Εθνικο Μετσοβιο Πολυτεχνειο

Σχολη Ηλεκτρολογων Μηχανικων και Μηχανικων Υπολογιστων

Τομεας Επικοινωνιων, Ηλεκτρονικης και Συστηματων Πληροφορικης

Υλοποίηση αλγορίθμου εξισορρόπησης φορτίου μεταξύ τοπικών υποδομών υπολογιστικού νέφους

Διπλωματικη Εργασια

του

ΣΠΑΘΑΡΑΚΗ ΔΗΜΗΤΡΙΟΥ

Επιβλέπων: Συμεών Παπαβασιλείου
Καθηγητής Ε.Μ.Π.

Αθήνα, Φεβρουάριος 2018

Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών
Τομέας Επικοινωνιών, Ηλεκτρονικής και Συστημάτων Πληροφορικής

Υλοποίηση αλγορίθμου εξισορρόπησης φορτίου μεταξύ τοπικών υποδομών υπολογιστικού νέφους

Διπλωματικη Εργασια

του

ΣΠΑΘΑΡΑΚΗ ΔΗΜΗΤΡΙΟΥ

Επιβλέπων: Συμεών Παπαβασιλείου
Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 26η Μαρτίου 2018.

(Υπογραφή)	*(Υπογραφή)*	*(Υπογραφή)*
.....	.....	.....
Συμεών Παπαβασιλείου	Θεοδώρα Βαρβαρίγου	Ιωάννα Ρουσσάκη
Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.

Αθήνα, Φεβρουάριος 2018

(Υπογραφή)

.....

Δημητριος Σπαθαρακης
Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.
© 2018 – All rights reserved

Με επιφύλαξη παντός δικαιώματος.

Απαγορεύεται η αντιγραφή, αποθήκευση και διανομή της παρούσας εργασίας, εξ ολοκλήρου ή τμήματος αυτής, για εμπορικό σκοπό. Επιτρέπεται η ανατύπωση, αποθήκευση και διανομή για σκοπό μη κερδοσκοπικό, εκπαιδευτικής ή ερευνητικής φύσης, υπό την προϋπόθεση να αναφέρεται η πηγή προέλευσης και να διατηρείται το παρόν μήνυμα. Ερωτήματα που αφορούν τη χρήση της εργασίας για κερδοσκοπικό σκοπό πρέπει να απευθύνονται προς τον συγγραφέα. Οι απόψεις και τα συμπεράσματα που περιέχονται σε αυτό το έγγραφο εκφράζουν το συγγραφέα και δεν πρέπει να ερμηνευθεί ότι αντιπροσωπεύουν τις επίσημες θέσεις του Εθνικού Μετσόβιου Πολυτεχνείου.

Περίληψη

Με την πρόοδο στην τεχνολογία των ασύρματων επικοινωνιών, όλο και περισσότεροι άνθρωποι εξαρτώνται από φορητές κινητές συσκευές για επιχειρήσεις, ψυχαγωγία και κοινωνικές αλληλεπιδράσεις. Αν και αυτές οι φορητές κινητές συσκευές μπορούν να προσφέρουν διάφορες εφαρμογές, το βασικό τους μειονέκτημα είναι ότι οι υπολογιστικοί πόροι παραμένουν περιορισμένοι. Αυτό όμως μπορεί να ξεπεραστεί με την εξ αποστάσεως εκτέλεση εργασιών που απαιτούν μεγάλη υπολογιστική ισχύ σε ομάδες υπολογιστών γνωστούς ως υπολογιστικά νέφη. Καθώς όλο και περισσότεροι άνθρωποι έχουν πρόσβαση στο Διαδίκτυο μέσω των κινητών συσκευών, είναι λογικό να οραματιστεί κανείς ότι στο εγγύς μέλλον οι υπηρεσίες υπολογιστικού νέφους θα είναι διαθέσιμες για το κοινό, μέσω εύκολα προσβάσιμων δημόσιων ασύρματων δικτύων μητροπολιτικής περιοχής. Ωστόσο, σήμερα θεωρείται ξεπερασμένη η ιδέα της αντιμετώπισης του υπολογιστικού νέφους ως απομονωμένα και απομακρυσμένα κέντρα δεδομένων, καθώς υπάρχουν σαφή οφέλη όταν αυτά είναι τοπικά προσβάσιμα και επίσης συνδέονται μεταξύ τους για να σχηματίσουν ένα δίκτυο. Σκοπός αυτής της διπλωματικής εργασίας είναι η ανάπτυξη και εφαρμογή ενός αλγορίθμου για την εξισορρόπηση του φορτίου των εργασιών μεταξύ των τοπικά προσβάσιμων υπολογιστικών νεφών μέσα σε ένα δημόσιο ασύρματο δικτύο μητροπολιτικής περιοχής, για τη μείωση του μέσου χρόνου απόκρισης των μεταφορτωμένων εργασιών. Αρχικά εισάγουμε ένα μοντέλο συστήματος για να καταγράψουμε τους χρόνους απόκρισης των εργασιών, και διατυπώνουμε ένα νέο αλγόριθμο βελτιστοποίησης με στόχο την εύρεση ανακατευθύνσεων των ροών εργασίας μεταξύ ενός δεδομένου συνόλου υπολογιστικών νεφών στο δίκτυο. Στη συνέχεια προτείνουμε ένα γρήγορη και κλιμακωτό αλγόριθμο για το πρόβλημα, χωρίζοντάς το σε δύο μέρη, μετατρέποντάς το από την εύρεση των ζητούμενων ροών ανακατεύθυνσης των εργασιών, στην εξεύρεση της ελάχιστου κόστους μέγιστης ροής. Επιπλέον προτείνουμε έναν ακόμα καλύτερο αλγόριθμο, με μικρότερη χρονική πολυπλοκότητα για την εξεύρεση της μέγιστης ροής. Τα πειραματικά αποτελέσματα καταδεικνύουν τις σημαντικές δυνατότητες του αλγορίθμου μας για τη μείωση των χρόνων απόκρισης των εργασιών.

Λέξεις Κλειδιά

Υπολογισμός Κινητού Νέφους, Υπολογιστικό Νέφος, Τοπικό Υπολογιστικό Νέφος, Εξισορρόπηση Φορτίου, Μεταφόρτωση Υπολογισμών, Αλγόριθμος Ελάχιστου Κόστους Μέγιστης Ροής, Αλγόριθμος Μέγιστης Ροής, Μέση Απόκριση Εργασιών.

Abstract

With advances in wireless communication technology, more and more people depend heavily on portable mobile devices for businesses, entertainments and social interactions. Although such portable mobile devices can offer various promising applications, their computing resources remain limited due to their portable size. This however can be overcome by remotely executing computation-intensive tasks on clusters of near by computers known as Cloudlets. As increasing numbers of people access the Internet via mobile devices, it is reasonable to envision in the near future that Cloudlet services will be available for the public through easily accessible public wireless metropolitan area networks (WMANs). However, the outdated notion of treating Cloudlets as isolated data-centers-in-a-box must be discarded as there are clear benefits to connecting multiple Cloudlets together to form a network. The purpose of this diploma thesis is to develop and implement an algorithm for load balancing tasks between Cloudlets within a WMAN, to reduce the average response time of offloaded tasks. We first introduce a system model to capture the response times of tasks, and formulate a novel optimization problem with the objective of finding redirections of tasks between a given set of Cloudlets in a network such that the maximum of the average response times of offloaded tasks among the Cloudlets is minimized. We then propose a fast and scalable algorithm for the problem, separating it into two parts, transforming it from finding the flows of tasks in WMAN, into finding the minimum cost maximum flow. In addition we propose an even better algorithm, with lower time complexity for finding the maximum flow. We finally evaluate the performance of the proposed algorithm through experimental simulations. The experimental results demonstrate the significant potential of our algorithm in reducing response times of user tasks and maximizing user experiences.

Keywords

Mobile Cloud Computing, Cloud, Cloudlet, Load Balancing, Computation Offloading, Min Cost Max Flow Algorithm, Max Flow Algorithm, Average Response Time.

Ευχαριστίες

Αρχικά θα ήθελα να ευχαριστήσω τον καθηγητή κύριο Συμεών Παπαβασιλείου για την επίβλεψη της διπλωματικής μου εργασίας, το συνεχές ενδιαφέρον του και την ευκαιρία που μου έδωσε να ασχοληθώ με ένα επίκαιρο και ενδιαφέρον θέμα στο εργαστήριο του.

Ένα μεγάλο ευχαριστώ οφείλω στον μεταδιδακτορικό ερευνητή Δημήτρη Δεχουνιώτη και τον διδακτορικό ερευνητή Μάριο Αυγέρη, για την άριστη και καθοριστική συνεργασία που είχαμε και οδήγησε στην ολοκλήρωση της παρούσας εργασίας, την αμέριστη βοήθεια που μου παρείχαν, καθώς και τον χρόνο που διέθεσαν με πραγματικό ενδιαφέρον.

Ευχαριστώ την φίλη μου Νατάσα Τρεκλή γιατί με την ψυχαναγκαστική υπομονή και την αμέριστη βοήθεια της, η συγγραφή αυτής της διπλωματικής κατέστη εφικτή. Ένα ιδιαίτερο ευχαριστώ οφείλω στους ανθρώπους του εργαστηρίου ΜΟΠ για την παρέα τους, την βοήθεια τους και την φαντασία που επιδείκνυαν στα διαλείμματα αυτής της διπλωματικής.

Η παρούσα διπλωματική εργασία επισφραγίζει το τέλος των σπουδών μου. Οπότε θα ήθελα να ευχαριστήσω τους γονείς μου για την υπομονή και την βοήθεια τους όλα αυτά τα χρόνια και τα αδέρφια μου για την προσπάθεια τους να με κάνουν λίγο περισσότερο άνθρωπο.
Τέλος, δεν μπορώ να μην ευχαριστήσω την Ελένη για την κατανόηση και την όμορφη συντροφιά της και τους φίλους μου για την στήριξη τους στα ξεχωριστά χρόνια που πέρασαν.

Περιεχόμενα

Κατάλογος Σχημάτων

Κατάλογος Πινάκων

Κεφάλαιο 1

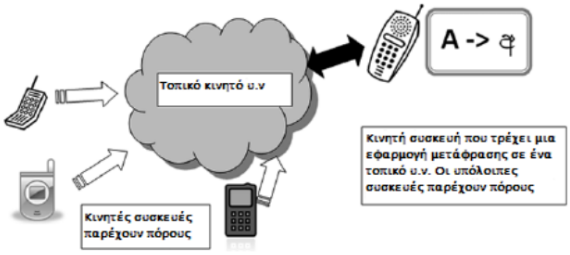
Εισαγωγή

Τα τελευταία χρόνια η ολοένα και αυξανόμενη χρήση κινητών συσκευών αλλά και μια κατεύθυνση σταδιακής μείωσης εφαρμογών, οι οποίες βασίζονται μόνο σε υλικό και δεδομένα που υπάρχουν σε έναν μόνο προσωπικό υπολογιστή, δημιουργεί μία μαζική μετατόπιση προς κατανεμημένα φορητά υπολογιστικά μοντέλα. Παρά τα εμφανή πλεονεκτήματα των φορητών συσκευών δεν έχουν ξεπεραστεί ακόμα και σήμερα τα έμφυτα προβλήματα τους. Οι περιορισμένοι πόροι, η πεπερασμένη ενέργεια (μπαταρία) και η χαμηλή συνδεσιμότητα σε δίκτυα, είναι μερικά από τα πιο δισεπίλυτα. Οι χρήστες χρειάζονται πλέον πρόσβαση όχι μόνο σε συστήματα διαδικτυακής (online) αποθήκευσης αλλά και σε διαδραστικές εφαρμογές και υπηρεσίες άμεσης απόκρισης, που με τη σειρά τους, απαιτούν έντονους υπολογιστικούς πόρους. Τα μέσα κοινωνικής δικτύωσης ή το στίγμα GPS απαιτούν εκτεταμένη χρήση αισθητήρων τοποθεσίας, που είναι ενεργοβόροι. Έτσι, αφενός εντείνεται το πρόβλημα της περιορισμένης μπαταρίας και αφετέρου γίνεται αδύνατη η παροχή στον χρήστη καλύτερων υπηρεσιών μέσω των ενσωματωμένων αισθητήρων. Τέλος οι σύγχρονες εφαρμογές (παιχνίδια, σύνθεση ομιλίας, φυσική επεξεργασία γλωσσών, επαυξημένη πραγματικότητα, κ.λ.π) απαιτούν υψηλές υπολογιστικές ικανότητες περιορίζοντας έτσι τους προγραμματιστές των εφαρμογών αυτών στις δυνατότητες των φορητών συσκευών. Λαμβάνοντας υπόψη τις τάσεις στην αρχιτεκτονική των κινητών τηλεφώνων και των μπαταριών είναι απίθανο τα προβλήματα αυτά να λυθούν στο μέλλον. Στην πραγματικότητα, δεν πρόκειται απλώς για μια προσωρινή τεχνολογική ανεπάρκεια, αλλά εγγενής στην κινητικότητα (mobility) και ένα εμπόδιο που πρέπει να ξεπεραστεί άμεσα προκειμένου να αξιοποιηθεί πλήρως οι δυνατότητες των φορητών συσκευών.

Ο Υπολογισμός Κινητού Νέφους (ΥΚΝ) (Mobile Cloud Computing) είναι από τις πιο επιδραστικές και ταχεώς εξελισσόμενες τεχνολογίες που στοχέυουν να αλλάξουν το μέλλον της πληροφορικής και των εφαρμογών στο πεδίο που αφορά τις φορητές συσκευές (κινητά, ταμπλετές, κ.λ.π). Η βασική ιδέα είναι η χρήση πόρων άλλων υποδομών εκτός από την ίδια την κινητή συσκευή για την εκτέλεση των εφαρμογών. Μια τέτοια υποδομή, όπου η αποθήκευση και επεξεργασία δεδομένων θα μπορούσε να συμβεί εκτός της κινητής συσκευής, θα μπορούσε να χαρακτηριστεί ως «κινητό νέφος». Με την αξιοποίηση των δυνατοτήτων της υπολογιστικής ισχύς και αποθήκευσης των υποδομών Υπολογιστικού Νέφους (ΥΝ) (cloud computing)

για κινητά, οι εφαρμογές μπορούν να εκτελούνται σε κινητές συσκευές παρότι αυτές έχουν πρόσβαση σε λίγους πόρους. Οι εφαρμογές νέφους μπορούν να εκτελούν και να διαμοιράζονται τους υπολογιστικούς πόρους και τα δεδομένα τους με παράλληλη πρόσβαση σε τεράστιους όγκους δεδομένων. Έτσι, ο Υπολογισμός Νέφους (ΥΝ) επιτρέπει μία νεα γενιά υπηρεσιών, με προσανατολισμό στους υψηλά κατανεμημένους διαδικτυακους υπολογισμούς (on-line computing) και στην πραγματοποίηση ενός νέου μοντέλου υπολογισμών κατ΄ απαίτηση των χρηστών (on-demand computing) υψηλής απόδοσης και εύκολα προσβάσιμο. Συνήθως ο όρος ΥΚΝ σημαίνει να εκτελείται μια εφαρμογή σε μια απομακρυσμένη πηγή υπολογιστικών πόρων (σύννεφο), οπως οι υποδομες ΥΝ. Μία άλλη προσέγγιση είναι να συμπεριλάμβάνονται και οι κινητές συσκευές στο μοντέλο του σύννεφου, με αποτέλεσμα να εκμεταλλεύονται όλοι οι συλλογικοί πόροι από μία τοπική περιοχή για την εκτέλεση μίας εργασίας. Η τρίτη προσέγγιση είναι η μεταφόρτωση της εργασίας σε ένα τοπικό υπολογιστικό νέφος -ΤΥΝ (cloudlet) που αποτελείται από πολλούς υπολογιστές πολλαπλών πυρήνων (cloud) και μικρό αριθμό εξυπηρετητών. Το Σχήμα 1.1 δείχνει τις διαφορετικές αυτές οπτικές.

(α΄) Ένα απομακρυσμένο υπολογιστικό νέφος δίνει πόρους σε συσκευές μέσω διαδικτύου



(β΄) Ένα εικονικό υπολογιστικό νέφος που αποτελείται από συσκευές που βρίσκονται στην περιοχή

(γ΄) Ένα τοπικό υπολογιστικό νέφος που τρέχει μία εφαρμογή που μεταφορτώθηκε από μία κινητή συσκευή

Σχήμα 1.1: Οι τρεις ορισμοί του ΥΚΝ [1]

1.1 Υπολογισμός Κινητού Νέφους

Τα τελευταία χρόνια υπάρχουν πάρα πολλές εφαρμογές που βασίζονται στη λογική του ΥΚΝ. Οι μελλοντικές εφαρμογές αναμένονται ακόμα πιο πρωτοπόρες και χρήσιμες, και έχουν στόχο να αλλάξουν ριζικά την καθημερινή ζωή. Μερικές κατηγορίες εφαρμογών και κάποια παραδείγματα είναι τα παρακάτω. Το Ηλεκτρονικό Εμπόριο, όπως ηλεκτρονική τραπεζική (e-banking), η ηλεκτρονική διαφήμιση και το ηλεκτρονικό εμπόριο (e-commerce), χρησιμοποιούν κλιμακούμενη ισχύς επεξεργασίας και μέτρα ασφαλείας για να φιλοξενήσουν μεγάλο όγκο της κυκλοφορίας λόγω της ταυτόχρονης πρόσβασης των χρηστών και της επεξεργασίας συναλλαγών. Η Κοινή Χρήση Πολυμέσων παρέχει ασφαλή προβολή και κοινή χρήση πολυμέσων στις πληροφορίες που αποθηκεύονται σε κινητά τηλέφωνα. Η Κινητή Μάθηση επιτρέπει σε ένα τερματικό να αποκτήσει πρόσβαση σε άφθονα μαθησιακά υλικά αποθηκευμένα στο ΥΝ οποιαδήποτε στιγμή και οπουδήποτε. Ο πληθοπορισμός (Crowdsourcing) είναι μία από τις αναδυόμενες εφαρμογές ΥΚΝ [2]. Χρησιμοποιεί λειτουργίες των αισθητήρων των κινητών συσκευών και την υψηλή ικανότητα επεξεργασίας του υπολογιστικού νέφους. Μία μελλοντική εργαρμογή που προτάθηκε στο [3], είναι η προσπάθεια ανάκτησης δεδομένων μέσα από φωτογραφίες. Για παράδειγμα μετά από μια καταστροφή, χρησιμοποιώντας κάμερες κινητών συσκευών, μπορεί να φτιαχτεί ένας λεπτομερής χάρτης του χώρου που πρέπει να ανακατασκευαστεί. Μια άλλη αναδυόμενη και μελλοντική εφαρμογή είναι η συλλογική ανίχνευση με τη χρήση αισθητήρων [4],[5]. Οι αναδυόμενες εφαρμογές συλλογικής ανίχνευσης μπορούν να συνθέσουν έναν χάρτη της κυκλοφορίας στους δρόμους σε πραγματικό χρόνο από την συλλογή δεδομένων συλλογικής κίνησης [6], την παρακολούθηση της ρύπανσης του περιβάλλοντος [7], ακόμα και την υγειονομική περίθαλψη [8]. Για παράδειγμα, στην υγειονομική περίθαλψη με την συλλογή δεδομένων δίνεται η δυνατότητα σε ένα γιατρό να κοιτάξει τα αρχεία ασθενών στην κινητή συσκευή του για απομακρυσμένη διάγνωση ή για την παρακολούθηση της κατάστασης του ασθενούς. Η υπηρεσία κινητής τηλεφωνίας βάσει τοποθεσίας είναι επίσης μια αναδυόμενη εφαρμογή ΥΚΝ. Εκτός από τη δυνατότητα συνεκτίμησης δεδομένων και πληροφοριών βάση θέσης [9], οι υπηρεσίες κινητής τηλεφωνίας που βασίζονται σε τοποθεσίες λαμβάνουν επίσης υπόψη το περιβάλλον του χρήστη, άλλες συσκευές και τον χρόνο μεταξύ των αλλαγών στο περιβάλλον. Τέλος, η επαυξημένη πραγματικότητα (augmented reality) και τα ηλεκτρονικά παιχνίδια εμφανίζονται ως πολύ διάσημες εφαρμογή ΥΚΝ. Ενώ η παραδοσιακά επαυξημένη πραγματικότητα είναι δυνατή μόνο με ειδικό εξοπλισμό και απαιτεί τεράστια επεξεργαστική ισχύ, πλέον είναι δυνατή με το ΥΚΝ με κλιμακούμενους υπολογισμούς και μεγάλη δυνατότητα αποθήκευσης δεδομένων [10].

Όπως φαίνονται και στο σχήμα 1.2 ερευνητές και επιστημονικές μελέτες έχουν εστιάσει στις παρακάτω προσέγγισεις, τα θέματα και τις προκλήσεις όσον αφορά το ΥΚΝ:

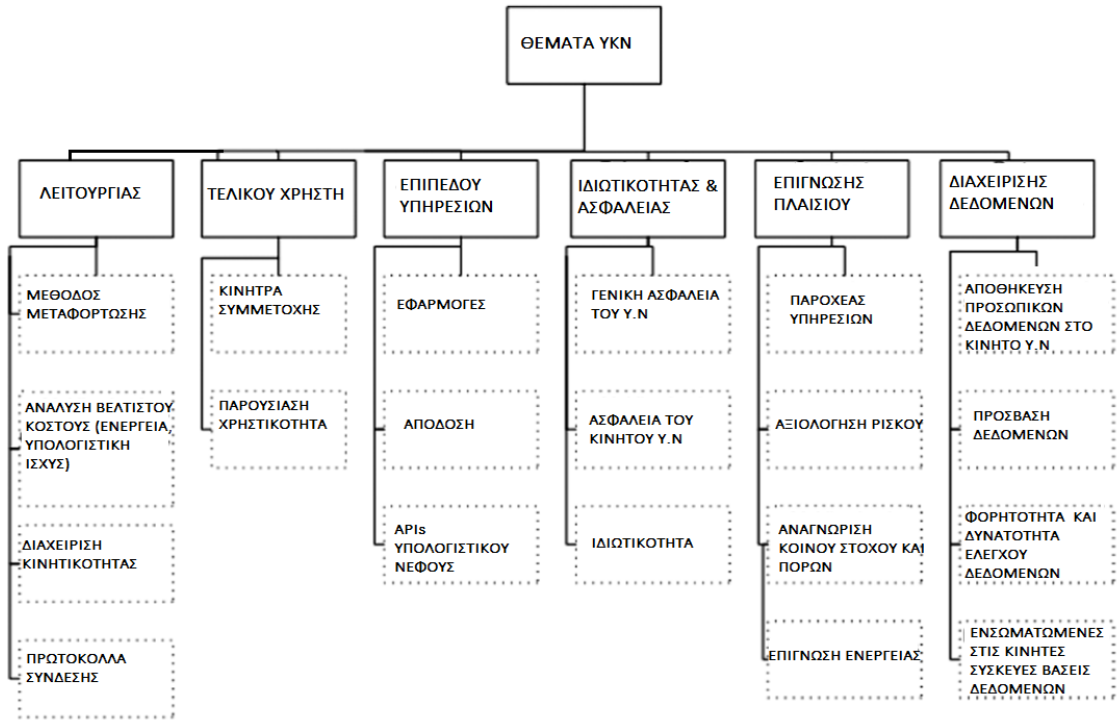
Τα θέματα λειτουργίας (επιχειρησιακά) αναφέρονται σε βασικά τεχνολογικά θέματα όπως η μέθοδος μεταφόρτωσης υπολογισμών (computation offloading) [11],[12],[13],[14], η λήψη αποφάσεων για μεταφόρτωση ή οχι από μοντέλα κόστους-όφελους [15],[16], η υποστήριξη-

διαχείριση της κινητηκότητας των κινητών συσκευών [17] και η χρήση πρωτοκόλλων σύνδεσης στο διαδίκτυο [18].

Τα θέματα τελικού χρήστη σχετίζονται με θέματα που αφορούν την διαλειτουργικότητα [12], το κόστος και τα κίνητρα συμμετοχής των χρηστών [19],[20]. Για παράδειγμα εάν πολλοί χρήστες πρέπει να εκτελέσουν την ίδια εργασία, μπορεί αυτή να χωριστεί έτσι ώστε τελικά ο κάθε χρήστης πρέπει μόνο να κάνει ένα μικρό μέρος της, συνεισφέροντας όλοι τελικά στο σύννεφο πόρων.

Τα θέματα επιπέδου υπηρεσιών και εφαρμογών σχετίζονται με τους παράγοντες σχετικά με τις μετρήσεις απόδοσης του συστήματος και την ποιότητα των υπηρεσιών (Quality of Service) [21] του συστήματος. Για παράδειγμα, με ποιο τρόπο τα συστήματα ΥΚΝ εξασφαλίζουν τη διαθεσιμότητα των πόρων τους· Ποια είναι τα μοντέλα ανοχής σφαλμάτων που χρησιμοποιούνται για την εξασφάλιση ομαλής εκτέλεσης και αδιάλειπτης εξυπηρέτησης·

Τα θέματα ιδιωτικότητας, ασφάλειας και εμπιστοσύνης. Είτε στην διαδικασία μεταφόρτωσης υπολογισμών, είτε στην αποθήκευση δεδομένων, η χρήση του σύννεφου για κινητές συσκευές θέτει ζητήματα ασφάλειας και θέματα εμπιστοσύνης [22]. Λόγω της χαμηλής χωρητικότητας αποθήκευσης των φορητών συσκευών, πολλοί χρήστες αρχίζουν να αποθηκεύουν δεδομένα όπως επαφές, ημερολόγια σε υποδομες ΥΝ. Ωστόσο, αυτές οι υπηρεσίες θεωρούνται ευάλωτες και οι χρήστες ενδέχεται να χάσουν τα δεδομένα τους, εάν οι υπηρεσίες απλώς σταματήσουν κάποια στιγμή να λειτουργούν ή εάν η ανταλλαγή δεδομένων αποτύχει λόγω τεχνικών προβλημάτων.



Σχήμα 1.2: Χαρακτηριστικά προβλήματα του ΥΚΝ [1]

Τα θέματα επίγνωσης πλαισίου. Ο Schilit [94] περιγράφει τις τρεις σημαντικές πτυχές ως εξής: η τοποθεσία του χρήστη, η ύπαρξη άλλων χρήστων στην περιοχή, και τους πόρους στο περιβάλλον του χρήστη [23]. Η επίγνωση του πλαισίου δίνει την δυνατότητα σε συστήματα να επαναρυθμίστουν αυτόματα για να προσαρμοστούν στο πλαίσιο που ανήκουν αυτή τη στιγμή. Στην περίπτωση του κινητών συστημάτων, είναι πολύ χρήσιμη καθώς αυτά καλούνται να εκτελούν εργασίες σε ένα περιβάλλον που υπόκειται σε συνεχή αλλαγή.

Τα θέματα διαχείρισης δεδομένων που αφορούν τα ζητήματα εμπιστοσύνης και ιδιωτικότητας για την αποθήκευση προσωπικών δεδομένων, της συνδεσιμότητας στα δεδομένα ανά πάσα στιγμή και της προσωρινής αποθήκευσης δεδομένων σε κατάλληλες βάσεις δεδομένων για κινητές συσκευές για εύκολη πρόσβαση [24].
Οι δύο βασικές Τεχνικές που συναντήσαμε στην Βιβλιογραφία για τα μοντέλα ΥΚΝ είναι η Ανάθεση Πόρων και η Εξισορρόπηση Φορτίου.

1.1.1 Ανάθεση Πόρων

Η μία τεχνική λοιπόν έχει ως βασική λογική, ότι ένας κόμβος που του έχει ανατεθεί βαρύ φορτίο (πολλές εργασίες), και έχει συγκεκριμένους και ανεπαρκείς πόρους, μπορεί να δανειστεί πόρους από άλλους κόμβους που δεν τους χρειάζονται δηλαδή έχουν ελαφρύ φορτίο.
Έτσι θα επιτευχθούν:

- Οι πόροι να διατίθενται εύκολα αν ζητηθούν από τους κόμβους (On demand).
- Οι πόροι να χρησιμοποιούνται αποτελεσματικά με βασικό κριτήριο τις διακυμάνσεις του φορτιου που πρεπει να εξυπηρετησουν.
- Να μην υπάρχει σπατάλη ενέργειας σε περίπτωση χαμηλού φορτίου.
- Το κόστος χρήσης των πόρων μειώνεται.

1.1.2 Δρομολόγηση των εργασιών

Η δρομολόγηση των εργασιών γίνεται αφού έχουν ανατεθεί όλοι οι πόροι στους κόμβους του δικτύου. Με τον όρο δρομολόγηση εννοείται, ότι σε έναν κόμβο που του έχει ανατεθεί βαρύ φορτίο μπορεί να στείλει κάποιες εργασίες σε διπλανούς κατάλληλους κόμβους που έχουν ελαφρύ φορτίο. Αυτό συνεπάγεται ότι κανένας κόμβος δεν είναι υπερφορτωμένος (overloaded) ή δεν μένει αναξιοποίητος (underloaded). Αυτό έχει ως αποτέλεσμα να μειώνεται η συνολική απόκριση των εργασιών και το συνολικό φορτίο να είναι ισορροπημένο στους κόμβους του δικτύου.

1.1.3 Τυπικά Προβλήματα

Λόγω των χαρακτηριστικών των κατανεμημένων συστημάτων, εμφανίζονται μερικά τυπικά προβλήματα κατά τη διάρκεια της κατανομής εργασιών και της εξισορρόπησης φορτίου, όπως φαίνεται παρακάτω.

- *Μοντέλα ελέγχου.* Για να επιτευχθεί το βέλτιστο αποτέλεσμα για την κατανομή εργασιών και την εξισορρόπηση φορτίου, απαιτείται μια κεντρική μονάδα ελέγχου για τη συλλογή των πληροφοριών κατάστασης του συνόλου του συστήματος σε πραγματικό χρόνο. Ωστόσο, αυτό είναι δύσκολο να εφαρμοστεί επειδή τα κατανεμημένα συστήματα είναι πάντα μεγάλα, δυναμικά και χωρίς μονάδες ελέγχου. Αντίθετα, η πλήρως αποκεντρωμένη προσέγγιση μπορεί να απαιτεί σχετικά υψηλό υπολογιστικό κόστος από τους κόμβους, οι οποίοι ενδέχεται να βάζουν υπολογιστικά βαριά φορτία σε μεγάλα συστήματα με αποτέλεσμα να καθιστούν δύσκολο τον έλεγχο της διαδικασίας κατανομής των εργασιών [25].

- *Βελτιστοποίηση πόρων.* Πολλά υπάρχοντα μοντέλα κατανομής εργασιών και εξισορρόπησης φορτίου, έχουν υλοποιηθεί με βάση τη βελτιστοποίηση της προσβασιμότητας στους απαιτούμενους πόρους [26]. Σε αυτά τα μοντέλα η μέτρηση της προσβασιμότητας σε πόρους και ο τρόπος βελτιστοποίησης ενός δείκτη απόδοσης, είναι βασικά προβλήματα.

- *Αξιοπιστία.* Σε ανοικτά κατανεμημένα συστήματα, μερικοί κόμβοι μπορεί να είναι αναξιόπιστοι. Έτσι, ένα βασικό πρόβλημα είναι η εξασφάλιση πρόσβασης για τους κόμβους, σε αξιόπιστους πόρους καθώς και η δυνατότητα μετάβασης σε άλλους πόρους, για την περίπτωση σφάλματος σε έναν κόμβο [25]. Επομένως, θα πρέπει να σχεδιάσουμε κάποια προσέγγιση για την εύρεση ενός κυρίαρχα αξιόπιστου τρόπου καταμερισμού των εργασιών και τον πόρων.

- *Δικτυακές Δομές.* Η επικοινωνία και η αλληλεπίδραση κόμβων περιορίζεται από τις δικτυακές ιδιαιτερότητες. Η μέτρηση της επίδρασης των δομών (π.χ η σύνδεση δύο κόμβων) του δικτύου σχετικά με την εκτέλεση των καθηκόντων είναι ένα πρόβλημα. Για κατανομή εργασιών και η εξισορρόπηση φορτίου θα πρέπει να εξεταστούν και να συνυπολογίστούν τις τοποθεσίες των κόμβων στο δίκτυο.

1.2 Αντικείμενο της διπλωματικής

1.2.1 Το πρόβλημα

Σε περιβάλλοντα ΥΚΝ, οι ασύρματες κινητές συσκευές έχουν πρόσβαση στο υπολογιστικό νέφος μέσω ασύρματης επικοινωνίας, όπως Wi-Fi, 3G / 4G, κ.λπ. Ωστόσο, είναι γνωστό ότι η ασύρματη επικοινωνία είναι αναξιόπιστη και περιορισμένη από το εύρος ζώνης, καθιστόντας την μεγάλη καθυστέρηση της μεταφοράς δεδομένων αναπόφευκτη. Έτσι, η εκφόρτωση εργασιών από κινητές συσκευές στις υποδομές ΥΝ δεν είναι πάντα μια έξυπνη επιλογή αφού το υπολογιστικό νέφος είναι συνήθως μακριά (απομονωμένο) από τους χρήστες. Για να ξεπεραστεί αυτό, οι υποδομές αυτές πρέπει να μετακινηθεί πιο κοντά στους χρήστες των κινητών με τη μορφή του ΤΥΝ. Το ΤΥΝ είναι ένα αξιόπιστο, πλούσιο σε πόρους σύμπλεγμα υπολογιστών με δυνατότητα ασύρματης σύνδεσης με τους κοντινούς χρήστες κινητών τηλεφώνων.

Ως εκ τούτου, όταν οι κινητές συσκευές δεν μπορούν ή δεν θέλουν να συνδεθούν με το απομακρυσμένο σύννεφο, μπορούν να βρουν και να έχουν πρόσβαση σε έναν κοντινό ΤΥΝ. Πρόσφατες μελέτες [26] πρότειναν την ανάπτυξη ενός δικτύου με ΤΥΝ σε ασύρματα μητροπολιτικα δίκτυα (wireless metropolitan area networks). Όμως, το ΤΥΝ δεν έχει άφθονους πόρους, όπως το σύννεφο. Ιδιαίτερα, όταν εκτελούνται ταυτόχρονα πολλές αιτήσεις ή χρησιμοποιούνται απρόσκοπτα υπολογιστικοί πόροι, τότε μπορεί να εξαντληθούν οι πόροι του γρήγορα, αυξάνοντας δραματικά τους χρόνους απόκρισης των εργασιών που του έχουν ανατεθεί. Μια μεγάλη πρόκληση λοιπόν, που αντιμετωπίζουν οι πάροχοι υπηρεσιών WMAN είναι ο καταμερισμός του φόρτου εργασίας του χρήστη σε διαφορετικά γεωγραφικά κατανεμημένα ΤΥΝ, έτσι ώστε το συνολικό φορτίο στο δίκτυο να είναι καλά ισορροπημένο, μειώνοντας έτσι τους χρόνους απόκρισης των μεταφορτωμένων εργασιών. Αυτή ακριβώς η διαδικασία ονομάζεται εξισορρόπηση φορτίου (Load Balancing).
Στα πλαίσια της διπλωματικής εργασίας, υλοποιήθηκε η αλγοριθμική λύση που προτείνεται στην επιστημονική δημοσίευση [25]. Υλοποιήθηκε ένας διαφορετικός αλγόριθμος υπολογισμού της μέγιστης ροής που έχει μικρότερη πολυπλοκότητα. Ακολούθησε πειραματική αξιολόγηση των παραμέτρων του αλγορίθμου και της απόδοσης του σε διάφορες καταστάσεις.

1.2.2 Συνεισφορά

Τα πειραματικά αποτελέσματα δείχνουν ότι ο αλγόριθμος είναι μία γρήγορη και κλιμακωτή λύση στο πρόβλημα εξισορρόπησης φορτίου. Όμως σε συνθήκες όπου οι ροές αφίξεων εργασιών είναι παραπάνω από τις αναμενόμενες, η απόκριση των εργασιών είναι ιδιαίτερα υψηλή, με αποτέλεσμα ο αλγόριθμος να μην είναι τοσο ικανοποιητικός υπο αυτές τις συνθήκες. Επίσης προτάθηκε η τροποποίηση του αλγορίθμου Μέγιστης Ροής Ελάχιστου Κόστους με δύο αλγορίθμους για τον υπολογισμό της Μέγιστης Ροής, και παραθέτουμε πειραματικά αποτελέσματα που δείχνουν ότι για τις συγκεκριμένες καθυστερήσεις δικτύου που προτείνεται δεν υπάρχει μεγάλη διαφορά. Οι πολυπλόκοτητα του αλγορίθμου εύρεσης μέγιστης ροής είναι μικρότερη και άρα θα ήταν μια καλή λύση σε πιθανή εφαρμογή του αλγορίθμου που εκπονήθηκε στα πλαίσια της διπλωματικής εργασίας.

1.3 Οργάνωση του τόμου

Η παρούσα διπλωματική εργασία είναι οργανωμένη σε έξι κεφάλαια. Εργασίες σχετικές με το αντικείμενο της διπλωματικής παρουσιάζονται στο Κεφάλαιο 2. Στο Κεφάλαιο 3 παρουσιάζεται το θεωρητικό υπόβαθρο, καθώς και οι βασικές έννοιες και τεχνικές που χρησίμευσαν στη διπλωματική εργασία. Στο Κεφάλαιο 4 παρουσιάζεται εκτενώς το πρόβλημα, και οι αλγόριθμοι που χρησιμοποιήθηκαν στην διπλωματική για την επίλυση του. Στο Κεφάλαιο 5 παρουσιάζουμε τα πειραματικά αποτελέσματα των μετρήσεων και ακολουθεί ανάλυση αυτών και παρουσίαση των συμπεράσματων της διπλωματικής. Το Κεφάλαιο 6 παρέχει ιδέες και κατευθύνσεις για την προοπτική μελλοντικής εξέλιξής της.

Κεφάλαιο 2

Βιβλιογραφία

2.1 Στόχοι και Μετρικές Εξισορρόπησης Φορτίου

Η Εξισορρόπηση Φορτίου - ΕΦ εφαρμόζεται για τη διανομή του δυναμικού τοπικού φόρτου εργασίας σε όλους τους πόρους ή τις εικονικές μηχανές εξίσου. Υποστηρίζει την αξιοποίηση των πόρων και την υψηλή ικανοποίηση των χρηστών με τη διασφάλιση της κατάλληλης και δίκαιης κατανομής του κάθε πόρου. Μια κατάλληλη εξισορρόπηση φορτίου βοηθά στην ελαχιστοποίηση της κατανάλωσης πόρων, τη μεγιστοποίηση της κλιμακοσημότητας (scalability), και την αποφυγή σημείων συμφόρησης στους κόμβους. Οι σημαντικές ποιοτικές μετρικές και άρα οι στόχοι για την ΕΦ στα ΥΝ είναι οι παρακάτω:

Χρόνος απόκρισης: Μετράει το συνολικό χρόνο που χρειάζεται το σύστημα για να εξυπηρετήσει ένα αίτημα που του έχει υποβληθεί [34].

Κλιμακοσημότητα: Καθορίζει τον τρόπο με τον οποίο το σύστημα είναι σε θέση να επιτελέσει αλγόριθμους ΕΦ με περιορισμένο αριθμό ΕΜ ή εξυπηρετητών [35].

Χρήση πόρων: Είναι ο βαθμός στον οποίο οι πόροι του συστήματος χρησιμοποιειούνται. Ένας επιθυμητός αλγόριθμος ΕΦ παρέχει μέγιστη αξιοποίηση πόρων [36].

Διεκπεραιωτική Ικανότητα (throughput): Ο ρυθμός με τον οποίο ένας κόμβος στο σύστημα στέλνει ή λαμβάνει δεδομένα. Με απλά λόγια, ορίζεται ως ο αριθμός των κόμβων που αλλάζουν την κατάστασή τους όταν ολοκληρώσουν τις εργασίες που τους έχουν ανατεθεί σε κάποια μονάδα χρόνου. Για καλύτερη απόδοση του ΥΝ, απαιτείται υψηλή διεκπεραιωτική ικανότητα [37].

Χρόνος μεταφοράς: Ο χρόνος που απαιτείται για τη μετάβαση μιας εργασίας από ένα μηχάνημα σε οποιοδήποτε άλλο μηχάνημα στο ΥΝ. Θα έπρεπε να είναι ελάχιστο για τη βελτίωση της απόδοσης του συστήματος [38].

Makespan: Είναι η ελαχιστοποίηση του χρονικού διαστήματος μεταξύ της έναρξης του πρώτου έργου μέχρι και το τέλος όλων των εργασιών που πρέπει να ολοκληρωθούν [39].

Ανεκτό σφάλμα: Είναι η δυνατότητα του αλγόριθμου να εκτελέστει ομοιόμορφα και σωστά, ακόμη και σε συνθήκες αποτυχίας [40].

Ο βαθμός ανοχής σε σφάλματα: Μετράει την ανισορροπία μεταξύ εικονικών μηχανών στο ΥΝ [41].

Απόδοση: Αντιπροσωπεύει την αποτελεσματικότητα του συστήματος μετά την εκτέλεση ΕΦ [42]. Εάν πληρούνται όλες οι παραπάνω παράμετροι τότε η απόδοση του αλγορίθμου θα είναι υψηλή και θα επηρεάζει πολύ την συνολική απόδοση του ΥΝ.

2.2 Τεχνικές Εξισορρόπησης Φορτίου

Στην πραγματικότητα, είναι δύσκολο να ικανοποιηθούν όλοι οι παραπάνω στόχοι ταυτόχρονα. Η εξυσορρόπηση φορτίου είναι ένας μηχανισμός που προσπαθεί να πετύχει το καλύτερο δυνατό αποτέλεσμα δεδομένου ενός καταμερισμού εργασιών. Σε αυτό το σημείο είναι σημαντικό να προσδιορίσουμε τις υπάρχουσες τεχνικές και μεθοδολογίες σε αυτό τον τομέα. Οι παρακάτω τρόποι δείχνουν τις διαφορετικές λογικές για το πως μπορεί να εφαρμοστεί η ΕΦ σε ένα δίκτυο υπλογιστικών κόμβων.

2.2.1 Τεχνικές Στατικού Ελέγχου Εξισορρόπησης Φορτίου

Γενικά, οι τεχνικές στατικού ελέγχου χρησιμοποιούνται σε μικρά συστήματα, και χρησιμοποιούν ένα κεντρικό μοντέλο για να καθορίσει το πως κατανέμονται οι εργασίες στους κόμβους. Συνοπτικά, αν και το μοντέλο κεντρικού ελέγχου είναι μία απλή λύση που μπορεί να επιτύχει βέλτιστο αποτέλεσμα στο σύνολο του δικτύου, είναι συχνά ανεφάρμοστη στην πραγματικότητα επειδή μόνο ένας μικρός αριθμός των κατανεμημένων συστημάτων μπορούν να θεωρηθούν ως στατικά. Με τον όρο στατικά εννοοείται ότι δεν μεταβάλλονται οι συνθήκες με τον χρόνο, είτε όσον αφορά το φόρτο εργασίας, είτε αφορά την κατάσταση του κάθε κόμβου. Ως εκ τούτου, υπάρχουν μόνο λίγες μελέτες που υιοθετούν πλήρως κεντρικά μοντέλα ελέγχου.

2.2.2 Τεχνικές Δυναμικής Εξισορρόπησης Φορτίου

Σε αυτή την ενότητα, θα περιγράψουμε τις τεχνικές δυναμικής ΕΦ και τα βασικά χαρακτηριστικά των δημοφιλέστερων αλγορίθμων.
Οι δυναμικοί αλγόριθμοι ΕΦ έχουν περισσότερη πρακτική αξία όπως είπαμε από αυτούς που είναι στατικοί. Οι περισσότεροι από τους δυναμικούς αλγορίθμους βασίζονται στο συνδυασμό

της γνώσης από προηγούμενες πληροφορίες για τους κόμβους στο σύστημα και των δεδομένων που προκύπτουν κατά την διάρκεια εκτέλεσης του αλγορίθμου σε κάθε κόμβο. Αυτοί οι αλγόριθμοι αναθέτουν εργασίες σε ένα κόμβο και δυναμικά τις ανακατανέμουν σε άλλους κόμβους για να βελτιώσουν την συνολική απόδοση, βασιζόμενοι στα χαρακτηριστικά που συλλέγουν από τις πληροφορίες των κόμβων.

Μία από τις τεχνικές για να επιτευχθεί η ΕΦ σε περιβάλλοντα ΥΝ είναι η μεταφορά των εικονικών μηχανημάτων - ΕΜ από φυσικό μηχάνημα σε άλλο φυσικό μηχάνημα [37]. Στην ίδια δημοσίευση προτείνεται η χρήση μίας μεθόδου βασιζόμενη στη μεταφορά κάποιων εργασιών από υπερφορτωμένα ΕΜ αντί για ολόκληρο το ΕΜ. Η προσομοίωση είχε ως αποτέλεσμα τη μείωση της διάρκεια της ΕΦ και τη μείωση της κατανάλωσης ενέργειας σε σχέση με άλλες παραδοσιακές μεθόδους. Επιπλέον, η μέθοδος κατάφερε να μειώσει τη χρήση δέσμευσης της μνήμης κατά τη μεταφορά των ΕΜ και την συνολική κατανάλωση μνήμης αφου δεν χρειαζόταν το ΕΜ να τεθεί σε παύση κατά τη διάρκεια της μεταφοράς. Ωστόσο, η μεταφορά των εργασιών ήταν δυνατή μόνο σε ΕΜ ίδιου τύπου.

Στην εργασία [52] προτάθηκε ένας μηχανισμός ΕΦ για ένα σύνολο εφαρμογών που φιλοξενούναται σε μια ομάδα εξυπηρετητών ΥΝ. Ο αλγόριθμος ΕΦ επιλύει ενα πρόβλημα μεικτού ακέραιου γραμμικού προγραματισμού (Mixed Integer Linear Programming), λαμβάνοντας υπόψη μια εκτίμηση για την μελλοντική τιμή του φορτίου και ενα σύνολο διαφορετικών σημείων λειτουργίας καθε εφαρμογης, οπως υπολογίζεται στην [53]. Στοχος του αλγοριθμου ΕΦ είναι η επίτευξη συγκεκριμένου χρόνου απόκρισης για καθε εφαμογή. Η κλιμακωσιμότητα του αλγοριθμου σε σχεση μετον αριθμό των εφαρμογών και των εξυπηρετητών δεν ειναι υψηλη.

Μία άλλη λύση για ΕΦ προτάθηκε στο [34]. Ο αλγόριθμος που είναι γνωστός ως Honey Bee Behavior, θεωρεί ότι οι εργασίες που πρέπει να εξισορροπηθούν συμπερίφερονται σαν μέλισσες και τα ΕΜ σαν πηγές φαγητού, ενώ τα υποφορτωμένα ΕΜ είναι σαν τους κατάλληλους προορισμούς για τις εργασίες. Η ανάθεση μιας εργασίας σε ένα ΕΜ είναι σαν την αναζήτηση τροφής μιας μέλισσας. Όταν ένα ΕΜ είναι υπερφορτωμένο η εργασία θα ανατεθεί σε ένα που είναι υποφορτωμένο. Οι εργασίες που ανατίθενται από ένα ΕΜ σε άλλο ανανεώνουν την πληροφορία για το συγκεκριμένο ΕΜ. Έτσι, ανάλογα με το φόρτο εργασίας και τη διαθεσημότητα των ΕΜ, αυτή η ανανέωση της πληροφορίας βοηθάει στην απόφαση για το ποια εργασία θα ανατεθεί σε ποιο ΕΜ. Η μέθοδος βελτίωσε την διεκπαιρεωτική ικανότητα και επέφερε τη μείωση του χρόνου που μία εργασία πρέπει να περιμένει στην ουρά ενός ΕΜ. Επιπλέον μειώθηκε ο χρόνος απόκρισης και το makespan. Ωστόσο, θεωρεί τις εργασίες ανεξάρτητες και δεν τελικά ο αλγόριθμος δεν μπορεί να κατηγοριοποιηθεί ως υψηλά κλιμακώσιμη μέθοδος.

Ως μια άλλη λύση [42], με την παραλλαγή του προηγούμενου αλγορίθμου, θεώρησαν αυτή τη φορά ότι μόνο τα υποφορτωμένα ΕΜ είναι η πηγή φαγητού για τις μέλισσες. Η μέθοδος αυτή έχει τέσσερα διαφορετικά στάδια, τον υπολογισμό του φόρτου των ΕΜ, την απόφαση και προγραμματισμό της ΕΦ, την ομαδοποίηση των ΕΜ, και τελικά την ανάθεση των εργασιών

σε ΕΜ. Τα αποτελέσματα δείχνουν τη βελτίωση της ποιότητας υπηρεσιών για τους χρήστες και την ελαχιστοποίηση του makespan. Παρόλαυτα ακόμα δεν ξεπεράστηκε το πρόβλημα της χαμηλής κλιμακωσιμότητας.

Μία άλλη λύση για την μεταφορά των ΕΜ για την ΕΦ [43] πρότεινε, μία συνεργασία μεταξύ των φυσικών μηχανημάτων για την διαδικασία απόφασης της μεταφοράς των ΕΜ. Δηλαδή κάθε φυσικό μηχάνημα, ανάλογα με τις ρυθμίσεις του και την κατάσταση του, συμμετείχε στην δημοπρασία για να φιλοξενήσει ένα ΕΜ. Χρησιμοποιήθηκαν διάφορες ευριστικές, για το πότε ένα ΕΜ πρέπει να μεταφερθεί και πως ένα φυσικό μηχάνημα θεωρείται κατάλληλο για να αναλάβει ένα ΕΜ. Τα αποτελέσματα δείχνουν ότι η ΕΦ είναι πολύ αποδοτική, αλλά κρίνεται ώς αρκετά κεντρικοποιημένο μοντέλο με αποτέλεσμα να μην εφάρμοζεται εύκολα, και η μεταφορά ΕΜ είναι πολύ συχνή.

Επίσης η ποιότητα των υπηρεσιών είναι ένα σημαντικό πεδίο έρευνας που καλύπτει πολλά κρίσιμα θέματα όπως η αποτελεσματική εφαρμογή της ΕΦ. Στο [44] παρουσιάζεται η κατανομή βοηθητικών ΤΥΝ σε κατάλληλα ΕΜ των οποίων ο φόρτος εργασίας είναι μέγιστος μεταξύ των υπόλοιπων ΕΜ. Επομένως αυτό βοηθάει στην ισορροπημένη κατανομή των ΤΥΝ στα ΕΜ και κάνει το σύστημα πιο ενεργό και ισορροπημένο. Έτσι η ποιότητα των υπηρεσιών, η χρήση των πόρων και η διάρκεια για την ολοκλήρωση όλων των εργασιών είναι βελτιωμένα σε σύγκριση με άλλες μεθόδους που χρησιμοποιούν ΤΥΝ. Όμως, το αρνητικό είναι το υψηλό makespan και η χαμηλή κλιμακωσιμότητα.

Στο [45] προτείνεται ένας γρήγορη και αποτελεσματική τεχνική με τη χρήση του πρωτόκολλου μεταφοράς αρχείων (FTP) διπλής κατεύθυνσης για το κατέβασμα μεγάλων αρχείων από εξυπηρετητές αποθήκευσης δεδομένων ΥΝ. Αυτή η τεχνική προσφέρει αποτελεσματική ΕΦ μεταξύ διαφορετικών εξυπηρετητών με ελάχιστο επίπλεον φορτίο. Χρησιμοποιεί την λογική της επεξεργασίας των αρχείων σε δυο διαφορετικές κατευθύνσεις. Έτσι αν δύο αντίγραφα από ένα αρχείο υπάρχουν σε δύο διαφορετικούς εξυπηρετητές, ο ένας στέλνει την αρχή του αρχείου και ο άλλος το τέλος του, και συνεχίζουν μέχρι να σταλεί όλο το αρχείο. Αυτό παρέχει αυτόματη ΕΦ, παρέχοντας σε κάθε εξυπηρετητή να παρέχει υπηρεσίες ανάλογα με την κατάσταση του. Αυτό έχει σαν αποτέλεσμα, η χρήση του δικτύου να μεγιστοποιείται, ενώ οι πόροι του συστήματος και το φορτίο να διαμοιράζονται δυναμικά και η ΕΦ να παραμένει αποτελεσματική. Πάντως, ο χρόνος για το κατέβασμα των αρχείων είναι υψηλός, και απαιτείται να υπάρχουν αντίγραφα από όλα τα αρχεία σε όλους τους εξυπηρετητές.

Μία ακόμα τεχνική [46], προτείνει ένα μηχανισμό ΕΦ με βάση την πολιτική δέσμευσης για την κατανομή των εργασιών μεταξύ των εξυπηρετητών που περιέχουν αντίγραφα των αρχείων. Η μέθοδος επιτρέπει στους υπερφορτωμένους εξυπηρετητές να δεσμεύουν χώρο από απομακρυσμένους εξυπηρετητές πριν στείλουν σε αυτούς κάποιες εργασίες. Αυτός ο δεσμευμένος χώρος δεν μπορεί να χρησιμοποιηθεί από άλλους εξυπηρετητές, με αποτέλεσμα οι απομακρυσμένοι εξυπηρετητές να μην είναι ποτέ υπερφορτωμένοι. Έτσι ο μέσος χρόνος απόκρισης

μειώνεται, ο αριθμός των εργασιών που απορρίπτονται είναι χαμηλός, υπάρχει μεγαλύτερη ανεκτικότητα στις αλλαγές του φορτίου και καλύτερη ισορροπία στο σύστημα. Το μειονέκτημα είναι ότι ακόμα υπάρχουν αιτήσεις που απορρίπτονται.

Συνοψίζοντας στις τεχνικές δυναμικής ΕΦ, η επικοινωνία και ο συντονισμός μεταξύ των κόμβων για τις εργασίες είναι ζωτικής σημασίας. Συνοπτικά, το κύριο χαρακτηριστικό αυτών των τεχνικών είναι ότι οι κόμβοι μπορούν να συντονιστούν μεταξύ τους αυτόνομα (χωρίς κεντρική διαχείρηση) για την κατανομή εργασιών. Με την αυτόνομη και κατανεμημένη κατανομή εργασιών, ο έλεγχος μπορεί να προσαρμοστεί για μεταβαλλόμενες συνθήκες συστήματος (όχι στατικές) ακόμα και την πιθανή αποτυχία ορισμένων κόμβων. Έτσι ενώ η αντοχή είναι υψηλή και υπάρχει δυνατότητα για μεγάλης κλίμακας (large scale) συστήματα, κάθε κόμβος έχει μόνο πληροφορίες σχετικά με τους γείτονες του. Έτσι, τα αποτελέσματα μπορεί να είναι μόνο τοπικά (σε υποδίκτυα) βέλτιστα αλλά όχι απαραίτητα βέλτιστα και στο σύνολο των κόμβων. Επιπλέον, ακόμα και η επικοινωνία μεταξύ των κόμβων-γειτόνων μπορεί να οδηγήσει σε πρόσθετο υπολογιστικό κόστος.

2.2.3 Τεχνικές Υβριδικής Εξισορρόπησης Φορτίου

Σε περιπτώσεις που κανένα από τα παραπάνω μοντέλα δεν είναι κατάλληλο, εξετάζεται και η ενδιάμεση λύση των υβριδικών τεχνικών. Βασικό χαρακτηριστικό είναι ότι κάποιοι κόμβοι στο δίκτυο έχουν οριστεί ως διαχειριστές, αναλαμβάνοντας το συντονισμό με τους υπόλοιπους κόμβους, για τον μοιρασμό των απαιτούμενων πόρων και τον έλεγχο της κατανομής των εργασιών. Έτσι ενώ υπάρχει ακόμα το πλεονέκτημα του κατανεμημένου συστήματος, με ανοχή σε μεταβολές, υπάρχει ταυτόχρονα και μία μορφή κεντρικού ελέγχου για τις λειτουργίες της εξισορρόπηση φορτίου. Σε αυτή την ενότητα, θα περιγράψουμε τις τεχνικές υβριδικής ΕΦ και τα βασικά χαρακτηριστικά των δημοφιλέστερων αλγορίθμων. Οι υβριδικοί μηχανισμοί προτάθηκαν για να ξεπεραστούν τα μειονεκτήματα των στατικών και δυναμικών τεχνικών, με την σύνθεση τους διατηρώντας όμως τα πλεονεκτήματα του κάθε μηχανισμού ξεχωριστά.

Μία τεχνική υβριδικής ΕΦ [47] πρότεινε ένα κλιμακώσιμο μοντέλο με πολλά χαρακτηριστικά για την διάχυση της πληροφορίας και την εύρεση πόρων σε ΥΝ. Χρησιμοποιώντας ένα όμοτιμο δίκτυο χρηστών με έναν κεντρικό κόμβο για να συντονίζει τις λειτουργίες. Ο κάθε κόμβος θεωρείται γείτονας με άλλους κόμβους σε μία συγκεκριμένη ομάδα πόρων. Επιπλέον κάθε κόμβος έχει ένα επιτρεπτό όριο αποδεκτών αιτημάτων το οποίο εξαρτάται από τους πόρους του. Ο αλγόριθμος εκμεταλλεύεται τις συνδέσεις μεταξύ των κόμβων που έχουν πλεόνασμα πόρων και τη γνώση που έχει για τους γειτονικούς κόμβους και αν ένας κόμβος είναι υπερφορτωμένος στέλνει τα αιτήματα που του έρχονται σε κάποιον γείτονα. Τα αποτελέσματα δείχνουν ότι για μικρό αριθμό κόμβων οι υπερφορτωμένοι κόμβοι μπορούν να μεταφέρουν φορτίο στους

υποφορτωμένους αλλά υπάρχει μεγάλη καθυστέρηση λόγω του δικτύου και μερικές αποτυχίες στην μεταβίβαση των αιτημάτων από κόμβο σε κόμβο.

Για την επίλυση του προβλήματος του προγραμματισμού των ΕΜ προτάθηκε στο [48] η χρήση της υπάρχουσας πληροφορίες σε κάθε φυσικό μηχάνημα για να ανταπεξέλθει στις νέες αιτήσεις που καταφτάνουν. Επίσης για να μειωθεί ο χρόνος υπολογισμού της διαδικασίας προγραμματισμού των ΕΜ, προτάθηκε μία διαδικασία απόρριψης αιτημάτων τα οποία δεν θα μπορούσαν να ικανοποιηθούν στην υπάρχουσα κατάσταση. Η μέθοδος έφερε ισορροπημένο αποτέλεσμα μεταξύ των κόμβων συγκριτικά με παρόμοιες μεθόδους αλλά εφαρμόζεται μόνο σε ομοιογενείς εξυπηρετητές.

Για να ληφθεί υπόψη η συμπεριφορά της δυνατότητας χρήσης των πόρων στην απόφαση για ΕΦ παρουσιάστηκε ένα μοντέλο [49] όπου οι κόμβοι του συστήματος ανταλλάσουν πληροφορίες μεταξύ τους για την κατάσταση τους. Κάθε κόμβος έχει λειτουργία για ΕΦ αλλά και μία για εύρεση πόρων. Ανταλλάσοντας μηνύματα μεταξύ τους, κάθε κόμβος έχει πλήρη εποπτεία για το τι γίνεται στο σύστημα. Το μοντέλο μειώνει το χρόνο απόκρισης μεταξύ των μονάδων. Ως εκ τούτου, υπάρχει υψηλό επίπεδο κλιμακωσιμότητας, αλλά ο χρόνος απόκρισης είναι χαμηλός και υπάρχει μία αύξηση λόγω της διαδικασίας εύρεσης πόρων.

Για την κατανομή του φορτίου σε δυναμικά και κλιμακώσιμα συστήματα, προτάθηκε επίσης [50] μία κατηγορία αλγορίθμων Join-Idle-Queue, η οποία λύνει δύο προβλήματα. Για να λύσει το πρόβλημα της ανάθεσης εργασιών σε επεξεργαστές με ΕΦ λύνεται πρώτα το πρόβλημα της ανάθεσης ανενεργών επεξεργαστών σε ένα μοντέλο διαχείρησης τους. Ενώ η μείωση του μέσου μήκους της ουράς αναμονής για κάθε επεξεργαστή είναι η κύρια επιδίωξη, επιδιώκεται παράλληλα η εύκολη πρόσβαση σε ανενεργούς επεξεργαστές. Ο αλγόριθμος δεν έχει καθόλου επικοινωνία που να επιφέρει επιπλέον υπολογιστικό φορτίο, και έχει χαμηλή πολυπλοκότητα.

Για να ξεπεραστούν οι αποτυχίες στους εξυπηρετητές όταν μεγάλος αριθμός χρηστών δοκιμάζει να έχει πρόσβαση στις υπηρεσίες των ΥΝ, έχει προταθεί [51] μία δυναμική αρχιτεκτονική ΕΦ σε ΥΝ, που λαμβάνει υπόψη την επεξεργαστική ισχύ και τον υπολογιστικό φόρτο και έτσι διασφαλίζει ότι θα είναι πιο απίθανο ένας εξυπηρετητής να μην καταφέρει να διαχειριστεί ένα μεγάλο όγκο υπολογισμών. Στην ίδια δημοσίευση προτάθηκε ένας αλγόριθμος ΕΦ ικανός να εφαρμοστεί και σε εικονικούς διαδικτυακούς εξυπηρετητές αλλά και σε φυσικούς εξυπηρετητές, αλλά πρότεινε και την διαχείριση μίας πλατφόρμας που περιέχει όλα τα δεδομένα που χρειάζονται για την ΕΦ. Τα αποτελέσματα δείχνουν ότι τα ΥΝ που χρησιμοποίησαν την συγκεκριμένη αρχιτεκτονική μπορεί να εξισορροπήσει το φορτίο με υψηλά κλιμακώσιμη απόδοση. Όμως ο μέσος χρόνος απόκρισης εργασιών μεγαλώνει ανάλογα με τον αριθμό των συνδέσεων μεταξύ εικονικών και πραγματικών εξυπηρετητών.

Ακολουθεί ένας συγκεντρωτικός πίνακας για να καταδειχθεί ο διαφορετικός ρόλος της κάθε τεχνικής.

Τεχνικές Ελέγχου	Ελεγκτής	Λειτουργία κόμβων	Κλίμακα	Αξιοπιστία
Στατικές	Κεντρικός Ελεγκτής	Παθητικός ρόλος	Μικρή	Κακή
Δυναμικές	Δεν υπάρχει	Αυτόνομος ρόλος	Μεγάλη	Καλή
Υβριδικές	Μερικοί κόμβοι είναι ελεγκτές	Παθητικός ή Αυτόνομος ρόλος	Μεσαία	Μεσαία

Πίνακας 2.1: Οι συγκρίσεις μεταξύ των τριών τεχνικών ελέγχου ΕΦ

Κεφάλαιο 3

Σχεδίαση Συστήματος

Στην ενότητα αυτή περιγράφονται κάποιες βασικές και χρήσιμες έννοιες καθώς και το πρόβλημα της εξισορρόπησης φορτίου σε ένα δημόσιο ασύρματο μητροπολικό δίκτυο -ΔΑΜΔ.

3.1 Θεωρία Γραφημάτων

Ένας κατευθυνόμενος γράφος (directed graph) είναι ένα διατεταγμένο ζεύγος από σύνολα (V, E), όπου V είναι το σύνολο των κόμβων του γράφου με πλήθος $|V|$ και E είναι το σύνολο των ακμών πλήθος $|E|$. Αναπαριστούμε μία ακμή που ενώνει δύο κόμβους i, j, ως (i, j). Θα χρησιμοποιούμε τη μεταβλητή n για να υποδηλώσουμε τον αριθμό των κόμβων και m τον αριθμό των ακμών. Ένας υπογράφος (subgraph) $G' = (V', E')$ είναι ένας υπογράφος του G , αν περιέχει κάποιους από τους κόμβους ή τις ακμές του G. Εάν $V' = V$, ανεξάρτητα από ποιές ακμές περιέχει, τότε ο G' καλείται επικαλύπτον ή συνδετικός (spanning) υπογράφος του G.

Δύο κόμβοι $u, v \in V$ καλούνται γείτονες (neighbours) όταν $(u, v) \in E$. Ένας συνηθισμένος τρόπος για την αναπαράσταση ενός γράφου είναι με τον πίνακα γειτνίασης (adjacency matrix) του, που συμβολίζεται $A = [a_{ij}]$ και έχει διάσταση $n \times n$, όπου $a_{ij} = 1$, όταν υπάρχει σύνδεση μεταξύ i και j, ενώ σε αντίθετη περίπτωση $a_{ij} = 0$. Μία άλλη μορφή αναπαράστασης των συνδέσεων ενός γράφου είναι μέσω της λίστας γειτνίασης (adjacency list), όπου για κάθε κόμβο υπάρχει μία λίστα των συνδέσεων-γειτόνων του. Ένας γράφος (κατευθυνόμενος ή μη-κατευθυνόμενος) είναι γράφος με βάρη (weighted), όταν μία μετρήσιμη ποσότητα ,που συνήθως αναφέρεται ως βάρος και συμβολίζεται με w, ανατίθεται σε κάθε ακμή.

3.1.1 Θεωρία Συστημάτων Αναμονής

Η θεωρία αναμονής εξετάζει τα φαινόμενα, τα οποία παρατηρούνται σε ουρές που σχηματίζονται οποτεδήποτε φτάνουν πελάτες σε έναν σταθμό εξυπηρέτησης (κόμβο). Αν ο πελάτης φτάσει και βρει όλους τους εξυπηρετητές στον κόμβο απασχολημένους, τότε πρέπει να περιμένει σε κάποια ουρά μέχρι να ελευθερωθεί κάποιος εξυπηρετητής. Οι ακολουθίες πελατών εκφράζονται από κατανομές Poisson(λ) που εκφράζουν κάθε πότε φτάνει κάποιος καινούργιος πελάτης στην ουρά αναμονής. Στα μοντέλα αναμονής (συστήματα με ένα σταθμό εξυπηρέτησης), το λ_n είναι ο μέσος ρυθμός αφίξεων, το μ_n είναι ο ρυθμός εξυπηρέτησης (Service Rate), όπου n εννοείται το πλήθος των μονάδων εξυπηρέτησης στο σύστημα. Ακόμα το $\rho_n(t)$ είναι η ένταση κυκλοφορίας ή αλλιώς βαθμός χρησιμοποιήσης της μονάδας εξυπηρέτησης και εκφράζει την πιθανότητα να περιμένει κάποιος πελάτης στην ουρά, δηλαδή την πιθανότητα το σύστημα να είναι απασχολημένο. Δίνεται από τη σχέση $\rho = \frac{\lambda_i}{\mu_i}$ για κάθε σταθμό εξυπηρέτησης.

Στο σύστημα αναμονής $M/M/c/ : c = n$ μονάδες εξυπηρέτησης, που θα συναντήσουμε παρακάτω, η πιθανότητα να περιμένει ένας πελάτης στην ουρά δίνεται από τον παρακάτω τύπο, ο οποιός λέγεται και Erlang C formula και χρησιμοποιείται ευρέως στην Τηλεφωνία:

$$C(n,\rho) = \frac{(\frac{(n\rho)^c}{n!})(\frac{1}{1-\rho})}{\sum_{k=0}^{n-1}\frac{(np)^k}{k!} + (\frac{(n\rho)^c}{n!})(\frac{1}{1-\rho})}$$

Αυτός ο τύπος χρησιμοποιείται για τον προσδιορισμό του αριθμού των αντιπροσώπων εξυπηρέτησης πελατών που χρειάζονται για να στελεχώσουν ένα τηλεφωνικό κέντρο, για μια συγκεκριμένη επιθυμητή πιθανότητα αναμονής. Ωστόσο, ο τύπος Erlang C υποθέτει ότι οι καλούντες δεν κλείνουν ποτέ ενώ βρίσκονται στην ουρά, γεγονός που καθιστά τον μαθηματικό τύπο, να προβλέπει ότι πρέπει να χρησιμοποιηθούν περισσότεροι πράκτορες από ό,τι είναι πραγματικά απαραίτητο για να διατηρηθεί το επιθυμητό επίπεδο υπηρεσίας.
Στο μοντέλο $M/M/c/$: ο χρόνος απόκρισης (response time) είναι ο συνολικός χρόνος που ο πελάτης δαπανά τόσο στην ουρά όσο και στην υπηρεσία. Ο μέσος χρόνος απόκρισης είναι ο ίδιος για όλους τους κλάδους υπηρεσιών συντήρησης και είναι:

$$T_i(\lambda_i) = \frac{C(n_i, \frac{\lambda_i}{\mu_i})}{n_i\mu_i - \lambda_i} + \frac{1}{\mu_i},$$

όπου $C(n_i, \frac{\lambda_i}{\mu_i})$ υπολογίζεται από τον τύπο Erlang C formula.

3.2 Εισαγωγή στο ΥΚΝ

Όπως είδαμε τα μοντέλα ΥΚΝ συνδέονται άμεσα με τα κατανεμημένα συστήματα. Το κατανεμημένο σύστημα είναι μια συλλογή από αυτόνομους υπολογιστές που συνδέονται μεταξύ τους μέσω ενός δικτύου και χρησιμοποιούν ειδικά σχεδιασμένο λογισμικό για την παροχή ενοποιημένων υπολογιστικών υπηρεσιών. Χωρίς βλάβη της γενικότητας, τα καθήκοντα σε κατανεμημένα συστήματα υλοποιούνται πάντα με την πρόσβαση σε απαιτούμενους πόρους που υπάρχουν στο δίκτυο. Δύο βασικές λειτουργίες που συναντάμε στα ΥΚΝ συστήματα είναι η μεταφόρτωση υπολογισμών (ΜΥ) computation offloading και η εξισορρόπηση φορτίου. Ακολουθεί ο τυπικός ορισμός της μεταφόρτωση υπολογισμών σε κατανεμημένα συστήματα, η οποία υλοποιείται με βάση τις απαιτήσεις σε πόρων των καθηκόντων:

Ορισμός 3.1. ***Μεταφόρτωση υπολογισμών σε Κατανεμημένα Συστήματα*** *Δεδομένου ενός γράφου, $N =< A, E >$, όπου A είναι το σύνολο των κόμβων και κάθε κόμβος διαθέτει διαφορετικό σύνολο πόρων, και για κάθε $< a_i, a_j >$ υπάρχει E που υποδηλώνει την ύπαρξη ενός συνδέσμου δικτύου (ακμή) μεταξύ του κόμβου a_i και a_j. Το σύνολο των πόρων στον κόμβο a_i είναι R_{a_i}, και το σύνολο των πόρων που απαιτούνται από το έργο t είναι R_t. Αν η εργασία t φτάσει στο δίκτυο, η κατανομή των εργασιών στο N μπορεί να οριστεί ως η ανάθεση της εργασίας t σε ένα σύνολο από κόμβους, A_t, οι οποίοι πρέπει να ικανοποιούν τις ακόλουθες προϋποθέσεις:*

1. *Οι απαιτήσεις πόρων του t μπορούν να ικανοποιηθούν. Δηλαδή οι πόροι που απαιτεί μία εργασία είναι λιγότεροι από τους πόρους που έχουν συνολικά οι κόμβοι στους οποιούς έχει ανατεθεί αυτή. $R_t \subseteq \cup^{a_i \in A_t} R_{a_i}$.*

2. *Ο προκαθορισμένος στόχος του μοντέλου εκφόρτωσης, μπορεί να επιτευχθεί από την εκτέλεση της εργασίας του A_t. Για παράδειγμα την ελαχιστοποίηση του χρόνου εκτέλεσης [85] ή την μεγιστοποίηση της αξιοπιστίας [86].*

3. *Οι κόμβοι του συνόλου A_t μπορούν να εκτελέσουν τις εργασίες που τους έχουν ανατεθεί στο πλαίσιο που καθορίζεται από τους περιορισμούς της δομής του δικτύου, π.χ. για κάθε a_t, a_j υπάρχει $A_t \Rightarrow P_{ij} \subseteq E$, όπου το P_{ij} υποδηλώνει τη διαδρομή αλληλεπίδρασης (δικτύωσης) μεταξύ a_i και a_j.*

Σύμφωνα με τον ορισμό 2.1, εάν ένας κόμβος έχει επιπλέον διαθέσιμους πόρους για τις εργασίες, μπορεί τελικά να του ανατεθούν περισσότερα καθήκοντα από κάποιον κόμβο που έχει λιγότερους πόρους. Ωστόσο, εάν υπάρχουν πάρα πολλές εργασίες που έχουν ανατεθεί σε ορισμένους κόμβους τα καθήκοντα (και οι πόροι που απαιτούν), θα χρειάζονται πολύ περισσότερο χρόνο αναμονής. Επομένως, θα πρέπει τώρα να εφαρμόσουμε εξισορρόπηση φορτίου μεταξύ των κόμβων αφού έχει συμβεί η κατανομή των εργασιών.

Ορισμός 3.2. ***Εξισορρόπηση φορτίου-Load Balancing σε Κατανεμημένα Συστήματα***

Δεδομένου ενός γράφου, $N =< A, E >$, όπου A είναι το σύνολο των κόμβων, για κάθε a_i που ανήκει στο A η ομάδα των εργασιών που βρίσκοντα στην ουρά για τον πόρο r_k του κόμβου a_i μπορεί να οριστεί ως Q_{ik}. Το μέγεθος του Q_{ik} είναι s_{ik} και η ικανότητα επεξεργασίας του a_i είναι v_i, η αρχική πιθανότητα του κόμβου a_i να λάβει καθήκοντα (που χρειάζονται πόρους τύπου k) είναι $P_i(k)$, η οποία μπορεί να υπολογιστεί λαμβάνοντας υπόψη τους πόρους του a_i. Θα πρέπει να κάνουμε εξισορρόπηση φορτίου όταν το s_{ik} είναι πάρα πολύ μεγάλο, η οποία εφαρμόζεται με την μείωση της πιθανότητας για την λήψη νέων καθηκόντων των $a_i's$, $P_i(k)$, του σε μια αναθεωρημένη πιθανότητα, $DP_i(k)$:

$$DP_i(k) = \psi(s_{ik}/v_i) * P_i(k), \qquad (1)$$

όπου το ψ είναι μία συνάρτηση εξασθένησης, $0 \leq \psi(s_{ik}/v_i) \leq 1$. Η τιμή του $\psi(s_{ik}/v_i)$ μειώνεται μονότονα από το 1 στο 0 καθώς η τιμή του s_{ik}/v_i αυξάνει.

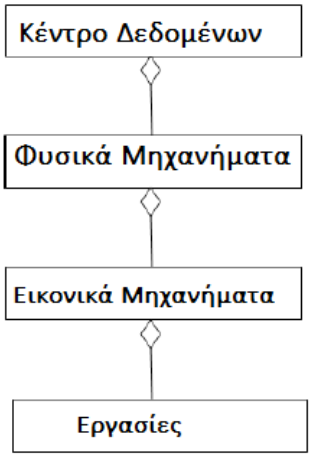
3.3 Εξισορρόπηση Φορτίου σε Υπολογιστικό Νέφος

Οι δύο βασικές μεθοδολογίες για να επιτευχθεί η ΕΦ σε ένα κατανεμημένο περιβάλλον, όπως είδαμε και σε προηγούμενα, είναι ο διαμοιρασμός πόρων με κατάλληλο τρόπο στους κόμβους του συστήματος και ο καταμερισμός (δρομολόγηση) των εργασιών στους κόμβους.

Στο πλαίσιο της διπλωματικής θεωρούμε 4 βασικές οντότητες. Αυτές οι οντότητες αναπαριστούν ένα βασικό ΥΝ. Το κέντρο δεδομένων (Datacenter) έχει την ευθύνη της παροχής υπηρεσιών υποδομής στους χρήστες. Οι φιλοξενούμενοι στο ΥΝ είναι τα φυσικά μηχανήματα (Hosts) που έχουν προεπιλεγμένες ικανότητες επεξεργασίας, έχουν δικιά τους μνήμη και δυνατότητα αποθήκευσης, και λειτουργούν ως οικοδεσπότες για τα εικονικά μηχανήματα -ΕΜ (Virtual Machines). Τέλος θεωρούμε ότι στο δίκτυο εισάγονται εργασίες που χρησιμοποιούν τους υπολογιστικούς πόρους του συστήματος (δικτύου).

1. *Αντιστοίχηση των εικονικών μηχανημάτων στα φυσικά μηχανήματα*

   Τα ΕΜ αντιστοιχούνται στα φυσικά μηχανήματα ανάλογα με τα χαρακτηριστικά που ταιριάζουν (μνήμη, λογισμικό, διαθέσιμες απαιτήσεις). Περισσότερα από ένα ΕΜ μπορούν να αντιστοιχιστούν σε ένα φυσικό μηχανήμα (Host) ανάλογα με τη διαθεσιμότητα και τις δυνατότητες του. Το κάθε φυσικό μηχανήμα είναι υπεύθυνο για την εκχώρηση πυρήνων επεξεργασίας σε αυτά, χρησιμοποιόντας μία πολιτική προβλέψεων που καθορίζει την κατανομή των πόρων κατόπιν ζήτησης. Αυτή η πολιτική, πρέπει επίσης, να διασφαλίζει ότι τα κρίσιμα χαρακτηριστικά του φυσικού μηχανήματος με κάθε εικονικό μηχάνημα δεν είναι ασυμβίβαστα ή ακόμα ότι δεν υπάρχει ζήτηση για περισσότερους πόρους από όσους μπορεί να διαθέσει.



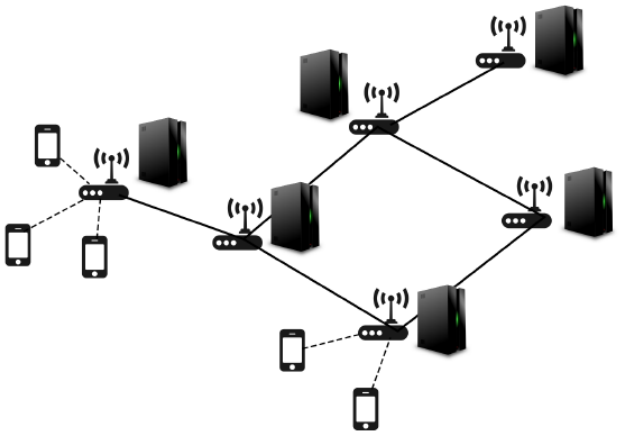
Σχήμα 3.1: Διάγραμμα Κλάσεων του υπολογιστικού νέφους

2. *Αντιστοίχηση των εργασιών σε εικονικά μηχανήματα*

   Τα ΕΜ είναι οι κεντρικές μονάδες επεξεργασίας για τις εφαρμογές-εργασίες που μπαίνουν στο δίκτυο και πρέπει να υλοποιηθούν. Κάθε εφαρμογή-εργασία απαιτεί ορισμένη ποσότητα υπολογιστικής ισχύος για την ολοκλήρωσή της, την οποία παρέχει το ΕΜ. Άρα μία εφαρμογή πρέπει να αντιστοιχίζεται σε κατάλληλο ΕΜ βάσει των απαιτήσεων της και των διαθέσιμων πόρων.

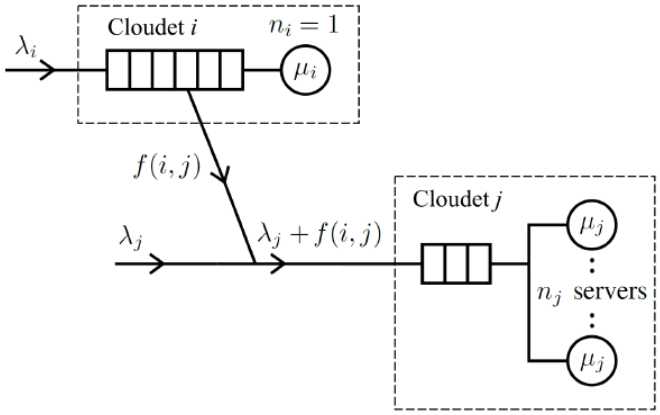
3.4 Γενική Περιγραφή του Προβλήματος

Υποθέτουμε ότι το δίκτυο στο οποίο θα αναφερθούμε είναι ένα Δημόσιο Ασύρματο Μητροπολιτικό Δίκτυο (Wireless Metropolitan Area Network) (ΔΑΜΔ), στο οποίο έχουν στηθεί K ΤΥΝ σε σταθερές θέσεις. Τα ΤΥΝ είναι συνδεδεμένα με σημεία ασύρματης πρόσβασης (Access Points-AP) και ταυτόχρονα είναι όλα συνδεδεμένα μεταξύ τους με τις ακμές του δικτύου, όπως φαίνετια στο Σχήμα 3.1.



Σχήμα 3.2: Ένα Ασύρματο Μητροπολιτικό Δίκτυο με σημεία πρόσβασης [25]

Υποθέτουμε ότι οι εφαρμογές που θα πρέπει να υλοποιήσουν τα cloudlets, είναι δυναμικά χωρισμένες σε ξεχωριστές εργασίες που μπορούν να τις επεξεργαστεί οποιοδήποτε από τα K cloudlets. Οπως φαινεται στο παρακατω σχημα, κάθε χρήστης θα μεταφορτώσει τις εργασίες που θέλει σε κοντινό του σημείο πρόσβασης, και αυτές θα μπουν στην ουρά αναμονής του κάθε ΤΥΝ.



Σχήμα 3.3: Ουρά Αναμονής σε ΤΥΝ [25]

Μοντελοποιούμε λοιπόν τα ΤΥΝ σαν ουρές αναμονής $M/M/n$ όπου το κάθε ένα (i) από τα K ΤΥΝ έχει δικούς του εξυπηρετητές (servers) n_i, με ρυθμό εξυπηρέτησης μ_i. Λόγω της ταχέως εναλλασσόμενης φύσης των απαιτήσεων των χρηστών, ο ρυθμός των εισερχόμενων αιτήσεων (εκφορτωμένες εργασίες) μπορεί να έχει μεγάλη διακύμανση. Για να μοντελοποιήσουμε μία πραγματική κατάσταση του ΔΑΜΔ, θεωρούμε ότι σε κάθε ΤΥΝ i οι εισερχόμενες εργασίες χρήστη καταφθάνουν τυχαία και ακολουθούν κατανομή Poisson με ρυθμό (λ_i). Ο μέσος χρόνος απόκρισης αποτελείται από τον χρόνο αναμονής και το χρόνο υπηρεσίας των εργασιών. Ορίζουμε ως T_i μία συνάρτηση για τον υπολογισμό του μέσου χρόνου απόκρισης των εργασιών σε ένα συγκεκριμένο ΤΥΝ. Σύμφωνα με την θεωρία συστημάτων αναμονής για τις ουρές $M/M/n$, τα T_i υπολογίζονται από τον παρακάτω τύπο:

$$T_i(\lambda_i) = \frac{C(n_i, \frac{\lambda_i}{\mu_i})}{n_i\mu_i - \lambda_i} + \frac{1}{\mu_i} \qquad (1)$$

όπου σύμφωνα πάλι με την θεωρία συστημάτων αναμονής το $C(n_i, \frac{\lambda_i}{\mu_i})$ υπολογίζεται από τον τύπο Erlang C.

$$C(n, \rho) = \frac{(\frac{(n\rho)^c}{n!})(\frac{1}{1-\rho})}{\sum_{k=0}^{n-1} \frac{(np)^k}{k!} + (\frac{(n\rho)^c}{n!})(\frac{1}{1-\rho})} \qquad (2)$$

Δεδομένου ότι οι ρυθμοί άφιξης των εργασιών σε διαφορετικά ΤΥΝ μπορεί να είναι σημαντικά διαφορετικοί, μερικά ΤΥΝ μπορεί να είναι υπερφορτωμένα ενώ άλλα ενδέχεται να υπολειτουργούν. Υποθέτουμε ότι όλα τα ΤΥΝ συνδέονται μεταξύ τους και ότι ένα ΤΥΝ μπορεί να ανακατευθύνει ένα μέρος του φόρτου εργασίας του σε ένα άλλο ΤΥΝ. Ορίζουμε ως $f(i, j)$ την ποσότητα φόρτου εργασίας που στέλνεται από το ΤΥΝ i στο ΤΥΝ j, βλέπε σχήμα 3.2. Οπότε προκύπτουν άμεσα οι παρακάτω λογικοί περιορισμοί για τις ροές μεταξύ των ΤΥΝ.

1. $f(i,j) = \begin{cases} -f(i,j) & \text{αν } i \neq j \\ 0 & \text{διαφορετικά} \end{cases}$, για κάθε από τα $K : i, j$ (3)

2. $\sum_{i=1}^{K} \sum_{j=1}^{K} f(i,j) = 0$ (4)

3. $\sum_{j=1}^{K} max\{f(i,j), 0\} \leq \lambda_i$, για κάθε i από τα K ΤΥΝ (5)

Η εξίσωση (3) διασφαλίζει ότι για οποιαδήποτε δύο i, j ΤΥΝ η ροή από το i στο j είναι η αντίστροφη από αυτή του j προς το i. Επίσης η ροή προς τον εαυτό του είναι λογικό να είναι μηδενική. Η εξίσωση (4) διασφαλίζει ότι η συνολική ροή του δικτύου διατηρείται. Δηλαδή, ότι δεν υπάρχει παραπάνω ροή στο δίκτυο από την συνολικά εισερχόμενη. Η εξίσωση (5) διασφαλίζει ότι σε κάθε ΤΥΝ το άθροισμα όλων των εξερχόμενων ροών από αυτό προς άλλα, δεν είναι μεγαλύτερο από την εισερχόμενο ρυθμό άφιξης εργασιών (λ_i).

Τέλος, θεωρούμε ότι όλες οι εκφορτωμένες εργασίες έχουν ίδιο μέγεθος πακέτου όσον αφορά την μεταφορά τους από ένα κόμβο του δικτύου σε άλλον, και άρα η καθυστέρηση της

μεταφοράς οποιουδήποτε πακέτου στην ίδια ακμή του δικτύου θα είναι πάντα ίδια. Για να μοντελοποιήσουμε αυτή την καθυστέρηση στο ΔΑΜΔ, θεωρούμε έναν πίνακα D, $K \times K$ διαστάσεων, όπου κάθε στοιχείο του πίνακα $d_{i,j}$ αναπαριστά την καθυστέρηση στην μεταφορά από το ΤΥΝ i στο j. Έτσι θεωρούμε ότι μία εισερχόμενη ροή εργασίας $f(i,j)$ έχει τελικά καθυστέρηση μεταφοράς: $-f(i,j) * d_{i,j}$. Η συνολική καθυστέρηση που προκύπτει από το δίκτυο για όλες τις ροές στο ΤΥΝ i είναι

$$T_{net}(i) = \sum_{j=1}^{K} |max(f(i,j), 0) * d_{i,j}| \tag{6}$$

Ο συνολικός χρόνος απόκρισης των εργασιών θα είναι τελικά η απόκριση του συνόλου των εργασιών που καταφθάνουν σε αυτό: λ_i (είτε από εισερχόμενη ροή από άλλους κόμβους, είτε από τον ρυθμό άφιξης στον συγκεκριμένο κόμβο) και δίνεται τελικά από τον τύπο:

$$D(i) = T_i(\overline{\lambda_i}) + T_{net}(i) \tag{7}$$

όπου το $\overline{\lambda_i}$ είναι όπως είπαμε:

$$\overline{\lambda_i} = \lambda_i - \sum_{i=1}^{K} f(i,j) \tag{8}$$

Τέλος πρέπει να επισημανθεί ότι όπως ξέρουμε από την θεωρία των συστημάτων αναμονής σε κάθε ΤΥΝ ο ρυθμός άφιξης λ_i στο ΤΥΝ i δεν μπορεί να ξεπερνάει την παρακάτω συνθήκη, γιατί τότε ο χρόνος αναμονής τείνει στο άπειρο

$$\lambda_i \leq \mu_i * n_i - 0.25$$

Κεφάλαιο 4

Υλοποίηση Αλγορίθμου Εξισορρόπησης Φορτίου

Στην ενότητα αυτή προτείνεται ένας γρήγορος αλγόριθμος με δυνατότητες επέκτασης, για την μείωση του μέσου χρόνου αποκρισης (average response time) των εργασιών. Αναλύονται επίσης οι αλγόριθμοι για την εύρεση της μέγιστης ροής καθώς και για την μέγιστη ροή ελάχιστου κόστους.

4.1 Το πρόβλημα

Το πρόβλημα της εξισορρόπηση φορτίου στον γράφο G του ΔΑΜΔ ορίζεται ως εξής. Έχουμε ένα σύνολο από K ΤΥΝ, όπου το κάθε ένα έχει λ_i ρυθμό άφιξης εργασιών, n_i αριθμό εξυπηρετητών (servers) με μ_i ρυθμό εξυπηρέτησης. Ζητείται να βρεθεί ένα σύνολο ροών εργασίας μεταξύ των κόμβων του δικτύου κάτω από τους περιορισμούς των εξισώσεων (3),(4),(5), έτσι ώστε η μέγιστη απόκριση για τις εργασίες μεταξύ των ΤΥΝ να είναι ελάχιστη. Με αυτό τον τρόπο το φορτίο μοιράζεται και επξεργάζεται από όλους τους κόμβους έτσι ώστε κανένα ΤΥΝ να μην μένει αναξιοποίητο ή να μην είναι υπερφορτωμένο από εργασίες.

4.2 Περιγραφή Υλοποίησης

Παραθέτουμε αρχικά έναν συγκεντρωτικό Πίνακα Συμβολισμών για καλύτερη κατανόηση.

Σύμβολο	Ορισμός-Περιγραφή
K	Ο αριθμός των ΤΥΝ στο δίκτυο
λ_i	Αρχικός ρυθμός άφιξης εργασιών στο ΤΥΝ i
n_i	Ο αριθμός των εξυπηρετητών στο ΤΥΝ i
$f(i,j)$	Η ροή που επαναδρομολογείται από το ΤΥΝ i στο ΤΥΝ j
$\overline{\lambda_i}$	Η τελική ροή εργασιών στο ΤΥΝ i
$T_i(\lambda)$	Η απόκριση στο ΤΥΝ i όταν ο ρυθμός άφιξης είναι λ_i
$T_{net}(i)$	Η καθυστέρηση που προκύπτει από το δίκτυο για την συνολική μεταφορά ροής στο ΤΥΝ i
$d_{i,j}$	Η καθυστέρηση του δικτύου μεταξύ των κόμβων i,j
$D(i)$	Η απόκριση στο ΤΥΝ i
ϕ_i	Το κομμάτι ροής που επαναδρομολογείται από ή προς το ΤΥΝ i
$\overline{D}$	Η επιθυμητή απόκριση τωβν εργασιών στα ΤΥΝ
T_{min}	Η ελάχιστη απόκριση βάσει ρυθμού άφιξης εργασιών μεταξύ των ΤΥΝ
T_{max}	Η μέγιστη απόκριση βάσει ρυθμού άφιξης εργασιών μεταξύ των ΤΥΝ
V_s	Το σύνολο των υπερφορτομένων ΤΥΝ
V_t	Το σύνολο των υποφορτομένων ΤΥΝ
ϵ, θ, δ	Όρια ακρίβειας που χρησιμοποιούνται στον αλγόριθμο
Δ	Η διαφορά μεταξύ της εξερχόμενης ροής εργασίας από τα υπερφορτομένων ΤΥΝ και την εισερχόμενη ροή εργασιών σε υποφορτομένων ΤΥΝ
$\overline{D}'$	Η μέγιστη απόκριση από το σύνολο των underloaded ΤΥΝ
$G = (V, E)$	Ένα δίκτυο ροής που χρησιμοποιείται για την εξισορρόπηση του φόρτου εργασίας των ΤΥΝ
$u(i,j)$	Η χωρητικότητα της ακμής i,j στο γράφο G
$c_{i,j}$	Η καθυστέρηση δικτύου της ακμής i,j στο γράφο G

Πίνακας 4.1: Πίνακας Συμβολισμών

4.2.1 Ο Αλγόριθμος

Σε αυτή την ενότητα προτείνουμε μια γρήγορη, και έυκολα κλιμακούμενη, λύση στο πρόβλημα εξισορρόπησης φορτίου σε ΤΥΝ. Η βασική ιδέα είναι να βρούμε έναν επιθυμητό μέσο χρόνο απόκρισης των εργασιών D πρώτα και στη συνέχεια να καθορίσουμε ένα μέρος του φόρτου εργασίας του κάθε ΤΥΝ που θα πρέπει να αναπροσανατολιστεί σε άλλα ΤΥΝ, έτσι ώστε η μέσος χρόνος απόκρισης των εργασιών να είναι περίπου D. Ψάχνουμε δηλαδή για κάθε ζευγάρι κόμβων στο δίκτυο, τη ροή φόρτου εργασίας που πρέπει να δρομολογηθεί από το ένα στο άλλο, για να έχουμε μία μορφή ισορροπίας για το μέσο χρόνο απόκρισης των εργασιών που έχουν μεταφορτωθεί. Διαχωρίζουμε το πρόβλημα σε δύο κομμάτια για να αναλυθούν καλύτερα τα επιμέρους αλγοριθμικά σκέλη της επίλυσης.

4.2.2 Η απόκριση των εργασιών

Α. Στόχος

Στόχος μας είναι να ελαχιστοποιήσουμε το μέγιστο μέσο χρόνο απόκρισης των εργασιών μεταξύ των ΤΥΝ. Καθώς η ροή εργασιών διατηρείται στο σύστημα (Εξίσωση (4)), όλες οι εργασίες πρέπει τελικά να εκτελεστούν. Είναι σαφές ότι πρέπει να προσπαθήσουμε να ανακατευθύνουμε ορισμένα καθήκοντα από και προς τα ΤΥΝ έτσι ώστε κάθε ΤΥΝ να έχει τον ίδιο μέσο όρο χρόνου απόκρισης των εργασιών. Η προσέγγιση που ακολουθείται είναι αρχικά να προσδιοριστεί έναν μέσο χρόνο απόκρισης των καθηκόντων μεταξύ των ΤΥΝ και κατόπιν να αποφασιστεί για κάθε υπερφορτωμένου ΤΥΝ η ποσότητα ροής εργασίας που πρέπει να δρομολογηθεί και αντίστοιχα για τα αναξιοποίητα ΤΥΝ η ποσότητα εισερχόμενου φόρτου εργασίας. Έτσι μπορεί να αποφασιστεί η συγκεκριμένη ροή από κάθε ΤΥΝ σε ένα άλλο.

Β. Καθορισμός του συνολικού μέσου χρόνου απόκρισης

Έστω $\overline{D}$ ο συνολικός μέσος χρόνος απόκρισης των εργασιών. Για να βρεθεί το $\overline{D}$ και το ποσό του εξερχόμενου / εισερχόμενου φόρτου εργασίας για κάθε ΤΥΝ, εκτιμούμε μία τιμή για το $\overline{D}$ και επεξεργαζόμαστε διαδοχικά την εκτίμηση μέχρι ο μέσος χρόνος απόκρισης των εργασιών σε κάθε ΤΥΝ (T_i) να είναι μέσα σε ένα συγκεκριμένο όριο ϵ. Η αρχική τιμή του $\overline{D}$ είναι $(T_{min} + T_{max})/2$, όπου θεωρούμε ότι

$T_{max} = max\{T_i(\lambda_i)\} \quad | \quad 1 \leq i \leq K$ και

$T_{min} = min\{T_i(\lambda_i)\} \quad | \quad 1 \leq i \leq K$

Γ. Ομαδοποίηση των ΤΥΝ και καθορισμός των ροών (flows)

Ταξινομούμε τα ΤΥΝ σε υπερφορτωμένα και υποφορτωμένα σύνολα ανάλογα με το αν η απόκριση στο συγκεκριμένο ΤΥΝ ξεπερνάει το $\overline{D}$. Δηλαδή:

$$V_s = \{i \mid T_i(\lambda_i\} < \overline{D}$$

$$V_t = \{j \mid T_j(\lambda_j\} \leq \overline{D}$$

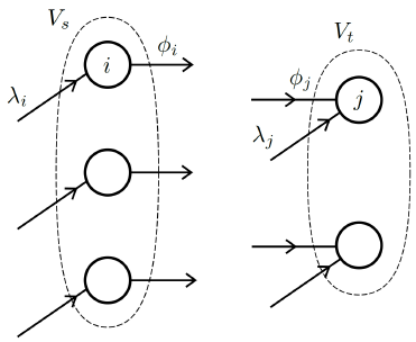
Για κάθε υπερφορτωμένο ΤΥΝ i, πρέπει όπως είπαμε να προσδιορίσουμε την ροή φορτίου ϕ_i που πρέπει να φύγει απο αυτό και να πάει σε κάποιο άλλο ώστε η απόκριση σε αυτό το ΤΥΝ να πλησιάζει το $\overline{D}$ που έχουμε εκτιμήσει. Διώχνοντας ένα κομμάτι ροής, ο ρυθμός άφιξης εργασιών αλλάζει και γίνεται : $\lambda_i - \phi_i$ Το όριο που θέτουμε είναι ϵ.

$$|\overline{D} - T_i(\lambda_i - \phi_i)| \leq \epsilon \qquad (9)$$

Για κάθε υποφορτωμένο ΤΥΝ j πρέπει να προσδιορίσουμε πάλι τη ροή φορτίου ϕ_j που πρέπει να έρθει σε αυτό από κάποιο άλλο, ώστε η απόκριση στο i να πλησιάζει στο $\overline{D}$ που έχουμε εκτιμήσει. Προσθέτοντας ένα κομμάτι ροής στο υπάρχον, ο συνολικός ρυθμός άφιξης στο j γίνεται $\lambda_j + \phi_j$. Πάλι το όριο που καθορίζουμε είναι ϵ.

$$|\overline{D} - T_j(\lambda_j + \phi_j)| \leq \epsilon \qquad (10)$$

Τα παραπάνω φαίνονται γραφικά στο παρακάτω σχήμα:



Σχήμα 4.1: Μια γραφική απεικόνιση για τα ϕ_i, ϕ_j [25]

Δ. Επιθυμητές ροές

Οι ροές που βρήκαμε παραπάνω είναι ιδανικά και δεν λαμβάνουν υπόψη την εξίωση (4) που δείχνει ότι η συνολική ροή του συστήματος είναι δεσμευμένη. Δεν αντιπροσωπεύουν, δηλαδή, τις ροές μεταξύ των ΤΥΝ $f(i, j)$ που θα φέρουν ισορροπια στο σύστημα. Ορίζουμε λοιπόν ως Δ τη διαφορά μεταξύ της εξερχόμενης ροής εργασίας από τα υπερφορτωμένα ΤΥΝ V_s και την εισερχόμενη ροή εργασιών σε υποφορτωμένα ΤΥΝ V_t.

$$\Delta = \sum_{i \in V_s} \phi_i - \sum_{j \in V_t} \phi_j \tag{11}$$

Αν ισχύει ότι $\Delta = 0$ τότε η εισερχόμενη ροή με την εξερχόμενη ταιριάζει απόλυτα, οπότε υπάρχει η επιθυμητη ισορροπία. Αν $\Delta > 0$ τότε υπάρχει ένα πλεόνασμα εξερχόμενης ροής. Δηλαδή τα ϕ_i που έχουμε επιλέξει ως ιδανικά από τα υπερφορτωμένα ΤΥΝ είναι περισσότερα σε τιμή από τα ϕ_j που έχουμε επιλέξει ως ιδανικά από τα υποφορτωμένα ΤΥΝ. Αυτό υπονοεί ότι η εκτίμηση που έχουμε κάνει για το $\overline{D}$ είναι πολύ μικρή με αποτέλεσμα πολλά υπερφορτωμένα ΤΥΝ να προσπαθούν να μεταφορτώνουν τις εργασίες τους. Για να λύσουμε αυτό το πρόβλημα αυξάνουμε το T_{min}.

$$T_{min} = \overline{D}$$

Έτσι στην επόμενη επανάληψη, το $\overline{D}$ θα είναι μεγαλύτερο λύνοντας το προβλημα καθώς το Δ θα τείνει στο 0. Σε πλήρη αντιστοιχία αν το $\Delta < 0$ τότε υπάρχει ένα πλεόνασμα εισερχόμενης τώρα ροής. Δηλαδή τα ϕ_j που έχουμε επιλέξει ως ιδανικά από τα υποφορτωμένα ΤΥΝ είναι περισσότερα σε τιμή από τα ϕ_i που έχουμε επιλέξει ως ιδανικά από τα υπερφορτωμένα ΤΥΝ. Αυτό υπονοεί ότι η εκτίμηση που έχουμε κάνει για το $\overline{D}$ (μέσος χρόνος απόκρισης εργασιών στο σύνολο του δικτύου) είναι πολύ μεγάλη (σε τιμή) με αποτέλεσμα πολλά υποφορτωμένα ΤΥΝ να δέχονται περισσότερη ροή εργασίας απ΄ότι χρειάζεται. Για να λύσουμε αυτό το πρόβλημα μειώνουμε το T_{max}.

$$T_{max} = \overline{D}$$

Έτσι στην επόμενη επανάληψη, το $\overline{D}$ θα είναι μικρότερο λύνοντας το προβλημα καθώς το Δ θα τείνει στο 0.

Στην ουσία σε κάθε επανάληψη διαλέγουμε $\overline{D} = (T_{min} + T_{max})/2$ και εφαρμόζουμε δυαδική αναζήτηση για να έχουμε όσο το δυνατόν πιο στενή εκτίμηση για το $\overline{D}$. Η αναζήτηση θα τερματίσει όταν το Δ πέσει σε ένα συγκεκριμένο όριο,δηλαδή

$$|\Delta| \leq \delta,$$

Ε. Επιπλέον καθορισμός του $\overline{D}$

Η ανακατεύθυνση ρόων εργασίας από τον έναν κόμβο στον άλλον στο δίκτυο αποφέρει μία καθυστέρηση δικτύου (network delay). Για αυτό το λόγο πρέπει να επανακαθορίσουμε το $\overline{D}$ για να συμπεριλάβουμε και αυτή την καθυστέρηση μετάβασης της ροής. Θέλουμε, δηλαδή, σε κάθε υποφορτωμένα ΤΥΝ το άθροισμα του χρόνου απόκρισης εργασίας αλλά και καθυστέρησης ροής προς τον κόμβο να πλησιάζει, όσο γίνεται, το χρόνο απόκρισης που έχουμε εκτιμήσει.

$$\overline{D} = T_j(\overline{\lambda_j}) + T_{net}(j)$$

Σε αυτό το σημείο, πρέπει να υπάρχει ένα έλλειμμα των εξερχόμενων εργασιών από τα υπερφορτωμένα ΤΥΝ. Στην συνέχεια ορίζουμε τη μέγιστη απόκριση από το σύνολο των υποφορτωμένα ΤΥΝ: $\overline{D}'$.

$$\overline{D}' = max\{D(j)|j \in V_t\}$$

Ο αλγόριθμος μας θα τερματίζει αν η διαφορά μεταξύ μέγιστης απόκρισης από τους υπερφορτωμένους κόμβους και η απόκριση που έχουμε εκτιμήσει είναι μεταξύ ενός προκαθορισμένου ορίου θ. Αν $\overline{D} < \overline{D}'$ τότε υπάρχει ένα πλεόνασμα εξερχόμενης ροής. Δηλαδή, τα ϕ_i που έχουμε επιλέξει ως ιδανικά από τα υπερφορτωμένα ΤΥΝ είναι περισσότερα σε τιμή από τα ϕ_j που έχουμε επιλέξει ως ιδανικά από τα υποφορτωμένα ΤΥΝ. Αυτό υπονοεί ότι η εκτίμηση που έχουμε κάνει για το $\overline{D}$ είναι πολύ μικρή με αποτέλεσμα πολλά υπεροφρτωμένα ΤΥΝ να προσπαθούν να μεταφορτώσουν (offload) τις εργασίες τους. Σε πλήρη αντιστοιχία αν το $\overline{D} < \overline{D}'$ τότε υπάρχει ένα πλεόνασμα εισερχόμενης τώρα ροής. Δηλαδή τα ϕ_j που έχουμε επιλέξει ως ιδανικά από τα υποφορτωμένα ΤΥΝ είναι περισσότερα σε τιμή από τα ϕ_i που έχουμε επιλέξει ως ιδανικά από τα υπερφορτωμένα ΤΥΝ. Αυτό υπονοεί ότι η εκτίμηση που έχουμε κάνει για το $\overline{D}$ (μέσος χρόνος απόκρισης εργασιών στο σύνολο του δικτύου) είναι πολύ μεγάλη (σε τιμή) με αποτέλεσμα πολλά υπερφορτωμένα ΤΥΝ να δέχονται περισσότερη ροή εργασίας απ΄ότι χρειάζεται.

Για να λύσουμε αυτά τα προβληματα αυξάνουμε ή μειώνουμε το $\overline{D}$. Σε κάθε επανάληψη ψάχνουμε ροές εισερχόμενες και εξερχόμενες ώστε να είναι στα όρια που θέτουν οι εξισώσεις (9),(10), και αφού υπολογίσουμε το $\overline{D'}$, επανακαθορίζουμε την εκτίμηση μας $\overline{D} = (\overline{D} + \overline{D'})/2$ και συνεχίζουμε μέχρι να πέσουμε στο επιθυμητό όριο. Αυτή η περιγραφή που προηγήθηκε είναι ο βασικός πυρήνας της λύσης μας και φαίνεται παρακάτω σε ψευδοκώδικα για καλύτερη κατανόηση.

Algorithm 1 *Cloudlet Load Balance Algorithm*

Input: The task arrival rate λ_i, the number of server n_i, and the service rate μ_i of each cloudlet $i \in \{1, \ldots, K\}$; the network delay matrix $d_{i,j}$ $(i, j \in \{1, \ldots, K\})$; and tuning parameters θ, ϵ, δ.

Output: Inter-cloudlet task flow $f(i, j)$ for each pair of cloudlets $i, j \in \{1, \ldots, K\}$.

$T_{\max} \leftarrow \max\{T_i(\lambda_i) \mid 1 \leq i \leq K\}$;
$T_{\min} \leftarrow \min\{T_i(\lambda_i) \mid 1 \leq i \leq K\}$;
$\Delta \leftarrow \infty$;
/* find the initial value of $\overline{D}$ using binary search */
while $(|\Delta| > \delta)$ **do**
  $\overline{D} \leftarrow (T_{\max} + T_{\min})/2$;
  $V_s \leftarrow \{i \mid T_i(\lambda_i) > \overline{D}\}$;
  $V_t \leftarrow \{i \mid T_i(\lambda_i) \leq \overline{D}\}$;
  for each overloaded cloudlet $i \in V_s$ **do**
    Find a value of ϕ_i such that $\left|\frac{\overline{D} - T_i(\lambda_i - \phi_i)}{\overline{D}}\right| \leq \epsilon$;
  end for
  for each underloaded cloudlet $j \in V_t$ **do**
    Find a value of ϕ_j such that $\left|\frac{\overline{D} - T_j(\lambda_j + \phi_j)}{\overline{D}}\right| \leq \epsilon$;
  end for
  $\Delta \leftarrow \sum_{i \in V_s} \phi_i - \sum_{j \in V_j} \phi_j$;
  if $|\Delta| > 0$ **then**
    $T_{\min} \leftarrow \overline{D}$;
  else
    $T_{\max} \leftarrow \overline{D}$;
  end if
end while
/* adjust the value of $\overline{D}$ for the network delay between cloudlets */
$V_s \leftarrow \{i \mid T_i(\lambda_i) > \overline{D}\}$;
$V_t \leftarrow \{j \mid T_j(\lambda_j) \leq \overline{D}\}$;
$\overline{D}' \leftarrow \infty$;
while $(|\overline{D} - \overline{D}'| > \theta)$ **do**
  for each $i \in V_s$ **do**
    Find a value of ϕ_i such that $\left|\frac{\overline{D} - T_i(\lambda_i - \phi_i)}{\overline{D}}\right| \leq \epsilon$;
  end for
  for each $j \in V_t$ **do**
    Find a value of ϕ_j such that $\left|\frac{\overline{D} - T_j(\lambda_j + \phi_j)}{\overline{D}}\right| \leq \epsilon$;
  end for
  Calculate $f(i, j)$, by invoking Procedure `minLatencyFlow`$(\overline{D}, V_s, V_t, \epsilon)$;
  for each $j \in V_t$ **do**
    Calculate $D(j)$, using Eq.(7);
  end for
  $\overline{D}' \leftarrow \max\{D(j) \mid j \in V_t\}$;
  $\overline{D} \leftarrow (\overline{D} + \overline{D}')/2$;
end while

Σχήμα 4.2: Αλγόριθμος υπολογισμού μέσου χρόνου απόκρισης [25]

4.2.3 Ροή ελάχιστης καθυστέρησης

Στη συνέχεια θα ακολουθήσει η διαδικασία εύρεσης των ζητούμενων $f(i,j)$ για όλα τα ΤΥΝ, που δημιουργούν τις μικρότερες καθυστερήσεις δικτύου ενώ ταυτόχρονα ικανοποιούν τις συνθήκες συντήρησης που ορίσαμε στις εξισώσεις (3),(4),(5). Ως εδώ έχουμε χρησιμοποιήσει δύο φορές δυαδική αναζήτηση για να κάνουμε μία εκτίμηση για τον μέσο χρόνο απόκρισης εργασιών που θεωρούμε ότι θα είναι κοντά στην πραγματικότητα και όλα τα ΤΥΝ θα μπορούν να ικανοποιήσουν, επαναδρομολογόντας ροή εργασίας ή δεχόμενα επιπλέον ροή εργασίας.

Α. Δημιουργία γράφου
Τροποποιούμε το πρόβλημα του προσδιορισμού των ποσών των εξερχόμενων ροών των υπερφορτωμένων ΤΥΝ στα υποφορτωμένα ΤΥΝ στο γνωστό πρόβλημα μέγιστη ροή ελάχιστου κόστους (minimum cost maximum flow problem).
Συγκεκριμένα, θέλουμε να φτιάξουμε ένα γράφο $G = (V, E)$ που αναπαριστά το ΔΑΜΔ. Όπως και σε κάθε γράφο που χρησιμοποιείται για κάποιο πρόβλημα μέγιστης ροής στους αλγορίθμους χρειαζόμαστε έναν αρχικό (source) κόμβο s καθώς και έναν τελικό (sink) κόμβο t, που αναπαριστούν την ροή που ξεκινάει από ένα σημείο αρχικό διασχίζει τον γράφο μας ανάλογα με την χωρητικότητα των ακμών και καταλήγει στον τελικό κόμβο. Έχοντας διαχωρίσει τους κόμβους μας σε δύο σύνολα (όπως προηγουμένως) σε υπερφορτωμένα και αναξιοποίητα ΤΥΝ ανάλογα με το $\overline{D}$ που βρήκαμε, έχουμε τελικά τα σύνολα κορυφών:

$$V = V_s \cup V_t \cup \{s, t\}$$

Στην συνέχεια για κάθε κόμβο των υπεφορτωμένων ΤΥΝ ενώνουμε με μία κατευθυνόμενη ακμή από την αρχική μας κορυφή s, ενώ για κάθε υποφορτωμένο ΤΥΝ ενώνουμε με μία ακμή με την τελική κορυφή t. Τέλος κάθε υπερφορτωμένο φτιάχνουμε μία ακμή προς κάθε υποφορτωμένο. Τελικά έχουμε το εξής σύνολο ακμών:

$$E = \{\langle s, i \mid i \in V_s\rangle \cup \langle j, t \mid j \in V_t\rangle \cup \langle i, j \mid (i \in V_s, j \in V_t)\}$$

Β. *Καθορισμός των ακμών*
Για την εφαρμογή των αλγορίθμων μέγιστης ροής πρέπει ακόμα να καθορίσουμε την χωρητικότητα $u(i,j)$ και τα κόστη $c_{i,j}$ των ακμών. Για τις χωρητικότητες: σε κάθε ακμή που ενώνει την αρχική κορυφή με οποιοδήποτε από τα i υπερφορτωμένα Cloudlets, θέτουμε χωρητικότητα ϕ_i, όπως βρήκαμε σε προηγούμενα βήματα.

$$u(s, i) = \phi_i$$

για κάθε $i \in V_s$.
Αντιστοίχως, σε κάθε ακμή που ενώνει την τελική κορυφή με οποιοδήποτε από τα υποφορτωμένα Cloudlets, θέτουμε χωρητικότητα ϕ_j, όπως βρήκαμε σε προηγούμενα βήματα.

$$u(j, t) = \phi_j$$

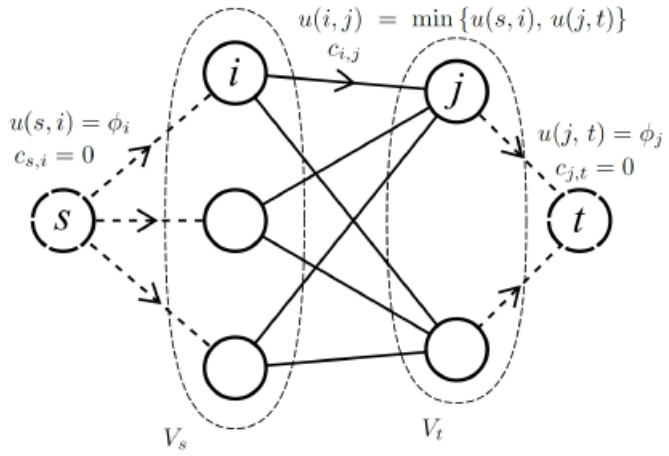
για κάθε $j \in V_t$.
Τέλος, για κάθε ακμή που ενώνει τα υπερφορτωμένο Cloudlets με τα υποφορτωμένο Cloudlets θέτουμε ως χωρητικότητα, την ελάχιστη από τις δύο παραπάνω χωρητικότητες που ορίσαμε, δηλαδή

$$u(i,j) = min\{u(s,i), u(j,t)\}$$

Για τα κόστη των ακμών για κάθε ακμή που ενώνει την αρχική κορυφή με τους κόμβους των υπερφορτωμένο ΤΥΝ ή για κάθε ακμή που ενώνει την τελική κορυφή με τους κόμβους των υποφορτωμένο ΤΥΝ, θέτουμε κόστος χρήσης μηδέν. Για τις υπόλοιπες ακμές που ενώνουν τα δύο σύνολα κόμβων θέτουμε κόστος όσο και την καθυστέρηση μετάδοσης μεταξύ των δύο ΤΥΝ δηλαδή:

$$c_{i,j} = d_{i,j}$$

Όλα τα παραπάνω φαίνοντα γραφικά στο παρακάτω σχήμα.



Σχήμα 4.3: Ο γράφος G για την μέγιστη ροή [25]

Έχοντας κατασκευάσει το δίκτυο ροής G, μπορεί να δει κανείς ότι το πρόβλημα της δρομολόγησης εξερχόμενων ροών από υπερφορτωμένα ΤΥΝ σε υποφορτωμένα, μετατρέπεται στο πρόβλημα του να βρεθεί μια μέγιστη ροή ελάχιστου κόστους στο G από s έως t συνυπολογίζοντας της καθυστέρησης του δικτύου. Δηλαδή, ο στόχος μας είναι να ελαχιστοποιήσουμε το

$\sum_{(i,j)\in E} f(i,j) * c_{i,j}$ (12)

χρησιμοποιόντας τις παρακάτω προυποθέσεις:

$f(i,j) \leq u(i,j)$ για κάθε i,j (13)

$f(i,j) = -f(j,i)$ για κάθε i,j εκτός s,t (14)

$\sum_{j\in V} f(i,j) = 0$ για κάθε i,j εκτός s,t (15)

όπου το $f(i,j) * c_{i,j}$ είναι η ποσότητα καθυστέρησης δικτύου που προκαλείται μεταφέροντας

εργασίες από ΤΥΝ i σε ΤΥΝ j.

Procedure 1 *minLatencyFlow*

Input: An average response time $\overline{D}$ of tasks; a set of overloaded cloudlets V_s; a set of underloaded cloudlets V_t; and a tuning parameter ϵ

Output: Redirected task flow $f(i,j)$ for all $i,j \in \{1,\ldots,K\}$.

1: /* Construct a flow network with latency weighted edges. */
2: $V \leftarrow \{1,\ldots,K\} \cup \{s,t\}$;
3: $V_s \leftarrow \{i \mid i \in \{1,\ldots,K\}, T_i(\lambda_i) > \overline{D}\}$;
4: $V_t \leftarrow \{j \mid i \in \{1,\ldots,K\}, T_j(\lambda_j) \leq \overline{D}\}$;
5: $E \leftarrow \emptyset$;
6: **for** each overloaded cloudlet $i \in V_s$ **do**
7: $E \leftarrow E \cup \{\langle s,i\rangle\}$;
8: Find a value of ϕ_i such that $\left|\frac{\overline{D}-T_i(\lambda_i-\phi_i)}{\overline{D}}\right| \leq \epsilon$;
9: $u(s,i) \leftarrow \phi_i$;
10: $c_{s,i} \leftarrow 0$;
11: **end for**
12: **for** each underloaded cloudlet $j \in V_t$ **do**
13: $E \leftarrow E \cup \{\langle j,t\rangle\}$;
14: Find a value of ϕ_j such that $\left|\frac{\overline{D}-T_j(\lambda_j+\phi_j)}{\overline{D}}\right| \leq \epsilon$;
15: $u(j,t) \leftarrow \phi_j$;
16: $c_{j,t} \leftarrow 0$;
17: **end for**
18: **for** each $i \in V_s$ **do**
19: **for** each $j \in V_t$ **do**
20: $E \leftarrow E \cup \{\langle i,j\rangle\}$;
21: $u(i,j) \leftarrow \min\{u(s,i), u(j,t)\}$;
22: **end for**
23: **end for**
24: Find a minimum-cost maximum-flow in the flow network $G(V,E)$, subject to edge capacities $u(u,v)$ for $\langle u,v\rangle \in E$, by invoking the transportation algorithm in [9].

Σχήμα 4.4: Η κατασκευή του γράφου G για την εύρεση των επιθυμητών ροών [25]

4.2.4 Αλγόριθμοι για Μέγιστη Ροή

Πέρα από τον αλγόριθμο που θα περιγράφτει στην επόμενη υποενότητα, υλοποιήσαμε και δύο αλγορίθμους για τον υπολογισμό της Μέγιστης Ροής σε ένα γράφο. Όπως θα φανεί στην επόμενη ενότητα οι μετρήσεις από τους παρακάτω αλγορίθμους έχουν το ίδιο αποτέλεσμα με τον αλγόριθμο για Μέγιστη Ροή Ελάχιστου Κόστους. Όμως επειδή το πρόβλημα της μέγιστης ροής είναι πιο απλουστευμένο οι πολυπλοκότητες είναι μικρότερες και άρα οι αλγόριθμοι είναι χρήσιμοι στο πεδίο εφαρμογής.

Ο αλγόριθμος Ford-Fulkerson [30]
Είναι ένας απλός αλγόριθμος για την εύρεση της μέγιστης ροής δικτύου και μια εφαρμογή στο πρόβλημα Hitchcock Transportation Problem. Συνοπτικά ο γνωστός αυτός αλγόριθμος έχει τις ίδιες προυποθέσεις που θέσαμε και εμείς προηγουμένως.

- Η ροή σε μια ακμή δεν μπορεί να υπερβαίνει τη δεδομένη χωρητικότητα της ακμής.
- Η εισερχόμενη ροή είναι ίση με την εξερχόμενη ροή για κάθε κορυφή εκτός από s και t.

Ας ορίσουμε πρώτα την έννοια του υπολειπόμενου γράφου (residual graph) που είναι απαραίτητη για την κατανόηση της υλοποίησης. Ο υπολειπόμενος γράφος ενός δικτύου ροής είναι ένας γράφος που υποδεικνύει μια επιπλέον πιθανή ροή. Εάν υπάρχει μια διαδρομή από την αρχική κορυφή (πηγή) έως την τελική κορυφή στο υπολειπόμενο γράφο, τότε είναι δυνατή η προσθήκη ροής. Κάθε ακμή ενός υπολειπόμενου γράφου έχει μια τιμή που ονομάζεται υπολειπόμενη χωρητικότητα που είναι ίση με την αρχική χωρητικότητα της ακμής μείον τη ροή ρεύματος. Η υπολειπόμενη χωρητικότητα είναι βασικά η τρέχουσα χωρητικότητα της ακμής. Ας μιλήσουμε τώρα για τις λεπτομέρειες του αλγορίθμου. Η υπολειπόμενη χωρητικότητα είναι 0, αν δεν υπάρχει ακμή μεταξύ δύο κορυφών του υπολειπόμενου γραφήματος. Μπορούμε να αρχικοποιήσουμε τον υπολειπόμενο γράφο ως τον αρχικό γράφο, καθώς δεν υπάρχει αρχική ροή και η συνολική υπολειπόμενη χωρητικότητα είναι ίση με την αρχική χωρητικότητα. Για να βρούμε μια διαδρομή αύξησης της ροής, μπορούμε να κάνουμε μία Αναζήτηση Κατά Πλάτος -ΑΚΠ (Breadth First Search) στο υπολειπόμενο γραφήμα. Χρησιμοποιώντας την ΑΚΠ, μπορούμε να ανακαλύψουμε εάν υπάρχει μια διαδρομή από s προς t. Η ΑΚΠ επίσης χτίζει έναν γονικό πίνακα, με ποια σειρά συναντάω τους κόμβους. Χρησιμοποιώντας τον γονικό πίνακα, διασχίζουμε τη διαδρομή που βρέθηκε και βρίσκουμε πιθανή ροή μέσω αυτής της διαδρομής, αναζητώντας την ελάχιστη υπολειπόμενη χωρητικότητα κατά μήκος της διαδρομής. Αν βρούμε, προσθέτουμε τη ροή της διαδρομής που βρέθηκε στη συνολική ροή. Το σημαντικό είναι ότι πρέπει να ενημερώσουμε τις υπόλοιπες υπολειπόμενες χωρητικότητες στον υπολειπόμενο γράφο. Αφαιρούμε τη ροή της διαδρομής από όλες τις ακμές κατά μήκος της διαδρομής και προσθέτουμε ροή ανάποδης διαδρομής κατά μήκος των ανάποδων ακμών. Πρέπει να προσθέσουμε ροή διαδρομής κατά μήκος των ανάποδων ακμών επειδή μπορεί αργότερα να χρειαστεί να στείλουμε ροή σε αντίστροφη διεύθυνση.

Algorithm 1 Ford-Fulkerson

1: **Procedure:** *The simple idea of Ford-Fulkerson*
2: Initialize with source node as 0.
3: While there is a path in the residual graph from source to sink node such as $c_f(u, v) > 0$ for every edge.
4: Find the minimum capacity between the edges in the residual graph
5: Add it to the path
6: **End Procedure**

Η πολυπλοκότητα του Αλγορίθμου είναι $O(V^2E)$ που στην περιπτωσή μας καταλήγει σε $O(V^4)$ αφού για τα κάθε κόμβο στο σύνολο των υπερφορτωμένων έχουμε μία ακμή προς κάθε κόμβο του συνόλου των υποφορτωμένων ΤΥΝ, οπότε $E = V^2$.
Ο αλγόριθμος Push-Relabel
Ο αλγόριθμος Push-Relabel είναι ένας από τους πιο αποτελεσματικούς αλγόριθμους για τον υπολογισμό μιας μέγιστης ροής. Ο γενικός αλγόριθμος έχει πολυπλοκότητα χρόνου $O(V^2E)$, ενώ η τροποποίηση με τον κανόνα επιλογής κορυφής Relabel to Front, όπως θα δούμε παρακάτω, έχει πολυπλοκότητα $O(V^3)$, πόλυ καλύτερο δηλαδή από τον προηγούμενο αλγόριθμο. Η βασική λογική του Push-Relabel είναι η παρακάτω.
Απαιτήσεις:

- Η ροή σε μια ακμή δεν μπορεί να υπερβαίνει τη δεδομένη χωρητικότητα της ακμής.
- Η εισερχόμενη ροή είναι μεγαλύτερη, ίση με την εξερχόμενη ροή για κάθε κορυφή εκτός από s και t.

Η δεύτερη συνθήκη είναι πιο ελαστική από την αντίστοιχη στον προηγούμενο αλγόριθμο που είναι και η βασική διαφορά στην υλοποίηση. Θεωρούμε λοιπόν ότι για κάθε κορυφή υπάρχει επιπλέον ροή αν η εισερχόμενη ροή είναι μεγαλύτερη από την εξερχόμενη. Επίσης ορίζουμε μία συνάρτηση height(v) που για κάθε κορυφή, δείχνει σε ποιο ύψος βρίσκεται. Αρχικά θεωρούμε τη διαδικασία preflow που ορίζει τις αρχικές συνθήκες.
Preflow

- Το ύψος όλων των κορυφών ίσο με μηδέν, εκτός από την αρχική μας κορυφή s η οποία έχει ύψος όσο και το πλήθος των κόμβων (συμπεριλαμβανομένου του s,t).
- Αρχικοποιούμε τη ροή σε κάθε ακμή ίση με μηδέν
- Για κάθε ακμή που συνδέεται με την αρχική κορυφή s, η ροή και η επιπλέον ροή αρχικοποιείται στην τιμή της χωρητικότητας της εκάστοτε ακμής.

Οι δύο βασικές υπορουτίνες στον αλγόριθμο είναι το push και το relabel.

Το push χρησιμοποιείται για να σπρώξει τη ροή από έναν κόμβο ο οποίος έχει επιπλέον ροή. Αν μια κορυφή έχει επιπλέον ροή και υπάρχει ένας διπλανός κόμβος με μικρότερο ύψος (στο υπολειπόμενο γράφημα), πιέζουμε τη ροή από την κορυφή προς το γειτονικό με το χαμηλότερο

ύψος. Η ποσότητα της ροής που πιέζεται μέσω μιας ακμής είναι ίση με το ελάχιστο από την επιπλέον ροή ή τη χωρητικότητας της ακμής.

Η λειτουργία Relabel χρησιμοποιείται όταν μια κορυφή έχει επιπλέον ροή και κανένας από τους γειτονικούς της κόμβους δεν βρίσκεται σε χαμηλότερο ύψος. Δηλαδή αυξάνουμε το ύψος της κορυφής έτσι ώστε να μπορούμε να εκτελέσουμε push. Το νέο ύψος, είναι το ελάχιστο ύψος (στο υπολειπόμενο γράφημα) των διπλανών κορυφών αυξανόμενο κατά ένα.

Ο Αλγόριθμος όταν τελειώνει έχει βρει την μέγιστη ροή ελάχιστου κόστους στο γράφο.

Algorithm 2 Push-Relabel

Preflow *Initialize Flows and Heights*
while (it is possible to perform a Push or Relablel on a vertex // Or while there is a vertex that has excess flow) **do**
  Do Push() or Relabel();
end while

Χρησιμοποιήσαμε την παραλλαγή Relabel To Front του αλγορίθμου Push-Relabel, που έχει την επιπλέον λειτουργία Discharge, για να γίνονται λιγότερες προσπάθειες για Push μεταξυ των κόμβων. Ο αλγόριθμος οργανώνει όλους τους κόμβους σε μια συνδεδεμένη λίστα και διατηρεί τη λίστα τοπολογικά ταξινομημένη σε σχέση με το αποδεκτό δίκτυο. Ο αλγόριθμος πραγματοποιεί σάρωση της λίστας από μπροστά προς τα πίσω και πραγματοποιεί την λειτουργία Discharge στον τρέχοντα κόμβο αν είναι ενεργός, μπορεί ακόμα να κάνει λειτουργίες Push. Αν ο κόμβος έχει κάνει Relabel, μετακινείται στο μπροστινό μέρος της λίστας και η σάρωση ξεκινάει πάλι από μπροστά.

Το *Discharge* προσπαθεί να κάνει push σε όσους γειτονικούς κόμβους δεν έχουν δοκιμαστεί από την τελευταία φορά που κάποιος κόμβος έκανε relabel. Αν δεν μπορεί να κάνει προώθηση σε κανέναν τότε κάνει ο ίδιος κόμβος relabel. Προφανώς για να γίνει αυτό πρέπει ο κάθε κόμβος να ξέρει την κατάσταση των υπολοίπων. Ο Αλγόριθμος 3 συνοψίζει την παραπανω διαδικασία με μορφη ψευδοκώδικα.

Algorithm 3 Push-Relabel to Front

Preflow *Initialize Flows and Heights. Send as much flow from s as possible.*
Build a list of all vertices except s and t.
while (As long as we have not traversed the entire list: **do**
  Discharge the current vertex.
  if If the height of the current vertex changed: **then**
    Move the current vertex to the front of the list
    Restart the traversal from the front of the list.
  end if
end while

4.2.5 Αλγόριθμος για Μέγιστη Ροή Ελάχιστου Κόστους

Για να λύσουμε αυτό το πιο σύνθετο πρόβλημα θεωρούμε ότι αρκεί να βρούμε μία Μέγιστη Ροή σε ένα γράφο (μπορεί να υπάρχουν και άλλες) χωρίς να μας ενδιαφέρει το κόστος, και μετά να βρούμε την Μέγιστη Ροή με το ελάχιστο κόστος. Η Μέγιστη Ροή μπορεί να βρεθεί με έναν από τους αλγορίθμους που περιγράφηκαν στην προηγούμενη υποενότητα. Ο αλγόριθμος αυτός είναι γνωστός στην βιβλιογραφία ως Cycle Cancelling.

Θεωρώντας λοιπόν δεδομένη μία Μέγιστη Ροή, κατασκευάζουμε έναν γράφο κόστους G_c από το υπολοιπόμενο γράφημα G_f. Για κάθε ακμή που ανήκει στο G_f αν αυτή η ακμή είναι πραγματική, δηλαδή ανήκει στον αρχικό γράφο (και όχι συμπληρωματική από τον υπολειπόμενο γράφο) τότε το κόστος αυτής της ακμής θα είναι $c(u, v)$ στον γράφο G_c, αλλιώς θα είναι $-c(u, v)$. Δηλαδή για κάθε ακμή που χρησιμοποιήθηκε για τον υπολογισμό της μέγιστης ροής βάζουμε το κόστος της κανονικά. Στην συνέχεια ο αλγόριθμος ξεκινάει από την τελική κορυφή t και ψάχνει για αρνητικούς κύκλους C στο γράφημα του κόστους, με την βοήθεια του αλγορίθμου Bellman-Ford. Όπου βρίσκει αρνητικό κύκλο τον αφαιρεί και αυξάνει τη ροή κατά μήκος της διαδρομής από την χωρητικότητα στένωσης (bottleneck capacity) του αρνητικού κύκλου C. Έπειτα δημιουργεί τον γράφο από την αρχή και συνεχίζει την διαδικασία. Τελικά επιστρέφει την τιμή του αθροίσματος της ροής επί του κόστους όλων των ακμών. Ο αλγόριθμος Bellman-Ford υπολογίζει τα συντομότερα μονοπάτια σε ένα γράφο ή τερματίζει αν βρει έναν αρνητικό κύκλο στο γράφο. Ο Αλγόριθμος 4 παρουσιαζει τα βήματα του Cycle Cancelling.

Algorithm 4 Min Cost Max Flow

```
Find a feasible maximum flow f in graph G using Ford Fulkerson (Max Flow)
Create a cost network Gc of the residual graph Gf
if if edge (u,v) belongs to G then
    cost cf(u,v) = c(u,v) in Gc
else
    cf(u,v) = -c(u,v) in Gc
end if
while there are negative cost cycles C, using the cost network G (Use modified Bellman Ford algorithm to find negative cycles reachable from sink t) do
    Remove the negative cost cycle by increasing the flow along each directed edge (u,v) in C by the bottleneck capacity in the cycle C.
    Goto Step 2
end while
Calculate and return the sum of the total cost cij*xij along each edge of the flow in residual graph Gf (Min cost)
```

Κεφάλαιο 5

Πειραματική Αξιολόγηση

Στο κεφάλαιο αυτό θα παρουσιαστούν οι τεχνικές λεπτομέρειες και η πειραματική αξιολόγηση του αλγορίθμου που περιγράφηκε.

5.1 Λεπτομέρειες Υλοποίησης

Υποθέτουμε ότι οι θέσεις των ΤΥΝ στο Δημόσιο Ασύρματο Μητροπολιτικό Δίκτυο ακολουθούν μια κατανομή χωρίς κλίμακα και δημιουργούμε τυχαίες τοπολογίες δικτύου ΤΥΝ χωρίς κλίμακα, χρησιμοποιώντας το Barabasi-Albert Model [32]. Παρόμοια με το [33], υποθέτουμε ότι η καθυστέρηση δικτύου μεταξύ δύο σημείων πρόσβασης στο δίκτυο είναι ανάλογη στη φυσική τους απόσταση. Οι αποστάσεις μεταξύ πραγματικών σημείων πρόσβασης είναι ουσιαστικά τυχαία μεγέθη. Αναθέτουμε την καθυστέρηση του δικτύου μεταξύ κάθε ζεύγους συνδεδεμένων ΤΥΝ τυχαία, σύμφωνα με την κανονική κατανομή $0.1 \leq \mathcal{N}(0.15, 0.05) \leq 0.2$. Έτσι τυχαιοποιούμε την καθυστέρηση στο δίκτυο διατηρώντας παράλληλα την τριγωνική ανισότητα στις αποστάσεις, διατηρώντας ότι κάθε ζεύγος κόμβων έχει μια ενδιάμεση απόσταση τουλάχιστον 0.2.
Για κάθε ΤΥΝ i, θεωρούμε τον ρυθμό εξυπηρέτησής μ_i του με δειγματοληψία από την κανονική κατανομή $\mathcal{N}(5, 2) > 0$, και ο αριθμός των εξυπηρετητών n_i με δειγματοληψία της κατανομής Poisson με μέσο όρο τρία. Ο ρυθμός άφιξης λ_i στο ΤΥΝ i καθορίζεται από την Κανονική κατανομή $0 \leq \mathcal{N}(15, 6) \leq \mu_i * n_i - 0.25$
Είναι σημαντικό να διατηρούμε το ρυθμό άφιξης κάτω από αυτό το όριο διαφορετικά όπως προκύπτει από την θεωρία αναμονής ο χρόνος αναμονής θα είναι άπειρος στην ουρά αναμονής του ΤΥΝ.

Ο προσομοιωτής της επιλογής μας ήταν το πακέτο εργαλείων του CloudSim Plus [31]. Το CloudSim Plus είναι ένα λειτουργικό, ιδιαίτερα εκτεταμένο πλαίσιο προσομοίωσης που επιτρέπει τη μοντελοποίηση, την προσομοίωση και τον πειραματισμό των αναδυόμενων υποδομών υπολογιστών υπολογιστικού νέφους, και των υπηρεσιών εφαρμογής. Επιτρέπει στους χρήστες να εστιάζουν σε συγκεκριμένα ζητήματα σχεδίασης του συστήματος που πρέπει να διερευνηθούν, χωρίς να ανησυχούν για τις λεπτομέρειες χαμηλού επιπέδου που σχετίζονται με τις

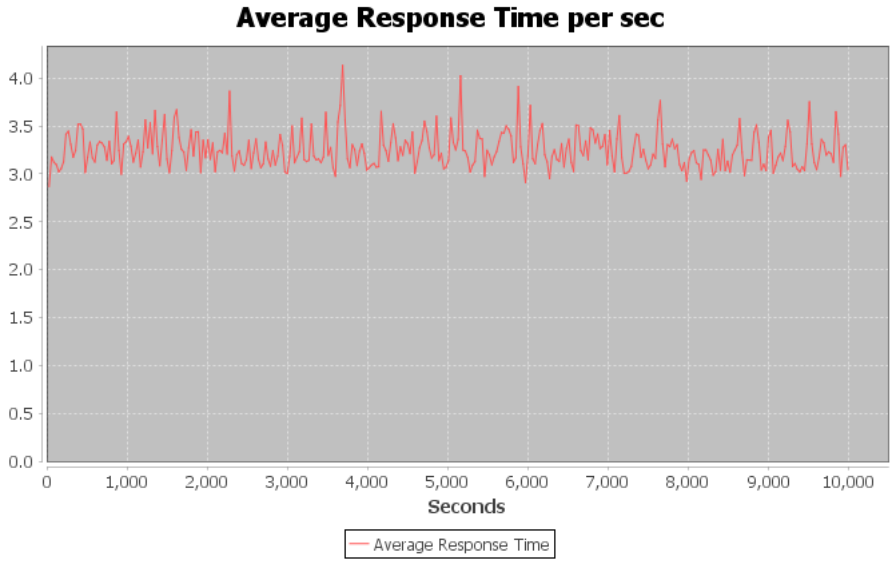
υποδομές και τις υπηρεσίες που βασίζονται στο υπολογογιστικό νέφος.
Στο CloudSim Plus όπως περιγράψαμε στο Κεφάλαιο 2, έχουμε τις οντότητες των Κέντρων Δεδομένων, τα Φυσικά Μηχανήματα και τα ΕΜ. Σε κάθε Φυσικό Μηχάνημα θεωρήσαμε ότι φιλοξενείται ένα ΕΜ. Στο CloudSim Plus στα ΕΜ παραμετροποιούνται πυρήνες υπολογιστικής επεξεργασίας, μνήμη και ο ρυθμος εξυπηρετησης εκφράζεται σε εκατομμύρια εντολών ανά δευτερόλεπτο (MIPS). Για να προσομοιώσουμε καλύτερα πραγματικά δεδομένα, σε όλη τα πειράματα θεωρήσαμε ότι ο ρυθμός εξυπηρέτησης και ο αριθμός των εξυπηρετητών ταυτίζονται ανάλογα με το MIPS και το πλήθος των πυρήνων σε ένα φυσικό μηχάνημα.
Όλα τα παρακάτω πειράματα διεξάχθηκαν σε προσωπικό υπολογιστή με επεξερταστή Intel(R) Core(TM) i5-5200U CPU @ 2.20GHz με μνήμη 8.00 GB και x64-based processor.

5.2 Πειραματική Αξιολόγηση

Στις επόμενες υποενότητες περιλαμβάνονται πειράματα που δείχνουν την λειτουργία του Αλγορίθμου με τις μετρήσεις για τον μέσο χρόνο απόκρισης, την κατανάλωση CPU, αλλά και διάφορες συγκριτικές μετρήσεις για διαφορετικές ρυθμίσεις συστήματος.

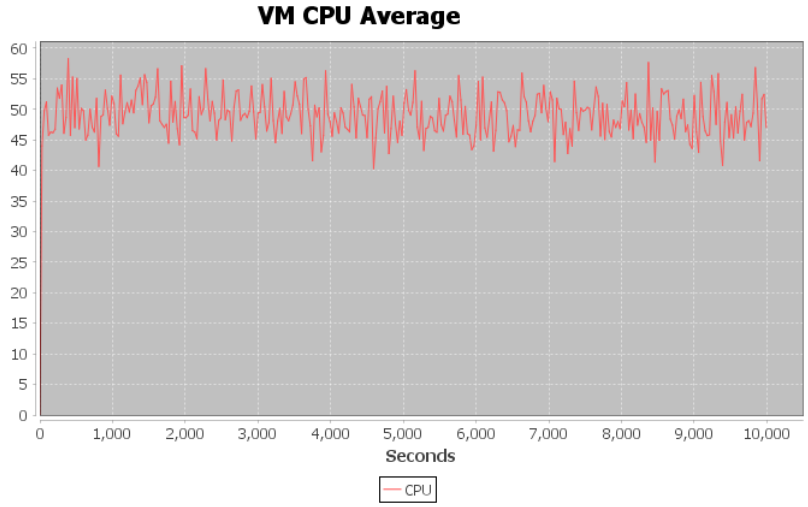
5.2.1 Πείραμα Α: Λειτουργία Αλγορίθμου

Για αρχή εκτελέσαμε τον αλγόριθμο για δέκα χιλιάδες δευτερόλεπτα αλλάζοντας το φόρτο εργασιών που φτάνει σε κάθε μηχάνημα ανά τριάντα δευτερόλεπτα, στοχεύοντας σε ελαφρύ φορτίο (περίπου 5 εργασίες ανά δευτερόλεπτο) για να δείξουμε την αποτελεσματικότητα του αλγορίθμου. Επίσης στο σύστημα υπάρχουν είκοσι ΤΥΝ. Κάθε τριάντα δευτερόλεπτα λοιπόν αλλάζουμε τα φορτία που κατά μέσο όρο φτάνουν στα ΤΥΝ και ξανατρέχουμε τον αλγόριθμο με τα νέα δεδομένα. Ο ρυθμός εξυπηρέτησης και οι εξυπηρετητές κάθε ΤΥΝ μένουν σταθερά στις αρχικές τους τιμές καθόλη τη διάρκεια του πειράματος.



Σχήμα 5.1: Μέσος χρόνος απόκρισης όλων των εργασιών σε όλα τα ΤΥΝ

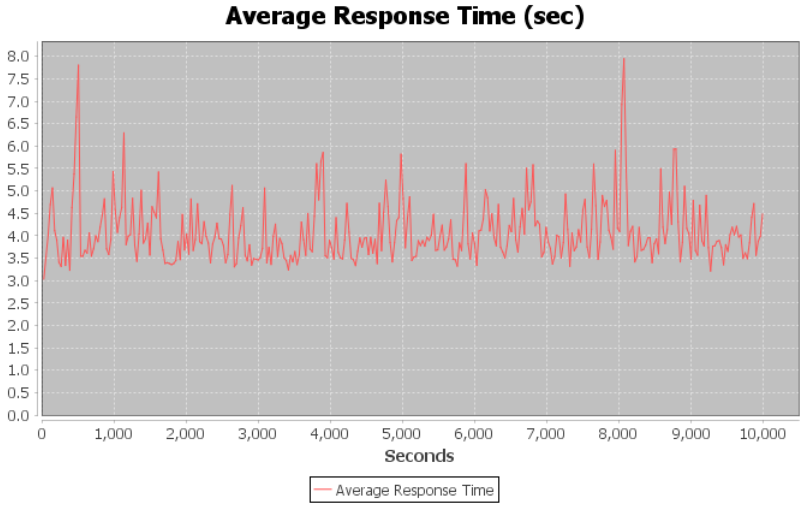
Στο Σχήμα 5.1 Βλέπουμε ότι ο μέσος χρόνος απόκρισης όλων των εργασιών σε όλα τα ΤΥΝ, παραμένει σε αναμενόμενα πλαίσια και διατηρείται σταθερός καθόλη τη διάρκεια του πειράματος. Στο Σχήμα 5.2 βλέπουμε ότι το ελαφρύ φορτίο απασχολεί περίπου τους μισούς διαθέσιμους πυρήνες επεξεργασίας στα ΕΜ, και το σύνολο των ΤΥΝ δεν δυσκολεύονται να αντεπεξέλθουν στο σύνολο του φόρτου εργασίας που δέχονται.



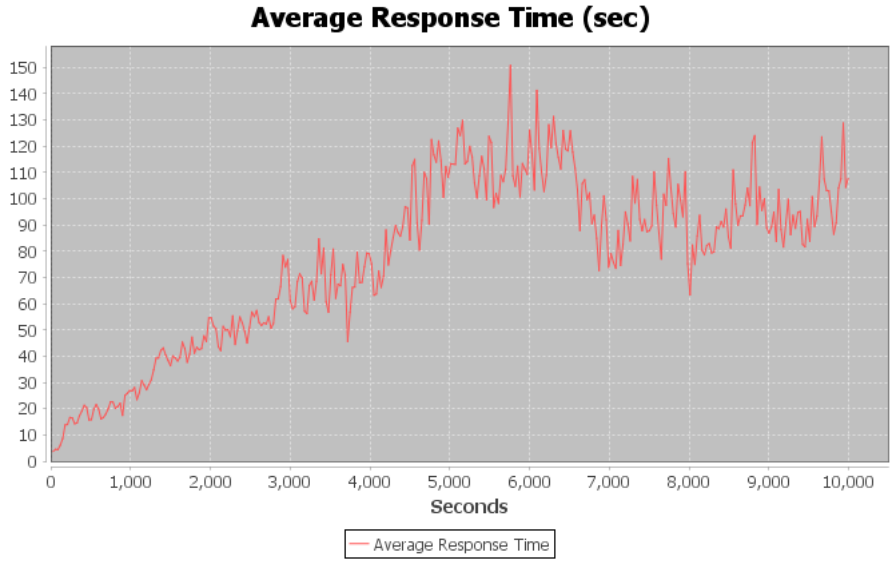
Σχήμα 5.2: Ο μέσος όρος χρήσης των επεξεργαστών στα ΤΥΝ

5.2.2 Πείραμα Β: Μετρήσεις για Ελαφρύ και Μεγάλο Φορτίο

Στο δεύτερο πείραμα τρέξαμε δύο φορές τον αλγόριθμο, στη μία θέσαμε ελαφρύ φορτίο στο σύστημα και στην άλλη μεγάλο για να δούμε την συμπεριφορά του. Και στις δύο περιπτώσεις έχουμε δέκα ΤΥΝ και παρακάτω ακολουθούν γραφικές παραστάσεις των αποτελεσμάτων για τον μέσο χρόνο απόκρισης συγκεκριμένων ΤΥΝ που είναι χαρακτηριστικά. Ο ρυθμός εξυπηρέτησης και οι εξυπηρετητές κάθε ΤΥΝ μένουν σταθερά στις αρχικές τους τιμές καθόλη τη διάρκεια του πειράματος. Κάθε τριάντα δευτερόλεπτα αλλάζουμε τις ροές σε κάθε ΤΥΝ και ξανατρέχουμε τον αλγόριθμο. Το μεγάλο φορτίο θεωρούμε ότι περιέχει περίπου δέκα εργασίες ανά δευτερόλεπτο κατά μέσο όρο για κάθε ΤΥΝ ενώ το ελαφρύ περίπου πέντε ανά δευτερόλεπτο.

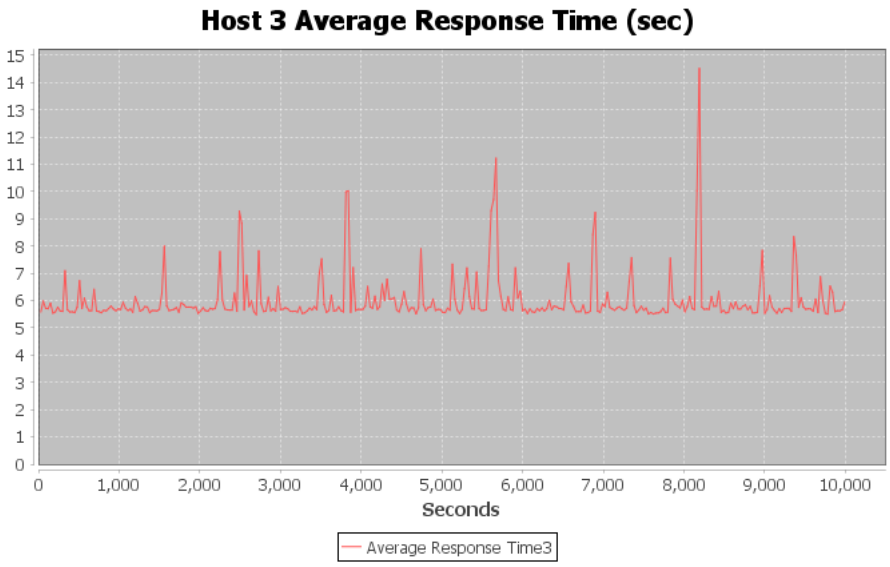


Σχήμα 5.3: Συνολικός μέσος χρόνος απόκρισης σε όλα τα ΤΥΝ στο ελαφρύ φορτίο

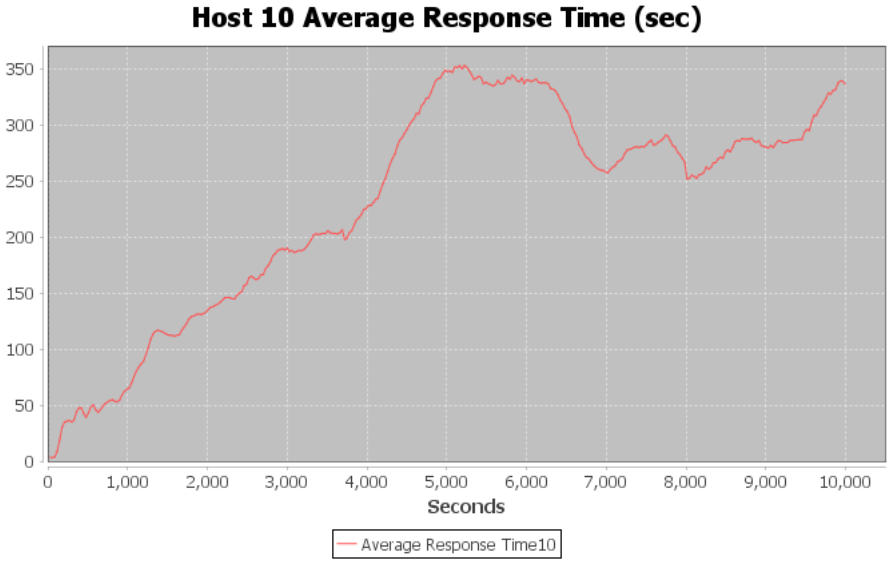


Σχήμα 5.4: Συνολικός μέσος χρόνος απόκρισης σε όλα τα ΤΥΝ στο μεγάλο φορτίο

Από τις γραφικές παραστάσεις βλέπουμε ότι σε βαρύ φορτίο ο αλγόριθμος αδυνατεί να καθορίσει ροές φορτίου εργασίας που να εξισορροπούν το μέσο χρόνο απόκρισης των εργασιών. Δηλαδή το υπερφορτωμένο ΤΥΝ 10 δεν φαίνεται να στέλνει αρκετή ροή σε άλλα υποφορτωμένα και ο μέσος χρόνος απόκρισης ανεβαίνει εκθετικά καθώς και η κατανάλωση της CPU είναι στο 100%.



Σχήμα 5.5: Μέσος χρόνος απόκρισης σε χαρακτηριστικό ΤΥΝ με ελαφρύ φορτίο

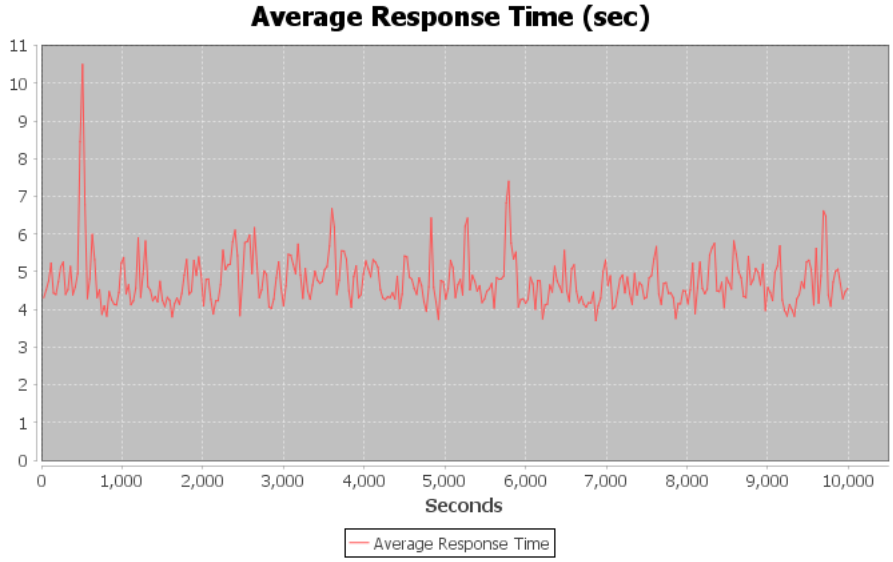


Σχήμα 5.6: Μέσος χρόνος απόκρισης στο ΤΥΝ 10 με μεγάλο φορτίο

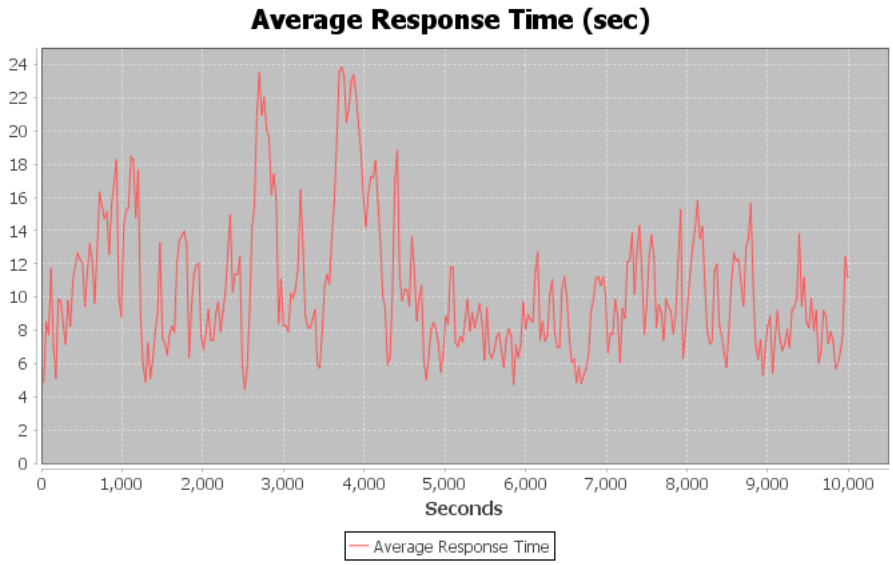
Αντιστοίχως βλέπουμε ότι ο μέσος χρόνος απόκρισης των εργασιών είναι πάρα πολύ υψηλός και μη αποδεκτός, καθώς στο ΤΥΝ 10 δεν αρκούν οι προδιαγραφές του (πυρήνες επεξεργασίας, ρυθμός εξυπηρέτησης) για να ανταπεξέλθει στο φορτίου που δέχεται.

5.2.3 Πείραμα Γ: Επίδραση Παραμέτρου ϵ

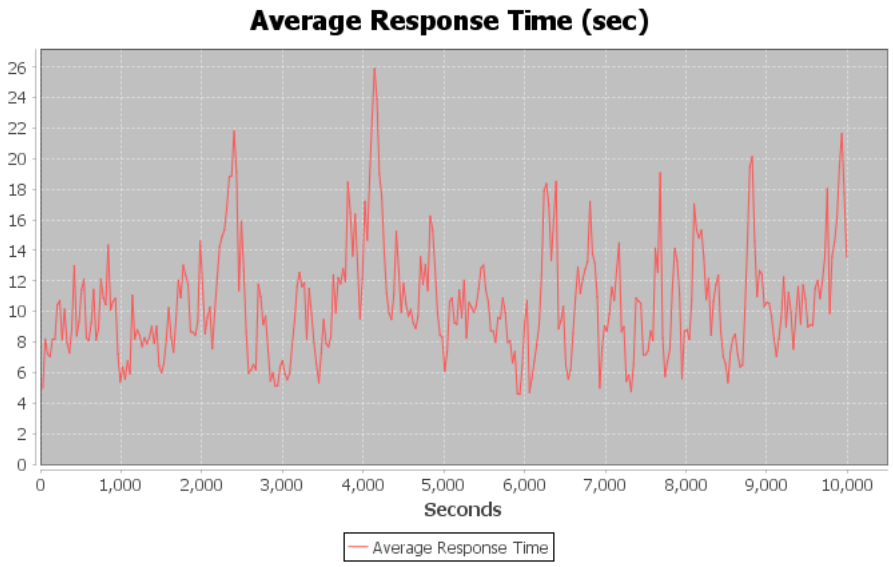
Σε αυτό το πείραμα τρέξαμε για σταθερές συνθήκες δηλαδή σταθερο ρυθμό εξυπηρέτησης, αριθμό εξυπηρετητών και ρυθμό φόρτου εργασίας σε κάθε ΤΥΝ και μετρήσαμε την επίδραση της παραμέτρου ϵ στο μέσο χρόνο απόκρισης των εργασιών. Θυμίζουμε ότι αυτή η παράμετρος χρησιμοποιείται για τον καθορισμό του αποδεκτού εύρους για την διαφορά του $\overline{D}$ με το θεωρητικό νούμερο που προκύπτει από το μοντέλο Erlang C σε κάθε ΤΥΝ.



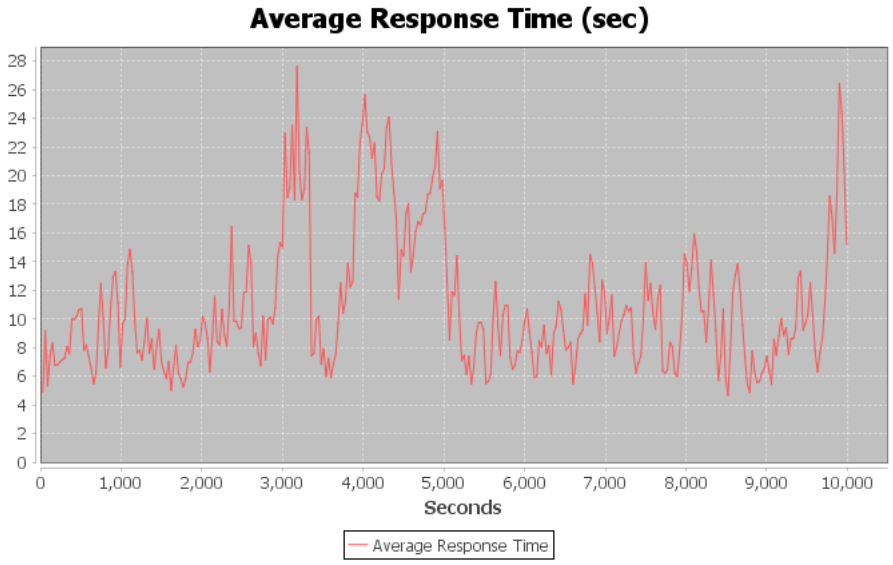
Σχήμα 5.7: Μέσος χρόνος απόκρισης για $\epsilon = 0.5$



Σχήμα 5.8: Μέσος χρόνος απόκρισης για $\epsilon = 1.0$



Σχήμα 5.9: Μέσος χρόνος απόκρισης για $\epsilon = 1.5$

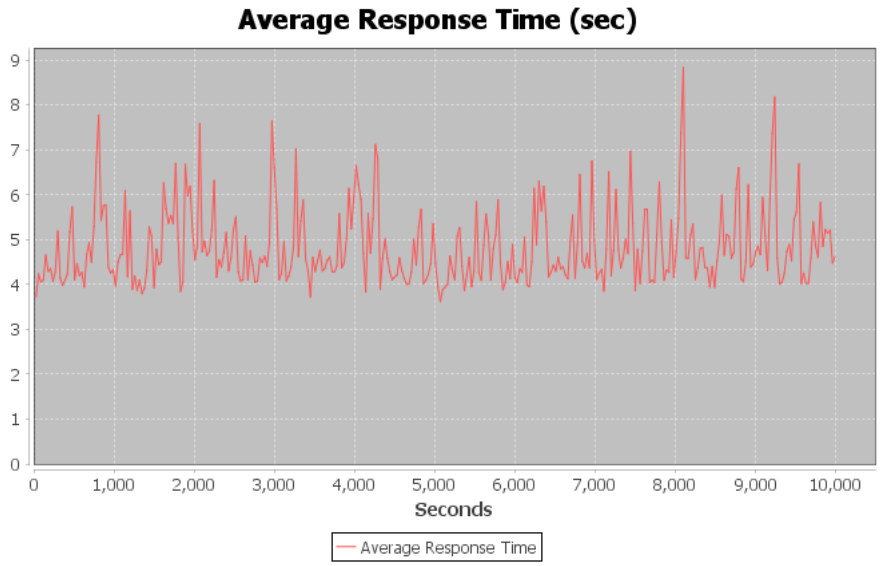


Σχήμα 5.10: Μέσος χρόνος απόκρισης για $\epsilon = 2.0$

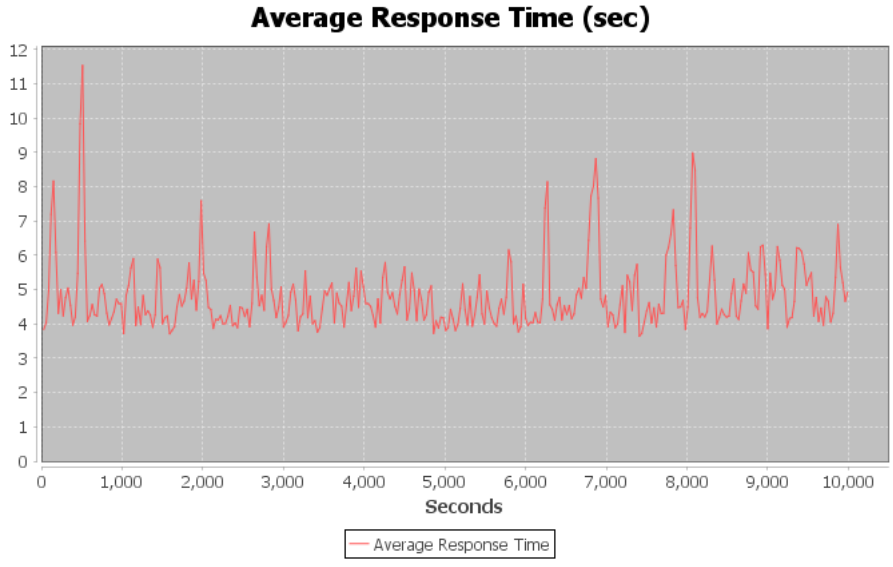
Βλέπουμε λοιπόν την φανερή επίδραση της συγκεκριμένης παραμέτρου στον Αλγόριθμο εξισορρόπησης φόρτου εργασίας. Όσο μεγαλώνει το εύρος που επιτρέπουμε για να βρούμε τις ροές, δηλαδή χαλαρώνουμε την συνθήκη που αποφασίζει ποια ροή είναι κατάλληλη για να πάει από ένα ΤΥΝ σε ένα άλλο, τότε αυξάνεται ο μέσος χρόνος απόκρισης. Για να λειτουργήσει αποδοτικά ο αλγόριθμος πρέπει το εύρος να είναι μικρό και άρα να υπάρχει καλύτερη ταξινόμηση στο ποια υπερφορτωμένα ΤΥΝ στλένουν τις εργασίες τους σε άλλα υποφορτωμένα.

5.2.4 Πείραμα Δ: Επίδραση Παραμέτρου θ

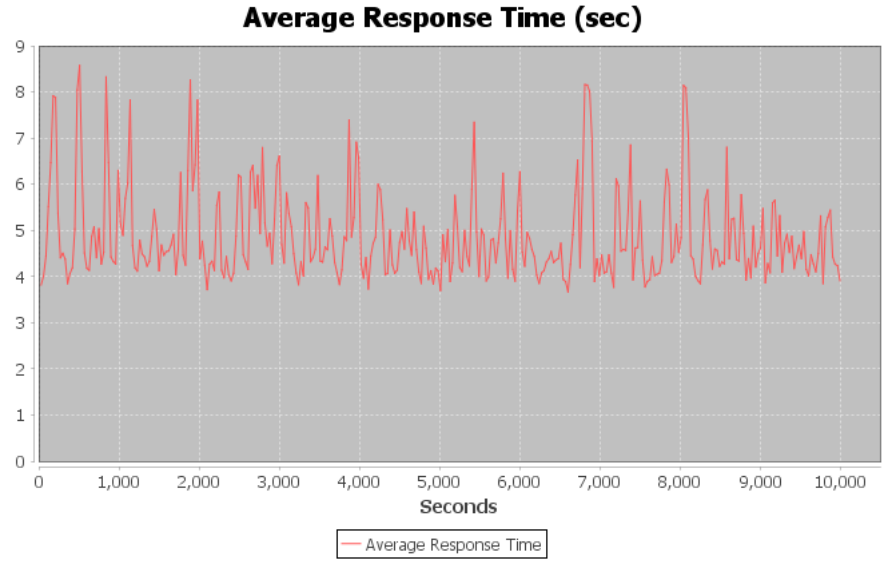
Σε αυτό το πείραμα τρέξαμε για σταθερές συνθήκες δηλαδή σταθερο ρυθμό εξυπηρέτησης, αριθμό εξυπηρετητών και ρυθμό φόρτου εργασίας σε κάθε ΤΥΝ και μετρήσαμε την επίδραση της παραμέτρου θ στο μέσο χρόνο απόκρισης των εργασιών. Θυμίζουμε ότι αυτή η παράμετρος χρησιμοποιείται για τον τερματισμό της δεύτερης βασικής ρουτίνας του αλγορίθμου ΕΦ και δείχνει το αποδεκτό εύρος μεταξύ της μέγιστης απόκρισης από τους υπερφορτωμένα ΤΥΝ και της απόκρισης που έχουμε εκτιμήσει από το πρώτο σκέλος του αλγορίθμου. Δηλαδή $|\overline{D} - \overline{D'}| \leq \theta$.



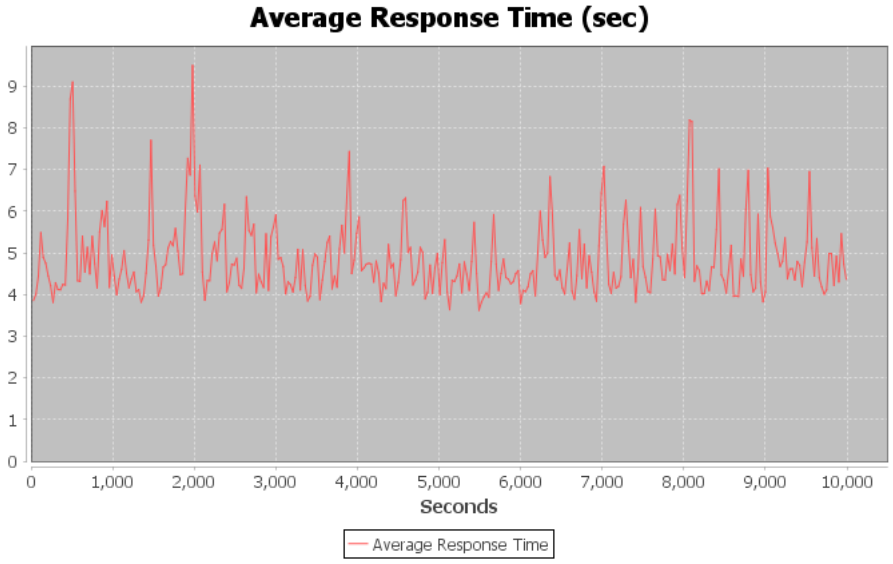
Σχήμα 5.11: Μέσος χρόνος απόκρισης για $\theta = 0.5$



Σχήμα 5.12: Μέσος χρόνος απόκρισης για $\theta = 1.0$



Σχήμα 5.13: Μέσος χρόνος απόκρισης για $\theta = 1.5$



Σχήμα 5.14: Μέσος χρόνος απόκρισης για $\theta = 2.0$

Βλέπουμε λοιπόν ότι η αύξηση της συγκεκριμένης παραμέτρου θ δεν φέρνει μεγαλύτερα νούμερα στην μέση χρονική απόκριση στον Αλγόριθμο ΕΦ. Όμως όσο μεγαλώνει το εύρος που επιτρέπουμε μεταξύ της μέγιστης απόκρισης και αυτής που έχουμε εκτιμήσει, δηλαδή χαλαρώνουμε την συνθήκη που αποφασίζει πότε μπορεί να τερματίσει ο αλγόριθμος, τότε παρατηρούμε ότι εμφανίζονται ο μέσος χρόνος απόκρισης δεν είναι τόσο σταθερός γύρω από ένα νούμερο αλλά παρουσιάζει αυξομειώσεις. Για να λειτουργήσει αποδοτικά ο αλγόριθμος πρέπει το εύρος να είναι μικρό και άρα να υπάρχει καλύτερη ταξινόμηση στο ποια υπερφορτωμένα ΤΥΝ στλένουν τις εργασίες τους σε άλλα υποφορτωμένα. Πάντως να τονίσουμε ότι η προηγούμενη παράμετρος ϵ είχε μεγαλύτερη επίδραση από την παράμετρο θ.

5.2.5 Πείραμα Ε: Σύγκριση των δύο Αλγορίθμων

Όπως είπαμε στο κεφάλαιο της υλοποίησης αντί να λύσουμε το πρόβλημα της μέγιστης ροής ελάχιστου κόστους, τρέξαμε και κάποια πειράματα με αλγορίθμους υπολογισμού της μέγιστης ροής. Οι πολυπλοκότητες αυτών των αλγορίθμων είναι πιο μικρές και άρα σε πιθανή πρακτική εφαρμογή θα έχουν σημαντική διαφορά στο χρόνο εκτέλεσης. Πιο συγκεκριμένα ο αλγόριθμος υπολογισμού της μέγιστης ροής ελάχιστου κόστους επειδή λαμβάνει υποψιν του τις ακμές του γράφου, και στην δικιά μας περίπτωση κάθε ΤΥΝ ενώνεται με κάθε άλλο στο ΔΑΜΔ ο αριθμός των ακμών στο δίκτυο είναι πάρα πολύ μεγάλος με αποτέλεσμα για μεγάλο αριθμό ΤΥΝ να αυξάνεται αρκετά ο χρόνος εκτέλεσης. Στο πείραμα Ε, τρέξαμε δύο φορές με σταθερές όλες τις παραμέτρους του συστήματος τους δύο αλγορίθμους Min Cost Max Flow, Push Relabel. Στον παρακάτω πίνακα συγκρίνουμε τα αποτελέσματα των ροών που ανακατευθύνει κάθε φορά ο αλγόριθμος και βλέπουμε ότι δεν υπάρχουν σημαντικές διαφορές. Αυτό όπως είναι λογικό, συμβαίνει επειδή η καθυστέρηση στο δίκτυο είναι μικρότερη τάξη μεγέθους από το φόρτο εργασίας, οπότε δεν είναι το βασικό κριτήριο για την επιλογή ανακατεύθυνσης της ροής. Παρόλα αυτά για πιο ακριβή απολέσματα και για διαφορετικές παραμέτρους καθυστέρησης δικτύου που θα επηρεάζουν περισσότερο το δίκτυο ο προτεινόμενος αλγόριθμος υπολογισμού της μέγιστης ροής ελάχιστου κόστους θα είναι καταλληλότερος. Στις συνθήκες που επιλέξαμε και που αντιπροσωπεύουν ένα ΔΑΜΔ, οι αλγόριθμοι μέγιστης ροής δίνουν παρόμοια αποτελέσματα. Επιλέξαμε επίσης μεγάλη ακρίβεια (δύο δεκαδικά ψηφία) στο πείραμα μας για να έχουμε πιο αναλυτικά αποτελέσματα. Δεν παραθέτουμε γραφικές παραστάσεις γιατί ο χρόνος ήταν πολύ μικρός για να βγει κάποιο ασφαλές συμπέρασμα, αλλά οι χρόνοι απόκρισης ήταν ίδιοι και στις δύο περιπτώσεις. Στον Πίνακα 5.1, βλέπουμε το φόρτο εργασίας πριν και μετά την εφαρμογή του κάθε αλγορίθμου για διάρκεια 90 δευτερόλεπτα.

Χρόνος	ΤΥΝ	Min Cost Max Flow		Push Relabel	
(sec)	#	Πριν	Μετά	Πριν	Μετα
0	1	3.8	5.95	3.8	5.95
0	2	6.8	3.05	6.8	4.41
0	3	8.2	4.67	8.2	4.62
0	4	6.2	9.98	6.2	8.57
0	5	15.6	17.05	15.6	17.05
30	1	10.05	10.05	10.05	10.05
30	2	2.1	2.1	2.1	2.1
30	3	3.1	1.49	3.1	2.1
30	4	4.5	4.5	4.5	3.89
30	5	3.9	5.51	3.9	5.51
60	1	7.8	3.01	7.8	3.65
60	2	3.6	5.38	3.6	5.39
60	3	2.8	7.97	2.8	7.97
60	4	8.1	8.74	8.1	8.73
60	5	19.1	16.3	19.1	16.3
90	1	3.01	6.49	3.01	6.49
90	2	5.38	4.62	5.38	4.61
90	3	7.97	5.25	7.97	5.25
90	4	8.74	7.79	8.74	7.8
90	5	16.3	17.25	16.3	17.22

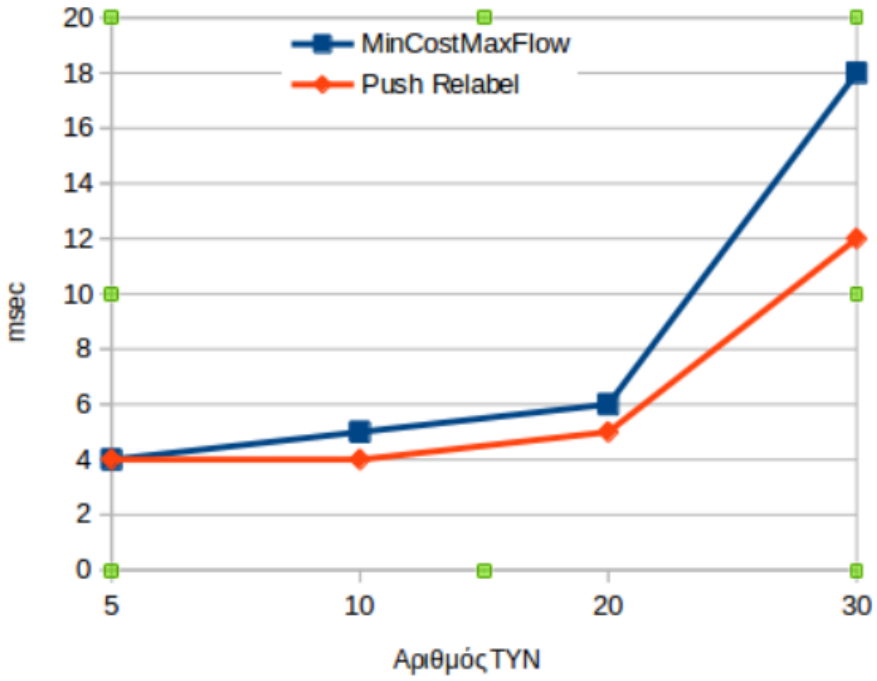
Πίνακας 5.1: Οι συγκρίσεις στις ροές μεταξύ των δύο αλγορίθμων

Οι παραπάνω μετρήσεις είχαν ως ρυθμίσεις στα ΤΥΝ:

ΤΥΝ	αριθμός εξυπηρετητών	ρυθμός εξυπηρέτησης
1	4	2.75
2	1	7.49
3	3	3.48
4	2	5.71
5	5	4.00

Πίνακας 5.2: Ρυθμίσεις ΤΥΝ

Το τελευταίο πείραμα αφορά την σύγκριση του χρόνου ταχύτητας εκτέλεσης των δύο αλγορίθμων Min Cost Max Flow , Push Relabel. Στην ουσία αποδεικνύουμε ότι η θεωρητική πολυπλοκότητα του κάθε αλγορίθμου (όπως φάνηκε και στο Κεφάλαιο 4) φαίνεται και στον χρόνο εκτέλεσης αν βάλουμε περισσότερους κόμβους (και άρα ΤΥΝ) στο δίκτυο. Εκτελέσαμε δύο φορές, μία για κάθε αλγόριθμο, με σταθερές παραμέτρους (φόρτο εργασίας σε κάθε ΤΥΝ, ρυθμό εξυπηρέτησης, αριθμό εξυπηρετητών) το συγκεκριμένο πείραμα διάρκειας 600sec στο οποίο θεωρούμε ότι ο κάθε αλγόριθμος εκτελείται ανά 30sec (άρα 20 φορές συνολικά) και υπολογίσαμε τον μέσο όρο του χρόνου εκτέλεσης της υπορουτίνας του κάθε αλγορίθμου. Έτσι φαίνεται ότι ο Push Relabel έχει μικρότερο χρόνο εκτέλεσης κατά μέσο όρο για τον αριθμό των ΤΥΝ που δοκιμάσαμε στο πείραμα. Ενώ για τον Min Cost Max Flow ανεβαίνει ο μέσος όρος του χρόνου εκτέλεσης ανάλογα με τα ΤΥΝ που έχουμε στο σύστημα. Έτσι με τα δύο τελευταία πειράματα δείξαμε ότι οι δύο αλγόριθμοι δίνουν παρόμοια αποτελέσματα αλλά ο Push Relabel χρειάζεται πολύ λιγότερο χρόνο για να εκτελεστεί ειδικά αν εφαρμοστεί σε ένα ΔΑΜΔ όπου ο αριθμός των ΤΥΝ είναι πολύ μεγάλος.



Σχήμα 5.15: Σύγκριση χρόνων εκτέλεσης των δύο αλγορίθμων Push Relabel, Min Cost Max Flow

Κεφάλαιο 6

Συμπεράσματα και Μελλοντική Εργασία

6.1 Σύνοψη συμπερασμάτων αξιολόγησης

Στα πλαίσια αυτής της διπλωματικής, υλοποιήσαμε και κάναμε πειραματική αξιολόγηση στον αλγόριθμο για την εξισορρόπηση φορτίου μεταξύ τοπικών υπολογιστικών νεφών σε ένα δημόσιο ασύρματο μητροπολιτικό δίκτυο. Ο αλγόριθμος, υπολογίζει στην αρχή ένα θεωρητικό μέσο χρόνο απόκρισης και διαχωρίζει τα ΤΥΝ ανάλογα με αυτή την τιμή σε υπερφορτωμένα και υποφορτωμένα. Μετά τον καθορισμό αυτής της τιμής, ο αλγόριθμος προχωράει στον υπολογισμό των ροών φορτίου που πρέπει να επαναδρομολογηθούν ώστε να υπάρχει ισορροπία μεταξύ των ΤΥΝ όσον αφορά τον μέσο χρόνο απόκρισης των εργασιών που αναλαμβάνουν. Για τον υπολογισμό των ζητούμενων ροών εργασιών, υλοποιήθηκαν δύο αλγόριθμοι, ένας για την Μέγιστη Ροή και ένας για την Μέγιστη Ροή Ελάχιστου Κόστους. Συνοψίζοντας τα πειραματικά αποτελέσματα, μπορούμε να πούμε ότι ο αλγόριθμος ΕΦ που χρησιμοποιήθηκε στα πλαίσια αυτής της διπλωματικής είναι αρκετά αποδοτικός και καταφέρνει να ρίξει το μέσο χρόνο απόκρισης στο σύστημα και να διατηρήσει ισορροπία μεταξύ των ΤΥΝ. Είναι εύκολα κλιμακώσιμος και χρονικά αποδοτικός και μπορεί να εφαρμοστεί σε πραγματικές συνθήκες. Το βασικό πλεονέκτημα του είναι ότι είναι μία εύκολη λύση που μπορεί να εφαρμοστεί και σε διαφορετικά μοντέλα (σε ένα δυναμικό σύστημα διαχείρισης των πόρων). Όμως, για μεγάλα φορτία ο αλγόριθμος δείχνει να μην μπορεί να τα διαχειριστεί και άρα θα πρέπει να υπάρχει μεγαλύτερη διερεύνυση γύρω από τα όρια των παραμέτρων που αντέχει να επεξεργαστεί ο συγκεκριμένος αλγόριθμος. Επίσης οι δύο αλγόριθμοι που χρησιμοποιήθηκαν έχουν παρόμοια αποτελέσματα και έτσι η εφαρμογή του Push Relabel που έχει μικρότερη πολυπλοκότητα είναι καλύτερη επιλογή. Οι δύο παράμετροι που εξετάστηκαν πρέπει να έχουν χαμηλές τιμές διαφορετικά ο μέσος χρόνος απόκρισης ανεβαίνει πολύ. Οι ουρές αναμονής που χρησιμοποιήθηκαν είναι δεσμευτικές ως προς το φορτίο εργασιών που μπορεί να αντέξει κάθε ΤΥΝ αφού υπάρχει πάντα η θεωρητική δέσμευση:

$$\lambda_i \leq \mu_i * n_i - 0.25$$

Οπουδήποτε στην πειραματική αξιολόγηση προσεγγίστηκε αυτό το όριο ο χρόνος απόκρισης των εργασιών δεν ήταν σε φυσιολογικά όρια.

6.2 Μελλοντικές Προκτάσεις

Το έργο της παρούσας διπλωματικής εργασίας μπορεί να επεκταθεί και στις εξής ακόλουθες ενδεικτικές, αλλά συγκεκριμένες, κατευθύνσεις:

- Την εφαρμογή ενός δυναμικού μοντέλου όπως εξετάστηκε στην βιβλιογραφία για τις ουρές αναμονής. Το κεντρικοποιημένο μοντέλο που χρησιμοποιήθηκε είναι αρκετά δεσμευτικό και δεν εκμεταλλεύεται τις δυνατότητες ενός κατανεμημένου μοντέλου διαχείρισης μεταξύ των ΤΥΝ του συστήματος. Έτσι πιστεύουμε ότι ακόμα και σε υπερβολικά μεγάλα φορτία θα γίνεται καλύτερη εξισορρόπηση και δεν θα έχουμε τα αποτελέσματα που είδαμε στην προηγούμενη ενότητα.
- Την προσθήκη δυναμικού αντί για στατικού μοντέλου τροφοδοσίας των πόρων. Στην λύση που προτείναμε κάθε ΤΥΝ έχει εξαρχής παραμέτρους και ρυθμίσεις που δεν αλλάζουν ανάλογα με το φορτίο που δέχονται. Ένα δυναμικό μοντέλο τροφοδοσίας θα μπορούσε να φέρει πολύ καλύτερα αποτελέσματα στον αλγόριθμο ΕΦ.
- Την πειραματική αξιολόγηση σε πραγματικό σύστημα και όχι σε πρόγραμμα προσομοίωσης για να φανούν οι πλήρεις δυνατότητες του αλγορίθμου σε ένα πραγματικό περιβάλλον και να ακολουθήσουν μετρήσεις που θα καταδείχνουν τα πλεονεκτήματα του αλγορίθμου ΕΦ.

Βιβλιογραφία

[1] Niroshinie Fernando, Seng W. Loke, Wenny Rahayu. Future Generation Computer Systems *The International Journal of eScience*, 29:84-106, 2013.

[2] A. Campbell, S. Eisenman, N. Lane, E. Miluzzo, R. Peterson, H. Lu, X. Zheng, M. Musolesi, K. Fodor, and G. Ahn. The rise of peoplecentric sensing *The International Journal of eScience*, IEEE Internet Computing, vol. 12, no. 4, pp. 12–21, 2008.

[3] M. Satyanarayanan, "Mobile computing: the next decade," *ACM Workshop on Mobile Cloud Computing Services* Social Networks and Beyond, 2010, pp. 1–6.

[4] N. Lane, E. Miluzzo, H. Lu, D. Peebles, T. Choudhury, and A. Campbell, "A survey of mobile phone sensing," *IEEE Communications Magazine*, vol. 48, no. 9, pp. 140–150, 2010.

[5] H. Cheng, F. Sun, S. Buthpitiya, and M. Griss, "Sensorchestra: Collaborative sensing for symbolic location recognition," *Mobile Computing, Applications, and Services*, pp. 195–210, 2012.

[6] A. Thiagarajan, L. Ravindranath, K. LaCurts, S. Madden, H. Balakrishnan, S. Toledo, and J. Eriksson, "Vtrack: accurate, energy-aware road traffic delay estimation using mobile phones," *in ACM Conference on Embedded Networked Sensor Systems*, 2009, pp. 85–98.

[7] M. Mun, S. Reddy, K. Shilton, N. Yau, J. Burke, D. Estrin, M. Hansen, E. Howard, R. West, and P. Boda, "Peir, the personal environmental impact report, as a platform for participatory sensing systems research," *ACM international conference on Mobile systems, applications, and services*, 2009, pp. 55–68.

[8] R. Cimler, J. Matyska, and V. Sobeslav, "Cloud based solution for mobile healthcare application," *ACM 18th International Database Engineering and Applications Symposium*, July 2014.

[9] K. Tamai and A. Shinagawa, "Platform for location-based services," *Fujitsu Sci. Tech. J*, vol. 47, no. 4, pp. 426 – 433, 2011.

[10] K. Kangas and J. Roning, "Using code mobility to create ubiquitous and active augmented reality in mobile computing," *ACM/IEEE international conference on Mobile computing and networking*, 1999, pp.48–58.

[11] J. Flinn, S. Park, M. Satyanarayanan, "Balancing performance, energy, and quality in pervasive computing", *Proceedings of the 22nd International Conference on Distributed Computing Systems*, 2002, IEEE, 2002, pp. 217–226.

[12] E.E. Marinelli, Hyrax: *cloud computing on mobile devices using MapReduce*, Masters Thesis, Carnegie Mellon University 2009.

[13] G. Huerta-Canepa, D. Lee, "A virtual cloud computing provider for mobile devices", *Proceedings of the 1st ACM Workshop on Mobile Cloud Computing and Services* Social Networks and Beyond, MCS'10, ACM, New York, NY, USA, 2010, pp. 6:1–6:5.

[14] M. Satyanarayanan, P. Bahl, R. Caceres, N. Davies, "The case for VM-based cloudlets in mobile computing", IEEE Pervasive Computing 8 (2009) 14–23.

[15] E. Walker, W. Brisken, J. Romney, "To lease or not to lease from storage clouds",Computer 43 (2010) 44–50.

[16] L. Xinhui, L. Ying, L. Tiancheng, Q. Jie, W. Fengchun, "The method and tool of cost analysis for cloud computing", *Proceedings of IEEE International Conference on Cloud Computing*, CLOUD'09, pp. 93–100.

[17] I. Constandache, X. Bao, M. Azizyan, R.R. Choudhury, "Did you see bob?: human localization using mobile phones", *Proceedings of the Sixteenth Annual International Conference on Mobile Computing and Networking*, MobiCom'10, ACM, New York, NY, USA, 2010, pp. 149–160.

[18] A. Madhavapeddy, A. Tse, "A study of bluetooth propagation using accurate indoor location mapping", *M. Beigl, S. Intille, J. Rekimoto, H. Tokuda (Eds.), UbiComp 2005: Ubiquitous Computing*, in: Lecture Notes in Computer Science, vol. 3660, Springer, Berlin, Heidelberg, 2005, pp. 105–122.

[19] J.H. Ye, J. Herbert, "Interface tailoring for mobile computing devices", in: C. Stary, C. Stephanidis (Eds.), User-Centered Interaction Paradigms for Universal Access in the Information Society, *Lecture Notes in Computer Science*, vol. 3196, Springer, Berlin, Heidelberg, 2004, pp. 175–182.

[20] J. Landay, T. Kaufmann, "User interface issues in mobile computing", *Proceedings of the Fourth Workshop on Workstation Operating Systems*, IEEE, 1993, pp. 40–47.

[21] F. Samimi, P. McKinley, S. Sadjadi, "Mobile service clouds: a self-managing infrastructure for autonomic mobile computing services", in: A. Keller, J.P. Martin-Flatin (Eds.), "Self-Managed Networks, Systems, and Services", *Lecture Notes in Computer Science*, vol. 3996, Springer, Berlin, Heidelberg, 2006, pp. 130–141.

[22] M. Shiels, "Phone sales hit by sidekick loss", 2009.

[23] H.J. La, S.D. Kim, "A conceptual framework for provisioning context-aware mobile cloud services", *Proceedings of the IEEE 3rd International Conference on Cloud Computing*, CLOUD, pp. 466–473.

[24] H.T. Dinh, C. Lee, D. Niyato, P. Wang, "A survey of mobile cloud computing: architecture, applications, and approaches", Wireless Communications and Mobile Computing (2011).

[25] Mike Jia, Weifa Liang, Zichuan Xu, and Meitian Huang "Research School of Computer Science" *The Australian National University Canberra*, ACT 2601, Australia.

[26] M. Jia, J. Cao, and W. Liang. "Optimal cloudlet placement and user to cloudlet allocation in wireless metropolitan area networks". *IEEE Transactions on Cloud Computing*, April, 2015.

[27] Y. Jiang, Y. Zhou, and W. Wang, "Task allocation for undependable multiagent systems in social networks," *IEEE Trans. Parallel Distributed. Systems*, vol. 24, no. 8, pp. 1671–1681, Aug. 2013.

[28] Y. Jiang and J. Jiang, "Contextual resource negotiation-based task allocation and load balancing in complex software systems," *IEEE Trans. Parallel Distributed Systems*, vol. 20, no. 5, pp. 641–653, May 2009.

[29] M.M. Weerdt, Y. Zhang, and T. Klos, "Multiagent task allocation in social networks," *Auton. Agents Multi-Agent Systems*, vol. 25, no. 1, pp. 46–86, 2012.

[30] L.R. Ford and D.R. Fulkerson. "A simple algorithm for finding maximal network flows and an application to the Hitchcock problem". Citeseer,1955.

[31] Manoel C. Silva Filhoy, Raysa L. Oliveiray, ClaudioC. Monteiro, Pedro R. M. Inácioy, Mário M. Freirey "CloudSim Plus: A Cloud Computing Simulation Framework Pursuing Software Engineering Principles for Improved Modularity, Extensibility and Correctness", 978-3-901882-89-0 @2017 IFIP.

[32] R. Albert, H. Jeong, and A. Barabási. "Internet: Diameter of the worldwide web" . *Nature, Vol. 401*, pp. 130–131, 1999.

[33] M. Jia, J. Cao, and W. Liang. "Optimal cloudlet placement and user to cloudlet allocation in wireless metropolitan area networks". *To appear in IEEE Transactions on Cloud Computing*, April, 2015.

[34] Dhinesh Babu, L.D. Venkata Krishna,"Honey bee behavior inspired load balancing of tasks in cloud computing environments" *Appl.SoftComput* 2013 , 13(5), 2292–2303.

[35] Yiqiu Fang, Fei Wang, Junwei Ge: "A Task Scheduling Algorithm Based on Load Balancing in Cloud Computing". WISM 2010: 271-277.

[36] Siva Theja Maguluri, R. Srikant, Lei Ying: "Stochastic models of load balancing and scheduling in cloud computing clusters". INFOCOM 2012: 702-710.

[37] Fahimeh Ramezani, Jie Lu, Farookh Khadeer Hussain: "Task-Based System Load Balancing in Cloud Computing Using Particle Swarm Optimization. International Journal of Parallel Programming" 42(5): 739-754 (2014).

[38] Shafii Muhammad Abdulhamid, Muhammad Shafie Abd Latiff, Ismaila Idris: "Tasks Scheduling Technique Using League Championship Algorithm for Makespan Minimization" in IaaS Cloud. CoRR abs/1510.03173 (2015).

[39] William Voorsluys, James Broberg, Srikumar Venugopal, Rajkumar Buyya: "Cost of Virtual Machine Live Migration in Clouds: A Performance Evaluation". CloudCom 2009: 254-265.

[40] Mohammed Abdullahi, Md. Asri Ngadi, Shafii Muhammad Abdulhamid: "Symbiotic Organism Search optimization based task scheduling in cloud computing environment". Future Generation Comp. Syst. 56: 640-650 (2016).

[41] Zhenhua Wang, Haopeng Chen, Ying Fu, Delin Liu, Yunmeng Ban: "Workload balancing and adaptive resource management for the swift storage system on cloud". Future Generation Comp. Syst. 51: 120-131 (2015).

[42] K. R. Remesh Babu, Philip Samuel: "Enhanced Bee Colony Algorithm for Efficient Load Balancing and Scheduling in Cloud". IBICA 2015: 67-78.

[43] J. Octavio Gutiérrez-García, Adrian Ramirez Nafarrate: "Agent-based load balancing in Cloud data centers". Cluster Computing 18(3): 1041-1062 (2015).

[44] Sourav Banerjee, Mainak Adhikari ,Sukhendu Kar, Utpal Biswas, "Development and Analysis of a New Cloudlet Allocation Strategy for QoS Improvement in Cloud",Arabian Journal for Science and Engineering, May 2015, Volume 40, Issue 5, pp 1409–142.

[45] Nader Mohamed, Jameela Al-Jaroodi, Abdulla Eid: "A dual-direction technique for fast file downloads with dynamic load balancing in the Cloud." J. Network and Computer Applications 36(4): 1116-1130 (2013).

[46] Alan Massaru Nakai, Edmundo Roberto Mauro Madeira, Luiz Eduardo Buzato: "On the Use of Resource Reservation for Web Services Load Balancing". J. Network Syst. Manage. 23(3): 502-538 (2015).

[47] Kuan-Chou Lai, You-Fu Yu: "A scalable multi-attribute hybrid overlay for range queries on the cloud". Information Systems Frontiers 14(4): 895-908 (2012).

[48] Keng-Mao Cho, Pang-Wei Tsai, Chun-Wei Tsai, Chu-Sing Yang: "A hybrid meta-heuristic algorithm for VM scheduling with load balancing in cloud computing". Neural Computing and Applications 26(6): 1297-1309 (2015).

[49] Mohammad Norouzi Arab, Mohsen Sharifi: "A model for communication between resource discovery and load balancing units in computing environments". The Journal of Supercomputing 68(3): 1538-1555 (2014).

[50] Yi Lu, Qiaomin Xie, Gabriel Kliot, Alan Geller, James R. Larus, Albert G. Greenberg: "Join-Idle-Queue: A novel load balancing algorithm for dynamically scalable web services." Perform. Eval. 68(11): 1056-1071 (2011).

[51] Shang-Liang Chen, Yun-Yao Chen, Suang-Hong Kuo: "CLB: A novel load balancing architecture and algorithm for cloud services". Computers and Electrical Engineering 58: 154-160 (2017).

[52] Leontiou, N., Dechouniotis, D., Athanasopoulos, N. and Denazis, S., 2014, June. On load balancing and resource allocation in cloud services. In Control and Automation (MED), 2014 22nd Mediterranean Conference of (pp. 773-778). IEEE.

[53] Dechouniotis, D., Leontiou, N., Athanasopoulos, N., Christakidis, A. and Denazis, S., 2015. A control-theoretic approach towards joint admission control and resource allocation of cloud computing services. International Journal of Network Management, 25(3), pp.159-180.