ΠΑΝΕΠΙΣΤΗΜΙΟ ΜΑΚΕΔΟΝΙΑΣ

ΔΙΑΤΜΗΜΑΤΙΚΟ ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ
ΣΤΑ ΠΛΗΡΟΦΟΡΙΑΚΑ ΣΥΣΤΗΜΑΤΑ
MASTER IN INFORMATION SYSTEMS
(MIS)

ΚΑΘΗΓΗΤΗΣ
Α.Α.Οικονομίδης

ΤΕΧΝΟΛΟΓΙΑ ΔΙΚΤΥΩΝ

ΕΡΓΑΣΙΑ

Load balancing and algorithms in P2P networks

ΓΙΩΡΓΟΣ ΒΕΡΑΝΗΣ
Α.Μ. 27/06

ΦΕΒΡΟΥΑΡΙΟΣ 2007

ΠΕΡΙΕΧΟΜΕΝΑ

1. Abstract

Generally we can face the problem of load balancing using hardware, special software and algorithms. In this paper we present some of the most efficient static and dynamic algorithms for this problem. Load balancing is a critical issue for the efficient operation of peer-to-peer networks and we refer to protocols in order to balance the load. Structured peer-to-peer (P2P) systems address the load balancing issue in a rather naive way, by simply resorting to the uniformity of the hash function utilized to generate object IDs. Such a random choice of object IDs could result in O(logN) load imbalance. Also we show some rules for simulation for the algorithms.

1.Περίληψη

Γενικά μπορούμε να αντιμετωπίσουμε το πρόβλημα της ισορροπίας φόρτου χρησιμοποιώντας το hardware, πρόσθετο λογισμικό και τους αλγορίθμους. Σε αυτό το έγγραφο παρουσιάζουμε μερικούς από τους πιο γνωστούς και αποδοτικούς στατικούς και δυναμικούς αλγορίθμους. Η εξισορρόπηση φόρτου είναι ένα κρίσιμο ζήτημα για την αποδοτική λειτουργία των *peer-to-peer* δικτύων και γίνετε αναφορά σε πρωτόκολλα προκειμένου να υπάρξει ισορροπία φόρτου. Τα δομημένα P2P συστήματα αντιμετωπίζουν το ζήτημα αυτό με έναν μάλλον αφελή τρόπο, απλά προσφεύγουν στην ομοιομορφία της hash λειτουργίας που χρησιμοποιείται για να παραγάγει το αντικείμενο ID. Μια τέτοια τυχαία επιλογή του αντικειμένου ID θα μπορούσε να οδηγήσει στη δυσαναλογία φορτίων O (logN). Οι δυναμικοί αλγόριθμοι που προτείνονται επιλύουν το πρόβλημα αυτό. Επίσης δίνονται ορισμένοι κανόνες για την προσομοίωση των αλγορίθμων.

ΜΕΡΟΣ Α'
ΕΙΣΑΓΩΓΗ - ΙΣΤΟΡΙΚΗ ΑΝΑΔΡΟΜΗ – P2P ΔΙΚΤΥΑ

1. Εισαγωγή

Όταν το Internet βρισκόταν στα αρχικά του στάδια δηλαδή ήταν περισσότερο σε ακαδημαϊκό επίπεδο (παρόμοια εικόνα επικρατούσε και στα συστήματα των δικτύων), όπως ήταν φυσικό δεν υπήρχαν και μεγάλες ανάγκες να αναπυχθούν αλγόριθμοι ή και συστήματα που θα έκαναν την εξισσορόπηση του φόρτου μεταξύ των εξυπηρετητών.
Με την πάροδο του χρόνου εμφανίστηκαν πολλές εμπορικές διαδικτυακές εφαρμογές που είχαν μεγάλες απαιτήσεις από τους servers που τις φιλοξενούσαν . Έτσι οι κατασκευαστές άρχισαν να επενδύουν πολλά από τα κεφάλαια τους στην ανάπτυξη εκείνων των τεχνολογιών και συσκευών που θα τους επέτρεπαν να διαχειρίζονται την πληροφορία μεταξύ των server και των client με τον βέλτιστο τρόπο.
Μία από της πρώτες λύσης που παρουσιάστηκαν στον τομέα αυτό που μελετάμε είναι ο πρόγονος των εξισσοροπιστών φόρτου και ονομάστηκε DNS – Based Load Balancing. Οι διαχειριστές των server που φιλοξενούσαν αυτές τις σχετικά «βαριές» εφαρμογές για την εποχή εκείνη εφάρμοζαν ένα αλγόριθμο κυκλικής επαναφοράς γνωστός και ως Round Robin. Αυτή η μέθοδος χρημοποιούσε μία συνάρτηση του DNS που επέτρεπε να αντιστοιχηθούν περισσότερες από μία IP διευθύνσεις σε ένα hostname. Το αποτέλεσμα ήταν να κατανεμηθεί η κίνηση της πληρφορίας σε όσες IP εφαρμοζόταν ο αλγόριθμος.
Εκ πρώτης όψεως αυτό φαίνεται μια ικανοποιητική λύση για την κατανομή του φόρτου με δικαιοσύνη μεταξύ των εξυπηρετητών. Αλλά ωστόσο η μέθοδος αυτή είχε συγκεκριμένους περιορισμούς που περιλαμβάνουν την απρόσμενη εξισορρόπηση φόρτου, περιορισμού στην cache των εξυπηρετητών καθώς και απουσία μετρικών για την απόδοση των εξυπηρετητών.
Έτσι στα επόμενα χρόνια δόθηκε έμφαση σε λύσεις που αντιμετώπιζαν το πρόβλημα του φόρτου δεδομένων με μεγαλύτερη ακρίβεια και μεθοδικότητα. Η μέθοδος της εξισσορόπησης του φόρτου δεδομένων (γενικά της πληροφορίας) έχει σημαντικά πλεονετήματα για τους εξυπηρετητές ιστοτόπων , για παράδειγμα επιρέπουν στους διαχειστές τους να εκτελούν λειτουργίες συντήρησης των server σε ώρες αιχμής με δυναμικούς ή στατικούς αλγόριθμους.
Η έννοια της εξισσορόπησης φόρτου (Load balancing) με απλούς όρους σημαίνει την διαδικασία κίνησης που παρατηρείτε μεταξύ πολών κόμβων (π.χ εξυπηρετητές δικτύου) οι οποίοι είναι ανομοιογενής μεταξύ τους και γίνετε με τέτοιο τρόπο ώστε να μην προκληθεί η κατάρρευση τους από υπερφόρτωση. Είναι μία διαδικασία που δεν γίνετε αντιληπτή από τους τελικούς χρήστες ωστόσο παίζει σημαντικό ρόλο στην ποιοτική ιανοποίηση των χρηστών από το σύστημα που την εφαρμόζει.

Ο όρος Load Balancing είναι γνωστός και ώς Load Transfer ή Process Migration , είναι η μεταφορά δεδομένων από κόμβους με μεγάλο βάρος φόρτωσης πρός άλλους κόμβους που έχουν μικρότερη φόρτωση δεδομένων. Ο σκοπός αυτής της ενέργειας είναι να γίνει η εφαρμογή-διεργασία, που χρησιμοποιεί αυτούς τους κόμβους , καλύτερη , δηλαδή να έχει πιο ποιοτική απόδοση. Γενικά χρησιμοποιείται σε συστήματα που λειτουργούν για μεγάλο χρονικό διάστημα και ειδικότεα σε αυτά που

εκτελούν διάφορους υπολγισμούς. Το σημαντικότερο για να επιτευχθεί η εξισσορόπηση φόρτου είναι να καθοριστούν κριτήρια:

- Για το ποιοι κόμβοι είναι πολύ φορτωμένοι με δεδομένα και ποιοι έχουν λιγότερα (heavily-loaded, lightly loaded),
- Πως θα γίνει η μέτρηση της φόρτωσης (πολιτική , υλοποίηση) , ανταλλαγή πληροφοριών μεταξύ των κόμβων σχετικά με την φόρτωση και την κατάσταση στην οποία βρίσκονται,
- Ποιές εργασίες θα μεταφερθούν ώστε να επέλθει η ισσοροπία,
- Πότε θα γίνετε η μεταφορά των εργασιών (καινούργιες εργασίες μόνο , ήδη υπαρχουσες εργασίες)

Τα κριτήρια που προαναφέρθηκαν είναι πολύ γενικά καθώς κάθε σύστημα μπορεί να έχει και άλλα επιμέρους χαρακτηριστικά που πρέπει να παραμετροποιηθούν, ώστε να εφαρμοστεί κατάλληλος αλγόριθμος και να αρχίσει η εξισσορόπηση φόρτου μεταξύ των κόμβων και να μεγιστποιηθεί η απόδοση του συστήματος γενικότερα.

Η διαδικασία της εξισσορόπησης φόρτου μπορεί να επιτευχθεί είτε με μία συσκευή (hardware) είτε με λογισμικό (software) είτε με συνδυασμό αυτών. Παρακάτω γίνετε αναφορά στο load balancing software και στο load balancing hardware.

1.1. Load balancing software

Για εξοικονόμηση χρημάτων, πολλοί διαχειριστές δικτύων χρησιμοποιούν μια έτοιμη διαθέσιμη λύση που ονομάζεται DNS Round Robin, η οποία ανήκει στο κύριο web server και διακόπτει τα πακέτα μόλις φτάνουν στο site. Η DNS Round Robin είναι αρκετά απλή: στέλνει ένα πακέτο στον αρχικά υποδεδειγμένο server και αν αυτός δεν απαντήσει (είτε επειδή είναι απασχολημένος είτε επειδή επειδή είναι κλειστός), στέλνει το πακέτο στον αμέσως επόμενο υποδεδειγμένο server και ούτω καθ' εξής μέχρι να ξεκινήσει ξανά από τον πρώτο server. Αν και αποτελεί μια καλή λύση για ένα site με λίγους server, υπάρχουν περιορισμοί:
Ο DNS Round Robin δεν ενημερώνει για το αν ο server είναι κλειστός (και επομένως χρειάζεται επιδιόρθωση) ή είναι απλά απασχολημένος.
Δεν λαμβάνει υπ'όψην τον τύπο του περιεχομένου που ζητήθηκε, όπως για παράδειγμα τα γραφικά ή μια συγκεκριμένη URL.
Απασχολεί πολύτιμο χώρο στη μνήμη του server όταν τρέχει
Αν ο κύριος web server «πέσει» ο DNS Round Robin κάνει το ίδιο, εκτός αν τον αντεγάψετε πιστά σε έναν back up server. Ακόμα και τότε όμως, πώς φτάνει το πακέτο από το internet στον back up server;
Αν υπάρχει ένα μεγάλο site με πολύπλοκη server farm (cluster) διευθέτηση, ο DNS Round Robin γίνεται πολύ πιο δύσκολος στο σχηματισμό.

Υπάρχουν κι άλλες λύσεις software στην αγορά, που έχουν βελτιωμένα χαρακτηριστικά αναφορικά με τον DNS Round Robin, αλλά δεν μπορούν να εξισορροπήσουν επαρκώς το φόρτο σε μεγάλα sites ή σε sites με κατανεμημένο δίκτυο.

1.2. Load balancing hardware

Η βασισμένη σε HARDWARE εξισορρόπηση φόρτωσης παρουσιάζεται σε δύο τομείς: PC and Routing/Switching.

PC Load Balancing: μια λύση software που τίθεται σε ισχύ σε έναν υπολογιστή και πωλείται ως πακέτο. Ορισμένες φορές ο πωλητής δημιουργεί μια γέφυρα δια μέσου της εγκτάστασης δύο NIC Cards στον υπολογιστή. Αυτό προσφέρει κάποια προσομοίωση λειτουργικότητας του router και αποταμιεύει το χρόνο και την προσπάθεια που απαιτείται για την ολοκλήρωση μιας λύσης software με τον υπάρχων εξοπλισμό. Κάποιες βασισμένες σε pc λύσεις έχουν μεγάλα feature sets και πολύπλοκα GUI's, αλλά η διαχείριση και η εγκατάσταη είναι συνήθως πολύπλοκες και ο επακόλουθος σχεδιασμός είναι τις περισσότερες φορές ακριβός, απαιτώντας πρόσθετα modules software και hardware.Στην πραγματικότητα σε τελική ανάλυση, ξοδεύονται πολύ περισσότερα χρήματα από ότι αρχικά είχαν προβλεφτεί. Επιπρόσθετα υπάρχουν πολύ περισσότερα σημεία αποτυχίας σε έναν υπολογιστή από όσα βρίσκονται σε ένα switch ή σε ένα router.

Routing/Switching: Οι βασισμένοι σε switch και router load balancers προσφέρουν ενοποίηση χωρίς να είναι σε σύνδεση με το υπάρχον δίκτυο, σε αντίθεση με την βασισμένη σε PC λύση. Επιπλέον, οι σε switch ή router βασισμένες λύσεις δουλεύουν σε οποιοδήποτε λειτουργικό σύστημα και πλατφόρμα, και οι switches και οι routers σπάνια αποτυγχάνουν.
Οι router based load balancers επιτρέπουν τη χρήση συναρτήσεων για δρομολογητές για την καθιέρωση δυναμικών και στατικών μονοπατιών ώστε να διευκολυνθεί το traffic throughput και η ανακατεύθυνση.

Το φάσμα υλοποίησης των αλγορίθμων που προσπαθούν να επιλύσουν το πρόβλημα της εξισσορόπησης είναι αρκετά μεγάλο καθώς μπορούν να βρούν εφαρμογή σε πληθώρα δικτύων. Σε αυτήν την εργασία θα ασχοληθούμε περισσότερο με τους αλγόριθμους που είναι υπεύθυνοι για την ομαλή διακίνηση της ζήτησης του περιεχομένου σε peer-to-peer δίκτυα ή ακόμα και σε δίκτυα που έχουν αρκετά κοινά σημεία με τα προαναφερθέντα δίκτυα , τέτοια είναι για παράδειγμα τα game server δίκτυα [1].

2. P2P δίκτυα – γενικά χαρακτηριστικά

Τα P2P δίκτυα έχουν ξεχωρίσει σαν κατηγορία από τα υπόλοιπα λόγω της δυνατότητας τους να διανέμουν δεδομένα μεταξύ των διαφόρων γειτόνων χωρίς να υπάρχει κάποιος συγκεκριμένος κόμβος που να παίζει το ρόλο του εξυπηρετητή. Η βασική τους προϋπόθεση είναι ότι οποιαδήποτε κόμβος (γείτονας) από ένα σύνολο πανομοιότυπων κόμβων μπορεί να παρέχει στα άλλα μέλη του δικτύου τα δεδομένα που έχει , αυξάνοντας την διαθεσιμότητα των περιεχομένων καθώς και το εύρος διακίνησης τους χωρίς να απαιτείται η παρουσία οποιοδήποτε άλλος κόμβος εξυπηρέτησης. Στα δίκτυα αυτά οι καταχωρήσεις των δεδομένων τους απλώνονται σαν δίχτυ σε όλο το δίκτυο που οι κόμβοι (peer) απαρτίζουν και οι κόμβοι με την σειρά τους ψάχνουν για ένα συγκεκριμένο αποτέλεσμα. Επομένως όταν βρεθεί το επιθυμητό αποτέλεσμα στο ερώτημα του από τον κόμβο τότε θα φορτωθεί για κάποιο χρονικό διάστημα , κατά την διάρκεια αυτή όμως μπορεί να φορτωθεί και από άλλο κόμβο μέσω αυτού που το πήρε πρώτος. Έτσι έχουμε μία ταυτόχρονη download –

upload δεδομένων κατάσταση από έναν κόμβο του P2P δικτύου. Το τελευταίο γεγονός μας προτρέπει να ερευνήσουμε τις δυνατότητες του load balancing.

Τα τελευταία χρόνια λόγω της ραγδαίας ανάπτυξης τέτοιων μεγάλων υπολογιστικών συστημάτων έχει δωθεί έμφαση στην βελτίωση των ήδη υπαρχόντων δυναμικών αλγορίθμων, καθώς και στην επινόηση νέων. Υπάρχουν στατικοί και δυναμικοί αλγόριθμοι Load Balancing. Εδώ θα ήταν σκόπιμο να αναφερθεί πως κάθε αλγόριθμος έχει συνήθως ένα δικό του περιβάλλον δικτύου στο οποίο μπορεί να εφαρμοστεί , αυτό είναι εφικτό γιατί τα δίκτυα αυτά χαρακτηρίζονται από μεγάλη ανομοιογένεια εξαιτίας τη φύσης τους.
Τα περισσότερα P2P δίκτυα χρησιμοποιούν έναν DHT (Distributed Hash Table) πίνακα που αναφέρετε στα τυχαία κατανεμημένα αντικείμενα μεταξύ των γειτονικών κόμβων. Με τέτοιο τρόπο έτσι ώστε κάποιοι κόμβοι να έχουν Θ(logN) περισσότερες φορές αντικείμενα απ' ότι ο μέσος κόμβος . Η περαιτέρω ανισσοροπία μπορεί να είναι αποτέλεσμα της μη καταλληλότητας των κατανεμημένων αντικειμένων στον αναγνωρισμένο χώρο και ο υψηλός βαθμός ανομοιότητας στην φόρτωση αντικειμένων αλλά και στην κομβική χωρητικότητα[2].

Ο κυριότερος σκοπός των αλγορίθμων είναι να μοιράσουν δίκαια την ισσοροπία στα upload bandwidth των γειτόνων (peer) του δικτύου που θέλουν να εξυπηρετηθούν. Άλλωστε και η πιο μεγάλη αδυναμία των δικτύων αυτής της κατηγορίας είναι να εξασφαλίσουν στους χρήστες τους σε υψηλή τιμή το χαρακτηριστικό αυτό. Το να ισορροπηθεί το upload bandwidth αποτελεί πρόκληση για πολλούς λόγους [6].

1. Σε ένα P2P δίκτυο δεν υπάρχει κεντρικός αποστολέας (εξυπηρετητής) δεδομένων που να εκπληρώνει την ισορροπία φόρτωσης των απαιτήσεων. Κάθε κόμβος – γείτονας (peer) παίρνει ανεξάρτητα τις αποφάσεις του πάνω στο πως θα αντιστοιχίσει τις εισερχόμενες αιτήσεις σε αντίγραφα.
2. Οι κόμβοι τυπικά δεν ξέρουν τις ταυτότητες άλλων γειτονικών κόμβων στο δίκτυο και επομένως δεν μπορούν να συγχρονιστούν με εκείνους τους κόμβους.
3. Οι κόμβοι αντίγραφα σε P2P δίκτυα δεν χαρακτηρίζονται απαραιτήτως από ομοιογένεια. Κάποιοι κόμβοι αντίγραφα μπορεί να είναι ισχυροί με καλή συνδεσιμότητα ενώ άλλοι μπορεί να έχουν περιορισμένη κληρονομημένη ικανότητα να χειρίζονται απαιτήσεις περιεχομένων.

Οι προηγούμενες τεχνικές εξισορρόπησης φόρτωσης βασίζουν τη λήψη των αποφάσεων τους σε περιοδικές ή συνεχόμενες ανανεώσεις που περιέχουν πληροφορίες πάνω στη φορτωτική ή διαθέσιμη χωρητικότητα. Στη διεθνή βιβλιογραφία η πληροφορία αυτή αναφέρεται ως load balancing information (LBI). Οι προηγούμενες προσεγγίσεις

- Δεν λαμβάνουν υπόψη την ετερογένεια των γειτονικών κόμβων ή
- Χρησιμοποιούν τεχνικές όπως μετανάστευση ή μετάδοση των εργασιών που απαιτούν κλειστό συγχρονισμό ανάμεσα σε οντότητες υπηρεσίας που δεν μπορούν να επιτευχθούν σε ένα P2P περιβάλλον , ή
- Υποφέρουν από σημαντικές περιοδικές μεταβολές φόρτωσης , ή από «κατευθυνόμενη συμπεριφορά» , όπου οι γειτονικοί κόμβοι ταυτόχρονα προωθούν ένα απρόβλεπτο νούμερο απαιτήσεων σε αντίγραφα με χαμηλή αναφερόμενη φορτωτική ή υψηλή αναφερόμενη διαθέσιμη χωρητικότητα που

προκαλεί υπερφόρτωση. Αυτή η «κατευθυνόμενη συμπεριφορά» ματαιώνει την προσπάθεια παροχής load balancing.

Οι περισσότερες από αυτές τις τεχνικές βασίζονται επίσης στην επικαιρότητα των ανανεώσεων των LBI. Το ευρύ φυσικό περιβάλλον των P2P δικτύων και η ποικιλομορφία στις καθυστερήσεις μεταφοράς ανάμεσα στους γειτονικούς κόμβους καθιστά την εγγύηση της επικαιρότητας των ανανεώσεων δύσκολη. Οι γειτονικοί κόμβοι θα υπόκεινται σε πολλούς διαφορετικούς βαθμούς εμπειρίας στις LBI ανανεώσεις που λαμβάνουν εξαρτώμενοι από την απόσταση τους από την πηγή των ανανεώσεων. Επιπροσθέτως, η διατήρηση της επικαιρότητας των LBI ανανεώσεων έχει υψηλό κόστος, από την στιγμή που όλες οι ανανεώσεις πρέπει να ταξιδέψουν στο ίντερνετ για να φτάσουν στους ενδιαφερόμενους peer κόμβους. Όσο μικρότερη η ενδο-ανανεωτική περίοδος και όσο μεγαλύτερο είναι το επικαλυπτόμενο peer δίκτυο , τόσο μεγαλύτερη είναι η καθυστέρηση λόγω κίνησης του δικτύου που προκαλείται από τις LBI ανανεώσεις. Επομένως , σε ένα P2P περιβάλλον ένας αποτελεσματικός load balancing αλγόριθμος δεν θα πρέπει να είναι σημαντικά εξαρτώμενος από την επικαιρότητα των ανανεώσεων [3].

3. Πρωτόκολλα και P2P δίκτυα

Ένα βασικό πρόβλημα στα P2P συστήματα είναι η κατανομή των στοιχείων που αποθηκεύονται ή υπολογίζονται για να μεταφερθούν σε κόμβους που συμμετέχουν στο σύστημα. Για την επίλυση του προβλήματος αυτού έχουν κατά καιρούς αναπτυχθεί διάφορες προσεγγίσεις.Το DHT (Distributed Hash Table) είναι η πιο γνωστή και έχει γίνει η κλασική προσέγγιση σε αυτό το ζήτημα. Οι λόγοι είναι ότι:

- Το DHT εκπληρώνει τις υποσχέσεις μιας λειτουργίας κατακερματισμού η οποία χαρτογραφεί οποιοδήποτε δεδομένο στοιχείο σε ένα ορισμένο μηχάνημα («bucket») στο P2P δίκτυο και
- Το DHT διαφέρει από τους παραδοσιακούς πίνακες κατακερματισμού (hash tables) με δύο τρόπους:
    1. **στην εισαγωγή και διαγραφή των στοιχείων:** τα DHT εκτός από την εισαγωγή κα διαγραφή στοιχείων πρέπει να υποστηρίζουν και την εισαγωγή και διαγραφή δεσμών, ενώ μηχανές συνδέονται και αποσυνδέονται από το δίκτυο, τα στοιχεία πρέπει να «μεταναστεύουν» σε άλλα μηχανήματα και η λειτουργία κατακερματισμού πρέπει να επαναληφθεί ώστε να αντανακλά την νέα θέση.
    2. **ένα είδος πρωτοκόλλου routing είναι συνήθως απαραίτητο:** από τη στιγμή που δεν είναι εφικτό σε ένα P2P σύστημα για κάθε κόμβο να διατηρήσει επίκαιρη ενημέρωση για κάθε άλλο κόμβο του συστήματος, ένα αντικείμενο αναζητείται ή εισάγεται ακολουθώντας μια συνέχεια από επαναλαμβανόμενα πηδήματα μέσα στο P2P δίκτυο.

Κάποια από τα πρωτόκολλα που έχουν προταθεί για τα P2P δίκτυα στηρίζονται σε δομές δεδομένων με “consistent hashing” οι οποίες αποτελούν τη βασική εξήγηση για το CHORD (και Koorde) P2P δίκτυα. Ο όρος “consistent hashing” ορίζεται ως ένα στιγμιότυπο του DHT παράδειγμα για την ανάθεση στοιχείων σε κόμβους σε ένα P2P

σύστημα : Στοιχεία και κόμβοι χαρτογραφούνται σε ένα συνηθισμένο διάστημα διευθυνσιοδότησης και οι κόμβοι πρέπει να αποθηκεύουν όλα τα στοιχεία που είναι διανεμημένα κοντά στο διάστημα διευθυνσιοδότησης.
Δύο από τα πρωτόκολλα που έχουν προταθεί για την επίλυση των προβλημάτων αυτών είναι τα εξής :

Το πρώτο πρωτόκολλο ισορροπεί την κατανομή του διαστήματος διευθύνσεων των κλειδιών σε κόμβους, η οποία αποδίδει ένα Load balanced σύστημα όταν το DHT χαρτογραφεί τα στοιχεία με τυχαίο τρόπο μέσα στο χώρο διευθύνσεων. Αυτό, αποδίδει το πρώτο P2P σχήμα ενώ ταυτόχρονα επιτυγχάνει O(log n) βαθμούς πολυπλοκότητας, O(log n) βελτιωμένο κόστος και σταθερό παράγοντα εξισορρόπησης φόρτωσης.
Το δεύτερο πρωτόκολλο στοχεύει στην απευθείας εξισορόπηση της κατανομής στοιχείων μεταξύ των κόμβων. Αυτό είναι χρήσιμο όταν η κατανομή των στοιχείων στο χώρο διευθύνσεων δε μπορεί να τυχαιοποιηθεί [2].

ΜΕΡΟΣ Β'
LOAD BALANCING – ΑΛΓΟΡΙΘΜΟΙ

1. Το Load Balancing ως πρόβλημα

Το πρόβλημα της εξισορρόπησης φόρτωσης έχει συζητηθεί στη βιβλιογραφία των παραδοσιακά κατανεμημένων συστημάτων για περισσότερες από δύο δεκαετίες. Ποικίλες στρατηγικές και αλγόριθμοι έχουν προταθεί, υλοποιηθεί και αρχειοθετηθεί σε έναν αριθμό μελετών. Οι αλγόριθμοι εξισορρόπησης φόρτωσης μπορούν να αρχειοθετηθούν σε δύο κατηγορίες: στατικοί ή δυναμικοί. Στους στατικούς αλγορίθμους οι αποφάσεις που συνδέονται με την εξισορρόπηση φόρτωσης λαμβάνονται στο χρόνο μεταγλώττισης όταν υπολογίζονται οι απαιτήσεις των πόρων. Ένας workstation υπολογιστής με δυναμική εξισορρόπησης φόρτωσης κατανέμει και επανακατανέμει τους πόρους στο χρόνο εκτέλεσης βασισμένος σε όχι εκ των προτέρων πληροφορία αποστολών που μπορεί να διευκρινίσει πότε και ποιών οι αποστολές μπορούν να «μεταναστεύσουν».
Πρόσφατα, έχουν αναφερθεί από τον Houle (και άλλους της ομάδας του) αλγόριθμοι για στατική εξισορρόπηση φόρτωσης σε δέντρα, υποθέτοντας ότι η συνολική φόρτωση είναι σταθερή. Σε αντίθεση με τα παραδοσιακά κατανεμημένα συστήματα για τα οποία μια πληθώρα αλγορίθμων έχει προταθεί για τους δικτυακούς υπολογισμούς. Αυτό οφείλεται στην καινοτομία και τα ιδιαίτερα χαρακτηριστικά αυτής της υποδομής. Οι αλγόριθμοι εξισορρόπησης φόρτωσης διακρίνονται από την υλοποίηση τους στις παρακάτω πολιτικές:

- Information Policy: καθορίζει ποια πληροφορία περιβάλλοντος εργασίας θα συγκεντρωθεί, πότε και πού
- Triggering Policy: διευκρινίζει την κατάλληλη περίοδο για να ξεκινήσει μια λειτουργία εξισορρόπησης φόρτωσης
- Resource Type Policy: Ταξινομεί έναν πόρο ως server ή receiver αποστολών ανάλογα με την κατάσταση διαθεσιμότητας του.
- Location Policy: χρησιμοποιεί τα αποτελέσματα της resource type policy για να βρει έναν κατάλληλο σύντροφο για server ή receiver.

- Selection Policy: διευκρινίζει, ορίζει τις αποστολές που πρέπει να «μεταναστεύσουν» από τους υπερφορτωμένους πόρους (source) στους πιο αδρανείς (receiver).

Οι πλέον προτεινόμενοι αλγόριθμοι εξισορρόπησης φόρτωσης αναπτύχθηκαν ιδεατά υποθέτοντας ένα σύνολο από sites συνδεδεμένο με ομοιογενή και γρήγορα δίκτυα. Αν και αυτή η υπόθεση είναι αληθής σε παραδοσιακά κατανεμημένα συστήματα δεν είναι ρεαλιστική σε αρχιτεκτονικές δικτύων εξαιτίας των παρακάτω ιδιαιτεροτήτων που τα χαρακτηρίζουν:

1. Ετερογένεια : Ένα δίκτυο περιλαμβάνει πολλαπλούς πόρους που είναι ετερογενείς στη φύση τους και μπορεί να αναπτύξουν πολλούς διοικητικούς χώρους κυριαρχίας σε παγκόσμια έκταση.
2. Κλιμάκωση : Ένα δίκτυο μπορεί να αναπτυχθεί από λίγους σε εκατομύρια πόρους αυτό δημιουργεί το πρόβλημα της πιθανής υποβάθμισης της αποδοτικότητας όσο το δίκτυο μεγαλώνει.
3. Προσαρμοστικότητα : Σε ένα δίκτυο μία αποτυχία πόρου είναι ο κανόνας όχι η εξαίρεση, αυτό σημαίνει ότι η πιθανότητα να αποτύχει , να πέσει κάποιος πόρος είναι φυσικά υψηλή. Οι διαχειριστές πόρων πρέπει να προσαρμόσουν την συμπεριφορά τους δυναμικά ώστε να εξάγουν την υψηλότερη απόδοση από τους διαθέσιμους πόρους και υπηρεσίες.

Αυτές οι ιδιότητες κάνουν το πρόβλημα εξισορρόπησης φόρτωσης πιο πολύπλοκο από ότι τα παραδοσιακά κατανεμημένα συστήματα τα οποία προσφέρουν ομοιογένεια και σταθερότητα στους πόρους τους. Επίσης τα διασυνδεδεμένα δίκτυα πάνω σε δίκτυα υπολογιστών έχουν πολύ ανόμοιες αποδόσεις και οι αποστολές που υποβάλλονται στο σύστημα μπορεί να είναι πολυδιάστατες και παράξενες. Αυτές οι πολλές και διαφορετικές παρατηρήσεις δείχνουν ότι είναι πολύ δύσκολο να οριστεί ένα σύστημα εξισορρόπησης φόρτωσης που να ενοποιεί όλους αυτούς τους παράγοντες.

Παράγοντες εκτέλεσης

Ο κύριος στόχος των εξισορρόπησης φόρτωσης μεθόδων είναι να επιταχύνει την εκτέλεση των εφαρμογών στους πόρους των οποίων το περιβάλλον εργασίας ποικίλει σε χρόνο εκτέλεσης με απρόβλεπτο τρόπο. Γι’ αυτό το λόγο είναι σημαντικό να ορίσουμε μετρικές και κανόνες ώστε να εξασφαλίσουμε την σωστή μέτρηση του περιβάλλοντος εργασίας , πάνω στο οποίο θα εφαρμοστούν και οι αλγόριθμοι που θα προσπαθήσουν να επιλύσουν το πρόβλημα που παρουσιάστηκε στην προηγούμενη ενότητα.
Κάθε δυναμική μέθοδος εξισορρόπησης φόρτωσης μπορεί να υπολογίζει εγκαίρως τις πληροφορίες και το φόρτο εργασίας για κάθε πόρο. Αυτό είναι ένα κλειδί πληροφορίας σε ένα σύστημα εξισορρόπησης φόρτωσης όπου απαντήσεις δίνονται στα παρακάτω ερωτήματα:

1. Πως να μετρήσω το φόρτο εργασίας σε ένα πόρο
2. Τι κριτήρια εξασφαλίζουν ένα περιβάλλον ως το ιδανικό για περιβάλλον εργασίας

3. Πως αποφεύγονται οι αρνητικές επιδράσεις της δυναμικότητας των πόρων στο περιβάλλον εργασίας και
4. Πως να λάβουμε υπ' όψη την ετερογένεια των πόρων ώστε να κερδίσουμε ένα στιγμιαίο μέσο αντιπροσωπευτικό όρο του φόρτου εργασίας στο σύστημα;

Πολλοί δείκτες φόρτωσης έχουν προταθεί στην βιβλιογραφία όπως το μέγεθος (μήκος) ουράς της CPU , ο μέσος όρος του μήκους της ουράς της CPU , ο βαθμός χρήσης της CPU , κτλ. Η επιτυχία ενός αλγορίθμου load balancing εξαρτάται από την σταθερότητα του αριθμού των μηνυμάτων , το περιβάλλον υποστήριξης , το χαμηλό κόστος ανανέωσης του περιβάλλοντος εργασίας και το μέσο χρόνο απόκρισης που είναι σημαντική μέτρηση για έναν χρήστη. Είναι επίσης απαραίτητο να μετρηθεί και το κόστος επικοινωνίας που προωθείται από μία λειτουργία της εξισορρόπησης φόρτωσης [3].

2. P2P δίκτυα και αλγόριθμοι load balancing

Στατικοί αλγόριθμοι load balancing [5]

Στην διεθνή βιβλιογραφία υπάρχουν αρκετοί αλγόριθμοι που χειρίζονται το πρόβλημα ισορροπίας φόρτωσης με στατικό τρόπο. Παρακάτω θα γίνει αναφορά σε εκείνους που κρίθηκαν ως οι πιο κατάλληλοι να εφαρμοστούν σε δίκτυα , χωρίς να προϋποθέτουν πολλές συνθήκες για την εφαρμογή τους.
Παρακάτω θα αναφερθούν μερικοί από τους πιο γνωστούς αλγορίθμους της κατηγορίας αυτής ,μερικοί από αυτούς ακόμα και σήμερα εφαρμόζονται σε P2P δίκτυα ως αξιόπιστες λύσεις , αν και τα τελευταία χρόνια υπάρχει μία μεγάλη στροφή στην εφαρμογή δυναμικών αλγορίθμων .

Ο πιο απλός ονομάζεται one–to–one , κάθε «ελαφρώς» φορτωμένος κόμβος v περιοδικά επικοινωνεί με τυχαίους κόμβους w . Αν w είναι «βαριά» φορτωμένος , εικονικές μονάδες μεταφέρονται από τον w στον v έτσι ώστε ο τελευταίος να μην υπερφορτωθεί και ο πρώτος να μπει σε μια φυσιολογική κατάσταση.
Ένας άλλος αλγόριθμος απλός και αυτός στην τεχνική του ονομάζεται one–to–many , επιτρέπει τους υπερφορτωμένους κόμβους να ελέγχουν περισσότερους από έναν γειτονικούς κόμβους στην μονάδα του χρόνου. Ένας υπερφορτωμένος κόμβος h εξετάζει την φόρτωση σε ένα σύνολο «ελαφρών» κόμβων επικοινωνώντας με ένα τυχαίο κατάλογο – κόμβο, όπου το σύνολο των «ελαφρών» κόμβων στέλνει τις πληροφορίες ισορροπίας φόρτωσης (LBI). Μερικοί από τους h εικονικούς κόμβους μεταφέρουν πληροφορίες σε έναν ή περισσότερους κόμβους που ανήκουν στον ίδιο κατάλογο - κόμβο.
Ένας άλλος αλγόριθμος είναι ο many – to – many όπου κάθε κατάλογος περιέχει πληροφορίες φόρτωσης για ένα σύνολο «ελαφρών» και «βαρέων» κόμβων. Ο αλγόριθμος τρέχει σε κάθε κατάλογο και αποφασίζει την ξανά-αντιστοίχιση των εικονικών κόμβων από τους υπερφορτωμένους κόμβους που είναι εγγεγραμμένοι σε ένα κατάλογο στους ελαφρούς κόμβους που είναι εγγεγραμμένοι σε κάποιον άλλο κατάλογο.

Δυναμικοί αλγόριθμοι load balancing

Γενικά στην βιβλιογραφία αναφέρεται πως οι αλγόριθμοι που προσπαθούν να λύσουν το πρόβλημα της ισορροπίας φόρτωσης πρέπει να πληρούν έστω και σε κάποιο βαθμό τους παρακάτω στόχους [5]:

1. *Ελαχιστοποίηση της ανισορροπίας φόρτωσης:* Για να παρέχεται η μεγαλύτερη ποιότητα υπηρεσίας , κάθε κόμβος θα πρέπει να έχει τον ίδιο βαθμό χρήσης.
2. *Ελαχιστοποίηση της συνολικής κίνησης της φόρτωσης :* Μεταφέροντας μεγάλες «ποσότητες» δεδομένων εξαιτίας της φόρτωσης χρησιμοποιείται εξίσου μεγάλο bandwidth και μπορεί να είναι μη πρακτικό εάν η φόρτωση ενός κόμβου αλλάζει γρήγορα σε σχέση με το χρόνο που χρειάζεται να κινηθούν τα αντικείμενα.

Οι αλγόριθμοι που χειρίζονται με δυναμικό τρόπο το πρόβλημα της ισορροπίας φόρτωσης στηρίζονται κατά βάση στον DHT πίνακα για τον οποίο έγινε λόγος σε προηγούμενε ενότητες. Εξαιτίας της φύσης του προβλήματος (load balancing) και την καθυστέρηση στην ανάπτυξη μεγάλων δικτυακών υπολογιστικών συστημάτων , όπου το πρόβλημα της ισορροπίας φόρτωσης κάνει την εμφάνιση του με μεγαλύτερη έμφαση , οι δυναμικοί αλγόριθμοι άρχισαν να ερευνώνται. Οι δυναμικοί αλγόριθμοι έχουν το σημαντικό μειονέκτημα σε σχέση με τους στατικούς ότι για να εφαρμοστούν σε κάποιο δίκτυο πρέπει να συντρέχουν πολλές προϋποθέσεις, και μάλιστα πολλές φορές «τρέχουν» σε ένα και μόνο τύπο δικτύου [6].

<u>*Avail-Cap : Allocation Proportional to Available Capacity :*</u>

Σε αυτόν τον αλγόριθμο κάθε κόμβος επιλέγει να προωθήσει μία αίτηση σε έναν όμοιο με αυτό στην κατάσταση κόμβο με την πιθανότητα αναλογίας να είναι κόμβος με διαθέσιμη χωρητικότητα.

Διαθέσιμη χωρητικότητα είναι η μέγιστη εκτιμημένη αίτηση που μπορεί να αγγίξει την μικρότερη φόρτωση από την εμπειρία (ιστορικό) του κόμβου. Ο αλγόριθμος αυτός λαμβάνει υπ ‘όψη του την ετερογένεια επειδή διακρίνεται μεταξύ των κόμβων που έχουν στο ιστορικό τους την ίδια φόρτωση αλλά έχουν διαφορετικές μέγιστες χωρητικότητες. Είναι ένας από τους «διαισθητικούς» αλγορίθμους καθώς στέλνει περισσότερες αιτήσεις στους κόμβους που είναι όμοιοι που είναι διαθέσιμοι , από όσες αυτοί μπορούν να δεχτούν και να ελέγξουν. Οι κόμβοι που υπερφορτώνονται δίνουν αναφορά ότι η διαθέσιμη χωρητικότητα τους είναι μηδέν και βγαίνουν εκτός της διαδικασίας μέχρι ωσότου ξανά δώσουν αναφορά και έχουν κάποια ελεύθερη διαθεσιμότητα.

Υπάρχουν ωστόσο και παραλλαγές του ανωτέρου αλγορίθμου που σε πραγματικό επίπεδο έχει διαπιστωθεί πως φέρνει καλύτερα αποτελέσματα ως προς τον χρόνο , αλλά όπως και άλλοι δυναμικοί αλγόριθμοι περιορίζεται σε συγκεκριμένο περιβάλλον εργασίας. Ένας από αυτούς είναι και ο

<u>*Max – Cap : Allocation Proportional to Maximum Capacity:*</u>

Σε αυτόν τον αλγόριθμο κάθε κόμβος επιλέγει να προωθήσει την αίτηση σε έναν όμοιο με αυτόν κόμβο στην κατάσταση με την πιθανότητα αναλογίας να είναι ο κόμβος με την μέγιστη χωρητικότητα. Η μέγιστη χωρητικότητα στηρίζεται στην δυνατότητα του κάθε κόμβου να χειρίζεται κάποιο αριθμό αιτήσεων στην μονάδα του χρόνου. Η φόρτωση , η διαθέσιμη και μέγιστη χωρητικότητα δεν επηρεάζεται από τις αλλαγές των αιτήσεων στο περιβάλλον εργασίας. Ο αλγόριθμος αυτός δεν επηρεάζεται από τις ανανεώσεις των πληροφοριών που προέρχονται από το

πρόβλημα της ισορροπίας φόρτωσης (LBI). Στην πραγματικότητα οι κόμβοι «σπρώχνουν» τις ανανεώσεις πληροφοριών όταν επιλέγουν να δημοσιεύσουν την νέα μέγιστη χωρητικότητα τους. Η επιλογή τους αυτή εξαρτάται από μη συσχετιζόμενους και ανεξάρτητους παράγοντες ως προς το περιβάλλον εργασίας. Εάν για κάποιο τυχαίο λόγο x οι κόμβοι αλλάξουν συμπεριφορά τότε ο αλγόριθμος τους φέρνει σε κατάσταση υπερφόρτωσης ώστε να μην μπορούν να εκτελέσουν κάποια λειτουργία μέχρι να επανέλθει το σύστημα στην μορφή που ο αλγόριθμος «επιθυμεί».

Οι περισσότεροι δυναμικοί αλγόριθμοι κάνουν υποθέσεις για το περιβάλλον εργασίας , γιατί στην ουσία κανείς δεν μπορεί να ελέγξει την ετερογένεια του συστήματος εξ' ολοκλήρου. Παρακάτω θα αναφερθεί ένας αλγόριθμος [5] που παρουσιάζει ιδιαίτερη σημασία καθώς μπορεί να εφαρμοστεί σε όλα εκείνα τα δίκτυα που παρουσιάζουν ετερογένεια και ανήκουν στην ομάδα δομημένων δικτύων που ονομάζονται P2P.
Κάποιες από τις προϋποθέσεις που έχει ο αλγόριθμος για το περιβάλλον εργασίας είναι :

1. Συνέχεια εισάγονται και διαγράφονται μονάδες δεδομένων
2. Κόμβοι με διαφορετικές χωρητικότητες συγχωνεύονται και απομακρύνονται από το σύστημα συνέχεια
3. Η κατανομή και το μέγεθος των μονάδων δεδομένων μπορούν να παρεκκλίνουν.

Ο αλγόριθμος χρησιμοποιεί την έννοια των εικονικών εξυπηρετητών (virtual servers). Ένας virtual server αντιπροσωπεύει έναν κόμβο στο DHT του οποίου οι πληροφορίες μπορούν να αποθηκεύονται σε εικονικό επίπεδο το οποίο διαφέρει από το φυσικό επίπεδο. Ένας φυσικός κόμβος μπορεί να έχει περισσότερους από έναν εικονικούς κόμβους.
Η βασική ιδέα του αλγορίθμου είναι η αποθήκευση των LBI πληροφοριών από τους γειτονικούς κόμβους σε πλήθος καταλόγων οι οποίοι περιοδικά σχεδιάζουν ξανά την αντιστοίχιση των εικονικών κόμβων ώστε να πετύχει καλύτερη ισορροπία. Κάθε κατάλογος έχει ένα γνωστό ID γνωστό σε όλους τους κόμβους και είναι αποθηκευμένο στον κόμβο που είναι υπεύθυνος για το κατάλογο αυτό. Η ιδιοκτησία του καταλόγου μπορεί να αλλάξει καθώς αλλάζει η κατανομή των κόμβων στο χώρο , που συμβαίνει εξαιτίας των διαφόρων λειτουργιών για ισορροπία φόρτωσης του συστήματος. Οποιοσδήποτε κόμβος μπορεί να επικοινωνήσει με οποιοδήποτε κατάλογο μέσω του πρωτοκόλλου DHT. Υπόθεση του αλγορίθμου ότι ο αριθμός των καταλόγων είναι συγκεκριμένος.
Παρακάτω δίνεται ο ψευδοκώδικας του αλγορίθμου που τρέχει σε κάθε κόμβο:

Node(time period T, threshold k_e)

- *Initialization*:
  (1) $d \leftarrow$ `RandomDirectory()`
  (2) $m_n \leftarrow \lfloor m \cdot c_n/\tilde{c}_d + 1/2 \rfloor$, where $\tilde{c}_d$ is the average capacity of nodes reporting to d
  (3) Instantiate m_n virtual servers at random IDs
  (4) Send $(c_n, \{\ell_{v_1}, \ldots, \ell_{v_{m_n}}\})$ to d
- *Emergency action*: When un jumps above k_e:
  (1) Repeat up to twice while $u_n > k_e$:
  (2) Send $(c_n, \{\ell_{v_1}, \ldots, \ell_{v_m}\})$ to d
  (3) `PerformTransfer` (v, n') for each transfer $v \rightarrow n'$ scheduled by d
  (4) $d \leftarrow$ `RandomDirectory()`
- *Periodic action*: Upon receipt of list of transfers from a directory:
  (1) `PerformTransfer`(v, n') for each transfer $v \rightarrow n'$
  (2) Report $(c_n, \{\ell_{v_1}, \ldots, \ell_{v_m}\})$ to `RandomDirectory()`

Η διαδικασία RandomDirectory() στον παραπάνω ψεύδοκώδικα επιλέγει δύο τυχαίους καταλόγους και επιστρέφει αυτόν με τους λιγότερους κόμβους που έχουν αναφερθεί στην τελευταία περίοδο ισορροπίας. Αυτό μειώνει την ανισορροπία του αριθμού των κόμβων που αναφέρονται στους καταλόγους. PerformTransfer(v,n΄) μεταφέρει εικονικούς κόμβους v σε κόμβους n΄ αν δεν υπάρχουν υπερφορτωμένοι n΄.

Σε κάθε κατάλογο τρέχει ο παρακάτω αλγόριθμος:

Directory(time period T, thresholds k_e, k_p)

- *Initialization*: $I \leftarrow \{\}$
- *Information receipt and emergency balancing*: Upon receipt of $J = (c_n, \{\ell_{v_1}, \ldots, \ell_{v_m}\})$ from node n:
  (1) $I \leftarrow I \cup J$
  (2) If $u_n > k_e$:
  (3) reassignment $\leftarrow$ `ReassignVS`(I, k_e)
  (4) Schedule transfers according to *reassignment*
- *Periodic balancing*: Every T seconds:
  (1) *reassignment* $\leftarrow$ `ReassignVS`(I, k_p)
  (2) Schedule transfers according to *reassignment*
  (3) While average number of virtual servers per node in I is >1.25 m or <0.75 m, remove the least-loaded virtual server or split the largest-loaded virtual server in half, respectively.
  (4) $I \leftarrow \{\}$

Η υπορουτίνα ReassignVS , δίνει ένα όριο k και η πληροφορία της φόρτωσης I αναφέρεται σε έναν κατάλογο , υπολογίζει και ξανά αντιστοιχίζει τους εικονικούς servers από τους κόμβους με βαθμό χρήσης μεγαλύτερη από k σε αυτούς με βαθμό χρήσης μικρότερη από k.

ReassignVS(Load & Capacity information I , threshold k)

(1) pool ← {}
(2) For each node n E I , while $l_n/c_n > k$, remove the virtual server on n with highest ratio of load to movement cost , and move it to pool.

(3) For each virtual server v E pool , from heaviest to lightest, assign v to the node n which minimizes $(l_n + l_v) / c_n$.
(4) Return the virtual server reassignment

Αλγόριθμοι σε Game Server δίκτυα [4]

Τα game server δίκτυα , αποτελούνται από έναν τεράστιο αριθμό χρηστών οι οποίοι μπορούν να παίζουν ένα κοινό παιχνίδι μέσω ενός αλληλοσυνδεμένου δικτύου. Τα διαδικτυακά παιχνίδια έχουν γίνει αρκετά δημοφιλή στην κοινωνία των gamers , οπότε και κρίθηκε απαραίτητο να δημιουργηθούν κατάλληλη αλγόριθμοι για την εξισορρόπηση φόρτωσης σε αυτά τα νέου είδους δίκτυα ,που μοιάζουν αρκετά θα λέγαμε στα P2P δίκτυα.
Για να είναι ένα τέτοιο δίκτυο αποτελεσματικό , κλιμακωτό και δυνατό σε κάθε κατάσταση , το κατανεμημένο σύστημα (τα κατανεμημένα συστήματα αποτελούν την πιο πρακτική λύση για τέτοια δίκτυα εξαιτίας της διαθεσιμότητας των υψηλών αποδόσεων των Η/Υ και την υψηλή ταχύτητα των LAN με χαμηλό σχετικά κόστος.) θα πρέπει να σχεδιαστεί προσεκτικά εξαιτίας των κληρονομικών περιορισμών. Οι περιορισμοί αυτοί μπορεί να είναι η έλλειψη μνήμης και η υψηλή πιθανότητα αποτυχίας των Η/Υ που λαμβάνουν μέρος.

Έχουν αναφερθεί δύο είδη network game server:

- *S-Style*: Χαρακτηρίζετε από τον μεγάλο αριθμό περιοχών-περιόδων παιχνιδιού όπου σε κάθε game-session μπορούν να συμμετέχουν μέχρι το πολύ 8 χρήστες.
- *Uo-Style*: Χαρακτηρίζετε από μία μεγάλη περιοχή – περίοδο παιχνιδιού με χιλιάδες χρήστες.

Ο αλγόριθμος που έχει προταθεί , εφαρμόζετε κυρίως σε συστήματα που χαρακτηρίζονται από το *Uo-Style*, για τέτοιου είδους συστήματα έχει δύο κύριες προϋποθέσεις:

- Πως θα χωριστεί το σύνολο του παιχνιδιού σε υποσύνολα τα οποία σχετίζονται με τον αριθμό των servers που συμμετέχουν. Η στρατιγική των κενών υποσυνόλων πρέπει να είναι απλή και αρκετά αποτελεσματική ώστε να ελαχιστοποιείται ο μέσος χρόνος αλληλεπίδρασης των χρηστών που περιμένουν να εισέλθουν στο παιχνίδι.
- Πως θα διανεμηθεί ο έλεγχος της ισορροπίας φόρτωσης εργασιών , έτσι ώστε να είναι κλιμακωτή χωρίς να υπολήπτεται των αριθμό των χρηστών ή τον αριθμό των επεξεργαστών . Κάθε game processing server μπορεί να πάρει την απόφαση πότε θα κατανεμηθεί το δικό του φορτίο δεδομένων με μερικές πληροφορίες που έχει συλλέξει από μικρό αριθμό γειτονικών server.

Επίσης λαμβάνει υπόψη του την γεωγραφική σχέση ανάμεσα στις μονάδες του παιχνιδιού και τον μικρότερο χρόνο απόκρισης της συχνότητας αλληλεπιδράσεων των χρηστών.

Για να γίνει περισσότερο κατανοητή η περιγραφή του αντιστοιχούμε κάθε game processing server σε ένα κόμβο. Ο κόμβος μπορεί να βρεθεί στις παρακάτω πέντε καταστάσεις:

1. *Normal*: Όταν ο κόμβος χειρίζεται τις εργασίες του σε κανονική φάση με μεσαία φόρτωση. Αν η τιμή της φόρτωσης του ξεπεράσει μία σταθερή τιμή για το δίκτυο τότε μεταβαίνει στην επόμενη κατάσταση της υπερφόρτωσης.
2. *Overloaded*: Όταν η τιμή φόρτωσης του κόμβου είναι μεγαλύτερη από την τιμή της overload – threshold τότε γίνεται αποστολέας και προσπαθεί να μεταφέρει μονάδες φόρτου εργασίας σε άλλους γειτονικούς κόμβους που μπορούν να τα δεχτούν.
3. *Locked*: Όταν ένας κόμβος βρίσκετε στην 2 κατάσταση και λάβει μηνύματα από γειτονικούς κόμβους ότι είναι υπερφορτωμένοι τότε ο κόμβος αυτός κλειδώνει επειδή δεν μπορεί να μειώσει το φόρτο εργασίας του άμμεσα.
4. *Underloaded*: Όταν ο κόμβος έχει μικρότερη τιμή από την τιμή της underload – threshold . Σε αυτήν την κατάσταση ο κόμβος μπορεί να λάβει από άλλους κόμβους που είναι υπερφορτωμένοι.
5. *Unknown*: Είναι ο κόμβος που δεν έχει ελεγχθεί ακόμα.

Περιγραφή δυναμικού αλγορίθμου

Κάθε κόμβος ελέγχει περιοδικά την κατάσταση στην οποία βρίσκεται και την ανανεώνει σύμφωνα με το αποτέλεσμα της σύγκρισης με την τιμή της μεταβλητής threshold. Εάν είναι overloaded κόμβος στέλνει στον δεξί και αριστερό του γείτονα το μήνυμα για έλεγχο φόρτωσης , για να ελέγξει εάν μπορεί να στείλει σε αυτούς τις μονάδες που του προκαλούν υπερφόρτωση.
Στον αλγόριθμο βασική λειτουργία είναι να αποφασίζει τι είδους μύνημα θα λάβει ο κόμβος. Εάν ο κόμβος λάβει μήνυμα για load-checking ελέγχει την κατάσταση του και στέλνει πίσω ένα μήνυμα απάντησης (ack_load_checking), που περιέχει ώς δεδομένο την κατάσταση του κόμβου.
Εάν ο κόμβος λάβει ένα ack_load_checking μήνυμα και ο κόμβος που το έστειλε μπορεί να λάβει μία δέσμη μονάδων χωρίς να αλλάξει την φυσιολογική κατάσταση στην οποία βρίσκεται τότε ο κόμβος στέλνει τις μονάδες σε αυτόν που μπορεί να τις δεχτεί. Εάν ο κόμβος «καταλάβει» από τα μηνύματα που δέχεται ότι οι γειτονικοί του κόμβοι είναι υπερφορτωμένοι τότε στέλνει remote-help μήνυμα στους γειτονικούς του κόμβους και αυτοί το προωθούν στους γειτονικούς τους κόμβους μέχρι να βρεθεί κάποιος κόμβος που δεν είναι υπερφορτωμένος. Όταν βρεθεί ο κόμβος αυτός στέλνει απαντητικό μήνυμα προς τα πίσω (ack_remote_help) με την σειρά των κόμβων όπως σε αυτό έφτασε. Αυτό γίνεται ώστε να κρατηθεί ο όρος του κατανεμημένου έλεγχου. Όταν το ack_remote_help μήνυμα φτάσει στον αρχικό κόμβο αρχίζει η μεταφορά μονάδων εργασίας στον κόμβο που απάντησε.
Η κύρια λειτουργία του αλγορίθμου είναι να χειρίζεται μονάδες και γεγονότα σε ουρές.

ΜΕΡΟΣ Γ’
ΠΡΟΣΟΜΟΙΩΣΗ - ΠΡΟΟΠΤΙΚΕΣ

1. Προσομοίωση αλγορίθμων

Για να αποτιμήσουμε την αποτελεσματικότητα και την πρακτικότητα των αλγορίθμων που αναφέρθηκαν στην προηγούμενη ενότητα καλό είναι να τους αναπτύξουμε σε έναν προσομοιωτή δικτύου (π.χ. NS-2 ή κάποιος προσομοιωτής σε java). Ο προσομοιωτής μπορεί να έχει τις παρακάτω πιθανές ομάδες παραμέτρων:

1. *CE's παράμετροι* (CE : Computing Elements) : Αυτές οι παράμετροι δίνουν πληροφορίες για τα διαθέσιμα CE κατά την διάρκεια της load balancing περιόδου όπως ο αριθμός των sites , ο αριθμός των CE σε κάθε site , οι ταχύτητες των CE , ημερομηνία αποστολής φόρτου εργασίας από τα CE , παράγοντες ανοχής.
2. *Παράμετροι εργασιών* (tasks) : Αυτές οι παράμετροι περιλαμβάνουν των αριθμό των εργασιών που βρίσκονται στην ουρά σε κάθε CE , την ημερομηνία υποβολής της κάθε εργασίας , ο αριθμός των εντολών για κάθε εργασία , το μέγεθος της εργασίας , την προτεραιότητα της κάθε εργασίας.
3. *Παράμετροι δικτύου*: Το μέγεθος του bandwidth
4. *Workload index* : Ως περιβάλλον εργασίας για τα CE χρησιμοποιήθηκε ο μέσος όρος απασχόλησης του workload =inst/speed , όπου το inst είναι ο συνολικός αριθμός των εντολών που βρίσκονται στην ουρά σ' ένα δεδομένο σύνολο CE και speed είναι η ταχύτητα.
5. *Παράμετροι εκτέλεσης*: Αυτές μπορεί να είναι ο μέσος χρόνος απόκρισης των εργασιών και το κόστος επικοινωνίας.

Είναι σωστό εδώ να σημειωθεί πως κάθε αλγόριθμος έχει το δικό του περιβάλλον εργασίας και ορισμένες προϋπόθεσης κάτω από τις οποίες είναι αποδοτικός. Οι παραπάνω παράμετροι που αναφέρθηκαν αποτελούν μία γενική εικόνα των παραμέτρων που μπορούν να ισχύσουν στον εκάστοτε αλγόριθμό , χωρίς αυτό να σημαίνει ότι είναι και οι μοναδικές [3].

2. Προοπτικές του προβλήματος load balancing

Στο μέλλον προβλέπεται να υπερισχύσουν οι δυναμικοί αλγόριθμοι έναντι των στατικών καθώς χειρίζονται με καλύτερο και περισσότερο οικονομικό τρόπο την πληροφορία της εξισορρόπησης φόρτου. Το βασικό μειονέκτημα των δυναμικών αλγορίθμων (συγκεκριμένα περιβάλλοντα εργασίας) θα ξεπεραστεί με νέες τεχνολογίες που εφαρμόζονται και βρίσκονται σε πειραματικό στάδιο , όπως ο δυναμικός DHT πίνακας. Σύντομα αναμένονται από την διεθνή κοινότητα τα πρώτα αποτελέσματα των ερευνητών για τους δυναμικούς αλγορίθμους που δεν υπόκεινται σε περιορισμούς περιβάλλοντος. Επίσης η τεχνολογία των εικονικών κόμβων (virtual servers) είναι κάτι που μπορεί σημαντικά να προωθήσει την επιστημονική κοινότητα στην βελτίωση ήδη υπαρχόντων δυναμικών αλγορίθμων.
Θα πρέπει επίσης να σημειωθεί ότι το πρόβλημα του load balancing έχει επάξια αντιμετωπιστεί και από την μεριά του hardware.

Βιβλιογραφία – Πηγές

1. Δικτυακός τόπος http://www.loadbalancing.net
2. David R. Karger , Mathias Ruhl , "Simple efficient load balancing algorithms for peer-to-peer systems", ACM SPAA , 2004
3. Belabbas Yagoubi , Yahya Slimani , "Dynamic load balancing strategy for grid computing" , ENFORMATIKA TRANSACTIONS ON ENGINEERING , COMPUTING AND TECHNOLOGY VOLUME 13 MAY 2006 ISSN 1305-5313
4. Dugki Min , Eunmi Choi , Donghoon Lee , Byungseok Park , "A load balancing algorithm for a distributed multimedia game server architecture" , IEEE , 1999
5. Sonesh Surana , Brighten Godfrey , Karthik Lakshminarayanan , Richard Karp , Ion Stoica , "Load balancing in dynamic structured peer-to-peer systems" , Elsevier B. V. Performance Evaluation 63 (2006) 217-240, February 2005 ,
6. Mema Roussopoulos , Mary Baker , "Practical load balancing for content requests in peer-to-peer networks" , Springer Distrib. Comput. (2006) 18 (6):421 - 434 , 2006