*}_{[.d][sd]}(t_0) = \mathbf{X}^{k}_{[.d][sd],0} \quad \forall\ [sd] \in \mathbf{SD}^{k},\ k$$

$$\left[\frac{\partial g^c_{[sd]}(t, \mathbf{X}^*_{o[sd]}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))}{\partial \phi^c_{o[sd]}} + \right.$$

$$+ \sum_k \mathbf{P}^{k,c}_{o[sd]} * \nabla_{\phi^c_{o[sd]}} \mathbf{f}^k_{o[sd]}(t, \mathbf{X}^*_{o[sd]}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$\left. - Q^c_{[sd]}(t) \right] * \phi^{c*}_{o[sd]}(t) = 0 \qquad \forall\ [sd] \in \mathbf{SD}^c,\ c$$

$$\left[\sum_{ij} \frac{\partial g^c_{ij}(t, \mathbf{X}^*_{ij}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))}{\partial \phi^c_{\pi[sd]}} + \sum_i \frac{\partial g^c_i(t, \mathbf{X}^*_i(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))}{\partial \phi^c_{\pi[sd]}} + \right.$$

$$+ \sum_k \sum_{[s'd']} \sum_{ij} \mathbf{P}^{k,c}_{ij[s'd']}(t) * \nabla_{\phi^c_{\pi[sd]}} \mathbf{f}^k_{ij[s'd']}(t, \mathbf{X}^*_{ij}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) +$$

$$+ \sum_k \sum_{[s'd']} \sum_i \mathbf{P}^{k,c}_{i[s'd']}(t) * \nabla_{\phi^c_{\pi[sd]}} \mathbf{f}^c_{i[s'd']}(t, \mathbf{X}^*_i(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$\left. - Q^c_{[sd]}(t) \right] * \phi^{c*}_{\pi[sd]}(t) = 0 \qquad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c,\ c$$

$$\Bigg[\sum_{ij}\frac{\partial g^c_{ij}(t,\mathbf{X}^*_{ij}(t),\mathbf{\Phi}^*(t),\mathbf{\Psi}^*(t))}{\partial \psi^c_{[sd]}}+\sum_{i}\frac{\partial g^c_{i}(t,\mathbf{X}^*_{i}(t),\mathbf{\Phi}^*(t),\mathbf{\Psi}^*(t))}{\partial \psi^c_{[sd]}}+$$

$$+\frac{\partial g^c_{[sd]}(t,\mathbf{X}^*_{o[sd]}(t),\mathbf{\Phi}^*(t),\mathbf{\Psi}^*(t))}{\partial \psi^c_{[sd]}}+\frac{\partial g^c_{[.d]}(t,\mathbf{X}^*_{[.d]}(t),\mathbf{\Phi}^*(t),\mathbf{\Psi}^*(t))}{\partial \psi^c_{[sd]}}+$$

$$+\sum_{k}\sum_{[s'd']}\sum_{ij}\mathbf{P}^{k,c}_{ij[s'd']}(t)*\nabla_{\psi^c_{[sd]}}\mathbf{f}^k_{ij[s'd']}(t,\mathbf{X}^*_{ij}(t),\mathbf{\Phi}^*(t),\mathbf{\Psi}^*(t))+$$

$$+\sum_{k}\sum_{[s'd']}\sum_{i}\mathbf{P}^{k,c}_{i[s'd']}(t)*\nabla_{\psi^c_{[sd]}}\mathbf{f}^k_{i[s'd']}(t,\mathbf{X}^*_{i}(t),\mathbf{\Phi}^*(t),\mathbf{\Psi}^*(t))+$$

$$+\sum_{k}\mathbf{P}^{k,c}_{o[sd]}(t)*\nabla_{\psi^c_{[sd]}}\mathbf{f}^k_{o[sd]}(t,\mathbf{X}^*_{o[sd]}(t),\mathbf{\Phi}^*(t),\mathbf{\Psi}^*(t))+$$

$$+\sum_{k}\sum_{[s'd]}\mathbf{P}^{k,c}_{[.d][s'd]}(t)*\nabla_{\psi^c_{[sd]}}\mathbf{f}^k_{[.d][s'd]}(t,\mathbf{X}^*_{[.d]}(t),\mathbf{\Phi}^*(t),\mathbf{\Psi}^*(t))-$$

$$-Q^c_{[s.]}(t)\Bigg]*\psi^{c*}_{[sd]}(t)=0 \qquad \forall\ [.d]\in\mathbf{D}^c_{[s.]},\ [s.]\in\mathbf{S}^c,\ c$$

$$\frac{\partial g^c_{[sd]}(t, \mathbf{X}^*_{o[sd]}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))}{\partial \phi^c_{o[sd]}} +$$

$$\sum_k \mathbf{P}^{k,c}_{o[sd]} * \nabla_{\phi^c_{o[sd]}} \mathbf{f}^k_{o[sd]}(t, \mathbf{X}^*_{o[sd]}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-Q^c_{[sd]}(t) \geq 0 \qquad \forall\ [sd] \in \mathbf{SD}^c,\ c$$

$$\sum_{ij} \frac{\partial g^c_{ij}(t, \mathbf{X}^*_{ij}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))}{\partial \phi^c_{\pi[sd]}} + \sum_i \frac{\partial g^c_i(t, \mathbf{X}^*_i(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))}{\partial \phi^c_{\pi[sd]}} +$$

$$+ \sum_k \sum_{[s'd']} \sum_{ij} \mathbf{P}^{k,c}_{ij[s'd']}(t) * \nabla_{\phi^c_{\pi[sd]}} \mathbf{f}^k_{ij[s'd']}(t, \mathbf{X}^*_{ij}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) +$$

$$+ \sum_k \sum_{[s'd']} \sum_i \mathbf{P}^{k,c}_{i[s'd']}(t) * \nabla_{\phi^c_{\pi[sd]}} \mathbf{f}^c_{i[s'd']}(t, \mathbf{X}^*_i(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-Q^c_{[sd]}(t) \geq 0 \qquad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c,\ c$$

$$\sum_{ij} \frac{\partial g^c_{ij}(t, \mathbf{X}^*_{ij}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))}{\partial \psi^c_{[sd]}} + \sum_{i} \frac{\partial g^c_{i}(t, \mathbf{X}^*_{i}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))}{\partial \psi^c_{[sd]}} +$$

$$+ \frac{\partial g^c_{[sd]}(t, \mathbf{X}^*_{o[sd]}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))}{\partial \psi^c_{[sd]}} + \frac{\partial g^c_{[.d]}(t, \mathbf{X}^*_{[.d]}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))}{\partial \psi^c_{[sd]}} +$$

$$+ \sum_{k} \sum_{[s'd']} \sum_{ij} \mathbf{P}^{k,c}_{ij[s'd']}(t) * \nabla_{\psi^c_{[sd]}} \mathbf{f}^k_{ij[s'd']}(t, \mathbf{X}^*_{ij}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) +$$

$$+ \sum_{k} \sum_{[s'd']} \sum_{i} \mathbf{P}^{k,c}_{i[s'd']}(t) * \nabla_{\psi^c_{[sd]}} \mathbf{f}^k_{i[s'd']}(t, \mathbf{X}^*_{i}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) +$$

$$+ \sum_{k} \mathbf{P}^{k,c}_{o[sd]}(t) * \nabla_{\psi^c_{[sd]}} \mathbf{f}^k_{o[sd]}(t, \mathbf{X}^*_{o[sd]}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) +$$

$$+ \sum_{k} \sum_{[s'.]} \mathbf{P}^{k,c}_{[.d][s'd]}(t) * \nabla_{\psi^c_{[sd]}} \mathbf{f}^k_{[.d][s'd]}(t, \mathbf{X}^*_{[.d]}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-Q^c_{[s.]}(t) \geq 0 \qquad \forall\, [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c,\ c$$

$$\dot{\mathbf{P}}^{c,k}_{ij[sd]}(t) = -\nabla_{\mathbf{X}^k_{ij[sd]}} g^c_{ij}(t, \mathbf{X}^*_{ij}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-\sum_n \sum_{[s'd']} \mathbf{P}^{c,n}_{ij[s'd']}(t) * \nabla_{\mathbf{X}^k_{ij[sd]}} \mathbf{f}^n_{ij[s'd']}(t, \mathbf{X}^*_{ij}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\forall\ ij,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{P}}^{c,k}_{i[sd]}(t) = -\nabla_{\mathbf{X}^k_{i[sd]}} g^c_i(t, \mathbf{X}^*_i, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-\sum_n \sum_{[s'd']} \mathbf{P}^{c,n}_{i[s'd']}(t) * \nabla_{\mathbf{X}^k_{i[sd]}} \mathbf{f}^n_{i[s'd']}(t, \mathbf{X}^*_i, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\forall\ i,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{P}}^{c,k}_{o[sd]}(t) = -\nabla_{\mathbf{X}^k_{o[sd]}} g^c_{[sd]}(t, \mathbf{X}^*_{o[sd]}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-\sum_n \mathbf{P}^{c,n}_{o[sd]}(t) * \nabla_{\mathbf{X}^k_{o[sd]}} \mathbf{f}^n_{o[sd]}(t, \mathbf{X}^*_{o[sd]}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{P}}^{c,k}_{[.d][sd]}(t) = -\nabla_{\mathbf{X}^k_{[.d][sd]}} g^c_{[.d]}(t, \mathbf{X}^*_{[.d]}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-\sum_n \sum_{[s'.]} \mathbf{P}^{c,n}_{[.d][sd]}(t) * \nabla_{\mathbf{X}^k_{[.d][sd]}} \mathbf{f}^n_{[.d][s'd]}(t, \mathbf{X}^*_{[.d]}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\phi^{c*}_{o[sd]}(t) + \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \phi^{c*}_{\pi[sd]}(t) = 1 \quad \forall\ [sd] \in \mathbf{SD}^c,\ c$$

$$\sum_{[.d] \in \mathbf{D}^c_{[s.]}} \psi^{c*}_{[sd]}(t) = 1 \qquad \forall\ [s.] \in \mathbf{S}^c,\ c$$

$$\phi^{c*}_{o[sd]}(t),\ \phi^{c*}_{\pi[sd]}(t) \geq 0 \qquad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c,\ c$$

$$\psi^{c*}_{[sd]}(t) \geq 0 \qquad \forall\ [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c,\ c$$

The partial derivatives of the cost function $g^c(t, \mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi})$ with respect to the path fractions $\phi^c_{\pi[sd]}$ can be written with respect to the link flows λ^c_{ij} and node flows λ^c_i:

$$\frac{\partial g^c_{ij}(t, \mathbf{X}_{ij}, \mathbf{\Phi}, \mathbf{\Psi})}{\partial \phi^c_{\pi[sd]}} = \frac{\partial g^c_{ij}(t, \mathbf{X}_{ij}, \mathbf{\Lambda}_{ij})}{\partial \lambda^c_{ij}} * \frac{\partial \lambda^c_{ij}}{\partial \phi^c_{\pi[sd]}} =$$

$$= \frac{\partial g^c_{ij}(t, \mathbf{X}_{ij}, \mathbf{\Lambda}_{ij})}{\partial \lambda^c_{ij}} * (\gamma^c_{[sd]}(t) + \gamma^c_{[s.]}(t) * \psi^c_{[sd]}(t)) * 1_{ij \in \pi[sd]}(t)$$

$$\frac{\partial g^c_i(t, \mathbf{X}_i, \mathbf{\Phi}, \mathbf{\Psi})}{\partial \phi^c_{\pi[sd]}} = \frac{\partial g^c_i(t, \mathbf{X}_i, \mathbf{\Lambda}_i)}{\partial \lambda^c_i} * \frac{\partial \lambda^c_i}{\partial \phi^c_{\pi[sd]}} =$$

$$= \frac{\partial g^c_i(t, \mathbf{X}_i, \mathbf{\Lambda}_i)}{\partial \lambda^c_i} * (\gamma^c_{[sd]}(t) + \gamma^c_{[s.]}(t) * \psi^c_{[sd]}(t)) * 1_{i \in \pi[sd]}(t)$$

$$\frac{\partial g^c_{[sd]}(t, \mathbf{X}_{o[sd]}, \mathbf{\Phi}, \mathbf{\Psi})}{\partial \phi^c_{o[sd]}} = \frac{\partial g^c_{[sd]}(t, \mathbf{X}_{o[sd]}, \mathbf{\Lambda}_{o[sd]})}{\partial \lambda^c_{o[sd]}} * \frac{\partial \lambda^c_{o[sd]}}{\partial \phi^c_{o[sd]}} =$$

$$= \frac{\partial g^c_{[sd]}(t, \mathbf{X}_{o[sd]}, \mathbf{\Lambda}_{o[sd]})}{\partial \lambda^c_{o[sd]}} * (\gamma^c_{[sd]}(t) + \gamma^c_{[s.]}(t) * \psi^c_{[sd]}(t))$$

$$\frac{\partial g^c_{ij}(t,\mathbf{X}_{ij},\mathbf{\Phi},\mathbf{\Psi})}{\partial \psi^c_{[sd]}} = \frac{\partial g^c_{ij}(t,\mathbf{X}_{ij},\mathbf{\Lambda}_{ij})}{\partial \lambda^c_{ij}} * \frac{\partial \lambda^c_{ij}}{\partial \psi^c_{[sd]}} =$$

$$= \sum_{\pi[sd]\in\mathbf{\Pi}^c_{[sd]}} \frac{\partial g^c_{ij}(t,\mathbf{X}_{ij},\mathbf{\Lambda}_{ij})}{\partial \lambda^c_{ij}} * \gamma^c_{[s.]}(t) * \phi^c_{\pi[sd]}(t) * 1_{ij\in\pi[sd]}(t)$$

$$\frac{\partial g^c_{i}(t,\mathbf{X}_{i},\mathbf{\Phi},\mathbf{\Psi})}{\partial \psi^c_{[sd]}} = \frac{\partial g^c_{i}(t,\mathbf{X}_{i},\mathbf{\Lambda}_{i})}{\partial \lambda^c_{i}} * \frac{\partial \lambda^c_{i}}{\partial \psi^c_{[sd]}} =$$

$$= \sum_{\pi[sd]\in\mathbf{\Pi}^c_{[sd]}} \frac{\partial g^c_{i}(t,\mathbf{X}_{i},\mathbf{\Lambda}_{i})}{\partial \lambda^c_{i}} * \gamma^c_{[s.]}(t) * \phi^c_{\pi[sd]}(t) * 1_{i\in\pi[sd]}(t)$$

$$\frac{\partial g^c_{[sd]}(t,\mathbf{X}_{o[sd]},\mathbf{\Phi},\mathbf{\Psi})}{\partial \psi^c_{[sd]}} = \frac{\partial g^c_{[sd]}(t,\mathbf{X}_{o[sd]},\mathbf{\Lambda}_{o[sd]})}{\partial \lambda^c_{o[sd]}} * \frac{\partial \lambda^c_{o[sd]}}{\partial \psi^c_{[sd]}} =$$

$$= \frac{\partial g^c_{[sd]}(t,\mathbf{X}_{o[sd]},\mathbf{\Lambda}_{o[sd]})}{\partial \lambda^c_{o[sd]}} * \gamma^c_{[s.]}(t) * \phi^c_{o[sd]}(t)$$

$$\frac{\partial g^c_{[.d]}(t,\mathbf{X}_{[.d]},\mathbf{\Phi},\mathbf{\Psi})}{\partial \psi^c_{[sd]}} = \frac{\partial g^c_{[.d]}(t,\mathbf{X}_{[.d]},\mathbf{\Lambda}_{[.d]})}{\partial \lambda^c_{[.d]}} * \frac{\partial \lambda^c_{[.d]}}{\partial \psi^c_{[sd]}} =$$

$$= \frac{\partial g^c_{[.d]}(t,\mathbf{X}_{[.d]},\mathbf{\Lambda}_{[.d]})}{\partial \lambda^c_{[.d]}} * \gamma^c_{[s.]}(t)$$

Then Pontryagin's maximum principle becomes:

$$\dot{\mathbf{X}}^{k*}_{ij[sd]}(t) = \mathbf{f}^{k}_{ij[sd]}(t, \mathbf{X}^{*}_{ij}(t), \mathbf{\Phi}^{*}(t), \mathbf{\Psi}^{*}(t)) \quad \forall\ ij,\ [sd] \in \mathbf{SD}^{k},\ k$$

$$\dot{\mathbf{X}}^{k*}_{i[sd]}(t) = \mathbf{f}^{k}_{i[sd]}(t, \mathbf{X}^{*}_{i}(t), \mathbf{\Phi}^{*}(t), \mathbf{\Psi}^{*}(t)) \quad \forall\ \subset,\ [sd] \in \mathbf{SD}^{k},\ k$$

$$\dot{\mathbf{X}}^{k*}_{o[sd]}(t) = \mathbf{f}^{k}_{o[sd]}(t, \mathbf{X}^{*}_{o[sd]}(t), \mathbf{\Phi}^{*}(t), \mathbf{\Psi}^{*}(t)) \quad \forall\ [sd] \in \mathbf{SD}^{k},\ k$$

$$\dot{\mathbf{X}}^{k*}_{[.d][sd]}(t) = \mathbf{f}^{k}_{[.d][sd]}(t, \mathbf{X}^{*}_{[.d]}(t), \mathbf{\Phi}^{*}(t), \mathbf{\Psi}^{*}(t)) \quad \forall\ [sd] \in \mathbf{SD}^{k},\ k$$

$$\mathbf{X}^{k*}_{ij[sd]}(t_0) = \mathbf{X}^{k}_{ij[sd],0} \quad \forall\ ij,\ [sd] \in \mathbf{SD}^{k},\ k$$

$$\mathbf{X}^{k*}_{i[sd]}(t_0) = \mathbf{X}^{k}_{i[sd],0} \quad \forall\ i,\ [sd] \in \mathbf{SD}^{k},\ k$$

$$\mathbf{X}^{k*}_{o[sd]}(t_0) = \mathbf{X}^{k}_{o[sd],0} \quad \forall\ [sd] \in \mathbf{SD}^{k},\ k$$

$$\mathbf{X}^{k*}_{[.d][sd]}(t_0) = \mathbf{X}^{k}_{[.d][sd],0} \quad \forall\ [sd] \in \mathbf{SD}^{k},\ k$$

$$\left[\frac{\partial g^c_{[sd]}(t,\mathbf{X}^*_{o[sd]}(t),\mathbf{\Lambda}^*_{o[sd]}(t))}{\partial \lambda^c_{o[sd]}} * (\gamma^c_{[sd]}(t)+\gamma^c_{[s.]}(t) * \psi^{c*}_{[sd]}(t))+\right.$$

$$+\sum_k \mathbf{P}^{k,c}_{o[sd]} * \nabla_{\phi^c_{o[sd]}} \mathbf{f}^k_{o[sd]}(t,\mathbf{X}^*_{o[sd]}(t),\mathbf{\Phi}^*(t),\mathbf{\Psi}^*(t))-$$

$$\left.-Q^c_{[sd]}(t)\right] * \phi^{c*}_{o[sd]}(t)=0 \qquad \forall\ [sd]\in \mathbf{SD}^c,\ c$$

$$\left[\sum_{ij} \frac{\partial g^c_{ij}(t,\mathbf{X}^*_{ij}(t),\mathbf{\Lambda}^*_{ij}(t))}{\partial \lambda^c_{ij}} * (\gamma^c_{[sd]}(t)+\gamma^c_{[s.]}(t) * \psi^{c*}_{[sd]}(t)) * 1_{ij\in\pi[sd]}(t)+\right.$$

$$+\sum_i \frac{\partial g^c_i(t,\mathbf{X}^*_i(t),\mathbf{\Lambda}^*_i(t))}{\partial \lambda^c_i} * (\gamma^c_{[sd]}(t)+\gamma^c_{[s.]}(t) * \psi^{c*}_{[sd]}(t)) * 1_{i\in\pi[sd]}(t)+$$

$$+\sum_k \sum_{[s'd']} \sum_{ij} \mathbf{P}^{k,c}_{ij[s'd']}(t) * \nabla_{\phi^c_{\pi[sd]}} \mathbf{f}^k_{ij[s'd']}(t,\mathbf{X}^*_{ij}(t),\mathbf{\Phi}^*(t),\mathbf{\Psi}^*(t))+$$

$$+\sum_k \sum_{[s'd']} \sum_{i} \mathbf{P}^{k,c}_{i[s'd']}(t) * \nabla_{\phi^c_{\pi[sd]}} \mathbf{f}^c_{i[s'd']}(t,\mathbf{X}^*_i(t),\mathbf{\Phi}^*(t),\mathbf{\Psi}^*(t))-$$

$$\left.-Q^c_{[sd]}(t)\right] * \phi^{c*}_{\pi[sd]}(t)=0 \qquad \forall\ \pi[sd]\in \mathbf{\Pi}^c_{[sd]},\ [sd]\in \mathbf{SD}^c,\ c$$

$$\Bigg[\sum_{ij}\sum_{\pi[sd]\in\mathbf{\Pi}^c_{[sd]}}\frac{\partial g^c_{ij}(t,\mathbf{X}^*_{ij}(t),\mathbf{\Lambda}^*_{ij}(t))}{\partial\lambda^c_{ij}}*\gamma^c_{[s.]}(t)*\phi^{c*}_{\pi[sd]}(t))*1_{ij\in\pi[sd]}(t)+$$

$$+\sum_{i}\sum_{\pi[sd]\in\mathbf{\Pi}^c_{[sd]}}\frac{\partial g^c_{i}(t,\mathbf{X}^*_{i}(t),\mathbf{\Lambda}^*_{i}(t))}{\partial\lambda^c_{i}}*\gamma^c_{[s.]}(t)*\phi^{c*}_{\pi[sd]}(t))*1_{i\in\pi[sd]}(t)+$$

$$+\frac{\partial g^c_{[sd]}(t,\mathbf{X}^*_{o[sd]}(t),\mathbf{\Lambda}^*_{o[sd]}(t))}{\partial\lambda^c_{o[sd]}}*\gamma^c_{[s.]}(t)*\phi^{c*}_{o[sd]}(t))+$$

$$+\frac{\partial g^c_{[.d]}(t,\mathbf{X}^*_{[.d]}(t),\mathbf{\Lambda}^*_{[.d]}(t))}{\partial\lambda^c_{[.d]}}*\gamma^c_{[s.]}(t)+$$

$$+\sum_{k}\sum_{[s'd']}\sum_{ij}\mathbf{P}^{k,c}_{ij[s'd']}(t)*\nabla_{\psi^c_{[sd]}}\mathbf{f}^k_{ij[s'd']}(t,\mathbf{X}^*_{ij}(t),\mathbf{\Phi}^*(t),\mathbf{\Psi}^*(t))+$$

$$+\sum_{k}\sum_{[s'd']}\sum_{i}\mathbf{P}^{k,c}_{i[s'd']}(t)*\nabla_{\psi^c_{[sd]}}\mathbf{f}^k_{i[s'd']}(t,\mathbf{X}^*_{i}(t),\mathbf{\Phi}^*(t),\mathbf{\Psi}^*(t))+$$

$$+\sum_{k}\mathbf{P}^{k,c}_{o[sd]}(t)*\nabla_{\psi^c_{[sd]}}\mathbf{f}^k_{o[sd]}(t,\mathbf{X}^*_{o[sd]}(t),\mathbf{\Phi}^*(t),\mathbf{\Psi}^*(t))+$$

$$+\sum_{k}\sum_{[s'.]}\mathbf{P}^{k,c}_{[.d][s'd]}(t)*\nabla_{\psi^c_{[sd]}}\mathbf{f}^k_{[.d][s'd]}(t,\mathbf{X}^*_{[.d]}(t),\mathbf{\Phi}^*(t),\mathbf{\Psi}^*(t))-$$

$$-Q^c_{[s.]}(t)\Bigg]*\psi^{c*}_{[sd]}(t)=0\qquad\qquad\forall\ [.d]\in\mathbf{D}^c_{[s.]},\ [s.]\in\mathbf{S}^c,\ c$$

$$\frac{\partial g^c_{[sd]}(t,\mathbf{X}^*_{o[sd]}(t),\mathbf{\Lambda}^*_{o[sd]}(t))}{\partial \lambda^c_{o[sd]}} * (\gamma^c_{[sd]}(t)+\gamma^c_{[s.]}(t) * \psi^{c*}_{[sd]}(t))+$$

$$+\sum_k \mathbf{P}^{k,c}_{o[sd]} * \nabla_{\phi^c_{o[sd]}} \mathbf{f}^k_{o[sd]}(t,\mathbf{X}^*_{o[sd]}(t),\mathbf{\Phi}^*(t),\mathbf{\Psi}^*(t))-$$

$$-Q^c_{[sd]}(t) \geq 0 \qquad \forall\ [sd] \in \mathbf{SD}^c,\ c$$

$$\sum_{ij} \frac{\partial g^c_{ij}(t,\mathbf{X}^*_{ij}(t),\mathbf{\Lambda}^*_{ij}(t))}{\partial \lambda^c_{ij}} * (\gamma^c_{[sd]}(t)+\gamma^c_{[s.]}(t) * \psi^{c*}_{[sd]}(t)) * 1_{ij\in\pi[sd]}(t)+$$

$$+\sum_i \frac{\partial g^c_i(t,\mathbf{X}^*_i(t),\mathbf{\Lambda}^*_i(t))}{\partial \lambda^c_i} * (\gamma^c_{[sd]}(t)+\gamma^c_{[s.]}(t) * \psi^{c*}_{[sd]}(t)) * 1_{i\in\pi[sd]}(t)+$$

$$+\sum_k \sum_{[s'd']} \sum_{ij} \mathbf{P}^{k,c}_{ij[s'd']}(t) * \nabla_{\phi^c_{\pi[sd]}} \mathbf{f}^k_{ij[s'd']}(t,\mathbf{X}^*_{ij}(t),\mathbf{\Phi}^*(t),\mathbf{\Psi}^*(t))+$$

$$+\sum_k \sum_{[s'd']} \sum_i \mathbf{P}^{k,c}_{i[s'd']}(t) * \nabla_{\phi^c_{\pi[sd]}} \mathbf{f}^c_{i[s'd']}(t,\mathbf{X}^*_i(t),\mathbf{\Phi}^*(t),\mathbf{\Psi}^*(t))-$$

$$-Q^c_{[sd]}(t) \geq 0 \qquad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c,\ c$$

$$\sum_{ij} \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \frac{\partial g^c_{ij}(t, \mathbf{X}^*_{ij}(t), \mathbf{\Lambda}^*_{ij}(t))}{\partial \lambda^c_{ij}} * \gamma^c_{[s.]}(t) * \phi^{c*}_{\pi[sd]}(t)) * 1_{ij \in \pi[sd]}(t) +$$

$$+ \sum_{i} \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \frac{\partial g^c_{i}(t, \mathbf{X}^*_{i}(t), \mathbf{\Lambda}^*_{i}(t))}{\partial \lambda^c_{i}} * \gamma^c_{[s.]}(t) * \phi^{c*}_{\pi[sd]}(t)) * 1_{i \in \pi[sd]}(t) +$$

$$+ \frac{\partial g^c_{[sd]}(t, \mathbf{X}^*_{o[sd]}(t), \mathbf{\Lambda}^*_{o[sd]}(t))}{\partial \lambda^c_{o[sd]}} * \gamma^c_{[s.]}(t) * \phi^{c*}_{o[sd]}(t)) +$$

$$+ \frac{\partial g^c_{[.d]}(t, \mathbf{X}^*_{[.d]}(t), \mathbf{\Lambda}^*_{[.d]}(t))}{\partial \lambda^c_{[.d]}} * \gamma^c_{[s.]}(t) +$$

$$+ \sum_{k} \sum_{[s'd']} \sum_{ij} \mathbf{P}^{k,c}_{ij[s'd']}(t) * \nabla_{\psi^c_{[sd]}} \mathbf{f}^k_{ij[s'd']}(t, \mathbf{X}^*_{ij}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) +$$

$$+ \sum_{k} \sum_{[s'd']} \sum_{i} \mathbf{P}^{k,c}_{i[s'd']}(t) * \nabla_{\psi^c_{[sd]}} \mathbf{f}^k_{i[s'd']}(t, \mathbf{X}^*_{i}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) +$$

$$+ \sum_{k} \mathbf{P}^{k,c}_{o[sd]}(t) * \nabla_{\psi^c_{[sd]}} \mathbf{f}^k_{o[sd]}(t, \mathbf{X}^*_{o[sd]}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) +$$

$$+ \sum_{k} \sum_{[s'.]} \mathbf{P}^{k,c}_{[.d][s'd]}(t) * \nabla_{\psi^c_{[sd]}} \mathbf{f}^k_{[.d][s'd]}(t, \mathbf{X}^*_{[.d]}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-Q^c_{[s.]}(t) \geq 0 \qquad \forall\ [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c,\ c$$

$$\dot{\mathbf{P}}^{c,k}_{ij[sd]}(t) = -\nabla_{\mathbf{X}^k_{ij[sd]}} g^c_{ij}(t, \mathbf{X}^*_{ij}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-\sum_n \sum_{[s'd']} \mathbf{P}^{c,n}_{ij[s'd']}(t) * \nabla_{\mathbf{X}^k_{ij[sd]}} \mathbf{f}^n_{ij[s'd']}(t, \mathbf{X}^*_{ij}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\forall\ ij,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{P}}^{c,k}_{i[sd]}(t) = -\nabla_{\mathbf{X}^k_{i[sd]}} g^c_i(t, \mathbf{X}^*_i, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-\sum_n \sum_{[s'd']} \mathbf{P}^{c,n}_{i[s'd']}(t) * \nabla_{\mathbf{X}^k_{i[sd]}} \mathbf{f}^n_{i[s'd']}(t, \mathbf{X}^*_i, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\forall\ i,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{P}}^{c,k}_{o[sd]}(t) = -\nabla_{\mathbf{X}^k_{o[sd]}} g^c_{[sd]}(t, \mathbf{X}^*_{o[sd]}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-\sum_n \mathbf{P}^{c,n}_{o[sd]}(t) * \nabla_{\mathbf{X}^k_{o[sd]}} \mathbf{f}^n_{o[sd]}(t, \mathbf{X}^*_{o[sd]}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{P}}^{c,k}_{[.d][sd]}(t) = -\nabla_{\mathbf{X}^k_{[.d][sd]}} g^c_{[.d]}(t, \mathbf{X}^*_{[.d]}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-\sum_n \sum_{[s'.]} \mathbf{P}^{c,n}_{[.d][sd]}(t) * \nabla_{\mathbf{X}^k_{[.d][sd]}} \mathbf{f}^n_{[.d][s'd]}(t, \mathbf{X}^*_{[.d]}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\phi^{c*}_{o[sd]}(t) + \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \phi^{c*}_{\pi[sd]}(t) = 1 \quad \forall\ [sd] \in \mathbf{SD}^c,\ c$$

$$\sum_{[.d] \in \mathbf{D}^c_{[s.]}} \psi^{c*}_{[sd]}(t) = 1 \qquad \forall\ [s.] \in \mathbf{S}^c,\ c$$

$$\phi^{c*}_{o[sd]}(t),\ \phi^{c*}_{\pi[sd]}(t) \geq 0 \qquad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c,\ c$$

$$\psi^{c*}_{[sd]}(t) \geq 0 \qquad \forall\ [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c,\ c$$

Next, for each class c, we define the length for the rejected flow $[sd]$, the length for the path $\pi[sd]$ and the length for the source-destination pair $[sd]$:

$$l^{c,Nash}_{o[sd]}(t) = \frac{\partial g^c_{[sd]}(t, \mathbf{X}_{o[sd]}(t), \mathbf{\Lambda}_{o[sd]}(t))}{\partial \lambda^c_{o[sd]}} * (\gamma^c_{[sd]}(t) + \gamma^c_{[s.]}(t) * \psi^c_{[sd]}(t)) +$$

$$+ \sum_k \mathbf{P}^{k,c}_{o[sd]} * \nabla_{\phi^c_{o[sd]}} \mathbf{f}^k_{o[sd]}(t, \mathbf{X}_{o[sd]}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t))$$

$$\forall\ [sd] \in \mathbf{SD}^c,\ c$$

$$
\begin{aligned}
l^{c,Nash}_{\pi[sd]}(t) = & \sum_{ij} \frac{\partial g^c_{ij}(t, \mathbf{X}_{ij}(t), \mathbf{\Lambda}_{ij}(t))}{\partial \lambda^c_{ij}} * (\gamma^c_{[sd]}(t) + \gamma^c_{[s.]}(t) * \psi^c_{[sd]}(t)) * 1_{ij \in \pi[sd]}(t) + \\
& + \sum_{i} \frac{\partial g^c_{i}(t, \mathbf{X}_{i}(t), \mathbf{\Lambda}_{i}(t))}{\partial \lambda^c_{i}} * (\gamma^c_{[sd]}(t) + \gamma^c_{[s.]}(t) * \psi^c_{[sd]}(t)) * 1_{i \in \pi[sd]}(t) + \\
& + \sum_{k} \sum_{[s'd']} \sum_{ij} \mathbf{P}^{k,c}_{ij[s'd']}(t) * \nabla_{\phi^c_{\pi[sd]}} \mathbf{f}^k_{ij[s'd']}(t, \mathbf{X}_{ij}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t)) + \\
& + \sum_{k} \sum_{[s'd']} \sum_{i} \mathbf{P}^{k,c}_{i[s'd']}(t) * \nabla_{\phi^c_{\pi[sd]}} \mathbf{f}^c_{i[s'd']}(t, \mathbf{X}_{i}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t))
\end{aligned}
$$

$$
\forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c,\ c
$$

$$l^{c,Nash}_{[sd]}(t) = \sum_{ij} \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \frac{\partial g^c_{ij}(t, \mathbf{X}_{ij}(t), \mathbf{\Lambda}_{ij}(t))}{\partial \lambda^c_{ij}} * \gamma^c_{[s.]}(t) * \phi^c_{\pi[sd]}(t)) * 1_{ij \in \pi[sd]}(t) +$$

$$+ \sum_{i} \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \frac{\partial g^c_{i}(t, \mathbf{X}_{i}(t), \mathbf{\Lambda}_{i}(t))}{\partial \lambda^c_{i}} * \gamma^c_{[s.]}(t) * \phi^c_{\pi[sd]}(t)) * 1_{i \in \pi[sd]}(t) +$$

$$+ \frac{\partial g^c_{[sd]}(t, \mathbf{X}_{o[sd]}(t), \mathbf{\Lambda}_{o[sd]}(t))}{\partial \lambda^c_{o[sd]}} * \gamma^c_{[s.]}(t) * \phi^c_{o[sd]}(t)) +$$

$$+ \frac{\partial g^c_{[.d]}(t, \mathbf{X}_{[.d]}(t), \mathbf{\Lambda}_{[.d]}(t))}{\partial \lambda^c_{[.d]}} * \gamma^c_{[s.]}(t) +$$

$$+ \sum_{k} \sum_{[s'd']} \sum_{ij} \mathbf{P}^{k,c}_{ij[s'd']}(t) * \nabla_{\psi^c_{[sd]}} \mathbf{f}^k_{ij[s'd']}(t, \mathbf{X}_{ij}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t)) +$$

$$+ \sum_{k} \sum_{[s'd']} \sum_{i} \mathbf{P}^{k,c}_{i[s'd']}(t) * \nabla_{\psi^c_{[sd]}} \mathbf{f}^k_{i[s'd']}(t, \mathbf{X}_{i}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t)) +$$

$$+ \sum_{k} \mathbf{P}^{k,c}_{o[sd]}(t) * \nabla_{\psi^c_{[sd]}} \mathbf{f}^k_{o[sd]}(t, \mathbf{X}_{o[sd]}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t)) +$$

$$+ \sum_{k} \sum_{[s'.]} \mathbf{P}^{k,c}_{[.d][s'd]}(t) * \nabla_{\psi^c_{[sd]}} \mathbf{f}^k_{[.d][s'd]}(t, \mathbf{X}_{[.d]}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t)) -$$

$$\forall\ [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c,\ c$$

External arriving flow at a source is assigned to the destination that has the minimum length from the source. However, this flow may be rejected if the length of rejecting it is less than the lengths of the paths to its destination. If it is accepted, then it is routed to its destination via the minimum length path.

In the next section, we will derive the same conditions by an alternative way, and we shall state the above ideas more formally.

5.2.6 V.I. for Separable Cost Functions

Equivalently, the Nash equilibrium definition, each class c minimizes its cost function g^c given the optimum decisions of all other classes. We first solve the routing and congestion control problem assuming that all other classes act optimally for themselves. So, class c first solves the routing and congestion control problems

$$\textit{minimize} \qquad \int_{t_0}^{t_f} g^c(t, \mathbf{X}, \begin{matrix} \mathbf{\Phi}^{1*}(t), ..., \mathbf{\Phi}^c(t), ..., \mathbf{\Phi}^{C*}(t) \\ \mathbf{\Psi}^{1*}(t), ..., \mathbf{\Psi}^c(t), ..., \mathbf{\Psi}^{C*}(t) \end{matrix})dt =$$

$$= \sum_{ij} \int_{t_0}^{t_f} g^c_{ij}(t, \mathbf{X}_{ij}(t), \lambda^{1*}_{ij}(t), ..., \lambda^c_{ij}(t), ..., \lambda^{C*}_{ij}(t))dt \; +$$

$$+ \sum_{i} \int_{t_0}^{t_f} g^c_{i}(t, \mathbf{X}_{i}(t), \lambda^{1*}_{i}(t), ..., \lambda^c_{i}(t), ..., \lambda^{C*}_{i}(t))dt \; +$$

$$+ \sum_{[sd]} \int_{t_0}^{t_f} g^c_{[sd]}(t, \mathbf{X}_{o[sd]}(t), \lambda^{1*}_{o[sd]}(t), ..., \lambda^c_{o[sd]}(t), ..., \lambda^{C*}_{o[sd]}(t))dt \; +$$

$$+ \sum_{[.d]} \int_{t_0}^{t_f} g^c_{[.d]}(t, \mathbf{X}_{[.d]}(t), \lambda^{1*}_{[.d]}(t), ..., \lambda^c_{[.d]}(t), ..., \lambda^{C*}_{[.d]}(t))dt$$

with respect to $\mathbf{\Phi}^c(t)$

such that

$$\dot{\mathbf{X}}^k_{ij[sd]}(t) = \mathbf{f}^k_{ij[sd]}(t, \mathbf{X}_{ij}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t)) \quad \forall\ ij,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{X}}^k_{i[sd]}(t) = \mathbf{f}^k_{i[sd]}(t, \mathbf{X}_{i}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t)) \quad \forall\ i,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{X}}^k_{o[sd]}(t) = \mathbf{f}^k_{o[sd]}(t, \mathbf{X}_{o[sd]}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t)) \quad \forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{X}}^k_{[.d][sd]}(t) = \mathbf{f}^k_{[.d][sd]}(t, \mathbf{X}_{[.d]}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t)) \quad \forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\mathbf{X}^k_{ij[sd]}(t_0) = \mathbf{X}^k_{ij[sd],0} \quad \forall\ ij,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\mathbf{X}^k_{i[sd]}(t_0) = \mathbf{X}^k_{i[sd],0} \quad \forall\ i,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\mathbf{X}^k_{o[sd]}(t_0) = \mathbf{X}^k_{o[sd],0} \quad \forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\mathbf{X}^k_{[.d][sd]}(t_0) = \mathbf{X}^k_{[.d][sd],0} \quad \forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\phi^c_{o[sd]}(t) + \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \phi^c_{\pi[sd]}(t) = 1 \quad \forall\ [sd] \in \mathbf{SD}^c$$

$$\phi^c_{o[sd]}(t),\ \phi^c_{\pi[sd]}(t) \geq 0 \quad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c$$

The necessary optimality conditions for class c are

$$\sum_{[sd]\in\mathbf{SD}^c} \left\{ \frac{\partial g^c(t,\mathbf{X}^*(t),\mathbf{\Phi}^*(t),\mathbf{\Psi}^*(t))}{\partial \phi^c_{o[sd]} * (\phi^c_{o[sd]} - \phi^{c*}_{o[sd]}(t)) \; +} \right.$$

$$\left. + \sum_{\pi[sd]\in\mathbf{\Pi}^c_{[sd]}} \frac{\partial g^c(t,\mathbf{X}^*(t),\mathbf{\Phi}^*(t),\mathbf{\Psi}^*(t))}{\partial \phi^c_{\pi[sd]} * (\phi^c_{\pi[sd]} - \phi^{c*}_{\pi[sd]}(t))} \right\} \geq 0 \qquad \forall\ \mathbf{\Phi}^c \in \mathbf{RC}^c$$

such that

$$\begin{aligned}
\dot{\mathbf{X}}^{k*}_{ij[sd]}(t) &= \mathbf{f}^k_{ij[sd]}(t,\mathbf{X}^*_{ij}(t),\mathbf{\Phi}^*(t),\mathbf{\Psi}^*(t)) && \forall\ ij,\ [sd]\in\mathbf{SD}^k,\ k\\
\dot{\mathbf{X}}^{k*}_{i[sd]}(t) &= \mathbf{f}^k_{i[sd]}(t,\mathbf{X}^*_{i}(t),\mathbf{\Phi}^*(t),\mathbf{\Psi}^*(t)) && \forall\ i,\ [sd]\in\mathbf{SD}^k,\ k\\
\dot{\mathbf{X}}^{k*}_{o[sd]}(t) &= \mathbf{f}^k_{o[sd]}(t,\mathbf{X}^*_{o[sd]}(t),\mathbf{\Phi}^*(t),\mathbf{\Psi}^*(t)) && \forall\ [sd]\in\mathbf{SD}^k,\ k\\
\dot{\mathbf{X}}^{k*}_{[.d][sd]}(t) &= \mathbf{f}^k_{[.d][sd]}(t,\mathbf{X}^*_{[.d]}(t),\mathbf{\Phi}^*(t),\mathbf{\Psi}^*(t)) && \forall\ [sd]\in\mathbf{SD}^k,\ k\\
\mathbf{X}^{k*}_{ij[sd]}(t_0) &= \mathbf{X}^k_{ij[sd],0} && \forall\ ij,\ [sd]\in\mathbf{SD}^k,\ k\\
\mathbf{X}^{k*}_{i[sd]}(t_0) &= \mathbf{X}^k_{i[sd],0} && \forall\ i,\ [sd]\in\mathbf{SD}^k,\ k\\
\mathbf{X}^{k*}_{o[sd]}(t_0) &= \mathbf{X}^k_{o[sd],0} && \forall\ [sd]\in\mathbf{SD}^k,\ k\\
\mathbf{X}^{k*}_{[.d][sd]}(t_0) &= \mathbf{X}^k_{[.d][sd],0} && \forall\ [sd]\in\mathbf{SD}^k,\ k
\end{aligned}$$

$$\dot{\mathbf{P}}^{c,k}_{ij[sd]}(t) = -\nabla_{\mathbf{X}^k_{ij[sd]}} g^c_{ij}(t, \mathbf{X}^*_{ij}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-\sum_n \sum_{[s'd']} \nabla_{\mathbf{X}^k_{ij[sd]}} \mathbf{P}^{c,n}_{ij[s'd']}(t) * \mathbf{f}^n_{ij[s'd']}(t, \mathbf{X}^*_{ij}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\forall\ ij,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{P}}^{c,k}_{i[sd]}(t) = -\nabla_{\mathbf{X}^k_{i[sd]}} g^c_i(t, \mathbf{X}^*_i, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-\sum_n \sum_{[s'd']} \nabla_{\mathbf{X}^k_{i[sd]}} \mathbf{P}^{c,n}_{i[s'd']}(t) * \mathbf{f}^n_{i[s'd']}(t, \mathbf{X}^*_i, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\forall\ i,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{P}}^{c,k}_{o[sd]}(t) = -\nabla_{\mathbf{X}^k_{o[sd]}} g^c_{[sd]}(t, \mathbf{X}^*_{o[sd]}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-\sum_n \nabla_{\mathbf{X}^k_{o[sd]}} \mathbf{P}^{c,n}_{o[sd]}(t) * \mathbf{f}^n_{o[sd]}(t, \mathbf{X}^*_o, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{P}}^{c,k}_{[.d][sd]}(t) = -\nabla_{\mathbf{X}^k_{[.d][sd]}} g^c_{[.d]}(t, \mathbf{X}^*_{[.d]}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-\sum_n \sum_{[s'.]} \nabla_{\mathbf{X}^k_{[.d][sd]}} \mathbf{P}^{c,n}_{[.d][sd]}(t) * \mathbf{f}^n_{[.d][s'd]}(t, \mathbf{X}^*_{[.d]}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\phi^{c*}_{o[sd]}(t) + \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \phi^{c*}_{\pi[sd]}(t) = 1 \quad \forall\ [sd] \in \mathbf{SD}^c$$

$$\phi^{c*}_{o[sd]}(t),\ \phi^{c*}_{\pi[sd]}(t) \geq 0 \qquad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]},\ [sd] \in \mathbf{SD}^c$$

We can decompose these conditions for each source-destination pair $[sd] \in \mathbf{SD}^c$

$$\frac{\partial g^c(t, \mathbf{X}^*(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))}{\partial \phi^c_{o[sd]}} * (\phi^c_{o[sd]} - \phi^{c*}_{o[sd]}(t)) +$$

$$+ \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \frac{\partial g^c(t, \mathbf{X}^*(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))}{\partial \phi^c_{\pi[sd]}} * (\phi^c_{\pi[sd]} - \phi^{c*}_{\pi[sd]}(t)) \geq 0 \qquad \forall\ \mathbf{\Phi}^c \in \mathbf{RC}^c$$

such that

$$\begin{aligned}
\dot{\mathbf{X}}^{k*}_{ij[sd]}(t) &= \mathbf{f}^k_{ij[sd]}(t, \mathbf{X}^*_{ij}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) && \forall\ ij,\ [sd] \in \mathbf{SD}^k,\ k \\
\dot{\mathbf{X}}^{k*}_{i[sd]}(t) &= \mathbf{f}^k_{i[sd]}(t, \mathbf{X}^*_{i}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) && \forall\ i,\ [sd] \in \mathbf{SD}^k,\ k \\
\dot{\mathbf{X}}^{k*}_{o[sd]}(t) &= \mathbf{f}^k_{o[sd]}(t, \mathbf{X}^*_{o[sd]}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) && \forall\ [sd] \in \mathbf{SD}^k,\ k \\
\dot{\mathbf{X}}^{k*}_{[.d][sd]}(t) &= \mathbf{f}^k_{[.d][sd]}(t, \mathbf{X}^*_{[.d]}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) && \forall\ [sd] \in \mathbf{SD}^k,\ k \\
\mathbf{X}^{k*}_{ij[sd]}(t_0) &= \mathbf{X}^k_{ij[sd],0} && \forall\ ij,\ [sd] \in \mathbf{SD}^k,\ k \\
\mathbf{X}^{k*}_{i[sd]}(t_0) &= \mathbf{X}^k_{i[sd],0} && \forall\ i,\ [sd] \in \mathbf{SD}^k,\ k \\
\mathbf{X}^{k*}_{o[sd]}(t_0) &= \mathbf{X}^k_{o[sd],0} && \forall\ [sd] \in \mathbf{SD}^k,\ k \\
\mathbf{X}^{k*}_{[.d][sd]}(t_0) &= \mathbf{X}^k_{[.d][sd],0} && \forall\ [sd] \in \mathbf{SD}^k,\ k
\end{aligned}$$

$$\dot{\mathbf{P}}^{c,k}_{ij[sd]}(t) = -\nabla_{\mathbf{X}^k_{ij[sd]}} g^c_{ij}(t, \mathbf{X}^*_{ij}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-\sum_n \sum_{[s'd']} \nabla_{\mathbf{X}^k_{ij[sd]}} \mathbf{P}^{c,n}_{ij[s'd']}(t) * \mathbf{f}^n_{ij[s'd']}(t, \mathbf{X}^*_{ij}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\forall\ [sd] \in \mathbf{SD}^k,\ ij,\ k$$

$$\dot{\mathbf{P}}^{c,k}_{i[sd]}(t) = -\nabla_{\mathbf{X}^k_{i[sd]}} g^c_{i}(t, \mathbf{X}^*_{i}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-\sum_n \sum_{[s'd']} \nabla_{\mathbf{X}^k_{i[sd]}} \mathbf{P}^{c,n}_{i[s'd']}(t) * \mathbf{f}^n_{i[s'd']}(t, \mathbf{X}^*_{i}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\forall\ i,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{P}}^{c,k}_{o[sd]}(t) = -\nabla_{\mathbf{X}^k_{o[sd]}} g^c_{[sd]}(t, \mathbf{X}^*_{o[sd]}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-\sum_n \nabla_{\mathbf{X}^k_{o[sd]}} \mathbf{P}^{c,n}_{o[sd]}(t) * \mathbf{f}^n_{o[sd]}(t, \mathbf{X}^*_{o}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{P}}^{c,k}_{[.d][sd]}(t) = -\nabla_{\mathbf{X}^k_{[.d][sd]}} g^c_{[.d]}(t, \mathbf{X}^*_{[.d]}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-\sum_n \sum_{[s'.]} \nabla_{\mathbf{X}^k_{[.d][sd]}} \mathbf{P}^{c,n}_{[.d][sd]}(t) * \mathbf{f}^n_{[.d][s'd]}(t, \mathbf{X}^*_{[.d]}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\phi^{c*}_{o[sd]}(t) + \sum_{\pi[sd] \in \mathbf{\Pi}^c_{[sd]}} \phi^{c*}_{\pi[sd]} = 1$$

$$\phi^c_{o[sd]}(t),\ \phi^c_{\pi[sd]}(t) \geq 0 \qquad \forall\ \pi[sd] \in \mathbf{\Pi}^c_{[sd]}$$

Theorem : *Routing*

There must be flow only on minimum length paths:

$$\phi^{c*}_{\pi[sd]}(t) > 0 \quad only \quad if \qquad l^{c,Nash*}_{\pi[sd]}(t) = \min\{l^{c,Nash*}_{o[sd]}(t), \min_{p[sd]}\{l^{c,Nash*}_{p[sd]}(t)\}\}$$

$$\phi^{c*}_{\pi[sd]}(t) = 0 \qquad o.w.$$

$$\phi^{c*}_{o[sd]}(t) + \sum_{\pi[sd]\in\mathbf{\Pi}^c_{[sd]}} \phi^{c*}_{\pi[sd]}(t) = 1$$

$$\forall\ \pi[sd] \in \Pi^c_{[sd]}, \qquad [sd] \in \mathbf{SD}^c, \quad c$$

and satisfies the partial differential vectors for the state and the costate variables.

Theorem : *Congestion Control*

Flow is not admitted into the network only if its rejection length is less than the minimum length path to its destination:

$$\phi^{c*}_{o[sd]}(t) > 0 \quad only \quad if \qquad l^{c,Nash}_{o[sd]}(t) = \min\{l^{c,Nash*}_{o[sd]}(t), \min_{p[sd]}\{l^{c,Nash*}_{p[sd]}(t)\}\}$$

$$\phi^{c*}_{\pi[sd]} = 0 \qquad o.w.$$

$$\phi^{c*}_{o[sd]}(t) + \sum_{\pi[sd]\in\mathbf{\Pi}^c_{[sd]}} \phi^{c*}_{\pi[sd]}(t) = 1 \quad \forall\ [sd] \in \mathbf{SD}^c, \quad c$$

and satisfies the partial differential vectors for the state and the costate variables.

Having found the optimum routing and congestion control decisions, we proceed to solve the load sharing problem for class c assuming also that all other classes act at their optimum decisions. So, the load sharing problem for class c is

$$minimize \qquad \int_{t_0}^{tf} g^c(t, \mathbf{X}(t), \begin{matrix} \mathbf{\Phi}^{1*}(t), ..., \mathbf{\Phi}^{c*}(t), ..., \mathbf{\Phi}^{C*}(t) \\ \mathbf{\Psi}^{1*}(t), ..., \mathbf{\Psi}^{c}(t), ..., \mathbf{\Psi}^{C*}(t) \end{matrix})dt =$$

$$= \sum_{ij} \int_{t_0}^{tf} g_{ij}^c(t, \mathbf{X}_{ij}(t), \lambda_{ij}^{1*}(t), ..., \lambda_{ij}^{c}(t), ..., \lambda_{ij}^{C*}(t))dt \; +$$

$$+ \sum_{i} \int_{t_0}^{tf} g_{i}^c(t, \mathbf{X}_{i}(t), \lambda_{i}^{1*}(t), ..., \lambda_{i}^{c}(t), ..., \lambda_{i}^{C*}(t))dt \; +$$

$$+ \sum_{[sd]} \int_{t_0}^{tf} g_{[sd]}^c(t, \mathbf{X}_{o[sd]}(t), \lambda_{o[sd]}^{1*}(t), ..., \lambda_{o[sd]}^{c}(t), ..., \lambda_{o[sd]}^{C*}(t))dt \; +$$

$$+ \sum_{[.d]} \int_{t_0}^{tf} g_{[.d]}^c(t, \mathbf{X}_{[.d]}(t), \lambda_{[.d]}^{1*}(t), ..., \lambda_{[.d]}^{c}(t), ..., \lambda_{[.d]}^{C*}(t))dt$$

with respect to $\mathbf{\Psi}^c(t)$

such that

$$\dot{\mathbf{X}}^k_{ij[sd]}(t) = \mathbf{f}^k_{ij[sd]}(t, \mathbf{X}_{ij}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t)) \quad \forall\ ij,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{X}}^k_{i[sd]}(t) = \mathbf{f}^k_{i[sd]}(t, \mathbf{X}_{i}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t)) \quad \forall\ i,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{X}}^k_{o[sd]}(t) = \mathbf{f}^k_{o[sd]}(t, \mathbf{X}_{o[sd]}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t)) \quad \forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{X}}^k_{[.d][sd]}(t) = \mathbf{f}^k_{[.d][sd]}(t, \mathbf{X}_{[.d]}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t)) \quad \forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\mathbf{X}^k_{ij[sd]}(t_0) = \mathbf{X}^k_{ij[sd],0} \quad \forall\ ij,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\mathbf{X}^k_{i[sd]}(t_0) = \mathbf{X}^k_{i[sd],0} \quad \forall\ i,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\mathbf{X}^k_{o[sd]}(t_0) = \mathbf{X}^k_{o[sd],0} \quad \forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\mathbf{X}^k_{[.d][sd]}(t_0) = \mathbf{X}^k_{[.d][sd],0} \quad \forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\sum_{[.d] \in \mathbf{D}^c_{[s.]}} \psi^c_{[sd]}(t) = 1 \quad \forall\ [s.] \in \mathbf{S}^c$$

$$\psi^c_{[sd]}(t) \geq 0 \quad \forall\ [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c$$

The necessary optimality conditions for class c are:

$$\frac{\partial g^c(t, \mathbf{X}^*(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))}{\partial \psi^c_{[sd]}} * (\psi^c_{[sd]} - \psi^{c*}_{[sd]}(t)) \quad \geq 0 \qquad \forall\ \mathbf{\Psi}^c \in \mathbf{LS}^c$$

such

$$\dot{\mathbf{X}}^{k*}_{ij[sd]}(t) = \mathbf{f}^k_{ij[sd]}(t, \mathbf{X}^*_{ij}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) \qquad \forall\ ij,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{X}}^{k*}_{i[sd]}(t) = \mathbf{f}^k_{i[sd]}(t, \mathbf{X}^*_{i}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) \qquad \forall\ i,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{X}}^{k*}_{o[sd]}(t) = \mathbf{f}^k_{o[sd]}(t, \mathbf{X}^*_{o[sd]}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) \qquad \forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{X}}^{k*}_{[.d][sd]}(t) = \mathbf{f}^k_{[.d][sd]}(t, \mathbf{X}^*_{[.d]}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) \qquad \forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\mathbf{X}^{k*}_{ij[sd]}(t_0) = \mathbf{X}^k_{ij[sd],0} \qquad \forall\ ij,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\mathbf{X}^{k*}_{i[sd]}(t_0) = \mathbf{X}^k_{i[sd],0} \qquad \forall\ i,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\mathbf{X}^{k*}_{o[sd]}(t_0) = \mathbf{X}^k_{o[sd],0} \qquad \forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\mathbf{X}^{k*}_{[.d][sd]}(t_0) = \mathbf{X}^k_{[.d][sd],0} \qquad \forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{P}}^{c,k}_{ij[sd]}(t) = -\nabla_{\mathbf{X}^k_{ij[sd]}} g^c_{ij}(t, \mathbf{X}^*_{ij}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-\sum_n \sum_{[s'd']} \nabla_{\mathbf{X}^k_{ij[sd]}} \mathbf{P}^{c,n}_{ij[s'd']}(t) * \mathbf{f}^n_{ij[s'd']}(t, \mathbf{X}^*_{ij}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\forall\ ij,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{P}}^{c,k}_{i[sd]}(t) = -\nabla_{\mathbf{X}^k_{i[sd]}} g^c_{i}(t, \mathbf{X}^*_{i}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-\sum_n \sum_{[s'd']} \nabla_{\mathbf{X}^k_{i[sd]}} \mathbf{P}^{c,n}_{i[s'd']}(t) * \mathbf{f}^n_{i[s'd']}(t, \mathbf{X}^*_{i}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\forall\ i,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{P}}^{c,k}_{o[sd]}(t) = -\nabla_{\mathbf{X}^k_{o[sd]}} g^c_{[sd]}(t, \mathbf{X}^*_{o[sd]}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-\sum \nabla_{\mathbf{X}^k_{o[sd]}} \mathbf{P}^{c,n}_{o[sd]}(t) * \mathbf{f}^n_{o[sd]}(t, \mathbf{X}^*_{o}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{P}}^{c,k}_{[.d][sd]}(t) = -\nabla_{\mathbf{X}^k_{[.d][sd]}} g^c_{[.d]}(t, \mathbf{X}^*_{[.d]}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-\sum_n \sum_{[s'.]} \nabla_{\mathbf{X}^k_{[.d][sd]}} \mathbf{P}^{c,n}_{[.d][sd]}(t) * \mathbf{f}^n_{[.d][s'd]}(t, \mathbf{X}^*_{[.d]}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\sum_{[.d] \in \mathbf{D}^c_{[s.]}} \psi^{c*}_{[sd]}(t) = 1 \quad \forall\ [s.] \in \mathbf{S}^c$$

$$\psi^{c*}_{[sd]}(t) \geq 0 \qquad \forall\ [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c$$

We can decompose these conditions for each source node $[s.] \in \mathbf{S}^c$

$$\sum_{[.d] \in \mathbf{D}^c_{[s.]}} \frac{\partial g^c(t, \mathbf{X}^*(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))}{\partial \psi^c_{[sd]}} * (\psi^c_{[sd]} - \psi'^{c*}_{[sd]}(t)) \quad \geq 0 \qquad \forall\ \mathbf{\Psi}^c \in \mathbf{LS}^c$$

such that

$$\dot{\mathbf{X}}^{k*}_{ij[sd]}(t) = \mathbf{f}^k_{ij[sd]}(t, \mathbf{X}^*_{ij}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) \qquad \forall\ ij,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{X}}^{k*}_{i[sd]}(t) = \mathbf{f}^k_{i[sd]}(t, \mathbf{X}^*_{i}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) \qquad \forall\ i,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{X}}^{k*}_{o[sd]}(t) = \mathbf{f}^k_{o[sd]}(t, \mathbf{X}^*_{o[sd]}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) \qquad \forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{X}}^{k*}_{[.d][sd]}(t) = \mathbf{f}^k_{[.d][sd]}(t, \mathbf{X}^*_{[.d]}(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) \qquad \forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\mathbf{X}^{k*}_{ij[sd]}(t_0) = \mathbf{X}^k_{ij[sd],0} \qquad \forall\ ij,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\mathbf{X}^{k*}_{i[sd]}(t_0) = \mathbf{X}^k_{i[sd],0} \qquad \forall\ i,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\mathbf{X}^{k*}_{o[sd]}(t_0) = \mathbf{X}^k_{o[sd],0} \qquad \forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\mathbf{X}^{k*}_{[.d][sd]}(t_0) = \mathbf{X}^k_{[.d][sd],0} \qquad \forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{P}}^{c,k}_{ij[sd]}(t) = -\nabla_{\mathbf{X}^k_{ij[sd]}} g^c_{ij}(t, \mathbf{X}^*_{ij}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-\sum_n \sum_{[s'd']} \nabla_{\mathbf{X}^k_{ij[sd]}} \mathbf{P}^{c,n}_{ij[s'd']}(t) * \mathbf{f}^n_{ij[s'd']}(t, \mathbf{X}^*_{ij}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\forall\ ij,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{P}}^{c,k}_{i[sd]}(t) = -\nabla_{\mathbf{X}^k_{i[sd]}} g^c_{i}(t, \mathbf{X}^*_{i}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-\sum_n \sum_{[s'd']} \nabla_{\mathbf{X}^k_{i[sd]}} \mathbf{P}^{c,n}_{i[s'd']}(t) * \mathbf{f}^n_{i[s'd']}(t, \mathbf{X}^*_{i}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\forall\ i,\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{P}}^{c,k}_{o[sd]}(t) = -\nabla_{\mathbf{X}^k_{o[sd]}} g^c_{[sd]}(t, \mathbf{X}^*_{o[sd]}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-\sum_n \nabla_{\mathbf{X}^k_{o[sd]}} \mathbf{P}^{c,n}_{o[sd]}(t) * \mathbf{f}^n_{o[sd]}(t, \mathbf{X}^*_{o}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\dot{\mathbf{P}}^{c,k}_{[.d][sd]}(t) = -\nabla_{\mathbf{X}^k_{[.d][sd]}} g^c_{[.d]}(t, \mathbf{X}^*_{[.d]}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) -$$

$$-\sum_n \sum_{[s'.]} \nabla_{\mathbf{X}^k_{[.d][sd]}} \mathbf{P}^{c,n}_{[.d][sd]}(t) * \mathbf{f}^n_{[.d][s'd]}(t, \mathbf{X}^*_{[.d]}, \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\forall\ [sd] \in \mathbf{SD}^k,\ k$$

$$\sum_{[.d] \in \mathbf{D}^c_{[s.]}} \psi^{c*}_{[sd]}(t) = 1 \quad \forall\ [s.] \in \mathbf{S}^c$$

$$\psi^{c*}_{[sd]}(t) \geq 0 \qquad \forall\ [.d] \in \mathbf{D}^c_{[s.]}$$

Theorem : *Load Sharing*

For each source, there must be flow only to destinations whose length is minimum:

$$\psi^{c*}_{[sd]}(t) > 0 \quad only \quad if \qquad l^{c,Nash*}_{[sd]}(t) = \min_{[sd']}\{l^{c,Nash*}_{[sd']}(t)\}$$

$$\psi^{c*}_{[sd]}(t) = 0 \qquad\qquad o.w.$$

$$\sum_{[.d]\in \mathbf{S}^c_{[s.]}} \psi^{c*}_{[sd]}(t) = 1 \quad \forall\, [.d] \in \mathbf{D}^c_{[s.]},\ [s.] \in \mathbf{S}^c,\ c$$

and satisfies the partial differential vectors for the state and the costate variables.

So, in this section we have formulated and solved the dynamic join load sharing, routing and congestion control problem as a dynamic Nash game among multiple competing classes.

5.3 Stackelberg Equilibrium Solution

In this section, we formulate the dynamic join load sharing, routing and congestion control problem in distributed systems with two classes of jobs, one more powerful than the other, as a non-cooperative dynamic Stackelberg game. An example of such classes of jobs is when they have different priorities. Another example is when there is a system administrator (leader) and users (followers) with different objectives and power.

Customers of the most powerful class try to use the resources of the distributed system for their own benefit, ignoring the inconvenience that they cause to customers from the less powerful class.

Next, we briefly survey research on the dynamic Stackelberg game theory:

Starr & Ho [464] introduce nonzero-sum differential games and discuss Nash equilibrium, minimax and noninferior strategies. Then they solve the linear-quadratic game.

Chen & Cruz [96] analyze Stackelberg games with biased information. They present necessary conditions for open-loop strategies and use dynamic programming to define feedback strategies. Simaan & Cruz [449, 448] derive necessary and sufficient conditions for Stackelberg games. They also solve the linear-quadratic problem. Cruz [117] considers hierarchical games with multiple players at each level.

Basar & Selbuz [28, 29] consider linear-quadratic Stackelberg games. They derive a linear one-step memory closed-loop solution for the leader and a linear feedback solution for the follower. Basar [25] obtains the sufficient conditions for a three-player hierarchical game. Then he applies them to linear-quadratic games.

Papavassilopoulos & Cruz [376] analyze Stackelberg dynamic games, which are nonclassical control problems, since the control depends both on the state and time and its partial derivative with respect to the state appears in the state equation and in the cost function. They also [375] derive sufficient conditions for Stackelberg and Nash strategies for linear quadratic deterministic differential games when the players have memory.

In the following, we shall develop a methodology for the joint dynamic load sharing, routing and congestion control problem based on the Stackelberg game theory.

Next, we give some definitions for a two-hierarchical-class game similar to those in [27] for Stackelberg games:

Definition :

In a two class join load sharing, routing and congestion control problem, with the most powerful class α as the leader and the less powerful class β as the follower, the set $\mathcal{R}^{\beta}(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha})$, defined for the class α strategy $(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}) \in (\mathbf{RC}^{\alpha}, \mathbf{LS}^{\alpha})$, by:

$$\begin{aligned}\mathcal{R}^{\beta}(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}) = \{ \quad & (\mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta}) \in (\mathbf{RC}^{\beta}, \mathbf{\Psi}^{\beta}) \ such\ that: \\ & J^{\beta}(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}, \mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta}) \leq J^{\beta}(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}, \hat{\mathbf{\Phi}}^{\beta}, \hat{\mathbf{\Psi}}^{\beta}), \\ & \forall \ (\hat{\mathbf{\Phi}}^{\beta}, \hat{\mathbf{\Psi}}^{\beta}), \ \ such\ \ that \ \ (\hat{\mathbf{\Phi}}^{\beta}, \hat{\mathbf{\Psi}}^{\beta}) \in (\mathbf{RC}^{\beta}, \mathbf{LS}^{\beta})\}\end{aligned}$$

is the optimal response (rational reaction) set of the less powerful class β to the strategy of the most powerful class α.

What the above definition says is that the less powerful class β chooses its decision vector $(\mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta})$, that minimizes its cost function $J^{\beta}(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}, \mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta})$, for given strategy $(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha})$ of the most powerful class α.

Definition :

In a two class join load sharing, routing and congestion control problem with the most powerful class α as the leader, a strategy $(\mathbf{\Phi}^{\alpha*}, \mathbf{\Psi}^{\alpha*}) \in (\mathbf{RC}^{\alpha}, \mathbf{LS}^{\alpha})$ is called a Stackelberg equilibrium strategy for the most powerful class α if and only if

$$\inf_{(\mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta}) \in \mathcal{R}^{\beta}(\mathbf{\Phi}^{\alpha*}, \mathbf{\Psi}^{\alpha*})} J^{\alpha}(\mathbf{\Phi}^{\alpha*}, \mathbf{\Psi}^{\alpha*}, \mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta}) \leq$$

$$\leq \inf_{(\mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta}) \in \mathcal{R}^{\beta}(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha})} J^{\alpha}(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}, \mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta}) \quad \forall \ (\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}) \in (\mathbf{RC}^{\alpha}, \mathbf{LS}^{\alpha})$$

This means that the most powerful class α chooses its strategy $(\mathbf{\Phi}^{\alpha*}, \mathbf{\Psi}^{\alpha*})$ that minimizes its cost function $J^{\alpha}(\mathbf{\Phi}^{\alpha}, \mathbf{\Psi}^{\alpha}, \mathbf{\Phi}^{\beta}, \mathbf{\Psi}^{\beta})$, given the optimal response set $\mathcal{R}^{\beta}(\mathbf{\Phi}^{\alpha*}, \mathbf{\Psi}^{\alpha*})$ of the less powerful class β to its strategy $(\mathbf{\Phi}^{\alpha*}, \mathbf{\Psi}^{\alpha*})$.

Definition :

Let $(\mathbf{\Phi}^{\alpha*}, \mathbf{\Psi}^{\alpha*}) \in (\mathbf{RC}^{\alpha}, \mathbf{LS}^{\alpha})$ *be a Stackelberg strategy for the most powerful class* α. *Then any element* $(\mathbf{\Phi}^{\beta*}, \mathbf{\Psi}^{\beta*}) \in \mathcal{R}^{\beta}(\mathbf{\Phi}^{\alpha*}, \mathbf{\Psi}^{\alpha*})$ *is an optimal strategy for the less powerful class* β *that is in equilibrium with* $(\mathbf{\Phi}^{\alpha*}, \mathbf{\Psi}^{\alpha*})$. *The strategy* $(\mathbf{\Phi}^{\alpha*}, \mathbf{\Psi}^{\alpha*}, \mathbf{\Phi}^{\beta*}, \mathbf{\Psi}^{\beta*})$ *is a* Stackelberg solution *for the game with the most powerful class* α *as the leader and the cost pair* $J^{\alpha}(\mathbf{\Phi}^{\alpha*}, \mathbf{\Psi}^{\alpha*}, \mathbf{\Phi}^{\beta*}, \mathbf{\Psi}^{\beta*})$, $J^{\beta}(\mathbf{\Phi}^{\alpha*}, \mathbf{\Psi}^{\alpha*}, \mathbf{\Phi}^{\beta*}, \mathbf{\Psi}^{\beta*})$ *is the corresponding Stackelberg equilibrium outcome.*

5.3.1 Optimal Control Formulation

In this section, we formulate the dynamic non-cooperative join load sharing, routing and congestion control problem as an Optimal Control Problem (OCP).

Theorem :

Consider the dynamic join load sharing, routing and congestion control problem in distributed systems with two hierarchical classes, with fixed initial time t_0 *and final time* t_f.

If for each class c, $H^c(t, \mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi}, \mathbf{P}(t))$ *is differentiable and convex in* $(\mathbf{X}, \mathbf{\Phi}^c, \mathbf{\Psi}^c) \in (\mathbf{R}, \mathbf{RC}^c, \mathbf{LS}^c)$ $\forall\ t \in [t_0, t_f]$, *for each fixed value of* $(\mathbf{\Phi}^k, \mathbf{\Psi}^k) \in (\mathbf{RC}^k, \mathbf{LS}^k)$,

then $(\mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) \in (\mathbf{RC}, \mathbf{LS})$ *is a Stackelberg equilibrium if and only if it solves the following* *Optimal Control Problem:*

$$minimize \qquad \int_{t_0}^{t_f} g^{\alpha}(t, \mathbf{X}(t), \mathbf{\Phi}^{\alpha}(t), \mathbf{\Psi}^{\alpha}(t), \mathbf{\Phi}^{\beta}(t), \mathbf{\Psi}^{\beta}(t))dt$$

$$with\ respect\ to \quad (\mathbf{\Phi}^{\alpha}(t), \mathbf{\Psi}^{\alpha}(t), \mathbf{\Phi}^{\beta}(t), \mathbf{\Psi}^{\beta}(t)))$$

$$such\ that \qquad \dot{\mathbf{X}}(t) = \mathbf{f}(t, \mathbf{X}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t))$$

$$\mathbf{X}(t_0) = \mathbf{X}_0$$

$$(\mathbf{\Phi}^{\alpha}(t), \mathbf{\Psi}^{\alpha}(t)) \in (\mathbf{RC}^{\alpha}, \mathbf{LS}^{\alpha})$$

$$(\mathbf{\Phi}^{\beta}(t), \mathbf{\Psi}^{\beta}(t)) \in (\mathbf{RC}^{\beta}, \mathbf{LS}^{\beta})$$

$$\int_{t_0}^{t_f} g^{\beta}(t, \mathbf{X}(t), \begin{matrix} \mathbf{\Phi}^{\alpha}(t), \mathbf{\Phi}^{\beta}(t) \\ \mathbf{\Psi}^{\alpha}(t), \mathbf{\Psi}^{\beta}(t) \end{matrix})dt =$$

$$= \min_{(\hat{\mathbf{\Phi}}^{\beta}(t), \hat{\mathbf{\Psi}}^{\beta}(t)) \in (\mathbf{RC}^{\beta}, \mathbf{LS}^{\beta})} \int_{t_0}^{t_f} g^{\beta}(t, \mathbf{X}(t), \begin{matrix} \mathbf{\Phi}^{\alpha}(t), \hat{\mathbf{\Phi}}^{\beta}(t) \\ \mathbf{\Psi}^{\alpha}(t), \hat{\mathbf{\Psi}}^{\beta}(t) \end{matrix})dt$$

Proof: It follows from the definition of the Stackelberg equilibrium.□

Theorem :

Consider the dynamic join load sharing, routing and congestion control problem in distributed systems with two hierarchical classes, with fixed initial time t_0 *and final time* t_f.

Let for each class c, $H^c(t, \mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi}, \mathbf{P}(t))$ *is differentiable and convex in* $(\mathbf{X}, \mathbf{\Phi}^c, \mathbf{\Psi}^c) \in (\mathbf{R}, \mathbf{RC}^c, \mathbf{LS}^c) \quad \forall\ t \in [t_0, t_f]$, *for each fixed value of* $(\mathbf{\Phi}^k, \mathbf{\Psi}^k) \in (\mathbf{RC}^k, \mathbf{LS}^k)$.

If $(\hat{\mathbf{\Phi}}^*(t, \mathbf{X}_0), \hat{\mathbf{\Psi}}^*(t, \mathbf{X}_0)) = (\mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) \in (\mathbf{RC}, \mathbf{LS})$ *is an* <u>*open-loop Stackelberg equilibrium*</u> *and* $\{\mathbf{X}^*(t),\ t \in [t_0, t_f]\}$ *is the corresponding state trajectory, then* $\exists\ \mathbf{P}^c(t) : [t_0, t_f] \rightarrow \mathbf{R}^n$, $\forall\ c$ *continuous and piecewise continuously differentiable vector functions, such that* $\forall\ t \in [t_0, t_f]$:

$$\textit{minimize} \qquad \int_{t_0}^{t_f} g^{\alpha}(t, \mathbf{X}(t), \mathbf{\Phi}^{\alpha}(t), \mathbf{\Psi}^{\alpha}(t), \mathbf{\Phi}^{\beta}(t), \mathbf{\Psi}^{\beta}(t))dt$$

$$\textit{with respect to} \quad (\mathbf{\Phi}^{\alpha}(t), \mathbf{\Psi}^{\alpha}(t), \mathbf{\Phi}^{\beta}(t), \mathbf{\Psi}^{\beta}(t), \mathbf{Q}^{\beta}(t))$$

$$\textit{such that} \qquad \dot{\mathbf{X}}(t) = \mathbf{f}(t, \mathbf{X}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t))$$

$$\mathbf{X}(t_0) = \mathbf{X}_0$$

$$\left[\frac{\partial H^{\beta}}{\partial \phi^{\beta}_{o[sd]}} - Q^{\beta}_{[sd]}(t)\right] * \phi^{\beta}_{o[sd]}(t) = 0 \quad \forall\ [sd] \in \mathbf{SD}^{\beta}$$

$$\left[\frac{\partial H^{\beta}}{\partial \phi^{\beta}_{\pi[sd]}} - Q^{\beta}_{[sd]}(t)\right] * \phi^{\beta}_{\pi[sd]}(t) = 0 \quad \forall\ \pi[sd] \in \mathbf{\Pi}^{\beta}_{[sd]},\ [sd] \in \mathbf{SD}^{\beta}$$

$$\left[\frac{\partial H^{\beta}}{\partial \psi^{\beta}_{[sd]}} - Q^{\beta}_{[s.]}(t)\right] * \psi^{\beta}_{[sd]}(t) = 0 \quad \forall\ [.d] \in \mathbf{D}^{\beta}_{[s.]},\ [s.] \in \mathbf{S}^{\beta}$$

$$\frac{\partial H^{\beta}}{\partial \phi^{\beta}_{o[sd]}} - Q^{\beta}_{[sd]}(t) \geq 0 \quad \forall\ [sd] \in \mathbf{SD}^{\beta}$$

$$\frac{\partial H^{\beta}}{\partial \phi^{\beta}_{\pi[sd]}} - Q^{\beta}_{[sd]}(t) \geq 0 \quad \forall\ \pi[sd] \in \mathbf{\Pi}^{\beta}_{[sd]},\ [sd] \in \mathbf{SD}^{\beta}$$

$$\frac{\partial H^{\beta}}{\partial \psi^{\beta}_{[sd]}} - Q^{\beta}_{[s.]}(t) \geq 0 \quad \forall\ [.d] \in \mathbf{D}^{\beta}_{[s.]},\ [s.] \in \mathbf{S}^{\beta}$$

$$\dot{\mathbf{P}}^{\beta}(t) = -\nabla_{\mathbf{X}} H^{\beta}(t, \mathbf{X}, \mathbf{\Phi}(t), \mathbf{\Psi}(t), \mathbf{P}^{\beta}(t))$$

$$\mathbf{P}^{\beta}(t_f) = \mathbf{0}$$

$$\phi^{\alpha}_{o[sd]}(t) + \sum_{\pi[sd] \in \mathbf{\Pi}^{\alpha}_{[sd]}} \phi^{\alpha}_{\pi[sd]}(t) = 1 \quad \forall\ [sd] \in \mathbf{SD}^{\alpha}$$

$$\sum_{[.d] \in \mathbf{D}^{\alpha}_{[s.]}} \psi^{\alpha}_{[sd]}(t) = 1 \qquad \forall\ [s.] \in \mathbf{S}^{\alpha}$$

$$\phi^{\alpha}_{o[sd]}(t),\ \phi^{\alpha}_{\pi[sd]}(t) \geq 0 \qquad \forall\ \pi[sd] \in \mathbf{\Pi}^{\alpha}_{[sd]},\ [sd] \in \mathbf{SD}^{\alpha}$$

$$\psi^{\alpha}_{[sd]}(t) \geq 0 \qquad \forall\ [.d] \in \mathbf{D}^{\alpha}_{[s.]},\ [s.] \in \mathbf{S}^{\alpha}$$

$$\phi^{\beta}_{o[sd]}(t) + \sum_{\pi[sd] \in \mathbf{\Pi}^{\beta}_{[sd]}} \phi^{\beta}_{\pi[sd]}(t) = 1 \quad \forall\ [sd] \in \mathbf{SD}^{\beta}$$

$$\sum_{[.d] \in \mathbf{D}^{\beta}_{[s.]}} \psi^{\beta}_{[sd]}(t) = 1 \qquad \forall\ [s.] \in \mathbf{S}^{\beta}$$

$$\phi^{\beta}_{o[sd]}(t),\ \phi^{\beta}_{\pi[sd]}(t) \geq 0 \qquad \forall\ \pi[sd] \in \mathbf{\Pi}^{\beta}_{[sd]},\ [sd] \in \mathbf{SD}^{\beta}$$

$$\psi^{\beta}_{[sd]}(t) \geq 0 \qquad \forall\ [.d] \in \mathbf{D}^{\beta}_{[s.]},\ [s.] \in \mathbf{S}^{\beta}$$

Proof: The Lagrangian for the less powerful class β is

$$L^{\beta} = H^{\beta} + \sum_{[sd] \in \mathbf{SD}^{\beta}} Q^{\beta}_{[sd]} * \left[1 - \phi^{\beta}_{o[sd]} - \sum_{\pi[sd] \in \mathbf{\Pi}^{\beta}_{[sd]}} \phi^{\beta}_{\pi[sd]} \right] + \sum_{[s.] \in \mathbf{S}^{\beta}} Q^{\beta}_{[s.]} * \left[1 - \sum_{[.d] \in \mathbf{D}^{\beta}_{[s.]}} \psi^{\beta}_{[sd]} \right]$$

with $\phi^{\beta}_{o[sd]},\ \phi^{\beta}_{\pi[sd]},\ \psi^{\beta}_{[sd]} \geq 0 \quad \forall\ \pi[sd] \in \mathbf{\Pi}^{\beta}_{[sd]},\ [sd] \in \mathbf{SD}^{\beta}$

Pontryagin's maximum principle necessary conditions are:

$$\dot{\mathbf{X}}^*(t) = \mathbf{f}(t, \mathbf{X}^*(t), \mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t))$$

$$\mathbf{X}^*(t_0) = \mathbf{X}_0$$

$$\frac{\partial L^{\beta*}}{\partial \phi^{\beta}_{o[sd]}} * \phi^{\beta*}_{o[sd]}(t) = 0 \;\Rightarrow\; \left[\frac{\partial H^{\beta*}}{\partial \phi^{\beta}_{o[sd]}} - Q^{\beta}_{[sd]}(t)\right] * \phi^{\beta*}_{o[sd]}(t) = 0 \quad \forall\, [sd] \in \mathbf{SD}^{\beta}$$

$$\frac{\partial L^{\beta*}}{\partial \phi^{\beta}_{\pi[sd]}} * \phi^{\beta*}_{\pi[sd]}(t) = 0 \;\Rightarrow\; \left[\frac{\partial H^{\beta*}}{\partial \phi^{\beta}_{\pi[sd]}} - Q^{\beta}_{[sd]}(t)\right] * \phi^{\beta*}_{\pi[sd]}(t) = 0$$

$$\forall\, \pi[sd] \in \mathbf{\Pi}^{\beta}_{[sd]},\ [sd] \in \mathbf{SD}^{\beta}$$

$$\frac{\partial L^{\beta*}}{\partial \psi^{\beta}_{[sd]}} * \psi^{\beta*}_{[sd]}(t) = 0 \;\Rightarrow\; \left[\frac{\partial H^{\beta*}}{\partial \psi^{\beta}_{[sd]}} - Q^{\beta}_{[s.]}(t)\right] * \psi^{\beta*}_{[sd]}(t) = 0$$

$$\forall\, [.d] \in \mathbf{D}^{\beta}_{[s.]},\ [s.] \in \mathbf{S}^{\beta}$$

$$\frac{\partial L^{\beta*}}{\partial \phi^{\beta}_{o[sd]}} \geq 0 \;\Rightarrow\; \frac{\partial H^{\beta*}}{\partial \phi^{\beta}_{o[sd]}} - Q^{\beta}_{[sd]}(t) \geq 0 \quad \forall\, [sd] \in \mathbf{SD}^{\beta}$$

$$\frac{\partial L^{\beta*}}{\partial \phi^{\beta}_{\pi[sd]}} \geq 0 \;\Rightarrow\; \frac{\partial H^{\beta*}}{\partial \phi^{\beta}_{\pi[sd]}} - Q^{\beta}_{[sd]}(t) \geq 0 \quad \forall\, \pi[sd] \in \mathbf{\Pi}^{\beta}_{[sd]},\ [sd] \in \mathbf{SD}^{\beta}$$

$$\frac{\partial L^{\beta*}}{\partial \psi^{\beta}_{[sd]}} \geq 0 \;\Rightarrow\; \frac{\partial H^{\beta*}}{\partial \psi^{\beta}_{[sd]}} - Q^{\beta}_{[s.]}(t) \geq 0 \quad \forall\, [.d] \in \mathbf{D}^{\beta}_{[s.]},\ [s.] \in \mathbf{S}^{\beta}$$

$$\dot{\mathbf{P}}^{\beta}(t) = -\nabla_{\mathbf{X}} H^{\beta}(t, \mathbf{X}^{*}, \mathbf{\Phi}^{*}(t), \mathbf{\Psi}^{*}(t), \mathbf{P}^{\beta}(t))$$

$$\mathbf{P}^{\beta}(t_f) = \mathbf{0}$$

$$\frac{\partial L^{\beta *}}{\partial Q^{\beta}_{[sd]}} = 0 \;\Rightarrow \phi^{\beta *}_{o[sd]}(t) + \sum_{\pi[sd] \in \mathbf{\Pi}^{\beta}_{[sd]}} \phi^{\beta *}_{\pi[sd]}(t) = 1 \quad \forall\, [sd] \in \mathbf{SD}^{\beta}$$

$$\frac{\partial L^{\beta *}}{\partial Q^{\beta}_{[s.]}} = 0 \;\Rightarrow \sum_{[.d] \in \mathbf{D}^{\beta}_{[s.]}} \psi^{\beta *}_{[sd]}(t) = 1 \qquad \forall\, [s.] \in \mathbf{S}^{\beta}$$

$$\phi^{\beta *}_{o[sd]}(t),\ \phi^{\beta *}_{\pi[sd]}(t) \geq 0 \qquad \forall\, \pi[sd] \in \mathbf{\Pi}^{\beta}_{[sd]},\ [sd] \in \mathbf{SD}^{\beta}$$

$$\psi^{\beta *}_{[sd]}(t) \geq 0 \qquad \forall\, [.d] \in \mathbf{D}^{\beta}_{[s.]},\ [s.] \in \mathbf{S}^{\beta}$$

□

Theorem :

Consider the dynamic join load sharing, routing and congestion control problem in distributed systems with two hierarchical classes, with fixed initial time t_0 *and final time* t_f.

Let for each class c, $g^{c}(t, \mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi})$, $\mathbf{f}(t, \mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi})$, *are continuously differentiable with respect to* $(\mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi}) \in (\mathbf{R}^{n}, \mathbf{RC}, \mathbf{LS})$, $\forall\, t \in [t_0, t_f]$.

If $(\hat{\mathbf{\Phi}}^{*}(t, \mathbf{X}, \mathbf{X}_0), \hat{\mathbf{\Psi}}^{*}(t, \mathbf{X}, \mathbf{X}_0)) = (\mathbf{\Phi}^{*}(t), \mathbf{\Psi}^{*}(t)) \in (\mathbf{RC}, \mathbf{LS})$ *is a closed-loop memoryless Stackelberg equilibrium such that* $(\hat{\mathbf{\Phi}}^{c*}(t, \mathbf{X}, \mathbf{X}_0), \hat{\mathbf{\Phi}}^{c*}(t, \mathbf{X}, \mathbf{X}_0))$ *is continuously differentiable with respect to* $\mathbf{X} \in \mathbf{R}^{n}$, $\forall\, c$, $t \in [t_0, t_f]$ *and* $\{\mathbf{X}^{*}(t),\ t \in [t_0, t_f]\}$ *is the corresponding state trajectory, then* $\exists\ \mathbf{P}^{c}(t) : [t_0, t_f] \rightarrow \mathbf{R}^{n}$, $\forall\, c$, *continuous and piecewise continuously differentiable vector functions, such that* $\forall\, t \in [t_0, t_f]$:

$$\textit{minimize} \qquad \int_{t_0}^{t_f} g^{\alpha}(t, \mathbf{X}(t), \mathbf{\Phi}^{\alpha}(t), \mathbf{\Psi}^{\alpha}(t), \mathbf{\Phi}^{\beta}(t), \mathbf{\Psi}^{\beta}(t))dt$$

$$\textit{with respect to} \quad (\mathbf{\Phi}^{\alpha}(t), \mathbf{\Psi}^{\alpha}(t), \mathbf{\Phi}^{\beta}(t), \mathbf{\Psi}^{\beta}(t), \mathbf{Q}^{\beta}(t))$$

$$\textit{such that} \qquad \dot{\mathbf{X}}(t) = \mathbf{f}(t, \mathbf{X}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t))$$

$$\mathbf{X}(t_0) = \mathbf{X}_0$$

$$\left[\frac{\partial H^{\beta}}{\partial \phi^{\beta}_{o[sd]}} - Q^{\beta}_{[sd]}(t)\right] * \phi^{\beta}_{o[sd]}(t) = 0 \quad \forall\ [sd] \in \mathbf{SD}^{\beta}$$

$$\left[\frac{\partial H^{\beta}}{\partial \phi^{\beta}_{\pi[sd]}} - Q^{\beta}_{[sd]}(t)\right] * \phi^{\beta}_{\pi[sd]}(t) = 0 \quad \forall\ \pi[sd] \in \mathbf{\Pi}^{\beta}_{[sd]},\ [sd] \in \mathbf{SD}^{\beta}$$

$$\left[\frac{\partial H^{\beta}}{\partial \psi^{\beta}_{[sd]}} - Q^{\beta}_{[s.]}(t)\right] * \psi^{\beta}_{[sd]}(t) = 0 \quad \forall\ [.d] \in \mathbf{D}^{\beta}_{[s.]},\ [s.] \in \mathbf{S}^{\beta}$$

$$\frac{\partial H^{\beta}}{\partial \phi^{\beta}_{o[sd]}} - Q^{\beta}_{[sd]}(t) \geq 0 \quad \forall\ [sd] \in \mathbf{SD}^{\beta}$$

$$\frac{\partial H^{\beta}}{\partial \phi^{\beta}_{\pi[sd]}} - Q^{\beta}_{[sd]}(t) \geq 0 \quad \forall\ \pi[sd] \in \mathbf{\Pi}^{\beta}_{[sd]},\ [sd] \in \mathbf{SD}^{\beta}$$

$$\frac{\partial H^{\beta}}{\partial \psi^{\beta}_{[sd]}} - Q^{\beta}_{[s.]}(t) \geq 0 \quad \forall\ [.d] \in \mathbf{D}^{\beta}_{[s.]},\ [s.] \in \mathbf{S}^{\beta}$$

$$\dot{\mathbf{P}}^{\beta}(t) = -\nabla_{\mathbf{X}} H^{\beta}(t, \mathbf{X}, \hat{\mathbf{\Phi}}(t, \mathbf{X}, \mathbf{X}_0), \hat{\mathbf{\Psi}}(t), \mathbf{X}, \mathbf{X}_0), \mathbf{P}^{\beta}(t))$$

$$\mathbf{P}^{\beta}(t_f) = \mathbf{0}$$

$$\phi^{\alpha}_{o[sd]}(t) + \sum_{\pi[sd] \in \mathbf{\Pi}^{\alpha}_{[sd]}} \phi^{\alpha}_{\pi[sd]}(t) = 1 \quad \forall\ [sd] \in \mathbf{SD}^{\alpha}$$

$$\sum_{[.d] \in \mathbf{D}^{\alpha}_{[s.]}} \psi^{\alpha}_{[sd]}(t) = 1 \qquad \forall\ [s.] \in \mathbf{S}^{\alpha}$$

$$\phi^{\alpha}_{o[sd]}(t),\ \phi^{\alpha}_{\pi[sd]}(t) \geq 0 \qquad \forall\ \pi[sd] \in \mathbf{\Pi}^{\alpha}_{[sd]},\ [sd] \in \mathbf{SD}^{\alpha}$$

$$\psi^{\alpha}_{[sd]}(t) \geq 0 \qquad \forall\ [.d] \in \mathbf{D}^{\alpha}_{[s.]},\ [s.] \in \mathbf{S}^{\alpha}$$

$$\phi^{\beta}_{o[sd]}(t) + \sum_{\pi[sd] \in \mathbf{\Pi}^{\beta}_{[sd]}} \phi^{\beta}_{\pi[sd]}(t) = 1 \quad \forall\ [sd] \in \mathbf{SD}^{\beta}$$

$$\sum_{[.d] \in \mathbf{D}^{\beta}_{[s.]}} \psi^{\beta}_{[sd]}(t) = 1 \qquad \forall\ [s.] \in \mathbf{S}^{\beta}$$

$$\phi^{\beta}_{o[sd]}(t),\ \phi^{\beta}_{\pi[sd]}(t) \geq 0 \qquad \forall\ \pi[sd] \in \mathbf{\Pi}^{\beta}_{[sd]},\ [sd] \in \mathbf{SD}^{\beta}$$

$$\psi^{\beta}_{[sd]}(t) \geq 0 \qquad \forall\ [.d] \in \mathbf{D}^{\beta}_{[s.]},\ [s.] \in \mathbf{S}^{\beta}$$

Proof: The proof is similar to that for the open-loop Stackelberg equilibrium.□

One of the disadvantages of using Stackelberg strategies is that the principle of optimality does not hold for the leader. A modification of the Stackelberg strategy concept requires that the strategies for the remaining time-to-go after each stage should be optimal.

5.3.2 Nonlinear Complementarity Problem Formulation

In this section, we formulate the dynamic two hierarchical class load sharing, routing and congestion control problem as a Nonlinear Complementarity Problem (NCP).

Define the vector of classβ congestion control, routing and load sharing fractions as well as Lagrange multipliers:

$$\mathbf{Z}^{\beta}(t) = \left[\ldots \phi^{\beta}_{o[sd]}(t) \ldots \phi^{\beta}_{\pi[sd]} \ldots Q^{\beta}_{[sd]}(t) \ldots . \psi^{\beta}_{[sd]}(t) \ldots Q^{\beta}_{[s.]}(t) \ldots \right]^{T}$$

and the vector of class β derivative of its Lagrangian with respect to the congestion control, routing and load sharing fractions as well as Lagrange multipliers:

$$\begin{aligned} \nabla L^{\beta}(\mathbf{Z}(t)) \quad = \quad & \left[\ldots \left(\frac{\partial H^{\beta}}{\partial \phi^{\beta}_{o[sd]}} - Q^{\beta}_{o[sd]}(t) \right) \ldots \left(\frac{\partial H^{\beta}}{\partial \phi^{\beta}_{\pi[sd]}} - Q^{\beta}_{\pi[sd]}(t) \right) \ldots \right. \\ & \ldots \left(1 - \phi^{\beta}_{o[sd]}(t) - \sum_{\pi[sd] \in \mathbf{\Pi}^{\beta}_{[sd]}} \phi^{\beta}_{\pi[sd]}(t) \right) \ldots \\ & \left. \ldots \left(\frac{\partial H^{\beta}}{\partial \psi^{\beta}_{[sd]}} - Q^{\beta}_{[s.]}(t) \right) \ldots \left(1 - \sum_{[.d] \in \mathbf{D}^{\beta}_{[s.]}} \psi^{\beta}_{[sd]}(t) \right) \ldots \right] \end{aligned}$$

Theorem :

Consider the dynamic join load sharing, routing and congestion control problem in distributed systems with two hierarchical classes, with fixed initial time t_0 *and final time* t_f.

If for each class c, $H^c(t, \mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi}, \mathbf{P}(t))$ *is differentiable and convex in* $(\mathbf{X}, \mathbf{\Phi}^c, \mathbf{\Psi}^c) \in (\mathbf{R}, \mathbf{RC}^c, \mathbf{LS}^c) \quad \forall\, t \in [t_0, t_f]$, *for each fixed value of* $(\mathbf{\Phi}^k, \mathbf{\Psi}^k) \in (\mathbf{RC}^k, \mathbf{LS}^k)$,

then $(\mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) \in (\mathbf{RC}, \mathbf{LS})$ *is a Stackelberg equilibrium if and only if it solves the following problem* $\forall\, t \in [t_0, t_f]$:

$$minimize \qquad \int_{t_0}^{t_f} g^{\alpha}(t, \mathbf{X}(t), \mathbf{\Phi}^{\alpha}(t), \mathbf{\Psi}^{\alpha}(t), \mathbf{\Phi}^{\beta}(t), \mathbf{\Psi}^{\beta}(t))dt$$

$$with\ respect\ to \quad (\mathbf{\Phi}^{\alpha}(t), \mathbf{\Psi}^{\alpha}(t), \mathbf{\Phi}^{\beta}(t), \mathbf{\Psi}^{\beta}(t)))$$

$$such\ that \qquad \dot{\mathbf{X}}(t) = \mathbf{f}(t, \mathbf{X}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t))$$

$$\mathbf{X}(t_0) = \mathbf{X}_0$$

$$(\mathbf{\Phi}^{\alpha}(t), \mathbf{\Psi}^{\alpha}(t)) \in (\mathbf{RC}^{\alpha}, \mathbf{LS}^{\alpha})$$

$$(\mathbf{\Phi}^{\beta}(t), \mathbf{\Psi}^{\beta}(t)) \in (\mathbf{RC}^{\beta\beta}, \mathbf{LS}^{\beta})$$

$$\nabla L^{\beta}(\mathbf{Z}^{\beta*}(t)) * \mathbf{Z}^{\beta*}(t) = 0$$

$$\nabla L^{\beta}(\mathbf{Z}^{\beta*}(t)) \geq 0$$

$$\mathbf{Z}^{\beta*}(t) \geq 0$$

$$\dot{\mathbf{X}}(t) = \mathbf{f}(t, \mathbf{X}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t))$$

$$\mathbf{X}(t_0) = \mathbf{X}_0$$

$$\dot{\mathbf{P}}^{\beta}(t) = -\nabla_{\mathbf{X}} H^{\beta}(t, \mathbf{X}, \mathbf{\Phi}(t), \mathbf{\Psi}(t), \mathbf{P}^{\beta}(t))$$

$$\mathbf{P}^{\beta}(t_f) = \mathbf{0}$$

Proof: After some algebraic manipulations, we find that the NCP: $\nabla L(\mathbf{Z}(t)) * \mathbf{Z}(t) = 0$; $\nabla L(\mathbf{Z}(t)) \geq 0$; $\mathbf{Z}(t) \geq 0$ with $\mathbf{Z}(t)$ and $\nabla L(\mathbf{Z}(t))$ as defined above, is equivalent to the Pontryagin's maximum principle necessary conditions for the follower. □

5.3.3 Variational Inequality Formulation

In this section, we formulate the dynamic non-cooperative load sharing, routing and congestion control problem as a Variational Inequality Problem (VIP).

Define the vector of class β congestion control, routing and load sharing fractions:

$$(\mathbf{\Phi}^{\beta}(t), \mathbf{\Psi}^{\beta}(t)) = \left[... \ \phi^{\beta}_{o[sd]}(t) \ ... \ \phi^{\beta}_{\pi[sd]}(t) \ ... \ \psi^{\beta}_{[sd]}(t) \ ... \right]^{T}$$

as well the vector of class β derivatives of its Lagrangian with respect to the congestion control, routing and load sharing fractions:

$$\nabla H^{\beta}(t, \mathbf{X}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t), \mathbf{P}(t)) \ = \ \left[... \ \frac{\partial H^{\beta}}{\partial \phi^{\beta}_{o[sd]}} \ ... \ \sum_{\pi[sd] \in \mathbf{\Pi}^{\beta}_{[sd]}} \frac{\partial H^{\beta}}{\partial \phi^{\beta}_{\pi[sd]}} \ ... \ \frac{\partial H^{\beta}}{\partial \psi^{\beta}_{[sd]}} \ ... \right]$$

Theorem :

Consider the dynamic join load sharing, routing and congestion control problem in distributed systems with two hierarchical classes, with fixed initial time t_0 and final time t_f.

Let for each class c, $g^c(t, \mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi})$, $\mathbf{f}(t, \mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi})$, be continuously differentiable with respect to $(\mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi}) \in (\mathbf{R}^n, \mathbf{\Phi}, \mathbf{\Psi}) \ \ \forall \ t \in [t_0, t_f]$. If H^c is continuously differentiable and convex in $(\mathbf{X}, \mathbf{\Phi}^c, \mathbf{\Psi}^c) \in (\mathbf{R}^n, \mathbf{RC}^c, \mathbf{LS}^c), \ \ \forall \ t \in [t_0, t_f]$, for each fixed value of $(\mathbf{\Phi}^k(t), \mathbf{\Psi}^k(t)) \in (\mathbf{RC}^k, \mathbf{LS}^k)$,

then $(\mathbf{\Phi}^*(t), \mathbf{\Psi}^*(t)) \in (\mathbf{RC}, \mathbf{LS})$ is a Stackelberg equilibrium if and only if it solves the following problem $\forall \ t \in [t_0, t_f]$:

$$\textit{minimize} \qquad \int_{t_0}^{t_f} g^{\alpha}(t, \mathbf{X}(t), \mathbf{\Phi}^{\alpha}(t), \mathbf{\Psi}^{\alpha}(t), \mathbf{\Phi}^{\beta}(t), \mathbf{\Psi}^{\beta}(t))dt$$

$$\textit{with respect to} \quad (\mathbf{\Phi}^{\alpha}(t), \mathbf{\Psi}^{\alpha}(t), \mathbf{\Phi}^{\beta}(t), \mathbf{\Psi}^{\beta}(t)))$$

$$\textit{such that} \qquad \dot{\mathbf{X}}(t) = \mathbf{f}(t, \mathbf{X}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t))$$

$$\mathbf{X}(t_0) = \mathbf{X}_0$$

$$(\mathbf{\Phi}^{\alpha}(t), \mathbf{\Psi}^{\alpha}(t)) \in (\mathbf{RC}^{\alpha}, \mathbf{LS}^{\alpha})$$

$$(\mathbf{\Phi}^{\beta}(t), \mathbf{\Psi}^{\beta}(t)) \in (\mathbf{RC}^{\beta\beta}, \mathbf{LS}^{\beta})$$

$$\nabla H^{\beta}(t, \mathbf{X}^{*}(t), \mathbf{\Phi}^{*}(t), \mathbf{\Psi}^{*}(t), \mathbf{P}(t)) * ((\mathbf{\Phi}, \mathbf{\Psi}) - (\mathbf{\Phi}^{*}(t), \mathbf{\Psi}^{*}(t))) \geq 0$$
$$\forall\ (\mathbf{\Phi}, \mathbf{\Psi}) \in (\mathbf{RC}, \mathbf{LS})$$

$$\dot{\mathbf{X}}(t) = \mathbf{f}(t, \mathbf{X}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t))$$

$$\mathbf{X}(t_0) = \mathbf{X}_0$$

$$\dot{\mathbf{P}}^{\beta}(t) = -\nabla_{\mathbf{X}} H^{\beta}(t, \mathbf{X}, \mathbf{\Phi}(t), \mathbf{\Psi}(t), \mathbf{P}^{\beta}(t))$$

$$\mathbf{P}^{\beta}(t_f) = \mathbf{0}$$

Proof: If $(\mathbf{\Phi}^{\beta *}(t), \mathbf{\Psi}^{\beta *}(t))$ is a local minimum for the following minimization problem

$$
\begin{array}{ll}
\textit{minimize} & \displaystyle\int_{t_0}^{t_f} g^{\beta}(t, \mathbf{X}(t), \begin{array}{c}\boldsymbol{\Phi}^{\alpha *}(t), \boldsymbol{\Phi}^{\beta}(t) \\ \boldsymbol{\Psi}^{\alpha *}(t), \boldsymbol{\Psi}^{\beta}(t)\end{array}) dt \\
\textit{with respect to} & (\boldsymbol{\Phi}^{\beta}(t), \boldsymbol{\Psi}^{\beta}(t)) \\
\textit{such that} & \dot{\mathbf{X}}(t) = \mathbf{f}(t, \mathbf{X}(t), \boldsymbol{\Phi}(t), \boldsymbol{\Psi}(t)) \\
& \mathbf{X}(t_0) = \mathbf{X_0} \\
\\
& (\boldsymbol{\Phi}^{\beta}(t), \boldsymbol{\Psi}^{\beta}(t)) \in (\mathbf{RC}^{\beta}, \mathbf{LS}^{\beta})
\end{array}
$$

and g^{β} is a continuously differentiable convex function over the nonempty convex, closed and bounded set $(\mathbf{RC}^{\beta}, \mathbf{LS}^{\beta})$, then $\forall\, t \in [t_0, t_f]$:

$$
\sum_{[sd] \in \mathbf{SD}^{\beta}} \left\{ \frac{\partial H^{\beta *}}{\partial \phi^{\beta}_{o[sd]}} * (\phi^{\beta}_{o[sd]} - \phi^{\beta *}_{o[sd]}(t)) + \right.
$$

$$
+ \sum_{\pi[sd] \in \mathbf{\Pi}^{\beta}_{[sd]}} \frac{\partial H^{\beta *}}{\partial \phi^{\beta}_{\pi[sd]}} * (\phi^{\beta}_{\pi[sd]} - \phi^{\beta *}_{\pi[sd]}(t)) +
$$

$$
\left. + \frac{\partial H^{\beta *}}{\partial \psi^{\beta}_{[sd]}} * (\psi^{\beta}_{[sd]} - \psi^{\beta *}_{[sd]}(t)) \right\} \geq 0 \quad \forall\, (\boldsymbol{\Phi}^{\beta}, \boldsymbol{\Psi}^{\beta}) \in (\mathbf{RC}^{\beta}, \mathbf{LS}^{\beta})
$$

□

Another equivalent formulation is the following Theorem:

Theorem :

Consider the dynamic join load sharing, routing and congestion control problem in distributed systems with two hierarchical classes, with fixed initial time t_0 *and final time* t_f.

Let for each class c, $g^c(t, \mathbf{X}, \boldsymbol{\Phi}, \boldsymbol{\Psi})$, $\mathbf{f}(t, \mathbf{X}, \boldsymbol{\Phi}, \boldsymbol{\Psi})$, *be continuously differentiable with respect to* $(\mathbf{X}, \boldsymbol{\Phi}, \boldsymbol{\Psi}) \in (\mathbf{R}^n, \boldsymbol{\Phi}, \boldsymbol{\Psi})$ $\forall\, t \in [t_0, t_f]$. *If* H^c *is continuously differentiable and convex in* $(\mathbf{X}, \boldsymbol{\Phi}^c, \boldsymbol{\Psi}^c) \in (\mathbf{R}^n, \mathbf{RC}^c, \mathbf{LS}^c)$, $\forall\, t \in [t_0, t_f]$, *for each fixed value of* $(\boldsymbol{\Phi}^k(t), \boldsymbol{\Psi}^k(t)) \in (\mathbf{RC}^k, \mathbf{LS}^k)$,

then $(\boldsymbol{\Phi}^*(t), \boldsymbol{\Psi}^*(t)) \in (\mathbf{RC}, \mathbf{LS})$ *is a Stackelberg equilibrium if and only if it solves the following problem* $\forall\, t \in [t_0, t_f]$:

$$\textit{minimize} \qquad \int_{t_0}^{t_f} g^{\alpha}(t, \mathbf{X}(t), \mathbf{\Phi}^{\alpha}(t), \mathbf{\Psi}^{\alpha}(t), \mathbf{\Phi}^{\beta}(t), \mathbf{\Psi}^{\beta}(t))dt$$

$$\textit{with respect to} \quad (\mathbf{\Phi}^{\alpha}(t), \mathbf{\Psi}^{\alpha}(t), \mathbf{\Phi}^{\beta}(t), \mathbf{\Psi}^{\beta}(t)))$$

$$\textit{such that} \qquad \dot{\mathbf{X}}(t) = \mathbf{f}(t, \mathbf{X}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t))$$

$$\mathbf{X}(t_0) = \mathbf{X}_0$$

$$(\mathbf{\Phi}^{\alpha}(t), \mathbf{\Psi}^{\alpha}(t)) \in (\mathbf{RC}^{\alpha}, \mathbf{LS}^{\alpha})$$

$$(\mathbf{\Phi}^{\beta}(t), \mathbf{\Psi}^{\beta}(t)) \in (\mathbf{RC}^{\beta\beta}, \mathbf{LS}^{\beta})$$

$$\int_{t_0}^{t_f} g^{\beta}(t, \mathbf{X}(t), \begin{matrix} \mathbf{\Phi}^{\alpha}(t), \mathbf{\Phi}^{\beta}(t) \\ \mathbf{\Psi}^{\alpha}(t), \mathbf{\Psi}^{\beta}(t) \end{matrix})dt =$$

$$= \min_{(\hat{\mathbf{\Phi}}^{\beta}(t), \hat{\mathbf{\Psi}}^{\beta}(t)) \in (\mathbf{RC}^{\beta\beta}, \mathbf{LS}^{\beta})} \int_{t_0}^{t_f} g^{\beta}(t, \mathbf{X}(t), \begin{matrix} \mathbf{\Phi}^{\alpha}(t), \hat{\mathbf{\Phi}}^{\beta}(t) \\ \mathbf{\Psi}^{\alpha}(t), \hat{\mathbf{\Psi}}^{\beta}(t) \end{matrix})dt$$

$$\nabla L^{\beta}(\mathbf{Z}^{\beta}(t)^*) * (\mathbf{Z}^{\beta} - \mathbf{Z}^{\beta}(t)^*) \geq 0 \qquad \forall\ \mathbf{Z}^{\beta} > 0$$

$$\dot{\mathbf{X}}(t) = \mathbf{f}(t, \mathbf{X}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t))$$

$$\mathbf{X}(t_0) = \mathbf{X}_0$$

$$\dot{\mathbf{P}}^{\beta}(t) = -\nabla_{\mathbf{X}} H^{\beta}(t, \mathbf{X}, \mathbf{\Phi}(t), \mathbf{\Psi}(t), \mathbf{P}^{\beta}(t))$$

$$\mathbf{P}^{\beta}(t_f) = \mathbf{0}$$

Proof: The NCP: $f(x^*) * x^* = 0 \quad f(x^*) \geq 0 \quad x^* > 0$
and the VIP: find x^* such that $f(x^*) * (x - x^*) \geq 0 \quad \forall x > 0$
are equivalent.□

5.3.4 Maximum Principle for Separable Cost Functions

In this section, we derive the first order necessary and sufficient conditions for a Stackelberg equilibrium on the path flows, when the cost function at each resource depends only on the flow on this resource.

The partial derivatives of the cost function $g^{\beta}(t, \mathbf{X}, \mathbf{\Phi}, \mathbf{\Psi})$ with respect to the path fractions $\phi^{\beta}_{\pi[sd]}$ can be written with respect to the link flows λ^{β}_{ij} and node flows λ^{β}_{i}:

$$\frac{\partial g^{\beta}_{ij}(t, \mathbf{X}_{ij}, \mathbf{\Phi}, \mathbf{\Psi})}{\partial \phi^{\beta}_{\pi[sd]}} = \frac{\partial g^{\beta}_{ij}(t, \mathbf{X}_{ij}, \mathbf{\Lambda}_{ij})}{\partial \lambda^{\beta}_{ij}} * \frac{\partial \lambda^{\beta}_{ij}}{\partial \phi^{\beta}_{\pi[sd]}} =$$

$$= \frac{\partial g^{\beta}_{ij}(t, \mathbf{X}_{ij}, \mathbf{\Lambda}_{ij})}{\partial \lambda^{\beta}_{ij}} * (\gamma^{\beta}_{[sd]}(t) + \gamma^{\beta}_{[s.]}(t) * \psi^{\beta}_{[sd]}) * 1_{ij \in \pi[sd]}(t)$$

$$\frac{\partial g^{\beta}_{i}(t, \mathbf{X}_{i}, \mathbf{\Phi}, \mathbf{\Psi})}{\partial \phi^{\beta}_{\pi[sd]}} = \frac{\partial g^{\beta}_{i}(t, \mathbf{X}_{i}, \mathbf{\Lambda}_{i})}{\partial \lambda^{\beta}_{i}} * \frac{\partial \lambda^{\beta}_{i}}{\partial \phi^{\beta}_{\pi[sd]}} =$$

$$= \frac{\partial g^{\beta}_{i}(t, \mathbf{X}_{i}, \mathbf{\Lambda}_{i})}{\partial \lambda^{\beta}_{i}} * (\gamma^{\beta}_{[sd]}(t) + \gamma^{\beta}_{[s.]}(t) * \psi^{\beta}_{[sd]}) * 1_{i \in \pi[sd]}(t)$$

$$\frac{\partial g^{\beta}_{o[sd]}(t, \mathbf{X}_{o[sd]}, \mathbf{\Phi}, \mathbf{\Psi})}{\partial \phi^{\beta}_{o[sd]}} = \frac{\partial g^{\beta}_{o[sd]}(t, \mathbf{X}_{o[sd]}, \mathbf{\Lambda}_{o[sd]})}{\partial \lambda^{\beta}_{o[sd]}} * \frac{\partial \lambda^{\beta}_{o[sd]}}{\partial \phi^{\beta}_{o[sd]}} =$$

$$= \frac{\partial g^{\beta}_{o[sd]}(t, \mathbf{X}_{o[sd]}, \mathbf{\Lambda}_{o[sd]})}{\partial \lambda^{\beta}_{o[sd]}} * (\gamma^{\beta}_{[sd]}(t) + \gamma^{\beta}_{[s.]}(t) * \psi^{\beta}_{[sd]})$$

$$\frac{\partial g_{ij}^{\beta}(t,\mathbf{X}_{ij},\mathbf{\Phi},\mathbf{\Psi})}{\partial \psi_{[sd]}^{\beta}} = \frac{\partial g_{ij}^{\beta}(t,\mathbf{X}_{ij},\mathbf{\Lambda}_{ij})}{\partial \lambda_{ij}^{\beta}} * \frac{\partial \lambda_{ij}^{\beta}}{\partial \psi_{[sd]}^{\beta}} =$$

$$= \sum_{\pi[sd]\in\mathbf{\Pi}_{[sd]}^{\beta}} \frac{\partial g_{ij}^{\beta}(t,\mathbf{X}_{ij},\mathbf{\Lambda}_{ij})}{\partial \lambda_{ij}^{\beta}} * \gamma_{[s.]}^{\beta}(t) * \phi_{\pi[sd]}^{c*} * 1_{ij\in\pi[sd]}(t)$$

$$\frac{\partial g_{i}^{\beta}(t,\mathbf{X}_{i},\mathbf{\Phi},\mathbf{\Psi})}{\partial \psi_{[sd]}^{\beta}} = \frac{\partial g_{i}^{\beta}(t,\mathbf{X}_{i},\mathbf{\Lambda}_{i})}{\partial \lambda_{i}^{\beta}} * \frac{\partial \lambda_{i}^{\beta}}{\partial \psi_{[sd]}^{\beta}} =$$

$$= \sum_{\pi[sd]\in\mathbf{\Pi}_{[sd]}^{\beta}} \frac{\partial g_{i}^{\beta}(t,\mathbf{X}_{i},\mathbf{\Lambda}_{i})}{\partial \lambda_{i}^{\beta}} * \gamma_{[s.]}^{\beta}(t) * \phi_{\pi[sd]}^{c*} * 1_{i\in\pi[sd]}(t)$$

$$\frac{\partial g_{o[sd]}^{\beta}(t,\mathbf{X}_{o[sd]},\mathbf{\Phi},\mathbf{\Psi})}{\partial \psi_{[sd]}^{\beta}} = \frac{\partial g_{o[sd]}^{\beta}(t,\mathbf{X}_{o[sd]},\mathbf{\Lambda}_{o[sd]})}{\partial \lambda_{o[sd]}^{\beta}} * \frac{\partial \lambda_{o[sd]}^{\beta}}{\partial \psi_{[sd]}^{\beta}} =$$

$$= \frac{\partial g_{o[sd]}^{\beta}(t,\mathbf{X}_{o[sd]},\mathbf{\Lambda}_{o[sd]})}{\partial \lambda_{o[sd]}^{\beta}} * \gamma_{[s.]}^{\beta}(t) * \phi_{o[sd]}^{c*}$$

$$\frac{\partial g_{[.d]}^{\beta}(t,\mathbf{X}_{[.d]},\mathbf{\Phi},\mathbf{\Psi})}{\partial \psi_{[sd]}^{\beta}} = \frac{\partial g_{[.d]}^{\beta}(t,\mathbf{X}_{[.d]},\mathbf{\Lambda}_{[.d]})}{\partial \lambda_{[.d]}^{\beta}} * \frac{\partial \lambda_{[.d]}^{\beta}}{\partial \psi_{[sd]}^{\beta}} =$$

$$= \frac{\partial g_{[.d]}^{\beta}(t,\mathbf{X}_{[.d]},\mathbf{\Lambda}_{[.d]})}{\partial \lambda_{[.d]}^{\beta}} * \gamma_{[s.]}^{\beta}(t)$$

5.4 Application to Datagram Networks

In this section, we apply the methodologies developed in the previous sections to datagram networks. We develop dynamic queueing models for the average number of class c packets in the queue and in the system (queue plus service) for multiple classes and priority classes $M/G/1$ queues. We also introduce the idea of using linearized approximate dynamic queueing models, in order to have a linear-quadratic problem for which there is extensive literature. We also suggest using second order dynamic queueing models, when the traffic can not be described only by first order models. Furthermore, we introduce Wiener process models for modeling the stochastic system. Finally, we present some cost functions and state constraints that can be used in the optimal control problem.

5.4.1 Dynamic Queueing Models for Multiple Classes

The general structure of the dynamic model introduced in this section is that the number of packets in a resource increases by the number of arrivals to and decreases by the number of departures from that resource. The departure rate should be a nonnegative, nondecreasing, continuous and concave function $\mu C * \rho(N)$ of the number of packets in the resource, with $\mu C * \rho(N) < N$. A dynamic model for $M/M/1$ queues, that was originally proposed by Agnew [4] and Rider [398] and was later used in network optimization studies by Filipiak [158, 159, 153], Economides, Ioannou & Silvester [137], Tipper & Sundareshan [485], is the following:

$$\dot{N}(t) = \lambda(t) - \mu C(t) * \frac{N(t)}{1 + N(t)}$$

Filipiak has also proposed a dynamic model for $M/M/\infty$ queues:

$$\dot{N}(t) = \lambda(t) - \mu C(t) * N(t)$$

as well as for $M/D/1$ queues:

$$\dot{N}(t) = \lambda(t) - \mu C(t) * \left(1 + N(t) - \sqrt{1 + (N(t))^2}\right)$$

Next, we extend the above models for multiple class $M/G/1$ queues. The average number of class c packets in $M/G/1$ queues is given by

$$N^c = \rho^c + \rho^c * \frac{\rho * \overline{x^2} * \mu^2}{2(1-\rho)} \quad \forall\ c$$

where ρ^c is the utilization for class c and $\rho = \sum_c \rho^c$ is the overall utilization.

Solving the above system of equations for ρ^c, we have the utilization for class c as a function of the average number of packets for each class:

$$\rho^c = \frac{2N^c * \left(1 - \overline{x^2} * \mu^2 - \sum_k N^k + \sqrt{\left(1 + \sum_k N^k\right)^2 - 2\sum_k N^k * (2 - \overline{x^2} * \mu^2)}\right)}{(2 - \overline{x^2} * \mu^2) * \left(1 - \sum_k N^k + \sqrt{\left(1 + \sum_k N^k\right)^2 - 2\sum_k N^k * (2 - \overline{x^2} * \mu^2)}\right)}$$

Then we propose the following dynamic model for multiple class $M/G/1$ queues:

$$\dot{N}^c(t) = \lambda^c(t) - \mu C(t) * \frac{2N^c(t)}{2 - \overline{x^2} * \mu^2} * \frac{\left(1 - \overline{x^2} * \mu^2 - \sum_k N^k(t) + \sqrt{\left(1 + \sum_k N^k(t)\right)^2 - 2\sum_k N^k(t) * (2 - \overline{x^2} * \mu^2)}\right)}{\left(1 - \sum_k N^k(t) + \sqrt{\left(1 + \sum_k N^k(t)\right)^2 - 2\sum_k N^k(t) * (2 - \overline{x^2} * \mu^2)}\right)}$$

For exponential service, general service and Processor Sharing ($P.S.$) discipline and deterministic service times, the above model gives the following dynamic models:

$$\dot{N}^c(t) = \lambda^c(t) - \mu C(t) * \frac{N^c(t)}{1 + \sum_k N^k(t)} \qquad M/M/1$$

$$\dot{N}^c(t) = \lambda^c(t) - \mu C(t) * \frac{w^c * N^c(t)}{1 + \sum_k w^k * N^k(t)} \quad \textit{class discriminating P.S.}$$

$$\dot{N}^c(t) = \lambda^c(t) - \mu C(t) * \frac{2N^c(t) * \left(-\sum_k N^k(t) + \sqrt{1 + \left(\sum_k N^k(t) \right)^2} \right)}{1 - \sum_k N^k(t) + \sqrt{1 + \left(\sum_k N^k(t) \right)^2}} \quad M/D/1$$

Also, for multiple class $M/M/\infty$ queues we have the following dynamic model:

$$\dot{N}^c(t) = \lambda^c(t) - \mu^c C(t) * N^c(t) \quad M/M/\infty$$

5.4.2 Linearized Dynamic Queueing Models

Although the above dynamic queueing models describe accurately the dynamic behavior of the queue, they depend nonlinearly on the average number of packets in the system (except the $M/M/\infty$ model). Therefore the analytical solution of the dynamic optimization problem usually becomes intractable. Next, we propose the linearization of the above dynamic queueing models, that gives simpler models. For example, the linearized multiple class $M/M/1$ queueing model is the following:

$$\dot{N}^c(t) = \lambda^c(t) - \mu C * \frac{N^c(t)}{1 + \sum_k N^k(t)}$$

$$\approx \lambda^c(t) - \mu C * \frac{\overline{N^c}}{1 + \sum_k \overline{N^k}} - \mu C * \sum_k \frac{\partial}{\partial \overline{N^k}} \left(\frac{\overline{N^c}}{1 + \sum_k \overline{N^k}} \right) * (N^k(t) - \overline{N^k})$$

$$\approx \lambda^c(t) - \mu C * \frac{\overline{N^c}}{1 + \sum_k \overline{N^k}} - \mu C * \frac{1 + \sum_{k \neq c} \overline{N^k}}{\left(1 + \sum_k \overline{N^k}\right)^2} * (N^c(t) - \overline{N^c}) +$$
$$+ \mu C * \sum_k \frac{N^c}{\left(1 + \sum_n \overline{N^n}\right)^2} * (N^k(t) - \overline{N^k})$$

$$\approx \lambda^c(t) - \mu C * \frac{\frac{\lambda^c}{\mu C - \sum_k \lambda^k}}{1 + \frac{\sum \lambda^k}{\mu C - \sum_k \lambda^k}} -$$

$$- \mu C * \frac{1}{\sum \lambda^k} * \left(N^c(t) - \frac{\lambda^c}{\mu C - \sum_k \lambda^k} \right) +$$
$$1 + \frac{\sum_k \lambda^k}{\mu C - \sum_k \lambda^k}$$

$$+ \mu C * \frac{\lambda^c}{\mu C \sum_n \lambda^n} * \sum_k \frac{1}{\left(1 + \frac{\sum_n \lambda^n}{\mu C - \sum_n \lambda^n}\right)^2} * \left(N^k(t) - \frac{\lambda^k}{\mu C - \sum_n \lambda^n} \right)$$

Finally, we have the following linearized model for multi-class $M/M/1$ queues:

$$\dot{N}^c(t) \approx \lambda^c(t) - \lambda^c * \frac{\sum_k \lambda^k}{\mu C} - (\mu C - \sum_k \lambda^k) * N^c(t) + \frac{\lambda^c * \mu C - \sum_n \lambda^n}{\mu C} * \sum_k N^k(t)$$

The above model satisfies the steady-state flow conservation

$$\lambda^c - \lambda^c * \frac{\sum_k \lambda^k}{\mu C} = (\mu C - \sum_k \lambda^k) * \overline{N^c} - \frac{\lambda^c * \mu C - \sum_n \lambda^n}{\mu C} * \sum_k \overline{N^k} \Leftrightarrow \overline{N^c} = \frac{\lambda^c}{\mu C - \sum_k \lambda^k}$$

Similarly, we may derive dynamic models for the average number of customers in the systems (queue plus service), or in the queue, for multiple class, priority class $M/G/1$ queues.

Another approximate model for multiple class $M/M/1$ queues is the following:

$$\begin{aligned}
\dot{N}^c(t) &= \lambda^c(t) - \mu C * \frac{N^c(t)}{1 + \sum_k N^k(t)} \\
&\approx \lambda^c(t) - \mu C * \frac{1}{1 + \sum_k \overline{N^k}} * N^c(t) \\
&\approx \lambda^c(t) - \mu C * \frac{1}{1 + \frac{\sum_k \lambda^k}{\mu C(t) - \sum_k \lambda^k}} * N^c(t) \\
&\approx \lambda^c(t) - (\mu C - \sum_k \lambda^k) * N^c(t)
\end{aligned}$$

The above model satisfies the steady-state flow conservation

$$\lambda^c = (\mu C - \sum_k \lambda^k) * N^c \Leftrightarrow N^c = \frac{\lambda}{\mu C - \sum_k \lambda^k}$$

The link length becomes

$$l = \left[\frac{\partial}{\partial N^c}\left((\mu C - \sum_k \lambda^k) * N^c\right)\right]^{-1} = \frac{1}{\mu C - \sum_k \lambda^k} = \frac{\frac{1}{\mu C}}{\frac{\sum \lambda^k}{k}} =$$

$$= \frac{\frac{1}{\mu C}}{\frac{\sum N^k}{k}} = \frac{1 + \sum_k N^k}{\mu C}$$

$$1 - \frac{k}{1 + \sum_k N^k}$$

This result explains why the shortest route routing achieves good performance (see section 5.6.5).

After the model linearization, the system state is described by the following state equation

$$\dot{\mathbf{X}} = \mathbf{A} * \mathbf{X} + \mathbf{B} * \mathbf{U} \qquad \mathbf{X}_0 : given$$

with cost function

$$\int_{t_0}^{t_f} \frac{1}{2} * (\mathbf{X}^T * \mathbf{Q} * \mathbf{X} + \mathbf{U}^T * \mathbf{R} * \mathbf{U}) dt$$

where A, B, Q, R are suitable matrices. Thus, we can use results from the optimal control theory on linear-quadratic problems, to solve the joint load sharing, routing and congestion control problem.

5.4.3 Dynamic Queueing Models for the Packets in Queue

In future high speed networks, we will have information only about the average number of packets in the queue (not both in the queue and in service), due to the enormous number of packets that will be in transit into the network. Therefore, it is also useful to have dynamic queueing models with state the average number of packets in the queue.

Here, we introduce a dynamic queueing model for the average number of packet in the multiple class $M/G/1$ queue. The average number of class c packets in queue for a multiple class $M/G/1$ queue is given by

$$N_Q^c = \rho^c * \frac{\rho * \overline{x^2} * \mu^2}{2(1-\rho)} \cdot \forall\, c$$

Solving the above system of equations, we have the utilization for class c, ρ^c, as a function of the average number of packets in queue for all classes

$$\rho^c = \frac{2N_Q^c * \left(\overline{x^2} * \mu^2 + \sum_k N_Q^k - \sqrt{\left(\sum_k N_Q^k \right)^2 + 2 \sum_k N^Q * \overline{x^2} * \mu^2} \right)}{\overline{x^2} * \mu^2 * \left(- \sum_k N_Q^k + \sqrt{\left(\sum_k N_Q^k \right)^2 + 2 \sum_k N^Q * \overline{x^2} * \mu^2} \right)}$$

Then we propose the following dynamic model for multiple class $M/G/1$ queues:

$$\dot{N}_Q^c(t) = \lambda^c(t) - \mu C(t) * \frac{2N_Q^c(t)}{\overline{x^2} * \mu^2} *$$

$$* \frac{\overline{x^2} * \mu^2 + \sum_k N_Q^k - \sqrt{\left(\sum_k N_Q^k \right)^2 + 2 \sum_k N^Q * \overline{x^2} * \mu^2}}{- \sum_k N_Q^k + \sqrt{\left(\sum_k N_Q^k \right)^2 + 2 \sum_k N^Q * \overline{x^2} * \mu^2}}$$

For exponential service, general service with Processor Sharing and deterministic service time, the above model gives the following dynamic models:

$$\dot{N}_Q^c(t) = \lambda^c(t) - \mu C(t) * N_Q^c(t) *$$

$$* \frac{2 + \sum_k N_Q^k - \sqrt{\left(\sum_k N_Q^k\right)^2 + 4\sum_k N^Q}}{-\sum_k N_Q^k + \sqrt{\left(\sum_k N_Q^k\right)^2 + 4\sum_k N^Q}} \quad M/M/1 \text{ or } P.S.$$

$$\dot{N}_Q^c(t) = \lambda^c(t) - \mu C(t) * 2N_Q^c(t) *$$

$$* \frac{1 + \sum_k N_Q^k - \sqrt{\left(\sum_k N_Q^k\right)^2 + 2\sum_k N^Q}}{-\sum_k N_Q^k + \sqrt{\left(\sum_k N_Q^k\right)^2 + 2\sum_k N^Q}} \quad M/D/1$$

Note that for single class, we have:

$$\dot{N}_Q(t) = \lambda(t) - \mu C(t) * \frac{-N_Q(t) + \sqrt{(N_Q)^2 + 2N_Q * \overline{x^2} * \mu^2}}{\overline{x^2} * \mu^2} \quad M/G/1$$

$$\dot{N}_Q(t) = \lambda(t) - \mu C(t) * \frac{-N_Q(t) + \sqrt{(N_Q)^2 + 4N_Q}}{2} \quad M/M/1 \text{ or } P.S.$$

$$\dot{N}_Q(t) = \lambda(t) - \mu C(t) * \left(-N_Q(t) + \sqrt{(N_Q)^2 + 2N_Q}\right) \quad M/D/1$$

5.4.4 Dynamic Queueing Models for Priority Classes

In this section, we derive dynamic models for queues with priority classes. For Poisson arrival and exponential service times. The average number of packets in the system of the high priority class α is given by

$$N^{\alpha} = \frac{\rho^{\alpha}}{1 - \rho^{\alpha}}$$

and the average number of packets in the system of the low priority class β is given by

$$N^{\beta} = \frac{\rho^{\beta} * (1 - \rho^{\alpha}) + \rho^{\alpha}\rho^{\beta} * \mu^{\beta}/\mu^{\alpha}}{(1 - \rho^{\alpha}) * (1 - \rho^{\alpha} - \rho^{\beta})}$$

Solving the above system, we have the utilization of class α as a function of the average number of class α packets in the system and the utilization of class β as a function of the average number of class α and β packets in the system

$$\rho^{\alpha} = \frac{N^{\alpha}}{1 + N^{\alpha}}$$

$$\rho^{\beta} = \frac{N^{\beta}}{(1 + N^{\alpha}) * (1 + N^{\alpha} * \frac{\mu^{\beta}}{\mu^{\alpha}} + N^{\beta})}$$

Then, we have the following dynamic model for the high preemptive priority class α:

$$\rho^{\alpha} = \frac{N^{\alpha}(t)}{1 + N^{\alpha}(t)}$$

and for the low preemptive priority class β:

$$\rho^{\beta} = \frac{N^{\beta}}{(1 + N^{\alpha}(t)) * (1 + N^{\alpha}(t) * \frac{\mu^{\beta}}{\mu^{\alpha}} + N^{\beta})}$$

Next, we give a dynamic model for the average number of packets in the queue. The average number of packets in the queue of the high priority class α is given by

$$N_Q^\alpha = \frac{(\rho^\alpha)^2}{1-\rho^\alpha}$$

and the average number of packets in the queue of the low priority class β is given by

$$N_Q^\beta = \frac{\rho^\beta * (\rho^\alpha + \rho^\beta) * (1-\rho^\alpha) + \rho^\beta * \rho^\alpha * \mu^\beta/\mu^\alpha}{(1-\rho^\alpha) * (1-\rho^\alpha-\rho^\beta)}$$

Solving the above system, we have the utilization of class α as a function of the average number of class α packets in queue and the utilization of class β as a function of the average number of class α and β packets in queue

$$\rho^\alpha = \frac{-N_Q^\alpha + \sqrt{(N_Q^\alpha)^2 + 4N_Q^\alpha}}{2}$$

$$\rho^\beta = \left[-\frac{\rho^\alpha * \mu^\beta/\mu^\alpha + (1-\rho^\alpha) * (\rho^\alpha + N_Q^\beta)}{2(1-\rho^\alpha)} + \right.$$

$$\left. + \frac{\sqrt{[\rho^\alpha * \mu^\beta/\mu^\alpha + (1-\rho^\alpha) * (\rho^\alpha + N_Q^\beta)]^2 + 4N_Q^\beta(1-\rho^\alpha)^3}}{2(1-\rho^\alpha)} \right]$$

Then the dynamic model of the average number of packets in queue for the high preemptive priority class α is

$$\dot{N}_Q^\alpha(t) = \lambda^\alpha(t) - \mu^\alpha C(t) * \frac{-N_Q^\alpha(t) + \sqrt{(N_Q^\alpha(t))^2 + 4N_Q^\alpha(t)}}{2}$$

and for the low preemptive priority class β

$$\dot{N}_Q^\beta(t) = \lambda^\beta(t) - \mu^\beta C(t) * \left[-\frac{\rho^\alpha * \mu^\beta/\mu^\alpha + (1-\rho^\alpha) * (\rho^\alpha + N_Q^\beta(t))}{2(1-\rho^\alpha)} + \right.$$

$$\left. + \frac{\sqrt{[\rho^\alpha * \mu^\beta/\mu^\alpha + (1-\rho^\alpha) * (\rho^\alpha + N_Q^\beta(t))]^2 + 4N_Q^\beta(t)(1-\rho^\alpha)^3}}{2(1-\rho^\alpha)} \right]$$

Similarly, for Poisson arrival and exponential service times, we have the following dynamic model of the average number of packets for the high non-preemptive priority class α:

$$\dot{N}_Q^\alpha(t) = \lambda^\alpha(t) - \mu C'(t) *$$

$$* \frac{N_Q^\alpha * (N_Q^\alpha + N_Q^\beta) + \sqrt{[N_Q^\alpha * (N_Q^\alpha + N_Q^\beta)]^2 + 4(N_Q^\alpha)^2 * (N_Q^\alpha + N_Q^\beta + N_Q^\alpha N_Q^\beta)}}{2(N_Q^\alpha + N_Q^\beta + N_Q^\alpha N_Q^\beta)}$$

and for the low non-preemptive priority class β

$$\dot{N}_Q^\beta(t) = \lambda^\beta(t) - \mu C(t) * \left(\frac{N_Q^\alpha * (1 - \rho^\alpha)}{\rho^\alpha} - \rho^\alpha \right)$$

5.4.5 Second Order Dynamic Queueing Models

In this section, we suggest using a second order model in optimization of systems with bursty traffic, where the variance of the number of packets can be large. Next, we suggest using the second order model by Rothkopf & Oren [407], and Clark [109]:

$$\dot{N}(t) = \lambda(t) - \mu C(t) * (1 - \pi_0(t))$$

$$Var(N(t)) = \lambda(t) + \mu C(t) - \mu C(t) * \pi_0(t) * (2N(t) + 1)$$

where

$$\pi_0(t) = \left(\frac{N(t)}{Var(N(t))} \right) \frac{(N(t))^2}{Var(N(t)) - N(t)}$$

5.4.6 Wiener Process Models

In this section, we introduce a Wiener process model for flow that fluctuates with a large variance around its average value. We introduce a stochastic term for the arrival and departure rate in the dynamic models presented in the previous section. For example, the dynamic model for $M/M/1$ queues becomes:

$$\dot{N}^c(t) = \left(\lambda^c(t) - a^c(t) * \frac{dw_a^c}{dt} \right) - \left(\mu C(t) * \frac{N^c(t)}{1 + \sum_k N^k(t)} - b^c(t) * \frac{dw_b^c}{dt} \right)$$

where $a^c(t)$ and $b^c(t)$ are the standard deviations of the arrival and departure rates for class c, and $w_a^c(t)$ and $w_b^c(t)$ are Wiener processes.

We can rewrite the above model as

$$\dot{N}^c(t) = \left(\lambda^c(t) - a^c(t) * \overline{\xi^c}_a(t)\right) - \left(\mu C(t) * \frac{N^c(t)}{1 + \sum_k N^k(t)} - b^c(t) * \overline{\xi^c}_b(t)\right)$$

where $\overline{\xi^c}_a(t) = \frac{dw_a^c(t)}{dt}$ and $\overline{\xi^c}_b(t) = \frac{dw_b^c(t)}{dt}$ are zero mean, unit variance normal random variables.

5.4.7 Cost Functions

In this section, we introduce cost functions that can be used in the dynamic problem. Desired properties of a cost function are to be: i) nonnegative, ii) nondecreasing, iii) continuous and iv) convex.

We may consider as cost function the total time packets spent on each network resource ij

$$g_{ij[sd]}^c = \int_{t_0}^{t_f} N_{ij[sd]}^c(t)dt$$

blocking at resource ij

$$g_{ij[sd]}^c = \int_{t_0}^{t_f} B_{ij[sd]}^c(t)dt$$

blocking at path $\pi[sd]$

$$g_{\pi[sd]}^c = \int_{t_0}^{t_f} \prod_{ij \in \pi[sd]} B_{ij[sd]}^c(t)dt$$

rejected flow at resource ij

$$g^c_{ij[sd]} = \int_{t_0}^{t_f} \phi^c_{o,ij[sd]}(t)dt$$

rejected flow at source $[s.]$ for destination $[.d]$

$$g^c_{0,[sd]} = \int_{t_0}^{t_f} \phi^c_{o,,[sd]}(t)dt$$

5.4.8 State Constraints

In this section, we define flow control constraints on the number of packets $N_{ij[sd]}(t) \geq 0$ that can coexist at the network resources. For clear exposition, we consider only one class in the network. The case of multiple classes follows trivially.

The total expected number of packets on every link ij should be less than the buffer (or window) size of link ij, $\sum_{[sd]} N_{ij[sd]}(t) \leq W_{ij}(t)$.

Also, in order to guarantee an upper bound on the delay that packets may suffer from source to destination, the total expected number of packets on every path $\pi[sd]$ should be less than the end-to-end window size on path $\pi[sd]$, $\sum_{[s_1d_1]} \sum_{ij} N_{ij[s_1d_1]}(t) * 1_{ij\in\pi[sd]}(t) \leq W_{\pi[sd]}$, where $1_{ij\in\pi[sd]}(t)$ is the indicator function that link ij is on the path $\pi[sd]$.

Although controlling the total number of packets in the network is optimum from the system point of view, it may also be unfair to some users. Some aggressive users (a source, a destination or a virtual circuit) may congest the network. If the flow and congestion control operate without paying attention to the identity of packets, other users may be unfairly penalized. So, we point out three identities that should also be controlled for fairness reasons:

1) *each source*:

In order that source $[s.]$ does not monopolize the network resources, the total expected number of packets originated from node $[s.]$ should be less than the network "capacity" for packets from source $[s.]$, $\sum_{[.d]} \sum_{ij} N_{ij[sd]}(t) \leq W_{[s.]}$. Also, in order that source $[s.]$ does not monopolize path $\pi[sd]$, the total expected number

of packets originated at node $[s.]$ on path $\pi[sd]$ should be less than the path's "capacity" for packets originated at $[s.]$, $\sum_{[.d_1]} \sum_{ij} N_{ij[sd_1]}(t) * 1_{ij \in \pi[sd]}(t) \leq W_{[s.],\pi[sd]}$. Finally, in order that source $[s.]$ does not monopolize link ij, the total expected number of packets originated at node $[s.]$ on link ij should be less than the link's "capacity" for packets originated at $[s.]$, $\sum_{[.d]} N_{ij[sd]}(t) \leq W_{[s.],ij}$.

2) *each destination*:

Similarly, in order that destination $[.d]$ does not monopolize the network resources, the total expected number of packets destined to node $[.d]$ should be less than the network "capacity" for packets to destination $[.d]$, $\sum_{[s.]} \sum_{ij} N_{ij[sd]}(t) \leq W_{[.d]}$. Also, in order that destination $[.d]$ does not monopolize path $\pi[sd]$, the total expected number of packets destined to node $[.d]$ on path $\pi[sd]$ should be less than the path's "capacity" for packets destined to $[.d]$, $\sum_{[s_1.]} \sum_{ij} N_{ij[s_1 d]}(t) * 1_{ij \in \pi[sd]}(t) \leq W_{[.d],\pi[sd]}$. Finally, in order that destination $[.d]$ does not monopolize link ij, the total expected number of packets destined to node $[.d]$ on link ij should be less than the link's "capacity" for packets destined to $[.d]$, $\sum_{[s.]} N_{ij[sd]}(t) \leq W_{[.d],ij}$.

3) *each class*:

In order that $[sd]$ packets do not monopolize the network resources, the total expected number of $[sd]$ packets should be less than the network capacity for $[sd]$ packets, $\sum_{ij} N_{ij[sd]}(t) \leq W_{[sd]}$. Also, in order that $[sd]$ packets do not monopolize path $\pi[sd]$, the total expected number of $[sd]$ packets on path $\pi[sd]$ should be less than the end-to-end window size for $[sd]$ packets on path $\pi[sd]$, $\sum_{ij} N_{ij[sd]}(t) * 1_{ij \in \pi[sd]}(t) \leq W_{\pi[sd],[sd]}$. Finally, in order that $[sd]$ packets do not monopolize link ij, the expected number of $[sd]$ packets on link ij, should be less than the buffer (or window) size for $[sd]$ packets on link ij, $N_{ij[sd]}(t) \leq W_{ij[sd]}(t)$, with $\sum_{[sd]} W_{ij[sd]}(t) \geq W_{ij}(t)$.

5.5 Application to Virtual Circuit Networks

In this section, we apply the methodologies developed in the previous sections to virtual circuit networks. We develop dynamic queueing models for the average number of packets coupled with dynamic queueing models for the average number of virtual circuits. Furthermore, we introduce Wiener process models for modeling the stochastic system. We also present some cost functions and state constraints that can be used in the optimal control problem. Finally, we propose a class of heuristic link lengths that can be used on-line.

5.5.1 Dynamic Queueing Models for Multiple Classes

We consider two levels of the flow in virtual circuit networks. At the virtual level, we model each network resource as an $M/M/\infty$ queue. That means that an infinite number of virtual circuits may coexist at every network resource (see section 5.6.2). Then the average number of class c virtual circuits is described by the following dynamic model:

$$\dot{V}^c(t) = \gamma^c(t) - \delta^c(t) * V^c(t)$$

At the packet level, we model each network resource using any model among those proposed in section 5.4 for datagram networks. For example, the average number of class c packets for $M/M/1$ or $P.S.$ queues is described by the following dynamic model:

$$\dot{N}^c(t) = r^c(t) * V^c(t) - \mu C(t) * \frac{N^c(t)}{1 + \sum_k N^k}$$

where $r^c(t)$ is the average packet arrival rate per virtual circuit of class c.

Next, we give several dynamic models for virtual circuit networks for different optimization formulations. For easy of exposition, we consider dynamic models only for links.

i) The routing decisions are done at each source node $[s.]$ on the path flow space. Therefore, the total arrival rate to link ij is the sum of all path flows that pass through this link:

$$\dot{V}_{ij[sd]}(t) = \sum_{\pi[sd]} \gamma_{[sd]}(t) * \phi_{\pi[sd]}(t) * 1_{ij \in \pi[sd]}(t) - \delta_{[sd]}(t) * V_{ij[sd]}(t)$$

ii) The routing decisions are done at each network node on the link flow space and the virtual circuit duration is very small. Therefore the virtual circuit departure rate from a link becomes arrival rate to the next link. The arrival rate to the outgoing links sj from the source node $[s.]$ is the fraction of flow that is routed through that link. The arrival rate to an intermediate link ij is the departure rate from the ingoing links to node i weighted by the fraction $\phi_{ij[sd]}$ that is assigned to outgoing link ij from node i:

$$\dot{V}_{sj[sd]}(t) = \gamma_{[sd]}(t) * \phi_{sj[sd]}(t) - \delta_{[sd]}(t) * V_{sj[sd]}(t)$$
$$\dot{V}_{ij[sd]}(t) = \sum_{k \in I(i)} \delta_{[sd]}(t) * V_{ki[sd]}(t) * \phi_{ij[sd]}(t) - \delta_{[sd]}(t) * V_{ij[sd]}(t) \quad s \neq i$$

iii) The routing decisions are done at each network node on the path flow space and the virtual circuit duration is very small. Therefore the virtual circuit departure rate from a link becomes arrival rate to the next link. The arrival rate to the outgoing links sj from the source node $[s.]$ is the sum of all fractions of path flows that are routed through that link. The arrival rate to an intermediate link ij is the departure rate from the ingoing links to node i that is assigned to outgoing link ij from node i:

$$\dot{V}_{sj[sd]}(t) = \sum_{\pi[sd]} \gamma_{[sd]}(t) * \phi_{\pi[sd]}(t) * 1_{sj \in \pi[sd]}(t) - \delta_{[sd]}(t) * V_{sj[sd]}(t)$$
$$\dot{V}_{ij[sd]}(t) = \sum_{k \in I(i)} \sum_{\pi[sd]} \delta_{[sd]}(t) * V_{ki[sd]}(t) * 1_{ki \in \pi[sd]}(t) * 1_{ij \in \pi[sd]}(t)$$
$$-\delta_{[sd]}(t) * V_{ij[sd]}(t) \quad s \neq i$$

iv) A dynamic model for the paths (virtual and not physical links): The average number of virtual circuits for $[sd]$ on path $\pi[sd]$ is:

$$\dot{V}_{\pi[sd]}(t) = \gamma_{[sd]}(t) * \phi_{\pi[sd]}(t) - \delta_{[sd]}(t) * V_{\pi[sd]}(t)$$

v) The routing decisions are done at each network node on the path flow space and the virtual circuit duration is very long. Therefore a virtual circuit stays at

each network link for long time, so we can assume that its arrival rate does not change drastically over time. Then the arrival rate at each link is the sum of path flows that pass through this link:

$$\dot{V}_{ij}(t) = \sum_{[sd]} \sum_{\pi[sd]} \gamma_{[sd]}(t) * \phi_{\pi[sd]}(t) * 1_{ij \in \pi[sd]}(t) - \delta(t) * V_{ij}(t)$$

vi) The routing decisions are done at each network node on the link flow space. However, if a virtual circuit is rejected with probability $\phi_{no[sd]}$ (for congestion control reasons) at node n, then this virtual circuit is reestablished from the source. Therefore, the successful arrival rate at the source $[s.]$ is $\gamma_{[sd]}(t) * \prod_n (1 - \phi_{no[sd]}(t))$ Virtual circuits may also be rejected at each network link for congestion control reasons:

$$\dot{V}_{sj[sd]}(t) = \gamma_{[sd]}(t) * \prod_n (1 - \phi_{no[sd]}(t)) * \phi_{sj[sd]}(t) - \delta_{[sd]}(t) * V_{sj[sd]}(t)$$

$$\dot{V}_{ij[sd]}(t) = \sum_{k \in I(i)} \delta_{[sd]}(t) * V_{ki[sd]}(t) * \phi_{ij[sd]}(t) - \delta_{[sd]}(t) * V_{ij[sd]}(t) \quad s \neq i$$

Next, we give several dynamic models for the packet level for different optimization formulations:

i) The routing decisions are done at each network node on the link flow space and the virtual circuit duration is very long. Then a virtual circuit stays for long time at each network link:

$$\dot{N}_{ij[sd]}(t) = r_{[sd]}(t) * V_{ij[sd]}(t) - \mu C_{ij}(t) * \rho_{ij[sd]}(\mathbf{N}_{ij}(t))$$

ii) The routing decisions are done at each network node on the link flow space. The packet departure rate from a node is routed to an outgoing link from that node:

$$\dot{N}_{sj[sd]}(t) = \quad r_{[sd]}(t) * V_{sj[sd]}(t) - \mu C_{sj}(t) * \rho_{sj[sd]}(\mathbf{N}_{sj}(t))$$

$$\dot{N}_{ij[sd]}(t) = \quad \sum_{k \in I(i)} \mu C_{ki}(t) * \rho_{ki[sd]}(\mathbf{N}_{ki}(t)) * \phi_{ij[sd]}(t)$$

$$-\mu C_{ij}(t) * \rho_{ij[sd]}(\mathbf{N}_{ij}(t)) \quad s \neq i$$

iii) The routing decisions are done at each network node on the link flow space. The packet departure rate from a node is routed to an outgoing link from that node. However, packets may also fail transmission on link ki, that has error rate $e_{ki}(t)$ and be retransmitted from the source node $[s.]$ after a time-out period τ. So, the source node $[s.$ receives an extra $[sd]$ flow, $\sum_{ki} \mu C_{ki}(t-\tau) * e_{ki}(t-\tau) * \rho_{ki[sd]}(\mathbf{N}_{ki}(t-\tau))$, due to packet failures at links ki inside the network. Then the source node $[s.]$ routes a fraction $\phi_{sj[sd]}(t)$ of this flow to its outgoing link sj. Any other node $i \neq s$, receives the successful packet flow, from its input neighbors and routes a fraction $\phi_{ij[sd]}(t)$ of this flow to its outgoing link ij.

$$\begin{aligned}
\dot{N}_{sj[sd]}(t) = \quad & r_{[sd]}(t) * V_{sj[sd]}(t) + \\
& + \sum_{ki} \mu C_{ki}(t-\tau) * e_{ki}(t-\tau) * \rho_{ki[sd]}(\mathbf{N}_{ki}(t-\tau)) * \phi_{sj[sd]} \\
& - \mu C_{sj}(t) * \rho_{sj[sd]}(\mathbf{N}_{sj}(t)) \\
\dot{N}_{ij[sd]}(t) = \quad & \sum_{k \in I(i)} \mu C_{ki}(t) * (1 - e_{ki}(t)) * \rho_{ki[sd]}(\mathbf{N}_{ki}(t)) * \phi_{ij[sd]}(t) \\
& - \mu C_{ij}(t) * \rho_{ij[sd]}(\mathbf{N}_{ij}(t)) \quad s \neq i
\end{aligned}$$

iv) The routing decisions are done at each network node on the link flow space and the virtual circuit duration is long. The packet departure rate from a node is routed to an outgoing link from that node. However, packets may also fail at a link ki, that has error rate $e_{ki}(t)$ and be retransmitted from the source after time τ:

$$\dot{N}_{sj[sd]}(t) = r_{[sd]}(t) * V_{sj[sd]}(t) + \sum_{ki} r_{[sd]}(t) * V_{ki[sd]}(t) * e_{ki(t)}$$

$$-\mu C_{sj}(t) * \rho_{sj[sd]}(\mathbf{N}_{sj}(t))$$

$$\dot{N}_{ij[sd]}(t) = \sum_{k \in I(i)} r_{[sd]}(t) * V_{ki[sd]}(t) * (1 - e_{ki}(t)) * \phi_{ij[sd]}(t)$$

$$-\mu C_{ij}(t) * \rho_{ij[sd]}(\mathbf{N}_{ij}(t)) \quad s \neq i$$

v) The routing decisions are done at each network node on the path flow space and the virtual circuit duration is long.

$$\dot{N}_{ij[sd]}(t) = \sum_{\pi[sd]} r_{[sd]}(t) * V_{\pi[sd]}(t) * 1_{ij \in \pi[sd]} - \mu C_{ij}(t) * \rho_{ij[sd]}(\mathbf{N}_{ij}(t))$$

vi) Finally, if we do not consider classes

$$\dot{N}_{ij}(t) = r(t) * V_{ij}(t) - \mu C_{ij}(t) * \rho_{ij}(\mathbf{N}_{ij}(t))$$

5.5.2 Cost Functions

In this section, we introduce cost functions for the dynamic problem. We consider as cost function for class c and source-destination $[sd]$, at time t, at network resource ij, the total time packets spent at ij

$$g^c_{ij[sd]}(t, \mathbf{X}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t)) = C^c_{N,i[sd]} * N^c_{i[sd]}(t)$$

the total time virtual circuits spent at ij,

$$g^c_{ij[sd]}(t, \mathbf{X}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t)) = C^c_{V,i[sd]} * V^c_{i[sd]}(t)$$

the rejected flow cost at ij

$$g^c_{ij[sd]}(t, \mathbf{X}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t)) = C^c_{o[sd]} * \lambda^c_{o[sd]}(t)$$

the negative throughput at ij

$$g^c_{ij[sd]}(t, \mathbf{X}(t), \mathbf{\Phi}(t), \mathbf{\Psi}(t)) = C^c_{\mu,i[sd]} * \mu C_{ij} * \rho^c_{ij[sd]}(\mathbf{N}_{ij}(t))$$

5.5.3 Length of a Link

In previous sections, we found that jobs are sent to destinations of minimum length and are routed through minimum length paths. At that time we defined the lengths to destinations, the path lengths and the rejection lengths. In this section, we introduce some lengths that are very simple, however they are based on heuristic arguments.

For dynamic (or adaptive) load sharing, routing and congestion control, we need to know the state of each system resource. We define as length of a system resource the load on this resource. So, if a resource is lightly loaded, then its length is small, while if the resource is heavily loaded, then its length is large. Then the dynamic algorithm chooses the resource with the minimum length. Depending on the information we select about the state of each system resource, we may define different lengths of the resource (link, node, computer site, etc.).

Next, we define the length of a resource at time t as a convex combination of its current length at t and its expected length in the future:

$$l_{ij}(t) = \epsilon_{ij}(t) * l_{ij}^{current}(t) + (1 - \epsilon_{ij}(t)) * l_{ij}^{future}(t) \quad 0 \leq \epsilon \leq 1$$

Based on models presented earlier, we define some simple approximations for these lengths:

$$
\begin{aligned}
l_{ij}^{current}(t) &= \frac{1 + N_{ij}(t)}{\mu C_{ij}(t)} \\
&= T_{ij}(t) \\
&= \begin{cases} \dfrac{1}{\mu C_{ij}(t) - r(t) * V_{ij}(t)} & \text{if } \mu C_{ij}(t) > r(t) * V_{ij}(t) \\ \infty & \text{o.w.} \end{cases} \\
&= \begin{cases} \dfrac{1}{\mu C_{ij}(t) - \lambda_{ij}(t)} & \text{if } \mu C_{ij}(t) > \lambda_{ij}(t) \\ \infty & \text{o.w.} \end{cases} \\
&= \frac{[1 + N_{ij}(t)]^2}{\mu C_{ij}(t)} \\
&= \begin{cases} \dfrac{\mu C_{ij}(t)}{[\mu C_{ij}(t) - r(t) * V_{ij}(t)]^2} & \text{if } \mu C(t)_{ij} > r * V_{ij}(t) \\ \infty & \text{o.w.} \end{cases} \\
&= \begin{cases} \dfrac{\mu C_{ij}(t)}{[\mu C_{ij}(t) - \lambda_{ij}(t)]^2} & \text{if } \mu C_{ij}(t) > \lambda_{ij}(t) \\ \infty & \text{o.w.} \end{cases} \\
&= \begin{cases} \dfrac{\mu C_{ij}(t)}{[\mu C_{ij}(t) - \lambda_{ij}(t)] * [\mu C_{ij}(t) - \lambda_{ij}(t) - r(t)]} & \\ & \text{if } \mu C_{ij}(t) > \lambda_{ij}(t) - r(t) \\ \infty & \text{o.w.} \end{cases} \\
&= \frac{1}{\mu C_{ij}(t) * [1 - \rho_{ij}(t)]^2}
\end{aligned}
$$

$$l_{ij}^{future} = \frac{1 + V_{ij}(t)}{\mu C_{ij}(t)}$$

$$= \begin{cases} \dfrac{1}{\mu C_{ij}(t) - r(t) * [1 + V_{ij}(t)]} & \text{if } \mu C_{ij}(t) > r(t) * [1 + V_{ij}(t)] \\ \infty & \text{o.w.} \end{cases}$$

$$= \begin{cases} \dfrac{1}{\mu C_{ij}(t) - \lambda_{ij}(t) - r(t)} & \text{if } \mu C_{ij}(t) > \lambda_{ij} + r(t) \\ \infty & \text{o.w.} \end{cases}$$

5.5.4 State Constraints

In this section, we define constraints on the number of virtual circuits $V_{ij[sd]}(t) \geq 0$ and packets $N_{ij[sd]}(t) \geq 0$ at the network resources.

The total expected number of virtual circuits at every link ij should be less than the virtual circuit "capacity" on link ij (for example the number of buffers), $\sum_{[sd]} V_{ij[sd]}(t) \leq \Omega_{ij}(t)$.

Although controlling the total number of virtual circuits and packets in the network is optimum from the system point of view, it may also be unfair to some users. Some aggressive users (a source, a destination or a virtual circuit) may congest the network. If the flow and congestion control operate without paying attention to the identity of virtual circuits and packets, other users may be unfairly penalized. So, we point out three identities that should also be controlled for fairness reasons:

1) *each source*:

In order that source $[s.]$ does not monopolize the network resources, the total expected number of virtual circuits originated from node $[s.]$ should be less than the network "capacity" for virtual circuits from source $[s.]$, $\sum_{[.d]} \sum_{ij} V_{ij[sd]}(t) \leq \Omega_{[s.]}$.

Also, in order that source $[s.]$ does not monopolize path $\pi[sd]$, the total expected number of virtual circuits originated at node $[s.]$ on path $\pi[sd]$ should be less than the path's "capacity" for virtual circuits originated at $[s.]$, $\sum_{[.d1]} \sum_{ij} V_{ij[sd_1]}(t) *$

$1_{ij\in\pi[sd]}(t) \leq \Omega_{[s.],\pi[sd]}$. Finally, in order that source $[s.]$ does not monopolize link ij, the total expected number of virtual circuits originated at node $[s.]$ on link ij should be less than the link's "capacity" for virtual circuits originated at $[s.]$, $\sum_{[.d]} V_{ij[sd]}(t) \leq \Omega_{[s.],ij}$.

Similarly, we may impose restrictions on the number of packets as for datagram networks (see section 5.4.8).

2) *each destination:*

Similarly, in order that destination $[.d]$ does not monopolize the network resources, the total expected number of virtual circuits destined to node $[.d]$ should be less than the network "capacity" for virtual circuits to destination $[.d]$, $\sum_{[s.]}\sum_{ij} V_{ij[sd]}(t) \leq \Omega_{[.d]}$. Also, in order that destination $[.d]$ does not monopolize path $\pi[sd]$, the total expected number of virtual circuits destined to node $[.d]$ on path $\pi[sd]$ should be less than the path's "capacity" for virtual circuits destined to $[.d]$, $\sum_{[s_1.]}\sum_{ij} V_{ij[s_1d]}(t) * 1_{ij\in\pi[sd]}(t) \leq \Omega_{[.d],\pi[sd]}$. Finally, in order that destination $[.d]$ does not monopolize link ij, the total expected number of virtual circuits destined to node $[.d]$ on link ij should be less than the link's "capacity" for virtual circuits destined to $[.d]$, $\sum_{[s.]} V_{ij[sd]}(t) \leq \Omega_{[.d],ij}$.

Similarly, we may impose restrictions on the number of packets as for datagram networks (see section 5.4.8).

3) *each class or each single virtual circuit:*

We can consider each virtual circuit as a different class, therefore the following restrictions which apply for each $[sd]$ class of virtual circuits may also apply for each virtual circuit. So, in order that $[sd]$ virtual circuits do not monopolize the network resources, the total expected number of $[sd]$ virtual circuits should be less than the network capacity for $[sd]$ virtual circuits, $\sum_{ij} V_{ij[sd]}(t) \leq \Omega_{[sd]}$. Also, in order that $[sd]$ virtual circuits do not monopolize path $\pi[sd]$, the total expected number of $[sd]$ virtual circuits on path $\pi[sd]$ should be less than the end-to-end window size for $[sd]$ virtual circuits on path $\pi[sd]$, $\sum_{ij} V_{ij[sd]}(t) * 1_{ij\in\pi[sd]}(t) \leq \Omega_{\pi[sd],[sd]}$. Finally, in order that $[sd]$ virtual circuits do not monopolize link ij, the expected number

of $[sd]$ virtual circuits link ij, should be less than the buffer (or window) size for $[sd]$ virtual circuits on link ij, $V_{ij[sd]}(t) \leq \Omega_{ij[sd]}(t)$.

Similarly, we may impose restrictions on the number of packets as for datagram networks (see section 5.4.8).

5.6 Example

5.6.1 Introduction

In this section, we solve the decentralized dynamic joint routing and congestion control problem for multi-class multi-destination dynamic virtual circuit networks.

Two of the most important algorithms for efficient virtual circuit network control are routing and congestion control. *Routing* decides which route the virtual circuit will follow from source to destination. *Congestion control* prevents network overload by controlling the virtual circuit traffic entering the network. Routing and congestion control are strongly related problems and each affects the other. For a more accurate model and better network performance, both problems should be modeled and solved simultaneously. Such an approach however may increase the modeling and optimization complexity. Previous studies on virtual circuit network control usually concentrate on the routing problem.

In virtual circuit networks, a call set-up packet, which may be part of the first packet of a message, initiates the establishment of a virtual path from source to destination. All other packets belonging to this message follow the same route which remains fixed for the duration of the call. In this way, a virtual circuit provides a reliable logical channel with packets delivered in order. The route selection for each virtual circuit is the virtual circuit routing problem.

First, we introduce a nonlinear dynamic queueing model for virtual circuit networks that considers the dynamic interaction among the multi-class multi-destination virtual circuit and packet processes. We also define a multi-objective cost function of rejecting, setting up & maintaining virtual circuits, as well as of the packet delay and throughput.

Then we formulate the joint problem as an optimal control problem. Necessary optimality conditions are provided by Pontryagin's maximum principle. Sufficient optimality conditions based on the convexity of the Hamiltonian function are also given. For the finite horizon, the optimal controls can be found after numerically solving a Two-Point Boundary-Value Problem. For the long-run stationary equilibrium, we derive the state dependent routing and congestion controls. We show (via simulation) that when the updating period is not much larger than the mean interarrival time of virtual circuits, this state dependent routing algorithm and a shortest queue routing algorithm are showed (via simulation) to be superior to the optimal quasi-static routing.

5.6.2 Virtual Circuit Network Model

Consider an arbitrary network topology with multiple classes of virtual circuit traffic between multiple source-destination pairs (Figure 5.1)

Instead of introducing an extra notational index for each class of virtual circuits, we can consider each class c of virtual circuits between a source-destination pair $[sd]$ as being established between a fictitious $[s_c d_c]$ pair, where physically $s_c = s$ and $d_c = d, \ \forall\, c$. The queueing models that we introduce in this section can handle this substitution. Note also that one extreme case is to consider each virtual circuit as a different class. Another extreme case is to consider all virtual circuits as belonging to the same class. Also, in contemporary networks, the nodal processing delays are negligible compared to the transmission and propagation delays and therefore they were ignored in network optimization and control procedures. However, in future high speed networks, the transmission delays will be very short and comparable to the nodal processing delays. Therefore, packets will be queued not only in front of the links but also in front of the nodes (Figure 5.2). However, instead of introducing extra variables to describe the state of each node, we can consider each node i as a link $i_1 i_2$. So, in the following analysis, the word "link" may mean physically either a link or a node.

Virtual circuits arrive at a source node s (according to a Poisson distribution) destined for a destination node d with rate $\gamma_{[sd]}(t) \geq 0$ (Figure 5.3).

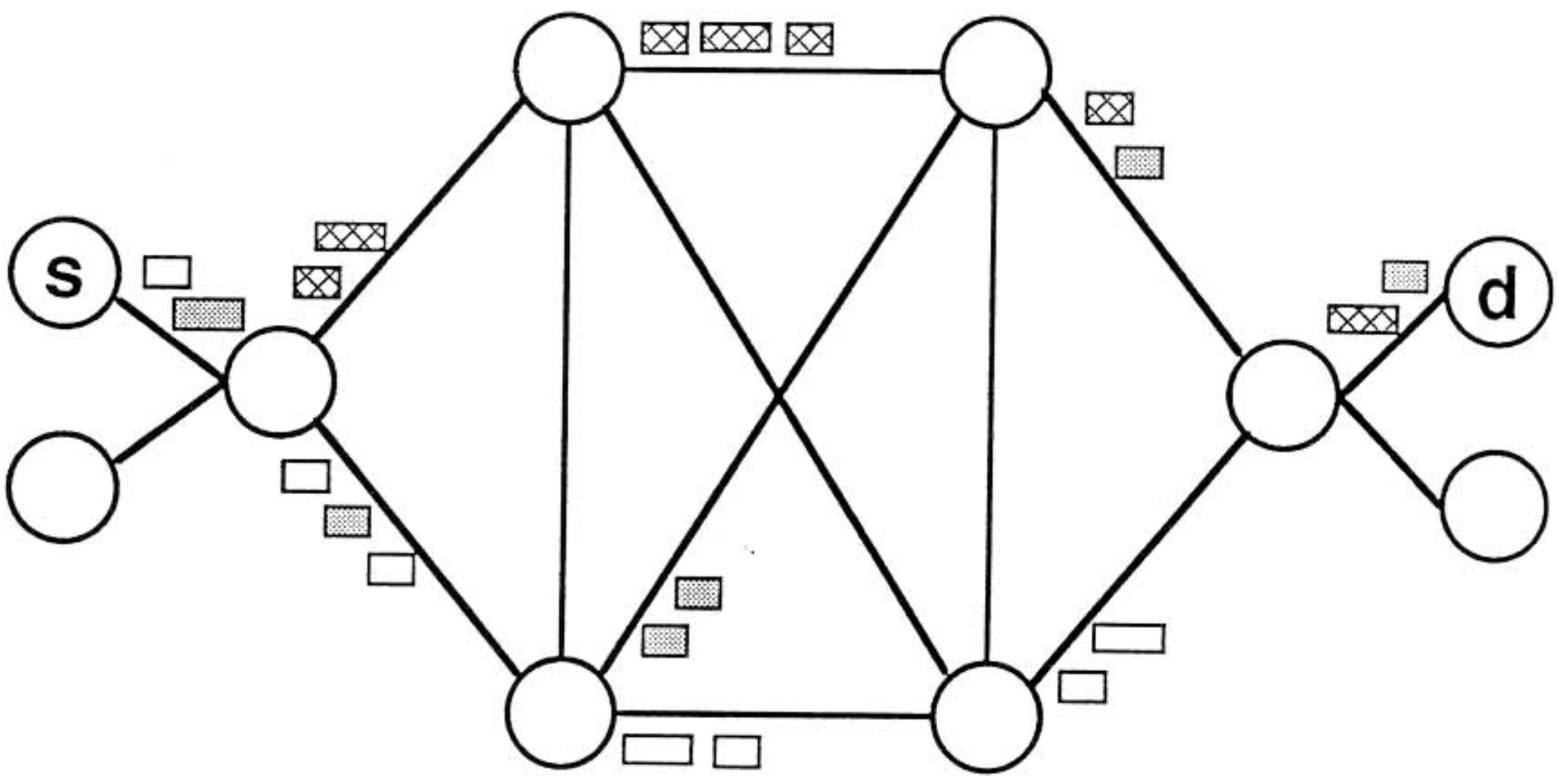


Figure 5.1: A Virtual Circuit Network.

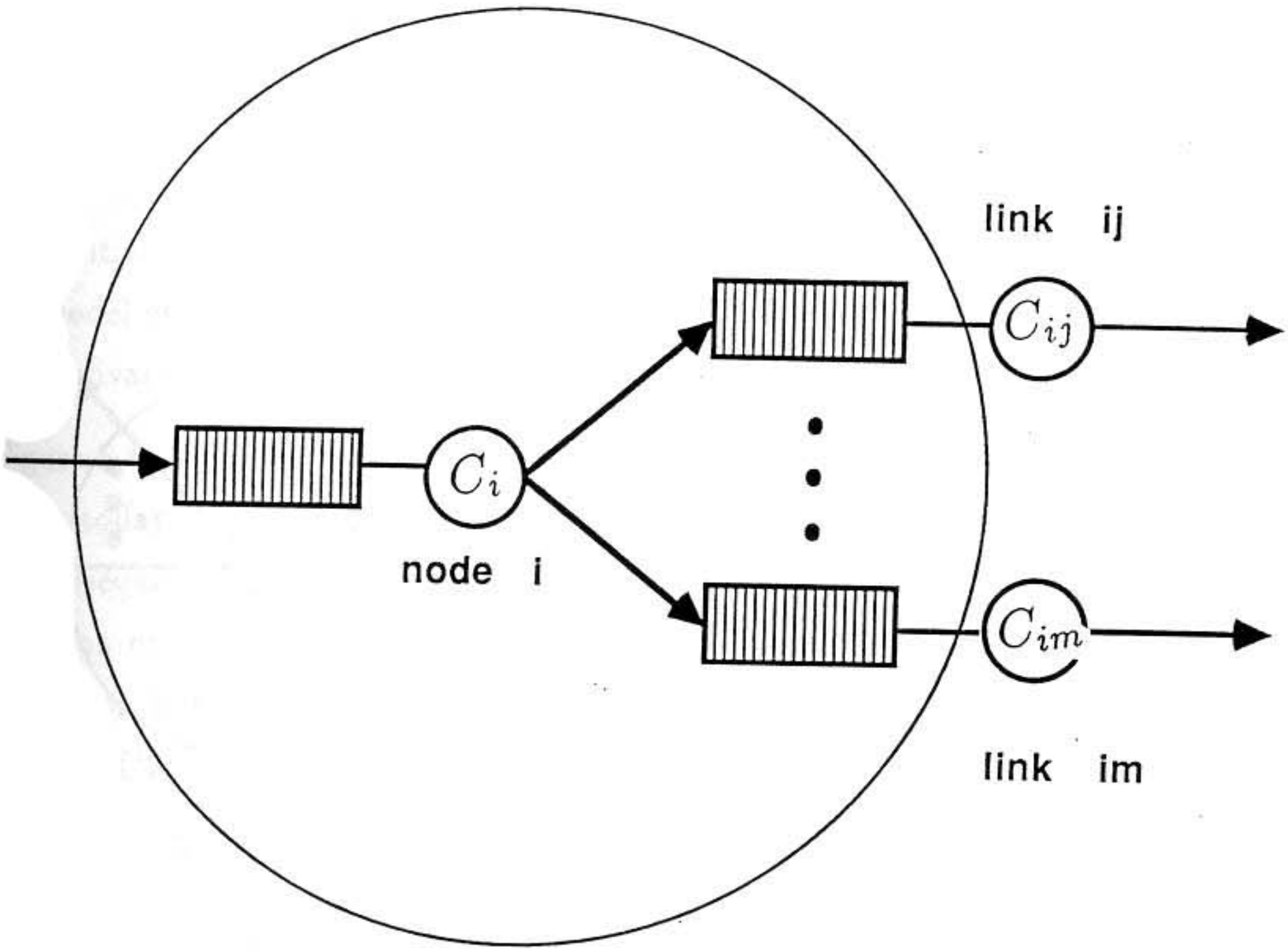


Figure 5.2: A network node.

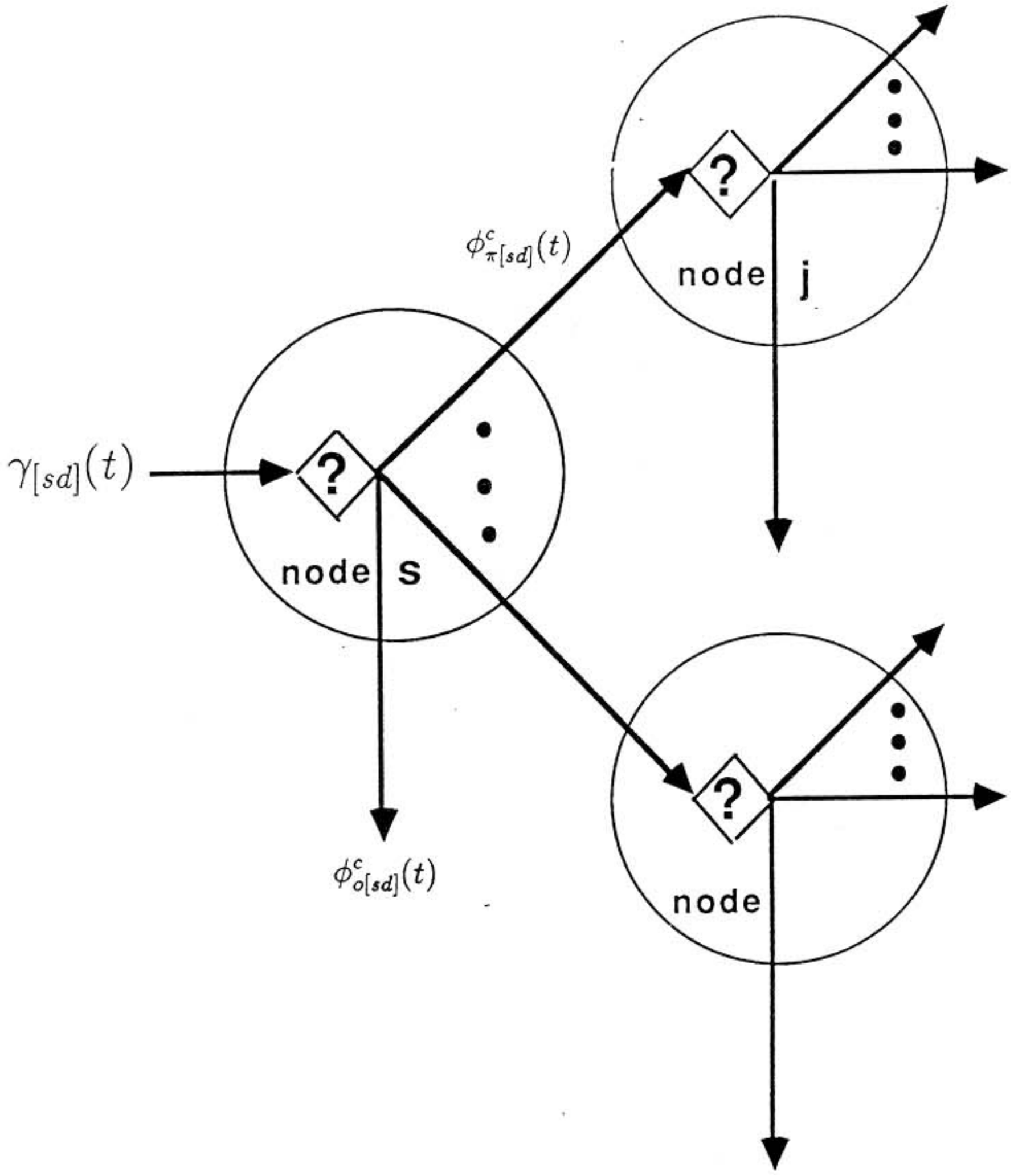


Figure 5.3: Virtual circuit routing and congestion control

For congestion control reasons, a fraction $\phi_{o[sd]}(t) \in [0,1]$ of these externally arriving $[sd]$ virtual circuits is rejected, while the remaining virtual circuits are accepted into the network. A fraction $\phi_{\pi[sd]}(t) \in [0,1]$ of the externally arriving $[sd]$ virtual circuits are routed from node s to its destination node d through path $\pi[sd]$, where $\phi_{o[sd]}(t) + \sum_{\pi[sd]} \phi_{\pi[sd]}(t) = 1$. Then the rejected $[sd]$ virtual circuit flow at the source node s is $\gamma_{[sd]}(t) * \phi_{o[sd]}(t)$ and the $[sd]$ virtual circuit flow on path $\pi[sd]$ is $\gamma_{[sd]}(t) * \phi_{\pi[sd]}(t)$. The above procedure happens for every source-destination pair in the network. Therefore the $[sd]$ virtual circuit flow on link ij is the sum of the $[sd]$ virtual circuit flows of all paths traversing this link, i.e. $\sum_{\pi[sd]} \gamma_{[sd]}(t) * \phi_{\pi[sd]}(t) * 1_{ij \in \pi[sd]}(t)$.

Finally, each $[sd]$ virtual circuit stays in the network for some time duration exponentially distributed with mean $1/\delta_{[sd]}(t) \geq 0$ and then terminates. So, we can model every link ij for the $[sd]$ virtual circuit process as an $M/M/\infty$ queue with arrival rate $\sum_{\pi[sd]} \gamma_{[sd]}(t) * \phi_{\pi[sd]}(t) * 1_{ij \in \pi[sd]}(t)$ and mean service time $1/\delta_{[sd]}(t)$ (Figure 5.4) We note that thousands of virtual circuits can coexist on a link (well within today's technology capabilities) [247].

Subsequently, we will introduce a state space approach to model the dynamic evolution of the virtual circuit processes. The expected number of $[sd]$ virtual circuits on link ij at time t, $V_{ij[sd]}(t) \geq 0$, increases during Δt by the expected number of $[sd]$ virtual circuits that arrive during this period, $\sum_{\pi[sd]} \gamma_{[sd]}(t) * \phi_{\pi[sd]}(t) * 1_{ij \in \pi[sd]}(t) * \Delta t$, minus the expected number of $[sd]$ virtual circuits that depart during this period, $\delta_{[sd]}(t) * V_{ij[sd]}(t) * \Delta t$ (Figure 5.5). So, the $[sd]$ virtual circuit process at link ij is described by

$$V_{ij[sd]}(t + \Delta t) = V_{ij[sd]}(t) + \sum_{\pi[sd]} \gamma_{[sd]}(t) * \phi_{\pi[sd]}(t) * 1_{ij \in \pi[sd]}(t) * \Delta t$$
$$- \delta_{[sd]}(t) * V_{ij[sd]}(t) * \Delta t \quad \forall\, ij \quad \forall\, [sd]$$

The expected number of $[sd]$ virtual circuits on every link ij at time t, $V_{ij[sd]}(t)$, is a continuous function of time, so let us define

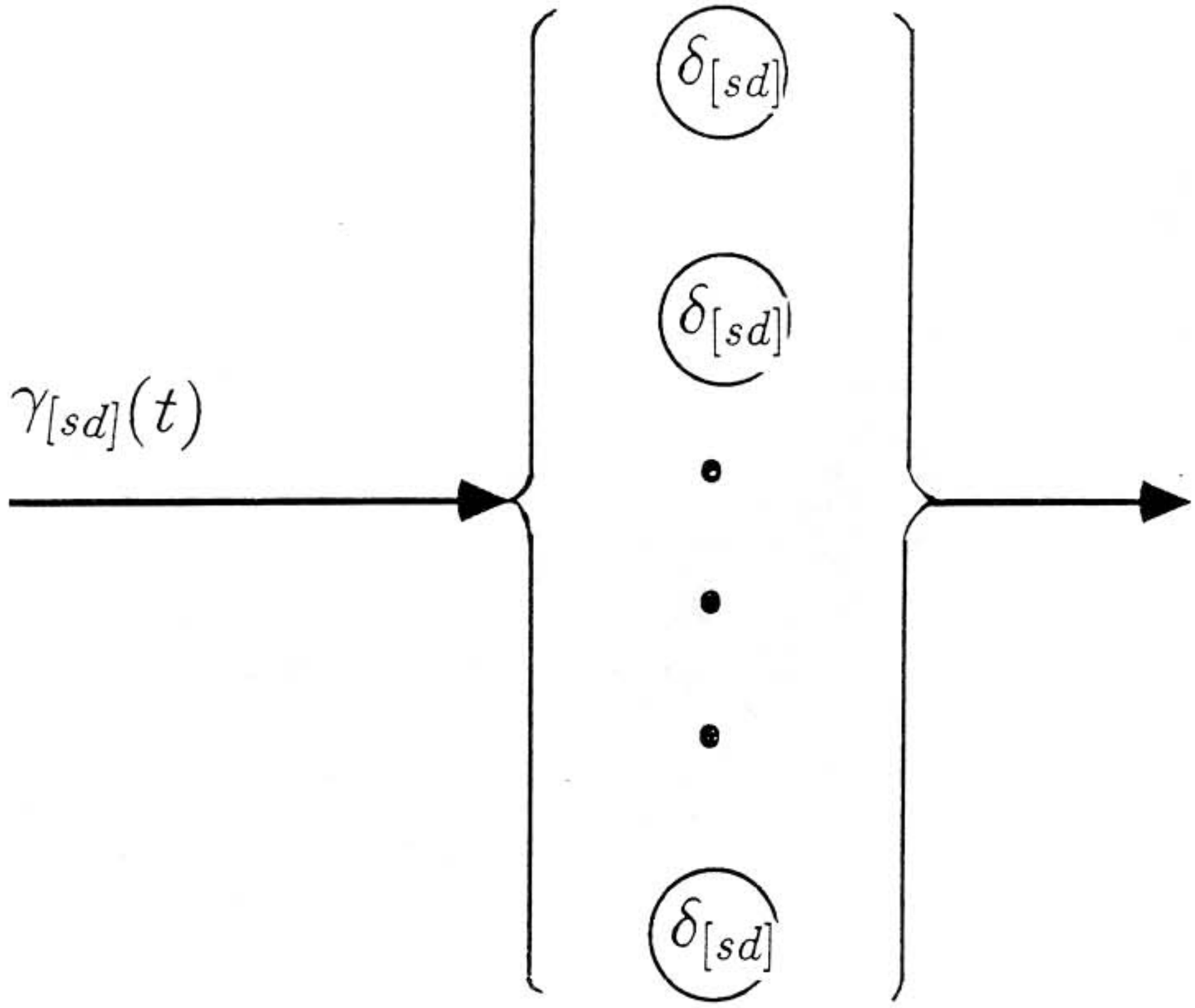


Figure 5.4: $M/M/\infty$ model for virtual circuit process.

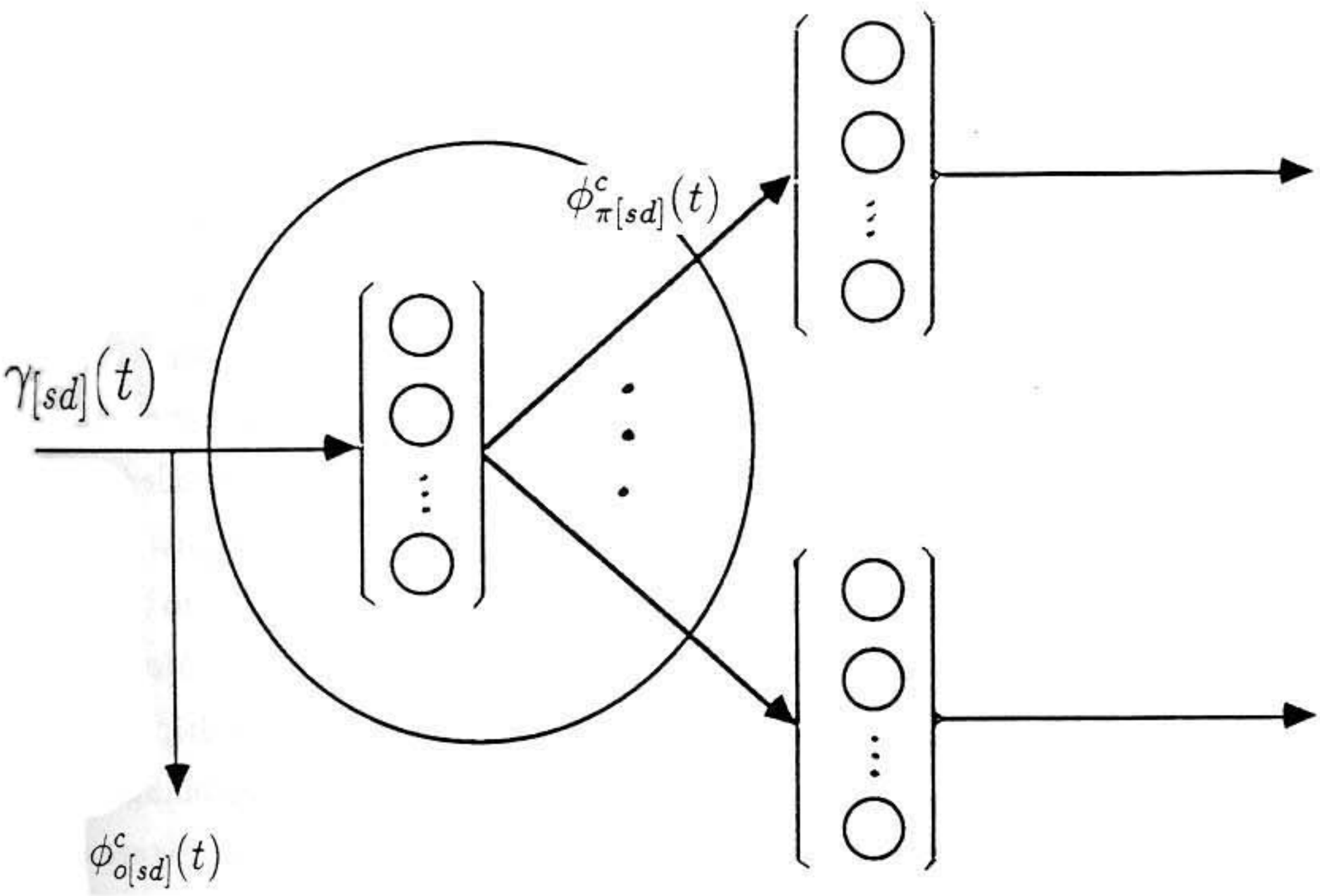


Figure 5.5: Virtual circuit processes.

$$\dot{V}_{ij[sd]}(t) = \lim_{\Delta t \to 0} \frac{V_{ij[sd]}(t + \Delta t) - V_{ij[sd]}(t)}{\Delta t} \qquad \forall\ ij \quad \forall\ [sd]$$

Therefore the $[sd]$ virtual circuit process on link ij at time t is described by

$$\dot{V}_{ij[sd]}(t) = \sum_{\pi[sd]} \gamma_{[sd]}(t) * \phi_{\pi[sd]}(t) * 1_{ij \in \pi[sd]}(t) - \delta_{[sd]}(t) * V_{ij[sd]}(t) \qquad \forall\ ij \quad \forall\ [sd]$$

Next, we describe the evolution of the packet process into the network. Let $r_{[sd]}(t) \geq 0$ be the packet arrival rate per $[sd]$ virtual circuit at time t (Poisson distribution) (Figure 5.6) If there are $V_{ij[sd]}(t)$ $[sd]$ virtual circuits on link ij at time t, then the total $[sd]$ packet arrival rate to link ij is $r_{[sd]}(t) * V_{ij[sd]}(t)$, since all packets belonging to a virtual circuit are transmitted through the same link.

Let the packet service requirement be exponentially distributed with mean $1/\mu > 0$ and the service rate at link ij be $C_{ij} > 0$. Then the mean packet service time at link ij is $1/\mu_{ij} = 1/(\mu * C_{ij})$. If the network is also controlled by link-by-link error and window flow control, then we can derive the equivalent mean packet service time at link ij [137]. Packets are serviced according to first-come-first-served or processor sharing scheduling. Katevenis [247] and Morgan [333] preallocate buffer space to each virtual circuit in every node and multiplex packets from different (thousands) virtual circuits using round-robin scheduling. So, for the $[sd]$ packet process, we model each link ij either as an $M/M/1$ (Figure 5.7) or as a Processor Sharing queue (Figure 5.8), with packet arrival rate $r_{[sd]}(t) * V_{ij[sd]}(t)$ and mean service time $1/\mu_{ij}(t)$. Note, that for the Processor Sharing discipline, the packet service requirement may be generally distributed and packets from different classes of virtual circuits may have different mean service requirements.

Let $N_{ij[sd]}(t) \geq 0$ be the expected number of $[sd]$ packets at link ij at time t and $\mathbf{N}_{ij}(t) = [...N_{ij[sd]}(t)...]^T$ be the vector of the expected number of packets on link ij for all source-destination processes. Let $\rho_{ij[sd]}(\mathbf{N}_{ij}(t))$ be the probability that there is an $[sd]$ packet at link ij (either in queue or in transmission) at time t (call this probability: "instantaneous utilization for link ij for the $[sd]$ traffic"), such that the $[sd]$ packet departure rate from link ij at time t is $\mu_{ij}(t) * \rho_{ij[sd]}(\mathbf{N}_{ij}(t))$.

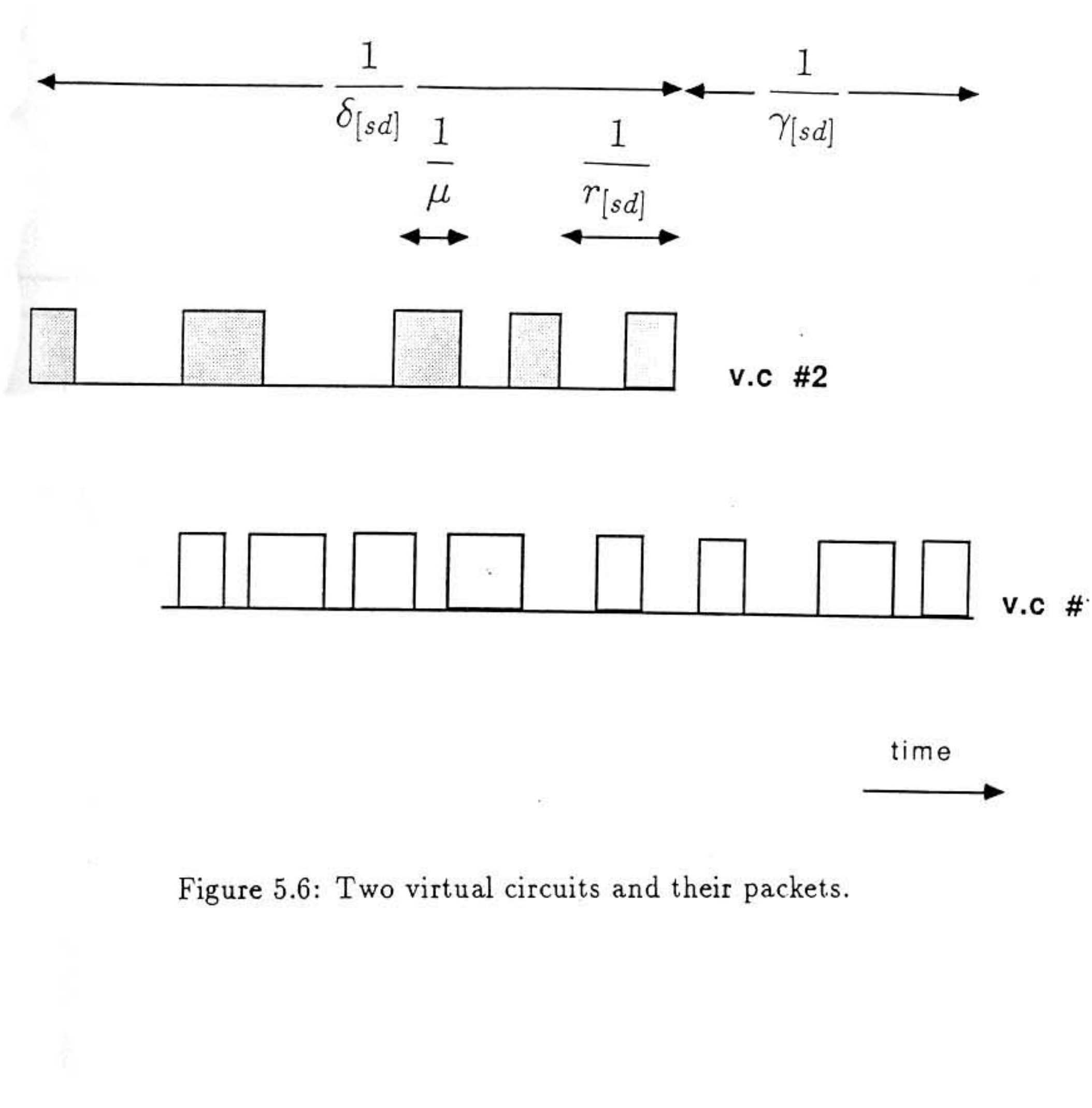


Figure 5.6: Two virtual circuits and their packets.

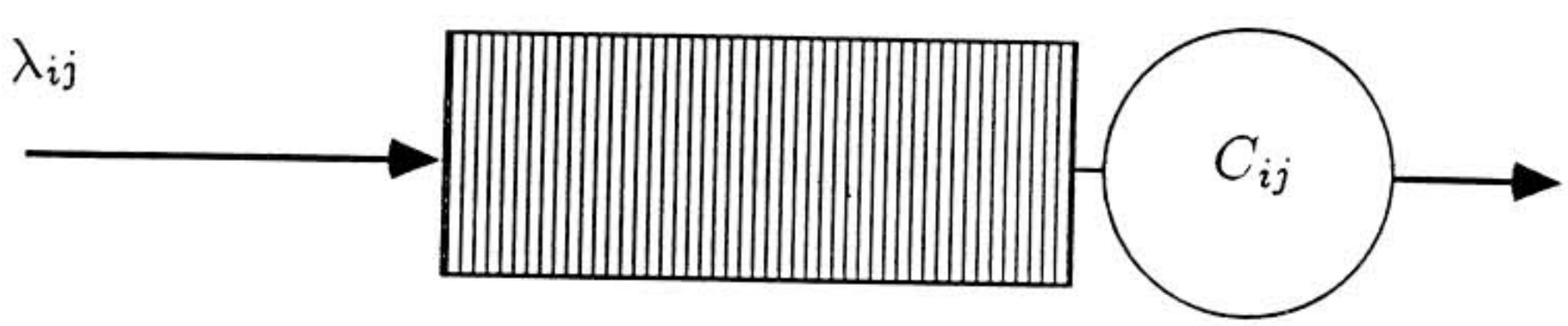


Figure 5.7: $M/M/1$ model for packet process.

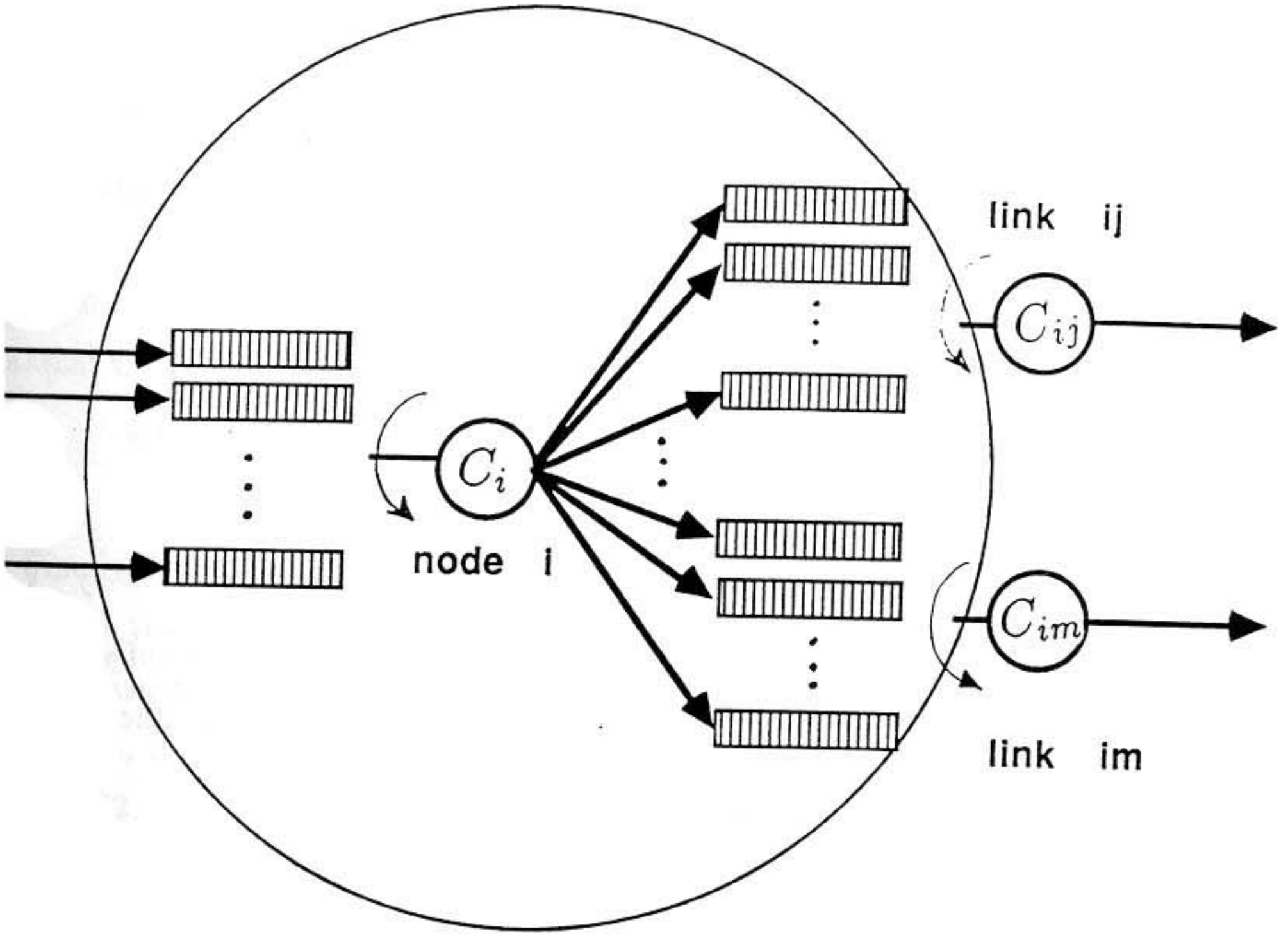


Figure 5.8: Processor Sharing model for packet process.

Then the expected number of $[sd]$ packets at link ij at time t, $N_{ij[sd]}(t)$, increases during Δt by the expected number of $[sd]$ packets that arrive during this period, $r_{[sd]}(t)*V_{ij[sd]}(t)*\Delta t$, minus the expected number of $[sd]$ packets that depart during this period, $\mu_{ij}(t) * \rho_{ij[sd]}(\mathbf{N}_{ij}(t))$. Since, the link utilization $\rho_{ij[sd]}(\mathbf{N}_{ij}(t))$, is a nonlinear function of the number of packets at link ij, $\mathbf{N}_{ij}(t)$, the $[sd]$ packet process at link ij is described by a nonlinear dynamic model

$$N_{ij[sd]}(t+\Delta t) = N_{ij[sd]}(t)+r_{[sd](t)}*V_{ij[sd]}(t)*\Delta t-\mu_{ij}(t)*\rho_{ij[sd]}(t)(\mathbf{N}_{ij}(t))*\Delta t \quad \forall\, ij,\ [sd]$$

The expected number of $[sd]$ packets at link ij at time t, $N_{ij[sd]}(t)$, is a continuous function of time. So, let us define

$$\dot{N}_{ij[sd]}(t) = \lim_{\Delta t \to 0} \frac{N_{ij[sd]}(t+\Delta t) - N_{ij[sd]}(t)}{\Delta t} \qquad \forall\, ij \quad \forall\, [sd]$$

then the $[sd]$ packet process at link ij at time t is described by

$$\dot{N}_{ij[sd]}(t) = r_{[sd]}(t) * V_{ij[sd]}(t) - \mu_{ij}(t) * \rho_{ij[sd]}(\mathbf{N}_{ij}(t)) \qquad \forall\, ij \quad \forall\, [sd]$$

The state of the network is described by the expected number of virtual circuits $V_{ij[sd]}(t)$ and of packets $N_{ij[sd]}(t)$ for each link ij for each $[sd]$ traffic. So, we define the network state as

$$\mathbf{X}(t) = \begin{bmatrix} \cdots \\ V_{ij[sd]}(t) \\ N_{ij[sd]}(t) \\ \cdots \end{bmatrix}$$

The control variables are the congestion control parameters $\phi_{o[sd]}(t)$ and the routing fractions $\phi_{\pi[sd]}(t)$ for each path $\pi[sd]$, for each $[sd]$ traffic. So, let define the control vector for the whole network as

$$\mathbf{U}(t) = \begin{bmatrix} \cdots \\ \phi_{o[sd]}(t) \\ \cdots \\ \phi_{\pi[sd]}(t) \\ \cdots \end{bmatrix}$$

In order to write the dynamic evolution of the network state in vector form, we define the following auxiliary functions

$$f_{V,ij[sd]}(t) = \sum_{\pi[sd]} \gamma_{[sd]}(t) * \phi_{\pi[sd]}(t) * 1_{ij \in \pi[sd]}(t) - \delta_{[sd]}(t) * V_{ij[sd]}(t) \quad \forall\ ij,\ [sd]$$

$$f_{N,ij[sd]}(t) = r_{[sd]}(t) * V_{ij[sd]}(t) - \mu_{ij}(t) * \rho_{ij[sd]}(\mathbf{N}_{ij}(t)) \quad \forall\ ij,\ [sd]$$

$$\mathbf{f}(t, \mathbf{X}(t), \mathbf{U}(t)) = \begin{bmatrix} \cdots \\ f_{V,ij[sd]}(t) \\ f_{N,ij[sd]}(t) \\ \cdots \end{bmatrix}$$

Then the network dynamics are described by the following nonlinear differential equation

$$\dot{\mathbf{X}}(t) = \mathbf{f}(t, \mathbf{X}(t), \mathbf{U}(t))$$

Finally, note that the $\phi_{o[sd]}$ and $\phi_{\pi[sd]}$'s represent the *fraction* of incoming flow to node s that is rejected or routed through path $\pi[sd]$. These fractions may be realized either with a *probabilistic* implementation or with a *deterministic* implementation, for example round-robin or thresholding. We discuss this further in section 1.2.

In this section, we have introduced a dynamic nonlinear queueing model for multi-class multi-destination virtual circuit networks. In the next section, we will use this nonlinear dynamic model to formulate and solve the combined routing and congestion control problem for dynamic virtual circuit networks as an optimal control problem.

5.6.3 Optimal Control Formulation

In this section, we formulate the joint routing and congestion control problem for multi-destination multi-class dynamic virtual circuit networks as an optimal control problem.

First, we define a multi-objective function $f(t, \mathbf{X}(t), \mathbf{U}(t))$ for the integrated problem. We would like to minimize the cost of rejecting virtual circuits from the network, of setting up and maintaining the virtual circuits inside the network, as well as of packet delay, while maximize the profit from servicing packets during a time interval $[t_0, t_f]$. To accomplish this, we define the following nonnegative costs and profits:

$C_{so[sd]}(t)$: cost of not admitting a new $[sd]$ virtual circuit into the network at time t.

$C_{V,ij[sd]}(t)$: cost per $[sd]$ virtual circuit for link ij at time t, for example the cost of setting up and maintaining the virtual circuit path through link ij.

$C_{N,ij[sd]}(t)$: cost per $[sd]$ packet at link ij at time t.

$C_{\mu,ij[sd]}(t)$: profit from servicing an $[sd]$ packet at link ij at time t.

So, given an initial time t_0 and a final time t_f, we define as our multi-objective function the following time-dependent function of the state $\mathbf{X}(t)$ and the controls $\mathbf{U}(t)$:

$$
\begin{aligned}
g(t, \mathbf{X}(t), \mathbf{U}(t)) = & \sum_{[sd]} C_{o[sd]}(t) * \gamma_{[sd]}(t) * \phi_{o[sd]}(t) + \\
& + \sum_{[sd]} \sum_{ij} C_{V,ij[sd]}(t) * V_{ij[sd]}(t) + \\
& + \sum_{[sd]} \sum_{ij} C_{N,ij[sd]}(t) * N_{ij[sd]}(t) - \\
& - \sum_{[sd]} \sum_{ij} C_{\mu,ij[sd]}(t) * \mu_{ij}(t) * \rho_{ij[sd]}(\mathbf{N}_{ij}(t))
\end{aligned}
$$

The first term of the objective function is the average loss of not admitting new virtual circuits into the network at every source node s for every $[sd]$ traffic. The second term is the average cost of setting up and maintaining $V_{ij[sd]}(t)$ virtual circuits on every link ij for every $[sd]$ traffic. The third term is the average cost

of packet delay at every link ij for every $[sd]$ traffic. Finally, the last term is the profit from servicing an [sd] packet on every link ij.

Next, we define the set for the controls as

$$\mathbf{V} = \{\phi_{o[sd]}(t),\ \phi_{\pi[sd]}(t)\ \ \forall\ \pi[sd]\ \ \forall\ [sd],\quad such\ that\ \ forall\ [sd]$$

$$\phi_{o[sd]}(t) \geq 0,\ \ \phi_{\pi[sd]}(t) \geq 0\ \ \forall\ \pi[sd],\ \ \phi_{o[sd]}(t) + \sum_{\pi[sd]} \phi_{\pi[sd]}(t) = 1\}$$

Nonnegative constraints on the network state $V_{ij[sd]}(t) \geq 0$ and $N_{ij[sd]}(t) \geq 0$ are always satisfied due to the structure of $\mathbf{f}(t, \mathbf{X}(t), \mathbf{U}(t))$.

Define also $P_{V,ij[sd]}(t)$ to be the costate variable for $V_{ij[sd]}(t)$, the expected number of $[sd]$ virtual circuits on link ij, and $P_{N,ij[sd]}(t)$ to be the costate variable for $N_{ij[sd]}(t)$, the expected number of $[sd]$ packets on link ij. Then the costate variable vector for all links ij for all $[sd]$ processes is $\mathbf{P}(t) = [\ldots P_{V,ij[sd]}(t)\ \ P_{N,ij[sd]}(t) \ldots]^T$.

Then our Dynamic Virtual Circuit Routing and Congestion Control problem *(DVCRCC)* is:

Problem DVCRCC:

$$\begin{aligned}
&minimize && \int_{t_0}^{t_f} g(t, \mathbf{X}(t), \mathbf{U}(t))dt \\
&with\ respect\ to && \mathbf{U}(t) \\
&such\ that && \dot{\mathbf{X}}(t) = \mathbf{f}(t, \mathbf{X}(t), \mathbf{U}(t)) \\
& && \mathbf{X}(t_0) = \mathbf{X}_0 \\
& && \mathbf{X}(t_f)\ free \\
& && \mathbf{U}(t) \in \mathbf{V}
\end{aligned}$$

where

t_0	fixed initial time,
t_f	fixed final time,
$\mathbf{X}(t)$	network state,
$\mathbf{U}(t)$	controls,
$g(t, \mathbf{X}(t), \mathbf{U}(t))$	objective function,
$\mathbf{f}(t, \mathbf{X}(t), \mathbf{U}(t))$	state dynamics,
$\mathbf{V}$	control set,
$\mathbf{X}(t_0) = \mathbf{X}_0$	initial network state,
$\mathbf{X}(t_f)$	final network state,

The Hamiltonian function of the state $\mathbf{X}(t)$, the controls $\mathbf{U}(t)$ and the costate variables $\mathbf{P}(t)$ at time t is

$$H(t, \mathbf{X}(t), \mathbf{U}(t), \mathbf{P}(t)) \quad = \quad g(t, \mathbf{X}(t), \mathbf{U}(t)) + \mathbf{P}(t) * \mathbf{f}(t, \mathbf{X}(t), \mathbf{U}(t), \mathbf{P}(t))$$

Note that the objective function g in the Hamiltonian has a multiplier equal to 1, since we have free final state conditions.

Necessary conditions for optimality are provided by Pontryagin's maximum principle [79, 33, 58, 162, 502].

Theorem 1. *Necessary conditions*

Let $\mathbf{U}^*(t)$ *be a piecewise continuous control defined on* $[t_0, t_f]$ *which solves Problem DVCRCC and let* $\mathbf{X}^*(t)$ *be the associated optimal path. Then there exists a continuous and piecewise continuously differentiable vector function* $\mathbf{P}(t) = [...P_{V,ij[sd]}(t)\ P_{N,ij[sd]}(t)...]^T$ *such that the following conditions are satisfied for all* $t \in [t_0, t_f]$,

$$\phi^*_{o[sd]}(t) \begin{cases} > 0 & only\ if\ C_{o[sd]}(t) = \min\{C_{o[sd]}, \min\limits_{p[sd]}\{\sum\limits_{ij} P_{V,ij[sd]}(t) * 1_{ij \in p[sd]}(t)\}\} \\ = 0 & o.w.\ \ \forall\ [sd] \end{cases}$$

$$\phi^*_{\pi[sd]}(t) \begin{cases} > 0 & only\ if \sum\limits_{ij} P_{V,ij[sd]}(t) * 1_{ij \in \pi[sd]}(t) = \\ & \quad = \min\{C_{o[sd]}(t), \min\limits_{p[sd]}\{\sum\limits_{ij} P_{V,ij[sd]}(t) * 1_{ij \in p[sd]}(t)\}\} \\ = 0 & o.w.\ \ \forall\ \pi[sd]\ \ \forall\ [sd] \end{cases}$$

$$\dot{V}^*_{ij[sd]}(t) = \sum_{\pi[sd]} \gamma_{[sd]}(t) * \phi^*_{\pi[sd]}(t) * 1_{ij\in\pi[sd]}(t) - \delta_{[sd]}(t) * V^*_{ij[sd]}(t) \quad \forall\ ij,\ [sd]$$

$$\dot{N}^*_{ij[sd]}(t) = r_{[sd]}(t) * V^*_{ij[sd]}(t) - \mu_{ij}(t) * \rho_{ij[sd]}(\mathbf{N}^*_{ij}(t)) \quad \forall\ ij,\ [sd]$$

$$V^*_{ij[sd]}(t_0) = V_{ij[sd],0} \quad \forall\ ij,\ [sd]$$
$$N^*_{ij[sd]}(t_0) = N_{ij[sd],0} \quad \forall\ ij,\ [sd]$$

$$\dot{P}_{V,ij[sd]}(t) = -\{\ C_{V,ij[sd]}(t) - P_{V,ij[sd]}(t) * \delta_{[sd]}(t) + P_{N,ij[sd]}(t) * r_{[sd]}(t)\ \}$$
$$\forall\ ij \quad \forall\ [sd]$$

$$\dot{P}_{N,ij[sd]}(t) = -\{\ C_{N,ij[sd]}(t) - \sum_{[s_1d_1]} C_{\mu,ij[s_1d_1]}(t) * \mu_{ij}(t) * \frac{d\rho_{ij[s_1d_1]}(\mathbf{N}^*_{ij}(t))}{dN_{ij[sd]}(t)} -$$

$$- \sum_{[s_1d_1]} P_{N,ij[s_1d_1]}(t) * \mu_{ij}(t) * \frac{d\rho_{ij[s_1d_1]}(\mathbf{N}^*_{ij}(t))}{dN_{ij[sd]}(t)}\ \} \quad \forall\ ij \quad \forall\ [sd]$$

$$P_{V,ij[sd]}(t_f) = 0 \quad \forall\ ij \quad \forall\ [sd]$$
$$P_{N,ij[sd]}(t_f) = 0 \quad \forall\ ij \quad \forall\ [sd]$$

Proof: The Hamiltonian must satisfy the following condition

$$H(t, \mathbf{X}^*(t), \mathbf{U}^*(t), \mathbf{P}(t)) \leq H(t, \mathbf{X}^*(t), \mathbf{U}, \mathbf{P}(t)) \quad \forall\ \mathbf{U} \in \mathbf{V}$$

which is equivalent to the following condition

$$\sum_{[sd]} \left\{ \gamma_{[sd]}(t) * [C_{o[sd]}(t) * \phi^*_{o[sd]}(t) + \sum_{\pi[sd]} \sum_{ij} P_{V,ij[sd]}(t) * \phi^*_{\pi[sd]}(t) * 1_{ij\in\pi[sd]}(t)] \right\} \leq$$

$$\leq \sum_{[sd]} \left\{ \gamma_{[sd]}(t) * [C_{o[sd]}(t) * \phi_{o[sd]}(t) + \sum_{\pi[sd]} \sum_{ij} P_{V,ij[sd]}(t) * \phi_{\pi[sd]}(t) * 1_{ij\in\pi[sd]}(t)] \right\}$$

$$\forall\ \phi_{o[sd]},\ \phi_{\pi[sd]} \in \mathbf{V} \quad \forall\ \pi[sd] \quad \forall\ [sd]$$

Since, there is no dependency among the controls for different source-destination pairs $[sd]$, we can decomposed the above conditions $\forall\ [sd]$ to

$$\gamma_{[sd]}(t) * [C_{o[sd]}(t) * \phi^*_{o[sd]}(t) + \sum_{\pi[sd]} \sum_{ij} P_{V,ij[sd]}(t) * \phi^*_{ij[sd]}(t) * 1_{ij\in\pi[sd]}(t)] \leq$$

$$\leq \gamma_{[sd]}(t) * [C_{o[sd]}(t) * \phi_{o[sd]}(t) + \sum_{\pi[sd]} \sum_{ij} P_{V,ij[sd]}(t) * \phi_{ij[sd]}(t) * 1_{ij \in \pi[sd]}(t)]$$

$\forall\ \phi_{o[sd]},\ \phi_{\pi[sd]} \in \mathbf{V} \quad \forall\ \pi[sd]$

Then the optimal controls satisfy the following conditions

$$\phi^*_{o[sd]}(t) \begin{cases} > 0 & \text{only if } C_{o[sd]}(t) = \min\{C_{o[sd]}(t), \min_{p[sd]}\{\sum_{ij} P_{V,ij[sd]}(t) * 1_{ij \in p[sd]}(t)\}\} \\ = 0 & \text{o.w. } \forall\ [sd] \end{cases}$$

$$\phi^*_{\pi[sd]}(t) \begin{cases} > 0 & \text{only if } \sum_{ij} P_{V,ij[sd]}(t) * 1_{ij \in \pi[sd]}(t) = \\ & = \min\{C_{o[sd]}(t), \min_{p[sd]}\{\sum_{ij} P_{V,ij[sd]}(t) * 1_{ij \in p[sd]}(t)\}\} \\ = 0 & \text{o.w. } \forall\ \pi[sd] \quad \forall\ [sd] \end{cases}$$

The optimal state and control pair $(\mathbf{X}^*(t), \mathbf{U}^*(t))$ must also satisfy the state dynamics

$\dot{\mathbf{X}}^*(t) = \mathbf{f}(t, \mathbf{X}^*(t), \mathbf{U}^*(t))$

which can be rewritten as

$$\dot{V}^*_{ij[sd]}(t) = \sum_{\pi[sd]} \gamma_{[sd]}(t) * \phi^*_{\pi[sd]}(t) * 1_{ij \in \pi[sd]}(t) - \delta_{[sd]}(t) * V^*_{ij[sd]}(t) \quad \forall\ ij,\ [sd]$$

$$\dot{N}^*_{ij[sd]}(t) = r_{[sd]}(t) * V^*_{ij[sd]}(t) - \mu_{ij}(t) * \rho_{ij[sd]}(\mathbf{N}^*_{ij}(t)) \quad \forall\ ij,\ [sd]$$

The optimal state must also satisfy the initial state $\mathbf{X}^*(t_0) = \mathbf{X}_0$, therefore

$$V^*_{ij[sd]}(t_0) = V_{ij[sd],0} \quad \forall\ ij,\ [sd]$$
$$N^*_{ij[sd]}(t_0) = N_{ij[sd],0} \quad \forall\ ij,\ [sd]$$

The costate variables must satisfy the following conditions

$\dot{\mathbf{P}}(t) = -\nabla_{\mathbf{X}} H(t, \mathbf{X}^*(t), \mathbf{U}^*(t), \mathbf{P}(t))$

which can be rewritten as

$$\dot{P}_{V,ij[sd]}(t) = -\frac{\partial H(t.\mathbf{X}^*(t).\mathbf{U}^*(t).\mathbf{P}(t))}{\partial V_{ij[sd]}(t)} =$$

$$= -\{\ C_{V,ij[sd]}(t) - P_{V,ij[sd]}(t) * \delta_{[sd]}(t) + P_{N,ij[sd]}(t) * r_{[sd]}(t)\ \}$$

$$\dot{P}_{N,ij[sd]}(t) = -\frac{\partial H(t,\mathbf{X}^*(t),\mathbf{U}^*(t),\mathbf{P}(t))}{\partial N_{ij[sd]}(t)} =$$

$$= -\{\ C_{N,ij[sd]}(t) - \sum_{[s_1 d_1]} C_{\mu,ij[s_1 d_1]}(t) * \mu_{ij}(t) * \frac{d\rho_{ij[s_1 d_1]}(\mathbf{N}^*_{ij}(t))}{dN_{ij[sd]}(t)} -$$

$$- \sum_{[s_1 d_1]} P_{N,ij[s_1 d_1]}(t) * \mu_{ij}(t) * \frac{d\rho_{ij[s_1 d_1]}(\mathbf{N}^*_{ij}(t))}{dN_{ij[sd]}(t)}\ \} \quad \forall\ ij,\ [sd]$$

Since we have no conditions on the final state $\mathbf{X}(t_f)$, the costate variables at the final time must be zero, $\mathbf{P}(t_f) = 0$. Therefore

$P_{V,ij[sd]}(t_f) = 0 \quad \forall\ ij \quad \forall\ [sd]$

$P_{N,ij[sd]}(t_f) = 0 \quad \forall\ ij \quad \forall\ [sd]$ □

Sufficient conditions for optimality are provided by the convexity of the Hamiltonian with respect to the state and the controls [320, 237, 439, 379].

Theorem 2. *Sufficient conditions*

Let $(\bar{\mathbf{X}}(t), \bar{\mathbf{U}}(t))$ *be an admissible pair in Problem DVCRCC. Assume that* $\rho_{ij[sd]}(\mathbf{N}_{ij}(t))$ *is defined for* $\mathbf{N}_{ij}(t) \geq 0$, *is concave monotonically increasing and twice differentiable in* $\mathbf{N}_{ij}(t)$. *If there exists a continuous and piecewise continuously differentiable vector function* $\mathbf{P}(t) = [...P_{V,ij[sd]}(t)\ P_{N,ij[sd]}(t)...]^T$ *such that the following conditions are satisfied for all* $t \in [t_0, t_f]$

$$\bar{\phi}_{o[sd]}(t) \begin{cases} > 0 & \text{only if } C_{o[sd]}(t) = \min\{C_{o[sd]}(t), \min\limits_{p[sd]}\{\sum\limits_{ij} P_{V,ij[sd]}(t) * 1_{ij \in p[sd]}(t)\}\} \\ = 0 & o.w. \quad \forall\ [sd] \end{cases}$$

$$\bar{\phi}_{\pi[sd]}(t) \begin{cases} > 0 & only\ if \sum_{ij} P_{V,ij[sd]}(t) * 1_{ij \in \pi[sd]}(t) = \\ & = \min\{C_{o[sd]}(t), \min_{p[sd]}\{\sum_{ij} P_{V,ij[sd]}(t) * 1_{ij \in p[sd]}(t)\}\} \\ = 0 & o.w. \ \ \forall\ \pi[sd] \ \ \forall\ [sd] \end{cases}$$

$$\dot{\bar{V}}_{ij[sd]}(t) = \sum_{\pi[sd]} \gamma_{[sd]}(t) * \bar{\phi}_{\pi[sd]}(t) * 1_{ij \in \pi[sd]}(t) - \delta_{[sd]}(t) * \bar{V}_{ij[sd]}(t) \ \ \forall\ ij,\ [sd]$$

$$\dot{\bar{N}}_{ij[sd]}(t) = r_{[sd]}(t) * \bar{V}_{ij[sd]}(t) - \mu_{ij}(t) * \rho_{ij[sd]}(\bar{\mathbf{N}}_{ij}(t)) \ \ \forall\ ij,\ [sd]$$

$$\bar{V}_{ij[sd]}(t_0) = V_{ij[sd],0} \quad \forall\ ij \ \ \forall\ [sd]$$
$$\bar{N}_{ij[sd]}(t_0) = N_{ij[sd],0} \quad \forall\ ij \ \ \forall\ [sd]$$

$$\dot{P}_{V,ij[sd]}(t) = -\{\ C_{V,ij[sd]}(t) - P_{V,ij[sd]}(t) * \delta_{[sd]}(t) + P_{N,ij[sd]}(t) * r_{[sd]}(t)\ \}$$

$$\dot{P}_{N,ij[sd]}(t) = -\{\ C_{N,ij[sd]}(t) - \sum_{[s_1 d_1]} C_{\mu,ij[s_1 d_1]}(t) * \mu_{ij}(t) * \frac{d\rho_{ij[s_1 d_1]}(\mathbf{N}_{ij}(t))}{dN_{ij[sd]}(t)} -$$

$$- \sum_{[s_1 d_1]} P_{N,ij[s_1 d_1]}(t) * \mu_{ij}(t) * \frac{d\rho_{ij[s_1 d_1]}(\bar{\mathbf{N}}_{ij}(t))}{dN_{ij[sd]}(t)}\ \} \quad \forall\ ij,\ [sd]$$

$$P_{N,ij[sd]}(t) \geq 0 \quad \forall\ ij \ \ \forall\ [sd]$$

$$P_{V,ij[sd]}(t_f) = 0 \quad \forall\ ij \ \ \forall\ [sd]$$
$$P_{N,ij[sd]}(t_f) = 0 \quad \forall\ ij \ \ \forall\ [sd]$$

then $(\bar{\mathbf{X}}(t), \bar{\mathbf{U}}(t))$ *is optimal.*

Proof: The first part of the proof is similar to that of Theorem 1.

In addition, the control set $\mathbf{V}$ is a convex set and since $-\rho_{ij[sd]}(\mathbf{N}_{ij}(t))$ is a convex (i.e. $\rho_{ij[sd]}(\mathbf{N}_{ij}(t))$ is concave) and differentiable function in $\mathbf{N}_{ij}(t)$, our objective function $g(t, \mathbf{X}(t), \mathbf{U}(t))$, as well as each component of $\mathbf{f}(t, \mathbf{X}(t), \mathbf{U}(t))$

are differentiable and convex functions in the variables $(\mathbf{X}(t), \mathbf{U}(t))$ for $t \in [t_0, t_f]$. Furthermore, if $P_{N,ij[sd]}(t) \geq 0 \quad \forall\ ij \quad \forall\ [sd]$, then the Hamiltonian function $H(t, \mathbf{X}(t), \mathbf{U}(t), \mathbf{P}(t))$ is a convex function in $(\mathbf{X}(t), \mathbf{U}(t))$ for $t \in [t_0, t_f]$ (we need nonnegativity of the costate variables only for those components of $\mathbf{f}(t, \mathbf{X}(t), \mathbf{U}(t))$ that are nonlinear in $\mathbf{X}(t)$ [320, 237, 439, 379]).

If all the above conditions are satisfied, then $(\bar{\mathbf{X}}(t), \bar{\mathbf{U}}(t))$ is optimal. □

Note that for an $M/M/1$ or Processor Sharing queue at steady state, $\rho_{ij[sd]}(\mathbf{N}_{ij}) = \frac{N_{ij[sd]}}{1 + \sum_{[s_1 d_1]} \rho_{ij[s_1 d_1]}}$ is defined for $\mathbf{N}_{ij} \geq 0$, is concave, monotonically increasing and twice differentiable in $\mathbf{N}_{ij}$ with $\lim_{\mathbf{N}_{ij} \to \infty} \rho_{ij}(\mathbf{N}_{ij}) = 1$.

So, after numerically solving a two-Point Boundary-Value Problem (TPBVP), we have the optimal congestion control and routing decisions. Numerical methods [14, 79, 203, 254, 292, 415] for the solution of such problems involve either flooding or iterative procedures. *Flooding* (or dynamic programming) procedures start from a point that satisfies one boundary condition and generates a trajectory. This is repeated many times until one of these trajectories satisfies the other condition or an interpolation of these trajectories can give an acceptable solution. *Iterative* procedures use successive linearization. A nominal solution is chosen such that to satisfy one or more of the following conditions: 1) state differential equations, 2) adjoint differential equations, 3) optimality conditions, 4) boundary conditions. Then this nominal solution is modified by successive linearization such that the remaining conditions are also satisfied. Three classes of iterative procedures may be used: i) *neighboring extremal*, ii) *gradient*, and iii) *quasi-linearization* procedures.

In this section, we are primarily interested in the optimal control formulation for the finite horizon problem and the long-run stationary equilibrium solution. So, we will not discuss further numerical techniques for the finite horizon optimal control problem. In this section, we formulated the combined routing and congestion control problem for multi-destination multi-class dynamic virtual circuit networks as an optimal control problem. Then for specific network configuration and traffic characteristics, we can find the optimum congestion control and routing decisions

by solving a TPBVP. We can decompose the above problem to many smaller subproblems, one for every source-destination. However, numerical solution may require long computational times for on line implementation. Therefore, in the next section, we also derive state dependent routing and congestion controls for the long-run stationary equilibrium that can be used for on-line implementation.

5.6.4 State Dependent Routing & Congestion Control

In this section, we consider a network with constant arrival rates and mean durations of virtual circuits, as well as constant costs and profits (autonomous system), and we find optimal state dependent virtual circuit routing and congestion controls for the long-run stationary equilibrium. Our problem becomes

$$
\begin{aligned}
minimize \qquad & \sum_{[sd]} C_{o[sd]} * \gamma_{[sd]} * \phi_{o[sd]} + \\
& + \sum_{[sd]} \sum_{ij} C_{V,ij[sd]} * V_{ij[sd]} + \\
& + \sum_{[sd]} \sum_{ij} C_{N,ij[sd]} * N_{ij[sd]} - \\
& - \sum_{[sd]} \sum_{ij} C_{\mu,ij[sd]} * \mu_{ij} * \rho_{ij[sd]}(\mathbf{N}_{ij})
\end{aligned}
$$

$$\textit{with respect to} \quad \text{the congestion controls } \phi_{o[sd]} \geq 0 \qquad \forall\ [sd]$$

$$\text{the routing fractions } \phi_{\pi[sd]} \geq 0 \qquad \forall\ \pi[sd]\ \ \forall\ [sd]$$

$$\textit{such that} \quad 0 = \sum_{\pi[sd]} \gamma_{[sd]} * \phi_{\pi[sd]} * 1_{ij \in \pi[sd]} - \delta_{[sd]} * V_{ij[sd]} \qquad \forall\ ij\ \ \forall\ [sd]$$

$$0 = r_{[sd]} * V_{ij[sd]} - \mu_{ij} * \rho_{ij[sd]}(\mathbf{N}_{ij}) \qquad \forall\ ij\ \ \forall\ [sd]$$

$$\phi_{o[sd]},\ \ \phi_{\pi[sd]} \geq 0 \qquad \forall\ \pi[sd]\ \ \forall\ [sd]$$

$$\phi_{o[sd]} + \sum_{\pi[sd]} \phi_{\pi[sd]} = 1 \qquad \forall\ [sd]$$

The minimization of the Hamiltonian with respect to the congestion control and routing fractions is equivalent to the following minimization problem

$$\textit{minimize} \sum_{[sd]} \left\{ \gamma_{[sd]} * [C_{o[sd]} * \phi_{o[sd]} + \sum_{\pi[sd]} \sum_{ij} P_{V,ij[sd]} * \phi_{\pi[sd]} * 1_{ij \in \pi[sd]}] \right\}$$

$$\textit{with respect to} \quad \phi_{o[sd]}, \phi_{\pi[sd]}, \quad \forall\ \pi[sd],\ [sd]$$

$$\textit{such that} \quad \phi_{o[sd]} + \sum_{\pi[sd]} \phi_{\pi[sd]} = 1 \quad \phi_{o[sd]},\ \phi_{\pi[sd]} \geq 0 \quad \forall\ \pi[sd] \quad \forall\ [sd]$$

where the costate variables $P_{V,ij[sd]}$ for the expected number of virtual circuits and the costate variables $P_{N,ij[sd]}$ for the expected number of packets for each link ij, for each $[sd]$ pair will be found later.

The above problem can be decomposed for each source-destination pair $[sd]$ to the following problem

$$\textit{minimize} \qquad \gamma_{[sd]} * [C_{o[sd]} * \phi_{o[sd]} + \sum_{\pi[sd]} \sum_{ij} P_{V,ij[sd]} * \phi_{ij[sd]} * 1_{ij \in \pi[sd]}]$$

$$\textit{with respect to} \qquad \phi_{o[sd]}, \phi_{\pi[sd]} \quad \forall\ \pi[sd]$$

$$\textit{such that} \qquad \phi_{o[sd]} + \sum_{\pi[sd]} \phi_{\pi[sd]} = 1, \quad \phi_{o[sd]},\ \phi_{\pi[sd]} \geq 0 \quad \forall\ \pi[sd]$$

Define the minimum cost at source node s for the $[sd]$ virtual circuit traffic to be $P^*_{V,s[sd]} = \min\{C_{o[sd]}, \min_{p[sd]}\{\sum_{ij} P_{V,ij[sd]} * 1_{ij \in p[sd]}\}\}$. Then the optimum congestion controls are:

$$\phi^*_{o[sd]} \begin{cases} > 0 & \text{only if } C_{o[sd]} = P^*_{V,s[sd]} \\ = 0 & \text{o.w.} \end{cases}$$

and the optimum routing fractions are:

$$\phi^*_{\pi[sd]} \begin{cases} > 0 & \text{only if } \sum_{ij} P_{V,ij[sd]} * 1_{ij \in \pi[sd]} = P^*_{V,s[sd]} \\ = 0 & \text{o.w.} \end{cases}$$

Therefore, an $[sd]$ virtual circuit is rejected at source node s only if the cost of rejecting it is equal to the minimum cost at node i, i.e. $C_{o[sd]} = P^*_{V,s[sd]}$. Also, path $\pi[sd]$ will be used for the $[sd]$ traffic only if its costate variable achieves the minimum cost, i.e. $\sum_{ij} P_{V,ij[sd]} * 1_{ij \in \pi[sd]} = P^*_{V,s[sd]}$.

When the congestion control and routing fractions achieve their optimum values, we have

$$C_{o[sd]} * \phi^*_{o[sd]} + \sum_{ij} \sum_{\pi[sd]} P_{V,ij[sd]} * \phi^*_{ij[sd]} * 1_{ij \in \pi[sd]} = P^*_{V,s[sd]}$$

The optimum congestion control and routing decisions depend on the values of the costate variables $P_{V,ij[sd]} \quad \forall\ ij \quad \forall\ [sd]$. So, we have to calculate the costate variables $P_{V,ij[sd]}$ for each link ij for each $[sd]$ traffic.

At steady state, the costate variables must satisfy $\dot{P}_{V,ij[sd]} = 0 \quad \forall\, ij,\ [sd]$.

$$\dot{P}_{V,ij[sd]} = 0 \Rightarrow \quad -\{\ C_{V,ij[sd]} - P_{V,ij[sd]} * \delta_{[sd]} + P_{N,ij[sd]} * r_{[sd]}\ \} = 0 \quad \forall\, ij,\ [sd]$$

Then

$$P_{V,ij[sd]} \quad = \frac{C_{V,ij[sd]}}{\delta_{[sd]}} + \frac{r_{[sd]}}{\delta_{[sd]}} * P_{N,ij[sd]} \quad \forall\, ij \quad \forall\, [sd]$$

Next, in order to calculate the costate variables $P_{V,ij[sd]}$ for the expected number of $[sd]$ virtual circuits, we must first calculate the costate variables $P_{N,ij[sd]}$ for the expected number of $[sd]$ packets. At steady state, the costate variables must satisfy $\dot{P}_{N,ij[sd]} = 0 \quad \forall\, ij \quad \forall\, [sd]$.

$$\dot{P}_{N,ij[sd]} = 0 \Rightarrow \quad -\{\ C_{N,ij[sd]} - \sum_{[s_1 d_1]} C_{\mu,ij[s_1 d_1]} * \mu_{ij} * \frac{d\rho_{ij[s_1 d_1]}(\mathbf{N}^*_{ij})}{dN_{ij[sd]}} -$$

$$- \sum_{[s_1 d_1]} P_{N,ij[s_1 d_1]} * \mu_{ij} * \frac{d\rho_{ij[s_1 d_1]}(\mathbf{N}^*_{ij})}{dN_{ij[sd]}}\ \} = 0 \quad \forall\, ij \quad \forall\, [sd]$$

In order to find the costate variables $P_{N,ij[sd]}$ for the expected number of $[sd]$ packets on every link ij, we must solve a system of equations for all source-destination processes that use this link:

$$C_{N,ij[sd]} - \sum_{[s_1 d_1]} C_{\mu,ij[s_1 d_1]} * \mu_{ij} * \frac{d\rho_{ij[s_1 d_1]}(\mathbf{N}^*_{ij})}{dN_{ij[sd]}} -$$

$$- \sum_{[s_1 d_1]} P_{N,ij[s_1 d_1]} * \mu_{ij} * \frac{d\rho_{ij[s_1 d_1]}(\mathbf{N}^*_{ij})}{dN_{ij[sd]}} = 0 \quad \forall\, [sd]$$

Note that for an $M/M/1$ or Processor Sharing queueing model the expected number of $[sd]$ packets on link ij at steady state is

$$N_{ij[sd]} = \frac{\rho_{ij[sd]}}{1 - \sum_{[s_1 d_1]} \rho_{ij[s_1 d_1]}} \quad \forall\, [sd]$$

Solving the above system of equations (for all $[sd]$ traffic that use link ij), we have the utilization of link ij for each $[sd]$ process at steady state

$$\rho_{ij[sd]} = \frac{N_{ij[sd]}}{1 + \sum_{[s_1 d_1]} N_{ij[s_1 d_1]}}$$

Therefore we can rewrite the $P_{N,ij[sd]}$ costate variable system of equations for each link ij, as

$$C_{N,ij[sd]} - \sum_{[s_1 d_1]} C_{\mu,ij[s_1 d_1]} * \mu_{ij}$$

$$* \left\{ \frac{1 + \sum_{[s_2 d_2] \neq [sd]} N^*_{ij[s_2 d_2]}}{(1 + \sum_{[s_1 d_1]} N^*_{ij[s_1 d_1]})^2} \sum_{[s_2 d_2] \neq [sd]} \frac{N^*_{ij[s_2 d_2]}}{(1 + \sum_{[s_1 d_1]} N^*_{ij[s_1 d_1]})^2} \right\} -$$

$$- \sum_{[s_1 d_1]} P_{N,ij[s_1 d_1]} * \mu_{ij} * \left\{ \frac{1 + \sum_{[s_2 d_2] \neq [sd]} N^*_{ij[s_2 d_2]}}{(1 + \sum_{[s_1 d_1]} N^*_{ij[s_1 d_1]})^2} - \sum_{[s_2 d_2] \neq [sd]} \frac{N^*_{ij[s_2 d_2]}}{(1 + \sum_{[s_1 d_1]} N^*_{ij[s_1 d_1]})^2} \right\} = 0$$

$\forall\ [sd]$

The solution to the above system is

$$P_{N,ij[sd]} = \frac{1 + \sum_{[s_1 d_1]} N^*_{ij[s_1 d_1]}}{\mu_{ij}} * \left[C_{N,ij[sd]} * (1 + N^*_{ij[sd]}) + \right.$$

$$\left. + \sum_{[s_2 d_2] \neq [sd]} C_{N,ij[s_2 d_2]} * N^*_{ij[s_2 d_2]} \right] - C_{\mu,ij[sd]} \quad \forall\ [sd]$$

In the previous section, we stated that it must hold $P_{N,ij[sd]} \geq 0$. So, we must have

$$\frac{C_{N,ij[sd]}}{\mu_{ij}} - C_{\mu,ij[sd]} \geq 0 \quad \forall\ [sd]$$

i.e. the mean packet delay cost should be greater or equal to the profit from servicing this packet.

Note that for the special case of equal packet cost for the different $[sd]$ processes that use link ij, $C_{N,ij[s_2d_2]} = C_{N,ij} \quad \forall\ [s_2d_2]$, the above solution simplifies to

$$P_{N,ij[sd]} = \frac{C_{N,ij} * (1 + \sum_{[s_1d_1]} N^*_{ij[s_1d_1]})^2}{\mu_{ij}} - C_{\mu,ij[sd]} \quad \forall\ [sd]$$

Substituting the $P_{N,ij[sd]}$ into $P_{V,ij[sd]}$, we have the cost to go from node s to destination d through path $\pi[sd]$:

$$\mathrm{P}_{V,\pi[sd]} = \sum_{ij\in\pi[sd]} \left\{ \frac{C_{V,ij[sd]}}{\delta_{[sd]}} + \frac{r_{[sd]}}{\delta_{[sd]}} * \left[\frac{1 + \sum_{[s_1d_1]} N^*_{ij[s_1d_1]}}{\mu_{ij}} * \left[C_{N,ij[sd]} * (1 + N^*_{ij[sd]}) + \right.\right.\right.$$

$$\left.\left.\left. + \sum_{[s_2d_2]\neq[sd]} C_{N,ij[s_2d_2]} * N^*_{ij[s_2d_2]} \right] - C_{\mu,ij[sd]} \right] \right\} \quad \forall\ \pi[sd] \quad \forall\ [sd]$$

The following Theorems follow immediately:

Theorem 3. *Congestion Control*

For the long-run stationary equilibrium of the virtual circuit routing and congestion control problem, at every source node s, for every destination node d, $[sd]$ virtual circuits are rejected at a node s only if the cost of rejecting them is less than the minimum cost to go from node s to the destination d through any of the paths $\pi[sd]$

$\phi^*_{o[sd]} > 0$ *only if*

$$C_{o[sd]} < \min_{p[sd]} \left\{ \sum_{ij\in p[sd]} \left\{ \frac{C_{V,ij[sd]}}{\delta_{[sd]}} + \frac{r_{[sd]}}{\delta_{[sd]}} * \left[\frac{1 + \sum_{[s_1d_1]} N^*_{ij[s_1d_1]}}{\mu_{ij}} * \right] \right.\right.$$

$$\left.\left. * \left[C_{N,ij[sd]} * (1 + N^*_{ij[sd]}) + \sum_{[s_2d_2]\neq[sd]} C_{N,ij[s_2d_2]} * N^*_{ij[s_2d_2]} \right] - C_{\mu,ij[sd]} \right] \right\} \right\}$$

Theorem 4. *Routing Rule*

For the long-run stationary equilibrium of the virtual circuit routing and congestion control problem, $[sd]$ virtual circuits are routed through path $\pi[sd]$ only if the minimum cost to reach the destination d through path $\pi[sd]$ is the minimum

$\phi^*_{\pi[sd]} > 0$ *only if*

$$\sum_{ij\in\pi[sd]}\left\{\frac{C_{V,ij[sd]}}{\delta_{[sd]}}+\frac{r_{[sd]}}{\delta_{[sd]}}*\left[\frac{1+\sum_{[s_1d_1]}N^*_{ij[s_1d_1]}}{\mu_{ij}}\left[C_{N,ij[sd]}*(1+N^*_{ij[sd]})+\right.\right.\right.$$

$$\left.\left.\left.+\sum_{[s_2d_2]\neq[sd]}C_{N,ij[s_2d_2]}*N^*_{ij[s_2d_2]}\right]-C_{\mu,ij[sd]}\right]\right\}=$$

$$=\min_{p[sd]}\left\{C_{o[sd]},\sum_{ij\in p[sd]}\left\{\frac{C_{V,ij[sd]}}{\delta_{[sd]}}+\frac{r_{[sd]}}{\delta_{[sd]}}*\left[\frac{1+\sum_{[s_1d_1]}N^*_{ij[s_1d_1]}}{\mu_{ij}}*\left[C_{N,ij[sd]}*(1+N^*_{ij[sd]})+\right.\right.\right.\right.$$

$$\left.\left.\left.\left.+\sum_{[s_2d_2]\neq[sd]}C_{N,ij[s_2d_2]}*N^*_{ij[s_2d_2]}\right]-C_{\mu,ij[sd]}\right]\right\}\right\}$$

From the above analysis, we have derived that the length of each link ij for an $[sd]$ virtual circuit is

$$l_{ij[sd]}=\frac{C_{V,ij[sd]}}{\delta_{[sd]}}+\frac{r_{[sd]}}{\delta_{[sd]}}*\left[\frac{1+\sum_{[s_1d_1]}N^*_{ij[s_1d_1]}}{\mu_{ij}}\left[C_{N,ij[sd]}*(1+N^*_{ij[sd]})+\right.\right.$$

$$\left.\left.+\sum_{[s_2d_2]\neq[sd]}C_{N,ij[s_2d_2]}*N^*_{ij[s_2d_2]}\right]-C_{\mu,ij[sd]}\right]$$

The first term of the above length represents the cost $C_{V,ij[sd]}$ for setting up and maintaining an $[sd]$ virtual circuit passing through link ij, times the average virtual circuit duration $1/\delta_{[sd]}$. The second term of the above length represents

the average number of packets $r_{[sd]}/\delta_{[sd]}$ in this $[sd]$ virtual circuit times the cost $C_{N,ij[sd]}$ per packet. Finally, the last term of the above link length represents the profit $C_{\mu,ij[sd]}$ from servicing an $[sd]$ packet on link ij

Let us consider the special case where we have zero $[sd]$ virtual circuit set up and maintenance cost $C_{V,ij[sd]} = 0$, zero profit $C_{\mu,ij[sd]} = 0$ for servicing $[sd]$ packets, and unit delay costs $C_{N,ij[s_2d_2]} = 1$, $\forall\ [s_2d_2]$ on link ij. Then the length of link ij for an $[sd]$ virtual circuit is

$$l_{ij[sd]} = \frac{r_{[sd]}}{\delta_{[sd]}} * \frac{(1 + \sum_{[s_1d_1]} N^*_{ij[s_1d_1]})^2}{\mu_{ij}}$$

That means, that when our only objective is to minimize the average packet delay, then the link length is given by a quadratic function of the average number of packets on this link.

In this section, we have derived state dependent routing and congestion controls for multi-class multi- destination virtual circuit networks. In the next section, we investigate a simple case of this state dependent routing algorithm via simulation.

5.6.5 Simulation

In this section, we investigate a simple case of the derived state dependent virtual circuit routing algorithm via simulation. Simulation is a very effective model for a detailed investigation of the routing algorithm. While the analytical models provide a rigorous mathematical analysis of the system, they cannot afford too much complexity. If we try to include all the parameters that affect the system, then the analytical model become intractable. On the other hand, simulation models are computer programs [289, 288, 149, 428, 316, 351, 408, 314] and therefore we can program as much detail as we like. Their drawback is that they are time consuming. We have to spend a lot of time for writing the code as well as for running them. However, simulation is the only way (besides real implementation) to measure the performance of dynamic routing algorithms.

We have implemented three deterministic source routing algorithms along the minimum length path for single class virtual circuit networks (ties are broken

arbitrarily - though we seldom have ties). The first algorithm uses as link length a special case of that proposed in the previous section. The second algorithm uses as link length the expected packet delay on this link. Finally, the third algorithm is the optimal quasi-static routing algorithm.

1) *Quadratic routing*:

send a new virtual circuit along path π,

if $\sum_{ij} \frac{(1+N_{ij})^2}{\mu_{ij}} * 1_{ij\in\pi} = \min_p \left\{ \sum_{ij} \frac{(1+N_{ij})^2}{\mu_{ij}} * 1_{ij\in p} \right\}$

2) *Shortest queue routing* :

send a new virtual circuit along path π,

if $\sum_{ij} \frac{1+N_{ij}}{\mu_{ij}} * 1_{ij\in\pi} = \min_p \left\{ \sum_{ij} \frac{1+N_{ij}}{\mu_{ij}} * 1_{ij\in p} \right\}$

3) *Optimal quasi-static routing*

For updating the information at the source node about the link lengths in the network, we considered three factors:

1) *what estimate* of the number of packets N_{ij} at each link ij is sent to the source node from each node i.

2) *how often* this estimate is sent to the source node by each node i. It is well known that the updating period should be smaller than the average virtual circuit duration [177, 518].

3) *after how much delay* this information arrives back to the source node. We assume that no extra traffic is created from each node to the source node, but that this information is either piggybacked on other packets or it is transferred through a different channel.

First, we consider a single source-destination network with 2 paths from source to destination (Figure 5.9) that have the same capacity but the order of their links is different.

Path # has 7 links with transmission rates 5, 4, 3, 3, 2, 1 and 1. Path #2 has 7 links with transmission rates 1, 1, 2, 3, 3, 4 and 5.

The mean packet service time is $\frac{1}{\mu} = 1$ and therefore $\mu_{ij} = \mu * C_{ij} = C_{ij}$. The mean virtual circuit duration is $\frac{1}{\delta} = 1000$. The total packet arrival rate is

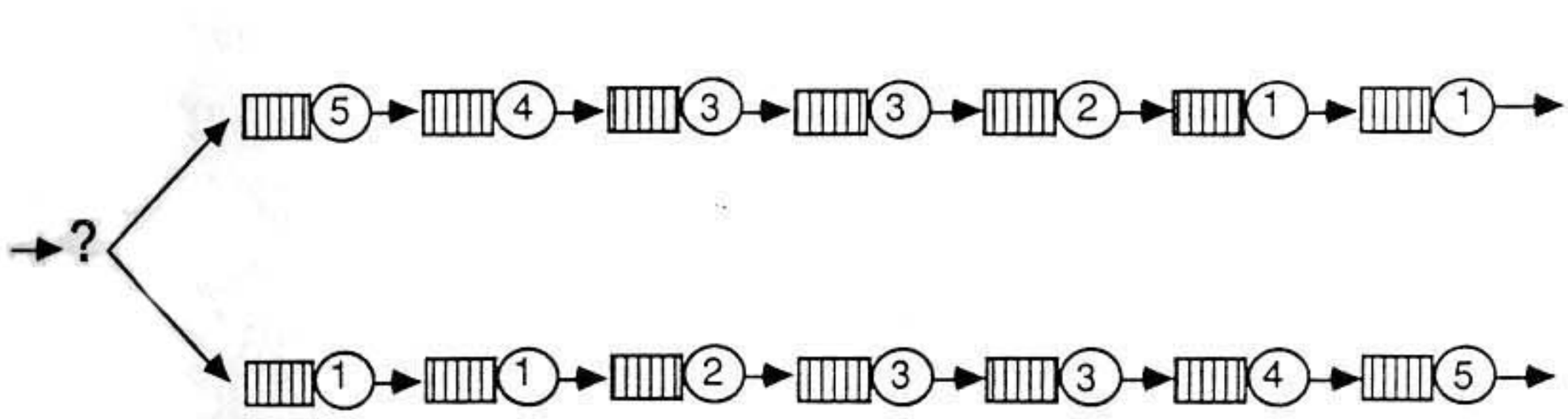


Figure 5.9: Simulated network.

$r * \frac{\gamma}{\delta} = \frac{1000}{700}$, however we considered 5 cases that achieve this total packet arrival rate:

γ	r	$\frac{\gamma}{\delta}$	$\frac{r}{\delta}$
$\frac{1}{7}$	$\frac{1}{100}$	$\frac{1000}{7}$	10
$\frac{1}{14}$	$\frac{1}{50}$	$\frac{1000}{14}$	20
$\frac{1}{26}$	$\frac{1}{27}$	$\frac{1000}{26}$	$\frac{1000}{27}$
$\frac{1}{50}$	$\frac{1}{14}$	20	$\frac{1000}{14}$
$\frac{1}{100}$	$\frac{1}{7}$	10	$\frac{1000}{7}$

where γ is the arrival rate of virtual circuits, r is the packet arrival rate per virtual circuit, $\frac{\gamma}{\delta}$ is the average number of virtual circuits into the network and $\frac{r}{\delta}$ is the average number of packets per virtual circuit.

The information at the source node about the link lengths in the network is updated according to two schemes:

a) *instantaneous* information, when at every instant, the source node knows and uses the current number of packets at every link and

b) *obsolete* information, when the information about the average number of packets at every link during a time interval of 100 time units is sent to the source node at the end of this time interval and it is used by the source node after 50 time units delay.

Figures 5.10, 5.11, 5.12, 5.13, 5.14 and Table 5.1 describe the simulation results of routing 100,000 virtual circuits into the network of Figure 5.9. In this network, the two paths have similar links but in different positions. Both paths receive on

the average the same number of the virtual circuits and have the same average packet delay.

Although all the above five cases have the same total packet arrival rate, the average packet delay is different in each case with an extremely large average packet delay in case the last case ($\gamma = 1/100 \;\; r = 1/7$), where each virtual circuit carries a large number of packets. This means that routing algorithms that consider only the packet arrival rate will achieve poor performance.

The more often that we update the link length information at the source node, the smaller average packet delay is achieved. The smaller the delay that the link length information becomes available to the source node, the smaller average packet delay is achieved. When the network state information is obsolete, the *Quadratic routing* seems to be slightly better than the *Shortest queue routing*, otherwise they achieve the same average packet delay.

The *Optimal static routing* assigns in a Round-Robin basis an odd numbered virtual circuit to path #1 and an even numbered virtual circuit to path #2.

When the updating period is not much larger than the mean interarrival time of virtual circuits, then both dynamic routing algorithms, *Quadratic routing* and *Shortest queue routing*, are clearly better than the *Optimal static routing*. However, when the updating period is extremely large compared to the mean interarrival time of virtual circuits, then the dynamic routing algorithms make many wrong decisions and therefore give larger average packet delay.

The *Shortest queue routing* is an approximation of the *Quadratic routing* and therefore they achieve similar average packet delay. Note also, that for single-link paths with equal link transmission speeds, both algorithms choose the same path. To see this, consider two single-link paths π and p, with link transmission speeds μ, N_π packets at path π link and N_p packets at path p link, such that

$$\frac{(1+N_\pi)^2}{\mu} < \frac{(1+N_p)^2}{\mu} \Leftrightarrow \frac{1+2*N_\pi+N_\pi^2}{\mu} < \frac{1+2*N_p+N_p^2}{\mu} \Leftrightarrow$$

$$\Leftrightarrow \frac{(N_\pi - N_p)*(N_\pi+N_\pi+N_p+2)}{\mu} < 0 \Leftrightarrow \frac{N_\pi - N_p}{\mu} < 0 \Leftrightarrow$$

γ=1/7 r=1/100	instantaneous	obsolete
quadratic	19.02 $\pm$ 0.80	29.69 $\pm$ 1.06
shortest queue	18.77 $\pm$ 0.63	31.64 $\pm$ 1.27
optimal quasi-static	22.98 $\pm$ 1.83	

γ=1/14 r=1/50	instantaneous	obsolete
quadratic	14.19 $\pm$ 0.19	20.65 $\pm$ 0.85
shortest queue	13.97 $\pm$ 0.48	20.39 $\pm$ 0.78
optimal quasi-static	17.98 $\pm$ 0.62	

γ=1/26 r=1/27	instantaneous	obsolete
quadratic	15.24 $\pm$ 0.56	21.99 $\pm$ 0.63
shortest queue	15.43 $\pm$ 0.28	21.75 $\pm$ 0.58
optimal quasi-static	20.41 $\pm$ 0.57	

γ=1/50 r=1/14	instantaneous	obsolete
quadratic	24.47 $\pm$ 0.99	34.88 $\pm$ 1.47
shortest queue	23.38 $\pm$ 0.85	34.65 $\pm$ 1.17
optimal quasi-static	39.69 $\pm$ 1.47	

γ=1/100 r=1/7	instantaneous	obsolete
quadratic	53.88 $\pm$ 2.64	71.35 $\pm$ 0.82
shortest queue	53.72 $\pm$ 3.67	72.94 $\pm$ 2.26
optimal quasi-static	99.36 $\pm$ 5.89	

Table 5.1: The average packet delay $\pm$ error (95% confidence interval) for the network of Figure 5.9 with $\gamma = 1/7$, $r = 1/100$, for the *Quadratic* routing with instantaneous and obsolete information, the *Shortest queue* routing with instantaneous and obsolete information and the *Optimal quasi-static* routing implemented as Round-Robin.

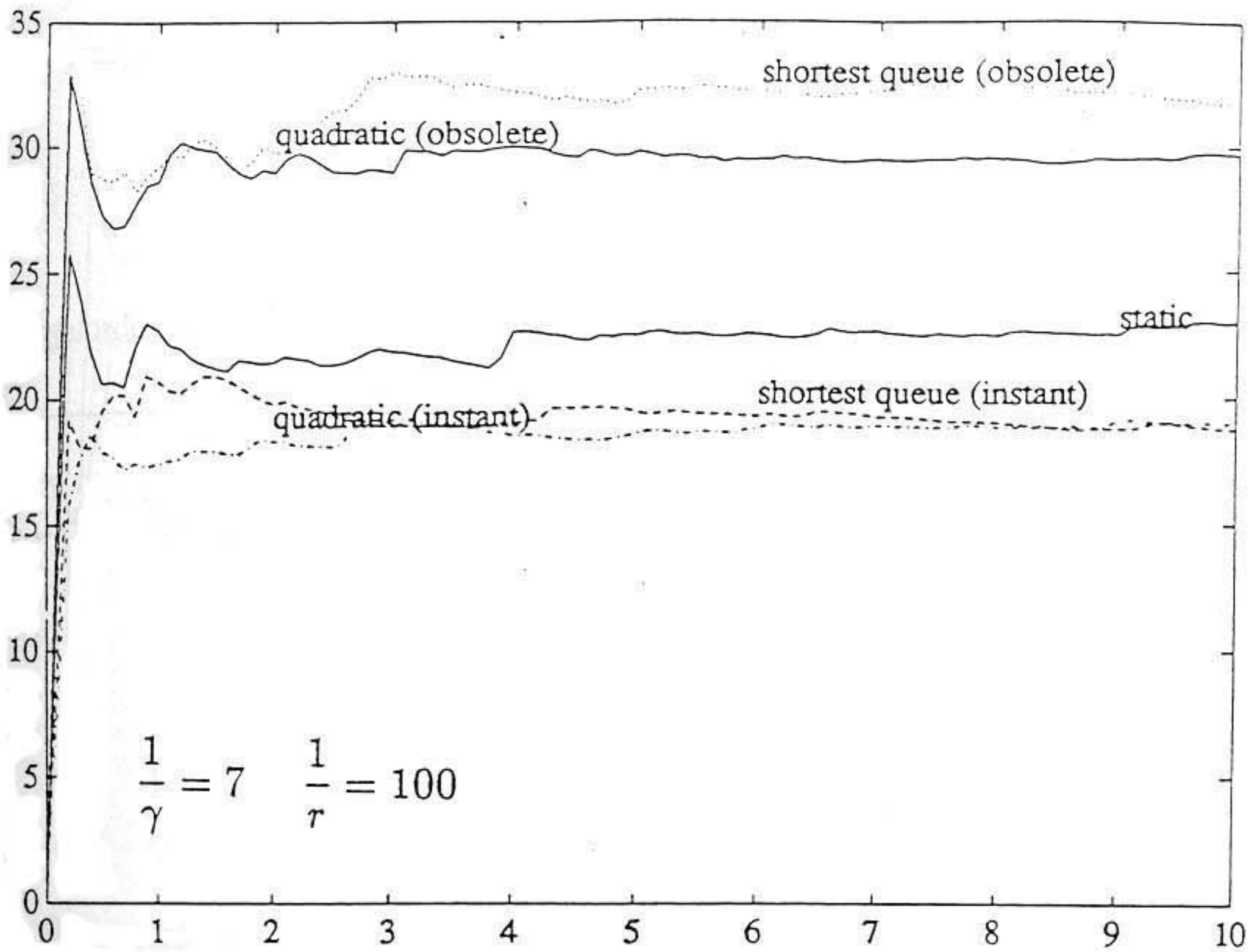


Figure 5.10: The average packet delay ± error (95% confidence interval) for the network of Figure 5.9 with $\gamma = 1/7$, $r = 1/100$, for the *Quadratic* routing with instantaneous and obsolete information, the *Shortest queue* routing with instantaneous and obsolete information and the *Optimal quasi-static* routing implemented as Round-Robin.

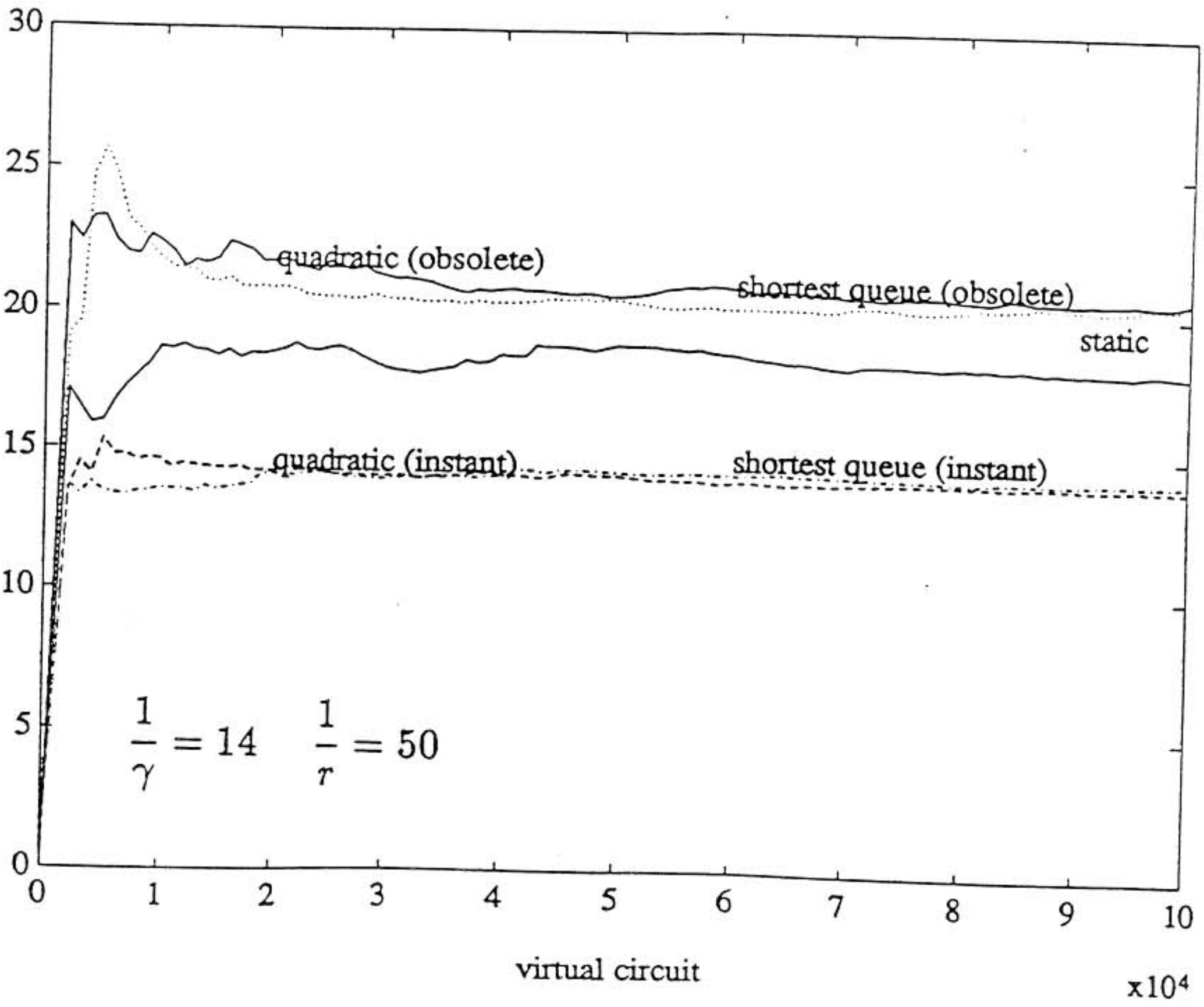


Figure 5.11: The average packet delay ± error (95% confidence interval) for the network of Figure 5.9 with $\gamma = 1/14$, $r = 1/50$, for the *Quadratic* routing with instantaneous and obsolete information, the *Shortest queue* routing with instantaneous and obsolete information and the *Optimal quasi-static* routing implemented as Round-Robin.

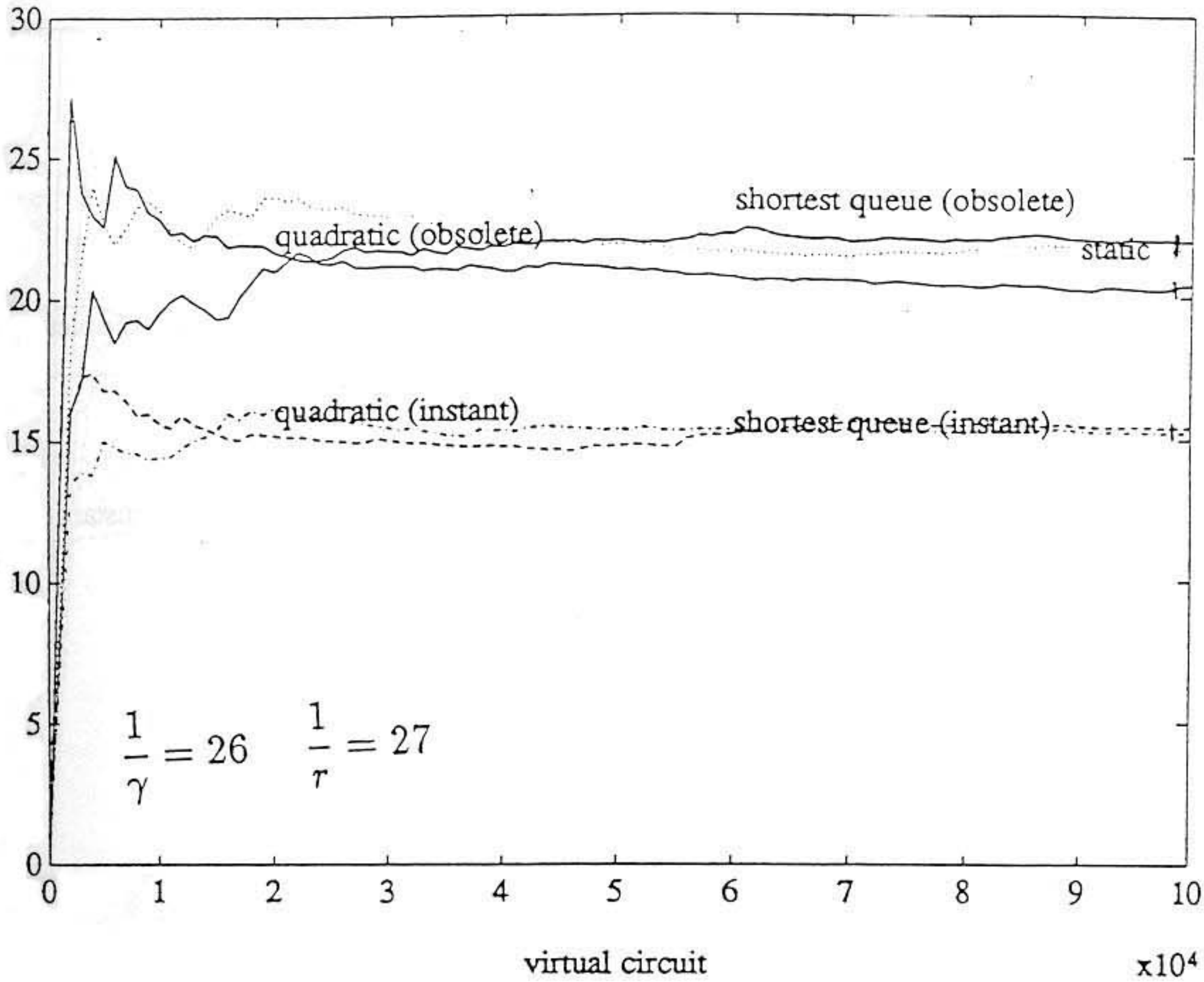


Figure 5.12: The average packet delay ± error (95% confidence interval) for the network of Figure 5.9 with $\gamma = 1/26$, $r = 1/27$, for the *Quadratic* routing with instantaneous and obsolete information, the *Shortest queue* routing with instantaneous and obsolete information and the *Optimal quasi-static* routing implemented as Round-Robin.

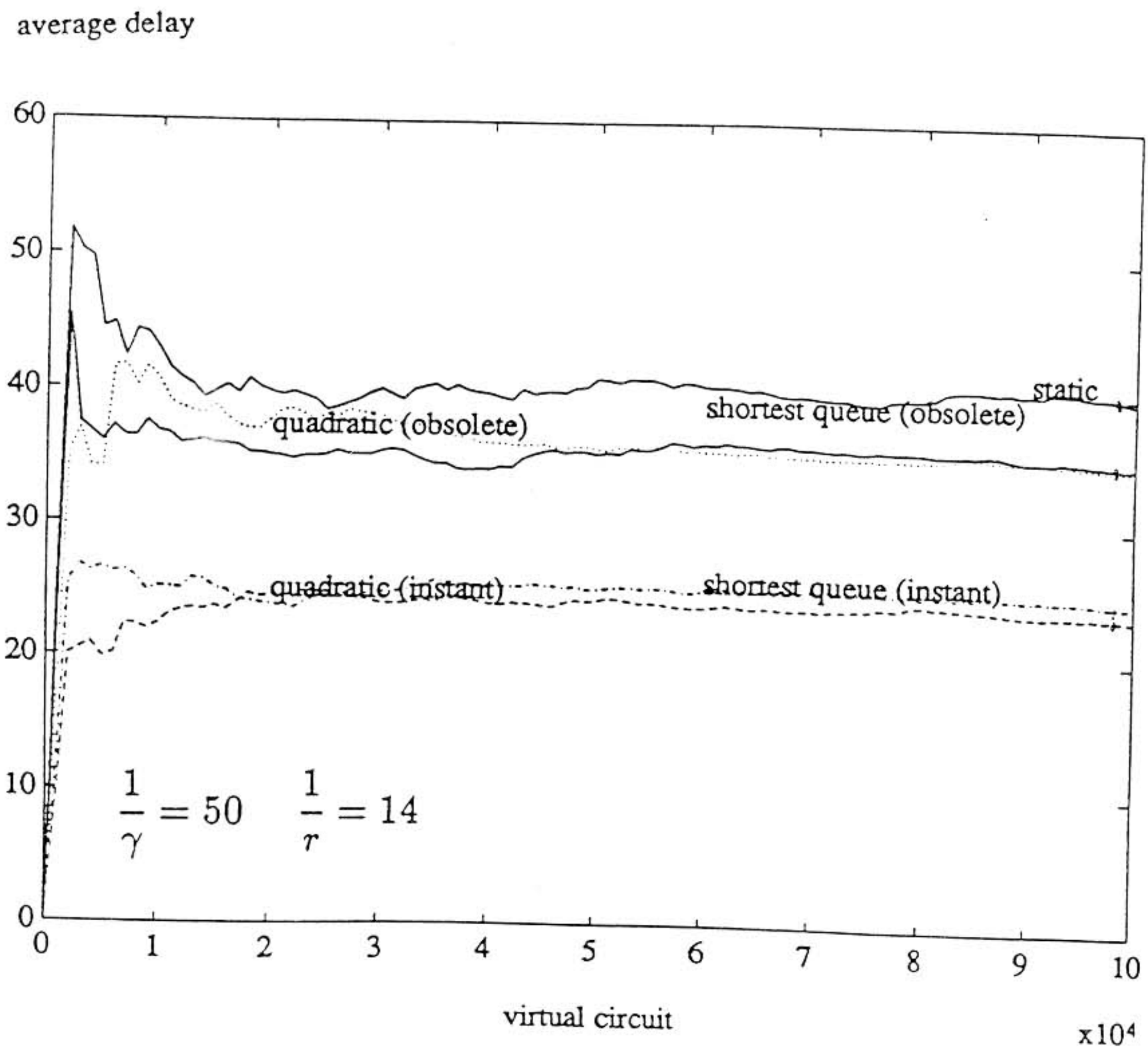


Figure 5.13: The average packet delay ± error (95% confidence interval) for the network of Figure 5.9 with $\gamma = 1/50$, $r = 1/14$, for the *Quadratic* routing with instantaneous and obsolete information, the *Shortest queue* routing with instantaneous and obsolete information and the *Optimal quasi-static* routing implemented as Round-Robin.

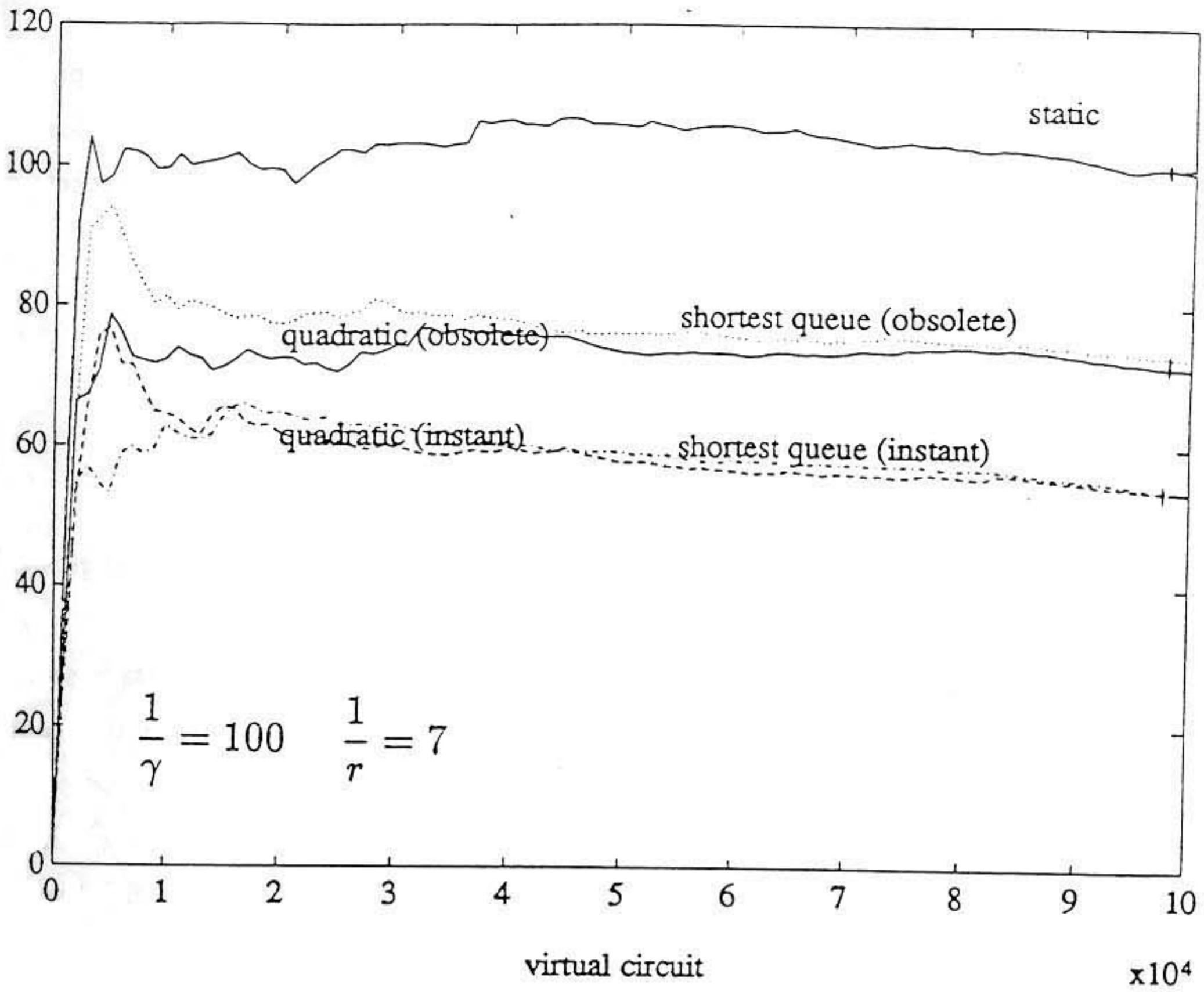


Figure 5.14: The average packet delay ± error (95% confidence interval) for the network of Figure 5.9 with $\gamma = 1/100$, $r = 1/7$, for the *Quadratic* routing with instantaneous and obsolete information, the *Shortest queue* routing with instantaneous and obsolete information and the *Optimal quasi-static* routing implemented as Round-Robin.

$$\Rightarrow \frac{N_\pi}{\mu} < \frac{N_p}{\mu} \Rightarrow \frac{1+N_\pi}{\mu} < \frac{1+N_p}{\mu}$$

That means that both algorithms choose path π since the ordering of the link lengths is the same for both algorithms.

In order that the *Quadratic routing* achieves different average packet delay than the *Shortest queue routing*, they should choose different paths for the same network state. Consider two paths π and p with the number of packets on their links satisfying the following relations simultaneously

$$\sum_{ij} \frac{(1+N_{ij})^2}{\mu_{ij}} * 1_{ij\in\pi} < \sum_{xy} \frac{(1+N_{xy})^2}{\mu_{xy}} * 1_{xy\in p}$$

$$\sum_{xy} \frac{1+N_{xy}}{\mu_{xy}} * 1_{xy\in p} < \sum_{ij} \frac{1+N_{ij}}{\mu_{ij}} * 1_{ij\in\pi}$$

then the *Quadratic routing* will choose path π, while the *Shortest queue routing* will choose path p.

Next, we further investigate the two dynamic algorithms for a more complex network with unbalanced paths. We consider a network with 5 paths from source to destination (Figure 5.15).

Path #1 has 3 links with transmission speeds 2, 1 and 3. Path #2 has 5 links with transmission speeds 4, 2, 0.5, 3 and 1. Path #3 has 7 links with transmission speeds 5, 1, 2, 3, 1, 4 and 2. Path #1 has 6 links with transmission speeds 1, 1, 1, 1, 1 and 1. Path #1 has 4 links with transmission speeds 2, 2, 2 and 2.

The mean packet service time is $\frac{1}{\mu} = 1$ and therefore $\mu_{ij} = \mu * C_{ij} = C_{ij}$. The mean virtual circuit duration is $\frac{1}{\delta} = 1000$. We consider two cases for the total packet arrival rate.

In case #1 The arrival rate of virtual circuits is $\gamma = \frac{1}{5}$ and the packet arrival rate per virtual circuit is $r = \frac{1}{50}$. Then the average number of virtual circuits into the network is $\frac{\gamma}{\delta} = 200$ and the average number of packets per virtual circuit is $\frac{r}{\delta} = 20$.

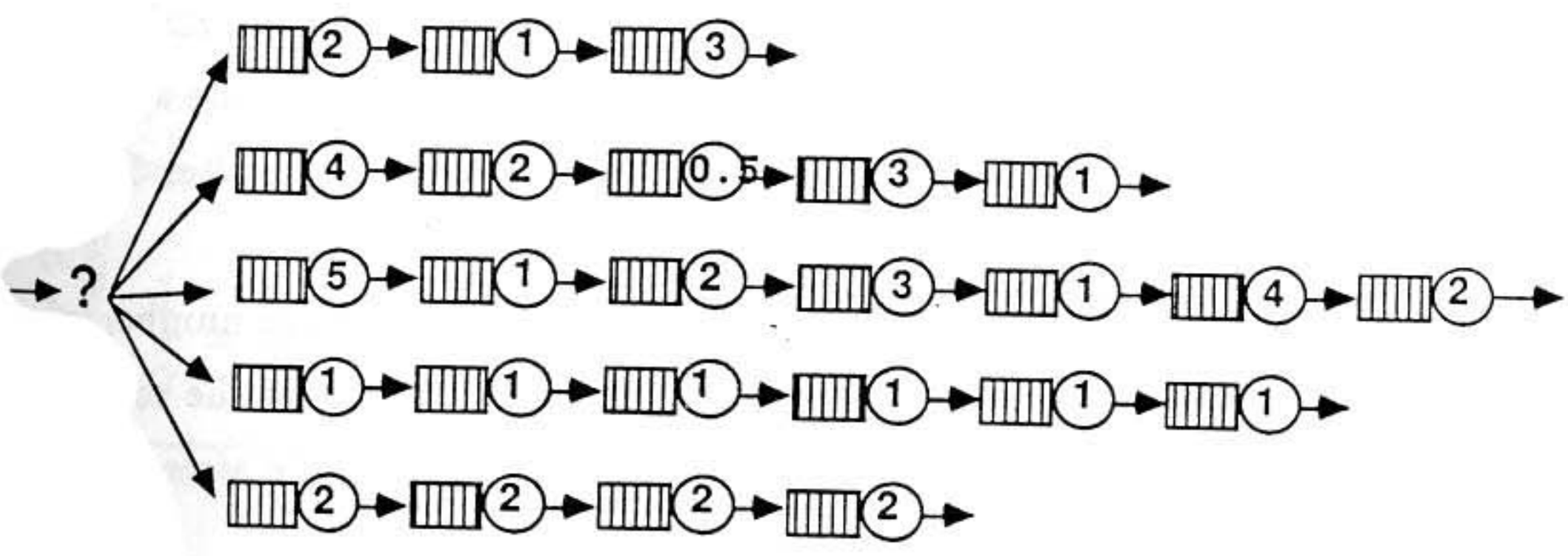


Figure 5.15: Simulated network.

In case #1 The arrival rate of virtual circuits is $\gamma = \frac{1}{50}$ and the packet arrival rate per virtual circuit is $r = \frac{1}{5}$. Then the average number of virtual circuits into the network is $\frac{\gamma}{\delta} = 20$ and the average number of packets per virtual circuit is $\frac{r}{\delta} = 200$.

The information at the source node about the link lengths in the network is updated according to four schemes:

a) *1 time unit*, when at every instant, the source node knows and uses the current number of packets at every link.

b) *20 time units*, when the information about the average number of packets at every link during a time interval of 20 time units is sent to the source node at the end of this time interval and it is used by the source node after 20 time units delay.

c) *50 time units*, when the information about the average number of packets at every link during a time interval of 50 time units is sent to the source node at the end of this time interval and it is used by the source node after 50 time units delay.

d) *100 time units*, when the information about the average number of packets at every link during a time interval of 100 time units is sent to the source node at the end of this time interval and it is used by the source node after 50 time units delay.

Figures 5.16, 5.17 and Table 5.2 describe the simulation results of routing 100,000 virtual circuits into the network of Figure 5.11. In this network, the paths are capacity inequivalent and they also have different number of links. Every path receives different number of virtual circuits and has different average packet delay. Similarly as in the previous network, the more often that we update the link length information at the source node, the smaller average packet delay is achieved. The smaller the delay that the link length information becomes available to the source node, the smaller average packet delay is achieved. However, the *Quadratic routing* achieves clearly smaller average packet delay than the *Shortest queue routing*, especially when the network state information becomes obsolete.

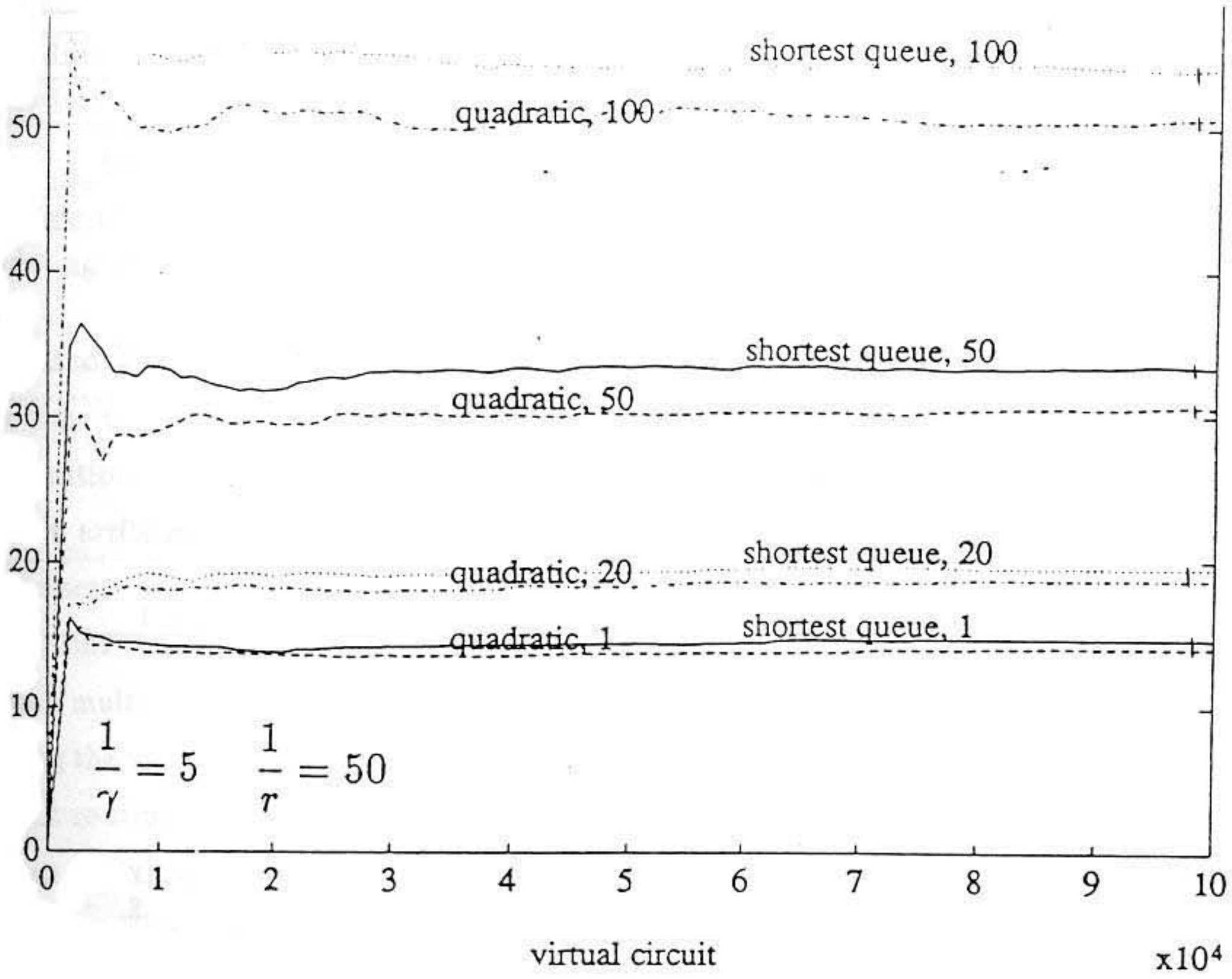


Figure 5.16: The average packet delay $\pm$ error (95% confidence interval) for the network of Figure 5.11 for $\gamma = 1/5$ $r = 1/50$, for the *Quadratic* and the *Shortest queue* routing with updating every 1, 20, 50, 100 time units.

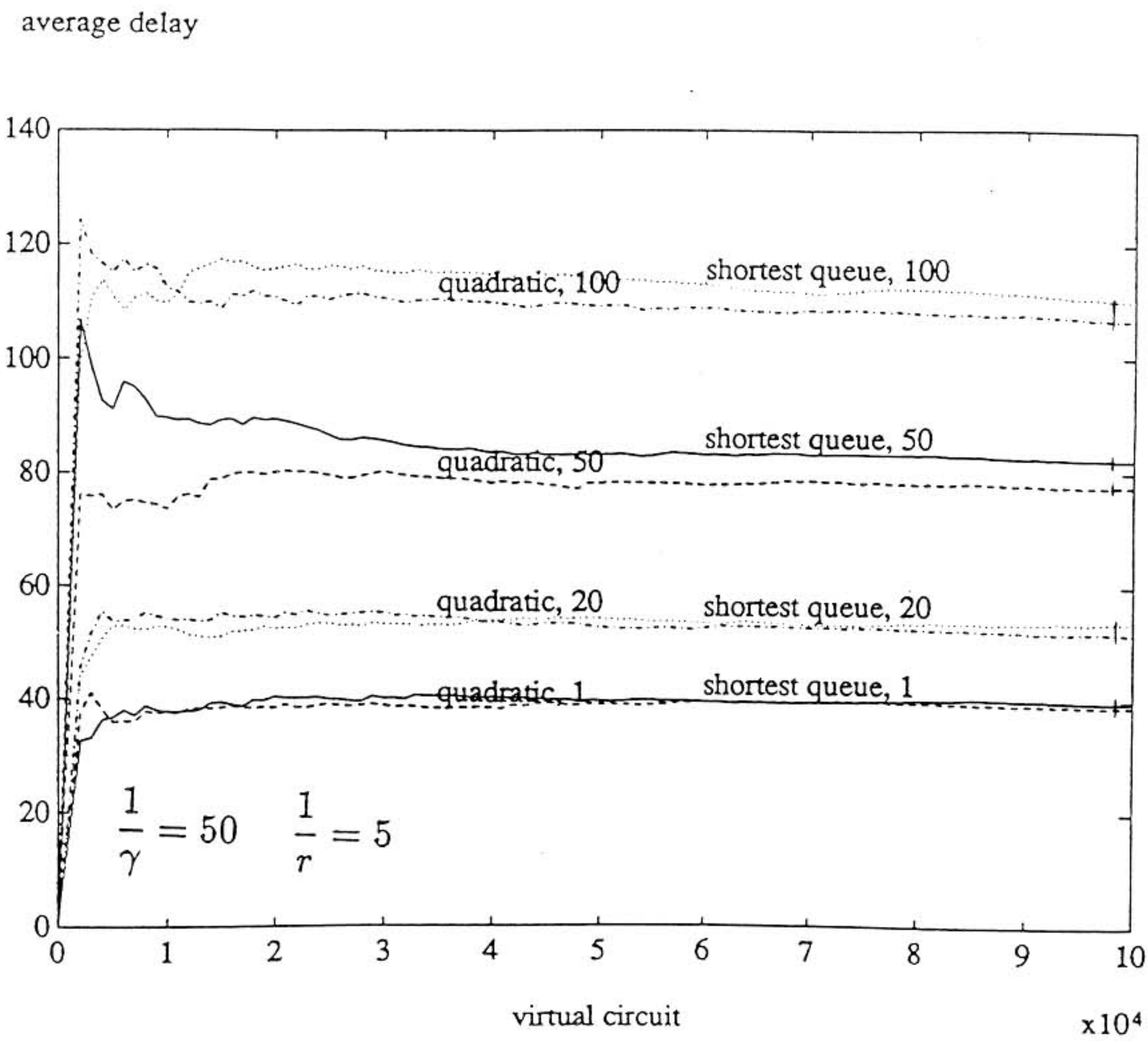


Figure 5.17: The average packet delay ± error (95% confidence interval) for the network of Figure 5.11 for $\gamma = 1/50$ $r = 1/5$, for the *Quadratic* and the *Shortest queue* routing with updating every 1, 20, 50, 100 time units.

γ=1/5 r=1/50	1 time	20 time	50 time	100 time
quadratic	14.06 $\pm$ 0.27	18.74 $\pm$ 0.30	30.55 $\pm$ 0.54	50.70 $\pm$ 0.87
shortest queue	14.65 $\pm$ 0.25	19.51 $\pm$ 0.30	33.38 $\pm$ 0.42	54.13 $\pm$ 1.32

γ=1/50 r=1/5	1 time	20 time	50 time	100 time
quadratic	38.98 $\pm$ 1.70	51.70 $\pm$ 1.84	77.53 $\pm$ 1.30	106.89 $\pm$ 1.61
shortest queue	39.59 $\pm$ 1.10	53.74 $\pm$ 0.81	82.21 $\pm$ 2.53	110.02 $\pm$ 2.62

Table 5.2: The average packet delay $\pm$ error (95% confidence interval) for the network of Figure 5.11 for the *Quadratic* and the *Shortest queue* routing with updating every 1, 20, 50, 100 time units.

Although for the above two cases, the total packet arrival rate is 4 packets per time unit, they give different average delay. This again confirm our previous observation that for virtual circuit networks is not enough to consider the aggregate packet arrival rate, but both the virtual circuit and packet per virtual circuit processes.

In this section, we present nonlinear dynamic queueing models of multi-destination multi-class virtual circuit networks, by explicitly considering the interaction among the virtual circuit and packet processes. We formulate the integrated virtual circuit routing and congestion control problem as an optimal control problem. We set up a multi-objective function and we solve it using the Pontryagin maximum principle. Then we derive state dependent routing and congestion control policies for virtual circuit network control and we define as link length a quadratic function of the average number of packets on it. Finally, we demonstrate via simulation, that for an unbalanced network, this Quadratic routing achieves smaller average packet delay than a Shortest queue routing. For a balanced network, both the Quadratic routing and the Shortest queue routing achieve similar average packet delay, that is also smaller than that achieved by the Optimal quasi-static routing, when the updating period is not extremely larger than the mean interarrival time of virtual circuits.

5.7 Application to Integrated Services Networks

In this section, we apply the methodologies developed in the previous sections to integrated services networks. In section 6, for each class, we model the traffic processes in two interacting levels: i) the virtual circuit process level and ii) the packet process level. For integrated services networks, we propose using more than two interacting levels. For example, four levels: i) subscriber level, ii) virtual circuit level, iii) burst level, iv) packet level.

Different dynamic queueing models (such as those of section 5.4 for datagram networks) will be used at each level to model the dynamic evolution of the corresponding processes.

Furthermore, we can also introduce other dynamic models based on finite population queueing models. For example, at the subscriber level, let $A(t)$ be the active number of subscribers among the existing $S(t)$, $a(t)$ the rate at which an idle subscriber becomes active and $b(t)$ the rate at which an active subscriber becomes idle. Then a dynamic queueing model that describes the average number of active subscribers is the following:

$$A(t) = a(t) * (S(t) - A(t)) - b(t) * A(t)$$

Now, each active subscriber creates virtual circuits, and each virtual circuit creates bursts, and each burst creates packets as in section 5.6. Therefore the state of each system resource is described by three variables: the number of active subscribers $A(t)$, the number of virtual circuits $V(t)$, the number of bursts $B(t)$ and the number of packets $N(t)$ at this resource. So, for each class c, the state of a resource is

$$\mathbf{X}^c(t) = [A^c(t), V^c(t), B^c(t), N^c(t)]$$

Similarly, for the cost functions, we add to the costs of section 5.6 another level of costs for the active number of subscribers, bursts etc.

Chapter 6

Stochastic Learning Automata for Decentralized Load Sharing, Routing & Congestion Control

In this chapter, we introduce another novel methodology for decentralized dynamic load sharing, routing and congestion control. We propose stochastic learning automata at the source nodes of the system for admitting or rejecting jobs, for selecting the destination node for job processing, and for selecting the routing path to the destination node. These decisions will be done probabilistically by learning automata algorithms that will update their action probabilities according to measurements of the path and source-destination lengths. The path and source-destination lengths are those derived in the dynamic optimality conditions of chapter 4. We also introduce novel classes of stochastic learning automata:
i) state dependent learning automata, whose adaptation rates are functions of the system state, ii) two-step learning automata, that use larger adaptation rates when the selected action repeatedly gives good performance, iii) multiple response automata, that use different adaptation rates for different system learning response (not just the favorable/unfavorable response of previous learning automata). We prove that these learning automata are feasible at each step, non-absorbing, strictly distance diminishing, ergodic and expedient. We apply this methodology to datagram, virtual circuits and integrated services networks. We give an example, where we make virtual circuit routing decisions learning automata algorithms. We show (via simulation) that by suitable tuning the adaptation rates of the algorithms, the learning automata achieve smaller average packet delay. We also show that

a path length proposed in chapter 5, is superior to a shortest-queue-type routing, usually used in real networks.

6.1 Introduction

Learning is defined as any relatively permanent change in behavior resulting from past experience, and a learning system is characterized by its ability to improve its behavior with time, in some sense tending towards an ultimate goal. In mathematical psychology, learning systems [80, 15, 494, 259] have been developed to explain behavior patterns among living organisms. These mathematical models in turn have lately been adapted to synthesize engineering systems [344].

Tsetlin [495] initially introduced the concept of learning automaton operating in an unknown random environment. He considered learning behaviors of finite deterministic automata under the stationary random environment. Varshavskii & Vorontsova [503] introduced variable structure stochastic automata in an unknown random environment. Chandrasekaran & Shen [92] Poznyak [386], Tsypkin & Poznyak [498], Flerov [163], Polyak [383, 384] Lakshmivarahan & Thathachar [283] and others further advanced the learning automaton theory.

A number of books on learning automaton theory have been also appeared. Norman [355, 356, 357] develops a Markov process-based approach to analyze the learning automaton and explain the learning processes in organisms. Lakshmivarahan [282] provides a rigorous analysis of the learning automaton theory. El-Fattah [143] presents learning automata used for pattern recognition systems and for simulation of collective behavior problems. Glorioso & Osorio [197] describe fundamental issues of learning and applications in engineering. Baba [19] presents learning automaton behavior under unknown multi-teacher environments. Narendra & Thathachar [342] provide a rigorous introduction to the theory of learning automata.

A number of papers have been also appeared recently [13, 281, 482, 367, 364, 363, 365, 451] that propose new reinforcement schemes for learning automata and investigate their properties.

Learning automata have been also applied to pattern recognition problems [22], to routing problems [346, 343, 456], to flow control problems [321, 322], to partitioning problems [366], to neural network models [24, 23, 473] etc.

In the previous two chapters, we found the conditions for team optimality, Nash and Stackelberg equilibrium for the joint load sharing, routing and congestion control problem. In this chapter, we introduce stochastic learning automata as decentralized decision-makers that will achieve these conditions. A stochastic learning automaton is an adaptive control algorithm that reacts to the system's response. It chooses an action and if the system's response is favorable, then it reinforces that action, otherwise it tends to choose another action.

The greatest potential of the learning automata methodology is that it permits the analysis of very complex dynamic systems, and global optimization is possible. Even when little information is available, they tend to stabilize a nonstationary system by predicting its behavior.

We propose using learning automata at the source nodes of the system for admitting or rejecting a job from the system (congestion control), for selecting the destination computer site for processing the job (load sharing) and selecting the path through which the job will reach this destination node (routing).

In previous chapters, we found what the optimal load sharing, routing and congestion control policies should be. These optimal control policies may be implemented directly as they were found. However, the underlying assumptions of the models (e.g. independent exponential distributions), or even other management problems that were not explicitly considered, may affect these optimal control policies. Therefore, we propose the use of learning automata that will drive to these optimal control policies. So, instead of deterministically choosing the minumum length path, learning automata will choose it with very high probability. Note, that if we appropriatelly calibrate the step size of these learning automata algorithms, then they may choose the minimum length path with probability 1.

6.2 Learning Automaton Theory

In this section, we review the basic learning automaton theory [344, 342].

A learning automaton is a feedback system (Figure 6.1) connecting a stochastic automaton $(X, \phi, a, \mathbf{P}, T, G)$ and an environment $C = \{C_1, \ldots, C_{|a|}\}$, where

X: input set or environment response.

1) if $X \in \{0, 1\}$, i.e. the environment response takes only two values, where $X = 0$ can be considered as reward and $X = 1$ as penalty, we have a P-model

2) if $X \in \{X^1, \ldots, X^k\}$, $X^i \in [0, 1]$, i.e. the environment response takes a certain number of values in the interval [0,1], with $X = 0$ to be "full reward" and $X = 1$ to be "full penalty", we have a Q-model.

3) if $X \in [0, 1]$, i.e. the environment response takes any value in the interval [0,1], with $X = 0$ to be "full reward" and $X = 1$ to be "full penalty", we have an S-model.

$\phi = \{\phi_1, \ldots, \phi_s\}$, $s < \infty$: set of internal states.

$a = \{a_1, \ldots, a_{|a|}\}$, $|a| \leq s$: output or action set.

$\mathbf{P}(n) = [P_1(n), \ldots, P_{|a|}(n)]^T$ state probability vector, where $P_i(n) = P[a(n) = a_i]$.

T: algorithm, updating or reinforcement scheme, that generates the action probability $\mathbf{P}(n+1) = T[\mathbf{P}(n), a(n), X(n)]$.

G: $\phi \rightarrow a$: output function.

$c_i(n) = E[X(n)/a(n) = a_i]$: expected penalty for action a_i, which are unknown and there is a unique minimum.

In order to evaluate a learning automaton, some measures are defined [344]:

1) **Expedience** : if at a certain time instant n, the automaton selects action a_i with probability $P_i(n)$, then the average penalty received by the automaton conditioned on $\mathbf{P}(n)$ is $M(n) = E[X(n)/\mathbf{P}(n)]$. If no a priori information is available and the actions are chosen at random then $M_0 = (C_1 + \ldots + C_{|a|})/|a|$. A learning automaton is called expedient if $\lim_{n \rightarrow \infty} E[M(n)] < M_0$ and of course it performs better than one that selects its actions randomly.

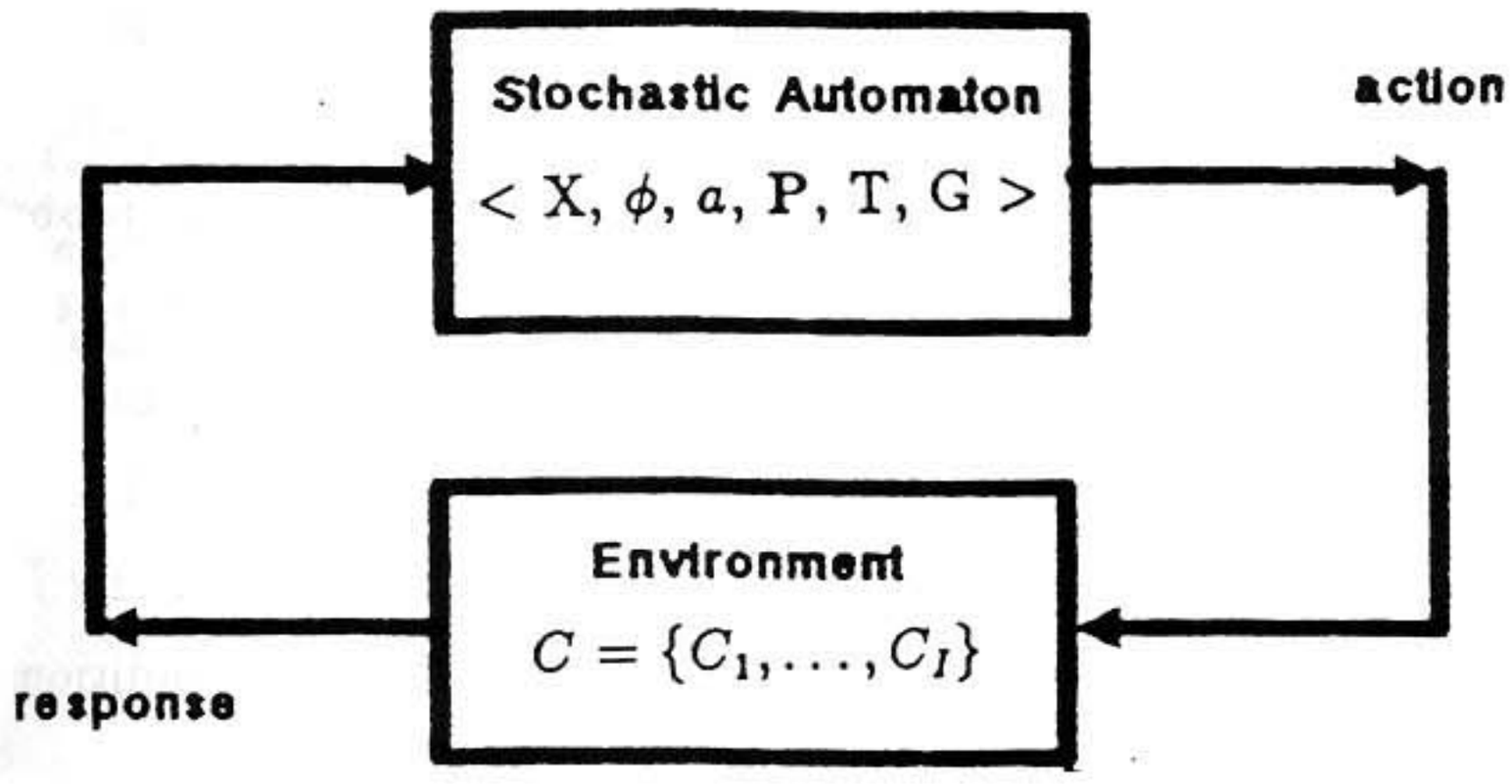


Figure 6.1: Learning Automaton.

2) **Optimality** : a learning automaton is called optimal if

$$\lim_{n\to\infty} E[M(n)] = c_l, \quad \text{where} \quad c_l = min\{c_1, \ldots, c_{|a|}\}$$

or if

$$\lim_{n\to\infty} E[P_l(n)] = 1, \quad \text{where} \quad P_l(n) = P[a(n) = a_l]$$

i.e. optimality means that asymptotically the action associated with the minimum expected penalty is chosen with probability one.

A learning automaton is called ϵ-optimal if

$$\lim_{n\to\infty} E[M(n)] < c_l + \epsilon, \ \epsilon > 0$$

Next, the operation of a learning automaton is described: The automaton selects action $a(n) = a_i$ with probability $P_i(n)$ at each instant n. Action $a(n)$ becomes input to the environment (Figure 6.1). If this results in a favorable outcome for the network performance ($X(n) \to 0$), then the probability $P_i(n)$ is increased by $\Delta P_i(n) = P_i(n+1) - P_i(n)$ and the $P_j(n), j \neq i$, are decreased by $\Delta P_j(n) = P_j(n+1) - P_j(n)$. Otherwise, if an unfavorable outcome ($X(n) \to 1$) appears, then the $P_i(n)$ is decreased by $\Delta P_i(n) = P_i(n+1) - P_i(n)$ and the $P_j(n), j \neq i$ are increased by $\Delta P_j(n) = P_j(n+1) - P_j(n)$. By iteration of the algorithm, we achieve adaptation to varying environment conditions.

Let $a(n) = a_i$ and

if $X(n) \to 0$ then

$$P_i(n+1) = P_i(n) + \sum_{j\neq i} f_j[\mathbf{P}(n)]$$

$$P_j(n+1) = P_j(n) - f_j[\mathbf{P}(n)] \ , \forall j \neq i$$

else $X(n) \to 1$ then

$$P_i(n+1) = P_i(n) - \sum_{j\neq i} g_j[\mathbf{P}(n)]$$

$$P_j(n+1) = P_j(n) + g_j[\mathbf{P}(n)] \ , \forall j \neq i$$

where f_j and g_j are nonnegative continuous functions and $0 < P_i(n) < 1$, $\sum_{i=1}^{|a|} P_i(n) = 1$. f_j and g_j can be linear or nonlinear functions of $\mathbf{P}$ (n). A class of linear algorithms is :

$$f_j(\mathbf{P}(n)) = \alpha * P_j(n), \quad 0 < \alpha < 1, \ \alpha : \text{reward parameter}$$

$$g_j(\mathbf{P}(n)) = \beta * \left[\frac{1}{|a| - 1} - P_j(n) \right], \quad 0 \leq \beta < 1, \quad \beta : \text{penalty parameter}$$

Three linear schemes exhibit interesting behavior [344]. In the L_{R-I} (Linear Reward-Inaction) algorithm ($\beta = 0$), every sample path converges to selecting only one action with probability one and it is ϵ-optimal. For the L_{R-P} (Linear Reward Penalty) algorithm ($\beta = \alpha$), and the $L_{R-\epsilon P}$ (Linear Reward Infinitesimal Penalty) algorithm ($\beta << \alpha$), $\mathbf{P}(n)$ converges in distribution to a random vector $\mathbf{P}$, whose distribution is independent of $\mathbf{P}(0)$. Further the $L_{R-\epsilon P}$ is ϵ-optimal and not be locked in on a nonoptimal action.

For the $L_{R-\epsilon P}$ algorithm [343, 456] there is a unique $\mathbf{P}^*$ such that $C_i(\mathbf{P}^*) = C_j(\mathbf{P}^*) \ \forall \ i, j$, i.e. in the limit the expected penalties of the actions are equalized and the action corresponding to the lowest expected penalty is chosen with probability close to 1.

6.3 State Dependent Learning Automata

In this section, we extend the learning automata theory by making the updating scheme a function of the environment state.

At time n, action i is selected according to an action probability $P_i(n) \geq 0$, $\sum_{i=1}^{A} P_i(n) = 1$. We define as response $X_i(n)$ of the environment a continuous, monotonous, non-decreasing function of the cost $C_i(n)$ of the selected action i normalized to the [0,1] interval, i.e.

$$X_i(n) = \varphi(C_i(n)) \quad 0 \leq X(n) \leq 1$$

In this way, we correspond $X_i(n) \rightarrow 0$ to a favorable outcome (small $C_i(n)$), and $X_i(n) \rightarrow 1$ to an unfavorable outcome (large $C_i(n)$). Examples of such functions are :

$$X_i(n) = \frac{C_i(n)}{C_{max}(n)}$$

$$X_i(n) = \frac{C_i(n) - C_{min}(n)}{C_{max}(n) - C_{min}(n)}$$

$$X_i(n) = e^{\alpha * (C_i(n) - C_{max}(n))}$$

The choice of the suitable function depends on our desire to stress some response areas or to have uniform adaptation speed to the response.

If the environment's response $X_i(n)$ to the selected action i is the minimum among all alternative actions, then its action probability increases by $\Delta P_i(n)$ and the action probabilities of the other actions are decreased. Otherwise, its action probability is decreased by $\Delta P_i(n)$ and the action probabilities of the other actions are increased.

Then we propose the following *State-Dependent (SD) algorithm:*

Let $a(n) = a_i$

If $X_i(n) = \min_j \{X_j(n)\}$, then

$$P_i(n) = P_i(n-1) + \alpha * [1 - X(n)] * \sum_{j \neq i} f_j[\mathbf{P}(n)]$$

$$P_j(n) = P_j(n-1) - \alpha * [1 - X(n)] * f_j[\mathbf{P}(n)]$$
$$\forall j \neq i$$

else

$$P_i(n) = P_i(n-1) - \beta * X(n) * \sum_{j \neq i} g_j[\mathbf{P}(n)]$$

$$P_j(n) = P_j(n-1) + \beta * X(n) * g_j[\mathbf{P}(n)]$$
$$\forall j \neq i$$

where $0 < \alpha, \beta < 1$.

where f_j and g_j are nonnegative continuous functions and $0 < P_i(n) < 1$, $\sum_{i=1}^{|a|} P_i(n) = 1$. f_j and g_j can be linear or nonlinear functions of $\mathbf{P}(n)$.

If the reward and penalty functions are linear functions of the action probabilities, then we have a *State-Dependent Linear (SDL) algorithm.*

6.4 Two-Step Learning Automata

In this section, we propose a learning automaton that uses two action levels to update its action probabilities. If the action that is chosen at step n is the best, and was also the best at step $n-1$, then we reward this action a lot by increasing its probability with a large step size. If the action that is chosen at step n is the best, but was not the best in the previous step, then we reward this action a little by increasing its probability with a small step size. Otherwise, if the action that is chosen at step n is not the best, and was also not the best at step $n-1$, then we penalize this action a lot by decreasing its probability with a large step size. If the action that is chosen at step n is not the best, but was the best in the previous step, then we penalize this action a little by decreasing its probability with a small step size. The above concepts leads us to the following *Two-Step algorithm*:

Let $a(n) = a_i$:

$If \quad X_i(n) = \min_j\{X_j(n)\}, then$

$if \qquad X_i(n-1) = \min_j\{X_j(n-1)\}, then$

$$P_i(n) = P_i(n-1) + \alpha^1 * [1 - P_i(n-1)]$$

$$P_j(n) = P_j(n-1) - \alpha^1 * P_j(n-1) \quad \forall j \neq p$$

else

$$P_i(n) = P_i(n-1) + \alpha^2 * [1 - P_i(n-1)]$$

$$P_j(n) = P_j(n-1) - \alpha^2 * P_j(n-1) \quad \forall j \neq p$$

else

$$if \quad X_i(n-1) = \min_j\{X_j(n-1)\}, then$$

$$P_i(n) = P_i(n-1) - \beta^2 * P_i(n-1)$$

$$P_j(n) = P_j(n-1) + \beta^2 * \left[\frac{1}{|a|-1} - P_j(n-1)\right] \quad \forall j \neq p$$

else

$$P_i(n) = P_i(n-1) - \beta^1 * P_i(n-1)$$

$$P_j(n) = P_j(n-1) + \beta^1 * \left[\frac{1}{|a|-1} - P_j(n-1)\right] \quad \forall j \neq p$$

where $0 < \alpha_2 < \alpha_1, \quad \beta_2 < \beta_1 < 1.$

6.5 Virtual Updating

Another way to update the action probabilities with less overhead, (but also less accuracy) is to update less frequently, for example at times τ_n. We consider two cases according to if we observe or not the environment between the update instants:

6.5.1 Observable State

We assume that action selection (based on the $P_i(t_k)$'s) is made at the update points. Knowing that there are $n_i(\tau_n)$ successes (favorable outcomes) and $u_i(\tau_n)$ failures (unfavorable outcomes) during $[\tau_n, \tau_{n+1})$, then we must increase the action probability of the selected action $n_i(\tau_n)$ times and decrease it $u_i(\tau_n)$ times. In a similar way we must decrease and increase the action probabilities of the other actions.

Since we do not want to keep track of the exact sequence of occurrence of failures and successes, we assume such sequences. There are several ways to accomplish this, for example :

i) Increase P_i in $n_i(\tau_n)$ updates, then decrease it in $u_i(\tau_n)$ updates.

ii) Let $n_i(\tau_n) \leq u_i(\tau_n)$. Increase and decrease P_i in $n_i(\tau_n)$ updates, then decrease it in $u_i(\tau_n) - n_i(\tau_n)$ updates.

iii) Let $n_i(\tau_n) > u_i(\tau_n)$. Increase and decrease P_i in $u_i(\tau_n)$ updates, then increase it in $n_i(\tau_n) - u_i(\tau_n)$ updates.

iv) Decrease P_i in $u_i(\tau_n)$ updates, then increase it in $n_i(\tau_n)$ updates.

v) Let $n_i(\tau_n) \leq u_i(\tau_n)$. Decrease and increase P_i in $n_i(\tau_n)$ updates, then decrease it in $u_i(\tau_n) - n_i(\tau_n)$ updates.

vi) Let $n_i(\tau_n) > u_i(\tau_n)$. Decrease and increase P_i in $u_i(\tau_n)$ updates, then increase it in $n_i(\tau_n) - u_i(\tau_n)$ updates.

We can solve these recurrence equations and have $P_i(\tau_{n+1}) = Function\ (P_i(\tau_n), n_i(\tau_n), u_i(\tau_n))$. Thus instead of updating $P_i(\tau_n)$ at every action success or failure, we update at the times τ_n.

A simpler idea is to weight the reward adaptation step size with the number of successes and the penalty step size with the number of failures. So, instead of α, we can use

$$\alpha * \frac{n_i(\tau_n)}{n_i(\tau_n) + u_i(\tau_n)}$$

and instead of β, we can use

$$\beta * \frac{u_i(\tau_n)}{n_i(\tau_n) + u_i(\tau_n)}$$

6.5.2 Non Observable State

Another way to update the action probabilities multiple times is based on a single measurement. If the system state does not change too rapidly, then we assume that the same outcome would have been repeated if we were continually measuring the system state, say l times until the next real measurement. So, we update the probabilities l times assuming the last outcome still holds. Note that the updating scheme is composed of recursive equations. This leads us to extend the previously proposed updating scheme by using one network state measurement, but many (for example l_k in region k) iterations of the scheme in one actual computing step (updating step from *n-1* to n). For clarity we show the transformation of only one network response region (the full detail is given in the appendix).

If $X_i(n) \leq \phi_1(m(n))$, then

$$\begin{aligned} P_i(n) &= P_i(n-1) * \{1 - \alpha_1 * [1 - X(n)]\} + \\ &\quad + \alpha_1 * [1 - X(n)] \end{aligned}$$

$$P_j(n) = P_j(n-1) * \{1 - \alpha_1 * [1 - X(n)]\} \quad \forall j \neq i$$

Since the measurements for $X_j(n)$ do not change between $n-1$ and n, then $P_j(n)$ did not not change according to the previous updating schemes, so call them X_j and P_j. However, we shall update the action probabilities P_j multiple times,

say l_1, based on the measurements X_j. So, by solving these recursive equations, we have the following equations

$$\begin{aligned} P_i(l_1) &= P_i * \{1 - \alpha_1 * [1 - X]\}^{l_1} + \\ &\quad + \alpha_1 * [1 - X] * \sum_{i=0}^{l_1 - 1} \{1 - \alpha_1 * [1 - X]\}^i \end{aligned}$$

$$P_j(l_1) = P_j * \{1 - \alpha_1 * [1 - X]\}^{l_1} \quad \forall j \neq i$$

and the updating scheme becomes

If $X_i(n) \leq \phi_1(m(n))$, then

$$\begin{aligned} P_i(n) &= P_i(n-1) * \{1 - \alpha_1 * [1 - X(n)]\}^{l_1} + \\ &\quad + \alpha_1 * [1 - X(n)] * \sum_{i=0}^{l_1 - 1} \{1 - \alpha_1 * [1 - X(n)]\}^i \end{aligned}$$

$$P_j(n) = P_j(n-1) * \{1 - \alpha_1 * [1 - X(n)]\}^{l_1} \quad \forall j \neq i$$

We can use different $l_k, k = 1, ..., R$ and $m_k, k = 1, ..., P$ for different regions, where $l_1 \geq l_2 \geq ... \geq l_R > 0$, and $m_1 \geq m_2 \geq ... \geq m_P > 0$, are positive integers.

6.5.3 Frequent Updating

In the previous section, we updated as little as possible in order to reduce the measurement and computation overhead. However, the best results will be achieved if we measure and update the action probabilities as often as possible. Then the action algorithm will track the system state faster and the decisions will be better. Of course this will introduce more overhead of transmitting, selecting, storing and computing the state statistics.

6.6 Multiple Response Learning Automata

In this section, we introduce Multiple Response (MR) learning automata algorithms. The idea is to use different adaptation rates for different environment responses ($X(n)$). If the environment response is far away from optimum, the algorithm should converge faster, while if the environment response is near to optimum the algorithm should have smaller fluctuation. Whenever the environment response is very good ($X(n) \rightarrow 0$) (reward response 1), then the probability of the selected action increases very fast ($\alpha \rightarrow 1$). When the environment response is almost good (reward response R), then the probability of the selected action increases slowly ($\alpha \rightarrow 0$). Correspondingly, whenever the environment response is very bad ($X(n) \rightarrow 1$) (penalty response 1), then the probability of the selected action decreases very fast ($\beta \rightarrow 1$). When the cost of the environment response is almost bad (penalty response P), then the probability of the selected action decreases slowly ($\beta \rightarrow 0$).

6.6.1 Q-MR Learning Automata

In this section, we introduce a Q-model MR learning automaton algorithm, for which the environment's response takes discrete values. So, if action a_i was selected at time n, the environment's response is an element of the set $\{X_i^1, \ldots, X_i^R, \bar{X}_i^P, \ldots, \bar{X}_i^1\}$, i.e.

Let $a(n) = a_i$

reward response 1: $\quad X(n) = X_i^1(X(n))$

reward response 2: $\quad X(n) = X_i^2(X(n))$

...

reward response R: $\quad X(n) = X_i^R(X(n))$

penalty response P: $\quad X(n) = \bar{X}_i^P(X(n))$

penalty response P-1: $\quad X(n) = \bar{X}_i^{P-1}(X(n))$

...

penalty response 1: $\quad X_i(n) = \bar{X}_i^1(X(n))$

where $0 \leq X_i^1 < X_i^2 < ... < X_i^R < m_i < \bar{X}_i^P < \bar{X}_i^{P-1} < ... < \bar{X}_i^1 \leq 1$ are functions of $X(n)$.

A possible sequence for these functions $\{X_i^r(X(n))\}$ could be a Fibonacci sequence (normalized to the $[0, m_i(X(n))$ interval). Also a possible sequence for the functions $\{\bar{X}_i^p(X(n))\}$ could be a Fibonacci sequence (normalized to the $(m_i(X(n)), 1]$ interval).

If the selected action a_i results in good environment response ($0 \leq X(n) < m_i(X(n))$), then we reward this action, otherwise ($m_i(X(n)) < X(n) \leq 1$), we penalize it. The reward (penalty) parameters depend on how good (bad) the environment response was. Therefore, for each of the above environment responses, we use different reward rates $\alpha^r, r = 1, ..., R$ and penalty rates $\beta^p, p = 1, ..., P$, with $1 > \alpha^1 > \alpha^2 > ... > \alpha^R > 0$, and $1 > \beta^1 > \beta^2 > ... > \beta^P > 0$.

The above concepts produce the *Q-model Multiple Response (Q-MR) algorithm*:
Let $a(n) = a_i$

$If\ X(n) = X_i^1(X(n)), then$

$$P_i(n+1) = P_i(n) + g_i^1(X(n))[1 - P_i(n)]$$

$$P_j(n+1) = P_j(n) - g_i^1(X(n))P_j(n) \qquad \forall\ j \neq i$$

...

$If\ X(n) = X_i^R(X(n)), then$

$$P_i(n+1) = P_i(n) + g_i^R(X(n))[1 - P_i(n)]$$

$$P_j(n+1) = P_j(n) - g_i^R(X(n))P_j(n) \qquad \forall\ j \neq i$$

$If\ X(n) = \bar{X}_i^P(X(n)), then$

$$P_i(n+1) = P_i(n) - h_i^P(X(n))P_i(n)$$

$$P_j(n+1) = P_j(n) + h_i^P(X(n))\left[\frac{1}{|a|-1} - P_j(n)\right] \qquad \forall\ j \neq i$$

...

$If\ X(n) = \bar{X}_i^1(X(n)), then$

$$P_i(n+1) = P_i(n) - h_i^1(X(n))P_i(n)$$

$$P_j(n+1) = P_j(n) + h_i^1(X(n))\left[\frac{1}{|a|-1} - P_j(n)\right] \qquad \forall\ j \neq i$$

where $g_i^r(.),\ h_i^p(.) \in (0,1)\ \ r = 1, ..., R\ \ p = 1, ..., P$ are nonnegative continuous functions.

Define $\mathbf{P}(n) = [P_1(n), ..., P_{|a|}(n)]^T$: vector of action probabilities.

Define $d_i^r = P[X(n) = X_i^r(n)/a(n) = a_i] \in (0,1)$ the probability for reward response r, when action a_i is selected, and $c_i^p = P[X(n) = \bar{X}_i^p(n)/a(n) = a_i] \in (0,1)$ the probability for penalty response p, when action a_i is selected, such that

$$\sum_{r=1}^{R} d_i^r + \sum_{p=1}^{P} c_i^p = 1$$

Define $M_0 = \frac{1}{|a|} \sum_{i=1}^{|a|} \left[\sum_{r=1}^{R} X_i^r d_i^r + \sum_{p=1}^{P} \bar{X}_i^p c_i^p \right]$.

The average penalty received by the automaton conditioned on $\mathbf{P}(n)$ is

$$M(n) = E[X(n)/\mathbf{P}(n)] =$$

$$= \sum_{i=1}^{|a|} E[X(n)/\mathbf{P}(n), a(n) = a_i] P_i(n) =$$

$$= \sum_{i=1}^{|a|} \left[\sum_{r=1}^{R} X_i^r d_i^r + \sum_{p=1}^{P} \bar{X}_i^p c_i^p \right] P_i(n) \Rightarrow$$

$$\lim_{n \to \infty} E[M(n)] = \sum_{i=1}^{|a|} \left[\sum_{r=1}^{R} X_i^r d_i^r + \sum_{p=1}^{P} \bar{X}_i^p c_i^p \right] \lim_{n \to \infty} E[P_i(n)]$$

Next, we prove that at each iteration of the *Q-MR algorithm*, the action probabilities are always non-negative and sum to 1.

Lemma : *feasibility*

The Q-model Multiple Response (Q-MR) algorithm preserves the feasibility of the action probability space.

Proof:

Let at time n the action probabilities are feasible and action a_i is selected, i.e. $0 \leq P_i(n) \leq 1$, $\sum_{i=1}^{|a|} P_i(n) = 1$ and

let $a(n) = a_i$:

If $X(n) = X_i^r(X(n))$, then

$$P_i(n+1) = g_i^r(X(n)) + P_i(n)[1 - g_i^r(X(n))] \geq 0$$
$$since\ 0 < g_i^r(X(n)) < 1\ and\ P_i(n) \geq 0,$$

$$P_j(n+1) = P_j(n)[1 - g_i^r(X(n))] \geq 0\ \forall j \neq i$$
$$since\ g_i^r(X(n)) < 1\ and\ P_j(n) \geq 0,$$

$$P_i(n+1) = g_i^r(X(n)) + P_i(n)[1 - g_i^r(X(n))] \leq g_i^r(X(n)) + [1 - g_i^r(X(n))] = 1,$$
$$since\ P_i(n) \geq 0,$$

$$P_j(n+1) = P_j(n)[1 - g_i^r(X(n))] \leq 1 - g_i^r(X(n)) < 1,$$
$$since\ P_j(n) \geq 0\ and\ g * r_i(X(n)) > 0,$$

$$\sum_{i=1}^{|a|} P_i(n+1) = \sum_{i=1}^{|a|} P_i(n) + g_i^r(X(n))[1 - P_i(n)] - \sum_{j=1 j \neq i}^{|a|} g_i^r(X(n)) P_j(n)$$
$$= \sum_{i=1}^{|a|} P_i(n) = 1.$$

If $X(n) = \bar{X}_i^p(X(n))$, then

$$P_i(n+1) = P_i(n)[1 - h_i^p(X(n))] \geq 0$$
$$since\ P_i(n) \geq 0\ and\ h_i^p(X(n)) < 1,$$

$$P_j(n+1) = h_i^p(X(n))\frac{1}{|a|-1} + P_j(n)[1 - h_i^p(X(n))] \geq 0,$$

$$since\ |a| > 1,\ 0 < h_i^p(X(n)) < 1\ and\ P_j(n) \geq 0,$$

$$P_i(n+1) = P_i(n)[1 - h_i^p(X(n))] \leq 1 - h_i^p(X(n)) < 1,$$
$$since\ 0 \leq P_i(n) \leq 1\ and\ 0 < h_i^p(X(n)),$$

$$P_j(n+1) = h_i^p(X(n))\frac{1}{|a|-1} + P_j(n)[1 - h_i^p(X(n))] \leq$$
$$\leq h_i^p(X(n))\frac{1}{|a|-1} + [1 - h_i^p(X(n))] \leq$$
$$\leq \frac{h_i^p(X(n))[2 - |a|] + |a| - 1}{|a|-1} \leq 1,$$
$$since\ 0 \leq P_j(n) \leq 1,\ 0 < h_i^p(X(n)) < 1\ and\ |a| \geq 2,$$

$$\sum_{i=1}^{|a|} P_i(n+1) = \sum_{i=1}^{|a|} P_i(n) - h_i^p(X(n))P_i(n) + \sum_{j=1 j\neq i}^{|a|} h_i^p(X(n))\left[\frac{1}{|a|-1} - P_j(n)\right]$$

$$= \sum_{i=1}^{|a|} P_i(n) = 1$$

□

Next, we prove that the *Q-MR algorithm* is not trapped in a specific action, i.e. no action is selected with probability 1. This is a desirable property for the problem that we consider, since the system conditions continuously change and even if an action is the best for a long time interval, it may not be always so. So, we like to give a chance to the other actions in case the have become better.

Lemma : *non-absorbing*

The Q-model Multiple Response (Q-MR) algorithm is non-absorbing.

Proof:

Let $a(n) = a_i$:

Since $\sum_{i=1}^{|a|} P_i(n) = 1$, not all $P_i(n)$'s are equal to 0. Therefore, $\exists\ j$ such that $P_j(n) \in (0, 1]$. Since the reward response r happens with nonzero reward probability $d_j^r \in (0, 1)$,

$P_j(n+1) = P_j(n) - g_j^r * P_j(n) < P_j(n)$ with positive probability $d_j^r > 0$.

Therefore,

$\mathbf{P}(n+1) \neq \mathbf{P}(n)$ with positive probability. □

For the special case of $f_i^r(.) = \theta * \alpha_i^r$ and $g_i^p(.) = \theta * \beta_i^p$, with $0 < \theta \leq 1$, $0 < \alpha_i^r < 1$, $0 < \beta_i^p < 1$, we have the *Q-model Multiple Response Linear (Q-MRL) algorithm:*

Let $a(n) = a_i$

$If\ X(n) = X_i^1(X(n)),\ then$

$$P_i(n+1) = P_i(n) + \theta\alpha_i^1[1 - P_i(n)]$$
$$P_j(n+1) = P_j(n) - \theta\alpha_i^1 P_j(n) \qquad \forall\ j \neq i$$

...

$If\ X(n) = X_i^R(X(n)),\ then$

$$P_i(n+1) = P_i(n) + \theta\alpha_i^R[1 - P_i(n)]$$
$$P_j(n+1) = P_j(n) - \theta\alpha_i^R P_j(n) \qquad \forall\ j \neq i$$

$If\ X(n) = \bar{X}_i^P(X(n)),\ then$

$$P_i(n+1) = P_i(n) - \theta\beta_i^P P_i(n)$$
$$P_j(n+1) = P_j(n) + \theta\beta_i^P \left[\frac{1}{|a|-1} - P_j(n)\right] \qquad \forall\ j \neq i$$

...

$If\ X(n) = \bar{X}_i^1(X(n)),\ then$

$$P_i(n+1) = P_i(n) - \theta\beta_i^1 P_i(n)$$
$$P_j(n+1) = P_j(n) + \theta\beta_i^1 \left[\frac{1}{|a|-1} - P_j(n)\right] \qquad \forall\ j \neq i$$

Next, we prove that at each step of the *Q-MRL algorithm*, we approach to the optimum action.

Theorem : stricly distance diminishing

The Q-model Multiple Response Linear (Q-MRL) algorithm with $\alpha_i^r = \alpha^r \quad \forall i$ *and* $\beta_i^P = \beta \quad \forall i$ *is strictly distance diminishing.*

Proof:

Let $\mathbf{P}(n)$ and $\mathbf{Q}(n)$ be two different trajectories of the action probabilities.

Let $a(n) = a_i$:

If $X(n) = X_i^r(X(n))$, then

$$P_i(n+1) = P_i(n) + \theta\alpha^r[1 - P_i(n)]$$
$$P_j(n+1) = P_j(n) - \theta\alpha^r P_j(n) \quad \forall j \neq i$$

$$Q_i(n+1) = Q_i(n) + \theta\alpha^r[1 - Q_i(n)]$$
$$Q_j(n+1) = Q_j(n) - \theta\alpha^r Q_j(n) \quad \forall j \neq i$$

Then

$$||\mathbf{P}(n+1) - \mathbf{Q}(n+1)|| = \left[\sum_{j=1}^{|a|} [P_j(n+1) - Q_j(n+1)]^2 \right]^{1/2} =$$

$$= \Big[[P_i(n) + \theta\alpha^r[1 - P_i(n)] - Q_i(n) - \theta\alpha^r[1 - Q_i(n)]]^2 +$$

$$+ \sum_{j=1 j\neq i}^{|a|} [P_j(n) - \theta\alpha^r P_j(n) - Q_j(n) + \theta\alpha^r Q_j(n)]^2 \Big]^{1/2} =$$

$$= \left[(1 - \theta\alpha^r)^2 \sum_{j=1}^{|a|} [P_j(n) - Q_j(n)]^2 \right]^{1/2} =$$

$$= (1 - \theta\alpha^r)||\mathbf{P}(n) - \mathbf{Q}(n)|| < ||\mathbf{P}(n) - \mathbf{Q}(n)||$$

since $0 < \theta < 1,\ 0 < \alpha^r < 1.$

If $X(n) = \bar{X}_i^p(X(n))$, then

$$P_i(n+1) = P_i(n) - \theta\beta^p P_i(n)]$$

$$P_j(n+1) = P_j(n) + \theta\beta^p \left[\frac{1}{|a| - 1} - P_j(n) \right] \quad \forall j \neq i$$

$$Q_i(n+1) = Q_i(n) - \theta\beta^p Q_i(n)]$$

$$Q_j(n+1) = Q_j(n) + \theta\beta^p \left[\frac{1}{|a| - 1} - Q_j(n) \right] \quad \forall j \neq i$$

Then

$$\|\mathbf{P}(n+1) - \mathbf{Q}(n+1)\| = \left[\sum_{j=1}^{|a|} [P_j(n+1) - Q_j(n+1)]^2 \right]^{1/2} =$$

$$= \Big[P_i(n) - \theta\beta^p P_i(n) - Q_i(n) + \theta\beta^p Q_i(n)]^2 +$$

$$+ \sum_{j=1 j\neq i}^{|a|} \left[P_j(n) + \theta\beta^p \left[\frac{1}{|a| - 1} - P_j(n) \right] - Q_j(n) - \theta\beta^p \left[\frac{1}{|a| - 1} - Q_j(n) \right] \right]^2 \Big]^{1/2} =$$

$$= \left[(1-\theta\beta^p)^2 \sum_{j=1}^{|a|} [P_j(n) - Q_j(n)]^2 \right]^{1/2} =$$

$$=(1\text{-}\theta\beta^p)||\mathbf{P}(n) - \mathbf{Q}(n)|| < ||\mathbf{P}(n) - \mathbf{Q}(n)||$$

since $0 < \theta < 1,\ 0 < \alpha^r < 1$. □

Define also the Q-MR$L_{\alpha=\beta}$ *algorithm*, when $R = P$ and $\alpha_i^k = \beta_i^k \quad k = 1, ..., R$, and the Q-MR$L_{\alpha=\epsilon\beta}$ *algorithm*, when $R = P$ and $\alpha_i^k = \epsilon\beta_i^k \quad k = 1, ..., R$.

Next, we evaluate the conditional expectation of $P_i(n+1)$ given $P_i(n)$:

$$E[P_i(n+1)/P_i(n)] =$$

$$\sum_{r=1}^{R} [P_i(n) + \theta\alpha_i^r[1 - P_i(n)]]\, P_i(n) d_i^r +$$

$$+\sum_{p=1}^{P} [P_i(n) - \theta\beta_i^p P_i(n)]\, P_i(n) c_i^p +$$

$$+ \sum_{j=1}^{|a|} \sum_{j\neq i}^{R} \sum_{r=1} \left[P_i(n) - \theta\alpha_j^r P_i(n)\right]\, P_j(n) d_j^r +$$

$$+ \sum_{j=1}^{|a|} \sum_{j\neq i}^{P} \sum_{p=1} \left[P_i(n) + \theta\beta_j^p \left[\frac{1}{|a|-1} - P_i(n)\right]\right]\, P_j(n) c_j^p =$$

$$= P_i(n) + \theta P_i(n) \sum_{r=1}^{R} \left[\alpha_i^r [1 - P_i(n)] d_i^r - \sum_{j=1 j \neq i}^{|a|} \alpha_j^r P_j(n) d_j^r \right] +$$

$$+\theta \sum_{p=1}^{P} \left[\sum_{j=1 j \neq i}^{|a|} \beta_j^p \left[\frac{1}{|a|-1} - P_i(n) \right] P_j(n) c_j^p - \beta_i^p [P_i(n)]^2 c_i^p \right] =$$

$$= P_i(n) + \theta P_i(n) \sum_{r=1}^{R} \left[\alpha_i^r \sum_{j=1 j \neq i}^{|a|} P_j(n) d_i^r - \sum_{j=1 j \neq i}^{|a|} \alpha_j^r P_j(n) d_j^r \right] +$$

$$+\theta \sum_{p=1}^{P} \left[\sum_{j=1 j \neq i}^{|a|} \beta_j^p \left[\frac{1}{|a|-1} - P_i(n) \right] P_j(n) c_j^p - \beta_i^p [P_i(n)]^2 c_i^p \right] =$$

$$= P_i(n) + \theta P_i(n) \sum_{r=1}^{R} \sum_{j=1 j \neq i}^{|a|} P_j(n)(\alpha_i^r d_i^r - \alpha_j^r d_j^r) +$$

$$+\theta \sum_{p=1}^{P} \left[\sum_{j=1 j \neq i}^{|a|} \beta_j^p \left[\frac{1}{|a|-1} - P_i(n) \right] P_j(n) c_j^p - \beta_i^p [P_i(n)]^2 c_i^p \right]$$

For the *Q-MRL algorithm* with $\alpha_i^r = \alpha^r \quad \forall i \quad \forall r$ and $\beta_i^p = \beta^p \quad \forall i \quad \forall p$, we have

$$E[P_i(n+1)/P_i(n)] = P_i(n) + \theta P_i(n) \sum_{r=1}^{R} \alpha^r \sum_{j=1 j \neq i}^{|a|} P_j(n)(d_i^r - d_j^r) +$$

$$+\theta \sum_{p=1}^{P} \beta^p \sum_{j=1 j \neq i}^{|a|} \left[\left[\frac{1}{|a|-1} - P_i(n) \right] P_j(n) c_j^p - [P_i(n)]^2 c_i^p \right]$$

For the Q-MR$L_{\alpha=\beta}$ *algorithm* with $\alpha_i^r = \beta_i^r = \alpha^r \quad \forall i \quad \forall r$, we have

$$E[P_i(n+1)/P_i(n)] = P_i(n) +$$

$$\theta \sum_{k=1}^{R} \alpha^k \left[\sum_{j=1 j \neq i}^{|a|} \left[P_i(n) P_j(n) (d_i^k - d_j^k - c_j^k) + \frac{1}{|a|-1} P_j(n) c_j^k \right] - [P_i(n)]^2 c_i^k \right]$$

For the two-action ($|a| = 2$) *Q-MRL algorithm*, we have

$$\begin{aligned} E[P_1(n+1)/P_1(n)] &= P_1(n) + \theta P_1(n)[1 - P_1(n)] \sum_{r=1}^{R} (\alpha_1^r d_1^r - \alpha_2^r d_2^r) + \\ &+ \theta \sum_{p=1}^{P} \left[\beta_2^p [1 - P_1(n)]^2 c_2^p - \beta_1^p [P_1(n)]^2 c_1^p \right] \end{aligned}$$

For the two-action *Q-MRL algorithm* with $\alpha_i^r = \alpha^r \quad i = 1, 2 \quad \forall\, r$ and $\beta_i^p = \beta^p \quad i = 1, 2 \quad \forall\, p$, we have

$$\begin{aligned} E[P_1(n+1)/P_1(n)] &= P_1(n) + \theta P_1(n)[1 - P_1(n)] \sum_{r=1}^{R} \alpha^r (d_1^r - d_2^r) + \\ &+ \theta \sum_{p=1}^{P} \beta^p \left[[1 - P_1(n)]^2 c_2^p - [P_1(n)]^2 c_1^p \right] \end{aligned}$$

For the two-action Q-MR$L_{\alpha=\beta}$ *algorithm* with $\alpha_i^k = \beta_i^k = \alpha^k \quad i = 1, 2 \quad k = 1, ..., R$, we have

$$\begin{aligned} E[P_1(n+1)/P_1(n)] &= P_1(n) + \theta \sum_{k=1}^{R} \alpha^k \left[P_i(n)[1 - P_1(n)](d_1^k - d_2^k) + \right. \\ &+ \left. [1 - P_1(n)]^2 c_2^k - [P_1(n)]^2 c_1^k \right] \end{aligned}$$

If in addition $d_1^k + c_1^k = d_2^k + c_2^k \quad k = 1, ..., R$, then the above relation becomes

$$E[P_1(n+1)/P_1(n)] = P_1(n) +$$

$$+\theta \sum_{k=1}^{R} \alpha^k \left[P_1(n)(d_1^k - d_2^k) - [P_1(n)]^2(d_1^k - d_2^k) - \right.$$

$$\left. - [P_1(n)]^2 c_1^k + c_2^k - 2P_1(n)c_2^k + [P_1(n)]^2 c_2^k \right] =$$

$$= P_1(n) + \theta \sum_{k=1}^{R} \alpha^k \left[P_1(n)(d_1^k - d_2^k - 2c_2^k) + c_2^k \right] =$$

$$= P_1(n) - \theta \sum_{k=1}^{R} \alpha^k \left[P_1(n)(c_1^k + c_2^k) - c_2^k \right] \Rightarrow$$

$$E[P_1(n+1)] = E[P_1(n)] - \theta \sum_{k=1}^{R} \alpha^k \left[E[P_1(n)](c_1^k + c_2^k) - c_2^k \right]$$

$$= [1 - \theta \sum_{k=1}^{R} \alpha^k (c_1^k + c_2^k)] E[P_1(n)] - \theta \sum_{k=1}^{R} \alpha^k c_2^k \Rightarrow$$

$$E[P_1(n)] = \left[1 - \theta \sum_{k=1}^{R} \alpha^k (c_1^k + c_2^k) \right]^n P_1(0) -$$

$$- \frac{1 - \left[1 - \theta \sum_{k=1}^{R} \alpha^k (c_1^k + c_2^k) \right]^n}{\theta \sum_{k=1}^{R} \alpha^k (c_1^k + c_2^k)} \theta \sum_{k=1}^{R} \alpha^k c_2^k$$

Finally, if

$$\left|1 - \theta \sum_{k=1}^{R} \alpha^k (c_1^k + c_2^k)]\right| < 1$$

then

$$\lim_{n\to\infty} E[P_1(n)] = \frac{\sum_{k=1}^{R} \alpha^k c_2^k}{\sum_{k=1}^{R} \alpha^k (c_1^k + c_2^k)}$$

Thus if $\sum_{k=1}^{R} \alpha^k c_2^k < \sum_{k=1}^{R} \alpha^k c_1^k$, i.e. the penalty probability for action a_2 is smaller than the action probability for action a_1, then $\lim_{n\to\infty} E[P_1(n)] < \lim_{n\to\infty} E[P_2(n)]$, i.e. on the average action a_2 is chosen asymptotically with a higher probability than action a_1. The following Theorems follow:

Theorem : *ergodic*

The Q-MR algorithm is ergodic and $\mathbf{P}(n)$ *converges in distribution to a random variable* $\mathbf{P}^*$ *independent of the initial probability* $\mathbf{P}(0)$.

Theorem : *expedient*

The Q-MRL$_{\alpha-\beta}$ *algorithm with* $A = B$, $\alpha^k = \beta^k$, $d_1^k + c_1^k = d_2^k + c_2^k$, *is expedient.*

Proof:

$$\lim_{n\to\infty} E[M(n)] = \sum_{i=1}^{|a|} \left[\sum_{r=1}^{R} X_i^r d_i^r + \sum_{p=1}^{P} \bar{X}_i^p c_i^p \right] \frac{\frac{1}{\sum_{k=1}^{R} \alpha^k c_i^k}}{\sum_{k=1}^{|a|} \frac{1}{\sum_{k=1}^{R} \alpha^k c_i^k}} < M_0$$

□

Rewritting the conditional expected difference of the probability P_i from step n to step $n+1$, we have

$$E[P_i(n+1) - P_i(n)/\mathbf{P}(n) = \mathbf{P}] = \theta \left[\sum_{r-1}^{R} \alpha^r [1 - P_i] P_i d_i^r - \right.$$

$$- \sum_{p=1}^{P} \beta^p [P_i]^2 c_i^p -$$

$$- \sum_{r=1}^{R} \alpha^r P_i \sum_{j=1 j \neq i}^{|a|} P_j d_j^r +$$

$$\left. + \sum_{p=1}^{P} \beta^p \left[\frac{1}{|a| - 1} - P_i \right] \sum_{j=1 j \neq i}^{|a|} P_j c_j^p \right]$$

Define the follwing functions:

$$W_i^R(\mathbf{P}) = P_i \sum_{r=1}^{R} \alpha^r \sum_{j=1 j \neq i}^{|a|} P_j (d_i^r - d_j^r)$$

$$W_i^P(\mathbf{P}) = \sum_{p=1}^{P} \beta^p \sum_{j=1 j \neq i}^{|a|} \left[\left[\frac{1}{|a| - 1} - P_i \right] P_j c_j^p - [P_i]^2 c_i^p \right]$$

$$W_i(\mathbf{P}) = W_i^R(\mathbf{P}) + W_i^P(\mathbf{P}), \quad \sum_{i=1}^{|a|} W_i(\mathbf{P}) = 0$$

$$\mathbf{W}(\mathbf{P}) = [W_1(\mathbf{P}), ..., W_{|a|}(\mathbf{P})]^T$$

Then we can write the expectation of the conditional incremental action probabilities as:

$$E[\mathbf{P}(n+1) - \mathbf{P}(n)/\mathbf{P}(n) = \mathbf{P}] = \theta\mathbf{W}(\mathbf{P})$$

$$E[[\mathbf{P}(n+1) - \mathbf{P}(n)][\mathbf{P}(n+1) - \mathbf{P}(n)]^T/\mathbf{P}(n) = \mathbf{P}] = \theta^2\mathbf{W}'(\mathbf{P})$$

$$E[|\mathbf{P}(n+1) - \mathbf{P}(n)|^n/\mathbf{P}(n) = \mathbf{P}] = \theta^n\mathbf{W}''(\mathbf{P})$$

The above defined functions have also the following properties:

$\mathbf{W}(\mathbf{P})$ is twice continuously differentiable in S_r.
$\mathbf{W}'(\mathbf{P}) - \mathbf{W}(\mathbf{P}) * \mathbf{W}^T(\mathbf{P})$ is differentiable in S_r.

Next, we evaluate the probability of action a_1, when there are only two possible actions ($|a| = 2$):

$$P_1(n+1) = \begin{cases} P_1(n) + \theta\alpha^1[1 - P_1(n)] & if\ a(n) = a_1\ and\ reward\ response\ 1 \\ \cdots \\ P_1(n) + \theta\alpha^R[1 - P_1(n)] & if\ a(n) = a_1\ and\ reward\ response\ R \\ \\ P_1(n) - \theta\beta^P P_1(n) & if\ a(n) = a_1\ and\ penalty\ response\ P \\ \cdots \\ P_1(n) - \theta\beta^1 P_1(n) & if\ a(n) = a_1\ and\ penalty\ response\ 1 \\ \\ P_1(n) - \theta\alpha^1 P_1(n) & if\ a(n) = a_2\ and\ reward\ response\ 1 \\ \cdots \\ P_1(n) - \theta\alpha^R P_1(n) & if\ a(n) = a_2\ and\ reward\ response\ R \\ \\ P_1(n) + \theta\beta^P[1 - P_1(n)] & if\ a(n) = a_2\ and\ penalty\ response\ P \\ \cdots \\ P_1(n) + \theta\beta^1[1 - P_1(n)] & if\ a(n) = a_2\ and\ penalty\ response\ 1 \end{cases}$$

Then the previously defined functions become:

$$W_1^R(P_1) = P_1 \sum_{r=1}^{R} \alpha^r (1 - P_1)(d_1^r - d_2^r)$$

$$W_1^P(P_1) = \sum_{p=1}^{P} \beta^p \left[(1 - P_1)^2 c_2^p - (P_1)^2 c_1^p\right]$$

The following Theorem characterizes the zeros of the function $W(\mathrm{P})$:

Theorem :

For the Q-MRL algorithm,

$\exists$ *unique* $Q_1, Q_2 \in (0,1)$ *such that* $W^P(Q_1) = 0$ *and* $W(Q_2) = 0$ *and*

i) $Q_2 > Q_1 > \frac{1}{2}$, *when* $\sum_{r=1}^{R} \alpha^r (d_1^r - d_2^r) > 0$ *and* $\sum_{p=1}^{P} \beta^p (c_2^p - c_1^p) > 0$ *or*

ii) $Q_2 < Q_1 < \frac{1}{2}$, *when* $\sum_{r=1}^{R} \alpha^r (d_1^r - d_2^r) < 0$ *and* $\sum_{p=1}^{P} \beta^p (c_2^p - c_1^p) < 0$

Proof:

$$W_1^P(0) = \sum_{p=1}^{P} \beta^p c_2^p > 0$$

$$W_1^P(\frac{1}{2}) = \frac{1}{4} \sum_{p=1}^{P} \beta^p (c_2^p - c_1^p)$$

$$W_1^P(1) = \sum_{p=1}^{P} \beta^p (-c_1^p) < 0$$

$$If \quad \sum_{p=1}^{P} \beta^p (c_2^p - c_1^p) = 0, \qquad then \;\; W_1^P(Q_1) = 0 \;\; for \;\; Q_1 = \tfrac{1}{2}$$

$$if \sum_{p=1}^{P} \beta^p (c_2^p - c_1^p) > 0, \;\; then \;\; W_1(Q_2) = 0 \;\; for \;\; Q_2 = Q_1 = \tfrac{1}{2}$$

$$If \quad \sum_{p=1}^{P} \beta^p (c_2^p - c_1^p) > 0, \qquad then \;\; W_1^P(Q_1) = 0 \;\; for \;\; Q_1 > \tfrac{1}{2}$$

$$if \sum_{p=1}^{P} \beta^p (c_2^p - c_1^p) > 0, \;\; then W_1(Q_2) = 0 \;\; for \;\; Q_2 = Q_1 = \tfrac{1}{2}$$

$$If \quad \sum_{p=1}^{P} \beta^p (c_2^p - c_1^p) = 0, \qquad then \;\; W_1^P(Q_1) = 0 \;\; for \;\; Q_1 < \tfrac{1}{2}$$

$$if \sum_{p=1}^{P} \beta^p (c_2^p - c_1^p) > 0, \;\; then \;\; W_1(Q_2) = 0 \;\; for \;\; Q_2 = Q_1 = \tfrac{1}{2}$$

Similarly for the other cases □

If in the algorithm, we replace β^p by $\epsilon\beta^p$, then $\mathbf{W}(\mathbf{P})$ and $\mathbf{W}^{P'}(\mathbf{P})$ get multiplied by ϵ and ϵ^2.

Define

$$\mathbf{W}(\epsilon, \mathbf{P}) = \mathbf{W}^R(\mathbf{P}) + \epsilon \mathbf{W}^P(\mathbf{P}) \quad 0 < \epsilon \leq 1$$

$$\mathbf{W}(1, \mathbf{P}) = \mathbf{W}(\mathbf{P})$$

Then the following Theorem follows:

Theorem :

For the Q-MRL algorithm $\exists\ Q(\epsilon) \in (0,1)$, *such that* $W(\epsilon, Q(\epsilon)) = 0$ *and*

i) $Q(\epsilon) \geq Q_2$ *and* $Q(\epsilon) \to 1$ *as* $\epsilon \to 0$,

when $\sum_{r=1}^{R} \alpha^r (d_1^r - d_2^r) > 0$ *and* $\sum_{p=1}^{P} \beta^p (c_2^p - c_1^p) > 0$ *or*

ii) $Q(\epsilon) \leq Q_2$ *and* $Q(\epsilon) \to 0$ *as* $\epsilon \to 0$,

when $\sum_{r=1}^{R} \alpha^r (d_1^r - d_2^r) < 0$ *and* $\sum_{p=1}^{P} \beta^p (c_2^p - c_1^p) < 0$

Proof:

i)

$$\mathbf{W}(\epsilon, \mathbf{P}) = \mathbf{W}(1, \mathbf{P}) - (1 - \alpha)\mathbf{W}^P(\mathbf{P})$$

$$\mathbf{W}(1, \mathbf{P}) = \mathbf{W}(\mathbf{P})$$

Then

$$\mathbf{W}(\epsilon, Q_2) = -(1 - \alpha)\mathbf{W}^P(Q_2) > 0$$

$$\mathbf{W}(\epsilon, 1) = 0$$

Therefore $\mathbf{W}(\epsilon, Q(\epsilon)) = 0 \quad Q(\epsilon) \geq Q_2$

If $\epsilon' > \epsilon$, then $\mathbf{W}(\epsilon', Q(\epsilon)) = \mathbf{W}^P(Q(\epsilon))(\epsilon' - \epsilon) > 0$

i.e. $Q(\epsilon)$ increases as ϵ decreases.

$\mathbf{W}(0,1) = 0$, thus 1 is the least upper bound on $Q(\epsilon)$.

Similarly for case ii). □

6.6.2 S-MR Learning Automata

In this section, we introduce an S-model MR learning automaton algorithm, for which the environment's response takes continuous values:

Let $a(n) = a_i$

reward response 1: $0 \leq X(n) \leq X_i^1(X(n))$

reward response 2: $X_i^1(X(n)) < X(n) \leq X_i^2(X(n))$

...

reward response R: $X_i^{R-1}(X(n)) < X(n) \leq m_i(X(n))$

penalty response P: $m_i(X(n)) < X(n) \leq \bar{X}_i^{P-1}(X(n))$

penalty response P-1: $\bar{X}_i^{P-1}(X(n)) < X(n) \leq \bar{X}_i^{P-2}(X(n))$

...

penalty response 1: $\bar{X}_i^1(X(n)) < X_i(n) \leq 1$

where $0 \leq X_i^1 < X_i^2 < ... < X_i^R < m_i << \bar{X}_i^{P-1} < ... < \bar{X}_i^1 \leq 1$ are functions of $X(n)$.

A possible sequence for these functions $\{X_i^r(X(n))\}$ could be a Fibonacci sequence (normalized to the $[0, m_i(X(n))$ interval). Also a possible sequence for the functions $\{\bar{X}_i^p(X(n))\}$ could be a Fibonacci sequence (normalized to the $(m_i(X(n)), 1]$ interval).

If the selected action a_i results in good environment response ($0 \leq X(n) < m_i(X(n))$), then we reward this action, otherwise ($m_i(X(n)) < X(n) \leq 1$), we penalize it. The reward (penalty) parameters depend on how good (bad) the environment response was. Therefore, for each of the above environment responses, we use different reward rates $\alpha^r, r = 1, ..., R$ and penalty rates $\beta^p, p = 1, ..., P$, with $1 > \alpha^1 > \alpha^2 > ... > \alpha^R > 0$, and $1 > \beta^1 > \beta^2 > ... > \beta^P > 0$.

The above concepts produce the *S-model Multiple Response (Q-MR) algorithm*:
Let $a(n) = a_i$

$If\ 0 \leq X(n) \leq X_i^1(X(n)), then$

$$P_i(n+1) = P_i(n) + g_i^1(X(n))[1 - P_i(n)]$$

$$P_j(n+1) = P_j(n) - g_i^1(X(n))P_j(n) \qquad \forall\ j \neq i$$

...

$If\ X_i^{R-1}(X(n)) < X(n) \leq m_i(X(n)), then$

$$P_i(n+1) = P_i(n) + g_i^R(X(n))[1 - P_i(n)]$$

$$P_j(n+1) = P_j(n) - g_i^R(X(n))P_j(n) \qquad \forall\ j \neq i$$

$If\ m_i(X(n)) < X(n) \leq \bar{X}_i^{P-1}(X(n)), then$

$$P_i(n+1) = P_i(n) - h_i^P(X(n))P_i(n)$$

$$P_j(n+1) = P_j(n) + h_i^P(X(n))\left[\frac{1}{|a|-1} - P_j(n)\right] \qquad \forall\ j \neq i$$

...

$If\ \bar{X}_i^1(X(n)) > X(n) \leq 1, then$

$$P_i(n+1) = P_i(n) - h_i^1(X(n))P_i(n)$$

$$P_j(n+1) = P_j(n) + h_i^1(X(n))\left[\frac{1}{|a|-1} - P_j(n)\right] \qquad \forall\ j \neq i$$

where $g_i^r(.),\ h_i^P(.)\ \in (0,1)\ \ r = 1, ..., R\ \ p = 1, ..., P$ are nonnegative continuous tunctions.

We can also prove several properties of the *S-MR* algorithm similar to those of the *Q-MR* algorithm.

Lemma : *feasibility*

The S-model Multiple Response (Q-MR) algorithm preserves the feasibility of the action probability space.

Lemma : *non-absorbing*

The S-model Multiple Response (Q-MR) algorithm is non-absorbing.

Theorem : strictly distance diminishing

The S-model Multiple Response Linear (Q-MRL) algorithm with $\alpha_i^r = \alpha^r \quad \forall i$ *and* $\beta_i^p = \beta \quad \forall i$ *is strictly distance diminishing.*

Theorem : *ergodic*

The S-MR algorithm is ergodic and $\mathbf{P}(n)$ *converges in distribution to a random variable* $\mathbf{P}^*$ *independent of the initial probability* $\mathbf{P}(0)$.

Theorem : *expedient*

The S-MRL$_{\alpha-\beta}$ *algorithm with* $A = B, \quad \alpha^k = \beta^k, \quad d_1^k + c_1^k = d_2^k + c_2^k,$ *is expedient.*

Theorem :

For the S-MRL algorithm $\exists\ Q(\epsilon) \in (0,1)$, *such that* $W(\epsilon, Q(\epsilon)) = 0$ *and*

i) $Q(\epsilon) \geq Q_2$ *and* $Q(\epsilon) \rightarrow 1$ *as* $\epsilon \rightarrow 0$,

when $\sum_{r=1}^{R} \alpha^r (d_1^r - d_2^r) > 0$ *and* $\sum_{p=1}^{P} \beta^p (c_2^p - c_1^p) > 0$ *or*

ii) $Q(\epsilon) \leq Q_2$ *and* $Q(\epsilon) \rightarrow 0$ *as* $\epsilon \rightarrow 0$,

when $\sum_{r=1}^{R} \alpha^r (d_1^r - d_2^r) < 0$ *and* $\sum_{p=1}^{P} \beta^p (c_2^p - c_1^p) < 0$

6.7 Application to Datagram Networks

In this section, we propose using stochastic learning automata for load sharing, routing and congestion control decisions in datagram networks. Glorioso & Colon [195, 194], Srikantakumar [454, 453, 455] Chrystall, Mars & Narendra [104, 370] Nedzelnitsky & Narendra [349, 350], Mason [321, 322] and Narendra & Wheeler [345] use learning automata in datagram routing and they update the routing probabilities according to the delay experienced by a packet.

Learning automata have also been used for routing decisions in datagram networks with variable quality links [141]. When packets repeatedly fail transmission through a link, due to high error rate of this link, then they are driven by learning automata to use a different link. For a full description of learning automata-based routing in such an unreliable network, we refer to our paper [141].

The methodology that we propose for learning automata-based load sharing, routing and congestion control decisions for new arriving packets (in datagram networks) is similar to that of the next section for new arriving virtual circuits (in virtual circuit networks). So, we do not reiterate it here.

6.8 Application to Virtual Circuit Networks

In this section, we propose using stochastic learning automata for load sharing, routing and congestion control decisions in virtual circuit networks. We have introduced learning automata for virtual circuit routing [136], where the routing probabilities are updated according to the unfinished work on the selected path (for user optimum), or the increase in the number of packets on a path (or the increase in the portion of the overall network delay corresponding to this path) due to the addition of a new virtual circuit on this path (for system optimum). We considered three cases regarding the availability of traffic measurements: i) measurements of the number of virtual ciruits and packets on each path are known, ii) measurements of the number of virtual ciruits on each path are known, iii) no measurements of the network state are known, but the virtual circuit arrival rates are known.

Perturbation analysis may also be used in order to estimate the derivatives of the cost function.

Everything in this section is for each class c, but for easier exposition we do not show the superscript of the class. At every source node $[s.]$, a load sharing decision maker selects the destination node $[.d]$ where a new virtual circuit will be executed. Then a router selects the path $\pi_{[sd]}$ through which the virtual circuit will be transferred to the destination node $[.d]$ or rejects the virtual circuit for congestion control reasons. The length of a path is given by the Theorems of chapters 4 and 5.

However, network conditions change very rapidly and the minimum length path at a time instant may not be the same at the next time instant. Also, the information about the network state is always obsolete and inacurrate. Therefore the load sharing, routing and congestion control decisions should not overreact and immediately send a new virtual circuit to the estimated minimum length destination through the minimum length path, because oscillations may appear [45]. The system management decisions should fast track the current network state but without introducing instability.

The proposed adaptive routing algorithms are based on a "Probabilistic Selection of the Minimum Length Path" idea. Instead of using a definitive decision as to where to send a newly arriving virtual circuit, we vary the load sharing, routing and congestion control probabilities favoring the minimum length destination and path. Finally, the lengths of the paths are equalized.

Every source node $[s.]$ has a learning automaton for selecting a destination node where a new arriving virtual circuit will be executed. These learning automata operate asynchronously and base their decisions on the current system state. The actions, $a(n)$, of each automaton are the selection of a particular destination node $[.d]$ for processing the virtual circuit.

The automaton selects action $a(n) = a_{[sd]}$ with probability $P_{[sd]}(n)$ at each instant n. Action $a(n)$ becomes input to the environment. If this results in a favorable outcome for the network performance ($X(n) \rightarrow 0$), then the probability $P_{[sd]}(n)$ is increased by $\Delta P_{[sd]}(n)$ and the $P_{[sd']}(n)$, $\forall [sd'] \neq [sd]$, are decreased by

$\Delta P_{[sd']}(n)$. Otherwise, if an unfavorable outcome ($X(n) - 1$) appears, then the $P_{[sd]}(n)$ is decreased by $\Delta P_{[sd]}(n)$ and the $P_{[sd']}(n)$, $\forall [sd'] \neq [sd]$ are increased by $\Delta P_{[sd']}(n)$.

We propose the following adaptive algorithm at every source node $[s.]$, for the load sharing decisions:

Probabilistic Selection of the Minimum Length Destination:

Suppose destination $[.d]$ was selected at time n-1, with $P_{[sd]}(n-1)$.

Compute the lengths to all destinations, $l_{[sd']}(n)$ $\forall [sd']$

Calculate $X(n) = X(..., l_{[sd']}(n), ...)$.

Update the load sharing probabilities $P_{[sd']}(n)$ $\forall [sd']$.

Select the destination for the n^{th} virtual circuit probabilistically according to $P_{[sd']}(n)$.

Similarly, for the routing and congestion control decisions, every source node $[s.]$ has a learning automaton for every destination node $[.d]$ that routes new arriving virtual circuits at node $[s.]$ and destined for node $[.d]$. These learning automata operate asynchronously and base their decisions on the current network state. The actions, $a(n)$, of each automaton are to select some particular path $\pi[sd]$ to the destination node $[.d]$.

The automaton selects action $a(n) = a_{\pi[sd]}$ with probability $P_{\pi[sd]}(n)$ at each instant n. Action $a(n)$ becomes input to the environment. If this results in a favorable outcome for the network performance ($X(n) \rightarrow 0$), then the probability $P_{\pi[sd]}(n)$ is increased by $\Delta P_{\pi[sd]}(n)$ and the $P_{p[sd]}(n)$, $\forall p[sd] \neq \pi[sd]$, are decreased by $\Delta P_{p[sd]}(n)$. Otherwise, if an unfavorable outcome ($X(n) \rightarrow 1$) appears, then the $P_{\pi[sd]}(n)$ is decreased by $\Delta P_{\pi[sd]}(n)$ and the $P_{p[sd]}(n)$, $\forall p[sd] \neq \pi[sd]$ are increased by $\Delta P_{p[sd]}(n)$.

We propose the following adaptive algorithm at every source node [$s.$], for routing virtual circuits to a certain destination node [$.d$] or for rejecting them when congestion exists into the network:

Probabilistic Selection of the Minimum Length Path:

Suppose path $\pi[sd]$ *was selected at time n-1, with* $P_{\pi[sd]}(n-1)$.

Compute all paths lengths, $l_{p[sd]}(n)\ \forall p[sd] and l_{o[sd]}(n)$.

Compute $X(n) = X(..., l_{p[sd]}(n), ...)$.

Update the routing probabilities $P_{p[sd]}(n)\ \forall p[sd],\ P_{o[sd]}(n)$.

Select the path for the n^{th} *virtual circuit or reject it probabilistically according to* $P_{p[sd]}(n),\ P_{o[sd]}(n)$.

6.8.1 Simulation Comparison of Algorithms

In this section, we apply three learning automata algorithms to the routing problem in virtual circuits networks and we compare their performance. All algorithms have the same reward and penalty parameters for different actions and the parameter $\theta = 1$. We consider as length of a path $\pi[sd]$ its average packet delay $T_{\pi[sd]}(n)$.

The simplest information that someone can measure and transfer about the network state is the packet delay through each path from source to destination. A new arriving virtual circuit is routed from its source to its destination along the path that promises the minimum packet delay. Instead of using a definitive decision as to where to send a newly arriving virtual circuit, we vary the path routing probabilities favoring the minimum delay path.

The first algorithm is the $L_{R-\epsilon P}$ learning automaton with reward parameter $\alpha = 0.2$ and penalty parameter $\beta = 0.8$. If the selected path has the minimum packet delay at the next iteration, then we increase the probability of selecting it again, otherwise we decrease it. More specifically:

Let path $\pi[sd]$ is selected at time n

$$If\ T_{\pi[sd]}(n) = \min_{p[sd] \in \Pi_{[sd]}} \{T_{p[sd]}(n)\},\ then$$

$$P_{\pi[sd]}(n+1) = P_{\pi[sd]}(n) + 0.2 * [1 - P_{\pi[sd]}(n)]$$

$$P_{p[sd]}(n+1) = P_{p[sd]}(n) - \ 0.2 * P_{p[sd]}(n) \qquad \forall\ p[sd] \neq \pi[sd]$$

$$else$$

$$P_{\pi[sd]}(n+1) = P_{\pi[sd]}(n) - 0.8 * P_{\pi[sd]}(n)$$

$$P_{p[sd]}(n+1) = P_{p[sd]}(n) + 0.8 * \left[1 - P_{p[sd]}(n)\right] \qquad \forall\ p[sd] \neq \pi[sd]$$

The second algorithm is the $MRL_{R-\epsilon P}$ learning automaton with reward parameters $\alpha^1 = 0.8$ (excellent choice), $\alpha^2 = 0.2$ (good choice, but not excellent) and penalty parameters $\beta^2 = 0.8$ (bad choice), $\beta^1 = 1$ (very bad choice). We consider two response and penalty regions for the algorithm ($P = R = 2$) and the functions that define these regions are linear functions with parameter 2. More specifically:

Let path $\pi[sd]$ is selected at time n

$$If\ T_{\pi[sd]}(n) \leq \min_{p[sd]\in\Pi_{[sd]}} \{T_{p[sd]}(n)/2\},\ then$$

$$P_{\pi[sd]}(n+1) = P_{\pi[sd]}(n) + 0.8 * [1 - P_{\pi[sd]}(n)]$$

$$P_{p[sd]}(n+1) = P_{p[sd]}(n) - 0.8 * P_{p[sd]}(n) \qquad \forall\ p[sd] \neq \pi[sd]$$

$$If \min_{p[sd]\in\Pi_{[sd]}} \{T_{p[sd]}(n)/2\} < T_{\pi[sd]}(n) \leq \min_{p[sd]\in\Pi_{[sd]}} \{T_{p[sd]}(n)\},$$

$$P_{\pi[sd]}(n+1) = P_{\pi[sd]}(n) + 0.2 * [1 - P_{\pi[sd]}(n)]$$

$$P_{p[sd]}(n+1) = P_{p[sd]}(n) - 0.2 * P_{p[sd]}(n) \qquad \forall\ p[sd] \neq \pi[sd]$$

$$If \min_{p[sd]\in\Pi_{[sd]}} \{T_{p[sd]}(n)\} \leq T_{\pi[sd]}(n) \leq \min_{p[sd]\in\Pi_{[sd]}} \{2 * T_{p[sd]}(n)\},$$

$$P_{\pi[sd]}(n+1) = P_{\pi[sd]}(n) - 0.8 * P_{\pi[sd]}(n)$$

$$P_{p[sd]}(n+1) = P_{p[sd]}(n) + 0.8 * \left[1 - P_{p[sd]}(n)\right] \qquad \forall\ p[sd] \neq \pi[sd]$$

$$If \min_{p[sd]\in\Pi_{[sd]}} \{2 * T_{p[sd]}(n)\} \leq T_{\pi[sd]}(n),$$

$$P_{\pi[sd]}(n+1) = P_{\pi[sd]}(n) - 1 * P_{\pi[sd]}(n)$$

$$P_{p[sd]}(n+1) = P_{p[sd]}(n) + 1 * \left[1 - P_{p[sd]}(n)\right] \qquad \forall\ p[sd] \neq \pi[sd]$$

Finally, the third algorithm is the $SDL_{R-\epsilon P}$ learning automaton with reward parameter $\alpha = 0.2$ and penalty parameter $\beta = 0.8$. The state dependent parameter is an exponential function of the difference of the selected path average delay and the maximum average delay of paths between this source-destination. More specifically:

Let path $\pi[sd]$ is selected at time n

$$If\ T_{\pi[sd]}(n) = \min_{p[sd] \in \Pi_{[sd]}} \{T_{p[sd]}(n)\},\ then$$

$$P_{\pi[sd]}(n+1) = P_{\pi[sd]}(n) + 0.2 * (1 - e^{T_{\pi[sd]}(n) - \max\limits_{p[sd]} T_{p[sd]}}) * [1 - P_{\pi[sd]}(n)]$$

$$P_{p[sd]}(n+1) = P_{p[sd]}(n) - 0.2 * (1 - e^{T_{\pi[sd]}(n) - \max\limits_{p[sd]} T_{p[sd]}}) * P_{p[sd]}(n)$$

$$\forall\ p[sd] \neq \pi[sd]$$

$else$

$$P_{\pi[sd]}(n+1) = P_{\pi[sd]}(n) - 0.8 * (1 - e^{T_{\pi[sd]}(n) - \max\limits_{p[sd]} T_{p[sd]}}) * P_{\pi[sd]}(n)$$

$$P_{p[sd]}(n+1) = P_{p[sd]}(n) + 0.8 * (1 - e^{T_{\pi[sd]}(n) - \max\limits_{p[sd]} T_{p[sd]}}) * \left[1 - P_{p[sd]}(n)\right]$$

$$\forall\ p[sd] \neq \pi[sd]$$

It is important to understand that the above values for the reward and penalty parameters are not the optimum. Depending on the network topology, the number of paths between source-destination pairs, the traffic characteristics, the information about the network state, the updating time interval and other variables, we should choose the best parameters by experimentation. Note that the traditionally used shortest path algorithm is a special case of the learning automata algorithm, since by suitable tuning the parameters, we can select the minimum length path with probability 1.

In this section, we compare the performance of the three learning automata algorithms (see previous section) via simulation. We consider a network with two paths from source to destination. So, each learning automaton has two actions $\|a\| = 2$ to choose. Path # 1 has seven links with service rates 1. Path # 2 has seven links with service rates 1, 0.5, 2, 2, 2, 0.5 and 1 (Figure 6.2).

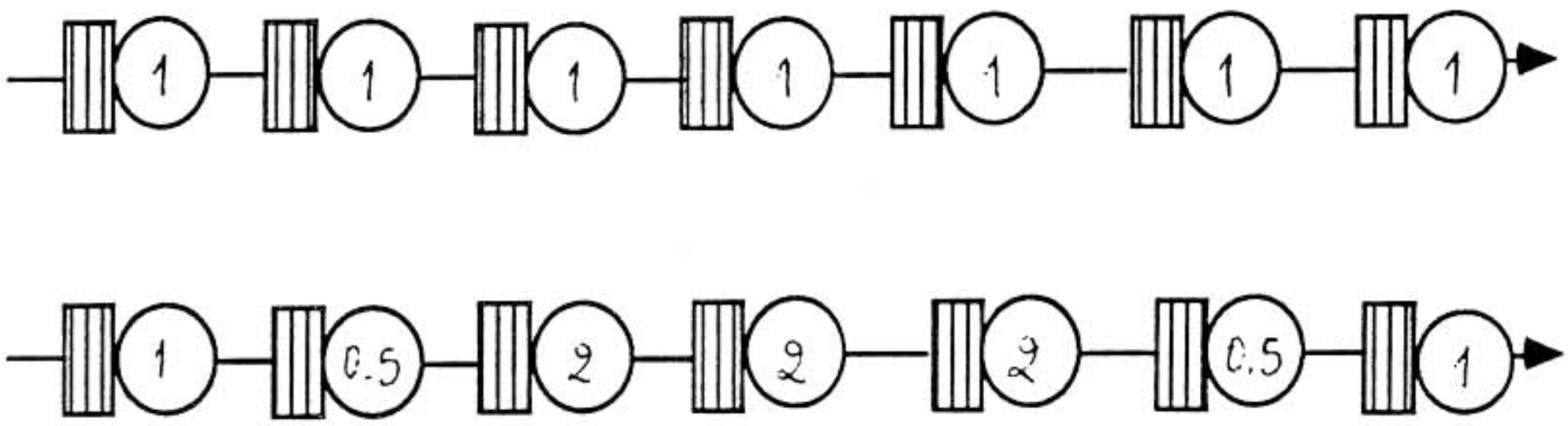


Figure 6.2: Simulated network.

The mean packet service requirement is $1/\mu = 1$ and therefore $\mu_{ij} = \mu * C_{ij} = C_{ij}$. For the rest traffic characteristics, we considered two cases:

i) 30/2/40: the virtual circuit arrival rate is $\gamma = 1/30$, the packet arrival rate per virtual circuit is $r = 1/2$ and the mean virtual circuit duration is $\delta = 40$.

ii) 50/5/200: the virtual circuit arrival rate is $\gamma = 1/50$, the packet arrival rate per virtual circuit is $r = 1/5$ and the mean virtual circuit duration is $\delta = 200$.

For measuring the path delay, we consider two cases:

i) 1 : at every packet departure from the network through a path, the destination sends to the source the packet delay through the path of that last packet.

ii) 50 : at every 50 packet departure from the network through a path, the destination sends to the source the average packet delay through the path of these 50 last packets.

The source node keeps and updates the information about the delay of its paths to the destination. The information about the delay of a path is updated every time a packet arrives at the destination through this path. However, this updating is not done immediately, but we assume a feedback delay so that this information becomes available to the source node. We assume that no extra traffic is created for transferring this feedback information to the source node (it is either piggybacked on regular packets or uses a different channel). We consider two cases:

i) *instantaneous* information, when the feedback delay is 7 time units. In this case, we assume that the feedback information has higher priority over other packets and does not wait in queues.

ii) *obsolete* information, when the feedback delay is 60 time units. In this case, we assume that the feedback information is piggybacked on regular packets and is transferred back to the source node.

Updating the information of a path asynchronously at packet departure instances has an undesirable characteristic. If a path becomes unattractive for routing packets through it, then we may not route any more packets through it. However, our information about its length remains the same, although after some time this path may become idle. We have overcome this problem by sending a

30/2/40	1 instant	1 obsolete	50 instant	50 obsolete
deterministic	50.59 ±0.89	63.59 ±1.28	55.38 ±0.93	61.97 ±1.36
L automaton	50.27 ±1.15	61.29 ±1.36	57.37 ±0.88	61.44 ±1.25
MRL automaton	50.64 ±0.73	61.27 ±1.63	61.15 ±0.92	64.04 ±1.36
SDL automaton	48.92 ±0.51	62.52 ±1.04	57.37 ±1.14	60.60 ±1.46

50/5/200	1 instant	1 obsolete	50 instant	50 obsolete
deterministic	46.79 ±1.75	57.84 ±1.92	60.52 ±2.21	68.30 ±2.23
L automaton	45.35 ±1.45	54.85 ±2.31	61.43 ±1.76	65.43 ±1.77
MRL automaton	43.25 ±1.45	56.45 ±2.13	62.05 ±3.16	65.67 ±2.52
SDL automaton	46.22 ±1.36	57.45 ±2.17	60.81 ±1.79	67.24 ±1.68

Table 6.1: The average packet delay ± error (95% confidence interval) for the network of Figure 6.2 for deterministic, Linear automaton, Multiple Response automaton and State Dependent automaton based routing.

probe packet through a path that has not been used for 100 time units and therefore updating our information about its delay.

In Figures 6.3-6.10 and Table 6.1, we show the simulation results for the average packet delay for 10,000 virtual circuits.

Although the reward and penalty parameters of the learning automata were not chosen to be the best possible, all four algorithms achieve similar performance. However, the learning automata have more flexibility, since we can calibrate their parameters depending on the particular system. Note, that the deterministic algorithm is a special case of the learning automata, since we can choose their parameters, such that the minimum length path is chosen. The more frequent we update the algorithms and the more recent state information we have, the better the performance.

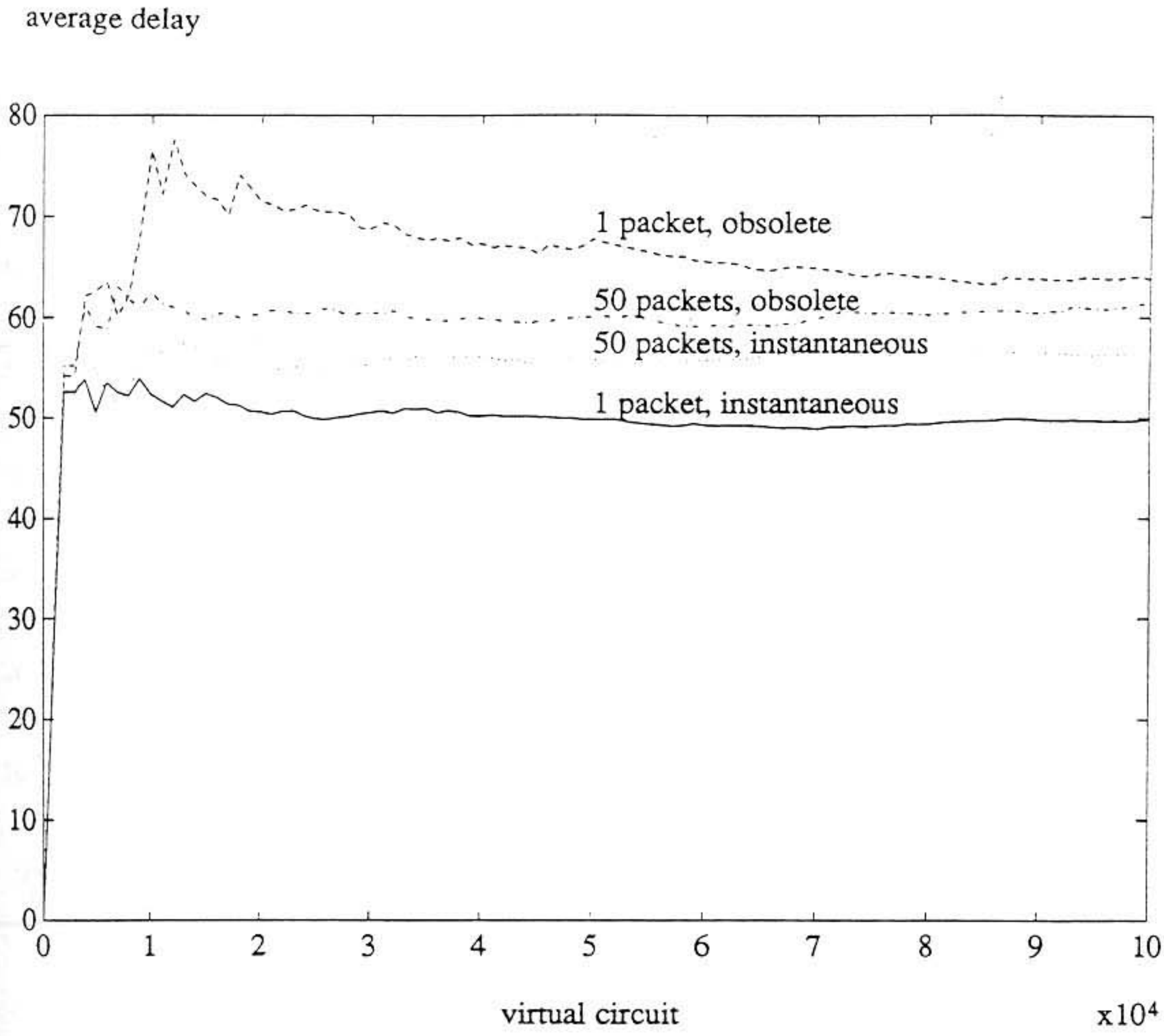


Figure 6.3: The average packet delay $\pm$ error (95% confidence interval) for the network of Figure 6.2 with $\gamma = 1/30$ for deterministic routing.

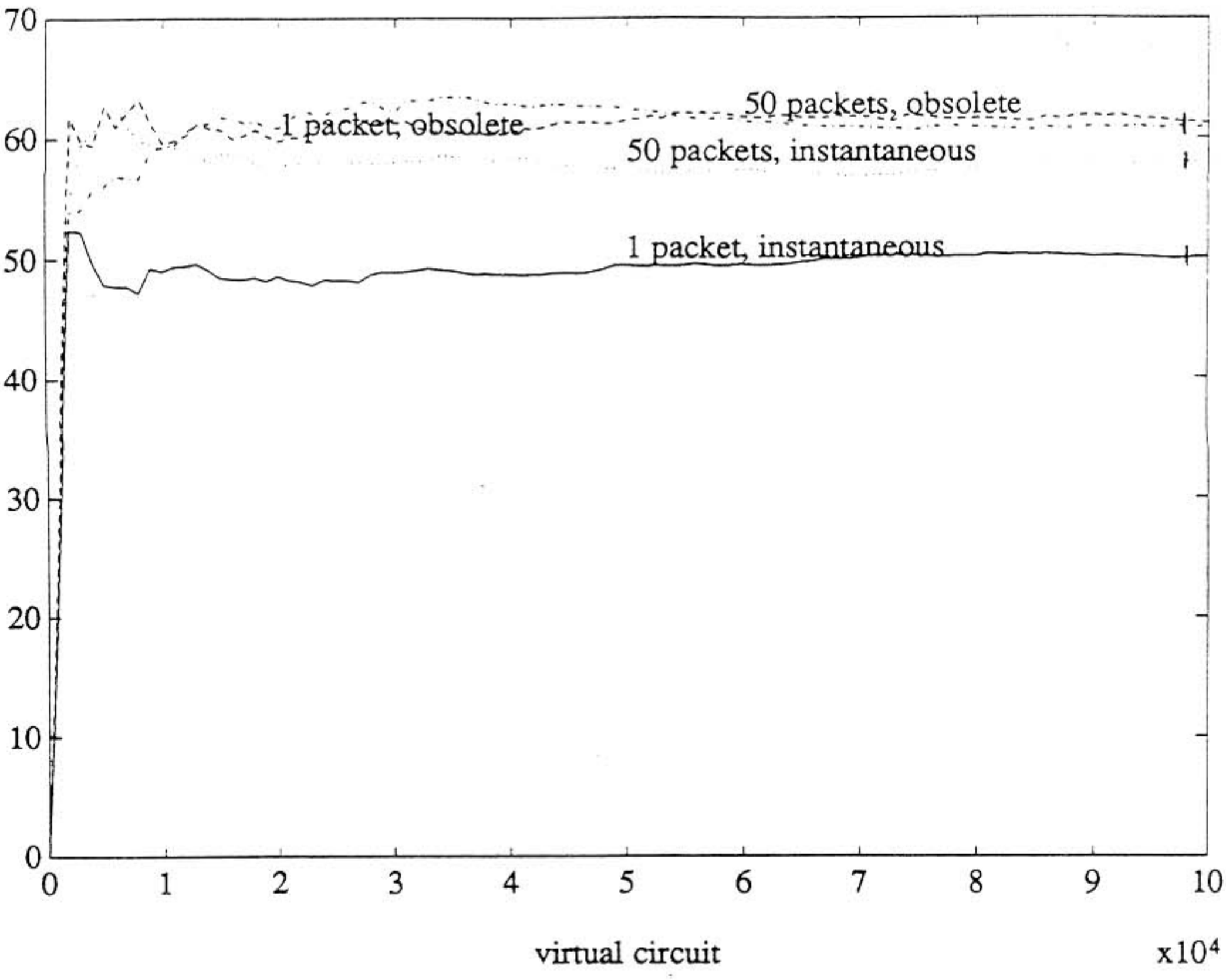


Figure 6.4: The average packet delay $\pm$ error (95% confidence interval) for the network of Figure 6.2 with $\gamma = 1/30$ for Linear learning automaton based routing.

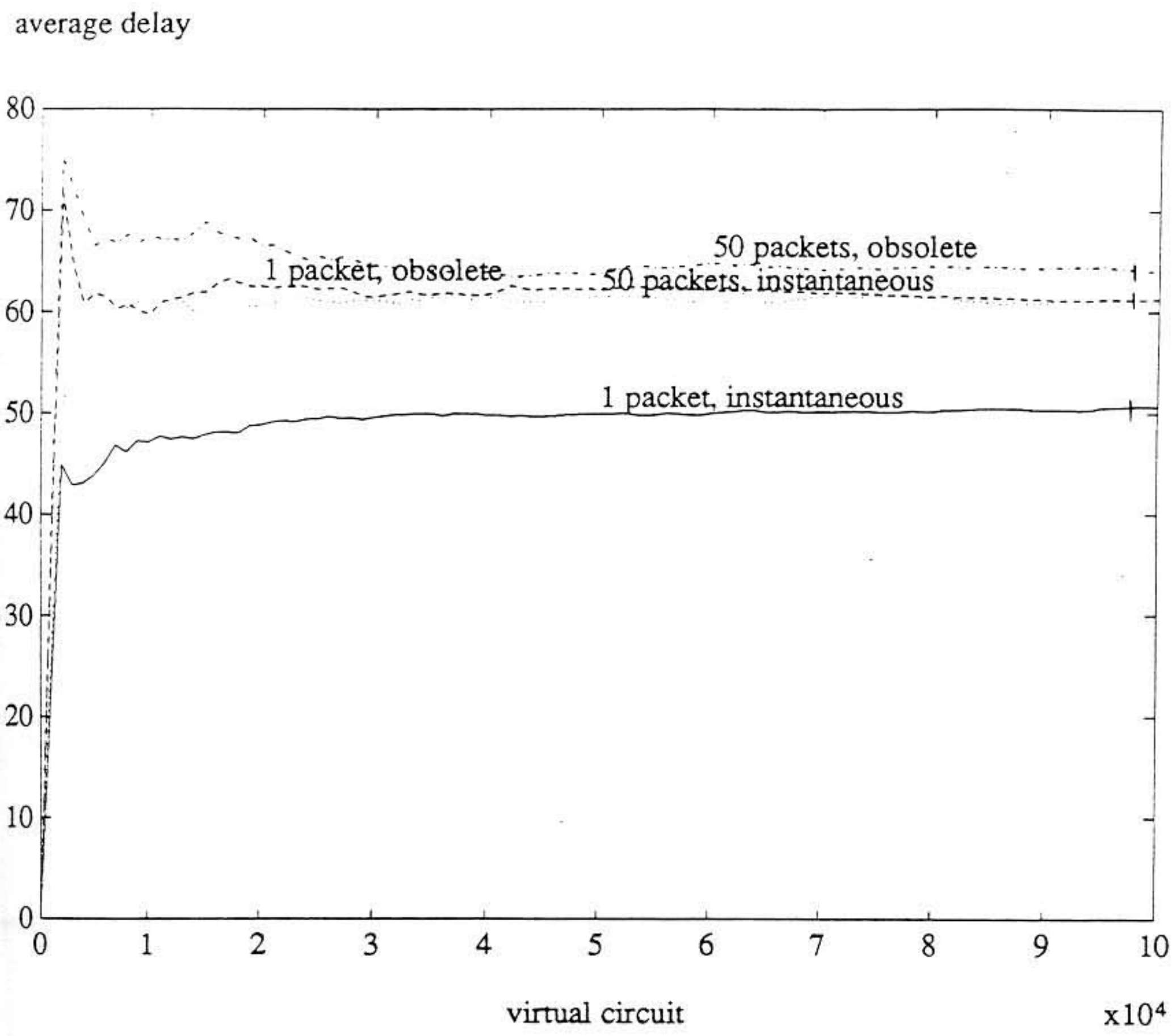


Figure 6.5: The average packet delay $\pm$ error (95% confidence interval) for the network of Figure 6.2 with $\gamma = 1/30$ for Multiple Response learning automaton based routing.

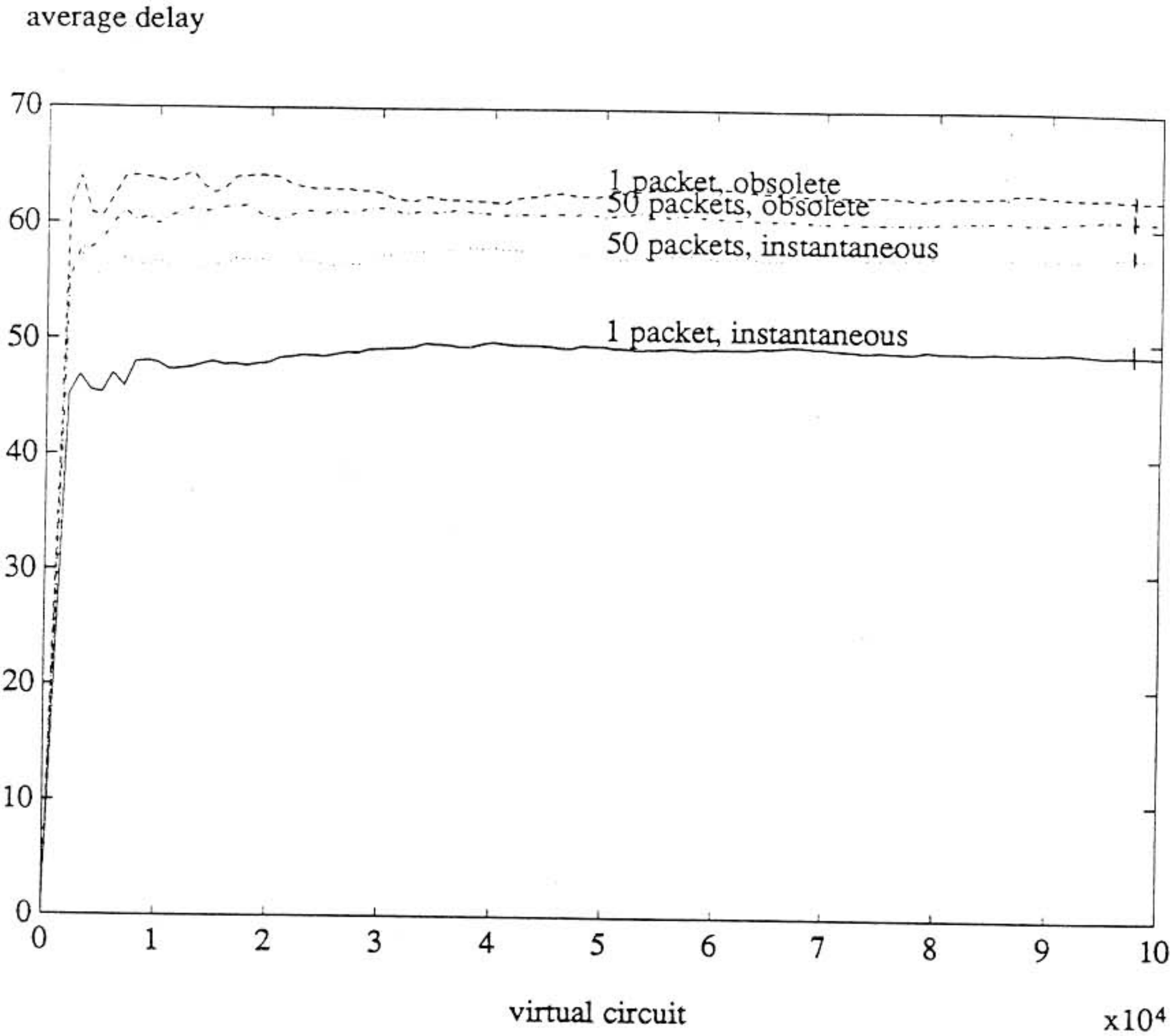


Figure 6.6: The average packet delay $\pm$ error (95% confidence interval) for the network of Figure 6.2 with $\gamma = 1/30$ for State Dependent learning automaton based routing.

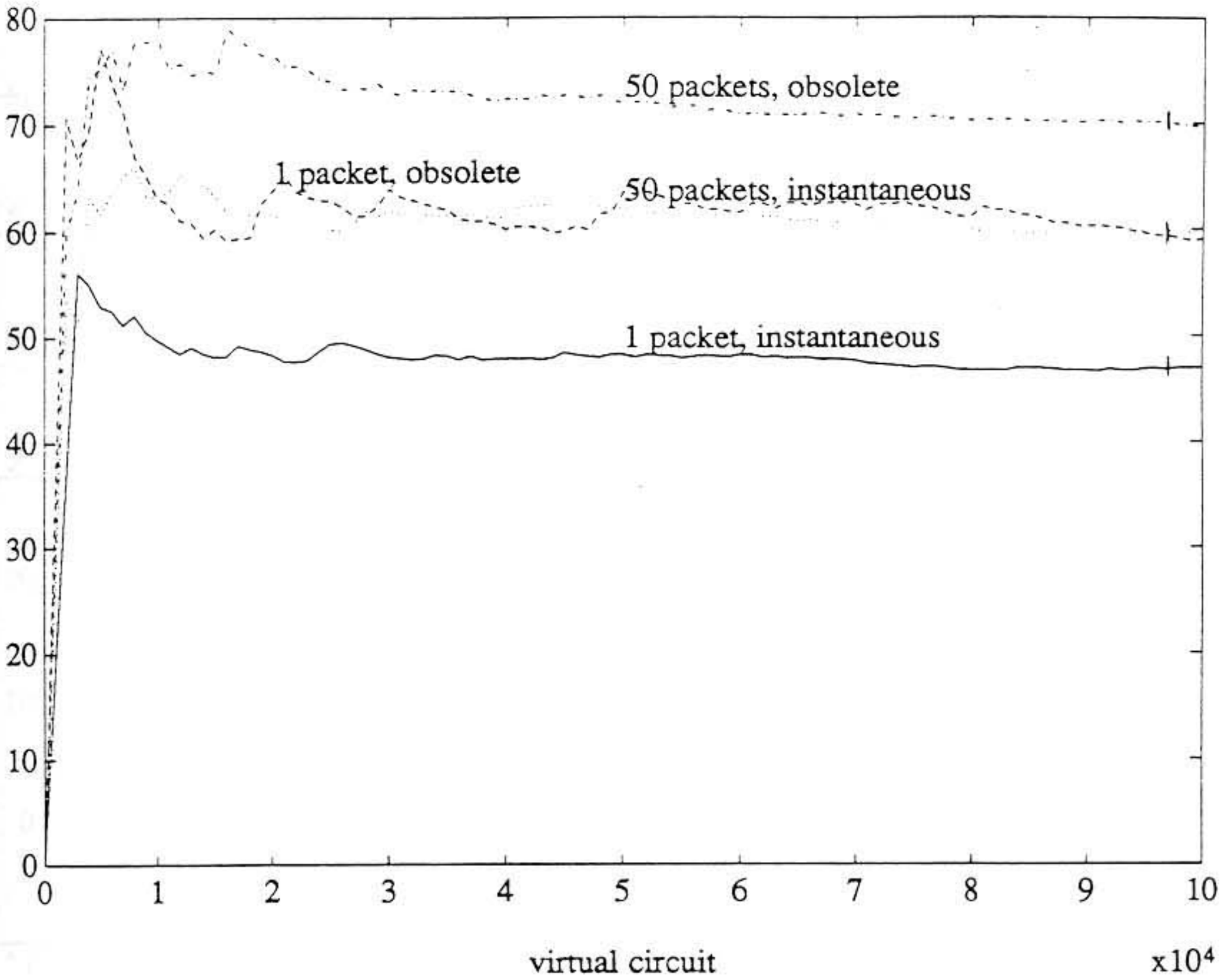


Figure 6.7: The average packet delay ± error (95% confidence interval) for the network of Figure 6.2 with $\gamma = 1/50$ for deterministic routing.

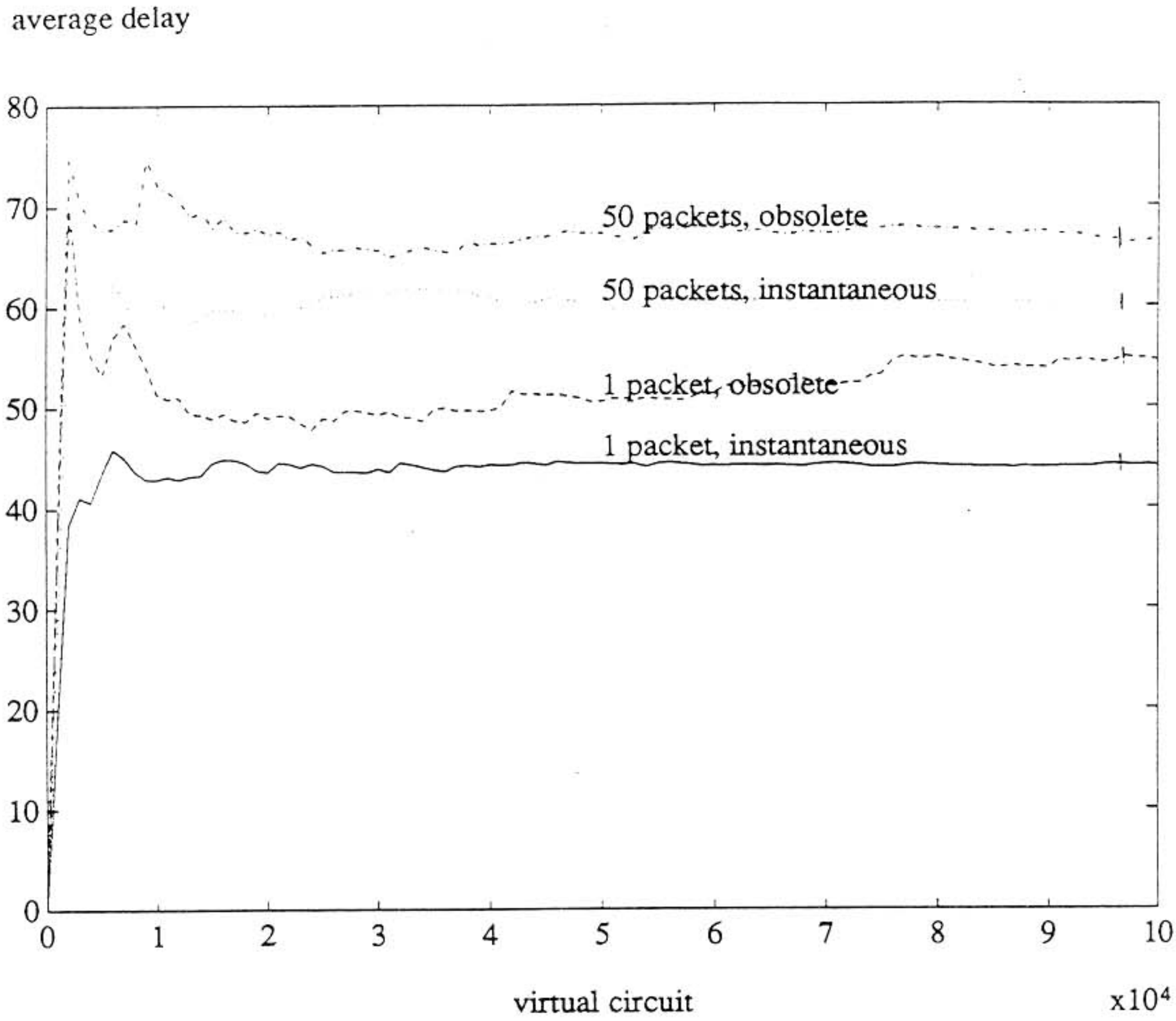


Figure 6.8: The average packet delay $\pm$ error (95% confidence interval) for the network of Figure 6.2 with $\gamma = 1/50$ for Linear learning automaton based routing.

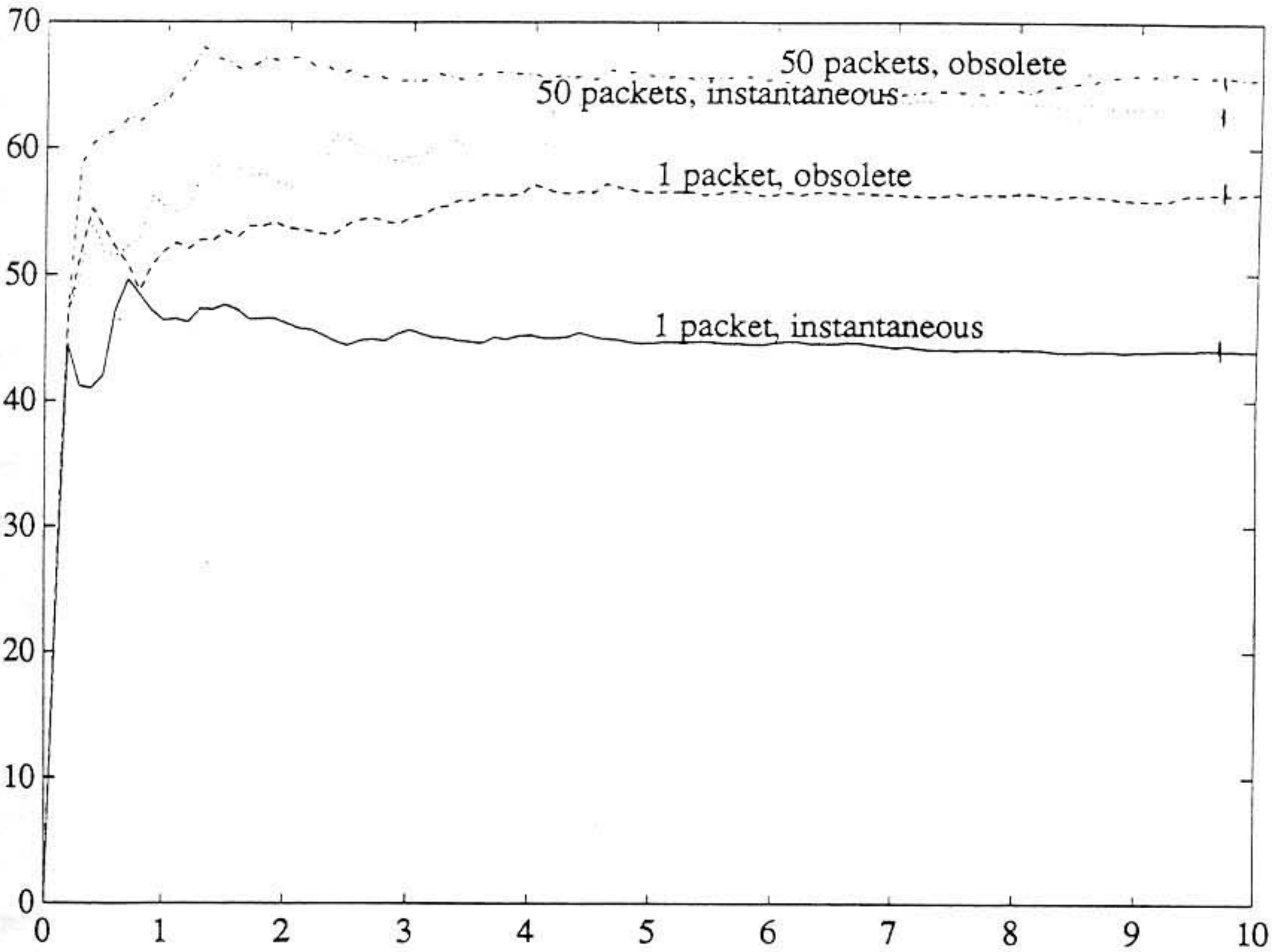


Figure 6.9: The average packet delay $\pm$ error (95% confidence interval) for the network of Figure 6.2 with $\gamma = 1/50$ for Multiple Response learning automaton based routing.

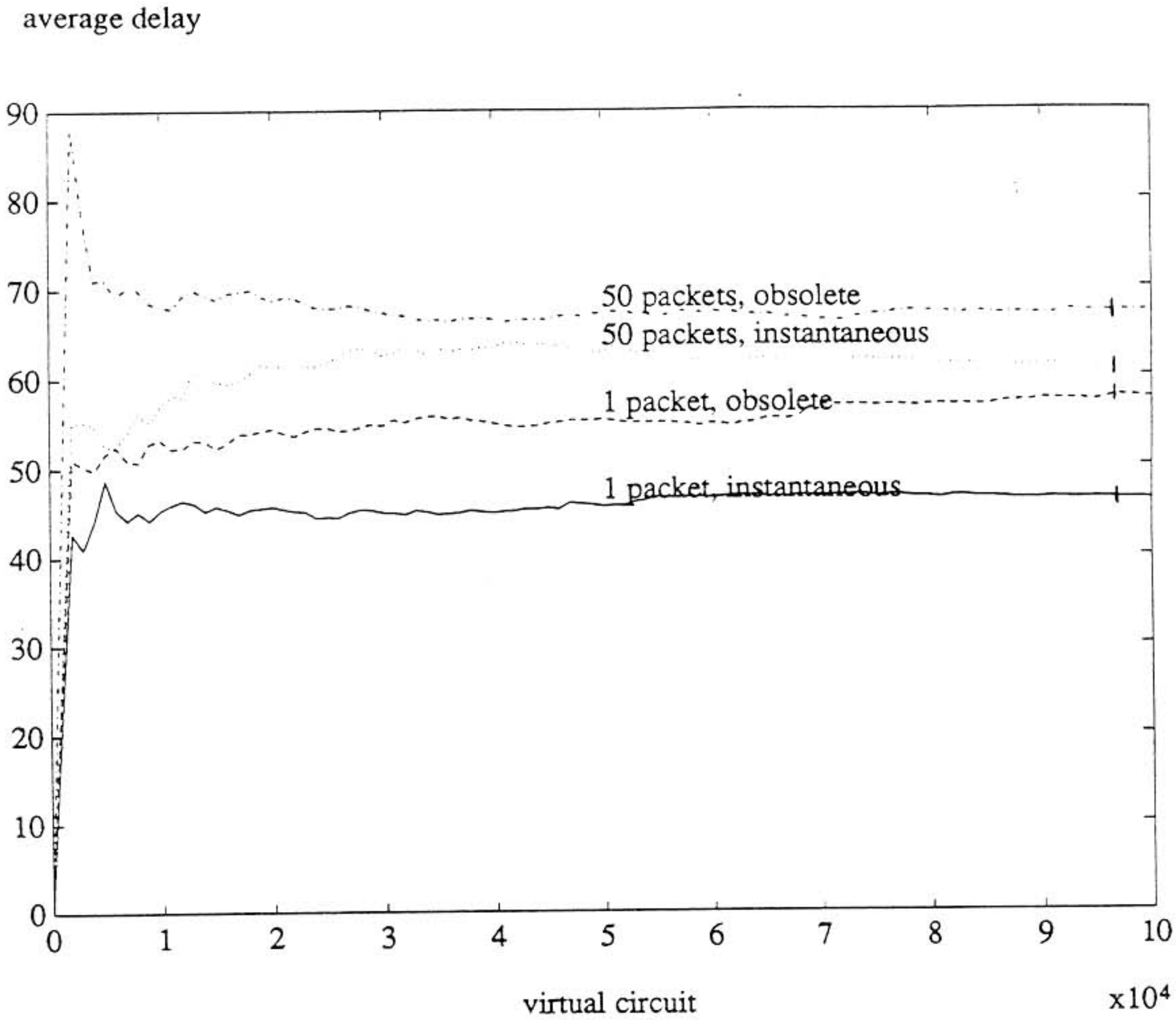


Figure 6.10: The average packet delay $\pm$ error (95% confidence interval) for the network of Figure 6.2 with $\gamma = 1/50$ for State Dependent learning automaton based routing.

6.8.2 Simulation Comparison of Performance Measures

In this section, we use the $L_{R-\epsilon P}$ algorithm. We compare a class of performance measures, that we introduced in section 5.5.3.

We consider as link length $l_{ij}(n)$ a convex combination of its current length $l_{ij}^{current}(n)$ and its future length $l_{ij}^{future}(n)$. We consider as current length $l_{ij}^{current}(n) = \frac{1+N_{ij}(n)}{\mu C_{ij}}$, a linear function of the number of packets on link ij. We consider as future length $l_{ij}^{current}(n) = \frac{1+V_{ij}(n)}{\mu C_{ij}}$, a linear function of the number of virtual circuits on link ij. Then the length of link ij is

$$l_{ij} = \epsilon * \frac{1+N_{ij}(n)}{\mu C_{ij}} + (1-\epsilon) * \frac{1+V_{ij}(n)}{\mu C_{ij}} \quad 0 \leq \epsilon \leq 1$$

A special case of this measure is the unfinished work [136]

$$U_{ij}(n) = \frac{1+N_{ij}(n)}{\mu C_{ij}} + \frac{r}{\delta} * \frac{1+V_{ij}(n)}{\mu C_{ij}}$$

The length of a path $\pi[sd]$ is

$$l_{\pi[sd]}(n) = \sum_{ij} l_{ij}(n) * 1_{ij \in \pi[sd]}(n)$$

Next, we investigate the effect of the parameter ϵ on the average packet delay.

The routing decisions are done by a $L_{R-\epsilon P}$ algorithm with reward parameter $\alpha = 0.2$ and penalty parameter $\beta = 0.8$: If the selected path has the minimum packet delay at the next iteration, then we increase the probability of selecting it again, otherwise we decrease it.

Let path $\pi[sd]$ is selected at time n

$$If\ l_{\pi[sd]}(n) = \min_{p[sd] \in \Pi_{[sd]}} \{l_{p[sd]}(n)\},\ then$$

$$P_{\pi[sd]}(n+1) = P_{\pi[sd]}(n) + 0.2 * [1 - P_{\pi[sd]}(n)]$$

$$P_{p[sd]}(n+1) = P_{p[sd]}(n) - 0.2 * P_{p[sd]}(n) \qquad \forall\ p[sd] \neq \pi[sd]$$

$$else$$

$$P_{\pi[sd]}(n+1) = P_{\pi[sd]}(n) - 0.8 * P_{\pi[sd]}(n)$$

$$P_{p[sd]}(n+1) = P_{p[sd]}(n) + 0.8 * \left[1 - P_{p[sd]}(n)\right] \qquad \forall\ p[sd] \neq \pi[sd]$$

We consider the same network as that of the previous section The mean packet service requirement is $1/\mu = 1$ and therefore $\mu_{ij} = \mu * C_{ij} = C_{ij}$. The total packet arrival rate is $r * \gamma/\delta = 4/5$. Two cases that achieve this rate are the following:

i) 5/50/200: the virtual circuit arrival rate is $\gamma = 1/5$, the packet arrival rate per virtual circuit is $r = 1/50$ and the mean virtual circuit duration is $\delta = 200$.

ii) 50/5/200: the virtual circuit arrival rate is $\gamma = 1/50$, the packet arrival rate per virtual circuit is $r = 1/5$ and the mean virtual circuit duration is $\delta = 200$.

For measuring the path length, we consider two cases:

i) 1 : the current number of packets at each link is sent to the source at every packet departure from that link.

ii) 50 : the average number of packets at each link during the last 50 time units is sent to the source at every 50th packet departure from that link.

The source node keeps and updates the information about the delay of its paths to the destination. The information about the delay of a path is updated every time a packet arrives at the destination through this path. However, this updating is not done immediately, but we assume a feedback delay so that this information becomes available to the source node. We assume that no extra traffic is created for th transferring this feedback information to the source node (it is either piggybacked on regular packets or uses a different channel). We consider two cases:

i) *instantaneous* information, when the feedback delay is 7 time units. In this case, we assume that the feedback information has higher priority over other packets and does not wait in queues.

ii) *obsolete* information, when the feedback delay is 60 time units. In this case, we assume that the feedback information is piggybacked on regular packets and is transferred back to the source node.

In Figure 6.11, 6.12 and Table 6.2, we show the simulation results for the average packet delay for 10,000 virtual circuits.

We notice that a proper value for the parameter ϵ should be experimentally selected for best performance. Using only the number of virtual circuits on each link ($\epsilon = 0$) as the link length is very inefficient (actually, for the case 5/50/200,

5/50/200	1 instant	1 obsolete	50 instant	50 obsolete
$\epsilon = 0.2$	104.22 $\pm$4.50	102.20 $\pm$5.51	133.30 $\pm$5.98	129.71 $\pm$4.92
$\epsilon = 0.4$	59.61 $\pm$3.31	59.97 $\pm$3.06	78.49 $\pm$2.75	73.94 $\pm$2.38
$\epsilon = 0.6$	46.98 $\pm$2.43	46.12 $\pm$1.79	60.88 $\pm$1.67	56.81 $\pm$1.53
$\epsilon = 0.8$	39.77 $\pm$1.05	42.68 $\pm$1.25	64.12 $\pm$2.05	77.94 $\pm$3.33
$\epsilon = 1$	37.19 $\pm$1.22	50.66 $\pm$2.06	104.38 $\pm$4.36	126.66 $\pm$4.45
delay	55.97 $\pm$3.98	97.02 $\pm$8.79	106.41 $\pm$8.03	121.67 $\pm$8.15

50/5/200	1 instant	1 obsolete	50 instant	50 obsolete
$\epsilon = 0$	73.01 $\pm$3.89	69.24 $\pm$5.15	125.73 $\pm$13.88	100.86 $\pm$7.56
$\epsilon = 0.2$	36.70 $\pm$0.98	37.20 $\pm$0.83	51.43 $\pm$1.66	51.50 $\pm$1.77
$\epsilon = 0.4$	34.39 $\pm$1.05	37.23 $\pm$1.42	64.44 $\pm$1.69	68.90 $\pm$1.71
$\epsilon = 0.6$	34.85 $\pm$1.05	39.41 $\pm$1.10	76.96 $\pm$1.50	85.60 $\pm$1.10
$\epsilon = 0.8$	35.29 $\pm$0.88	41.59 $\pm$1.11	83.39 $\pm$1.19	92.28 $\pm$2.13
$\epsilon = 1$	37.02 $\pm$1.02	44.15 $\pm$0.99	86.26 $\pm$2.34	97.26 $\pm$3.46
delay	45.35 $\pm$1.45	54.85 $\pm$2.31	61.43 $\pm$1.76	65.43 $\pm$1.77

Table 6.2: The average packet delay $\pm$ error (95% confidence interval) for the network of Figure 6.2 for different values of the parameter ϵ, when we use as link length $l_{ij} = \epsilon * \frac{1+N_{ij}}{} + (1-\epsilon) * \frac{1+V_{ij}}{}$.

the average network delay becomes extremely high and we do not even show it). Also, it is not always best to use only the number of packets on each link ($\epsilon = 1$) as the link length.

For comparison, we also show the average network delay, when we use the path delay as path length. It seems that using both the number of packets and virtual circuits as path length is much better than using the path delay.

The more frequent we update the algorithms and the more recent state information we have, the better the performance. Note also, that although the traffic characteristics 5/50/200 and 50/5/200 have the same packet arrival rate, the overall average packet delay is different. Traffic 5/50/200 has higher average packet delay than traffic 50/5/200, because the virtual circuits are arriving more

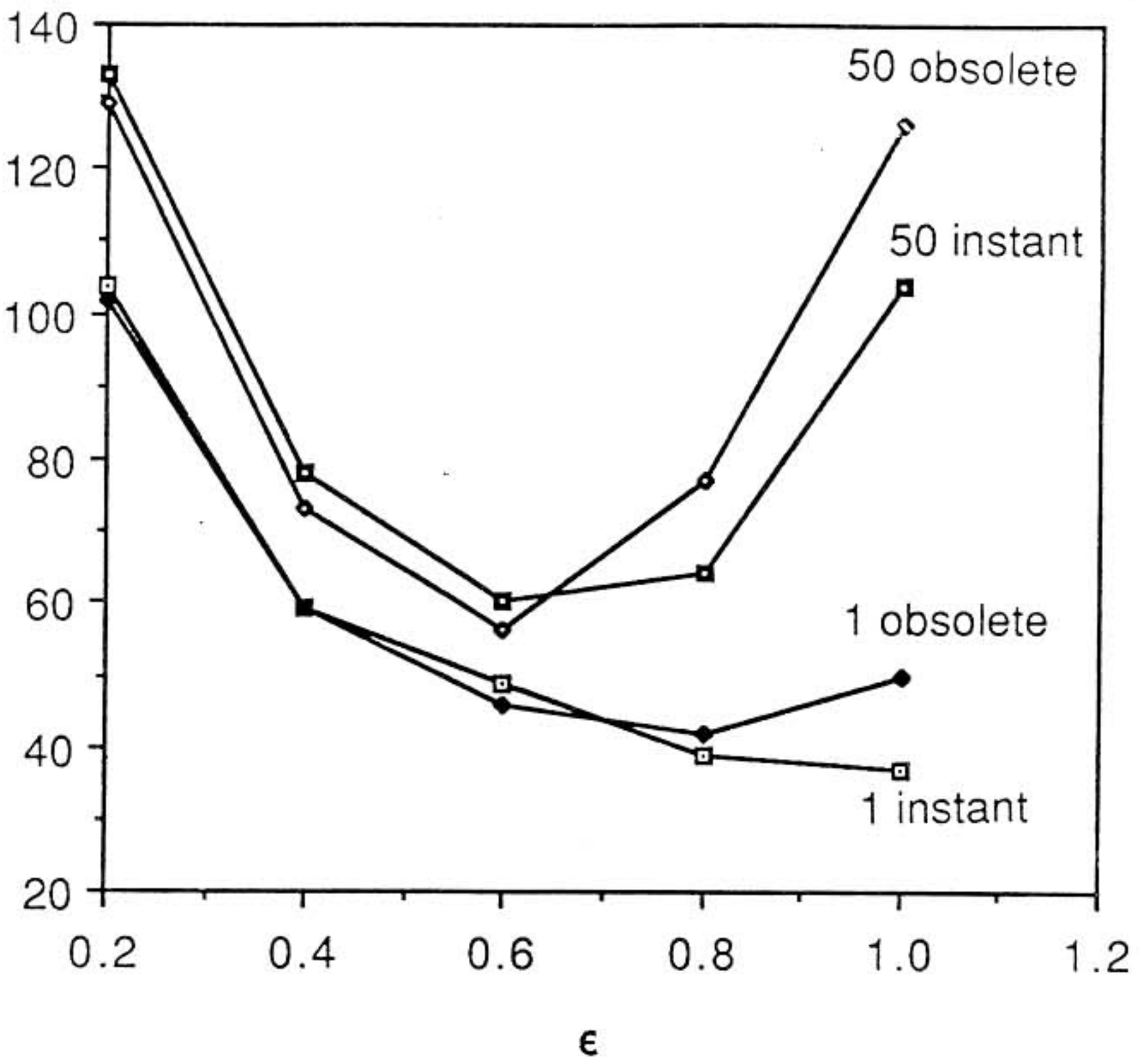


Figure 6.11: The average packet delay for the network of Figure 6.2 with $\gamma = 1/5$, $r = 1/50$, $\delta = 1/200$, for different values of the parameter ϵ, when we use as link length $l_{ij} = \epsilon * \frac{1 + N_{ij}}{ij} + (1 - \epsilon) * \frac{1 + V_{ij}}{ij}$.

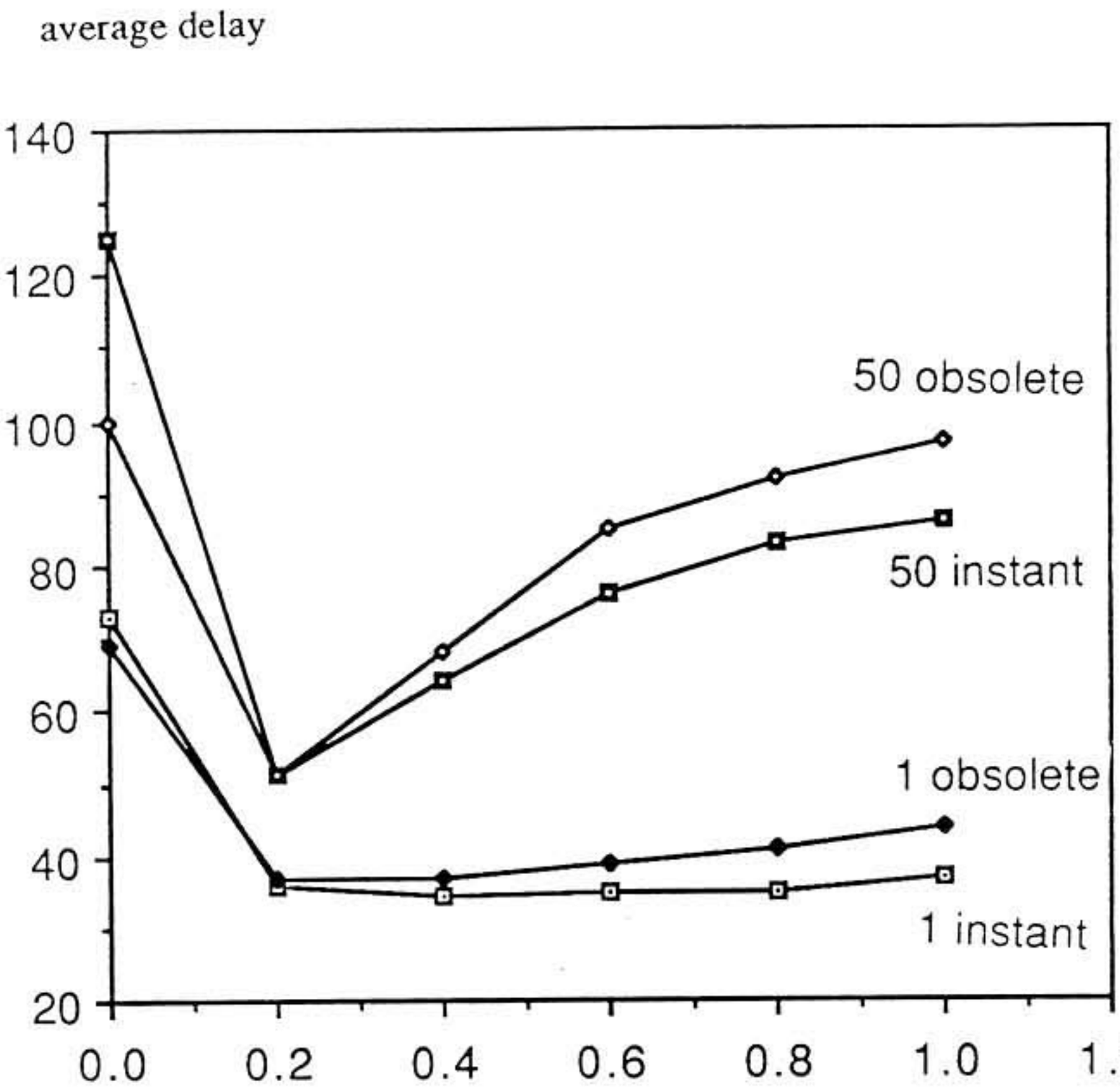


Figure 6.12: The average packet delay for the network of Figure 6.2 with $\gamma = 1/50$, $r = 1/5$, $\delta = 1/200$, for different values of the parameter ϵ, when we use as link length $l_{ij} = \epsilon * \frac{1 + N_{ij}}{ij} + (1 - \epsilon) * \frac{1 + V_{ij}}{ij}$.

frequently and therefore the network state changes more quickly. So, the state information that we use in the routing decisions is out-of-date.

6.9 Application to Integrated Services Networks

In this section, we make load sharing, routing and congestion control decisions in integrated services networks using stochastic learning automata.

The methology that we propose for learning automata-based load sharing, routing and congestion control decisions for new arriving virtual circuits (in connection-oriented ISN's) or new arriving packets (in connectionless-oriented ISN's) is similar to that of the previous section. So, we do not reiterate it here.

The only difference will be that the cost functions for each class in ISN's are different than those in virtual circuit networks. Therefore, the information that will be needed in order to calculate the lengths to destinations and the path lengths will be different than that for virtual circuit networks. For example, one class may use its blocking experience, while another class may use its packet delay to update its routing probability.

Chapter 7

Conclusions & Suggestions for Future Research

7.1 Conclusions

The major contribution of this dissertation is the introduction of a unified game-theoretic methodology for the multi-objective joint load sharing, routing and congestion control problem in distributed systems. And the introduction of stochastic learning automata algorithms for decentralized asynchronous computation of the solution.

We develop a novel mathematical approach, based on game theory, for the decentralized quasi-static and dynamic problem. After defining the joint problem, we model the distributed system on the path flow space using queueing and state space models. Then we develop three methodologies for both the quasi-static and the dynamic cases of the problem: i) *Team optimization methodology*, when the classes of jobs cooperate for the socially optimum, ii) *Nash game methodology*, when the classes of jobs compete among themselves and each class try to operate optimally for its own jobs and iii) *Stackelberg game methodology*, when some classes of jobs have more power than others, for example priority classes.

For each methodology, we formulate the problem as a *Nonlinear Programming or Optimal Control/Dynamic Programming, a Nonlinear Complementarity and a Variational Inequality problem.* We state conditions for existence/uniqueness of the solution and derive the optimality conditions for the quasi-static problem using

Karush-Kuhn-Tucker theorem, and for the dynamic problem using Pontryagin's maximum principle.

We apply the proposed methodologies to Datagram, Virtual Circuit and Integrated Services Networks and develop several new queueing models and performance measures, for each network type. We explicitly solve several examples and evaluate the system performance via simulation.

Finally, we introduce new classes of Stochastic Learning Automata algorithms and propose decentralized dynamic load sharing, routing and congestion control using Stochastic Learning Automata. Simulation is used to demonstrate improved system performance.

A variety of resource sharing problems arising in distributed systems may be formulated and solved using the proposed methodologies.

7.2 Suggestions for Future Research

Applications. We have presented several applications of the proposed methodologies and formulations that we have introduced in this dissertation. Obviously, we have not covered all possible applications. Thus, there is a huge research area to apply the proposed methodology. For example, by considering a specific network type (e.g. deterministic arrival and service distributions in ATM networks, threshold buffer management schemes, aging/deadline priorities, etc.), we may have different cost functions. Then selecting the appropriate scenario (cooperation, competition or hierarchy), we may formulate the problem as a team, Nash or Stackelberg game. Then we may choose either to solve the problem as a Nonlinear Programming, a Nonlinear Complementarity or a Variational Inequality Problem using appropriate algorithms.

Algorithms. We have developed several different formulations of the joint problem and suggested the use of iterative algorithms that solve the specific formulations. We have also introduced and tested via simulation one class of such decentralized, asynchronous dynamic algorithms, called stochastic learning automata.

Algorithms for solving cooperative or non-cooperative game problems are iterative algorithms that use first and possibly second derivatives. According to the iteration scheme, they can be classified as Gauss-Seidel, Successive Overrelaxation and Jacobi iteration algorithms. Instead of reiterating the existing bibliography on such algorithms, we rather refer to the original papers or books.

i) Nonlinear Programming algorithms: [152, 529, 339, 192, 30, 164, 165, 311, 387, 46],

ii) Optimal Control algorithms: [14, 292, 381, 415, 325, 131, 254, 412, 203, 440, 262, 301],

iii) Dynamic Programming algorithms: [220, 406, 45],

iv) Nash Games algorithms: [405, 429, 312, 175, 302, 110],

v) Stackelberg Games algorithms: [382, 54, 372, 371, 373, 442],

vi) Nonlinear Complementarity Problem algorithms: [113],

vii) Variational Inequalities algorithms: [198, 217].

Incentives. In this dissertation, we have assumed that the players either cooperate or compete for the resources. However, through the use of incentives, we can alter the scenario of the game and force the players to follow specific strategies.

Stochastic Discrete-Time. In chapter 5, we solved the dynamic deterministic optimal control problem, since we described the system state by the expected values of the stochastic processes. A direction for future research, is the solution of the stochastic problem either in continuous or discrete-time.

Hierarchical Games. In the Stackelberg game formulation, we considered two hierarchical levels, where at the upper level is the most powerful (e.g. higher priority) class of jobs and at the lower level (e.g. lower priority) is the less powerful class of jobs. One may extend these two levels to multiple hierarchical levels, where at each level there will be multiple classes. Then at each level the classes will play a Nash game, while classes among different levels will act as leaders and followers (Stackelberg game).

State Constraints. In chapter 5, we have introduced several possible constraints on the system state. However, we have not explicitly included them into the

solution of the optimization problems. One may extend this research by explicitly solving the dynamic problem with state constraints.

Information Structure. In solving the dynamic problem, we have assumed that the current network state is known. One area for future research is to solve the dynamic problem with delayed information about the network state.

State Observation Structure. In solving the dynamic problem, we have assumed perfect information about the network state. Another area for future research is to solve the dynamic problem with imperfect state observation.

Reference List

[1] H.Z. Aashtiani and T.L. Magnanti. Equilibria on a congested transportation network. *SIAM J. ALG. DISC. METH.*, Vol. 2, No. 3, pp. 213-226, Sept. 1981.

[2] C.E. Agnew. On the optimality of adaptive routing algorithms. *Proc. National Telecommunications Conference*, pp. 1021-1025, IEEE 1974.

[3] C.E. Agnew. On quadratic adaptive routing. *Communications of ACM*, Vol. 19, No. 1, pp. 18-22, Jan. 1976.

[4] C.E. Agnew. Dynamic modeling and control of congestion-prone systems. *Operations Research*, Vol. 24, No. 3, pp. 400-419, May-June 1976.

[5] A.K. Agrawala and S.K. Tripathi. On the optimality of semidynamic routing schemes. *Information Processing Letters*, Vol. 13, No. 1, pp. 20-22, Oct. 1981.

[6] A.K. Agrawala, S.K. Tripathi, and G. Ricart. Adaptive routing using a virtual waiting time technique. *IEEE Transactions on Software Engineering*, Vol. SE-8, No. 1, pp. 76-81, Jan. 1981.

[7] M. Aicardi, F. Davoli, and R. Minciardi. Decentralized dynamic routing as a Markov chain optimization problem. *Proc. Conference on Decision and Control*, pp. 1514-1517, IEEE 1987.

[8] V.M. Alekseev, V.M. Tikhomirov, and S.V. Formin. *Optimal Control.* Consultants Bureau, 1987.

[9] C.D. Aliprantis, D.J. Brown, and O. Burkinshaw. *Existence and Optimality of Competitive Equilibria.* Springer-Verlag, 1989.

[10] A.O. Allen. *Probability, Statistics, and Queueing Theory with Computer Science Applications.* Academic Press, 1978.

[11] K.J. Arrow and F.W. Hahn. *General Competitive Analysis.* Holden-Day Inc., 1971.

[12] K.J. Arrow and M. Kurz. *Public Investment, The Rate of Return And Optimal Fiscal Policy.* The JohnsHopkins Press, 1970.

[13] H. Aso and M. Kimura. Absolutely expediency of learning automata. *Informations Sciences*, Vol. 17, pp. 91-112, 1979.

[14] M. Athans and P.L. Falb. *Optimal Control.* Mc Graw-Hill, 1966.

[15] R.C. Atkinson and G.H. Bower. *An Introduction to Mathematical Learning Theory.* J.Wiley & Sons, 1965.

[16] J.-P Aubin. *Mathematical Methods of Game and Economic Theory.* North Holland Publ. Co., 1979.

[17] R.J. Aumann. *Lectures on Game Theory.* Westview Press, Inc., 1989.

[18] A. Avritzer, M. Gerla, B.A.N. Ribeiro, J.W. Carlyle, and W.J. Karplus. The advantage of dynamic tuning in distributed asymmetric systems. *Proc. Infocom 90 Conference*, pp. 811-818, IEEE 1990.

[19] N. Baba. *New Topics in Learning Automata Theory and Applications.* Springer-Verlag, 1984.

[20] K. Bala, I. Cidon, and K. Sohraby. Congestion control for high speed packet switched networks. *Proc. Infocom 90 Conference*, pp. 520-526, IEEE 1990.

[21] S.A. Banawan and J. Zahorjan. Load sharing in heterogeneous queueing systems. *Proc. Infocom 89 Conference*, pp. 731-739, IEEE 1989.

[22] A.G. Barto and P. Anandan. Pattern-recognized stochastic learning automata. *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. SMC-15, No. 3, pp. 360-375, MAy/June 1985.

[23] A.G. Barto, R.S. Sutton, and C.W. Anderson. Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. SMC-13, No. 5, pp. 834-846, Sept./Oct. 1983.

[24] A.G. Barto, R.S. Sutton, and P.S. Brouwer. Associative search networks: a reinforcement learning associative memory. *Biological Cybernetics*, 40, pp. 201-211, 1081.

[25] T. Basar. Equilibrium strategies in dynamic games with multi-levels of hierarchy. *Automatica*, pp. 749-754, 1981.

[26] T. Basar and S. Li. Distributed computation of Nash equilibria in linear-quadratic stochastic differential games. *SIAM J. Control and Optimization*, Vol. 27, No. 3, pp. 563-578, May 1989.

[27] T. Basar and G.J. Olsder. *Dynamic Noncooperative Game Theory*. Academic Press, 1982.

[28] T. Basar and H. Selbuz. A new approach for derivation of closed-loop Stackelberg strategies. *Proc. Conference on Decision and Control*, pp. 1113-1118, IEEE 1978.

[29] T. Basar and H. Selbuz. Closed-loop Stackelberg strategies with applications in the optimal control of multilevel systems. *IEEE Transactions on Automatic Control*, Vol. AC-24, No. 2, pp. 166-178, April 1979.

[30] M.S. Bazaraa and C.M. Shetty. *Nonlinear Programming: Theory and Algorithms*. J.Wiley & Sons, 1979.

[31] R. Bellman. *Dynamic Programming*. Princeton Univ. Press, 1957.

[32] A.W. Berger. Performance analysis of a rate control throttle where tokens and jobs queue. *Proc. Infocom 90 Conference*, pp. 30-38, IEEE 1990.

[33] L.D. Berkovitz. *Optimal Control Theory*. Springer-Verlag, 1974.

[34] L.D. Berkovitz. Necessary conditions for optimal strategies in a class of differential games and control problems. *J. SIAM Control*, Vol. 5, No. 1, pp. 1-24, 1967.

[35] F. Bernabei, C. Calabro, and M. Listanti. A fully distributed routing control scheme in an ATM switch. *Proc. International Communications Conference*, pp. 766-770, IEEE 1990.

[36] D. Bertsekas and S. Shreve. *Stochastic Optimal Control : the Discrete Time Case*. Academic Press, 1978.

[37] D.P. Bertsekas. *Dynamic Programming: Deterministic and Stochastic Models*. Prentice-Hall, Inc., 1987.

[38] D.P. Bertsekas. Dynamic models of shortest path routing algorithms for communication networks with multiple destinations. *Proc. Conference on Decision and Control*, pp. 127-133, IEEE 1979.

[39] D.P. Bertsekas. Optimal routing and flow control methods for communication networks. *Proc. 5th International Conference on Analysis and Optimization of Systems*, pp. 615-643, 1982.

[40] D.P. Bertsekas. A class of optimal routing algorithms for communication networks. *Proc. 5th Int. Conference on Computer Communications*, pp. 71-75, 1980.

[41] D.P. Bertsekas. Algorithms for nonlinear multicommodity network flow problems. *Proc. Int. Symposium on Systems Analysis and Optimization*, Springer-Verlag 1979.

[42] D.P. Bertsekas. Dynamic behavior of shortest path routing algorithms for communication networks. *IEEE Transactions on Automatic Control*, Vol. AC-27, No. 1, pp. 60-74, Febr. 1982.

[43] D.P. Bertsekas and E.M. Gafni. Projected Newton methods and optimization of multicommodity flows. *IEEE Transactions on Automatic Control*, Vol. AC-28, No. 12, pp. 1090-1096, Dec. 1983.

[44] D.P. Bertsekas, E.M. Gafni, and R.G. Gallager. Second derivative algorithms for minimum delay distributed routing in networks. *IEEE Transactions on Communications*, Vol. COM-32, No. 8, pp. 911-919, Aug. 1984.

[45] D.P. Bertsekas and R. Gallager. *Data Networks.* Prentice Hall, 1987.

[46] D.P. Bertsekas and J.N. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods.* Prentice-Hall, Inc., 1989.

[47] F.J. Beutler and D. Teneketzis. Optimal routing in queueing networks under imperfect information : stochastic dominance, thresholds and convexity. *ORSA/TIMS Joint National Meeting*, 1988.

[48] F.J. Beutler and D. Teneketzis. Routing in queueing networks under imperfect information: stochastic dominance and thresholds. *Stochastics and Stochastics Reports*, Vol. 26, pp. 81-100, 1989.

[49] K. Bharath-Kumar. Optimum end-to-end flow control in networks. *Proc. Int. Communications Conference*, pp. 23.3.1-23.3.6, IEEE 1980.

[50] K. Bharath-Kumar and J.M. Jaffe. A new approach to performance-oriented flow control. *IEEE Transactions on Communications*, Vol. COM-29, No. 4, pp. 427-435, July 1981.

[51] W.F. Bialas and M.H. Karwan. Multilevel optimization: a mathematical programming perspective. *Proc. Conference on Decision and Control*, pp. 761-765, IEEE 1980.

[52] W.F. Bialas and M.H. Karwan. On two-level optimization. *IEEE Transactions on Automatic Control*, Vol. AC-27, No. 1, pp. 211-214, Febr. 1982.

[53] J.P.C. Blanc. A note on waiting times in systems with queues in parallel. *J. Applied Probability*, Vol. 24, pp. 540-546, 1987.

[54] J.W. Blankenship and J.E. Falk. Infinitely constrained optimization problems. *J. of Optimization Theory and Applications*, Vol. 19, No. 2, pp. 261-281, June 1976.

[55] A. Blaquiere, F. Gerard, and G. Leitmann. *Quantitative and Qualitative Games.* Academic Press, 1969.

[56] B.W. Boehm and R.L. Mobley. Adaptive routing techniques for distributed communications systems. *IEEE Transactions on Communications*, Vol. COM-17, No. 3, pp. 340-349, June 1969.

[57] R.K. Boel and J.H.van Schuppen. Distributed routing for load balancing. *Proceedings of the IEEE*, Vol. 77, No. 1, pp. 210-221, Jan. 1989.

[58] V.G. Boltyanskii. *Mathematical Methods of Optimal Control.* Holt, Rinehart, Winston Inc., 1971.

[59] V.G. Boltyanskii. *Optimal Control of Discrete Systems.* J. Wiley & Sons, 1978.

[60] F. Bonomi. Performance analysis of some process-to-processor assignment algorithms for a UNIX multiprocessor system. *Distributed Processing*, M.H. Barton and E.L. Dagles and G.L. Reijns (eds.), pp. 491-504, Elsevier Science Publ., IFIP 1988.

[61] F. Bonomi. Adaptive optimal load balancing in a heterogeneous multiserver system with a central job scheduler. *8th Int, Conference on Distributed Computing Systems*, pp. 500-508, IEEE 1988.

[62] F. Bonomi. On job assignment for a parallel system of processor sharing queues. *IEEE Transactions on Computers*, Vol. 39, No.7, pp. 858-869, July 1990.

[63] F. Bonomi, P.J. Fleming, and P. Steinberg. An adaptive join-the biased-queue rule for load sharing on distributed computer systems. *Proc. 28th Conference on Decisions and Control*, pp. 2554-2559, IEEE 1988.

[64] F. Bonomi and A. Kumar. Optimality of weighted least squares load balancing. *Proc. 27th Conference on Decision and Control*, pp. 1480-1485, IEEE 1988.

[65] R.R. Boorstyn and A. Livne. A technique for adaptive routing in networks. *IEEE Transactions on Communications*, Vol. COM-29, No. 4, pp. 474-480, April 1981.

[66] K.C. Border. *Fixed Point Theorems with Applications to Economic and Game Theory*. Cambridge University Press, 1985.

[67] F. Borgonovo and E. Cadorin. Locally-optimal deflection routing in the Bidirectional Manhattan Network. *Proc. Infocom 90 Conference*, pp. 458-464, IEEE 1990.

[68] A.D. Bovopoulos. Resource allocation algorithms for packet switched networks. *Ph.D. Dissertation, Columbia University*, 1989.

[69] A.D. Bovopoulos. Resource allocation as a Nash game in a multiclass packet switched environment. *Computer Science Dept., Technical Report WUCS-89-18, pp. 1-9*, Washington University 1989.

[70] A.D. Bovopoulos and A.A. Lazar. Asynchronous algorithms for optimal flow control of BCMP networks. *Dept. of Computer Science, Washington University, WUCS-89-10*, pp. 1-22, 1989.

[71] A.D. Bovopoulos and A.A. Lazar. Decentralized network flow control. *Proc. Int. Computer Communications Conference*, pp. 139-143, 1988.

[72] A.D. Bovopoulos and A.A. Lazar. Optimal load balancing algorithms for Jacksonian networks with acknowledgement delay. *Proc. Computer Networking Symposium*, pp. 144-151, IEEE 1988.

[73] A.D. Bovopoulos and A.A. Lazar. Optimal routing and flow control of a network of parallel processors with individual buffers. *Proc. 23rd Annual Allerton Conference on Communications, Control, and Computing*, pp. 564-573, 1985.

[74] A.D. Bovopoulos and A.A. Lazar. Load balancing algorithms for Jacksonian networks with acknowledgement delays. *Proc. Infocom '89 Conference*, pp. 749-757, IEEE 1989.

[75] A.D. Bovopoulos and A.A. Lazar. Decentralized algorithms for optimal flow control. *25th Annual Allerton Conference on Communications, Control, and Computing*, pp. 979-988, 1987.

[76] J. Bracken and J.T. McGill. Mathematical programs with optimization problems in the constraints. *Operations Research*, Vol. 21, No. 1, pp. 37-44, 1973.

[77] S.C. Bruell and G. Balbo. *Computational Algorithms for Closed Queueing Networks*. North Holland, 1980.

[78] R. Bryant and R.A. Finkel. A stable distributed scheduling algorithm. *Proc. Second Int. Conf. on Distributed Computing Systems*, pp.314-323, April 1981.

[79] A.E.Jr. Bryson and Y. C. Ho. *Applied Optimal Control.* Hemisphere Publ. Co, 1975.

[80] R.R. Bush and F. Mosteller. *Stochastic Models and Learning.* J.Wiley & Sons, 1955.

[81] J.P. Buzen and P.P.S. Chen. Optimal load balancing in memory hierarchies. *Information Processing 74*, pp. 271-275, North Holland Publ. Co. 1974.

[82] D.H. Cansever. Decentralized algorithms for flow control in networks. *Proc. 25th Conference on Decision and Control*, pp. 2107-2112, IEEE 1986.

[83] D.G. Cantor and M. Gerla. Optimal routing in a packet-switched computer network. *IEEE Transactions on Computers*, Vol. C-23, No. 10, pp. 1062-1069, Oct. 1974.

[84] G. Casalino, F. Davoli, R. Minciardi, and R. Zoppoli. On the structure of decentralized dynamic routing strategies. *Proc. Conference on Decision and Control*, pp. 472-476, IEEE 1983.

[85] T.L. Casavant and J.G. Kuhl. Analysis of three dynamic distributed load-balancing strategies with varying global information requirements. *Proc. Int. Conf. on Distributed Computing Systems*, pp. 185-192, IEEE 1987.

[86] T.L. Casavant and J.G. Kuhl. Effects of response and stability on scheduling in distributed computing systems. *IEEE Transactions on Software Engineering*, Vol. 14, No. 11, pp. 1578-1588, Nov. 1988.

[87] J.H. Case. *Economics and the Competitive Process.* New York University Press, 1979.

[88] J.H. Case. Toward a theory of many player differential games. *SIAM J. Control*, Vol. 7, No. 2, pp. 179-197, May 1969.

[89] C.G. Cassandras, V.V. Abide, and D. Towsley. Distributed routing with on-line marginal delay estimation. *IEEE Transactions on Communications*, Vol. 38, No. 3, pp. 348-359, March 1990.

[90] C.G. Cassandras, M.H. Kallmes, and D. Towsley. Optimal routing and flow control in networks with real-time traffic. *Proc. Infocom 89 Conference*, pp. 784-791, IEEE 1989.

[91] D.A. Castanon and N.R.Jr. Sandel. On hierarchical Stackelberg optimization problems. *Proc. Conference on Decision and Control*, pp. 104-107, IEEE 1977.

[92] B. Chandrasekaran and D.W.C. Shen. On expedience and convergence in variable-structure automata. *IEEE Transactions on Systems Science and Cybernetics*, Vol. SSC-4, No. 1, pp. 52-60, March 1968.

[93] C.-J. Chang and J.-F. Chang. The effect of idle server first random routing in the behavior of a finite queue. *IEEE Transactions on Communications*, Vol. COM-35, No. 5, pp. 496-502, May 1987.

[94] F. Chang and L. Wu. An optimal adaptive routing algorithm. *IEEE Transactions on Automatic Control*, Vol. AC-31, No. 8, pp. 690-700, Aug. 1986.

[95] T.-S. Chang and P.B. Luh. Derivation of necessary and sufficient conditions for single-stage Stackelberg games via the inducible region concept. *IEEE Transactions on Automatic Control*, Vol. AC-29, No. 1, pp. 63-66, Jan. 1984.

[96] C.I. Chen and J.B.Jr. Cruz. Stackelberg solution for two-person games with biased information patterns. *IEEE Transactions on Automatic Control*, Vol. AC-17, No. 6, pp. 791-797, Dec. 1972.

[97] M.S. Chen and J.S. Meditch. A distributed adaptive routing algorithm. *Proc. International Communications Conference*, pp. B5.3.1-B5.3.9, IEEE 1983.

[98] L.A. Chenault. On the uniqueness of Nash equilibria. *Economic Letters*, Vol. 20, pp. 203-205, 1986.

[99] T.C.K. Chou and J.A. Abraham. Load balancing in distributed systems. *IEEE Trans. on Software Eng., Vol SE-8, No4*, pp. 401-412, July 1982.

[100] T.C.K. Chou and J.A. Abraham. Distributed control of computer systems. *IEEE Trans. on Computers, Vol C-35, No 6*, pp.564-567, June 1986.

[101] W. Chou, A.W. Bragg, and A.A. Nilsson. The need for adaptive routing in the chaotic and unbalanced traffic environment. *IEEE Transactions on Communications*, Vol. COM-29, No. 4, pp. 481-490, April 1981.

[102] T.-C. Chow and W.H. Kohler. Models for dynamic load balancing in a heterogeneous multiple processor system. *IEEE Trans. on Computers. Vol C-28, No 5*, pp. 354-361, May 1979.

[103] Y.C. Chow and W.H. Kohler. Dynamic load balancing in homogeneous two-processor distributed systems. *Int. Symposium on Computer Performance Modeling, Measurement and Evaluation*, pp. 39-52, August 1977.

[104] M.S. Chrystall and P. Mars. Adaptive routing in computer communication networks using learning automata. *Proc. of IEEE Nat. Telecomm. Conf.*, pp. A3.2.1-7, 1981.

[105] P.H.N. Chu, R.R. Boorstyn, and A. Kershenbaum. A simulation study of a dynamic routing scheme. *Proc. National Telecommunications Conference*, pp. A3.4.1-A3.4.11, IEEE 1981.

[106] W.W. Chu and M.Y. Shen. A hierarchical routing and flow control policy (HRFC) for packet switched networks. *Proc. Computer Performance Conference*, ACM 1977.

[107] W.W. Chu and M.Y. Shen. A hierarchical routing and flow control policy (HRFC) for packet switched networks. *IEEE Transactions on Computers*, Vol. C-29, No. 11, pp. 971-977, Nov. 1980.

[108] B. Ciciani, D.M. Dias, and P.S. Yu. Load sharing in hybrid distributed-centralized database systems. *Proc. 8th Int. Conference on Distributed Computing Systems*, pp. 274-281, IEEE 1988.

[109] G.M. Clark. Use of Polya distributions in approximate solutions to M/M/s queues. *Communications of the ACM*, Vol. 24, No. 4, pp. 206-217, April 1981.

[110] G. Cohen. Nash equilibria: gradient and decomposition algorithms. *Large Scale Systems*, Vol. 12, pp. 173-184, 1987.

[111] A.E. Conway and N.D. Georgana. *Queueing Networks - Exact Computational Algorithms.* The MIT Press, 1990.

[112] R.B. Cooper. *Introduction to Queueing Theory.* North Holland, 2nd ed. 1981.

[113] R.W. Cottle, F. Giannesi, and J.-L. Lions. *Variational Inequalities and Complementarity Problems: Theory and Applications.* J. Wiley & Sons, 1980.

[114] P.-J. Courtois and P. Semal. An algorithm for the optimization of nonbifurcated flows in computer communication networks. *Performance Evaluation*, pp. 139-152, 1981.

[115] P.J. Courtois. *Decomposability: Queueing and Computer System Applications.* Academic Press, 1977.

[116] D.R. Cox and W.L. Smith. *Queues.* Chapman and Hall, 1961.

[117] J.B.Jr. Cruz. Leader-follower strategies for multilevel systems. *IEEE Transactions on Automatic Control*, Vol. AC-23, No. 2, pp. 244-254, April 1978.

[118] S. Dafermos. Traffic equilibrium and variational inequalities. *Transportation Science*, Vol. 14, No. 1, pp. 42-54, Febr. 1980.

[119] S. Dafermos. An iterative scheme for variational inequalities. *Mathematical Programming*, Vol. 26, pp. 40-47, 1983.

[120] S.C. Dafermos and F.T. Sparrow. The traffic assignment problem for a general network. *Journal of Research of the National Bureau of Standards B.*, Mathematical Sciences, Vol. 73B, No. 2, pp.91-118, April-June 1969.

[121] B. Daneshrad and S.D. Morgera. Application of stochastic automaton theory for routing in a packet-switched network. *Proc. MILCOM*, pp. 205-209, IEEE 1989.

[122] J.M. Danskin. *The Theory of Max-Min.* Springer-Verlag, 1967.

[123] J.P. Dauer and W. Stadler. A survey of vector optimization in infinite-dimensional spaces, part 2. *J. of Optimization Theory and Applications*, Vol. 51, No. 2, pp. 205-241, Nov. 1986.

[124] E. de Souza e Silva and M. Gerla. Load balancing in distributed systems with multiple classes and site constraints. *Performance '84, E. Gelenbe (ed.)*, pp. 17-33, North Holland 1984.

[125] M. Decina and T. Toniatti. On bandwidth allocation to bursty virtual connections in ATM networks. *Proc. Int. Communications Conference*, pp. 844-850, IEEE 1990.

[126] M. Decina, T. Toniatti, P. Vaccari, and L. Verri. Bandwidth assignment and virtual call blocking in ATM networks. *Proc. Infocom 90 Conference*, pp. 881-888, IEEE 1990.

[127] R.L. Disney and P.C. Kiessler. *Traffic Processes in Queueing Networks: A Markov Renewal Approach.* The Johns Hopkins University Press, 1987.

[128] C. Douligeris and R. Mazumdar. A game theoretic approach to flow control in an integrated environment with two classes of users. *Proc. IEEE Computer Networking Symposium, pp. 214-221*, 1988.

[129] C. Douligeris and R. Mazumdar. More on Pareto optimal flow control. *Proc. 26th Annual Allerton Conference on Communication, Control, and Computing*, pp. 1181-1190, 1988.

[130] C. Douligeris and R. Mazumdar. On Pareto optimal flow control in a multiclass environment. *Proc. 25th Annual Allerton Conference on Communication, Control, and Computing*, pp. 989-997, 1987.

[131] P. Dyer and S.R. McReynolds. *The Computation and Theory of Optimal Control.* Academic Press, 1970.

[132] D.L. Eager, E.D. Lazowska, and J. Zahorjan. A comparison of receiver-initiated and sender-initiated adaptive load sharing. *ACM Sigmetrics Conf. on Measurement and Modeling of Computer Systems*, pp. 1-3, ACM 1985.

[133] D.L. Eager, E.D. Lazowska, and J. Zahorjan. The limiting performance benefits of migrating active processes for load sharing. *Proc. SIGMETRICS*, pp. 63-72, ACM 1988.

[134] D.L. Eager, E.D. Lazowska, and J. Zahorjan. Adaptive load sharing in homogeneous distributed systems. *IEEE Trans. on Software Eng., Vol SE-12, No 5*, pp. 662-675, May 1986.

[135] A.E.Jr. Eckberg, D.T. Luan, and D.M. Lucantoni. Meeting the challenge: Congestion and flow control strategies for Broadband information transport. *Proc. GLOBECOM 1989*, pp. 1769-1773, IEEE 1989.

[136] A.A. Economides, P.A. Ioannou, and J.A. Silvester. Decentralized adaptive routing for virtual circuit networks using stochastic learning automata. *Proc. of IEEE Infocom 88 Conference*, pp. 613-622, IEEE 1988.

[137] A.A. Economides, P.A. Ioannou, and J.A. Silvester. Adaptive routing and congestion control for window flow controlled virtual circuit networks. *Proc. 27th Annual Allerton Conference on Communications, Control, and Computing*, pp. 849-857, Sept. 1989.

[138] A.A. Economides and J.A. Silvester. Priority load sharing: an approach using Stackelberg games. *USC Technical Report CENG 89-39*, 1989.

[139] A.A. Economides and J.A. Silvester. Routing games. *USC Technical Report CENG 89-38*, 1989.

[140] A.A. Economides and J.A. Silvester. A game theory approach to cooperative and non-cooperative routing problems. *Proc. IEEE International Telecommunications Symposium*, Brazil, Sept. 3-6, 1990.

[141] A.A. Economides and J.A. Silvester. Optimal routing in a network with unreliable links. *Proc. of IEEE Computer Networking Symposium*, pp. 288-297, IEEE 1988.

[142] K. Efe and B. Groselj. Minimizing control overheads in adaptive load sharing. *Proc. 9th Int. Conference on Distributed Computing Systems*, pp. 307-315, IEEE 1989.

[143] Y.M. El-Fattah. *Learning Systems: Decisions, Simulation, and Control.* Springer-Verlag, 1978.

[144] A. Ephremides. Extension of an adaptive distributed routing algorithm to mixed media networks. *IEEE Transactions on Communications*, Vol. COM-26, No. 8, pp. 1262-1266, Aug. 1978.

[145] A. Ephremides, P. Varaiya, and J. Walrand. A simple dynamic routing problem. *IEEE Transactions on Automatic Control*, Vol. AC-25, No. 4, Aug. 1980.

[146] D. Ferguson, Y. Yemini, and C. Nikolaou. Microeconomic approach for load balancing in distributed computer systems. *Proc. 8th Int. Conference on Distributed Computing Systems*, pp. 491-499, IEEE 1988.

[147] D.F. Ferguson, C. Nikolaou, and Y. Yemini. An economy for flow control in computer networks. *Proc. Infocom '89 Conference*, pp. 110-118, IEEE 1989.

[148] D. Ferrari. A study of load indices for load balancing schemes. *Workload Characterization of Computer Systems*, G. Serazzi (ed.), pp. 91-99, Elsevier Science Publ. 1986.

[149] D. Ferrari, G. Serazzi, and A. Zeigner. *Measurement and Tuning of Computer Systems.* Prentice Hall, 1983.

[150] D. Ferrari and S. Zhou. An empirical investigation of load indices for load balancing algorithms. *Performance 87*, P.-J. Courtois and G. Latouche (eds), pp. 515-528, Elsevier Science Publ. 1988.

[151] D. Ferrari and S. Zhou. A load index for dynamic load balancing. *Proc. Fall Joint Computer Conference*, pp. 684-690, IEEE 1986.

[152] A.V. Fiacco and G.P. McCormick. *Nonlinear Programming: Sequential Unconstrained Minimization Techniques.* J. Wiley & Sons, 1968.

[153] J. Filipiak. *Modelling and Control of Dynamic Flows in Communication Networks.* Springer-Verlag, 1988.

[154] J. Filipiak. M-architecture: A structural model of traffic management and control in broadband ISDNs. *IEEE Communication Magazine*, pp. 25-31, May 1989.

[155] J. Filipiak. Analysis and synthesis of routing and flow control in a nonstationary environment. *Proc. of International Conference on Computer Communications*, pp. 446-451, 1985.

[156] J. Filipiak. Optimization of dynamic flows in networks of hierarchical structure. *Problems of Control and Information Theory*, Vol. 11, No. 3, pp. 155-165, 1982.

[157] J. Filipiak. Unloading of congestion in deterministic queueing networks. *Optimal Control Applications & Methods*, Vol. 2, pp. 23-45, 1981.

[158] J. Filipiak. Optimal control of store-and-forward networks. *Optimal Control Applications & Methods*, Vol. 3, pp. 155-176, 1982.

[159] J. Filipiak. Dynamic routing in queueing systems with a multiple service facility. *Operations Research*, Vol. 32, No. 5, pp. 1163-1180, 1984.

[160] J. Filipiak. Structured systems analysis methodology for desigh of an ATM network architecture. *Journal On Selected Areas In Communications*, Vol. JSAC-7, No. 8, pp. 1263-1273, Oct. 1989.

[161] L. Flatto and H.P. McKean. Two queues in parallel. *Communications on Pure and Applied Mathematics*, Vol. xxx, pp. 255-263, 1977.

[162] W.H. Fleming and R.W. Rishel. *Deterministic and Stochastic Optimal Control.* Springer-Verlag, 1975.

[163] Y.A. Flerov. Some classes of multi-input automata. *Journal of Cybernetics*, Vol. 2, No. 3, pp. 112-122, 1972.

[164] R. Fletcher. *Practical Methods of Optimization; Vol. 1: Unconstraint Optimization.* J.Wiley & Sons, 1980.

[165] R. Fletcher. *Practical Methods of Optimization; Vol. 2: Constraint Optimization.* J.Wiley & Sons, 1981.

[166] L.R. Ford and D.R. Fulkerson. *Flows in Networks.* Princeton University Press, 1962.

[167] G. Foschini and J. Salz. A basic dynamic routing problem and diffusion. *IEEE Transactions on Communications*, Vol. COM-26, No. 3, pp. 320-327, March 1978.

[168] G.J. Foschini. On heavy traffic diffusion analysis and dynamic routing in packet switched networks. *Computer Performance*, K.M.Chandy and M.Reiser (eds.), North Holland Publ. Co., pp. 499-513, 1977.

[169] H. Frank. Dynamic communication networks. *IEEE Transactions on Communication Technology*, Vol. COM-15, No. 2, pp. 156-163, April 1967.

[170] H. Frank and M.T. El-Bardai. Dynamic communication networks with capacity constraints. *IEEE Transactions on Communication Technology*, Vol. COM-17, No. 4, pp. 432-437, Aug. 1969.

[171] H. Frank and I.T. Frisch. *Communication, Transmission, and Transportation Networks.* Addison-Wesley, 1971.

[172] L. Fratta, M. Gerla, and L. Kleinrock. The flow deviation method : an approach to store-and-forward communication network design. *Networks, 3*, pp. 97-133, 1973.

[173] A. Friedman. *Differential Games.* J. Wiley & Sons, 1971.

[174] J.W. Friedman. *Oligopoly and the theory of games.* North Holland, 1977.

[175] D. Gabay and H. Moulin. On the uniqueness and stability of Nash-equilibria in noncooperative games. *Applied Stochastic Control in Econometrics and Management Science*, A.Bensoussan et al. (eds.), pp. 271-293, North Holland 1980.

[176] E.M. Gafni. and D.P. Bertsekas. Path assignment for virtual circuit routing. *Proc. ACM SIGCOMM 1983 Conference*, pp. 21-25, ACM 1983.

[177] E.M. Gafni and D.P. Bertsekas. Asymptotic optimality of shortest path routing algorithms. *IEEE Trans. on Information Theory*, Vol. IT-33, No. 1, pp. 83-90, Jan. 1987.

[178] R.G. Gallager. Distributed network optimization algorithms. *Proc. International Communications Conference*, pp. 43.2.1-43.2.2, IEEE 1979.

[179] R.G. Gallager. A minimum delay routing using distributed computation. *IEEE Trans. on Communications*, Vol. COM-25, No. 1, Jan. 1977.

[180] R.G. Gallager and S.J. Golestaani. Flow control and routing algorithms for data networks. *Proc. International Computer Communications Conference*, pp. 779-784, 1980.

[181] C. Gao, J.W.S. Liu, and M. Railey. Load balancing algorithms in homogeneous distributed systems. *Proc. Int. Conference on Parallel Processing*, pp. 302-306, IEEE 1984.

[182] B. Gavish and S.L. Hantler. An algorithm for optimal route selection in sna networks. *IEEE Transactions on Communications*, Vol. COM-31, No. 4, pp. 1154-1161, 1983.

[183] B. Gavish and I. Neuman. A system for routing and capacity assignment in computer communication networks. *IEEE Transactions on Communications*, Vol. COM-37, No. 4, pp. 360-366, Apr. 1989.

[184] E. Gelenbe and I. Mitrani. *Analysis and Synthesis of Computer Systems.* Academic Press, 1980.

[185] E. Gelenbe and G. Pujolle. *Introduction to Queueing Networks.* J.Wiley & Sons, 1987.

[186] M. Gerla. Packet, circuit and virtual circuit switching. *in W. Chou (ed.) "Computer Communications vol II : Systems and Applications"*, pp. 222-267, Prentice Hall 1985.

[187] M. Gerla, H.W. Chan, and J.R. Boisson De Marca. Routing, flow control and fairness in computer networks. *Proc. of International Communications Conference*, pp. 1272-1275, IEEE 1984.

[188] M. Gerla and P.O. Nilsson. Routing and flow control interplay in computer networks. *Proc. International Conference on Computer Communications*, pp. 84-89, 1980.

[189] A. Gersht. Analytical model of dynamic routing in virtual circuit packet switching network. *Proc. International Conference on Computer Communications*, pp.76-80, IEEE 1982.

[190] A. Gersht and K.J. Lee. Virtual-circuit load control in fast packet-switched broadband networks. *Prc. GLOBECOM*, pp. 214-220, IEEE 1988.

[191] A. Gersht and K.L. Lee. A congestion control framework for ATM networks. *Proc. Infocom'89 Conference*, pp. 701-710, IEEE 1989.

[192] P.E. Gill and W. Murray. *Numerical Methods for Constrained Optimization.* Academic Press, 1974.

[193] D.W. Glazer and C. Tropper. A new metric for dynamic routing algorithms. *IEEE Transactions on Communications*, Vol. 38, No. 3, pp. 360-367, March 1990.

[194] R.M. Glorioso and F.C. Colon. Cybernetic control of computer networks. *Modeling and Simulation Vol. 5, No 2, Proc. Fifth Annual Pittsburg Conf.*, pp.819-824, April 1974.

[195] R.M. Glorioso, G.R. Grueneich, and J.C. Dunn. Self organization and adaptive routing for communication networks. *EASCON '69 Record*, pp.243-250, 1969.

[196] R.M. Glorioso, G.R. Grueneich, and D. McElroy. Adaptive routing in a large communication network. *Proc. 9th Symposium on Adaptive Processes*, pp. XV.5.1-XV.5.4, 1970.

[197] R.M. Glorioso and F.C.C. Osorio. *Engineering Intelligent Systems: Concepts, Theory, and Applications.* Digital Press, 1980.

[198] R. Glowinski, J.T. Lions, and R. Tremolieres. *Numerical Analysis of Variational Inequalities.* North-Holland, 1981.

[199] S.J. Golestani. Congestion-free communication in broadband packet networks. *Proc. Int. Communications Conference*, pp. 489-494, IEEE 1990.

[200] S.J. Golestani. Congestion-free transmission of real-time traffic in packet networks. *Proc. Infocom 90 Conference*, pp. 527-536, IEEE 1990.

[201] P.M. Gopal, B.K. Kadaba, and G. Wieber. Load distribution in packet-switched networks. *Proc. IEEE International Conference on Communications*, pp. 980-986, IEEE 1987.

[202] D. Gross and C.M. Harris. *Fundamentals of Queueing Theory.* J.Wiley & Sons, 2nd ed. 1985.

[203] W.A. Gruver and E. Sachs. *Algorithmic Methods in Optimal Control.* Pitman Advanced Publ., 1980.

[204] W. Guth and B. Kalkofen. *Unique Solutions for Strategic Games.* Springer-Verlag, 1989.

[205] A. Hac and X. Jin. Dynamic load balancing in a distributed system using a decentralized algorithm. *Proc. Int. Conference on Distributed Computing Systems*, pp. 170-177, IEEE 1987.

[206] A. Hac and X. Jin. Dynamic load balancing in a distributed system using a sender-initiated algorithm. *Proc. Local Computer Networks*, pp. 172-180, IEEE 1988.

[207] A. Hac and T.J. Johnson. A study of dynamic load balancing in a distributed system. *Proc. SIGGCOM*, pp. 348-356, ACM 1986.

[208] E.L. Hahne. Dynamic routing in an unreliable manufacturing network with limited storage. *Lab. for Information and Decision Systems, MIT*, LIDS-TH-1063, OSP No. 87049, Febr. 1981.

[209] E.L. Hahne and R.G. Gallager. Round robin scheduling for fair flow control in data communication networks. *Proc. International Communications Conference*, pp. 103-107, IEEE 1986.

[210] B. Hajek and R.G. Ogier. Optimal dynamic routing in communication networks with continuous traffic. *Proc. Conference on Decision and Control*, pp. 369-374, IEEE 1982.

[211] B. Hajek and R.G. Ogier. Optimal dynamic routing in communication networks with continuous traffic. *Networks*, Vol. 14, pp. 457-487, 1984.

[212] S. Halfin. The shortest queue problem. *J. Applied Probability*, Vol. 22, pp. 865-875, 1985.

[213] J.C. Harsanyi and R. Selten. *A General Theory of Equilibrium Selection in Games.* The MIT Press, 1988.

[214] J.F. Hayes. *Modeling and Analysis of Computer Communication Networks.* Plenum Press, 1984.

[215] M.R. Hestenes. *Calculus of Variations and Optimal Control Theory.* J. Wiley & Sons, 1966.

[216] M. Hirano and N. Watanabe. Characteristics of a cell multiplexer for bursty ATM traffic. *Proc. Int. Communications Conference*, pp. 399-403, IEEE 1989.

[217] I. Hlavacek, J. Haslinger, J. Necas, and J. Lovisek. *Solution of Variational Inequalities in Mechanics.* Springer-Verlag, 1988.

[218] Y.-C. Ho. Team decision theory and information structure. *Proceedings of the IEEE*, Vol. 68, No, 6, pp. 644-654, June 1969.

[219] T.-C. Hou and D.M. Lucantoni. Performance analysis of an integrated video/data transport mechanism with built-in congestion control. *Proc. GLOBECOM*, pp. 231-238, IEEE 1988.

[220] R.A. Howard. *Dynamic Programming and Markov Processes.* The MIT Press, 1960.

[221] M.-T. Hsiao. Optimal decentralized flow control in computer communication networks. *Ph.D. Dissertation, Columbia University*, 1986.

[222] M.-T. Hsiao and A.A. Lazar. Optimal flow control of multi-class queueing networks with decentralized information. *Proc. IEEE Infocom 87*, pp. 652-661, IEEE 1987.

[223] M.-T.T. Hsiao and A. Lazar. A game theoretic approach to decentralized flow control of Markovian queueing networks. *Proc. Performance '87*, P.-J. Courtois and G. Latouche (eds.), Elsevier Sc. Publ., pp. 55-73, 1988.

[224] M.-T.T. Hsiao and A.A. Lazar. Optimal decentralized flow control of Markovian queueing networks with multiple controllers, part I: the team decision problem. *Data Communications Performance Evaluation*, pp. 357-372, 1987.

[225] M.-T.T. Hsiao and A.A. Lazar. Optimal flow control of multiclass queueing networks with partial information. *IEEE Transactions on Automatic Control*, Vol. 35, No. 7, pp. 855-860, July 1990.

[226] C.-Y.H. Hsu and J.W.-S. Liu. Dynamic load balancing algorithms in homogeneous distributed systems. *Proc.*, pp.216-223, IEEE 1986.

[227] J.Y. Hui. Resource allocation for broadband networks. *Journal On Selected Areas In Communications*, Vol. JSAC-6, pp. 1598-1608, 1988.

[228] P.A. Humblet, S.R. Soloway, and B. Steinka. Algorithms for data communication networks-part 2. *Codex*, pp. 1-23, Jun. 1986.

[229] B.M. Ibrahim and A.K. Elhakeem. Combined adaptive routing/capacity assignment in nested movable boundary frame integrated services networks. *Proc. GLOBECOM*, pp. 1867-1873, IEEE 1989.

[230] M.A. Iqbal, J.H. Saltz, and S.H. Bokhari. A comparative analysis of static and dynamic load balancing strategies. *Proc. Int. Conference on Parallel Processing*, pp. 1040-1047, IEEE 1986.

[231] R. Isaacs. *Differential Games.* J. Wiley & Sons, 1965.

[232] V. Jacobson. Congestion avoidance and control. *Proc. SIGCOMM*, pp. 314-329, ACM 1988.

[233] J.M. Jaffe. A decentralized "optimal", multiple-use, flow control algorithm. *Proc. International Computer Communications Conference*, pp. 839-843, 1980.

[234] J.M. Jaffe and A. Segall. Threshold design for dynamic routing. *Proc. IEEE International Communications Conference*, pp. 90-92, IEEE 1986.

[235] N.K. Jaiswal. *Priority Queues.* Academic Press, 1968.

[236] W. Jianhua. *The Theory of Games.* Oxford University Press, 1988.

[237] M.I. Kamien and N.L. Schwartz. Sufficient conditions in optimal control theory. *Journal Economic Theory*, Vol. 3, pp. 207-214, 1971.

[238] T. Kamitake and T. Suda. Evaluation of an admission control scheme for an ATM network considering fluctuations in cell loss rate. *Proc. GLOBECOM*, pp. 1774-1780, IEEE 1989.

[239] F. Kamoun. A drop and throttle flow control policy for computer networks. *IEEE Transactions on Communications*, Vol. COM-29, No. 4, pp. 444452, April 1981.

[240] F. Kamoun, A. Belguith, and J.L. Grange. Congestion control with a buffer management strategy based on traffic priorities. *Proc. Int. Conference on Computer Communications*, pp. 845-850, 1990.

[241] F. Kamoun and L. Kleinrock. Stochastic performance evaluation of hierarchical routing for large networks. *Computer Networks*, 3, pp. 337-353, 1979.

[242] E.L. Kaplan. *Mathematical Programming and Games.* J. Wiley & Sons, 1982.

[243] S. Karamardian. The complementarity problem. *Mathematical Programming*, Vol. 2, pp. 107-129, 1972.

[244] S. Karamardian. The nonlinear complementarity problem with applications, part 1. *J. of Optimization Theory and Applications*, Vol. 4, No. 2, pp. 87-98, 1969.

[245] S. Karamardian. The nonlinear complementarity problem with applications, part 2. *J. of Optimization Theory and Applications*, Vol. 4, No. 3, pp. 167-181, 1969.

[246] S. Karamardian. Generalized complementarity problem. *J. of Optimization Theory and Applicaions*, Vol. 8, No. 3, pp. 161-168, 1971.

[247] M.G.H. Katevenis. Fast switching and fair control of congested flow in broadband networks. *IEEE Journal on Selected Areas in Communications*, Vol. SAC-5, No. 8, pp. 1315-1326, Oct. 1987.

[248] S. Katz and I. Rubin. Session admission control for a multi-access communication channel. *Proc. Infocom '87 Conference*, pp. 635-642, IEEE 1987.

[249] F.P. Kelly. *Reversibility and Stochastic Networks.* J.Wiley & Sons, 1979.

[250] A. Khanna and J. Zinky. The revised ARPANET routing metric. *Proc. Communication Architectures and Protocols*, pp. 45-56, ACM 1989.

[251] A.Y. Khintchine. *Mathematical Methods in the Theory of Queues.* Hafner Publ. Co., 2nd ed., 1969.

[252] B.G. Kim and D. Towsley. Dynamic flow control protocols for packet-switching multiplexers serving real-time multipacket messages. *IEEE Transactions on Communications*, Vol. COM-34, No. 4, pp. 348-356, April 1986.

[253] D. Kinderlehrer and G. Stampacchia. *An Introduction to Variational Inequalities and their Applications.* Academic Press, 1980.

[254] D.E. Kirk. *Optimal Control Theory: an Introduction.* Prentice Hall, 1970.

[255] L. Kleinrock. *Queueing Systems, Vol. 1: Theory.* J. Wiley & Sons, 1975.

[256] L. Kleinrock. *Queueing Systems, Vol. 2: Applications.* J. Wiley & Sons, 1976.

[257] L. Kleinrock and W. Korfhage. Collecting unused processing capacity: an analysis of transient distributed systems. *Proc. 9th Int. Conference on Distributed Computing Systems*, pp. 482-489, IEEE 1989.

[258] L. Kleinrock and C.W. Tseng. Flow control based on limiting permit generation rates. *Proc. Int. Conference on Computer Communications*, pp. 785-790, 1980.

[259] A.H. Klopf. *The Hedonistic Neuron*. Hemisphere Publ. Co., 1982.

[260] C. Knessl, B.J. Matkowsky, Z. Schuss, and C. Tier. Two parallel queues with dynamic routing. *IEEE Transactions on Communications*, Vol. COM-34, No. 12, pp. 1170-1175, Dec. 1986.

[261] C. Knessl, B.J. Matkowsky, Z. Schuss, and C. Tier. Two parallel M/G/1 queues where arrivals join the system with the smaller buffer content. *IEEE Transactions on Communications*, Vol. COM-35, No. 11, pp. 1153-1158, Nov. 1987.

[262] G. Knowles. *An Introduction to Applied Optimal Control*. Academic Press, 1981.

[263] H. Kobayashi. *Modeling and Analysis: An Introduction to System Performance Evaluation Methodology*. Addison Wesley, 1978.

[264] H. Kobayashi and M. Gerla. Optimal routing in closed queueing networks. *ACM Transactions on Computer Systems*, Vol. 1, No. 4, pp. 294-310, Nov. 1983.

[265] N.N. Krasovskii and A.I. Subbotin. *Game-Theoretical Control Problems*. Springer-Verlag, 1988.

[266] I.A. Krass and S.M. Hammoudeh. *The Theory of Positional Games with Applications in Economics*. Academic Press, 1981.

[267] V.L. Kreps. Finite n-person non-cooperative games with unique equilibrium points. *International Journal of Game Theory*, Vol. 10, No. 3/4, pp. 125-129, 1981.

[268] K.R. Krishnan. Performance benefits of state-dependent routing of telephone traffic. *Proc. International Communications Conference*, pp. 1314-1318, IEEE 1990.

[269] K.R. Krishnan. Joining the right queue: a state dependent decision rule. *IEEE Transactions on Automatic Control*, Vol. 35, No. 1, pp. 104-107, Jan. 1990.

[270] H. Kroner. Comparative performance study of space priority mechanisms for ATM networks. *Proc. Infocom 90 Conference*, pp. 1136-1143, IEEE 1990.

[271] P. Krueger and M. Livny. The diverse objectives of distributed scheduling policies. *Proc. Int. Conference on Distributed Computing Systems*, pp. 242-249, IEEE 1987.

[272] A. Kumar. Adaptive load balancing in a multi-processor system with a central job scheduler. *Computer Performance and Reliability*, G. Iazeolla and P.J. Courtois and O.J. Boxma (eds), pp. 173-188, Elsevier Science Publ. 1988.

[273] A. Kumar. Adaptive load control of a central processor in a distributed system with a star topology. *IEEE Transactions on Computers*, Vol. 38, No. 11, pp. 1502-1512, Nov. 1989.

[274] P. Kumar and P. Varaiya. *Stochastic Systems: Estimation, Identification and Adaptive Control.* Prentice Hall, 1986.

[275] P.R. Kumar and J. Walrand. Individually optimal routing in parallel systems. *J. Applied Probability*, Vol. 22, pp. 989-995, 1985.

[276] J.F. Kurose and R. Chipalkatti. Load sharing in soft real-time distributed computer systems. *IEEE Transactions on Computers*, Vol. C-36, No. 8, pp. 993-1000, Aug. 1987.

[277] J.F. Kurose and R. Simha. Second derivative algorithms for optimal resource allocation in distributed computer systems. *Proc. Int. Conference on Distributed Computing Systems*, pp. 56-63, IEEE 1987.

[278] J.F. Kurose and R. Simha. A microeconomic approach to optimal resource allocation in distributed computer systems. *IEEE Transactions on Computers*, Vol. 38, No. 5, pp. 705-717, May 1989.

[279] J.F. Kurose and S. Simha. A distributed algorithm for optimum static load balancing in distributed computer systems. *Proc. Infocom 86 Conference*, pp. 458-467, IEEE 1986.

[280] J.F. Kurose, S. Singh, and R. Chipalkatti. A study of quasi-dynamic load sharing in soft real-time distributed computer systems. *Proc. Real Time Systems Symposium*, IEEE 1986.

[281] H.J. Kushner and H. Huang. Averaging methods for the asymptotic analysis of learning and adaptive systems, with small adjustment rate. *SIAM J. Control and Optimization*, Vol. 19, pp. 635-650, 1981.

[282] S. Lakshmivarahan. *Learning Algorithms Theory and Applications.* Springer Verlag, 1981.

[283] S. Lakshmivarahan and M.A.L. Thathachar. Absolutely expedient learning algorithms for stochastic automata. *IEEE Transactions on Systems, Man, and Cybernetics*, pp. 81-286, May 1973.

[284] S.S. Lam. Congestion control of packet communication networks by input buffer limits - a simulation study. *IEEE Transactions on Computers*, Vol. C-30, No. 10, pp. 733-742, Oct. 1981.

[285] S.S Lam and Y.C.L. Lien. Modeling and analysis of flow controlled packet switching networks. *Proc. 7th Data Communications Symposium*, pp.98-107, IEEE 1981.

[286] S.S. Lam and Y.L. Lien. Optimal routing in networks with flow-controlled virtual channels. *Proc. Computer Networks Performance Evaluation*, pp.38-46, ACM 1982.

[287] S.S. Lam and M. Reiser. Congestion control of store-and-forward networks by input buffer limits-an analysis. *IEEE Transactions on Communications*, Vol. COM-27, No. 1, pp. 127-134, Jan. 1979.

[288] S.S. Lavenberg. *Computer Performance Modeling Handbook.* Academic Press, 1983.

[289] A.M. Law and W.D. Kelton. *Simulation Modeling and Analysis.* McGraw-Hill, 1982.

[290] A.A. Lazar and M.-T. Hsiao. Network and user optimal flow control with decentralized information. *Proc. IEEE Infocom 86*, pp. 468-477, IEEE 1986.

[291] E. Lazowska, J. Zahorjan, G.S. Graham, and K.C. Sevcik. *Quantitative System Performance: Computer System Analysis Using Queueing Network Models.* Prentice-Hall, Inc., 1984.

[292] E.B. Lee and L. Markus. *Foundations of Optimal Control Theory.* J. Wiley & Sons, 1967.

[293] K.J. Lee and D. Towsley. A comparison of priority-based decentralized load balancing policies. *Proc. SIGMETRICS*, pp. 70-77, ACM 1986.

[294] K.J. Lee and D. Towsley. An asymptotic analysis of a threshold load balancing policy. *Proc. Infocom 89 Conference*, pp. 740-748, IEEE 1989.

[295] K.J. Lee, D. Towsley, and M. Choi. Distributed algorithms for minimum delay routing with constraints in communication networks. *Proc. Infocom Conference*, pp. 188-199, IEEE 1987.

[296] A. Leff, P.S. Yu, and Y.-H. Lee. Adaptive transaction routing in a heterogeneous database environment. *Proc. 9th Int. Conference on Distributed Computing Systems*, pp. 406-413, IEEE 1989.

[297] G. Leitmann. *Cooperative and Non-cooperative Many Player Differential Games.* Springer-Verlag, 1974.

[298] G. Leitmann. *The Calculus of Variations and Optimal Control: An Introduction.* Plenum Press, 1981.

[299] W.E. Leland and T.J. Ott. Unix process behavior and load balancing among loosely-coupled computers. *Teletraffic Analysis and Computer Performance Evaluation*, O.J. Boxma, J.W. Cohen, H.C. Tijms (eds.), pp. 191-208, Elsevier Science Publ. 1986.

[300] W.E. Leland and T.J. Ott. Load-balancing and process behavior. *Proc.*, pp. 54-69, ACM 1986.

[301] F.L. Lewis. *Optimal Control.* J. Wiley & Sons, 1986.

[302] S. Li and T. Basar. Distributed algorithms for the computation of noncooperative equilibria. *Automatica*, Vol. 23, No. 4, pp. 523-533, 1987.

[303] H.-C. Lin, J.R. Yee, and C.S. Raghavendra. Optimal joint load balancing and routing in message switched computer networks. *Proc. Infocom 88 Conference*, pp. 274-281, IEEE 1988.

[304] Y.-S. Lin and J.R. Yee. A distributed routing algorithm for virtual circuit data networks. *Proc. of IEEE infocom 89*, pp. 200-207, 1989.

[305] J.L. Lions and G. Stampacchia. Variational inequalities. *Communications on Pure and Applied Mathematics*, Vol. xx, pp. 493-519, 1967.

[306] H.T. Liu and J. Silvester. A two-mode dynamic algorithm (TDMA) for load balancing in distributed systems. *Second Annual Parallel Processing Symposium*, pp. 189-215, 1988.

[307] H.T. Liu and J. Silvester. An approximate performance model for load-dependent interactive queues with application to load balancing in distributed systems. *Proc. Infocom 88*, pp. 956-965, IEEE 1888.

[308] M. Livny and M. Melman. Load balancing in homogeneous broadcast distributed systems. *Proc. ACM Computer Network Performance Symposium*, pp.47-55, April 1982.

[309] H. Lu and M.J. Carey. Load-balanced task allocation in locally distributed computer systems. *Proc. Int. Conference on Parallel Processing*, pp. 1037-1039, IEEE 1986.

[310] R.D. Luce and H. Raiffa. *Games and Decisions.* J. Wiley & Sons, 1957.

[311] D.G. Luenberger. *Linear and Nonlinear Programming.* Addison-Wesley Publ. Co., 2nd ed., 1984.

[312] D.G. Luenberger. Complete stability of noncooperative games. *J. of Optimization Theory and Applications*, Vol. 25, No. 4, pp. 485-505, Aug. 1978.

[313] P.B. Luh, S.-C. Chang, and T.-S. Chang. Solutions and properties of multi-stage Stackelberg games. *Automatica*, Vol. 20, No. 2, pp. 251-256, 1984.

[314] M.H. MacDougall. *Simulating Computer Systems.* The MIT Press, 1987.

[315] J. Macki and A. Strauss. *Introduction to Optimal Control.* Springer-Verlag, 1982.

[316] E.A. MacNair and C.H. Sauer. *Elements of Practical Performance Modeling.* Prentice Hall, 1985.

[317] B. Maglaris, R. Boorstyn, S. Panwar, and T. Spirtos. Routing in burst-switched voice/data integrated networks. *Proc. Infocom 87 Conference*, pp. 162-169, IEEE 1987.

[318] B. Maglaris, R. Boorstyn, S. Panwar, and T. Spirtos. Routing of voice and data in burst-switched networks. *Transactions on Communications*, Vol. 38, No. 6, pp. 889-897, June 1990.

[319] B.S. Maglaris. An optimal local policy for two-level adaptive routing in computer networks. *Proc. of IEEE Infocom 84 Conf.*, 1984.

[320] O.L. Mangasarian. Sufficient conditions for the optimal control of nonlinear systems. *Journal SIAM Control*, Vol. 4, No. 1, pp. 139-152, 1966.

[321] L.G. Mason. Learning automata and telecommunications switching. *Third Yale Workshop on Applications of Adaptive Systems Theory*, pp.120-128, 1983.

[322] L.G. Mason. Equilibrium flows, routing patterns and algorithms for store-and-forward networks. *Large Scale Systems*, Vol. 8, pp. 187-209, 1985.

[323] J. Matsumoto and H. Mori. Flow control in packet-switched networks by gradual restrictions of virtual calls. *IEEE Transactions on Communications*, Vol. COM-29, No. 4, pp. 466-473, April 1981.

[324] N.F. Maxemchuk and M.El Zarki. Routing and flow control in high-speed wide-area networks. *Proceedings of the IEEE*, Vol. 78, No. 1, pp. 204-221, January 1990.

[325] I. McCausland. *Introduction to Optimal Control.* J. Wiley & Sons, 1969.

[326] J.M. McQuillan, I. Richer, and E.C. Rosen. The new routing algorithm for the ARPANET. *IEEE Transactions on Communications*, Vol. COJM-28, No. 5, pp. 711-719, May 1980.

[327] J.S. Meditch. On the state-space approach in modelingdata-communication networks. *Proc. 15th Annual Allerton Conference on Communication, Control, and Computing*, 1977.

[328] A. Mehlmann. *Applied Differential Games.* Plenum Press, 1988.

[329] R. Mirchandaney and J.A. Stankovic. Using stochastic learning automata for job scheduling in distributed processing systems. *Journal of Parallel, and Distributed Computing, 3*, pp. 527-552, 1986.

[330] R. Mirchandaney, D. Towsley, and J.A. Stankovic. Adaptive load sharing in heterogeneous systems. *Proc. 9th Int. Conference on Distributed Computing Systems*, pp. 298-306, IEEE 1989.

[331] R. Mirchandaney, D. Towsley, and J.A. Stankovic. Analysis of the effects of delays on load sharing. *IEEE Transactions on Computers*, Vol. 38, No. 11, pp. 1513-1525, Nov. 1989.

[332] I. Mitrani. *Modeling of Computer and Communication Systems.* Cambridge University Press, 1987.

[333] S.P. Morgan. Window flow control on a trunked bute-stream virtual circuit. *IEEE Tr. on Communications*, Vol. 36, No. 7, pp. 816-825, July 1988.

[334] P. Morse. *Queues, Inventories, and Maintenance.* J. Wiley & Sons, 1958.

[335] F.H. Moss and A. Segall. An optimal control approach to dynamic routing in networks. *IEEE Tr. on Automatic Control*, Vol.AC -27, No. 2, pp. 329-339, Apr. 1982.

[336] H. Moulin. *Game Theory for the Social Science.* New York University Press. 2nd ed., 1986.

[337] K.H. Muralidhar and M.K. Sundareshan. A hierarchical scheme for multi-objective adaptive routing in large communication networks. *Proceedings of the IEEE*, Vol. 71, No. 12, pp. 1461-1463, Dec. 1983.

[338] K.H. Muralidhar and M.K. Sundareshan. Combined routing and flow control in computer communication networks: a two-level adaptive scheme. *IEEE Transactions on Automatic Control*, Vol. AC-32, No. 1, pp. 15-25, Jan. 1987.

[339] W. Murray. *Numerical Methods for Unconstrained Optimization.* Academic Press, 1972.

[340] M.W. Mutka and M. Livny. Scheduling remore processing capacity in a workstation-processor bank network. *Proc. Int. Conference on Distributed Computing Systems*, pp. 2-9, IEEE 1987.

[341] S. Narasimhan, H. Pirkul, and P. De. Route selection in backbone data communication networks. *Computer Networks and ISDN Systems*, Vol. 15, pp. 121-133, 1988.

[342] K. Narendra and M.A.L. Thathacher. *Learning Automata: An Introduction.* Prentice Hall, 1989.

[343] K.S. Narendra and M.A.L. Thathachar. On the behavior of a learning automaton in a changing environment with application to telephone traffic routing. *IEEE Trans. on Systems, Man and Cybernetics*, Vol. SMC-10, No. 5, May 1980.

[344] K.S. Narendra and M.A.L. Thathachar. Learning automata : A survey. *IEEE Trans. on Systems, Man, and Cybernetics*, Vol. SMC-4, No. 4, pp. 323-334, July 1974.

[345] K.S. Narendra and R.M. Wheeler. Routing in communication networks - a case study of learning in large scale systems. *Large Scale Systems 8*, pp. 211-222, 1985.

[346] K.S. Narendra, E.A. Wright, and L.G. Mason. Application of learning automata to telephone traffic routing and control. *IEEE Trans. on Systems, Man, and Cybernetics*, Vol. SMC-7, No.11, pp. 785-792, Nov. 1977.

[347] J. Nash. Non-cooperative games. *Annals of Mathematics*, Vol. 54, No. 2, pp. 286-295, Sept. 1951.

[348] J.F.Jr. Nash. Equilibrium points in n-person games. *Proc. N.A.S. Mathematics*, Vol. 36, pp. 48-49, 1950.

[349] O.V. Nedzelnitsky. Learning algorithms in data communication networks. *Proc. Third Yale Workshop on Applications of Adaptive Systems Theory*, 1983.

[350] O.V.Jr. Nedzelnitsky and K.S. Narendra. Nonstationary models of learning automata routing in data communication networks. *IEEE Tr. on Systems, Man and Cybernetics*, Vol. SMC-17, No. 6, pp. 1004-1015, Nov./Dec. 1987.

[351] F. Neelamkavil. *Computer Simulation and Modelling.* J.Wiley & Sons, 1986.

[352] L.M. Ni and K. Hwang. Optimal load balancing strategies for a multiple processor system. *Proc. Int. Conference on Parallel Processing*, pp. 352-357, IEEE 1981.

[353] L.M. Ni and K. Hwang. Optimal load balancing in a multiple processor system with many job classes. *IEEE Transactions on Software Engineering*, Vol. SE-11, No. 5, pp. 491-496, May IEEE 1985.

[354] L.M. Ni, C.-W. Xu, and T.B. Gendreau. A distributed drafting algorithm for load balancing. *IEEE Transactions on Software Engineering*, Vol. SE-11, No. 10, pp. 1153-1161, Oct. 1985.

[355] M.F. Norman. *Markov Processes and Learning Models.* Academic Press, 1972.

[356] M.F. Norman. Markovian learning processes. *SIAM Review*, Vol. 16, No. 2, pp. 143-162, April 1974.

[357] M.F. Norman. A central limit theorem for Markov processes that move by small steps. *The Annals of Probability*, Vol. 2, No. 6, pp. 1065-1074, 1974.

[358] R.G. Ogier. Minimum-delay routing in continuous-time dynamic networks with piecewise-constraint capacities. *Networks*, Vol. 18, pp. 303-318, 1988.

[359] H. Ohnishi, T. Okada, and K. Noguchi. Flow control schemes and delay/loss tradeoff in ATM networks. *IEEE J. Selected Areas in Communications*, Vol. 6, No. 9, pp. 1609-1616, IEEE 1988.

[360] H. Okazaki and M. Schwartz. Preliminary results on routing in hybrid link systems. *Proc. International Communications Conference*, pp. 1527-1532, IEEE 1988.

[361] K. Okuguchi. Expectations and stability in oligopoly models. *Springer-Verlag*, 1976.

[362] G.J. Olsder and R. Suri. Time-optimal control of parts routing in a manufacturing system with failure-prone machines. *Proc. of IEEE 19th Conf. on Decision and Control*, pp.722-727, 1980.

[363] B.J. Oommen. Ergodic learning automata capable of incorporating a priori information. *IEEE Transactions on Systems, Man, and Cybernetics*, Vo. SMC-17, No. 4, pp. 717-723, July/Aug. 1987.

[364] B.J. Oommen. Absorbing and ergodic discretized two-action learning automata. *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. SMC-16, No. 2, pp. 282-293, March/April 1986.

[365] B.J. Oommen and J.P.R. Christensen. ϵ-optimal discretized linear reward-penalty learning automata. *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. SMC-18, No. 3, pp. 451-458, May/June 1988.

[366] B.J. Oommen and D.C.Y. Ma. Deterministic learning automata to the equipartitioning problem. *IEEE Transactions on Computers*, Vol. 37, No. 1, pp. 2-13, Jan. 1988.

[367] B.J. Oommen and M.A.L. Thathachar. Multiaction learning automata possessing ergodicity of the mean. *Information Sciences*, Vol. 35, pp. 183-198, 1985.

[368] J.M. Ortega and W.C. Rheinboldt. *Iterative Solution of Nonlinear Equations in Several Variables.* Academic Press, 1970.

[369] G. Owen. *Game Theory.* Academic Press, 2nd ed., 1982.

[370] K.S. Narendra P. Mars and M. Crystall. Learning automata control of computer communications networks. *Third Yale Workshop on Applications of Adaptive Systems Theory*, pp. 114-119, 1983.

[371] G.P. Papavassilopoulos. Algorithms for static Stackelberg games with linear costs and polyhedra constraints. *Proc. Conference on Decisions and Control*, pp. 647-652, IEEE 1982.

[372] G.P. Papavassilopoulos. Algorithms for leader-follower games. *Proc. 18th Annual Allerton Conference on Communication, Control, and Computing*, pp. 851-859, 1980.

[373] G.P. Papavassilopoulos. Solution of some stochastic quadratic Nash and leader-follower games. *SIAM J. Control and Optimization*, Vol. 19, No. 4, pp. 651-666, Sept. 1981.

[374] G.P. Papavassilopoulos. On the linear-quadratic-Gaussian Nash game with one-step delay observation sharing pattern. *IEEE Transactions on Automatic Control*, Vol. AC-27, No. 5, pp. 1054-1071, Oct. 1982.

[375] G.P. Papavassilopoulos and J.B.Jr. Cruz. Sufficient conditions for Stackelberg and Nash strategies with memory. *J. of Optimization Theory and Applications*, Vol. 31, No. 2, pp. 233-260, June 1980.

[376] G.P. Papavassilopoulos and J.B.Jr. Cruz. Nonclassical control problems and Stackelberg games. *IEEE Transactions on Automatic Control*, Vol. AC-24, No. 2, pp. 155-165, April 1979.

[377] F. Pavlidou. A variable reduction routing algorithm. *Proc. GLOBECOM*, pp. 2129-2132, IEEE 1987.

[378] M.C. Pennotti and M. Schartz. Congestion control in store and forward tandem links. *IEEE Tr. on Communications*, Vol. COM-23, No. 12, pp. 1434-1443, Dec. 1975.

[379] D.W. Peterson and J.H. Zalkind. A review of direct sufficient conditions in optimal control theory. *International Journal Control*, Vol. 28, No. 4, pp. 589-610, 1978.

[380] D.W. Petr and V.S. Frost. Optimal packet discarding: an ATM-oriented analysis model and initial results. *Proc. Infocom 90 Conference*, pp. 537-542, IEEE 1990.

[381] J.B. Plant. *Some Iterative Solutions in Optimal Control.* The MIT Press, 1968.

[382] E. Polak and D.Q. Mayne. An algorithm for optimization problems with functional inequality constraints. *IEEE Transactions on Automatic Control*, Vol. AC-21, No. 2, pp. 184-193, April 1976.

[383] B.T. Polyak. Convergence and convergence rate of iterative stochastic algorithms, I. the general case. *Avtomatika i Telemekhanika*, No. 12, pp. 83-94, Dec. 1976.

[384] B.T. Polyak. Convergence and convergence rate of iterative stochastic algorithms, II. the linear case. *Avtomatika i Telemekhanika*, No. 4, pp. 101-107, April 1977.

[385] J.-P. Ponssard. *Competitive Strategies.* North Holland Publ. Co., 1981.

[386] A.S. Poznyak. Investigation of the convergence of algorithms for the functioning of learning stochastic automata. *Avtomatika i Telemekhanika*, No. 1, pp. 88-103, Jan. 1975.

[387] W.H. Press, B.P. Flannery, S.A. Teukolsky, and W.T. Vetterling. *Numerical Recipes.* Cambridge University Press, 1986.

[388] G. Pujolle, D. Seret, D. Dromard, and E. Horlait. *Integrated Digital Communications Networks, Vol.1.* J. Wiley & Sons, 1988.

[389] G. Pujolle, D. Seret, D. Dromard, and E. Horlait. *Integrated Digital Communications Networks, Vol.2.* J. Wiley & Sons, 1988.

[390] S. Pulidas, D. Towsley, and J.A. Stankovic. Imbedding gradient estimators in load balancing algorithms. *8th Int. Conf. on Distributed Computing Systems*, pp. 482-490, IEEE 1988.

[391] J. Quirk and R. Saposnik. *Introduction to General Equilibrium Theory and welfare Economics.* McGraw-Hill Inc., 1968.

[392] K.K. Ramakrishnan and R. Jain. A binary feedback scheme for congestion avoidance in computer networks. *Transactions on Computer Systems*, Vol. 8, No. 2, pp. 158-181, May 1990.

[393] C.V. Ramamoorthy and W.-T. Tsai. An adaptive hierarchical algorithm. *Proc. COMPSAC*, pp. 93-104, IEEE 1983.

[394] G. Ramamurthy and R.S. Dighe. Integration of high speed continuous stream data traffic in a broadband packet network. *Proc. Infocom 89 Conference*, pp. 1063-1071, IEEE 1989.

[395] G. Ramamurthy and R.S. Dighe. Distributed source control: a network access control for integrated broadband packet networks. *Proc. Infocom 90 Conference*, pp. 896-907, IEEE 1990.

[396] A. Rapoport. *N-person Game Theory.* The University of Michigan Press, 1970.

[397] D.A. Reed and C.Kim. Packet routing algorithms for integrated switching networks. *Proc. SIGMETRICS*, pp. 7-15, ACM 1987.

[398] K.L. Rider. A simple approximation to the average queue size in the time-dependent M/M/1 queue. *Journal of the Association for Computing Machinery*, Vol. 23, No. 2, pp. 361-367, April 1976.

[399] J. Riordan. *Stochastic Service Systems.* J. Wiley & Sons. 1962.

[400] Z. Rosberg. Optimal dispatching to parallel heterogeneous servers in light traffic. *Proc. 28th Conference on Decision and Control*, pp. 1539-1543, IEEE 1989.

[401] Z. Rosberg. Deterministic routing to buffered channels. *IEEE Transactions on Communications*, Vol. COM-34, No. 5, pp. 504-507, May 1986.

[402] Z. Rosberg and A.M. Makowski. Optimal routing to parallel heterogeneous servers- small arrival rates. *IEEE Transactions on Automatic Control*, Vol. 35, No. 7, pp. 789-796, July 1990.

[403] J.B. Rosen. Existence and uniqueness of equilibrium points for concave n-person games. *Econometrica*, Vol. 33, No. 3, pp. 520-534, July 1965.

[404] J. Rosenmuller. *The Theory of Games and Markets.* North-Holland Publ. Co., 1981.

[405] J. Rosenmuller. On a generalization of the Lemke-Howson algorithm to noncooperative n-person games. *SIAM J. Applied Mathematics*, Vol. 21, No. 1, pp. 73-79, July 1971.

[406] S. Ross. *Introduction to Stochastic Dynamic Programming.* Academic Press, 1983.

[407] M.H. Rothkopf and S.S. Oren. A closure approximation for the nonstationary M/M/s queue. *Management Science*, Vol. 25, No. 6, pp. 522-534, June 1979.

[408] R.Y. Rubinstein. *Monte Carlo Optimization, Simulation and Sensitivity of Queueing Networks.* J.Wiley & Sons, 1986.

[409] H. Rudin. On routing and delta routing: a taxonomy and performance comparison of techniques for packet-switched networks. *IEEE Transactions on Communications*, Vol. COM-24, No. 1, pp. 43-59, Jan. 1978.

[410] H. Rudin and H. Muller. On routing and flow control. *Flow control in computer networks*, J.-L. Grange and M. Gien (eds.), pp. 241-255, North-Holland Publ. Co. 1979.

[411] H. Rudin and H. Muller. More on routing and flow control. *Proc. National Telecommunications Conference*, pp. 34.5.1-34.5.9, IEEE 1979.

[412] D.L. Russell. *Mathematics of Finite-Dimensional Control Systems.* Marcel Dekker Inc.. 11979.

[413] S. Saad and M. Schwartz. Input buffer limiting mechanisms for congestion control. *Proc. Int. Communications Conference*, pp. 23.1.1-23.1.5, IEEE 1980.

[414] T.L. Saaty. *Elements of Queueing Theory with Applications.* Mc Graw-Hill, 1961.

[415] A.P. Sage. *Optimum Systems Control.* Prentice Hall, 1968.

[416] V.R. Saksena. Analysis of routing issues in the design of packet networks. *IFAC Large Scale Systems*, pp. 803-808, 1986.

[417] N.S.Jr. Sandell. On open-loop and closed-loop Nash strategies. *IEEE Transactions on Automatic Control*, pp. 435-436, Aug. 1974.

[418] B.A. Sanders. An asynchronous, distributed flow control algorithm for rate allocation in computer networks. *IEEE Transactions on Computers*, Vol. 37, No. 7, pp. 779-787, July 1988.

[419] B.A. Sanders. An incentive compatible flow control algorithm for rate allocation in computer networks. *IEEE Transactions on Computers*, Vol. 37, No. 9, pp. 1067-1072, Sept. 1988.

[420] B.A. Sanders. A private good/public good decomposition for optimal flow control of an M/M/1 queue. *IEEE Transactions on Automatic Control*, Vol. AC-30, No. 11, pp. 1143-1145, Nov. 1985.

[421] P.E. Sarachik. A dynamic alternate route strategy fo traffic networks. *Proc. 21st Conference on Decision and Control*, pp. 120-124, IEEE 1982.

[422] P.E. Sarachik. Decentralized dynamic clearing of congested multi-destination networks. *Proc. 19th Annual Allerton Conference on Communication, Control, and Computing*, pp. 797-803, 1981.

[423] P.E. Sarachik. An effective local dynamic strategy to clear congested multi-destination networks. *IEEE Transactions on Automatic Control*, Vol. AC-27, No. 2, pp. 510-513, Apr. 1982.

[424] P.E. Sarachik and U. Ozguner. On decentralized dynamic routing for congested traffic networks. *IEEE Transactions on Automatic Control*, Vol. AC-27, No. 6, pp. 1233-1238, Dec. 1982.

[425] I.G. Sarma, R.K. Ragade, and U.R. Prasadi. Necessary conditions for optimal strategies in a class of noncooperative n-person differential games. *SIAM J. Control*, Vol. 7, No. 4, pp. 637-644, Nov. 1969.

[426] G. Sasaki and B. Hajek. Optimal dynamic routing in single commodity networks by iterative methods. *IEEE Tr. on Communications*, Vol. COM-35, No. 11, pp. 1199-1205, Nov. 1987.

[427] C.H. Sauer and K. M. Chandy. *Computer Systems Performance Modeling.* Prentice Hall, 1981.

[428] C.H. Sauer and E.A. MacNair. *Simulation of Computer Communication Systems.* Prentice Hall, 1983.

[429] H. Scarf. *The Computation of Economic Equilibria.* Yale University Press, 1973.

[430] H. Schulzrinne, J.F. Kurose, and D. Towsley. Congestion control for real-time traffic in high-speed networks. *Proc. Infocom 90 Conference*, pp. 543-550, IEEE 1990.

[431] M. Schwartz. *Computer Communication Network Design and Analysis.* Prentice Hall, 1977.

[432] M. Schwartz. *Telecommunication Networks: Protocols, Modeling and Analysis.* Addison-Wesley, 1987.

[433] M. Schwartz and C.K. Cheung. The gradient projection algorithm for multiple routing in message-switched networks. *IEEE Transactions on Communications*, pp. 449-456, April 1976.

[434] M. Schwartz and S. Saad. Analysis of congestion control techniques in computer communication networks. *Flow Control in Computer Networks*, J.-L. Grange and M. Gien (eds.), pp. 113-130, North-Holland Publ. Co. 1979.

[435] M. Schwartz and T.E. Stern. Routing technique used in computer communication networks. *IEEE Tr. on Communications*, Vol. COM-28, No. 4, pp. 539-552, Apr. 1980.

[436] M. Schwartz and T.-K. Yum. Distributed routing in computer communication networks. *Proc. Conference on Decision and Control*, pp. 600-603, IEEE 1982.

[437] A. Segall. The modeling of adaptive routing in data communication networks. *IEEE Tr. on Communications*, Vol. COM-25, No. 1, pp. 85-95, Jan. 1977.

[438] J. Seidler. *Principles of Computer Communication Network Design.* Ellis Horwood Lim., 1983.

[439] A. Seierstad and K. Sydsaeter. Sufficient conditions in optimal control theory. *International Economic Review*, Vol. 18, No. 2, pp. 367-391, June 1977.

[440] S.P. Sethi and G.L. Thompson. *Optimal Control Theory: Applications to Management Science.* Martinus Nijhoff Pu., 1981.

[441] S. Shenker and A. Weinrib. The optimal control of heterogeneous queueing systems: A paradigm for load-sharing and routing. *IEEE Transactions on Computers*, Vol. 38, No 12, pp. 1724-1735, Dec. 1989.

[442] K. Shimizu and E. Aiyoshi. A new computational method for Stackelberg and min-max problems by use of a penalty method. *IEEE Transactions on Automatic Control*, Vol. AC-26, No. 2, pp. 460-466, April 1981.

[443] M. Shubik. *Games for Society, Business and War: Towards a Theory of Gaming.* Elsevier Scientific Publ. Co., 1975.

[444] M. Shubik. *Game Theory in the Social Sciences: Concepts and Solutions.* The MIT Press, 1982.

[445] M. Shubik. *A Game-Theoretic Approach to Political Economy.* The MIT Press, 1984.

[446] M. Sidi, W.-Z. Liu, and I. Gopal. Congestion control through input rate regulation. *Proc. GLOBECOM 1989*, pp. 1764-1768, IEEE 1989.

[447] M. Simaan. Stackelberg optimization of two-level systems. *IEEE Transactions on Systems, Man, and Cybernetics*, pp. 554-557, July 1977.

[448] M. Simaan and J.B.Jr. Cruz. On the Stackelberg strategy in nonzero-sum games. *J. of Optimization Theory and Applications*, Vol. 11, No. 5, pp. 533-555, 1973.

[449] M. Simaan and J.B.Jr. Cruz. Additional aspects of the Stackelberg strategy in nonzero-sum games. *J. of Optimization Theory and Applications*, Vol. 11, No. 6. pp. 613-626, 1973.

[450] R. Simha and J.F. Kurose. Stochastic approximation schemes for a load balancing problem. *27th Annual Allerton Conference on Communication, Control, and Computing*, pp. 839-848, 1989.

[451] R. Simha and J.F. Kurose. Relative reward strength algorithms for learning automata. *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 19, No. 2, pp. 388-398, March/April 1989.

[452] M.J. Smith. Existence, uniqueness and stability of traffic equilibria. *Transportation Research*, Vol. 13B, pp. 295-304, 1979.

[453] P.R. Srikantakumar. Application of learning theory to communication networks control. *Third Yale Workshop on Applications of Adaptive Systems Theory*, pp. 135-141, 1983.

[454] P.R. Srikantakumar. Adaptive routing in large communication networks : Probabilistic study. *Proc. IEEE Conf. on Decision and Control*, pp. 398-401, 1981.

[455] P.R. Srikantakumar. Application of learning theory to communication networks control. *Proc. IEEE Cybernetics and Society Conf.*, pp. 555-559, 1983.

[456] P.R. Srikantakumar and K.S. Narendra. A learning model for routing in telephone networks. *SIAM J. Control and Optimization*, vol-20, no 1, Jan. 1982.

[457] K. Sriram. Dynamic bandwidth allocation and congestion control schemes for voice and data multiplexing in wideband packet technology. *Proc. Infocom'90 Conference*, pp. 1003-1009, IEEE 1990.

[458] W. Stadler. A survey of multicriteria optimization or the vector maximum problem. *J. of Optimization Theory and Applications*, Vol. 29, No. 1, pp. 1-52, Sept. 1979.

[459] H. Stalford and G. Leitmann. Sufficiency conditions for n-person differential games. *Topics in Differential Games*, A. Blaquiere (ed.), pp. 345-376, North Holland 1973.

[460] J.A. Stankovic. The analysis of a decentralized control algorithm for job scheduling utilizing Bayesian decision theory. *Proc. Int. Conference on Parallel Processing*, pp. 333-340, IEEE 1981.

[461] J.A. Stankovic. Simulations of three adaptive, decentralized controlled, job scheduling algorithms. *Computer Networks*, Vol. 8, pp. 199-217, 1984.

[462] J.A. Stankovic. An application of Bayesian decision theory to decentralized control of job scheduling. *IEEE Transactions on Computers*, Vol. C-34, No. 2, pp. 117-130, Feb. 1985.

[463] J.A. Stankovic. Stability and distributed scheduling algorithms. *IEEE Transactions on Software Engineering*, Vol. SE-11, No. 10, pp. 1141-1152, 1985.

[464] A.W. Starr and Y.C. Ho. Nonzero-sum differential games. *J. on Optimization Theory and Applications*, Vol. 3, No. 3, pp. 184-206, 1969.

[465] G.I. Stassinopoulos. Optimal dynamic routing in multidestination networks. *IEEE Transactions on Communications*, Vol. COM-35, No. 4, pp. 472-475, April 1987.

[466] G.I. Stassinopoulos and P. Konstantopoulos. Optimal congestion control in single destination networks. *IEEE Transactions on Communications*, Vol. COM-33, No. 8, pp. 792-800, Aug. 1985.

[467] G.I. Stassinopoulos and H. Kukutos. Optimal dynamic routing in double ring networks. *IEEE Transactions on Communications*, Vol. 37, No. 8, pp. 890-896, Aug. 1989.

[468] G.I. Stassinopoulos and E.N. Protonotarios. Dynamic routing and prospects for on line implementation. *Proc. of International Communications Conference*, pp. 141-144, IEEE 1986.

[469] G.I. Stassinopoulos and E.N. Protonotarios. Optimal dynamic routing in multidestination networks. *IEEE Tr. on Communications*, Vol. COM-35, No. 4, pp. 472-475, Apr. 1987.

[470] T.E. Stern. A class of decentralized routing algorithms using relaxation. *Proc. National Telecommunications Conference*, pp. 42.1.1-42.1-5, IEEE 1976.

[471] B.W. Stuck and E. Arthurs. *A Computer and Communications Network Performance Analysis Primer.* Prentic-Hall, Inc., 1985.

[472] S. Sumita. Synthesis of an output buffer management scheme in a switching system for multimedia communications. *Proc. Infocom 90 Conference*, pp. 1226-1233, IEEE 1990.

[473] R.S. Sutton and A.G. Barto. Toward a modern theory of adaptive networks: expectation and prediction. *Psychological Review*, Vol. 88, No. 2, pp. 135-170, 1981.

[474] F. Szidarovsky and S. Yakowitz. A new proof of the existence and uniqueness of the Cournot equilibrium. *International Economic Review*, Vol. 18, No. 3, pp. 787-789, Oct. 1977.

[475] T. Szymanski. An analysis of "Hot-Potato" routing in a fiber optic packet switched hypercube. *Proc. Infocom 90 Conference*, pp. 918-925, IEEE 1990.

[476] D. Tabak and B.C. Kuo. *Optimal Control by Mathematical Programming.* Prentice Hall, 1971.

[477] L. Takacs. *Introduction to the Theory of Queues.* Oxford University Press. 1962.

[478] A.S. Tanebaum. *Computer Networks.* Prentice Hall, 2nd ed., 1988.

[479] A.N. Tantawi and D. Towsley. Optimal static load balancing in distributed computer systems. *Journal of A.C.M., Vol. 32, No 2*, pp. 445-465, April 1985.

[480] L. Tesfatsion. Pure strategy Nash equilibrium points and the Lefschetz fixed point theorem. *International Journal of Game Theory*, Vol. 12, No. 3. pp. 181-191, 1983.

[481] G.H. Thaker and J.B. Cain. Interactions between routing and flow control algorithms. *IEEE Transactions on Communications*, Vol. COM-34, No. 3, pp. 269-277, March 1986.

[482] M.A.L. Thathachar and P.S. Sastry. A new approach to the design of reinforcement schemes for learning automata. *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. SMC-15, No. 1, pp. 168-175, Jan./Febr. 1985.

[483] A. Thomasian. A performance study of dynamic load balancing in distributed systems. *Proc. Int. Conf. on Distributed Computing Systems*, pp. 178-184, IEEE 1987.

[484] H.C. Tijms. *Stochastic Modelling and Analysis: A Computational Approach.* J.Wiley & Sons, 1986.

[485] D. Tipper and M.K. Sundareshan. An optimal control approach to decentralized dynamic virtual circuit routing in computer networks. *Proc. IEEE Infocom 90*, pp. 926-933, IEEE 1990.

[486] M.J. Todd. *The Computation of Fixed Points and Applications.* Springer-Verlag, 1976.

[487] D. Towsley and R. Mirchandaney. The effect of communication delays on the performance of load balancing policies in distributed systems. *Computer Performance and Reliability*, G. Iazeolla et.al. (eds), pp. 213-226, Elsevier Science Publ. 1988.

[488] K.S. Trivedi. *Probability and Statistics with Reliability, Queuing and Computer Science Applications.* Prentice-Hall, Inc., 1982.

[489] W.K. Tsai. Convergence of gradient projection routing methods in an asynchronous stochastic quasi-static virtual circuit network. *IEEE Tr. on Automatic Control*, Vol. AC-34, No. 1, pp. 20-33, Jan. 1989.

[490] W.K. Tsai, P.E. Cantrell, and J. Goos. Fairness of optimal routing in virtual circuit data networks. *Proc. of IEEE infocom 89*, pp. 119-126, 1989.

[491] W.K. Tsai, G. Huang, J.K. Antonio, and W.T. Tsai. Distributed iterative aggregation algorithms for box-constrained minimization problems and optimal routing in data networks. *IEEE Transactions on Automatic Control*, Vol. 34, No. 1, pp. 34-46, Jan. 1989.

[492] W.K. Tsai, J.N. Tsitsiklis, and D.P. Bertsekas. Some issues in distributed asynchronous routing in virtual circuit data networks. *Proc. IEEE 25th Conf. on Decision and Control*, pp.1335-1337, IEEE Dec. 1986.

[493] W.T. Tsai, C.V. Ramamoorthy, W.K. Tsai, and O. Nishiguchi. An adaptive hierarchical routing protocol. *Transactions on Computers*, Vol. 38, No. 8, pp. 1059-1074, Aug. 1989.

[494] M.L. Tsetlin. *Automaton Theory and Modeling of Biological Systems.* Academic Press, 1973.

[495] M.L. Tsetlin. On the behavior of finite automata in random media. *Automatika i Telemekhanika*, Vol. 22, No. 10, pp. 1345-1354, Oct. 1961.

[496] J.N. Tsitsiklis. Convexity and characterization of optimal policies in a dynamic routing problem. *J. of Optimization Theory and Applications*, Vol. 44, No. 1, pp. 105-136, Sept. 1984.

[497] J.N Tsitsiklis and D.P. Bertsekas. Distributed asynchronous optimal routing in data networks. *IEEE Transactions on Automatic Control*, Vol. AC-31, No. 4, pp. 325-332, Apr. 1986.

[498] Y.Z. Tsypkin and A.S. Poznyak. Finite learning automata. *Engineering Cybernetics*, Vol. 10, pp. 478-490, May-June 1972.

[499] M. Tu and G.P. Papavassilopoulos. Performance versus informativeness in linear-quadratic Gaussian noncooperative games. *J. of Optimization Theory and Control*, Vol. 57, No. 1, pp. 161-187, April 1988.

[500] M. Tu and G.P. Papavassilopoulos. Impact of explicit and implicit control sharing on the performance of two-person one-act LQG Nash games. *Large Scale Systems*, Vol. 7, pp. 219-226, 1984.

[501] M. Tu and G.P. Papavassilopoulos. On the informational properties of the Nash solution of LQG dynamic games. *IEEE Transactions on Automatic Control*, Vol. AC-30, No. 4, pp. 377-385, April 1985.

[502] P.P. Varaiya. *Notes on Optimization*. Van Nostrand Reinhold Co., 1972.

[503] V.I. Varshavskii and I.P. Vorontsova. On the behavior of stochastic automata with a variable structure. *Avtomatika i Telemekhanika*, Vol. 24, No. 3, pp. 353-360, March 1963.

[504] J. von Neuman and O. Morgenstern. *Theory of Games and Economic Behavior*. Princeton University Press, 3rd ed., 1953.

[505] J. Walrand. *An Introduction to Queueing Networks*. Prentice Hall, 1988.

[506] Y.-T. Wang and R.T.J. Morris. Load sharing in distributed systems. *IEEE Trans. on Computers, Vol C-34, No 3*, pp. 204-217, March 1985.

[507] J.G. Wardrop. Some theoretical aspects of road traffic research. *Proc. Inst. Civil Engineers*, Part III, 1, pp. 325-378, 1952.

[508] R.R. Weber. On the optimal assignment of customers to parallel servers. *J. Applied Probability*, Vol. 15, pp. 406-413, 1978.

[509] L. Wei, Y. Yiouwei, and H. Zheng. A distributed routing algorithm in hybrid circuit/packet switching networks. *Proc. Infocom 88 Conference*, pp. 791-795, IEEE 1988.

[510] A. Weinrib and S. Shenker. Greed is not enough: Adaptive load sharing in large heterogeneous systems. *Proc. Infocom 88*, pp. 986-994, IEEE 1988.

[511] W. Whitt. Deciding which queue to join: some counterexamples. *Operations Research*, Vol. 34, No. 1, pp. 55-62, Jan.-Febr. 1986.

[512] J.B. Williams. Stability in noncooperative games. *Operations Research*, Vol. 19, pp. 774-783, 1971.

[513] R.J. Williams. Sufficient conditions for Nash equilibria in n-person games over reflexive Banach spaces. *J. of Optimization Theory and Applications*, Vol. 30, No. 3, pp. 383-394, March 1980.

[514] R. Wilson. Computing equilibria of n-person games. *SIAM J. Applied Mathematics*, Vol. 21, No. 1, pp. 80-87, July 1971.

[515] W. Winston. Optimality of the shortest line discipline. *J. Applied Probability*, Vol. 14, pp. 181-189, 1977.

[516] R.W. Wolff. *Stochastic Modeling and the Theory of Queues.* Prentice Hall. 1989.

[517] G.M. Woodruff and R. Kositpaiboon. Multimedia traffic management principles for guaranteed ATM network performance. *IEEE J. on Selected Areas in Communications*, Vol. 8, No. 3, pp. 437-446, Aprl 1990.

[518] E.F. Wunderlich. An analysis of dynamic virtual circuit routing. *Proc. IEEE National Telecommunications Conference*, pp. A3.3. - A3.3.6, IEEE 1981.

[519] E.F. Wunderlich, L. Kaufman, and B. Gopinath. The control of store and forward congestion in packet switched networks. *Proc. Int. Conference on Computer Communications*, pp. 851-856, 1980.

[520] P.S. Yu, S. Balsamo, and Y.-H. Lee. On dynamic load sharing and transaction routing. *Proc. of International Computer Symposium*, pp. 1265-1271, 1986.

[521] P.S. Yu, S. Balsamo, and Y.-H. Lee. Dynamic load sharing in distributed database systems. *Proc. Fall Joint Computer Conference*, pp. 675-683, IEEE 1986.

[522] P.S. Yu, S. Balsamo, and Y.-H. Lee. Dynamic transaction routing in distributed database systems. *IEEE transactions on Software Engineering*, Vol. 14, No. 9, pp. 1307-1318, Sept. IEEE 1988.

[523] T. Yum. A semidynamic deterministic routing rule in computer communication networks. *Proc. National Telecommunications Conference*, pp. 34.4.1-34.4.7, IEEE 1979.

[524] T. Yum and M. Schwartz. Packet-switched performance with different circuit-switched routing procedures in nonhierarchical integrated circuit-switched and packet-switched networks. *IEEE Transactions on Communications*, Vol. COm-35, No. 3, pp. 362-366, 1987.

[525] T.-S. Yum and M. Schwartz. Comparison of adaptive routing algorithms for computer communication networks. *Proc. National Telecommunications Conference*, pp. 4.1.1-4.1.4, IEEE 1978.

[526] T.-S. Yum and M. Schwartz. The join-biased-queue rule and its application to routing in computer communication networks. *IEEE Transactions on Communications*, Vol. COM-29, No. 4, pp. 505-511, Apr. 1981.

[527] T.-S.P. Yum. The design and analysis of a semidynamic deterministic routing rule. *IEEE Transactions on Communications*, Vol. COM-29, No. 4, pp. 498-504, Apr. 1981.

[528] T.P. Yum and H.-C. Lin. Adaptive load balancing for parallel queues. *International Communications Conference*, pp. 1268-1271, IEEE 1984.

[529] W.I. Zangwill. *Nonlinear Programming.* Prentice Hall, 1969.

[530] C. Zhou and B.S. Maglaris. Adaptive routing in computer networks within two-level procedures: delay performance and effect on network flows. *Proc. GLOBECOM*, pp. 746-752, IEEE 1985.

[531] S. Zhou. A trace-driven simulation study of dynamic load balancing. *IEEE Transactions on Software Engineering*, Vol. 14, No. 9, pp. 1327-1341, Sept. 1988.

[532] S. Zhou and D. Ferrari. A measurement study of load balancing performance. *Proc. Int. Conference on Distributed Computing Systems*, pp. 490-497, IEEE 1987.

[533] J. Zinky, G. Vichniac, and A. Khanna. Performance of the revised routing metric in the ARPANET and MILNET. *Proc. MILCOM*, pp. 219-224, IEEE 1989.