ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΤΟΜΕΑΣ ΕΠΙΚΟΙΝΩΝΙΩΝ, ΗΛΕΚΤΡΟΝΙΚΗΣ & ΣΥΣΤΗΜΑΤΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ

Αλγόριθμοι Αντιστοίχισης Εικονικών Τοπολογιών Δικτύου με Ανθεκτικότητα σε Σφάλματα

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

της

ΣΤΑΘΑΚΟΠΟΥΛΟΥ ΧΡΥΣΟΥΛΑΣ

Επιβλέπων : Συμεών Παπαβασιλείου
Αν. Καθηγητής Ε.Μ.Π.

Αθήνα, Ιούλιος 2014

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΕΠΙΚΟΙΝΩΝΙΩΝ, ΗΛΕΚΤΡΟΝΙΚΗΣ
& ΣΥΣΤΗΜΑΤΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ

Αλγόριθμοι Αντιστοίχισης Εικονικών Τοπολογιών Δικτύου με Ανθεκτικότητα σε Σφάλματα

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

της

ΣΤΑΘΑΚΟΠΟΥΛΟΥ ΧΡΥΣΟΥΛΑΣ

Επιβλέπων : Συμεών Παπαβασιλείου
Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την

.....	.....	.....
Συμεών Παπαβασιλείου	Κακλαμάνη Δήμητρα-Θεοδώρα	Ρουσσάκη Ιωάννα
Καθηγητής Ε.Μ.Π.	Καθηγήτρια Ε.Μ.Π.	Λέκτορας Ε.Μ.Π.

Αθήνα, Ιούλιος 2014

.....

ΣΤΑΘΑΚΟΠΟΥΛΟΥ ΧΡΥΣΟΥΛΑ

Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Copyright © Χρυσούλα Σταθακοπούλου, Ιούλιος 2014
Με επιφύλαξη παντός δικαιώματος. All rights reserved

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 τον συγγραφέα και δεν πρέπει να ερμηνευθεί ότι αντιπροσωπεύουν τις επίσημες θέσεις του Εθνικού Μετσόβιου Πολυτεχνείου.

Περίληψη

Το διαδίκτυο με τη σημερινή του μορφή επιβάλει περιορισμούς στην ανάπτυξη νέων τεχνολογιών. Λύση σε αυτό το πρόβλημα έρχονται να προσφέρουν τα εικονικά δίκτυα διαχωρίζοντας το ρόλο του Παρόχου Υπηρεσιών από αυτόν του Παρόχου Υποδομής, προσφέροντας έτσι τη δυνατότητα συνύπαρξης ετερογενών εικονικών δικτύων.

Ένα από τα σημαντικότερα ζητήματα στην δημιουργία εικονικών δικτύων είναι η δέσμευση βέλτιστη δέσμευση πόρων για την ενσωμάτωσή του στο δικτυακό υπόστρωμα, γνωστή και ως το πρόβλημα Ενσωμάτωσης Εικονικού Δικτύου (ΕΕΔ). Για να είναι ωστόσο η εικονικοποίηση δικτύων πρακτικά εφαρμόσιμη πρέπει να έχει χαρακτηριστικά ανθεκτικότητας σε σφάλματα.

Σε αυτό το πλαίσιο, η εργασία αυτή επεκτείνοντας λύσεις που υπάρχουν στη βιβλιογραφία για ΕΕΔ και ΕΕΔ με ανοχή σε σφάλματα προτείνει και συγκρίνει την επίδοση αλγορίθμων που παρουσιάζουν ανεκτικότητα σε μεμονωμένα ή πολλαπλά σφάλματα σε επιθυμητό σύνολο κόμβων και στις προσκείμενες σε αυτούς ζεύξης.

Λέξεις Κλειδιά: Εικονικά Δίκτυα, Εικονικοποίηση Δικτύων, Ανοχή σε Σφάλματα, Ενσωμάτωση Πόρων

Abstract

Internet with its current structure imposes limitations on the development of innovative technologies. Network virtualization decouples the roles of Internet Service Provider and Infrastructure Provider allowing the coexistence of several heterogeneous network topologies.

One of the most important challenges of network virtualization, though, is the optimal resources allocation for the embedding of the virtual network, a problem known as Virtual Network Embedding (VNE). Furthermore, in order for Network Virtualization to be applicable, Virtual Networks must be able to survive link and node failures, problem known as Survival Virtual Network Embedding (SVNE).

In this context, this thesis extends existing solutions for both VNE and SVNE to propose and compare algorithms that provide survivability over either single or multiple failures to a set of critical nodes and adjacent links to these nodes.

Keywords: Survival Virtual Network Embedding, Survivability, Network Virtualization, Resource Allocation

Πρόλογος

Η παρούσα διπλωματική εργασία εκπονήθηκε στον τομέα Επικοινωνιών, Ηλεκτρονικής και Συστημάτων πληροφορικής στη διάρκεια του ακαδημαϊκού έτους 2013-2014 και επισφραγίζει τις σπουδές μου στο Εθνικό Μετσόβιο Πολυτεχνείο.

Θα ήθελα ιδιαιτέρως να ευχαριστήσω τον επιβλέποντα καθηγητή μου κ. Παπαβασιλείου Συμεών για την καθοδήγησή του και για την ευκαιρία που μου έδωσε να ασχοληθώ με ένα τόσο ενδιαφέρον αντικείμενο. Επίσης θα ήθελα να ευχαριστήσω την Παπαγιάννη Χρύσα, Senior Researcher στο Εργαστήριο Σχεδίασης και Βέλτιστου Σχεδιασμού Δικτύου για τον χρόνο που μου αφιέρωσε και την άψογη συνεργασία. Ακόμα, θα ήθελα να ευχαριστήσω τον υποψήφιο Διδάκτορα Λειβαδέα Άρη για την προθυμία του να με καθοδηγήσει όποτε αυτό χρειάστηκε.

Θα ήθελα τέλος να ευχαριστήσω την οικογένεια και τους φίλους μου για την υπομονή και κατανόηση τους καθ' όλη τη διάρκεια των σπουδών μου και ιδιαίτερα τους τελευταίους μήνες

Πίνακας περιεχομένων

Σχήματα

Πίνακες

1

Εισαγωγή

1.1 Εικονικά Δίκτυα και προβλήματα ανάθεσης πόρων

Το διαδίκτυο με τη σημερινή του μορφή, όπως έχει διαμορφωθεί εδώ και περισσότερο από τρεις δεκαετίες, έχει καθιερωθεί και έχει αλλάξει τον τρόπο που δουλεύουμε, μαθαίνουμε, διασκεδάζουμε, καθώς και πολλές ακόμα πτυχές τις ζωής μας. Ωστόσο, οι συνθήκες κάτω από τις οποίες αρχικά διαμορφώθηκε και οι ανάγκες που καλούταν να καλύψει έχουν αλλάξει και παρουσιάζει ακαμψία στην υποστήριξη των νέων τεχνολογιών, πρωτοκόλλων κλπ. Καθολική υιοθέτηση οποιασδήποτε αλλαγής στην αρχιτεκτονική του Διαδικτύου θα απαιτούσε τη συγκατάθεση όλων των Παρόχων Υπηρεσιών Διαδικτύου (Internet Service Provider - ISP), κάτι που δεν είναι εφικτό και ευέλικτο.

Το πρόβλημα αυτό έρχεται να λύσει η Εικονικοποίηση Δικτύων (Network Virtualization). Η Εικονικοποίηση Δικτύου διαχωρίζει το ρόλο του Παρόχου Υπηρεσιών Διαδικτύου στον Πάροχο Υποδομής (Infrastructure Provider – InP), που είναι υπεύθυνος για τη διαχείριση της φυσικής υποδομής, και στον Πάροχο Υπηρεσιών (Service Provider – SP) που δημιουργεί τα εικονικά δίκτυα (Virtual Networks) και παρέχει απ' άκρη σ' άκρη υπηρεσίες (Chowdhury, 2009a) σε αυτά. Το περιβάλλον αυτό επιτρέπει την συνύπαρξη ετερογενών αρχιτεκτονικών δικτύων παρακάμπτοντας του περιορισμούς που παρουσιάζονται στο Διαδίκτυο σήμερα.

Στα πλαίσια της εικονικοποίησης ο ρόλος του Παρόχου Υπηρεσιών μπορεί να διαιρεθεί περεταίρω σε τρεις νέους ρόλους όπως αναλύεται στο (Schaffrath, 2009):

- Στον ***Πάροχο Εικονικού Δικτύου*** (Virtual Network Provider – VPN), ο οποίος είναι υπεύθυνος να συγκεντρώνει πόρους από πολλούς Παρόχους Υποδομής σε μία εικονική τοπολογία.
- Στον ***Διαχειριστή Εικονικού Δικτύου*** (Virtual Network Operator - VNO), ο οποίος είναι υπεύθυνος για την εγκατάσταση και λειτουργία του εικονικού δικτύου πάνω από την τοπολογία που παρέχει ο Πάροχος Εικονικού Δικτύου.
- Στον ***Πάροχο Υπηρεσίας*** (Service Provider – SP) οποίος χρησιμοποιεί την εικονική τοπολογία για να παρέχει μια υπηρεσία, είτε πρόκειται για υπηρεσία εφαρμογής είτε για υπηρεσία δικτύου παρέχοντας το εικονικό δίκτυο εκ νέου ως δικτυακό υπόστρωμα.

Η διαδικασία δημιουργίας ενός Εικονικού Δικτύου – ΕΔ (Virtual Network – VN) ξεκινά αφού έχει πραγματοποιηθεί η εικονικοποίηση των πόρων του δικτυακού υποστρώματος από τον Πάροχο Εικονικού Δικτύου, δηλαδή η δημιουργία ενός αφαιρετικού επιπέδου όπου διατίθενται συνολικά οι πόροι από τα διάφορα δικτυακά υποστρώματα. Οι πόροι του εικονικοποιημένου υποστρώματος υπόκεινται στη συνέχεια σε τρία συνυφασμένα μεταξύ τους βήματα (Niebert, 2008):

- ***Ανακάλυψη Πόρων***: Συλλέγονται πληροφορίες για τους διαθέσιμους εικονικούς πόρους.
- ***Περιγραφή Πόρων***: Για να βοηθηθεί η ανακάλυψη πόρων θα πρέπει να ο Πάροχος Υπηρεσίας να περιγράψει στον Διαχειριστή Εικονικού Δικτύου τους πόρους που απαιτούνται. Η περιγραφή θα πρέπει να περιλαμβάνει και τις ιδιότητες και τους περιορισμούς που θα πρέπει οι πόροι να πληρούν.
- ***Δέσμευση Πόρων***: Πρόκειται για το βασικό βήμα για τη δημιουργία του νέου εικονικού δικτύου. Σύμφωνα με τις απαιτήσεις ως προς το κόστος και τις ανάγκες της υπηρεσίας που θα φιλοξενήσει το δίκτυο, κατάλληλοι πόροι θα πρέπει να βρεθούν και να εκχωρηθούν το εικονικό δίκτυο.

Η αντιστοίχιση των πόρων που θέλει να δεσμεύσει το εικονικό δίκτυο σε πόρους του δικτυακού υποστρώματος αποτελεί τη βασική πρόκληση στη δέσμευση πόρων για την εικονικοποίηση δικτύων (Haider, 2009) και αναφέρεται στη βιβλιογραφία ως Ενσωμάτωση Εικονικού Δικτύου - ΕΕΔ (Virtual Network Embedding – VNE). Η αντιστοίχιση αυτή θα πρέπει να μπορεί να ικανοποιήσει τις απαιτήσεις σε πόρους του εικονικού δικτύου, κάτω από τους περιορισμούς που τίθενται από τους διαθέσιμους πόρους του εικονικοποιημένου υποστρώματος και ταυτόχρονα να γίνεται με τρόπο που να αξιοποιεί βέλτιστα τους παρεχόμενους πόρους βάσει των κριτηρίων που θέτουν οι οντότητες που συμμετέχουν στην εικονικοποίηση, όπως περιγράφηκαν παραπάνω.

Προκειμένου ωστόσο η εικονικοποίηση δικτύων να είναι πρακτικά εφαρμόσιμη πρέπει να ληφθεί υπόψιν ο παράγοντας των σφαλμάτων στα διάφορα επίπεδα ενός εικονικού περιβάλλοντος, ώστε να εξασφαλίζεται η σταθερότητα και αξιοπιστία των παρεχόμενων υπηρεσιών στον τελικό χρήστη (π.χ., τον πάροχο υπηρεσίας). Συγκεκριμένα η ΕΕΔ θα πρέπει να παρουσιάζει ανοχή σε σφάλματα τόσο στην υποδομή όσο και στο ίδιο το Εικονικό Δίκτυο, είτε αυτά αφορούν προκύπτουν σε κόμβους είτε σε ζεύξεις του δικτύου.

1.2 Ενσωμάτωση εικονικών δικτύων με ανοχή σε σφάλματα

Στόχος αυτής της εργασίας είναι να προτείνει αλγορίθμους Ενσωμάτωσης Εικονικών Δικτύων με ανοχή σε σφάλματα. Συγκεκριμένα θέλουμε να εισάγουμε τη δυνατότητα ανάκαμψης από σφάλματα τόσο σε ζεύξεις όσο και σε κόμβους του δικτύου υποστρώματος, το οποίο αποτελείται από ετερογενείς κόμβους (διακομιστές, εξυπηρετητές), κατά τη διαδικασία ενσωμάτωσης. Η ΕΕΔ θα λαμβάνει επιπλέον υπόψιν της τη μεγιστοποίηση του κέρδους του παρόχου υποδομής, ελαχιστοποιώντας το κόστος του εικονικού αιτήματος. Ταυτόχρονα, θέλουμε να ελαχιστοποιήσουμε το μήκος μονοπατιού των ζεύξεων του δικτυακού υποστρώματος στις οποίες αντιστοιχίζεται μία εικονική ζεύξη, περιορίζοντας έτσι την καθυστέρηση στην μετάδοση της πληροφορίας.

Η λύση που προτείνουμε στο παραπάνω πρόβλημα μοντελοποιείται σε ένα πρόβλημα Μεικτού Ακέραιου Προγραμματισμού – ΜΑΠ (Mixed Linear Integer Programming – MILP) για τη λύση του οποίου υιοθετούνται κατάλληλοι ευριστικοί αλγόριθμοι από τη βιβλιογραφία.

1.2.1 Συνεισφορά

Η εργασία αυτή, επεκτείνοντας λύσεις που έχουν ήδη προταθεί για το πρόβλημα της ΕΕΔ και της ΕΔΔ με ανοχή σε σφάλματα, προτείνει τέσσερις αλγορίθμους ΕΕΔ με ανοχή σε σφάλματα σε συνδυασμό με δύο πολιτικές δέσμευσης εφεδρικών πόρων. Οι αλγόριθμοι που προτείνονται αφορούν στην ενσωμάτωση εικονικών τοπολογιών ετερογενών κόμβων. Εξασφαλίζουν την ανοχή σε σφάλματα σε επιθυμητό σύνολο κρίσιμων κόμβων ενώ οι δύο από τους τέσσερις, επεκτείνουν τους άλλους δύο για την εξασφάλιση ανοχής σε σφάλματα στις προσκείμενες στους κόμβους ζεύξεις.

Πιο συγκεκριμένα, για να βελτιώσουμε την υπολογιστική πολυπλοκότητα που λόγω του ΜΑΠ είναι εκθετική διαχωρίζουμε τη φάση ενσωμάτωσης κόμβων από τη φάση ενσωμάτωσης ζεύξεων σε δύο ξεχωριστά προβλήματα ΓΠ, όπως στο (Chowdhury, 2009b). Οι αλγόριθμοι που προτείνονται αρχικά επαυξάνουν το εικονικό αίτημα σε ένα αξιόπιστο εικονικό αίτημα με εφεδρικούς κόμβους και ζεύξεις και στη συνέχεια το ενσωματώνουν στο δικτυακό υπόστρωμα, όπως προτείνεται στο (Yu, 2013). Ο πρώτος αλγόριθμος που προτείνεται αντιστοιχίζει ταυτόχρονα τους εικονικούς και εφεδρικούς πόρους σε κάθε φάση.

Ο δεύτερος ενσωματώνει πρώτα το αρχικό εικονικού αίτημα και στη συνέχεια τους εφεδρικούς πόρους του. Τέλος, η ανοχή σε σφάλματα σε ζεύξεις επιτυγχάνεται με την τεχνική μετακίνησης κόμβων (migration) εξασφαλίζοντας ασύνδετα μονοπάτια για την κύρια και εφεδρική ροή κίνησης που έχει ήδη εξασφαλιστεί για να εξασφαλιστεί ανοχή σε σφάλματα σε κόμβους.

1.3 Οργάνωση κειμένου

Το υπόλοιπο κείμενο διαρθρώνεται ως εξής. Στο κεφάλαιο 2 παρουσιάζεται εκτενέστερα το πρόβλημα της ΕΕΔ και στη συνέχεια το πρόβλημα της ΕΕΔ με ανοχή σε σφάλματα. Παράλληλα, παρουσιάζονται και συγκρίνονται οι σχετικές λύσεις που έχουν ήδη προταθεί στη βιβλιογραφία. Στο κεφάλαιο 3 παρουσιάζεται α. ο ευριστικός αλγόριθμος που έχει υιοθετηθεί για την επίλυση του προβλήματος, β. η προτεινόμενη μοντελοποίηση του, ως πρόβλημα Μεικτού Ακεραίου Γραμμικού Προγραμματισμού, αναλύοντας τις παραμέτρους που θέλουμε να βελτιστοποιήσουμε και τους περιορισμούς κάτω από τους οποίους πρέπει να λυθεί γ. οι προτεινόμενες διαφοροποιήσεις της μοντελοποίησης με στόχο την αντιμετώπιση διαφορετικών επιπρόσθετων στόχων (πχ, πλεονασμός-κατά-1/1-redundancy και πλεονασμός - κατά-k/k-redundancy κλπ). Στο κεφάλαιο 4 παρουσιάζουμε και αξιολογούμε αποτελέσματα από προσομοιώσεις για τους αλγορίθμους που προτάθηκαν και αναλύονται το περιβάλλον, οι παράμετροι και οι δείκτες που χρησιμοποιήθηκαν για τα πειράματα. Στο κεφάλαιο 5 παρατίθεται και επεξηγείται συνοπτικά ο κώδικας σε Java για την κωδικοποίηση των αλγορίθμων και παρατίθενται στοιχεία για το υλικό και λογισμικό στο οποίο έγιναν τα πειράματα. Τέλος, στο κεφάλαιο 6 συνοψίζουμε τα συμπεράσματα που προέκυψαν από την παραπάνω μελέτη και αναφερόμαστε στις προκλήσεις που μένουν ανοιχτές πάνω στο πρόβλημα της ΕΕΔ με ανοχή σε σφάλματα.

2

Μέθοδοι Ενσωμάτωσης Εικονικών Δικτύων με Ανοχή σε Σφάλματα

2.1 Εισαγωγή

2.1.1 Το πρόβλημα της ενσωμάτωσης εικονικού δικτύου

Τα εικονικά δίκτυα αποτελούν κεντρικό στοιχείο του Διαδικτύου του μέλλοντος και αποτελούν τη βάση για τη παροχή της υποδομής ως υπηρεσία (Infrastructure as a Service – IaaS) και δίνουν τη δυνατότητα της διάθεσης των φυσικών πόρων για τη δημιουργία δικτύων για τις ανάγκες μιας συγκεκριμένης υπηρεσίας. Το πρόβλημα της ενσωμάτωσης εικονικού δικτύου ΕΕΔ (Virtual Network Embedding – VNE) συνίσταται στη εκχώρηση πόρων του φυσικού δικτύου σε μια εικονική αίτηση με στόχο τη βέλτιστη αξιοποίηση της υποδομής.

Το πρόβλημα ΕΕΔ αφορά στην αντιστοίχηση τόσο των κόμβων (π.χ. εξυπηρετητές, δρομολογητές) όσο και των ζεύξεων του εικονικού δικτύου στους αντίστοιχους πόρους του δικτυακού υποστρώματος (substrate network). Αναφερόμαστε σε δικτυακό υπόστρωμα καθώς δεν είναι απαραίτητο το εικονικό αίτημα να αντιστοιχίζεται σε φυσικό δίκτυο, αλλά το δικτυακό υπόστρωμα μπορεί με τη σειρά του να είναι εικονικό.

Για τη μοντελοποίηση του προβλήματος τόσο το δικτυακό υπόστρωμα όσο και το εικονικό δίκτυο αναπαρίστανται με γράφους, οι κορυφές των οποίων αναπαριστούν τους

κόμβους του δικτύου και οι ακμές τις ζεύξεις μεταξύ κόμβων. Οι κόμβοι υπόκεινται σε περιορισμούς ως προς τους υπολογιστικούς πόρους (π.χ. CPU, RAM) αλλά και άλλα κριτήρια όπως η γεωγραφική τους θέση κλπ. Οι ζεύξεις υπόκεινται σε περιορισμούς ως προς το εύρος ζώνης, την καθυστέρηση κλπ. Κατά την διάρκεια της ΕΕΔ, θα πρέπει να λαμβάνεται υπόψη το σύνολο των στόχων των οντοτήτων που συνθέτουν το περιβάλλον εικονικού δικτύου. Για παράδειγμα, οι πάροχοι υποδομής επιθυμούν την μεγιστοποίηση του κέρδους τους, με τη βελτιστοποίηση του κόστους ενσωμάτωσης που έχει να κάνει με τους πόρους που δεσμεύονται, το ποσοστό αποδοχής εικονικών αιτημάτων και την καθυστέρηση εξυπηρέτησης ενός αιτήματος παροχής ΕΔ.

Το πρόβλημα της ενσωμάτωσης εικονικού δικτύου ανήκει στην κατηγορία των NP-hard προβλημάτων καθώς ανάγεται στο NP-hard multi-way separator πρόβλημα (Yu, 2008), (Anderesen, 2002). Ακόμα και έχοντας πραγματοποιήσει την αντιστοίχηση των κόμβων, η βέλτιστη ανάθεση των εικονικών ζεύξεων σε μονοπάτια του γράφου υποστρώματος ανήκει στην NP-hard κλάση αφού ανάγεται στο πρόβλημα μη διαχωρίσιμων ροών (unsplittable flow problem). Ως εκ τούτου έχουν προταθεί διάφορες ευριστικές προσεγγίσεις, οι συνηθέστερες εκ των οποίων είναι οπισθοδρόμηση (backtracking) (Lischka, 2009), προσομοιωμένη εξέλιξη (simulated annealing) (Ricci, 2003), προσεγγιστικοί αλγόριθμοι (approximation algorithms) (Chowdhury, 2009b).

Συνοπτικά, οι λύσεις που έχουν προταθεί διακρίνονται σε πραγματικού χρόνου (online) και μη (offline), ανάλογα με το αν λύνουν το πρόβλημα της ενσωμάτωσης κατά την άφιξη του εικονικού αιτήματος ή εξ' αρχής θεωρώντας γνωστά όλα τα εικονικά αιτήματα αντίστοιχα. Διακρίνονται επίσης σε κατανεμημένες και μη ανάλογα με το αν το πρόβλημα λύνεται από μια οντότητα ή πολλές. Επιπλέον έχουμε λύσεις όπου οι πόροι που ενσωματώνουν είναι ομογενείς ή ετερογενείς (συνύπαρξη π.χ. διακομιστών, δρομολογητών και μεταγωγέων σε ένα εικονικό αίτημα) και τέλος λύσεις που λύνουν το πρόβλημα προσεγγιστικά με κάποια ευριστική ή όχι.

Επιπλέον διαφέρουν ως προς τον τρόπο που αντιμετωπίζουν τα υπό-προβλήματα ενσωμάτωσης κόμβων και ζεύξεων. Κάποιες λύσεις αντιμετωπίζουν ξεχωριστά τα δύο υπό-προβλήματα. Η ανάθεση των κόμβων γίνεται με κάποιο άπληστο αλγόριθμο και των ζεύξεων είτε με τον αλγόριθμο συντομότερων μονοπατιών (Eppstein, 1999), είτε, επιτρέποντας στην κίνηση μιας ζεύξης να διαχωριστεί σε περισσότερες από μία ζεύξεις στο υπόστρωμα, με τον αλγόριθμο πολλαπλών ροών (Pióro, 2004). Άλλες λύσεις λύνουν από κοινού τα δύο προβλήματα (Chowdhury, 2009b), (Cheng, 2011), (Papagianni, 2013).

2.1.2 Ενσωμάτωση εικονικών δικτύων με ανοχή σε σφάλματα

Ενώ έχει προταθεί μεγάλη ποικιλία αλγορίθμων για την ενσωμάτωση εικονικών δικτύων, το πρόβλημα της ενσωμάτωσης με ανοχή σε σφάλματα (Survival Virtual Network Embedding – SVNE) είναι ακόμη ανοιχτό. Το πρόβλημα έγκειται στην ενσωμάτωση του εικονικού δικτύου στο δίκτυο υποστρώματος με τέτοιο τρόπο ώστε μετά από κάποια βλάβη το δίκτυο να παραμένει λειτουργικό. Η επαναφορά από τη βλάβη θα πρέπει να είναι αδιαφανής στους χρήστες του εικονικού δικτύου.

2.1.2.1 Τύποι και χαρακτηριστικά σφαλμάτων

Το πρόβλημα της ενσωμάτωσης εικονικών δικτύων με αντοχή σε σφάλματα αφορά τόσο σε σφάλματα στο εικονικό δίκτυο, όσο και στο δίκτυο υποστρώματος. Γενικά μπορούμε να διακρίνουμε δύο μεγάλες κατηγορίες σφαλμάτων. Τα σφάλματα σε ζεύξεις και τα σφάλματα σε κόμβους. Πρακτικά σφάλμα σε κόμβο ισοδυναμεί με σφάλμα σε όλες τις προσκείμενες σε αυτόν ζεύξεις. Σύμφωνα με το (Markopoulou, 2008) 20% των βλαβών σε ένα IP δίκτυο κορμού προκύπτουν εξ' αιτίας της συντήρησης του δικτύου. Από το υπόλοιπο 80% το 30% παρατηρείτε σε ομαδικές βλάβες σε ζεύξεις που υποδεικνύει σφάλμα σε κάποιο κόμβο ενώ το 70% σε μια ζεύξη κάθε φορά. Επιπλέον, σύμφωνα με τη μελέτη που έγινε στο (Gill, 2011) σε Data Centers βλάβες σε ζεύξεις παρατηρούνται 10 φορές συχνότερα απ' ότι σε κόμβους με τις δεύτερες να προκαλούνται κατά βάση από εργασίες συντήρησης.

2.1.2.2 Μέθοδοι αντιμετώπισης σφαλμάτων

Δύο είναι οι κύριες κατηγορίες στις οποίες μπορούμε να χωρίσουμε τις πολιτικές αντιμετώπισης σφαλμάτων; οι προ-ενεργές (proactive) και οι ενεργητικές (reactive) μέθοδοι.

Στην πρώτη κατηγορία η προστασία από σφάλματα γίνεται με τη δέσμευση εφεδρικών πόρων πριν συμβεί οποιοδήποτε σφάλμα. Οι ενεργητικές μέθοδοι, γνωστοί αλλιώς και ως μηχανισμοί αποκατάστασης, δρουν αφού συμβεί η βλάβη.

Βλάβες στο εικονικό δίκτυο μπορούν να αντιμετωπιστούν με επανεγκατάσταση του εικονικού αιτήματος. Ωστόσο βλάβες στο δίκτυο υποστρώματος επηρεάζουν παραπάνω από ένα εικονικά δίκτυά. Έτσι, για βλάβες σε κόμβους τους δικτύου υποστρώματος πρέπει να βρεθεί ένας νέος κόμβος στο δίκτυο υποστρώματος στον οποίο θα μετακινηθούν (node migration) οι εικονικοί κόμβοι που φιλοξενούνται στον κόμβο που υπέστη τη βλάβη. Από την άλλη, βλάβη σε ζεύξη αντιμετωπίζεται είτε σε επίπεδο ζεύξης είτε σε επίπεδο μονοπατιού. Στην πρώτη περίπτωση κάθε κύρια ζεύξη θα καλύπτεται από ένα προκαθορισμένο μονοπάτι που θα την παρακάμπτει. Στη δεύτερη περίπτωση η κίνηση στο μονοπάτι σε ζεύξη του οποίου υπήρξε σφάλμα θα πρέπει να επαναδρομολογηθεί μέσω κάποιου άλλου μονοπατιού, ασύνδετο με το αρχικό, με τους ίδιου κόμβους προέλευσης και προορισμού.

2.2 Αλγόριθμοι για το πρόβλημα της ενσωμάτωσης εικονικών δικτύων με ανοχή σε σφάλματα

Σε αυτή την ενότητα παρουσιάζονται οι προτεινόμενοι στην βιβλιογραφία αλγόριθμοι για ενσωμάτωση εικονικών δικτύων με ανοχή σε σφάλματα.

2.2.1 Αλγόριθμοι για ανοχή σε σφάλματα σε ζεύξεις

2.2.1.1 Προστασία και αποκατάσταση Ζεύξεων

Η πιο δημοφιλής μέθοδος για την αποκατάσταση σφαλμάτων σε ζεύξεις είναι η ταχεία επαναδρομολόγηση της κίνησης πάνω από εφεδρικά μονοπάτια.

Στο (Rhaman, 2013) προτείνεται αρχικά μία προενεργός πολιτική για την προστασία και αποκατάσταση δε επίπεδο διαδρομής (path restoration). Ειδικότερα προτείνουν μια εκδοχή του προβλήματος πολλαπλών ροών με ανοχή σε σφάλματα σε ζεύξεις, θεωρώντας ότι έχει γίνει αρχικά η αντιστοίχιση των κόμβων, όπως πχ στο [6]. Για κάθε διαδρομή θα πρέπει να δεσμευτούν πόροι για την εφεδρική της, εξασφαλίζοντας ταυτόχρονα ότι οι δύο διαδρομές είναι ασύνδετες (disjoint). Πρόκειται για Μεικτό Ακέραιο Προγραμματισμό αφού για να εξασφαλιστεί ότι οι διαδρομές θα είναι ασύνδετες χρησιμοποιούνται δύο ακέραιες μεταβλητές. Η ευριστική που προτείνεται στη συνέχεια σπάει το πρόβλημα σε δύο προβλήματα Γραμμικού Προγραμματισμού. Στο πρώτο γίνεται κανονικά οι αντιστοίχιση των ζεύξεων για την κύρια ροή, κρατώντας όμως σε μία λογική μεταβλητή 1 για κάθε ζεύξη του υποστρώματος από την οποία περνάει κύρια ροή και 0 διαφορετικά. Στο δεύτερο επαναλαμβάνεται η διαδικασία για τις εφεδρικές ζεύξεις με τον περιορισμό να μην αντιστοιχηθούν σε ζεύξη υποστρώματος για την οποία η λογική μεταβλητή έχει πάρει την τιμή 1. Προτείνεται επίσης μια υβριδική μέθοδος για το ίδιο πρόβλημα. Αρχικά, σε μη πραγματικό χρόνο, για κάθε ζεύξη του υποστρώματος προϋπολογίζεται ένα σύνολο εναλλακτικών διαδρομών. Στη συνέχεια, σε πραγματικό χρόνο, με την άφιξη μιας εικονικής αίτησης γίνεται η αντιστοίχιση των κόμβων κατά τα γνωστά. Στη συνέχεια λύνεται το multi-commodity flow πρόβλημα για την κύρια ροή και υπολογίζεται για κάθε εικονική ζεύξη το εύρος ζώνης που εκχωρείται σε κάθε μονοπάτι απ' όπου περνάει ροή για τη ζεύξη αυτή. Όταν υπάρξει βλάβη λύνεται ένα νέο LP που εξασφαλίσει ότι η ροή που περνούσε από τη ζεύξη όπου υπήρξε σφάλμα θα περνάει τώρα από ένα από τα προϋπολογισμένα για τη ζεύξη αυτή εναλλακτικά μονοπάτια.

Στο (Guo, 2011) η διαφορά είναι ότι η κίνηση της ζεύξης που υπέστη βλάβη θα επαναδρομολογηθεί τοπικά και όχι απ' άκρη σ' άκρη. Το μειονέκτημα της απ' άκρη σ' άκρη

αποκατάστασης είναι ότι η πληροφορία θα έπρεπε να γυρίσει πίσω στην πηγή πριν επαναδρομολογηθεί. Προτείνονται δύο λύσεις: (i) το εύρος ζώνης του δικτύου υποστρώματος μοιράζεται σε μία πρωτεύουσα και εφεδρική ροή, με τις εφεδρικές να μπορούν να διαμοιράζονται εύρος ζώνης (Share on Demand) και (ii) για κάθε ζεύξη του δικτύου υποστρώματος δεσμεύεται εφεδρικό εύρος ζώνης πριν την άφιξη οποιουδήποτε εικονικού αιτήματος (Share Pre Allocation). Η πρώτη λύση μπορεί να εξοικονομήσει σημαντικά εύρος ζώνης, ειδικά για χαμηλό φορτίο αιτήσεων, αλλά ο αλγόριθμος θα πρέπει να εκτελείται για κάθε εικονικό αίτημα.

2.2.1.2 Προστασία μονοπατιού με μετακίνηση κόμβων

Στο (Yu, 2011) οι συντάκτες προτείνουν μια διαφορετική προσέγγιση, καθώς η επαναδρομολόγηση της κίνησης από εφεδρικά μονοπάτια μπορεί να προκαλέσει έντονο κατακερματισμό με αποτέλεσμα να μειώνεται το ποσοστό αποδοχών και υποβέλτιστη από πλευράς ελαχιστοποίησης κόστους εκχώρηση. Συγκεκριμένα υπολογίζεται ένα εφεδρικό δέντρο που προκύπτει μετακινώντας έναν από τους ακραίους κόμβους του μονοπατιού και εξασφαλίζοντας ότι το νέο μονοπάτι είναι ασύνδετο με το αρχικό. Σε περίπτωση που αυτό ελαχιστοποιεί το κόστος αντί να χρησιμοποιηθεί το εφεδρικό μονοπάτι προτιμάται η μεταφορά του ακραίου κόμβου.

2.2.2 Αλγόριθμοι για ανοχή σε σφάλματα σε κόμβους

2.2.2.1 Ενσωμάτωση σε δύο βήματα

Οι συντάκτες στο (Yu, 2013) προτείνουν την επαύξηση του εικονικού αιτήματος με εφεδρικούς εικονικούς κόμβους οι οποίοι καλύπτουν τους κόμβους του αιτήματος. Συγκεκριμένα, κάθε εφεδρικός κόμβος συνδέεται με όλους τους κόμβους με τους οποίους συνδέεται ο κόμβος που καλύπτει. Στη συνέχεια το επαυξημένο αυτό δίκτυο ενσωματώνεται στο δίκτυο υποστρώματος. Για τον περιορισμό της υπολογιστικής πολυπλοκότητας η ενσωμάτωση γίνεται σε δύο βήματα. Πρώτα γίνεται η αντιστοίχηση του αρχικού εικονικού αιτήματος με έναν από τους προτεινόμενους για το VNE πρόβλημα αλγορίθμους, πχ (Chowdhury, 2009b). Στο δεύτερο βήμα αντιστοιχίζονται στο δίκτυο υποστρώματος οι εφεδρικοί κόμβοι του επαυξημένου εικονικού αιτήματος και δεσμεύεται εύρος ζώνης για εφεδρικά μονοπάτια που εξασφαλίζουν την επικοινωνία τους με τους κόμβους του αρχικού αιτήματος. Οι συντάκτες προτείνουν διαμοιρασμό του εύρους ζώνης είτε ανάμεσα στα εφεδρικά μονοπάτια (backup share) είτε ανάμεσα στα εφεδρικά και ενεργά μονοπάτια (cross share).

2.2.2.2 Προστασία κόμβων με περιορισμούς τοπικότητας

Μετρικές που αφορούν στην ποιότητα παροχής υπηρεσιών (π.χ. χρόνος ανταπόκρισης, καθυστέρηση) καθώς και γεωγραφικοί περιορισμοί που τίθενται από τον πάροχο υπηρεσιών καθιστούν τον περιορισμό τοπικότητας ρεαλιστική απαίτηση για την ενσωμάτωση του εικονικού δικτύου στο δίκτυο υποστρώματος. Έτσι στο (Hu, 2012) προτείνεται ένας αλγόριθμος όπου για κάθε εικονικό κόμβο αρχικά καθορίζεται ένα σύνολο κόμβων του δικτύου υποστρώματος που πληροί τις προδιαγραφές εγγύτητας. Για την εκχώρηση των κόμβων του εικονικού αιτήματος χρησιμοποιείται ο αλγόριθμος που προτείνεται στο (Chowdhury, 2009b). Στη συνέχεια για από τους εφεδρικούς κόμβους, από τους κόμβους του δικτύου υποστρώματος που ικανοποιούν τους περιορισμούς τοπικότητας και χωρητικότητας επιλέγεται κάθε φορά αυτός που εμφανίζεται συχνότερα και αντιστοιχίζεται στον εγγύτερο ενσωματωμένο εικονικό κόμβο.

2.2.2.3 Διαμοιρασμός εφεδρικών κόμβων με ανοχή στα σφάλματα

Έστω ένα εικονικό δίκτυο i προς ενσωμάτωση με n_i κόμβους και k_i εφεδρικούς. Έστω ότι έχουμε τώρα m + 1 αιτήσεις. Στο (Yeow, 2011) προτείνεται το πρώτο αίτημα να δεσμεύει τουλάχιστον τόσους εφεδρικούς κόμβους όσους όλα τα άλλα μαζί. Έτσι το πρώτο θα μπορεί να μοιράζεται τους εφεδρικούς κόμβους του με όλα τα εικονικά δίκτυα, ενώ όλα τα υπόλοιπα μόνο με το πρώτο. Εξασφαλίζεται έτσι έως 50% εξοικονόμηση στους εφεδρικούς κόμβους.

2.2.2.4 Προστασία κόμβων για τοπικές βλάβες

Σε αντίθεση με τις προηγούμενες λύσεις που παρουσιάζουν ανοχή σε μεμονωμένα σφάλματα σε κόμβους, οι συντάκτες στο (Yu, 2010) προτείνουν έναν αλγόριθμο για την αντιμετώπιση αποτυχιών που συμβαίνουν συγχρόνως σε μια περιοχή από κόμβους. Με την προϋπόθεση ότι οι βλάβες είναι πεπερασμένες και κοντά μεταξύ τους το πρόβλημα αποσυντίθεται σε μεμονωμένα προβλήματα για κάθε πιθανή βλάβη, συν το αρχικό πρόβλημα για την ενσωμάτωση του εικονικού δικτύου. Υπάρχει ωστόσο κίνδυνος να μετακινηθούν ανεπηρέαστοι κόμβοι. Εναλλακτικά ο αλγόριθμος ενσωματώνει το αρχικό εικονικό αίτημα στο δίκτυο υποστρώματος και οι βλάβες αντιμετωπίζονται μία κάθε φορά, με κίνδυνο ωστόσο αυξημένη χρονική πολυπλοκότητα.

2.2.2.5 Προστασία κόμβων σε Data Centers

Στα υπολογιστικά κέντρα (Data Centers) παρατηρούνται δύο ιδιαιτερότητες. Αφενός ο εξοπλισμός είναι ετερογενής, αφετέρου η διαθεσιμότητα του συστήματος είναι πολύ μεγάλης σημασίας καθώς η μη διαθεσιμότητα υπηρεσιών έχει μεγάλο κόστος. Στόχος του SVNE σε αυτή την περίπτωση είναι η ελαχιστοποίηση των ενεργών μηχανημάτων και του

χρησιμοποιούμενου εύρους ζώνης σε συνδυασμό με τη μεγιστοποίηση εύρους ζώνης από τη διαθεσιμότητα των υπηρεσιών.

Στην ευριστική που προτείνεται στο (Xu, 2012) αρχικά, τα εικονικά μηχανήματα εκχωρούνται σε ενεργούς servers για εξοικονόμηση ενέργειας. Επιπλέον, τα εικονικά μηχανήματα της αίτησης εκχωρούνται με κριτήριο να είναι κοντά μεταξύ τους για ελαχιστοποίηση του κόστους επικοινωνίας. Στη συνέχεια, προστίθενται από το μηχανισμό σταδιακά εφεδρικοί πόροι μέχρι το σύστημα να φτάσει την επιθυμητή διαθεσιμότητα, ώστε να αποφευχθεί η υπερεκτίμηση της ανάγκης για εφεδρικούς πόρους.

2.2.3 Προστασία σε βλάβες από ζεύξεις ή κόμβους

Στο (Houidi, 2010), τέλος, παρουσιάζεται μια αρκετά διαφορετική προσέγγιση από τις προηγούμενες. Πρόκειται για έναν κατανεμημένο αλγόριθμο όπου με χρήστη δραστών που παρακολουθούν τους κόμβους και τις ζεύξεις του φυσικού δικτύου γίνεται η ανίχνευση σφαλμάτων. Σε περίπτωση σφάλματος σε κάποιο κόμβο του δικτύου υποστρώματος ο δράστης του κόμβου θα στείλει μήνυμα σε όλους τους δράστες κόμβων στο ίδιο σύμπλεγμα (cluster) και αυτός που μπορεί να φιλοξενήσει τον εικονικό κόμβο ελαχιστοποιώντας την αντικειμενική συνάρτηση θα επιλεγεί. Στη συνέχεια οι εικονικές ζεύξεις αντιστοιχίζονται μέσω ενός κατανεμημένου αλγορίθμου συντομότερων μονοπατιών (Houidi, 2008). Για σφάλματα σε ζεύξεις εκτελείται μόνο το τελευταίο βήμα.

2.3 Σύγκριση και αξιολόγηση των προτεινόμενων αλγορίθμων

Στα παραπάνω άρθρα είναι εμφανής ό διαχωρισμός των λύσεων σε αυτές με ανοχή σε σφάλματα σε ζεύξεις (Rhaman, 2013), (Guo, 2011), (Yu, 2011) και αυτές με ανοχή σε σφάλματα σε κόμβους (Yu, 2013), (Hu, 2012), (Yeow, 2011), (Yu, 2010), (Xu, 2012), ενώ μόνο ο αλγόριθμος που προτείνεται στο (Houidi, 2010) μεριμνά για την ανοχή σφαλμάτων και σε ζεύξεις και σε κόμβους. Επιπλέον, οι παραπάνω λύσεις περιορίζονται στην αντιμετώπιση μεμονωμένων σφαλμάτων σε κόμβους και ζεύξεις, βασιζόμενες σε παρατηρήσεις όπως αυτές στο (Markopoulou, 2008) που δίνουν πολύ μικρότερες πιθανότητες να συμβούν μαζικά σφάλματα. Μόνο στο (Yu, 2010) οι συντάκτες λαμβάνουν υπ' όψιν τους ταυτόχρονη αποτυχία πολλαπλών κόμβων, εφόσον αυτοί βρίσκονται κοντά μεταξύ τους.

Με σύγκριση των παραπάνω παρατηρούμε ότι οι πλειοψηφία των λύσεων που έχουν προταθεί είναι προενεργές (proactive). Μόνο η λύση η οποία προτείνεται από (Rhaman, 2013) έχει ενεργητικό (reactive) χαρακτήρα, καθώς αν και έχουν προϋπολογιστεί εφεδρικά μονοπάτια, το πρόβλημα λύνεται κάθε φορά που παρουσιάζεται σφάλμα και η λύση η οποία προτείνεται από (Houidi, 2010) όπου για κάθε σφάλμα αναζητείται επιτόπου ο κόμβος /

σύνολο ζεύξεων που θα αντικαταστήσουν ο στοιχείο όπου παρουσιάστηκε σφάλμα. Το δίλημμα με το οποίο ερχόμαστε αντιμέτωποι προκειμένου να διαλέξουμε μεταξύ μιας proactive και reactive πολιτικής είναι το tradeoff μεταξύ των πόρων που θα διαθέσουμε και υπολογιστικού κόστους και αποδοτικότητας του αλγορίθμου. Συγκεκριμένα οι proactive λύσεις δεσμεύουν προκαταβολικά πόρους που μπορεί να μην αξιοποιηθούν πλήρως. Από την άλλη, έχοντας προβλέψει τον τρόπο αποκατάστασης, η αποκατάσταση είναι άμεση. Αντίθετα, μια reactive λύση θα έχει χαμηλότερο κόστος ενσωμάτωσης και απαιτώντας λιγότερους πόρους θα έχει καλύτερα ποσοστά αποδοχών. Θα εισάγει ωστόσο μεγαλύτερη καθυστέρηση καθώς ο αλγόριθμος θα πρέπει να ξαναεκτελείται για κάθε βλάβη. Το ποια μέθοδος υπερτερεί εξαρτάται και από τη φύση του προβλήματος. Αν θεωρήσουμε χαμηλό φορτίο μια proactive μέθοδος μειονεκτεί, καθώς μεγάλο ποσοστό των προδεσμευμένων πόρων μένει αχρησιμοποίητο. Αντίθετα σε περιπτώσεις που είναι κρίσιμο το δίκτυο να ανακάμψει όσο το δυνατόν ταχύτερα μια reactive μέθοδος μειονεκτεί καθώς εισάγει καθυστέρηση έχοντας να λύσει σε πραγματικό χρόνο το πρόβλημα και ειδικά σε περιπτώσεις υψηλού φορτίου είναι πιθανό να μην βρεθεί αντιστοίχηση.

Προκειμένου να αντισταθμιστεί η κατασπατάληση πόρων που μπορεί να προκαλέσει μια proactive πολιτική, είναι διαδεδομένη η τακτική διαμοιρασμού πόρων (Guo, 2011), (Yu, 2011), (Yu, 2013), (Hu, 2012), (Yeow, 2011), (Yu, 2010) είτε ανάμεσα σε εφεδρικούς πόρους, είτε ανάμεσα σε εφεδρικούς πόρους και πόρους του λειτουργούντος συστήματος. Βέβαια η τακτική αυτή έχει αντίκτυπο στο γεγονός ότι οι προτεινόμενες λύσεις δεν μπορούν να καλύψουν πάνω από μία βλάβη σε κόμβο ή ζεύξη κάθε φορά.

Μία ακόμη κατηγοριοποίηση των μεθόδων που εξετάστηκαν μπορεί να γίνει βάσει της αντικειμενικής συνάρτησης που θέλουν να βελτιστοποιήσουν. Μια πρόσκληση που έχουν να αντιμετωπίσουν είναι η ελαχιστοποίηση των πόρων που δεσμεύουν και κατ' επέκταση του κόστους υλοποίησης (Guo, 2011), (Yu, 2011), (Yu, 2013), (Hu, 2012), (Yeow, 2011), (Yu, 2010). Εναλλακτικά στόχος μπορεί να είναι η ελαχιστοποίηση του κόστους λειτουργίας όπως στο (Xu, 2012). Στόχος μπορεί να είναι η μεγιστοποίηση του κόστους του παρόχου (Rhaman, 2013) ή και η μεγιστοποίηση του αριθμού των εικονικών αιτήσεων που μπορούν να εξυπηρετηθούν (Guo, 2011).

Τέλος, με μια ανασκόπηση των προτεινόμενων ως τώρα λύσεων βλέπουμε ότι ανοιχτά είναι τα θέματα της ανοχής σε σφάλματα ταυτοχρόνως για κόμβους και ζεύξεις, καθώς μόνο το (Houidi, 2010) κινείται προς αυτή την κατεύθυνση, της ανοχής σε μαζικά σφάλματα στις οντότητες του δικτύου υποστρώματος αλλά και σε σφάλματα που αφορούν ετερογενείς κόμβους, αφού στα (Yu, 2013), (Hu, 2012), (Yeow, 2011), (Yu, 2010), (Xu, 2012) οι κόμβοι που προστατεύονται είναι αποκλειστικά διακομιστές ή γενικά κόμβοι της υποδομής (facility nodes). Βάσει αυτών των ανοιχτών ερευνητικά ζητημάτων, στην εργασία αυτή προτείνονται

αλγόριθμοι που εξασφαλίζουν την ταυτόχρονη ανοχή σε σύνολο κρίσιμων κόμβων και στις προσκείμενες σε αυτούς ζεύξεις. Επιπλέον, εξετάζονται δύο πολιτικές για την πρόβλεψη εφεδρικών πόρων, εκ των οποίων η μια εξασφαλίζει ανοχή σε μαζικά σφάλματα, και συγκρίνουμε την επίδοσή τους.

3

Βέλτιστη Ενσωμάτωση Εικονικών Πόρων με Ανοχή σε Σφάλματα

Το πρόβλημα που θέλουμε να αντιμετωπίσουμε είναι η ενσωμάτωση εικονικού δικτύου με αντοχή σε σφάλματα τόσο σε κόμβους όσο και όσο και σε ζεύξεις, ελαχιστοποιώντας το κόστος της ενσωμάτωσης και το μέγεθος των μονοπατιών στο δίκτυο υποστρώματος, μεταξύ των κόμβων όπου αντιστοιχίζονται οι κόμβοι του εικονικού δικτύου. Η λύση συνίσταται στην επίλυση ενός προβλήματος Μεικτού Ακέραιου Γραμμικού προγραμματισμού ΜΑΓΠ (Mixed Integer Linear Programming – MILP).

Στη συνέχεια, δίνεται η μαθηματική αναπαράσταση των εικονικών αιτήσεων και του δικτύου υποστρώματος και, αρχικά, μοντελοποιείται το MILP πρόβλημα για την ενσωμάτωση εικονικού δικτύου με ανοχή σε σφάλματα σε κόμβους και ακολουθεί η τροποποίηση του προβλήματος για να εισαχθεί ταυτοχρόνως ανοχή και σε σφάλματα σε ζεύξεις.

3.1 Μοντελοποίηση του δικτύου και ορισμός του προβλήματος

Ένα εικονικό αίτημα αναπαρίσταται από ένα μη κατευθυνόμενο γράφο με βάρη $G^V = (N^V, C, E^V)$ όπου N^V είναι το σύνολο των εικονικών κόμβων, $C \subseteq N^V$ είναι το σύνολο των κρίσιμων κόμβων (κόμβοι που πρέπει να καλύπτονται από κάποιον εφεδρικό) και E^V το

σύνολο των εικονικών ζεύξεων, ακολουθώντας τη μοντελοποίηση που προτείνεται στο (Yu, 2013). Παρομοίως το δικτυακό υπόστρωμα αναπαρίσταται από έναν γράφο με βάρη $G^S = (N^S, E^S)$. όπου N^S το σύνολο των κόμβων και E^S το σύνολο των ζεύξεων. Ακολουθώντας τη προσέγγιση που προτείνεται στο (Papagianni, 2013), προκειμένου η λύση να γενικεύεται σε ετερογενή υποδομή δικτύου, κάθε κόμβος σχετίζεται με ένα τύπο $a \in A$ (π.χ. διακομιστής, δρομολογητής, κλπ.) έτσι ώστε $n^X \in V_a^X \subseteq N^X, a \in A, X \in \{V, S\}$ και $\bigcup_A V_a^X = N^X, X \in \{V, S\}$. Ανάλογα με τον τύπο του, ο κόμβος $n^X \in V_a^X$ αντιστοιχίζεται σε ένα σύνολο I μη λειτουργικών χαρακτηριστικών γνωρισμάτων, στα οποία θα αναφερόμαστε με τον όρο χωρητικότητες, $c_i(n^X), i \in I, n^X \in V_a^X, X \in \{V, S\}$(π.χ. μνήμη, αποθηκευτικός χώρος, αριθμός δικτυακών διεπαφών, κλπ.). Επιπλέον, κάθε ακμή $(n^X, m^X) \in E^X, \forall n^X, m^X \in N^X, X \in \{V, S\}$ σχετίζεται με ένα εύρος ζώνης, χωρητικότητας $bw(n^X, m^X)$.

Κάθε εικονικός κόμβος της αίτησης θα πρέπει να αντιστοιχηθεί σε ένα μοναδικό κόμβο του δικτύου υποστρώματος. Ορίζουμε έτσι τη συνάρτηση:

$$M^N: N^V \rightarrow N^S \tag{3.1}$$

όπου: $M^N(n^V) \in V_a^S, n^V \subseteq N^V$

ενώ κάθε εικονικός κόμβος για το ίδιο αίτημα απαιτούμε να αντιστοιχίζεται σε διαφορετικό κόμβο του δικτύου υποστρώματος:

$$M^N(n^V) = M^N(m^V) \Leftrightarrow n^V \equiv m^V \tag{3.2}$$

Τέλος, ένας εικονικός κόμβος προκειμένου να αντιστοιχηθεί σε ένα κόμβο του υποστρώματος υπόκειται σε περιορισμούς ως προς το διάνυσμα χωρητικοτήτων. Συγκεκριμένα, κάθε απαιτούμενη χωρητικότητα του εικονικού κόμβου $i \in I$ dδεν θα πρέπει να ξεπερνά την αντίστοιχη διαθέσιμη χωρητικότητα C_i του κόμβου του δικτύου υποστρώματος στον οποίο θα αντιστοιχηθεί:

$$c_i(n^V) \leq C_i(n^s) \tag{3.3}$$

όπου η διαθέσιμη χωρητικότητα ορίζεται ως η χωρητικότητα του κόμβου του δικτύου υποστρώματος από την οποία αφαιρείται το άθροισμα των χωρητικοτήτων των εικονικών κόμβων που έχουν ήδη αντιστοιχηθεί στον κόμβο αυτό:

$$C_i(n^s) = c_i(n^S) - \sum_{m^V, M^N(m^V)=n^S} c_i(m^V) \tag{3.4}$$

Μια εικονική ζεύξη αντιστοιχίζεται σε μία διαδρομή P^S ή , επιτρέποντας τη διάσπαση της ροής της εικονικής ζεύξης, σε ένα σύνολο διαδρομών $\boldsymbol{P^S}$ στο δικτυακό υποστρώματος ανάμεσα στους κόμβους στους οποίους έχουν απεικονιστεί τα άκρα της εικονικής ζεύξης. Ορίζουμε επομένως τη συνάρτηση:

$$M^E: E^V \rightarrow \boldsymbol{P^S} \tag{3.5}$$

όπου $M^E(n^V, m^V) \in \boldsymbol{P^S}\big(M^V(n^V), M^V(m^V)\big)$

Το εύρος ζώνης της εικονικής ζεύξης θα πρέπει να μην ξεπερνάει το άθροισμα του εύρους ζώνης των μονοπατιών που θα αντιστοιχηθούν σε αυτή:

$$bw(n^V, m^V) \leq \sum_{P^S \in M^E(m^V, n^V)} bw(P^S)$$

(3.6)

Το εύρος ζώνης του μονοπατιού περιορίζεται από το διαθέσιμο εύρος ζώνης της ζεύξης του δικτύου υποστρώματος με το μικρότερο διαθέσιμο εύρος ζώνης που ανήκει στο μονοπάτι αυτό:

$$bw(P^S) = \min_{u^S, v^S \in P^S} BW(u^S, v^S) \quad (3.7)$$

Το διαθέσιμο εύρος ζώνης μιας ζεύξης του δικτύου υποστρώματος, τέλος, ισούται με τη χωρητικότητα της ζεύξης μείον το άθροισμα εύρους ζώνης των εικονικών ζεύξεων, μονοπάτια των οποίων περνάνε από αυτή την ζεύξη αυτή:

$$\mathrm{BW}(\mathrm{u}^S, v^S) = bw(u^S, u^S) - \sum_{(i^V, j^V)} bw(i^V, j^V) \text{ όπου } (u^S, v^S) \in M^E(i^V, j^V) \quad (3.8)$$

Προκειμένου η παραπάνω αντιστοίχηση να έχει ανοχή σε σφάλματα, ο αρχικός γράφος του εικονικού αιτήματος θα πρέπει να επαυξηθεί με πλεονάζοντες εφεδρικούς κόμβους, δημιουργώντας έτσι ένα αξιόπιστο εικονικό αίτημα. Επιπλέον, ζεύξεις με επαρκές εύρος ζώνης πρέπει να προβλεφθούν για τη σύνδεση των εφεδρικών κόμβων με τους γειτονικούς κόμβους των κρίσιμων κόμβων. Τέλος για να έχει η λύση και ανοχή σε σφάλματα σε ζεύξεις ένα σύνολο ασύνδετων μονοπατιών με αυτά στα οποία αντιστοιχίζεται το αρχικό εικονικό αίτημα πρέπει να εξασφαλιστεί.

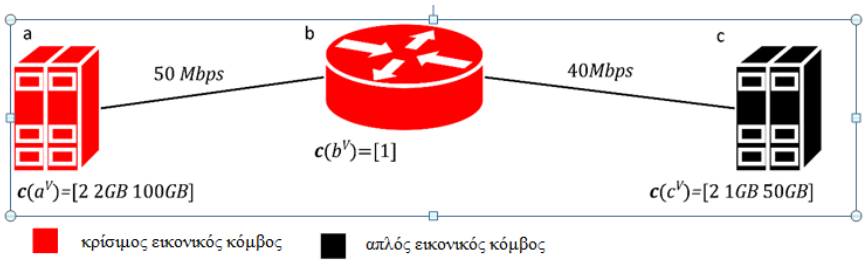
3.2 Ενσωμάτωση εικονικού αιτήματος με ανοχή σε σφάλματα σε κόμβους

3.2.1 Σχεδιασμός και ενσωμάτωση του αξιόπιστου εικονικού αιτήματος

Για την ενσωμάτωση αιτήματος με ανοχή σε σφάλματα, ακολουθείται μια μέθοδος δύο βημάτων (Yu, 2013). Στο πρώτο βήμα, το εικονικό αίτημα προσαυξάνεται με πλεονάζοντες εικονικούς κόμβους και ζεύξεις με επαρκές εύρος ζώνης. Ο επαυξημένος γράφος στη συνέχεια αντιστοιχίζεται στο δίκτυο υποστρώματος.

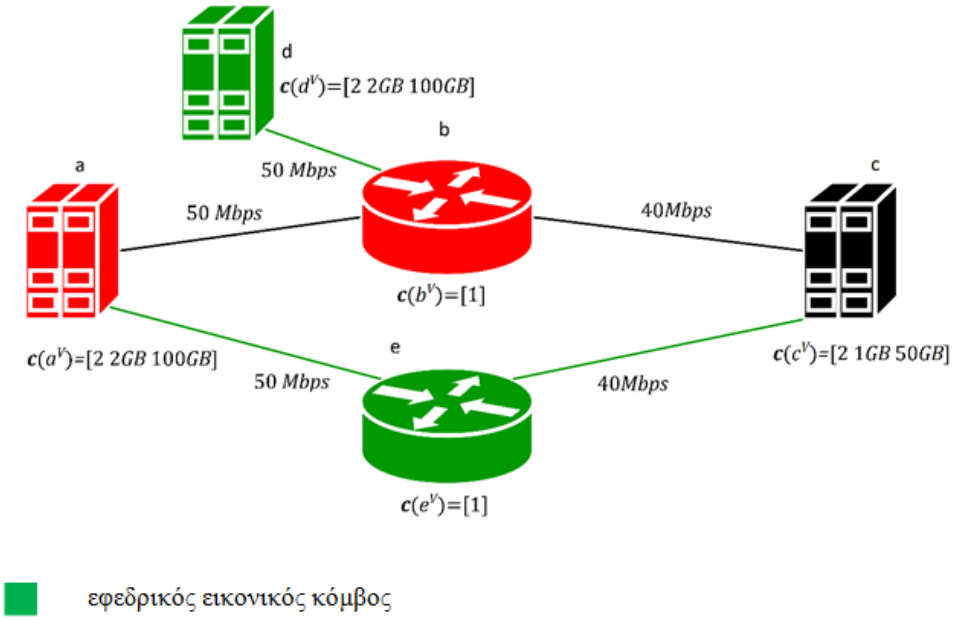
Παρακάτω παρουσιάζονται σε ένα απλό παράδειγμα τα δύο βήματα της ενσωμάτωσης, για το αίτημα του Σχήματος 3.1, όπου ο ένας διακομιστής καθώς και ο δρομολογητής θεωρούνται κρίσιμοι. Θεωρούμε δύο διαθέσιμους τύπους κόμβων (υπολογιστικοί κόμβοι και δρομολογητές). Οι υπολογιστικοί κόμβοι συσχετίζονται με μη λειτουργικά χαρακτηριστικά {*αριθμός πυρήνων ΚΜΕ, μνήμη, δίσκος αποθήκευσης*}, ενώ η χωρητικότητα του

δρομολογητή αντιστοιχεί στον αριθμό των λογικών δρομολογητών που μπορεί να φιλοξενήσει..



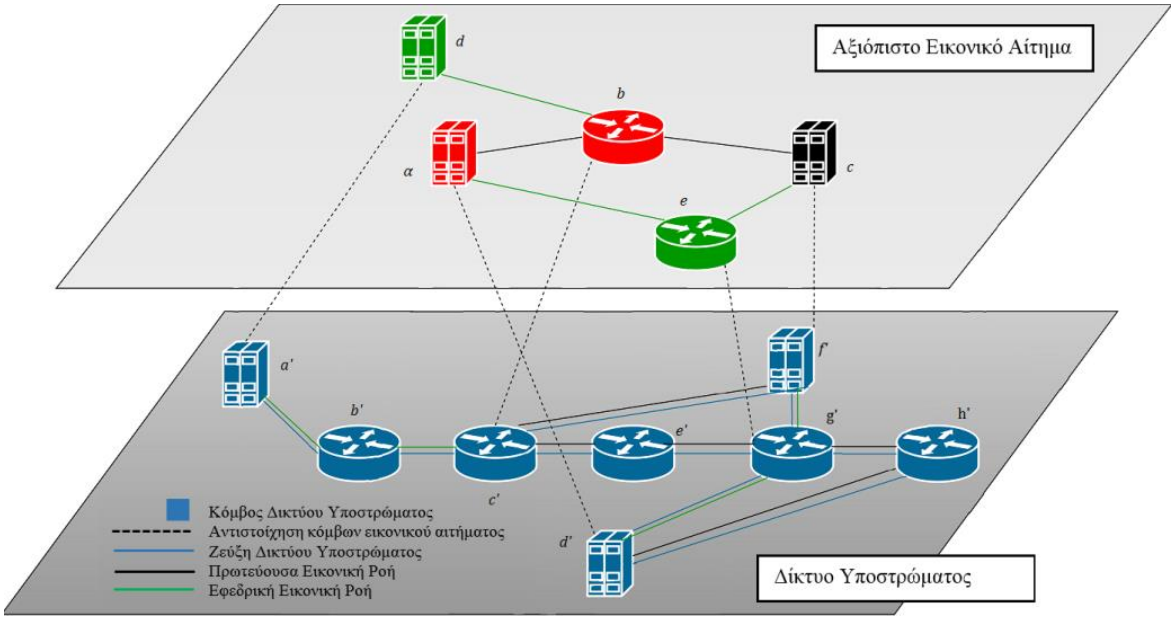
ΣΧΗΜΑ 3.4Αρχικό Εικονικό Αίτημα

Στη συνέχεια το αρχικό εικονικό αίτημα προσαυξάνεται με δύο εφεδρικούς εικονικούς κόμβους και τις απαραίτητες εικονικές ζεύξεις ώστε να σχηματιστεί το αξιόπιστο εικονικό αίτημα, όπως φαίνεται στο Σχήμα 3.2.



ΣΧΗΜΑ 3. 5 Αξιόπιστο Εικονικό Αίτημα

Τέλος, το αξιόπιστο εικονικό αίτημα αντιστοιχίζεται στο δίκτυο υποστρώματος, όπως φαίνεται στο Σχήμα 3.3.



ΣΧΗΜΑ 3. 6 Ενσωμάτωση εικονικού αιτήματος στο δίκτυο υποστρώματος

Στους παρακάτω πίνακες παρουσιάζονται οι χωρητικότητες των κόμβων και ζεύξεων του δικτύου υποστρώματος και οι διαθέσιμες χωρητικότητες μετά την εκχώρηση του εικονικού αιτήματος, θεωρώντας ότι δεν έχει πριν εκχωρηθεί άλλο εικονικό αίτημα:

u^S	$\boldsymbol{c}(u^S)$	$\boldsymbol{C}(u^S)$
a'	$[8\ 16GB\ 10TB]$	$[6\ 14GB\ 9.9TB]$
b'	$[15]$	$[15]$
c'	$[14]$	$[13]$
d'	$[8\ 16GB\ 50TB]$	$[6\ 14GB\ 4.9TB]$
e'	$[15]$	$[15]$
f'	$[8\ 16GB\ 10TB]$	$[6\ 15GB\ 9.95TB]$
g'	$[12]$	$[11]$
h'	$[14]$	$[14]$

ΠΙΝΑΚΑΣ 3. 3 Αρχική και διαθέσιμη χωρητικότητες κόμβων υποστρώματος

(u^S, v^S)	$bw(u^S, v^S)$	$BW(u^S, v^S)$
(a', b')	$100Mbps$	$50Mbps$
(b', c')	$1Gbps$	$974\ Mbps$
(c', e')	$1Gbps$	$974\ Mbps$

(c', f')	$100Mbps$	60 $Mbps$
(e', g')	$1Gbps$	974 $Mbps$
(g', d')	$100Mbps$	50 $Mbps$
(g', f')	$100Mbps$	60 $Mbps$
(g', h')	$1Gbps$	974 $Mbps$
(d', h')	$100Mbps$	50 $Mbps$

ΠΙΝΑΚΑΣ 3. 4 Αρχικό και διαθέσιμο εύρος ζώνης ζεύξεων υποστρώματος

3.2.1.1 Σχεδιασμός του αξιόπιστου εικονικού αιτήματος.

Για το σχηματισμό του αξιόπιστου εικονικού αιτήματος θα ακολουθήσουμε την πλεονάζουσα-κατά-1 και πλεονάζουσα-κατά-k πολιτική, παρόμοια με το (Yu, 2013). Πιο συγκεκριμένα, για την πλεονάζουσα-κατά-k πολιτική, ένα σύνολο από k εφεδρικούς κόμβους $b \in B^K$ προστίθενται στον αρχικό γράφο του εικονικού αιτήματος G^V έτσι ώστε για κάθε κρίσιμο κόμβος $c_a, a \in A, c_a \epsilon\, C$ υπάρχει ένας αντίστοιχος εφεδρικός κόμβος $b(c_a)$ του ίδιου τύπου $a \in A$. Κάθε χωρητικότητα του εφεδρικού κόμβου θα πρέπει να είναι ίση με τη μέγιστη αντίστοιχη χωρητικότητα των κόμβων που καλύπτει. Ορίζουμε έτσι το σύνολο των εφεδρικών κόμβων ως $B = \{b(c_a), c_a \in C_a \subseteq C\}$, όπου $C_a \subseteq N_a^V$ οι κρίσιμοι κόμβοι τύπου $a \in A$. Το σύνολο των εφεδρικών ζεύξεων που προστίθενται στο γράφο G^V ορίζεται ως $E^B = \{(b(c_a), v) | \exists (c_a, v \in E_V, c_a \in C, v \in N^V)\}$. Η απαίτηση σε εύρος ζώνης για κάθε εφεδρική ζεύξη με αυτή της αντίστοιχης ζεύξης του αρχικού εικονικού αιτήματος, δηλαδή $bw(b(c_a), v) = bw(c_a, v), \forall v: \exists (c_a, v) \in E^V$. Για τηνπλεονάζουσα-κατά-1 πολιτική, θεωρούμε ένα κόμβο από κάθε κατηγορία, που καλύπτει όλους τους κρίσιμους κόμβους της κατηγορίας αυτής. Ας σημειωθεί ότι εφόσον θεωρούμε ετερογενείς κόμβους η ελάχιστη τιμή που μπορεί να λάβει το k ισούται με την πληθυκότητα του συνόλου A, ένας εφεδρικός κόμβος για κάθε κατηγορία. Στην περίπτωση αυτή αναγόμαστε στην πλεονάζουσα-κατά-1 πολιτική. Στο εξής όποτε αναφερόμαστε στην πλεονάζουσα-κατά-k πολιτική, ωστόσο, θα εννοούμε ότι το k ισούται με την πληθυκότητα του συνόλου C των κρίσιμων κόμβων. Στην περίπτωση αυτή, η λύση θα είναι ανεκτική σε πολλαπλά ταυτόχρονα σφάλματα σε κόμβους.

Τον επαυξημένο, αξιόπιστο γράφο, θα συμβολίσουμε ως $G^R = \{N^R = N^V \cup B, E^R = E^V \cup E^B\}$

3.2.1.2 Ενσωμάτωση των αξιόπιστου εικονικού αιτήματος

Όπως στο (Papagianni, 2013) και στο (Yu, 2013) θα στηριχθούμε στην ιδέα που προτάθηκε στο (Chowdhury, 2009b), αγνοώντας του γεωγραφικούς περιορισμούς. Συγκεκριμένα, αρχικά επαυξάνουμε το δίκτυο υποστρώματος με τους κόμβους του αξιόπιστου εικονικού

αιτήματος, προσθέτουμε δηλαδή τους $|N^V|$ αρχικούς και $|B|$ εφεδρικούς κόμβους στο δίκτυο υποστρώματος. Κάθε εικονικός κόμβος $n^V \in N^R$ συνδέεται με κάθε κόμβο του δικτύου υποστρώματος $n^S \in N^S$ με μία ακμή άπειρης χωρητικότητας. Συμβολίζουμε τον επαυξημένο γράφο $G^{S'} = (N^{S'}, E^{S'})$.

Την αντιστοίχηση εικονικών κόμβων σε κόμβους του δικτύου υποστρώματος μας δίνει η λύση ενός προβλήματος Μεικτού Ακέραιου Προγραμματισμού (MILP) όπως περιγράφεται παρακάτω.

3.2.2 Μοντελοποίηση Μεικτού Ακέραιου Προγραμματισμού

Το πρόβλημα που θα λύσουμε έχει τις εξής μεταβλητές:

f_{uv}^{mn}: Το ποσό της κίνησης της εικονικής ζεύξης $(m,n) \in E^V$που δρομολογείται πάνω από τη ζεύξη $(u,v) \in E^{S'}$από τον κόμβο u στον v.

g_{uv}^{bn}: Το ποσό της κίνησης της εφεδρικής εικονικής ζεύξης $(b,n) \in E^B$, που δρομολογείται πάνω από τη ζεύξη $(u,v) \in E^{S'}$ από τον κόμβο u στον v.

x_{uv}^{mn}: δυαδική μεταβλητή ίση με 1 αν κίνηση της εικονικής ζεύξης $(n,m) \in E^V$δρομολογείται μέσω της ζεύξης $(u,v) \in E^{S'}$, 0 αλλιώς.

y_{uv}^{bn}: δυαδική μεταβλητή ίση με 1 αν κίνηση της εφεδρικής εικονικής ζεύξης $(b,n) \in E^B$ δρομολογείται μέσω της $(u,v) \in E^{S'}$, 0 αλλιώς.

Εξετάζοντας τελικά τις μεταβλητές x_{uv}^{mn} όπου ένας εκ των u,v, έστω ο v, είναι εικονικός κόμβος και ο άλλος κόμβος του Substrate network καταλήγουμε σε αντιστιχιση του v στον u όταν η αντιστοιχη μεταβλητή έχει τεθεί. Όμοια για τους εφεδρικούς κόμβους, ο κόμβος v θα αντιστοιχηθεί τελικά στον u, αν η μεταβλητή y_{uv}^{mn} έχει την τιμή 1, όπου ο κόμβος v είναι εφεδρικός και ο u κόμβος του substrate network.

Αντικειμενική Συνάρτηση:

min:

$$\underbrace{\sum_{uv \in E^S} \left(\sum_{nm \in E^V} C_{uv} f_{uv}^{nm} + \sum_{nm \in E^B} C_{uv} g_{uv}^{nm} \right)}_{\text{συνολικό εύρος ζώνης}} +$$

$$\underbrace{\sum_{a \in A} \sum_{w \in V_a^S} \sum_{p \in V_a^V \subseteq N^{S'} \setminus N^S} \left(\sum_{mn \in E^V} D_w x_{pw}^{nm} \sum_{i \in I} c_i(p) + \sum_{mn \in E^B} D_w y_{pw}^{nm} \sum_{i \in I} c_i(p) \right)}_{\text{σύνολο υπολογιστικών πόρων}} +$$

$$\underbrace{\sum_{uv \in E^S} \left(\sum_{nm \in E^V} C_{uv} x_{uv}^{nm} + \sum_{nm \in E^B} C_{uv} y_{uv}^{nm} \right)}_{\text{αριθμών αλμάτων}}$$

(3.9)

όπου:

$$C_{uv} = \frac{1}{BW_{uv} + \delta}, \qquad \delta \to 0$$

$$D_w = \frac{1}{\sum_{i \in I} C_i \ + \delta}\ , \qquad \delta \to 0$$

παράγοντες που εξασφαλίζουν την ισοκατανομή του φορτίου

Περιορισμοί Πεδίου Ορισμού:

$$f_{uv}^{mn} \geq\ 0, \forall u, v \in N^{S'}, \forall (nm) E^V \tag{3.10}$$

$$g_{uv}^{bn} \geq\ 0, \forall u, v \in N^{S'}, \forall (bn) \in E^B \tag{3.11}$$

$$x_{uv}^{mn} \in \{0,1\}, \forall u, v \in N^{S'}, \forall (nm) \in E^V \tag{3.12}$$

$$y_{uv}^{bn} \in \{0,1\}, \forall u, v \in N^{S'}, \forall (bn) \in E^B \tag{3.13}$$

Περιορισμοί ροής κίνησης:

$$\sum_{w \in N^{S'}} f_{uw}^{nm} - \sum_{w \in N^{S'}} f_{wu}^{nm} = 0\ , \qquad \forall (nm) \in E^V, \forall u \in N^{S'} \backslash \{nm\}$$

$$\sum_{w \in N^{S'}} f_{nw}^{nm} - \sum_{w \in N^{S'}} f_{wn}^{nm} = b(n,m), \qquad \forall (nm) \in E^V, n \in N^{S'}$$

$$\sum_{w \in N^{S'}} f_{mw}^{nm} - \sum_{w \in N^{S'}} f_{wm}^{nm} = -b(n,m), \qquad \forall (nm) \in E^V, m \in N^{S'}$$

(3.14a)

$$\sum_{w \in N^{S'}} g_{uw}^{nm} - \sum_{w \in N^{S'}} g_{wu}^{nm} = 0\ , \qquad \forall (nm) \in E^B, n \in B\ , \forall u \in N^{S'} \backslash \{nm\}$$

$$\sum_{w \in N^{S'}} g_{b(n)w}^{b(n)m} - \sum_{w \in N^{S'}} g_{wb(n)}^{b(n)m} = b(n,m), \qquad \forall (nm) \in E^B, b(n) \in N^{S'}$$

$$\sum_{w \in N^{S'}} g_{mw}^{b(n)m} - \sum_{w \in N^{S'}} g_{wm}^{b(n)m} = -b(n,m), \qquad \forall (b(n)m) \in E^V, m \in N^{S'}$$

(3.14b)

$$c_i(p) x_{pw}^{nm} \leq C_i(w), \forall p \in V_a^V, w \in V_a^S, i \in I, a \in A, (n,m) \in E^V \tag{3.15a}$$

$$c_i(d) y_{dw}^{nm} \leq C_i(w), \forall d \in V_a^B, w \in V_a^S, i \in I, a \in A, (n,m) \in E^B \tag{3.15b}$$

$$f_{uv}^{mn} + f_{vu}^{mn} \leq BW(u,v) x_{uv}^{mn},\ \forall u, v \in N^{S'}, \forall (n,m) \in E^V \tag{3.16a}$$

$$g_{uv}^{mn} + g_{vu}^{mn} \leq BW(u,v) y_{uv}^{mn},\ \forall u,v \in N^{S'}, \forall (n,m) \in E^B \quad (3.16b)$$

$$\sum_{mn \in E^V} (f_{uv}^{nm} + f_{vu}^{nm}) + \sum_{mn \in E^B} (g_{uv}^{nm} + g_{vu}^{nm}) \leq BW(u,v), \forall u,v \in N^{S'} \quad (3.17)$$

$$\sum_{p \in V_A^V \subseteq N^{S'} \setminus N^S} x_{pw}^{nm} \leq 1, \forall w \in N^S, \forall (n,m) \in E^V,\ \forall A \quad (3.18a)$$

$$\sum_{p \in V_A^B \subseteq N^{S'} \setminus N^S} y_{pw}^{nm} + \sum_{p \in V_A^V \subseteq N^{S'} \setminus N^S} y_{pw}^{nm} + \sum_{p \in V_A^V \subseteq N^{S'} \setminus N^S : \exists (c,p) \in E^V, c \in C,} x_{pw}^{nm} \leq 1,$$

$$\forall w \in N^S, \forall (n,m) \in E^B,\ \forall A \quad (3.18b)$$

$$\sum_{p \in V_A^V \subseteq N^S} x_{pw}^{nm} = 0, \forall w \in N_{A'}^S \subseteq N^{S'} \setminus N^S,\ \forall mn \in E^V, \forall A, A', A \neq A' \quad (3.19a)$$

$$\sum_{p \in V_A^B \subseteq N^S} y_{pw}^{nm} = 0, \forall w \in N_{A'}^S \subseteq N^{S'} \setminus N^S,\ \forall mn \in E^V, \forall A, A', A \neq A' \quad (3.19b)$$

$$\sum_{w \in V_A^S \subseteq N^S} x_{pw}^{mn} = 1, \forall p \in V_A^V \subseteq N^{S'} \setminus N^S, \forall mn \in E^V, \forall A \quad (3.20a)$$

$$\sum_{w \in V_A^S \subseteq N^S} y_{pw}^{mn} = 1, \forall p \in V_A^B \subseteq N^{S'} \setminus N^S, \forall mn \in E^B, \forall A \quad (3.20b)$$

$$\sum_{w \in V_A^S \subseteq N^S} y_{pw}^{mn} = 1, \forall p \in V_A^V \subseteq N^{S'} \setminus N^S : \exists (c,p) \in E^V, c \in C, \forall mn \in E^V, \forall A \quad (3.20c)$$

$$\sum_{w \in V_A^S \subseteq N^S} x_{pw}^{mn} = 0, \forall p \in B \subseteq N^{S' \setminus N^S}, \forall mn \in E^V, \forall A \quad (3.21)$$

$$x_{pw}^{mn} = y_{pw}^{kl}, \forall p \in N^V \subseteq N^{S'} \setminus N^S : \exists (c,p) \in E^V, c \in C,$$

$$\forall w \in N^S, mn \in E^V, kl \in E^B \quad (3.22)$$

$$x_{uv}^{mn} = x_{vu}^{mn}, \forall u,v \in N^{S'}, \forall (n,m) \in E^V \quad (3.23a)$$

$$y_{uv}^{mn} = y_{vu}^{mn}, \forall u,v \in N^{S'}, \forall (n,m) \in E^B \quad (3.23b)$$

$$x_{uv}^{mn} \leq \lceil f_{uv}^{mn} + f_{vu}^{mn} \rceil, \forall u,v \in N^{S'}, \forall (n,m) \in E^V \quad (3.24a)$$

$$y_{uv}^{mn} \leq \lceil g_{uv}^{mn} + g_{vu}^{mn} \rceil, \forall u,v \in N^{S'}, \forall (n,m) \in E^B \quad (3.24b)$$

$$x_{uv}^{nm} = x_{uv}^{nk} = x_{uv}^{lm}, \forall u,v \in N^{S'} \setminus N^R, \forall k,l \in N^{S'} \setminus N^S, \forall (n,m) \in E^V \quad (3.25a)$$

$$y_{uv}^{nm} = y_{uv}^{nk} = y_{uv}^{lm}, \forall u,v \in N^{S'} \setminus N^R, \forall k,l \in N^{S'} \setminus N^S, \forall (n,m) \in E^B \quad (3.25b)$$

Στόχος της αντικειμενικής συνάρτησης είναι αφ' ενός η ελαχιστοποίηση του κόστους ενσωμάτωσης του εικονικού αιτήματος στο υπόστρωμα και αφ' εταίρου η ελαχιστοποίηση των αλμάτων των μονοπατιών στο δίκτυο υποστρώματος στα οποία αντιστοιχίζονται οι ζεύξεις του εικονικού δικτύου.

Ο πρώτος στόχος καλύπτεται από τους δύο πρώτους όρους το αθροίσματος. Συγκεκριμένα ο πρώτος όρος αντιστοιχεί στο άθροισμα όλου του εύρους ζώνης που απαιτείται για τις ζεύξεις του αξιόπιστου εικονικού αιτήματος, κύριες και εφεδρικές. Ο δεύτερος όρος αντιστοιχεί στο σύνολο των υπολογιστικών πόρων που απαιτούν οι κόμβοι του αξιόπιστου εικονικού αιτήματος κύριοι και εφεδρικοί. Τα βάρη C_{uv} και D_w εξασφαλίζουν ότι θα προτιμηθούν οι

πόροι του δικτύου υποστρώματος με τη μικρότερη χρησιμοποίηση και ορίζονται ως οι αντίστροφες τιμές του διαθέσιμου εύρους ζώνης και διαθέσιμων υπολογιστικών πόρων αντίστοιχα. Η σταθερά δ λαμβάνει μικρή τιμή και έχει ως στόχο να αποφύγουμε τη διαίρεση με το 0.

Ο δεύτερος στόχος σχετίζεται με την εξασφάλιση ποιότητας παρεχόμενων υπηρεσιών καθώς σχετίζεται με τη μείωση της καθυστέρησης της επικοινωνίας και αντιστοιχεί στον τρίτο όρο του αθροίσματος. Η συσχέτιση αυτή προκύπτει από το γεγονός ότι σε κάθε βήμα εισάγεται κόστος επεξεργασίας και αναμονής στους ενδιάμεσους κόμβους. Το βάρος C_{uv} χρησιμοποιείται για να σχετίσει το μήκος της διαδρομής με το διαθέσιμο εύρος ζώνης.

Στη συνέχεια αναλύονται οι περιορισμοί του προβλήματος.

(3.10) - (3.13): Καθορίζεται το πεδίο ορισμού των μεταβλητών του προβλήματος

(3.14a),(3.14b): Είναι οι περιορισμοί για την διατήρηση της ροής.

(3.15a),(3.15b): Η απαιτούμενη χωρητικότητα τύπου $i \in I$ για τον εικονικό κόμβο τύπου $a \in A$ που αντιστοιχίζεται στον κόμβο υποστρώματος $w \in \mathrm{V}_{\mathrm{a}}^{S} \subseteq N^{S}$ δεν θα πρέπει να ξεπερνάει την αντίστοιχη διαθέσιμη χωρητικότητα του κόμβου υποστρώματος.

(3.16a),(3.16b): Η ροή (κύρια ή δευτερεύουσα αντίστοιχα) μίας εικονικής ζεύξης που δρομολογείται πάνω από μία ζεύξη του δικτύου υποστρώματος δεν μπορεί να υπερβαίνει την διαθέσιμη χωρητικότητα της ζεύξης του δικτύου υποστρώματος.

(3.17): Το σύνολο της ροής, κύρια και δευτερεύουσας, που δρομολογείται πάνω από μια ζεύξη του δικτύου υποστρώματος δεν μπορεί να υπερβαίνει την διαθέσιμη χωρητικότητα της ζεύξης του δικτύου υποστρώματος.

(3.18a) ,(3.18b): Το πολύ ένας εικονικός κόμβος αντιστοιχίζεται σε έναν κόμβο του δικτύου υποστρώματος, πρωτεύων ή εφεδρικός.

(3.19a) ,(3.19b): Ένας εικονικός κόμβος (πρωτεύων ή εφεδρικός) πρέπει να αντιστοιχηθεί σε κόμβο του δικτύου υποστρώματος της ίδιας κατηγορίας.

(3.20a),(3.20b),(3.20c): Ένας εικονικός κόμβος (πρωτεύων ή εφεδρικός) πρέπει να αντιστοιχιστεί σε έναν κόμβο του δικτυακού υποστρώματος.

Να σημειωθεί εδώ ότι στο επαυξημένο δικτυακό υπόστρωμα πρωτεύουσα ροή μπορεί να υπάρχει μόνο μεταξύ πρωτευόντων εικονικών κόμβων και κόμβων του υποστρώματος (όπως θα δούμε ότι εξασφαλίζεται στον επόμενο περιορισμό). Ωστόσο δευτερεύουσα ροή υπάρχει τόσο μεταξύ εφεδρικών εικονικών κόμβων και κόμβων του υποστρώματος όσο και μεταξύ πρωτευόντων εικονικών κόμβων, που είναι γειτονικοί με κρίσιμους κόμβους, και κόμβων του υποστρώματος.

(3.21): Δεν μπορεί να προέρχεται κύρια ροή από εφεδρικό εικονικό κόμβο.

(3.22): Ένας εικονικός κόμβος, που είναι γειτονικός κρίσιμου κόμβου και άρα άκρο εφεδρικής ζεύξης, θα πρέπει τόσο για την κύρια όσο και για τη δευτερεύουσα ροή να αντιστοιχίζεται στον ίδιο κόμβο του δικτύου υποστρώματος.

(3.23a),(3.24a),(3.23b),(3.24b): Η μεταβλητή x_{uv}^{nm} τίθεται κάθε φορά που κίνηση για την εικονική κύρια ζεύξη nm δρομολογείται μέσω της ζεύξης υποστρώματος uv ανεξάρτητα από την κατεύθυνση. Αντίστοιχα για την μεταβλητή y_{uv}^{nm} και κίνηση από εφεδρική εικονική ζεύξη.

(3.25a),(3.25b): Η λύση που προκύπτει από την αντιστοίχηση είναι ένας συνδεδεμένος γράφος.

Η παραπάνω μοντελοποίηση εξασφαλίζει ανθεκτικότητα σε σφάλματα μόνο σε τελικούς κόμβους, σε κόμβους δηλαδή από τους οποίους δεν διέρχεται κίνηση προς κάποιον άλλο κόμβο. Σε περίπτωση που θα θέλαμε να έχουμε εφεδρικούς ενδιάμεσους κόμβους (π.χ δρομολογητές, μεταγωγείς) θα έπρεπε να εξασφαλίσουμε ότι η κύρια και η δευτερεύουσα ροή από ζεύξεις στα άκρα τους περνάνε από ασύνδετα μονοπάτια. Η λύση μπορεί να επεκταθεί προσθέτοντας τον περιορισμό:

$x_{uv}^{mn} + y_{uv}^{b(m)n} \leq 1, \forall u, v \in N^{S'}, \forall\, n \in E^V, m \in V_a^V \cap C,$

α: εικονικός δρομολογητής ή μεταγωγέας (3.26)

3.2.3 Επίλυση Προβλήματος Μεικτού Ακέραιου Προγραμματισμού

Ο Μεικτός Ακέραιος Προγραμματισμός, αν και παρέχει μια ακριβή λύση του προβλήματος έχει εκθετικό υπολογιστικό κόστος και για μεγάλα εικονικά αιτήματα η λύση του είναι υπολογιστικά αδύνατή. Ως εκ τούτου ακολουθούμε τη μεθοδολογία που προτείνετε στο (Chowdhury, 2012) για να αναγάγουμε τον ΜΑΠ σε Ακέραιο Προγραμματισμό ΑΠ (Linear Programmin) που είναι πολυωνυμικής υπολογιστικής πολυπλοκότητας. Η μέθοδος που θα ακολουθήσουμε χωρίζει την ΕΕΔ σε δύο φάσεις. Αρχικά λύνουμε ένα πρόβλημα ΑΠ για την ενσωμάτωση των κόμβων και στη συνέχεια λύνουμε το πρόβλημα πολλαπλών διαδρομών (MCF) για την ενσωμάτωση των ζεύξεων.

Πιο συγκεκριμένα, προτείνουμε τους δύο ακόλουθους υποβέλτιστους αλγορίθμους:

3.2.3.1 Αντιστοίχιση εικονικών και εφεδρικών κόμβων σε ένα βήμα

Στον αλγόριθμο αυτό για να αναγάγουμε τον ΜΑΠ σε ΑΠ χαλαρώνουμε (relaxation) τις δύο ακέραιες μεταβλητές x_{uv}^{mn} και y_{uv}^{mn}επιτρέποντάς τους να λαμβάνουν πραγματικές τιμές στο διάστημα $[0,1]$. Στην συνέχεια λύνουμε το πρόβλημα ΑΠ όπως μοντελοποιείται στην παραπάνω ενότητα.

Κατόπιν οι τιμές που προκύπτουν για τις μεταβλητές x_{uv}^{mn} και y_{uv}^{mn} θα πρέπει να στρογγυλοποιηθούν. Για το λόγο αυτό χρησιμποποιείται η τεχνική της πιθανοτικής στρογγυλοποίησης που προτείνεται στο (Chowdhury, 2012), όπου η στρογγυλοποίηση γίνεται με κριτήριο τη μεγιστοποίηση των γινομένων $x_{uv}^{mn} f_{uv}^{mn}$ και $y_{uv}^{mn} g_{uv}^{mn}$ αντίστοιχα για την εκχώρηση πρώτα των κόμβων του αρχικού εικονικού αιτήματος και στη συνέχεια των εφεδρικών κόμβων. Αν και στη φάση αυτή αντιστοιχίζουμε μόνο τους κόμβους, λαμβάνοντας υπόψιν τη ροή πάνω στις ζεύξεις του υποστρώματος, όπως υπολογίζεται από τον ΑΠ, παρέχουμε μια λύση πιο κοντά στη βέλτιστη απ' ότι αν η εκχώρηση κόμβων γινόταν με κάποιο άπληστο τρόπο.

Στη συνέχεια επιλύουμε μια εκδοχή του προβλήματος πολλαπλών ροών (multi-commodity flow allocation) για την ταυτόχρονη αντιστοίχιση των ζεύξεων του αρχικού εικονικού αιτήματος και των εφεδρικών ζεύξεων. Η επέκταση της κλασικής μοντελοποίησης σε αυτή είναι αρκετά απλή.

Αρχικά τροποποιούμε την αντικειμενική συνάρτηση ώστε να ελαχιστοποιεί το εύρος ζώνης που δεσμεύεται τόσο για την κύρια όσο και για τη δευτερεύουσα ροή:

$$\min(\sum_{uv \in E^S}(\sum_{mn \in E^V} fi_{uv}^{mn} + \sum_{mn \in E^B} gi_{uv}^{mn}) \qquad 3.27$$

Όπου $fi_{uv}^{mn}, gi_{uv}^{mn}$ οι μεταβλητές του προβλήματος ΓΠ:

fi_{uv}^{mn}: η ροή της εικονικής ζεύξης $mn \in E^V$ που δρομολογείται μέσω της ζεύξης του δικτυακού υποστρώματος $uv \in E^S$

gi_{uv}^{mn}: η ροή της εικονικής ζεύξης $mn \in E^V$ που δρομολογείται μέσω της ζεύξης του δικτυακού υποστρώματος $uv \in E^S$

Στη συνέχεια προσθέτουμε στους αρχικούς περιορισμούς για τη διατήρηση της κύριας ροής τους περιορισμούς για τη δευτερεύουσα ροή:

$$\sum_{w \in N^S} fi_{uv}^{mn} - \sum_{w \in N^S} fi_{vu}^{mn} = 0, \forall mn \in E^V, \forall u \in N^S \{o^{mn}, d^{mn}\}$$

$$\sum_{w \in N^S} fi_{uv}^{mn} - \sum_{w \in N^S} fi_{vu}^{mn} = b(m,n), \forall mn \in E^V, u \equiv o^{mn} \in N^S$$

$$\sum_{w \in N^S} fi_{uv}^{mn} - \sum_{w \in N^S} fi_{vu}^{mn} = -b(m,n), \forall mn \in E^V, u \equiv d^{mn} \in N^S \qquad (3.28a)$$

$$\sum_{w \in N^S} gi_{uv}^{mn} - \sum_{w \in N^S} gi_{vu}^{mn} = 0, \forall mn \in E^B, \forall u \in N^S \{o^{mn}, d^{mn}\}$$

$$\sum_{w \in N^S} gi_{uv}^{mn} - \sum_{w \in N^S} gi_{vu}^{mn} = b(m,n), \forall mn \in E^B, u \equiv o^{mn} \in N^S$$

$$\sum_{w \in N^S} gi_{uv}^{mn} - \sum_{w \in N^S} gi_{vu}^{mn} = -b(m,n), \forall mn \in E^B, u \equiv d^{mn} \in N^S$$

(3.28a)

όπου o^{mn}, d^{mn} οι κόμβοι στο δικτυακό υπόστρωμα στους οποίους αντιστοιχίζονται τα άκρα m, n της ζεύξης του αξιόπιστου εικονικού αιτήματος $mn \in E^R$

Τέλος τροποποιείται κατάλληλα και ο περιορισμός για τη χωρητικότητα των ζεύξεων:

$\sum_{mn \in E^V}(fi_{uv}^{mn} + fi_{vu}^{mn}) + \sum_{mn \in E^V}\left(gi_{uv}^{mn} + gi_{vu}^{mn}\right) \leq BW(u,v), \forall u, v \in N^S$ (3.29)

3.2.3.2 Αντιστοίχιση εικονικών και εφεδρικών κόμβων σε δύο βήματα

Θέλοντας μειώσουμε περαιτέρω την υπολογιστική πολυπλοκότητα θα μειώσουμε τις μεταβλητές του ΑΠ από 4 σε 2 λύνοντας το πρόβλημα ξεχωριστά για την αντιστοίχιση των κόμβων του αρχικού εικονικού αιτήματος και την αντιστοίχιση των εφεδρικών κόμβων, ακολουθώντας τον αλγόριθμο που προτείνεται στο (Yu, 2013). Πιο συγκεκριμένα, στο πρώτο βήμα, λύνουμε το πρόβλημα της ΕΕΔ χωρίς να λαμβάνουμε υπόψιν την ανοχή σε σφάλματα. Στο βήμα αυτό υιοθετούμε τον αλγόριθμο που εισάγεται στο (Papagianni, 2013) για ΕΕΔ με ετερογενείς πόρους.

Στο δεύτερο βήμα, έχοντας βρει τις τιμές για τις μεταβλητές x_{uv}^{mn}, f_{uv}^{mn} λύνουμε το πρόβλημα που μοντελοποιείται στην προηγούμενη ενότητα για τις μεταβλητές y_{uv}^{mn}, g_{uv}^{mn}, χαλαρώνοντας τηςν g_{uv}^{mn} όπως περιγράφηκε παραπάνω.

Για την αντιστοίχιση των ζεύξεων, έχοντας ήδη αντιστοιχίσει τις κύριες ζεύξεις του εικονικού αιτήματος από το πρώτο βήμα, λύνουμε το πρόβλημα πολλαπλών ροών για την αντιστοίχιση των εφεδρικών μόνο ζεύξεων.

3.3 Ενσωμάτωση εικονικού αιτήματος με ανοχή σε σφάλματα σε κόμβους και ζεύξεις.

Όπως είδαμε στο κεφάλαιο 2, δύο είναι οι προσεγγίσεις για την εξασφάλιση ανοχής σε σφάλματα σε ζεύξεις: (i) η επαναδρομολόγηση της κίνησης μέσω κάποιας ασύνδετης

διαδρομής, είτε για όλο το μονοπάτι, είτε για τη ζεύξη που έπεσε και (ii) η μετακίνηση (migration) κάποιου από τους ακραίους κόμβους σε θέση από την οποία υπάρχει μονοπάτι για να δρομολογηθεί η κίνηση.

Γενικά η μετακίνηση κόμβων θα είχε το μειονέκτημα της δέσμευσης εφεδρικών κόμβων. Ωστόσο έχοντας ήδη δεσμεύσει εφεδρικούς κόμβους για να εξασφαλίσουμε ανοχή σε σφάλματα σε κόμβους δεν εισάγουμε επιπλέον κόστος στο εικονικό αίτημα. Αυτό που θα πρέπει ωστόσο να εξασφαλιστεί είναι ότι τα μονοπάτια της κύριας και δευτερεύουσας ροής θα είναι ασύνδετα για να είναι δυνατή η επαναδρομολόγηση.

Έτσι κρατώντας την μοντελοποίηση που έχουμε ως τώρα, αρκεί να τροποποιήσουμε τον περιορισμό (3.26) ώστε να ισχύει για κάθε κόμβο του εικονικού αιτήματος:

$$x_{uv}^{mn} + y_{uv}^{b(m)n} \leq 1, \forall u, v \in N^{S'}, \forall\, (mn) \in E^{V}, m \in C \qquad (3.30)$$

Έχοντας ένα σύνολο κρίσιμων κόμβων, οι οποίοι καλύπτονται από εφεδρικούς, η λύση αυτή, συνακόλουθα, καλύπτει σφάλματα σε ζεύξεις που πρόσκεινται στους κρίσιμους κόμβους. Ανάγεται δε σε ανεκτική για σφάλματα σε όλες τις ζεύξεις αν το σύνολο των κρίσιμων κόμβων ταυτιστεί με το σύνολο των κόμβων του εικονικού αιτήματος.

Θα πρέπει να σημειωθεί, ακόμα, ότι η πολιτική της μετακίνησης των κόμβων εισάγει πέραν της καθυστέρησης για την επαναδρομολόγηση της κίνησης και καθυστέρηση λόγω της μετακίνησης του κόμβου. Μια τέτοια μετακίνηση θα πρέπει να ανήκει στην κατηγορία των μετακινήσεων πραγματικού χρόνου (hot/live migration) αφού δεν θα πρέπει να γίνει αντιληπτή στο χρήστη.

Σύμφωνα με το (Voorsluys, 2009) τόσο στον κόμβο προορισμού όσο και στον κόμβο προέλευσης θα απαιτούνται επιπλέον κύκλοι της ΚΜΕ για τη μεταφορά των αρχείων από τον έναν στον άλλο κόμβο με αποτέλεσμα ο χρήστης να αντιλαμβάνεται μια επιβράδυνση του κυμαίνεται από 1% έως 8%, ενώ ο χρόνος που ο κόμβος δεν θα είναι διαθέσιμος κυμαίνεται από 60 ms μέχρι 3s.

Για την επίλυση του ΜΑΠ θα πρέπει να λάβουμε υπόψιν των περιορισμό των ασύνδετων μονοπατιών. Ο περιορισμός (3.27) δεν εξασφαλίζει ότι οι προσεγγιστικοί αλγόριθμοι που περιγράφηκαν στις ενότητες 3.3.1 και 3.3.2 θα πραγματοποιήσουν την αντιστοίχιση των ζεύξεων με τρόπο που να εξασφαλίζει ασύνδετα μονοπάτια για την κύρια και δευτερεύουσα ροή της ίδιας εικονικής ζεύξης αφού αφορά τη φάση της αντιστοίχισης των κόμβων. Θα πρέπει λοιπόν να τροποποιήσουμε την επίλυση του προβλήματος πολλαπλών ροών.

Αναλυτικότερα, θα ακολουθήσουμε την ιδέα που προτείνεται στο (Rhaman, 2013). Οι δύο αλγόριθμοι θα τροποποιηθούν ως εξής: Αρχικά λύνεται το πρόβλημα πολλαπλών ροών για την κύρια ροή. Στη συνέχεια κατασκευάζουμε ένα λογικό πίνακα $\varphi_{uv}^{kn}, \forall kn \in E^{B}, \forall u, v \in N^{S}$. Ο πίνακας αυτός λαμβάνει τιμή $1\,(true)$ αν από τη ζεύξη μεταξύ των κόμβων

υποστρώματος u, v διέρχεται ροή της εικονικής ζεύξης mn όπου $k = b(m)$. Κατόπιν λύνουμε το πρόβλημα πολλαπλών ροών εισάγοντας επιπλέον τον περιορισμό να μην δρομολογείται ροή της εφεδρικής ζεύξης kn πάνω από τη ζεύξη υποστρώματος uv αν $\varphi_{uv}^{kn} = 1$.

$$gi_{uv}^{mn} \leq BW(u, v) \times \varphi_{uv}^{mn}, \forall mn \in E^B, u, v \in E^S \quad (3.31)$$

4

Αξιολόγηση

Στο κεφάλαιο αυτό παρουσιάζονται τα αποτελέσματα στα πειράματα που πραγματοποιήθηκαν για να συγκρίνουμε τους αλγορίθμους ΕΕΔ με ανοχή σε σφάλματα που προτάθηκαν παραπάνω. Επίσης αντιπαραβάλουμε τους παραπάνω αλγορίθμους με τον αλγόριθμο ΕΕΔ χωρίς ανοχή σε σφάλματα.

Πιο συγκεκριμένα, συγκρίνουμε τους εξής αλγορίθμους:

- NCM: Ο αλγόριθμος ΕΕΔ για δίκτυο με ετερογενείς κόμβους όπως προτείνεται στο (Papagianni, 2013) χωρίς το γνώρισμα της παροχής ποιότητας υπηρεσίας.
- oneStep: Ενσωμάτωση εφεδρικών κόμβων και κόμβων εικονικού αιτήματος σε ένα βήμα, είτε με πλεονάζουσα-κατά-1 είτε με πλεονάζουσα-κατά-k πολιτική για τους εφεδρικούς κόμβους και στη συνέχεια επίλυση του προβλήματος πολλαπλών διαδρομών (MCF) από κοινού για την εφεδρική και κύρια ροή. Ο αλγόριθμος εξασφαλίζει ανοχή σε μεμονωμένα με την πλεονάζουσα-κατά-1 πολιτική και σε πολλαπλά με την πλεονάζουσα-κατά-k πολιτική σφάλματα σε σύνολο κρίσιμων κόμβων.
- twoStep: Ενσωμάτωση εφεδρικών κόμβων και κόμβων εικονικού αιτήματος σε 2 βήματα, είτε με πλεονάζουσα-κατά-1 είτε με πλεονάζουσα-κατά-k πολιτική για τους εφεδρικούς κόμβους και στη συνέχεια επίλυση του προβλήματος πολλαπλών διαδρομών (MCF) ξεχωριστά για την εφεδρική και κύρια ροή. Ο αλγόριθμος εξασφαλίζει ανοχή σε μεμονωμένα με την πλεονάζουσα-κατά-1 πολιτική και σε

πολλαπλά με την πλεονάζουσα-κατά-k πολιτική σφάλματα σε σύνολο κρίσιμων κόμβων.

- oneSepDisjPaths: Ενσωμάτωση εφεδρικών κόμβων και κόμβων εικονικού αιτήματος σε ένα βήμα με πλεονάζουσα-κατά-1 πολιτική για τους εφεδρικούς κόμβους και στη συνέχεια επίλυση του προβλήματος πολλαπλών διαδρομών (MCF) ξεχωριστά για την εφεδρική και κύρια ροή, εξασφαλίζοντας ασύνδετα μονοπάτια μεταξύ της κύριας και δευτερεύουσα ροής για την ίδια εικονική ζεύξη. Ο αλγόριθμος εξασφαλίζει ανοχή σε μεμονωμένα σε σύνολο κρίσιμων κόμβων και στις προσκείμενες σε αυτούς ζεύξεις με μετακίνηση κόμβων (node migration).
- twoStepDisjPaths: Ενσωμάτωση εφεδρικών κόμβων και κόμβων εικονικού αιτήματος σε δύο βήματα, με πλεονάζουσα-κατά-1 πολιτική για τους εφεδρικούς κόμβους και στη συνέχεια επίλυση του προβλήματος πολλαπλών διαδρομών (MCF) ξεχωριστά για την εφεδρική και κύρια ροή, εξασφαλίζοντας ασύνδετα μονοπάτια μεταξύ της κύριας και δευτερεύουσα ροής για την ίδια εικονική ζεύξη. Ο αλγόριθμος εξασφαλίζει ανοχή σε μεμονωμένα σφάλματα σε σύνολο κρίσιμων κόμβων και στις προσκείμενες σε αυτούς ζεύξεις με μετακίνηση κόμβων (node migration).

4.1 Παράμετροι αξιολόγησης

Προκειμένου να συγκρίνουμε τους παραπάνω αλγορίθμους εξετάζουμε τους εξής δείκτες:

- Ποσοστό αποδοχής εικονικών αιτημάτων: Η ενσωμάτωση ενός εικονικού αιτήματος μπορεί να αποτύχει είτε επειδή δεν βρέθηκαν οι απαιτούμενοι υπολογιστικοί πόροι στο δικτυακό υπόστρωμα στη φάση αντιστοίχισης των κόμβων, είτε επειδή δεν υπάρχει το απαιτούμενο εύρος ζώνης στα μονοπάτια που πρέπει να αντιστοιχηθούν οι εικονικές ζεύξεις, σύμφωνα με τους περιορισμούς που θέτουμε (ασύνδετα ή όχι). Επιπλέον η ενσωμάτωση του εικονικού αιτήματος μπορεί να αποτύχει επειδή ο αλγόριθμος δεν βρήκε λύση στο πρόβλημα σε ένα χρονικό όριο που εμείς θέτουμε. Έτσι είναι σημαντικό να δούμε για τι ποσοστό των αιτήσεων που έρχονται σε πραγματικό χρόνο επιτυγχάνεται τελικά η αντιστοίχιση. Το ποσοστό αποδοχής ορίζεται ως ο λόγος των αιτήσεων που αντιστοιχίζονται προς τον συνολικό αριθμό των αιτήσεων.
- Μέσο κέρδος: Το κέρδος που έχει ο Πάροχος Υποδομής (InP) από την ενσωμάτωση ενός εικονικού μπορεί να σχετιστεί με την ποσότητα :

$$R(G^V) = \sum_{e^V \in E^V} b(e^V) + \sum_{a \in A} \sum_{n^V \in N_a^V} \sum_{i \in I} c_i(n^V) \qquad (4.1)$$

όπου δεν λαμβάνουμε υπόψιν μας τους εφεδρικούς πόρους που δεσμεύονται γιατί το κέρδος σχετίζεται με το εικονικό δίκτυο που είναι σε λειτουργία. Το μέσο κέρδος

υπολογίζεται ως το άθροισμα του κέδρους από τα εικονικά αιτήματα που αντιστοιχίζονται προς τον συνολικό αριθμό εικονικών αιτημάτων.

- Μέσο κόστος εικονικού αιτήματος: Το κέρδος από μόνο του δεν είναι αρκετό αν δεν γνωρίζουμε το κόστος που έχει ο Πάροχος Υποδομής για την ενσωμάτωση ενός εικονικού αιτήματος. Ορίζουμε λοιπόν την ποσότητα:

$$C_{TOTAL}(G^V) = \sum_{e^R \in E^V} \sum_{e^S \in E^S} f_{e^S}^{e^R} + \sum_{a \in A} \sum_{n^R \in N_a^R} \sum_{i \in I} c_i(n^R) \qquad (4.2)$$

όπου λαμβάνουμε πλέον υπόψιν και τους εφεδρικούς πόρους, ενώ $f_{e^S}^{e^R}$ ορίζουμε το συνολικό εύρος ζώνης που εκχωρείται στην εικονική ζεύξη e^R πάνω στη ζεύξη υποστρώματος e^S. Για το μέσο κόστος εικονικού αιτήματος λαμβάνουμε το μέσο όρο του κόστους των εικονικών αιτημάτων που εκχωρήθηκαν επιτυχώς στο υπόστρωμα, σταθμισμένο με τον αριθμό των εικονικών αιτημάτων ανάλογα με τον αριθμό των κόμβων του αιτήματος. Ο μέσος όρος είναι σταθμισμένος για να μην επηρεαστεί ο δείκτης από τη συμπεριφορά του αλγορίθμου ως προς το μέγεθος του εικονικού αιτήματος.

Θέλοντας επιπλέον να εξετάσουμε την κλιμακοσιμότητα των αλγορίθμων εξετάζουμε τους παρακάτω δείκτες σε συνάρτηση με τον αριθμό των κόμβων του εικονικού αιτήματος.

- Κόστος του λειτουργούντος αιτήματος:

$$C_W(G^V) = \sum_{e^N \in E^V} \sum_{e^S \in E^S} f_{e^S}^{e^V} + \sum_{a \in A} \sum_{n^V \in N_a^V} \sum_{i \in I} c_i(n^V) \qquad (4.3)$$

- Κόστος για την ενσωμάτωση των κόμβων του εικονικού αιτήματος:

$$C_N(G^V) = \sum_{a \in A} \sum_{n^R \in N_a^R} \sum_{i \in I} c_i(n^R) \qquad (4.4)$$

- Κόστος σε εύρος ζώνης που απαιτείται για την ενσωμάτωση εικονικών ζέυξεων, όπου η δεύτερη ποσότητα ορίζεται ως:

$$C_L(G^V) = \sum_{e^R \in E^V} \sum_{e^S \in E^S} f_{e^S}^{e^R} \qquad (4.5)$$

4.2 Σύστημα αξιολόγησης

Για την αξιολόγηση των προτεινόμενων αλγορίθμων χρησιμοποιήθηκε προσομοιωτής διακριτών γεγονότων (event driven simulator) η ανάπτυξη του οποίου έγινε σε Java, ο Προσομοιωτής για Έλεγχο Εικονικών Υποδομών (Simulator for Controlling Virtual Infrastructures – CVI-Sim). Ο προσομοιωτής δρα ως εξομοιωτής Υπηρεσίας Αντιστοίχισης Πόρων. Επιτρέπει τη δημιουργία τυχαίου συνόλου αιτήσεων σύμφωνα με προδιαγραφές που επιλέγει ο χρήστης για την πιθανότητα ένας κόμβος να ανήκει σε μία από τις κατηγορίες διακομιστής, δρομολογητής ή μεταγωγέας και για την πιθανότητα σύνδεσης δύο κόμβων. Επιπλέον, ο χρήστης μπορεί να επιλέξει την συνάρτηση κατανομής πιθανότητας που

ακολουθούν οι αφίξεις των εικονικών αιτημάτων καθώς και η διάρκεια ζωής των αιτημάτων. Όμοια επιτρέπει τη δημιουργία δικτυακού υποστρώματος από τυχαίο σύνολο κόμβων με δυνατότητα πάλι καθορισμού της πιθανότητας του κόμβου να ανήκει σε μία από τις προαναφερθείσες κατηγορίες και της πιθανότητας σύνδεσής τους.

Αναφέρουμε ότι ο προσομοιωτής δίνει τη δυνατότητα δημιουργίας εικονικών αιτημάτων και δικτυακού υποστρώματος τόσο με γραφικό τρόπο, όσο και με είσοδο από αρχείο προδιαγραφών πραγματικών υποδομών. Οι δύο τελευταίες αυτές δυνατότητες δεν χρησιμοποιήθηκαν ωστόσο για την αξιολόγηση των προτεινόμενων αλγορίθμων.

Ο προσομοιωτής δίνει επίσης τη δυνατότητα στο χρήστη να καθορίσει μια σειρά από λειτουργικά και μη λειτουργικά χαρακτηριστικά γνωρίσματα των κόμβων και ζεύξεων. Συγκεκριμένα μπορεί να καθορίσει για ένα κόμβο το λειτουργικό σύστημα (π.χ. Linux, Windows), το υποστηριζόμενο περιβάλλον εικονικοποίησης (π.χ. VMware, Xen), την υποστηριζόμενη στοίβα δικτυακών προτοκόλλων (π.χ. TCP/IP) αλλά και την υπολογιστική χωρητικότητα της K.M.E, τη μνήμη, τον διαθέσιμο αποθηκευτικό χώρο, τον διαθέσιμο αριθμό δικτυακών διεπαφών και τον αριθμό υποστηριζόνμενων εικονικών μηχανημάτων.

Για την ανάπτυξη του λογισμικού του προσομοιωτή έγινε χρήση της βιβλιοθήκης JUNG (JUNG, 2014) που παρέχει μια απλή και επεκτάσιμη γλώσσα για τη μοντελοποίηση, ανάλυση και εικονικοποίηση δεδομένων με τη μορφή γράφου. Τέλος για η επίλυση των προβλημάτων ΓΠ έγινε με χρήση της βιβλιοθήκης CPLEX (CPLEX, 2014).

4.3 Οργάνωση πειραμάτων

Οι τοπολογίες στις οποίες εξετάσαμε την επίδοση των προτεινόμενων αλγορίθμων είναι ετερογενείς και αποτελούνται από διακομιστές και δρομολογητές. Για είναι ρεαλιστικές οι συνθήκες αξιολόγησης χρησιμοποιήθηκαν πολυεστιακές (multihoming) τοπολογίες, δηλαδή τοπολογίες στις οποίες δεν υπάρχουν ζεύξεις απευθείας μεταξύ διακομιστών αλλά μόνο μεταξύ δρομολογητών και μεταξύ δρομολογητών και διακομιστών.

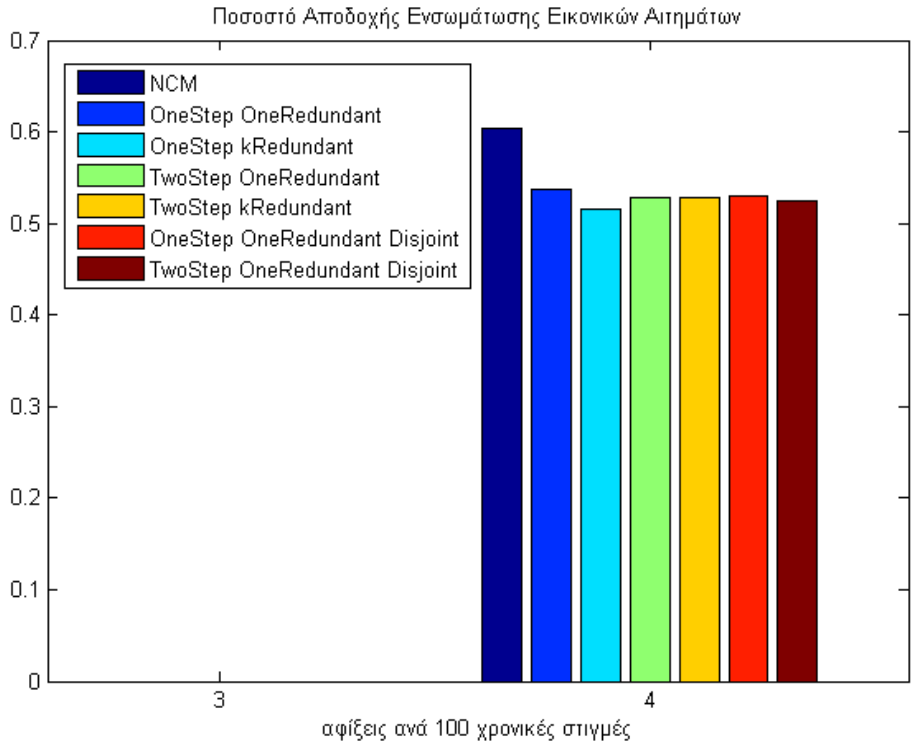
Οι κόμβοι έχουν διαφορετικά μη λειτουργικά χαρακτηριστικά γνωρίσματα, σύμφωνα με τον τύπο τους. Συγκεκριμένα, ακολουθώντας το (Papagianni, 2013), όσον αφορά το υπόστρωμα, οι διακομιστές έχουν ως μη λειτουργικά χαρακτηριστικά γνωρίσματα χωρητικότητα K.M.E., μνήμη και αποθηκευτικό χώρο τιμές των οποίων είναι ακέραιοι αριθμοί ομοιόμορφα κατανεμημένοι στο διάστημα [50-100]. Το εύρος ζώνης των ζεύξεων του δικτυακού υποστρώματος λαμβάνει επίσης ακέραιες τιμές στο διάστημα [50-100]. Κάθε δρομολογητής του υποστρώματος μπορεί να φιλοξενήσει μέχρι 15 εικονικούς δρομολογητές. Οι κόμβοι του υποστρώματος παράγονται τυχαία και έχουν πιθανότητα σύνδεσης 50% ενώ ένας κόμβος έχει 20% πιθανότητα να είναι δρομολογητής και 80% πιθανότητα να είναι διακομιστής.

Αντίστοιχα για τα εικονικά αιτήματα η χωρητικότητα της Κ.Μ.Ε, η μνήμη και ο αποθηκευτικός χώρος λαμβάνουν ακέραιες τιμές ομοιόμορφα κατανεμημένες στο διάστημα [0-20]. Το εύρος ζώνης των εικονικών ζεύξεων κατανέμεται επίσης ομοιόμορφα στις ακέραιες τιμές του διαστήματος [0-20]. Ο αριθμός των εικονικών κόμβων είναι ομοιόμορφα κατανεμημένος στο διάστημα [2-10]. Οι εικονικοί κόμβοι συνδέονται με πιθανότητα 30% έχουν 10% πιθανότητα να είναι δρομολογητές και 90% πιθανότητα να είναι εικονικά μηχανήματα (Virtual Machine – VM) . Επίσης, ένα εικονικό μηχάνημα ανήκει με 30% πιθανότητα στο σύνολο των κρίσιμων κόμβων ενώ στα πλαίσια των πειραμάτων δεν δημιουργούμε κρίσιμους εικονικούς δρομολογητές, αν και πρέπει να σημειωθεί ότι οι αλγόριθμοι υποστηρίζουν την ΕΕΔ με κρίσιμους δρομολογητές.

Τέλος, οι αφίξεις εικονικών αιτημάτων ακολουθούν κατανομή Poisson με ρυθμό άφιξης 4 εικονικά αιτήματα ανά 100 χρονικές στιγμές. Τέλος, τα πλαίσια των πειραμάτων έγιναν προσομοιώσεις με 1000 εικονικά αιτήματα και 4 πειράματα για κάθε αγλόριθμο ενώ προκειμένου να έχουμε σαφή σύγκριση των αλγορίθμων για κάθε πείραμα τρέξαμε τους αλγορίθμους για το ίδιο δικτυακό υπόστρωμα και με το ίδιο σύνολο εικονικών αιτήσεων.

4.4 Αποτελέσματα

Στα σχήματα 4.1 - 4.4 συγκρίνουμε τους τέσσερις αλγορίθμους ΕΕΔ με ανοχή σε σφάλματα μεταξύ τους και με τον NCM αλγόριθμο που προτάθηκε στο (Papagianni, 2013). Για κάθε έναν από τους τέσσερις αλγορίθμους ΕΕΔ με ανοχή σε σφάλματα σε κόμβους εξετάζουμε τόσο την εκδοχή με πλεονάζουσα_κατά_1 (oneRedundant) όσο και με πλεονάζουσα_κατά_k (kRedundant) πολιτική για τους εφεδρικούς κόμβους. Για τους αλγορίθμους με ανοχή και στις προσκείμενες ζεύξεις στους κρίσιμους κόμβους εξετάζουμε την εκδοχή με πλεονάζουσα_κατά_1 (oneRedundant) πολιτική.



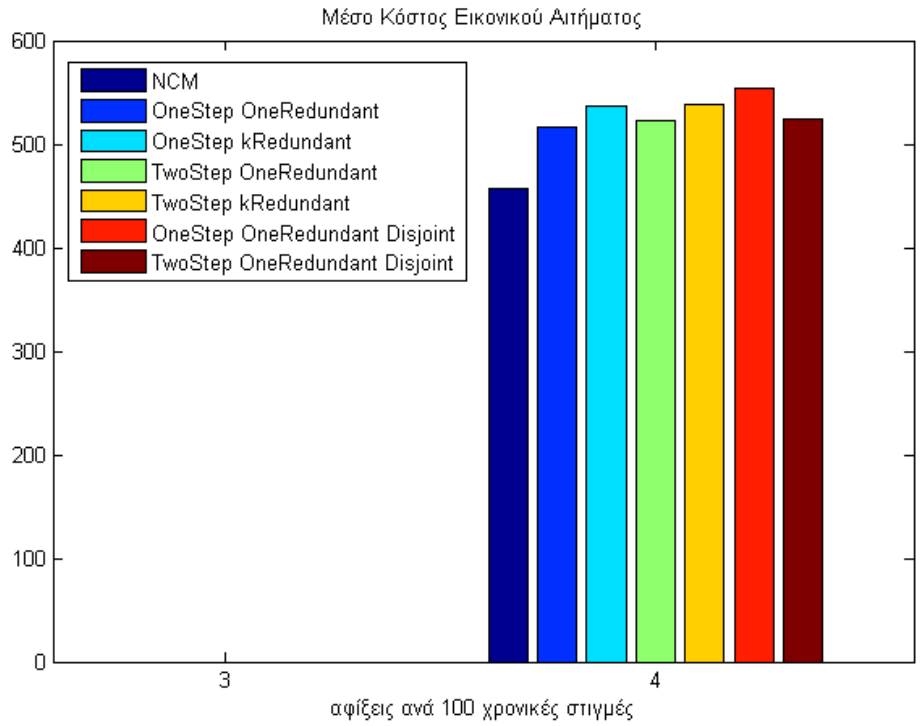
ΣΧΗΜΑ 4. 8 Ποσοστό αποδοχής εικονικών αιτημάτων

Όπως ήταν αναμενόμενο το ποσοστό του NCM αλγορίθμου είναι μεγαλύτερο αφού οι υπόλοιποι αλγόριθμοι ενσωματώνουν επαυξημένα εικονικά αιτήματα με εφεδρικούς κόμβους και ζεύξεις και άρα έχουν μεγαλύτερες απαιτήσεις σε εύρος ζώνης και υπολογιστικούς πότους.

Παρατηρούμε επίσης ότι τα ποσοστά αποδοχής των αλγορίθμων με πλεονάζουσα_κατά_k πολιτική είναι μικρότερα από αυτά του αντίστοιχου αλγορίθμου με πλεονάζουσα_κατά_1 πολιτική, γεγονός που οφείλεται στους επιπλέον εφεδρικούς κόμβους που δεσμεύονται (ένας για κάθε κύριο) και στις επιπλέον εφεδρικές ζεύξεις που απαιτούνται. Βλέπουμε ακόμη ότι οι αλγόριθμοι με ενσωμάτωση κόμβων σε ένα βήμα πετυχαίνουν γενικά καλύτερα ποσοστά αποδοχής σε σχέση με αυτά των αλγορίθμων με ενσωμάτωση κόμβων σε δύο βήματα για την αντίστοιχη πολιτική. Αυτό οφείλεται στο ότι οι δεύτεροι αποτελούν υποβέλτιστη λύση αφού δεν αντιμετωπίζουν το πρόβλημα συνολικά. η αντιστοίχιση στο πρώτο βήμα των κόμβων και συνακόλουθα των ζεύξεων δεν λαμβάνει υπόψιν της την ενσωμάτωση των εφεδρικών πόρων στη συνέχεια. Να σημειωθεί εδώ ότι το ποσοστό αποδοχής για ενσωμάτωση κόμβων σε ένα βήμα με πλεονάζουσα_κατά_k πολιτική παρουσιάζεται χαμηλότερο από το αναμενόμενο σε σχέση με την περίπτωση ενσωμάτωσης κόμβων σε 2 βήματα. Εδώ θα πρέπει να ληφθεί υπόψιν το μεγάλο υπολογιστικό κόστος του αλγορίθμου που σε συνδυασμό με το μεγάλο μέγεθος των αξιόπιστων εικονικών αιτημάτων έχει σαν αποτέλεσμα μεγάλο χρόνο εκτέλεσης ανά αίτημα. Επειδή όπως αναφέρθηκε στην ενότητα 4.1 είχε τεθεί χρονικό όριο (16 min) για

την αντιστοίχιση του αιτήματος, αυτό θα είχε ως αποτέλεσμα την αποτυχία εικονικών αιτημάτων λόγω υπέρβασης του χρονικού ορίου.

Τέλος, δεν παρατηρείται σημαντική διαφορά στα ποσοστά αποδοχής των αλγορίθμων που εξασφαλίζουν ασύνδετα μονοπάτια. Αν και όπως θα δούμε παρακάτω έχουν μεγαλύτερο κόστος ενσωμάτωσης, αναγκάζοντας την κύρια και εφεδρική ροή να διαχωρίζονται σε διαφορετικά μονοπάτια πετυχαίνουν καλύτερη ισοκατανομή φορτίου στις ζεύξεις.

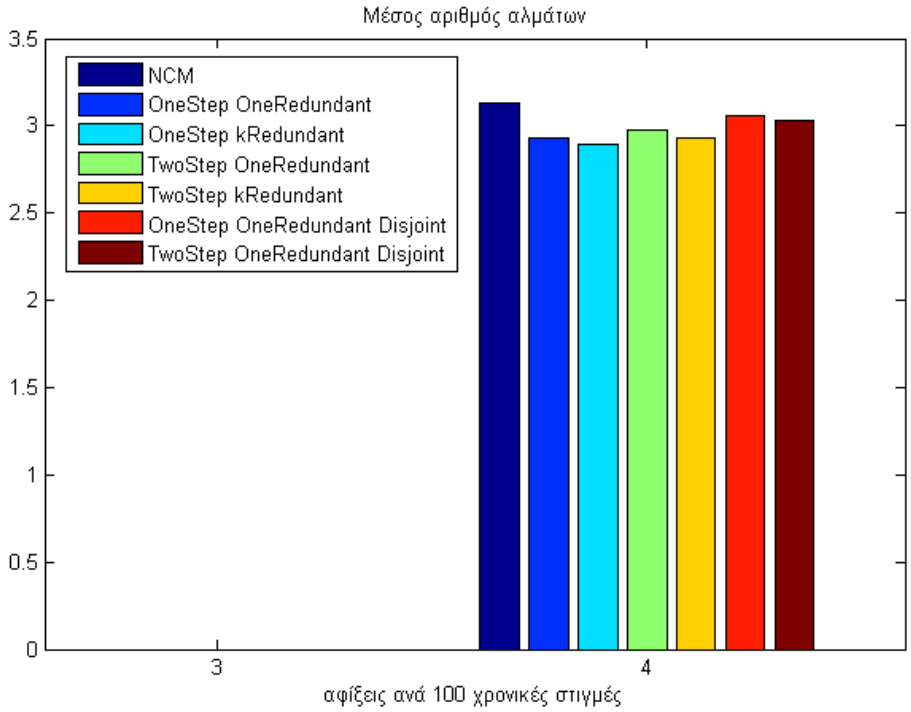


ΣΧΗΜΑ 4. 9 Μέσο Κόστος Εικονικού Αιτήματος

Όπως ήταν και πάλι αναμενόμενο το κόστος του NCM αλγορίθμου είναι μικρότερο καθώς δε περιλαμβάνει εφεδρικούς πόρους. Επιπλέον, το κόστος των αλγορίθμων με πλεονάζουσα_κατά_1 πολιτική είναι μικρότερο από των αντιστοίχων με πλεονάζουσα_κατά_k λόγω των επιπλέον εφεδρικών πόρων που δεσμεύουν οι δεύτεροι. Στους τέσσερις πρώτους αλγορίθμους με ανοχή σε σφάλματα παρουσιάζεται επίσης ελαφρώς μικρότερο κόστος σε αυτούς με ενσωμάτωση σε ένα βήμα, όπως περιμέναμε, λόγω της αντικειμενικής συνάρτησης της φάσης τόσο της ενσωμάτωσης κόμβων όσο και ενσωμάτωση ζεύξεων που λαμβάνει υπόψιν το πρόβλημα συνολικά στη δεύτερη περίπτωση.

Παράδοξο εμφανίζεται το κόστος ενσωμάτωσης για τον αλγόριθμο ενσωμάτωσης σε ένα βήμα με ανεκτικότητα σε σφάλματα σε ζεύξεις σε σχέση με τον αντίστοιχο σε δύο βήματα. Η εξήγηση που δίνουμε για το φαινόμενο είναι ότι ενώ η ενσωμάτωση των κόμβων γίνεται σε ένα βήμα, η ενσωμάτωση των ζεύξεων γίνεται σε δύο βήματα, έτσι ώστε όπως περιεγράφηκε στην ενότητα 3 να ληφθεί υπόψιν ο περιορισμός των ασύνδετων μονοπατιών. Έτσι αν και

στην πρώτη φάση η ενσωμάτωση των κόμβων έχει γίνει με κριτήριο και την κύρια και τη δευτερεύουσα ροή, στο επόμενο βήμα η κύρια ροή εγκαθίσταται άπληστα αναιρώντας το προηγούμενο βήμα. Ο συνδυασμός αυτός έχει όπως θα δούμε και στα ακόλουθα σχήματα σαν αποτέλεσμα αυξημένο μέσο αριθμό αλμάτων και συνακόλουθα αυξημένο κόστος ενσωμάτωσης ζεύξεων. Επιπλέον, το αυξημένο ποσοστό ενσωμάτωσης του αλγόριθμου με ενσωμάτωση σε ένα βήμα έχει ως αποτέλεσμα την αύξηση χρησιμοποίησης των πόρων του υποστρώματος. Αυτό οδηγεί με τη σειρά του στην αύξηση του μήκους μονοπατιών και κατ' επέκταση σε αυξημένο κόστος.

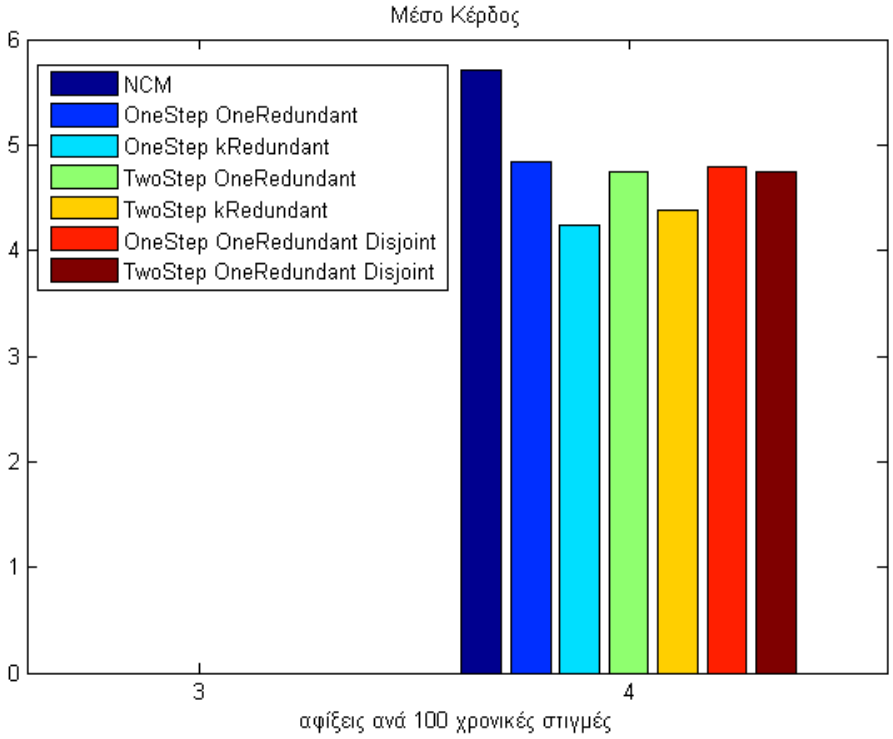


ΣΧΗΜΑ 4. 10 Μέσος αριθμός αλμάτων

Αρχικά παρατηρούμε ότι ο μέσος αριθμός αλμάτων του NCM αλγορίθμου προκύπτει μεγαλύτερος. Αυτό συμβαίνει διότι στους υπολοίπους αλγορίθμους λαμβάνονται υπόψιν και τα άλματα για τις εφεδρικές ζεύξεις. Έτσι ένας εφεδρικός κόμβος μπορεί να βρεθεί κοντύτερα στον γειτονικό κόμβο του κρίσιμου κόμβου που καλύπτει απ' ότι ο αρχικός, μειώνοντας έτσι τον μέσο αριθμό αλμάτων.

Παρατηρούμε ακόμη ότι η πλεονάζουσα_κατά_k πολιτική έχει μικρότερο μέσο αριθμό αλμάτων. Αυτό συμβαίνει γιατί στην πλεονάζουσα_κατά_1 πολιτική ο μοναδικός εφεδρικός κόμβος θα πρέπει να συνδεθεί με όλους τους γειτονικούς των κρίσιμων κόμβων. Αυτό αφενός

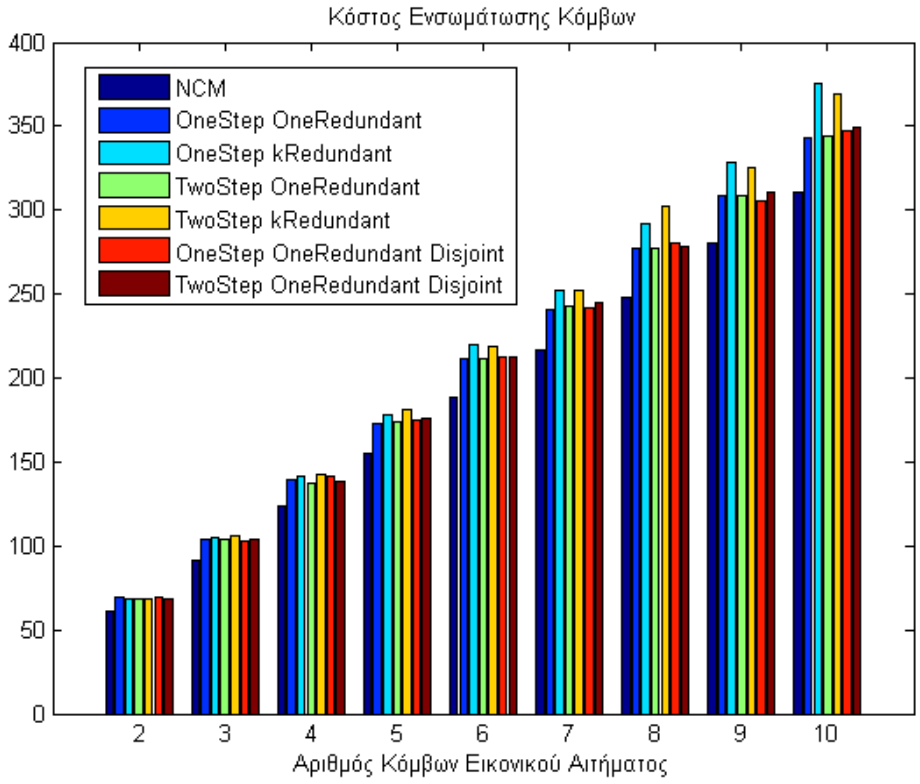
θα έχει ως συνέπεια να τοποθετηθεί μακριά από κάποιους κόμβους με τους οποίους θα πρέπει να συνδεθεί, από την άλλη έχει ως συνέπεια μεγάλη απαίτηση σε εύρος ζώνης προς τον κόμβο αυτόν με αποτέλεσμα ενδεχομένως ροή από κάποιους κόμβους να αναγκάζεται να ακολουθεί μεγαλύτερες διαδρομές λόγο έλλειψης χωρητικότητας στις προσκείμενες στον κρίσιμο κόμβο ζεύξεις.



ΣΧΗΜΑ 4. 11 Μέσο Κέρδος

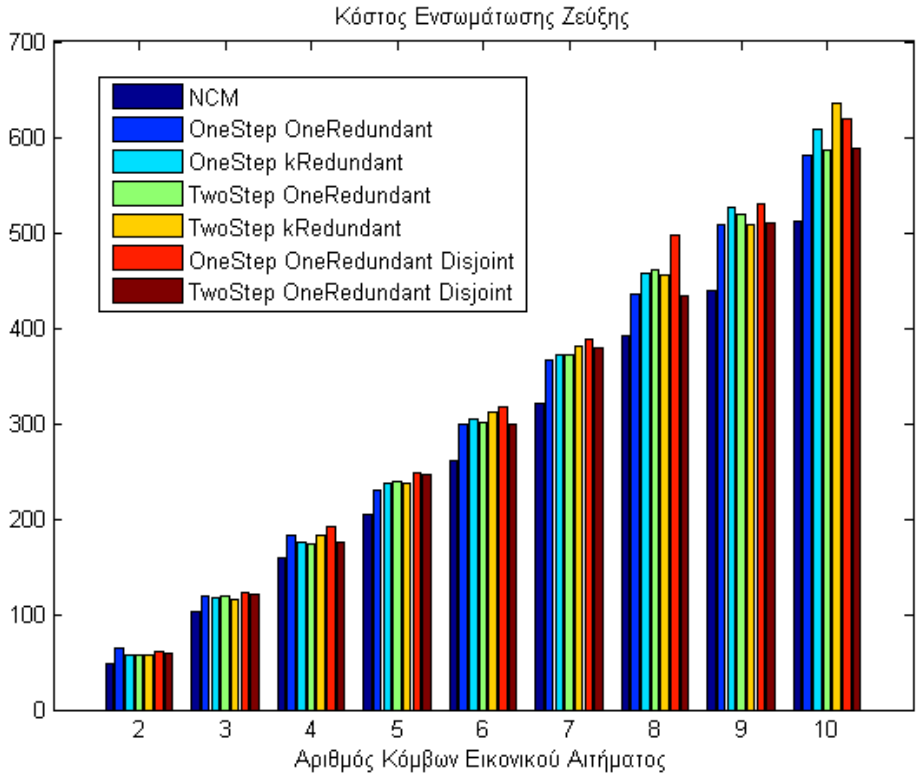
Το μέσο κέδρος, ακολουθώντας τα ποσοστά αποδοχής εμφανίζεται σταθερά καλύτερο για τις πλεονάζουσες κατά_1_πολιτικές. Οι πλεονάζουσες κατά_k_πολιτικές, λόγο αυξημένου κόστους, πού όπως θα δούμε παρακάτω αυξάνεται με μεγαλύτερο ρυθμό απ' ότι στις άλλες πολιτικές για με αύξηση του αριθμού των κόμβων, θα παρουσιάζουν μικρότερα ποσοστά αποδοχής για μεγάλα εικονικά αιτήματα με αποτέλεσμα το τελικό μέσο κέρδος να είναι μικρότερο.

Στα σχήματα 4.5 – 4.7 που ακολουθούν παρουσιάζεται το κόστος ενσωμάτωσης συναρτήσει του αριθμού των κόμβων.



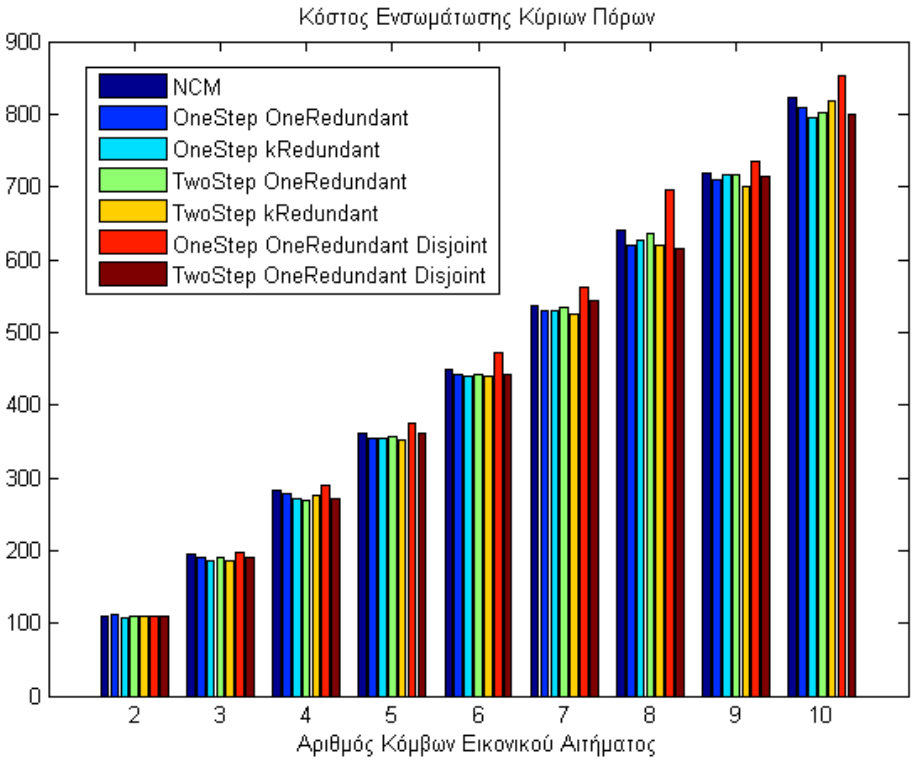
ΣΧΗΜΑ 4. 12 Μέσο κόστος ενσωμάτωσης κόμβων συναρτήσει του αριθμού των κόμβων

Το κόστος ενσωμάτωσης κόμβων αυξάνεται σχεδόν γραμμικά σε σχέση με τον αριθμό των κόμβων για την πλεονάζουσα_κατά_1 πολιτική και με μεγαλύτερο ρυθμό για την πλεονάζουσα_κατά_k πολιτική, αφού στη δεύτερη με αύξηση του αριθμού των εικονικών κόμβων και συνακόλουθα και του αριθμού κρίσιμων κόμβων θα πρέπει να αυξάνονται και οι εφεδρικοί κόμβοι.



ΣΧΗΜΑ 4. 13 Μέσο κόστος ενσωμάτωσης ζεύξεων συναρτήσει του αριθμού των κόμβων

Όμοια με το κόστος ενσωμάτωσης κόμβων, έτσι και το κόστος ενσωμάτωσης ζεύξεων αυξάνει με μεγαλύτερο ρυθμό για την ενσωμάτωση αιτημάτων με πλεονάζουσα_κατά_1 πολιτική. Επίσης όπως αναφέρθηκε και προηγουμένως παρατηρούμε αυξημένο κόστος ενσωμάτωσης ζεύξεων για τον αλγόριθμο με ενσωμάτωση κόμβων σε ένα βήμα που όμως η ενσωμάτωση των ζεύξεων γίνεται σε δύο βήματα για να εξασφαλιστούν ασύνδετα μονοπάτια που όμως εξηγήθηκε οφείλεται το μεγαλύτερο μήκος των μονοπατιών που προκύπτουν.



ΣΧΗΜΑ 4. 14 Μέσο κόστος ενσωμάτωσης κύριων πόρων συναρτήσει του αριθμού των κόμβων

Το κόστος ενσωμάτωσης κύριων πόρων παρουσιάζει μικρές διακυμάνσεις. Αξιοσημείωτη είναι η πάλι η περίπτωση με ενσωμάτωση κόμβων σε ένα βήμα που όμως η ενσωμάτωση των ζεύξεων γίνεται σε δύο βήματα για να εξασφαλιστούν ασύνδετα μονοπάτια, όπου ο συνδυασμός αυτός αναγκάζει την κύρια ροή να διασχίζει μεγαλύτερα μονοπάτια.

4.5 Σύνοψη συμπερασμάτων αξιολόγησης

Συγκρίνοντας τα αποτελέσματα συνολικά παρατηρούμε ότι οι αλγόριθμοι με ενσωμάτωση κόμβων σε ένα βήμα για ανοχή σε σφάλματα σε κόμβους πετυχαίνουν καλύτερα ποσοστά αποδοχής και μικρότερο κόστος ενσωμάτωσης. Το τίμημα που έχουμε να πληρώσουμε σε αυτή την περίπτωση είναι το μεγάλο υπολογιστικό κόστος επίλυση του ΓΠ που έχει 4 μεταβλητές σε σχέση με αυτό του αλγορίθμου με ενσωμάτωση σε δύο βήματα που έχει δύο μεταβλητές.

Για ανοχή επιπλέον σε σφάλματα και σε ζεύξεις προσκείμενες στους κρίσιμους κόμβους με την πολιτική της μετακίνησης (migration) βλέπουμε ότι ο αλγόριθμος με ενσωμάτωση σε δύο βήματα υπερτερεί καθώς όπως εξηγήθηκε δεν είναι καλός ο συνδυασμός της ενσωμάτωσης των κύριων και εφεδρικών κόμβων από κοινού σε συνδυασμό με την ενσωμάτωση της κύριας και εφεδρικής ροής σε διαφορετικά βήματα.

Παρατηρούμε επίσης ότι η πλεονάζουσα_κατά_k πολιτική, αν και έχει μεγαλύτερο κόστος ενσωμάτωσης και χαμηλότερα ποσοστά αποδοχής, αλλά και μεγαλύτερη υπολογιστική πολυπλοκότητα αφού ενσωματώνει μεγαλύτερα αξιόπιστα εικονικά αιτήματα, αποφέρει καλύτερη ποιότητα υπηρεσίας αφού μειώνεται ο μέσος αριθμός αλμάτων. Επιπλέον υπενθυμίζουμε ότι εξασφαλίζει ανοχή σε πολλαπλά σφάλματα. Στην περίπτωση αυτή συγκρούεται το επιχειρηματικό συμφέρον του παρόχου υποδομής με αυτό του χρήστης.

Τέλος αξίζει να παρατηρήσουμε ότι το επιπλέον κόστος για να παρέχουμε ανοχή σε σφάλματα σε προσκείμενες στους κρίσιμους κόμβους είναι μικρό και τα ποσοστά αποδοχής δεν διαφέρουν σημαντικά αλλά και το κέδρος δεν διαφέρουν σημαντικά, καθιστώντας έτσι την πολιτική μετακίνησης (migration) κόμβων ιδιαίτερα ελκυστική.

5

Επίλογος

5.1 Σύνοψη και συμπεράσματα

Στην εργασία αυτή μελετήσαμε σε βάθος το πρόβλημα της εικονικοποίησης δικτύων με ανοχή σε σφάλματα. Παρουσιάστηκαν και συγκρίθηκαν λύσεις που προτείνονται στη βιβλιογραφία και διαπιστώθηκε η ανάγκη λύσης που να παρέχει ανοχή ταυτόχρονα σε πολλαπλά σφάλματα σε ζεύξεις και κόμβους.

Προς αυτή την κατεύθυνση, σχεδιάστηκαν αλγόριθμοι για την ενσωμάτωση ετερογενών εικονικών δικτύων με ανοχή σε σφάλματα σε κρίσιμους κόμβους σε συνδυασμό με δύο πολιτικές δέσμευσης εφεδρικών πόρων για την ανοχή είτε σε μεμονωμένα είτε σε πολλαπλά σφάλματα. Στη συνέχεια επεκτάθηκαν για να εξασφαλιστεί ανεκτικότητα στις προσκείμενες σε αυτούς ζεύξεις. Οι αλγόριθμοι αυτοί συγκρίθηκαν σε περιβάλλον προσομοίωσης που τροποποιήθηκε κατάλληλα για την υποστήριξη εφεδρικών πόρων.

Με τη σύγκριση καταλήγουμε στο συμπέρασμα ότι ο αλγόριθμος για ανοχή σε σφάλματα σε κόμβους με ταυτόχρονη ενσωμάτωση εφεδρικών και εικονικών κόμβων, ως ακριβέστερη λύση, οδηγεί σε χαμηλότερο κόστος ενσωμάτωσης και υψηλότερα ποσοστά αποδοχής με μειονέκτημα ωστόσο το μεγαλύτερο υπολογιστικό κόστος. Παρατηρήσαμε επίσης ότι οι αλγόριθμοι με κρίσιμους κόμβους για κάθε εφεδρικό εκτός από ανοχή σε πολλαπλά σφάλματα, μειώνουν και το μήκος της διαδρομής που αντιστοιχίζονται στο δίκτυο υποδομής παρέχοντας καλύτερη ποιότητα υπηρεσίας με μικρότερη καθυστέρηση. Ο πάροχος υποδομής

ωστόσο θα έχει μεγαλύτερο κόστος και μικρότερο κέρδος. Αξίζει τέλος να σημειωθεί ότι η επέκταση της λύσης για ανοχή σε σφάλματα σε ζεύξεις δεν επηρεάζει σημαντικά την επίδοση των αλγορίθμων και καθίσταται εφικτή.

5.2 Μελλοντικές επεκτάσεις

Οι παραπάνω λύσεις εύκολα μπορούν να επεκταθούν ώστε να μπορούν να συμπεριλαμβάνονται στο σύνολο των κρίσιμων κόμβων και ενδιάμεσοι κόμβοι (μεταγωγείς, δρομολογητές), θέμα ακόμη για το οποίο δεν έχει προταθεί κάποια λύση στη βιβλιογραφία. Επέκταση μπορεί επίσης να γίνει ώστε η λύση να είναι ανεκτική σε σφάλματα στο σύνολο των ζεύξεων του εικονικού αιτήματος. Τέλος, οι αλγόριθμοι που προτάθηκαν θα μπορούσαν να συνδυαστούν με κάποια πολιτική διαμοιρασμού εφεδρικών πόρων ώστε να μειωθεί το κόστος ενσωμάτωσης και να αυξηθούν τα ποσοστά αποδοχής και κέρδους.

6

Βιβλιογραφία

(Anderesen, 2002)	D. G. Andersen, "Theoretical approaches to node assignment," Dec. 2002, unpublished Manuscript
(Cheng, 2011)	X. Cheng, S. Su, Z. Zhang, H. Wang, F. Yang, Y. Luo, and J. Wang, "Virtual network embedding through topology-aware node ranking," SIGCOMM Comput. Commun. Rev., vol. 41, pp. 38–47, April 2011.
(Chowdhury, 2009a)	N. Chowdhury, R. Boutaba, "Network Virtualization: State of the Art and Research Challenges", Communications Magazine, IEEE, vol 47, issue 7, pp. 20-26, July 2009.
(Chowdhury, 2009b)	N. Chowdhury, M. Rahman, and R. Boutaba, "Virtual network embedding with coordinated node and link mapping," in INFOCOM 2009, IEEE, April 2009, pp. 783 –791.
(CPLEX, 2014)	IBM ILOG CPLEX Optimizer, available at http://www-01.ibm.com/software/integration/optimization/cplex-
(CVI-SIM, 2014)	https://github.com/chrisap/CVI-SIM
(Eppstein, 1999)	D. Eppstein, "Finding the k shortest paths," SIAM J. Comput., vol. 28, pp. 652–673, February 1999.
(Gill, 2011)	P. Gill, N. Jain, and N. Nagappan, "Understanding network failures in data centers: measurement, analysis, and implications," in Proceedings of the ACM

	SIGCOMM 2011 conference. ACM, 2011, pp. 350–361.
(Guo, 2011)	T. Guo, N. Wang, K. Moessner and R. Tafazolli, "Shared Backup Network Provision for Virtual Network Embedding", in International Conference On Communications (ICC), June 2011, pp. 1-5
(Haider, 2009)	A. Haider, R. Potter, and A. Nakao, "Challenges in resource allocation in network virtualization," in 20th ITC pecialist Seminar, vol. 18, 2009, p. 20.
(Houidi, 2008)	I. Houidi, W. Louati, and D. Zeghlache, "A distributed virtual network mapping algorithm," in ICC '08. IEEE International Conference on Communications, May 2008, pp. 5634 –5640.
(Houidi, 2010)	I. Houidi, W. Louati, D. Zeghlache, P. Papadimitriou, and L. Mathy, "Adaptive virtual network provisioning," in Proceedings of the second ACM SIGCOMM workshop VISA. ACM, 2010, pp. 41–48.
(Hu, 2012)	Q. Hu, Y. Wang, X. Cao – Sarnoff , "Location-constrained survivable network virtualization", in Sarnoff Symposium (SARNOFF), 2012 IEEE, May 2012, pp 1-5
(JUNG, 2014)	JUNG 2.0.1 website, available athttp://jung.sourceforge.net.
(Lischka, 2009)	J. Lischka and H. Karl, "A virtual network mapping algorithm based on subgraph isomorphism detection," in Proceedings of the 1st ACM workshop VISA, ser. VISA '09. ACM, 2009, pp. 81–88.
(Markopoulou, 2008)	A. Markopoulou, G. Iannaccone, S. Bhattacharyya, C.-N. Chuah, Y. Ganjali, and C. Diot. Characterization of failures in an operational IP backbone network. IEEE/ACM Transactions on Networking, 2008.
(Niebert, 2008)	N. Niebert, I. E. Khayat, S. Baucke, R. Keller, R. Rembarz and J. Sachs, Network Virtualization: A Viable Path Towards the Future Internet Springer Wireless Pers. Commun., pp. 511-520, March 2008
(Papagianni, 2013)	C. Papagianni, A. Leivadeas, S. Papavassiliou, V. Maglaris, C. Cervello-Pastor and A. Monje, "On the optimal allocation of virtual resources in cloud computing networks", IEEE Transactions on Computers, 2013
(Pióro, 2004)	M. Pióro and D. Medhi, Routing, Flow, and Capacity Design in Communication and Computer Networks. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2004.
(Rhaman, 2013)	M. R. Rhaman and R. Boutaba, "SVNE: Survival Virtual Network Embedding Algorithms for Network Virtualization", in IEEE TRANSACTIONS ON NETWORK AND SERVICE MANAGEMENT, vol 10, no 2, June 2013, pp

	105-118
(Ricci, 2003)	R. Ricci, C. Alfeld, and J. Lepreau, "A solver for the network testbed mapping problem," SIGCOMM Comput. ommun. Rev., vol. 33, pp. 65– 81, April 2003.
(Schaffrath, 2009)	G. Schaffrath, C. Werle, P. Papadimitriou, A. Feldmann, R. Bless, A. Greenhalgh, A. Wundsam, M. Kind, O. Maennel, and L. Mathy, "Network virtualization architecture: proposal and initial prototype," in Proceedings of the 1st ACM workshop on Virtualized infrastructure systems and architectures, ser. VISA '09. New York, NY, USA: ACM, 2009, pp. 63–72.
(Voorsluys, 2009)	W. Voorsluys, J. Broberg, S. Venugopal, and R. Buyya, "Cost of virtual machine live migration in clouds: A performance evaluation," in Proceedings of the 1st International Conference on Cloud Computing, ser. CloudCom '09, 2009, pp. 254–265
(Xu, 2012)	J. Xu, J. Tang, K. Kwiat, W. Zhang, and G. Xue, "Survivable virtual infrastructure mapping in virtualized data centers," in IEEE 5th International Conference on Cloud Computing (CLOUD), 2012, June 2012.
(Yeow, 2011)	W. Yeow, C. Westphal, U. C. Kozat, "Designing and Embedding Reliable Virtual Infrastructures", in ACM SIGCOMM, vol 41, issue 2, April 2011, pp 57-64
(Yu, 2008)	M. Yu, Y. Yi, J. Rexford, and M. Chiang, "Rethinking virtual network embedding: substrate support for path splitting and migration," SIGCOMM Comput. Commun. Rev., vol. 38, pp. 17–29, March 2008.
(Yu, 2010)	H Yu, C Qiao, V Anand, X Liu, "Survivable virtual infrastructure mapping in a federated computing and networking system under single regional failures", GLOBECOM 2010, IEEE, 2010
(Yu, 2011)	H. Yu, V. Asand, C. Qiao and H. Di, "Migration Based Protection for Virtual Infrastructure Survivability for Link Failures", in Optical Fiber Communication Conference and Exposition (OFC/NFOEC), 2011 and the National Fiber Optic Engineers Conference, March 2011, pp 1-3
(Yu, 2013)	H. Yu, V. Anand, C. Qiao and Gang Sun, "Cost Efficient Design of Survival Virtual Infrastructure to Recover From Facility Node Failures", in International Conference on Communications (ICC), June 2013

7

Παράρτημα

7.1 Πίνακας Συμβόλων

Σύμβολο	Ερμηνεία
G^V	Γράφος που αναπαριστά το εικονικό αίτημα
N^V	Σύνολο εικονικών κόμβων
C	Σύνολο κρίσιμων εικονικών κόμβων
E^V	Σύνολο εικονικών ζεύξεων
G^S	Γράφος που αναπαριστά το δίκτυο υποσρτώματος
N^S	Σύνολο κόμβων δικτύου υποστρώματος
E^S	Σύνολο ζεύξεων δικτύου υποστρώματος
A	Σύνολο κατηγοριών κόμβων
V_a^X	Σύνολο κόμβων κατηγορίας $a \in A$ όπου , $X \in \{V, R, S, S'\}$ το εικονικό αίτημα, το αξιόπιστο εικονικό αίτημα, το δίκτυο υποστρώματος και το επαυξημένο δίκτυο υποστρώματος αντίστοιχα
I	Το σύνολο των λειτουργικών γνωρισμάτων των κόμβων (χωρητικοτήτων)
$c_i(n^X)$	Η χωρητικότητα του κόμβου n στο δίκτυο $X \in \{V, R, S, S'\}$ για το γνώρισμα $i \in I$

$bw(n^X, m^X)$	Το εύρος ζώνης μεταξύ των κόμβων $n.m$ στο δίκτυο $X \in \{V, R, S, S'\}$ για το γνώρισμα $i \in I$
B	Το σύνολο των εφεδρικών κόμβων
E^B	Το σύνολο των εφεδρικών ζεύξεων
C	Το σύνολο των κρίσιμων κόμβων
G^R	Ο γράφος που αναπαριστά το αξιόπιστο εικονικό αίτημα
N^R	Το σύνολο των κόμβων του αξιόπιστου εικονικού αιτήματος $N^R = N^V \cup B$
E^R	Το σύνολο των ζεύξεων του αξιόπιστου εικονικού αιτήματος $E^R = E^V \cup E^B$
$\|S\|$	Η πληθυκότητα του συνόλου S
$G^{S'}$	Το επαυξημένο με το αξιόπιστο εικονικό αίτημα δίκτυο υποστρώματος
$N^{S'}$	Το σύνολο των κόμβων του επαυξημένου δικτύου υποστρώματος $N^{S'} = N^R \cup N^S$
$E^{S'}$	Το σύνολο των ζεύξεων του επαυξημένου δικτύου υποστρώματος $E^{S'} = E^R \cup E^S$
P^S	Μονοπάτι στο δίκτυο υποστρώματος
$\boldsymbol{P^S}$	Σύνολο μονοπατιών στο δίκτυο υποστρώματος
$C_i(n^s)$	Η διαθέσιμη χωρητικότητα του λειτουργικού γνωρίσματος $i \in I$ για τον κόμβο $n^s \in N^S$
$BW(u^s, v^s)$	Το διαθέσιμο εύρος ζώνης της ζεύξης $(u^s, v^s) \in E^S$

7.2 Λεπτομέρειες υλοποίησης

Στο κεφάλαιο αυτό περιγράφονται οι τεχνικές λεπτομέρειες που αφορούν στην επέκταση του προσομοιωτή CVI-Sim (CVI-SIM, 2014) για την δημιουργία εικονικών αιτημάτων με εφεδρικούς κόμβους καθώς και την κωδικοποίηση του γραμμικού προγραμματισμού με τη χρήση της CPLEX βιβλιοθήκης

7.2.1 Επέκταση Μοντέλου Για Εικονικά Αιτήματα με Εφεδρικούς Πόρους

Προκειμένου να μοντελοποιηθούν οι εικονικοί κόμβοι επεκτάθηκαν οι κλάσεις `VirtualMachine`, `RequestRouter` και `RequestSwich` από τις κλάσεις `BackupVM`, `BackupRouter` και `BackupSwitch` αντίστοιχα. Οι νέες κλάσεις που δημιουργήθηκαν κληρονομούν από τις αντίστοιχες αρχικές για να είναι συμβατές με τον υπάρχοντα κώδικα για την ενσωμάτωση εικονικού αιτήματος στο δίκτυο υποστρώματος. Αντίστοιχα η κλάση `RequestLink` επεκτάθηκε από την κλάση `BackupLink`.

Επίσης αναγκαία ήταν η δημιουργία δυο διαπροσωπιών (Interfaces) για να μπορούμε να διαχωρίζουμε τους κόμβους του αρχικού εικονικού αιτήματος από τους εφεδρικούς. Παρακάτω παρατίθεται ο κώδικας των διαπροσωπιών.

```
public interface RequestNode {
        public Node getBackupNode() ;
        public void setBackupNode(Node backup);
        public boolean isCritical();
}
```

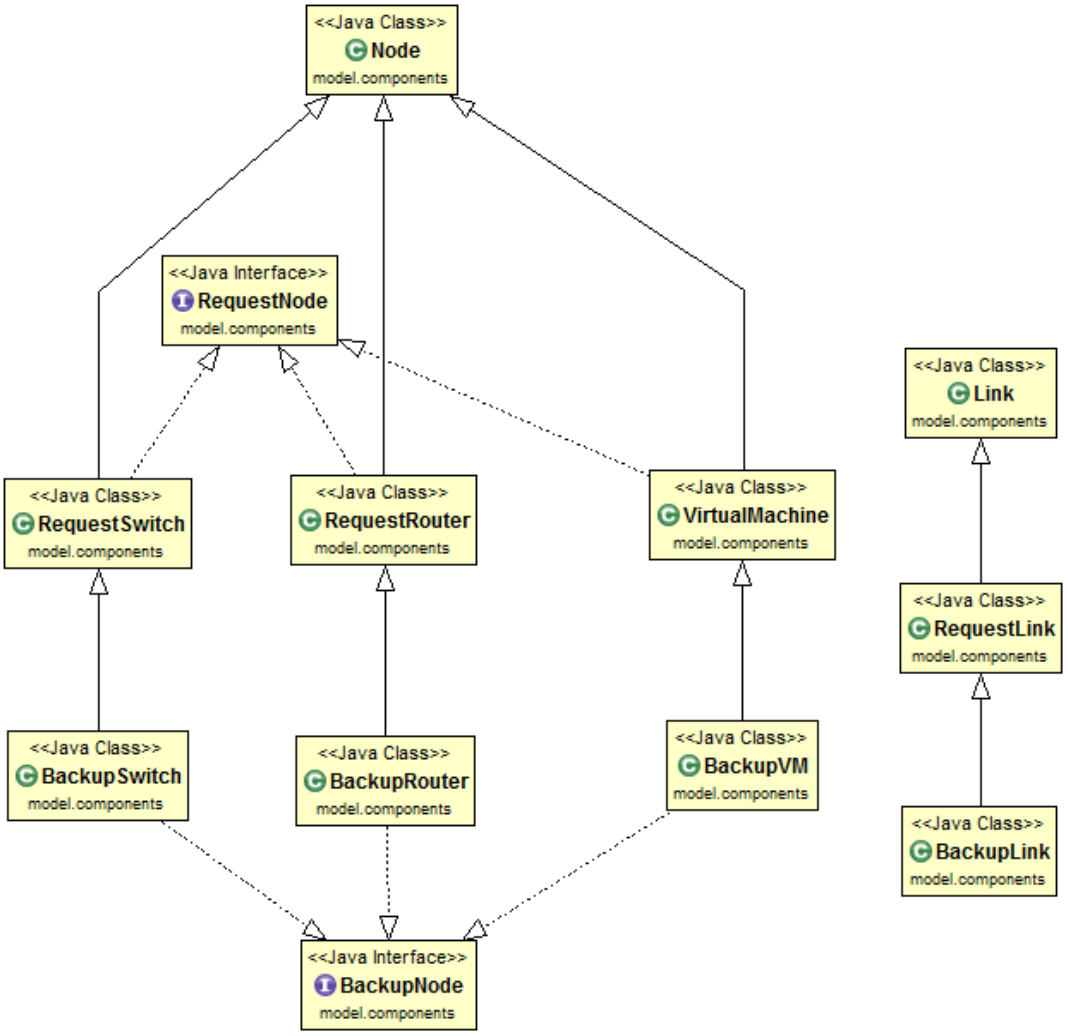
Κώδικας 7. 1 RequestNode Interface

```
public interface BackupNode {
        public List<Node> getCriticalNodes() ;
        public void setCriticalNodes(List<Node> criticalNodes);
        public void addCriticalNode(Node criticalNode);
}
```

Κώδικας 7. 2 BackupNode Interface

Όπως βλέπουμε ένας κόμβος εικονικού αιτήματος μπορεί ή όχι να είναι κρίσιμος. Ένας κρίσιμος κόμβος σχετίζεται με έναν εφεδρικό και τέλος ένας εφεδρικός σχετίζεται με μία λίστα από κρίσιμους κόμβους.

Στο παρακάτω απλοποιημένο διάγραμμα κλάσεων φαίνονται οι σχέσεις κληρονομικότητας που περιγράφηκαν παραπάνω:



ΣΧΗΜΑ 7. 2Απλοποιημένο Διάγραμμα Κλάσεων

Για τη δημιουργία των εφεδρικών κόμβων δημιουργήσαμε την κλάση `BackupNodeFactory` που διαθέτει δύο μεθόδους: μία σύμφωνα με την πλεονάζουσα_κατά_1 πολιτική και μία σύμφωνα με την πλεονάζουσα_κατα_k πολιτική. Παρακάτω παρατίθεται ο κώδικας των δύο μεθόδων της κλάσης:

```
        public Node createOneRedundunt(Node critical) {
                Node node = null;

                if(backupNodes.get(critical.getType())==null){
                        if (critical.getType().equalsIgnoreCase("Server")){
                                node = new BackupVM(nodeCount);
                                backupNodes.put(node.getType(), node);
                                ((VirtualMachine) node).setDiskSpace(((VirtualMachine)
critical).getDiskSpace());
                        }
                        else if (critical.getType().equalsIgnoreCase("Switch")){
                                node = new BackupSwitch(nodeCount);
                                backupNodes.put(node.getType(), node);
                        }
                        else if (critical.getType().equalsIgnoreCase("Router")){
                                node = new BackupRouter(nodeCount);
                                backupNodes.put(node.getType(), node);
                        }
                        node.setCpu(critical.getCpu());
                        node.setMemory(critical.getMemory());
                        nodeCount++;

                        ((BackupNode) node).addCriticalNode(critical);
                        ((RequestNode) critical).setBackupNode(node);

                        return node;
                }
                else{
                        node=backupNodes.get(critical.getType());
                        if (critical.getType().equalsIgnoreCase("Server"))
                                ((VirtualMachine) node).setDiskSpace(((VirtualMachine)
critical).getDiskSpace());
                        node.setCpu(Math.max(critical.getCpu(),node.getCpu()));
                        node.setMemory(Math.max(critical.getMemory(), node.getMemory()));
                        //node.setVlans(Math.max(critical.getVlans(), node.getVlans()));

                        ((BackupNode) node).addCriticalNode(critical);
                        ((RequestNode) critical).setBackupNode(node);

                        return null;
                }

        }
```

Κώδικας 7. 3 Μέθοδος δημιουργίας εφεδρικού κόμβου με πλεονάζουσα_κατά_1 πολιτική

```
public Node createKRedundunt(Node critical) {
                Node node = null;

                if (critical.getType().equalsIgnoreCase("Server")){
                        node = new BackupVM(nodeCount);
                        ((VirtualMachine) node).setDiskSpace(((VirtualMachine)
critical).getDiskSpace());
                }
                else if (critical.getType().equalsIgnoreCase("Switch")){
                        node = new BackupSwitch(nodeCount);
                }
                else if (critical.getType().equalsIgnoreCase("Router")){
                        node = new BackupRouter(nodeCount);
                }
                node.setCpu(critical.getCpu());
                node.setMemory(critical.getMemory());
                nodeCount++;

                ((BackupNode) node).addCriticalNode(critical);
                ((RequestNode) critical).setBackupNode(node);

                return node;
        }
```

Κώδικας 7. 4Μέθοδος δημιουργίας εφεδρικού κόμβου με πλεονάζουσα_κατά_k πολιτική

7.2.1.1 Αλγόριθμοι ΕΕΔ με ανοχή σε σφάλματα

Ακολουθεί ο κώδικας από τα κύρια βήματα των αλγορίθμων.

```
int backupLinkCount = req.getGraph().getEdgeCount();

for (Node node : reqCopy.getGraph().getVertices()){
        reliableRequest.getGraph().addVertex(node);
        if ( ((RequestNode) node).isCritical() ){
                Node backupNode = backupNodeFactory.createOneRedundunt(node);
                //Node backupNode = backupNodeFactory.createKRedundunt(node);
                if(backupNode!=null)
                        reliableRequest.getGraph().addVertex(backupNode);
                else
                        backupNode=((RequestNode) node).getBackupNode();

                for (Node y : reqCopy.getGraph().getNeighbors(node)){
                        Link l = reqCopy.getGraph().findEdge(node, y);
                        Link existingLink = reliableRequest.getGraph().findEdge(backupNode, y);
                        if(existingLink == null){
                                BackupLink backupLink = new
                                BackupLink(backupLinkCount,l.getBandwidth());
                                reliableRequest.getGraph().addEdge(backupLink, backupNode, y,
                                EdgeType.UNDIRECTED);
                                backupLink.addWorkingLink((RequestLink) l);
                                backupLinkCount++;
                        }
                        else{
                                existingLink.setBandwidth(Math.max(l.getBandwidth(),
                                existingLink.getBandwidth()));
                                ((BackupLink) existingLink).addWorkingLink((RequestLink) l);
                        }
                }
        }
}
```

Κώδικας 7. 5 Προσαύξηση του εικονικού αιτήματος με τους εφεδρικούς κόμβους ανάλογα με την επιθυμητή πολιτική

```
for (Node x : tmpRelReqNodes){
        augmentedSubstrate.getGraph().addVertex(x);
        for (Node subNode : substrateCopy.getGraph().getVertices()){
                int LinkID = subLinkNum++;
                SubstrateLink l = new SubstrateLink(LinkID,(int) 2.147483647E5);
                augmentedSubstrate.getGraph().addEdge(l, x, subNode, EdgeType.UNDIRECTED);
        }
}
```

Κώδικας 7. 6 Προσαύξηση του δικτυακού υποστρώματος με το αξιόπιστο εικονικό αίτημα

όπου `tmpRelReqNodes` δομή που περιέχει τους κόμβους του αξιόπιστου εικονικού αιτήματος

Στη συνέχεια καλούμαστε να λύσουμε το πρόβλημα ΓΠ όπου κωδικοποιείται διαφορετικά με τον αν η ενσωμάτωση εικονικών κόμβων γίνεται σε ένα ή δύο βήματα.

7.2.1.1.1 Αλγόριθμος με αντιστοίχιση εικονικών κόμβων σε ένα βήμα

Παρακάτω βλέπουμε την αρχικοποίηση των τεσσάρων μεταβλητών του προβλήματος:

```
IloNumVar[][][] x = new IloNumVar[reqLinksNum][][];
for (int k=0;k<reqLinksNum;k++){
        x[k]=new IloNumVar[augGraph][];
        for (int i=0;i<augGraph;i++){
                x[k][i]=cplex.numVarArray(augGraph, 0, 1);
        }
}

IloNumVar[][][] y = new IloNumVar[backupLinksNum][][];
for (int k=0;k<backupLinksNum;k++){
        y[k]=new IloNumVar[augGraph][];
        for (int i=0;i<augGraph;i++){
                y[k][i]=cplex.numVarArray(augGraph, 0, 1);
        }
}

IloNumVar[][][] f = new IloNumVar[reqLinksNum][][];
for (int k=0;k<reqLinksNum;k++){
        f[k]=new IloNumVar[augGraph][];
        for (int i=0;i<augGraph;i++){
                f[k][i]=cplex.numVarArray(augGraph, 0, 1000000);
        }
}

IloNumVar[][][] g = new IloNumVar[backupLinksNum][][];
for (int k=0;k<backupLinksNum;k++){
        g[k]=new IloNumVar[augGraph][];
        for (int i=0;i<augGraph;i++){
                g[k][i]=cplex.numVarArray(augGraph, 0, 1000000);
        }
}
```

Κώδικας 7. 7 Πεδίο ορισμού μεταβλητών

Ακολουθεί ο σχηματισμός των τριών όρων της αντικειμενικής συνάντησης:

```
//////////////////////////////////////////////////////////////////
////////////Objective function: Total amount of bandwidth/////////
//////////////////////////////////////////////////////////////////
IloLinearNumExpr flows = cplex.linearNumExpr();
for (int i=relReqNodeNum;i<augGraph; i++){
        for (int j=relReqNodeNum;j<augGraph; j++){
                for (int k =0; k<reqLinksNum; k++)
                        flows.addTerm(1.0/(augCapTable[i][j]+0.000001), f[k][i][j]);
                for (int k =0; k<backupLinksNum; k++)
                        flows.addTerm(1.0/(augCapTable[i][j]+0.000001),g[k][i][j]);
        }
}
```

Κώδικας 7. 8 Συνολικό εύρος ζώνης

```
//////////////////////////////////////////////////////////////////
////////////Objective function: Total amount of capacities/////////
//////////////////////////////////////////////////////////////////
IloLinearNumExpr capacities = cplex.linearNumExpr();
for (int i=0;i<v_types;i++){                    //for each category
        for (int w : Vs.get(i)){                //for each virtual node of the category
                for (int m : Vv.get(i)){        //for each substrate node of the category
                        double total_cap=0;
                        for (int j=0;j<3;j++){  //for each capacity of the category
                                total_cap=
                                ((total_cap+Vcost[j][m]/(Scost[j][w]+0.000001)));
                        }
                        for (int k=0;k<reqLinksNum;k++)  //for each virtual link
                                capacities.addTerm(total_cap,x[k][m][w+relReqNodeNum]);
                        for (int k=0;k<backupLinksNum;k++)//for each backup link
                                capacities.addTerm(total_cap,y[k][m][w+relReqNodeNum]);
                        }
                }
}
```

Κώδικας 7. 9 Άθροισμα υπολογιστικών πόρων

```
//////////////////////////////////////////////////////////////////
////////////Objective function: Total Hops per Virtual Link////////
//////////////////////////////////////////////////////////////////
IloLinearNumExpr hops = cplex.linearNumExpr();
for (int i=relReqNodeNum;i<augGraph; i++){
        for (int j=relReqNodeNum;j<augGraph; j++){
                for (int k=0;k<reqLinksNum;k++)  //for each virtual link
                        hops.addTerm(1.0/(augCapTable[i][j]+0.000001), x[k][i][j]);
                for (int k=0;k<backupLinksNum;k++)       //for each backup link
                        hops.addTerm(1.0/(augCapTable[i][j]+0.000001), y[k][i][j]);
        }
}
```

Κώδικας 7. 10 Άθροισμα βημάτων για την αντιστοίχιση εικονικών ζεύξεων σε ζεύξεις του υποστρώματος

Στη συνέχεια παρατίθεται ο κώδικας για τους περιορισμούς. Η αρίθμηση στα σχόλια αντιστοιχεί με την αρίθμηση των περιορισμών στο κεφάλαιο 3.

```
////////////////////////////////////////////////////////////////////
////////////// Primary Flow Reservation Constraints/////////////////
////////////////////////////////////////////////////////////////////

//14a1
for (int k=0;k<reqLinksNum;k++){
        for (int i=0;i<augGraph; i++){
                IloLinearNumExpr capReq = cplex.linearNumExpr();
                ArrayList<Integer> link = reqLinksToNodes.get(k);
                if (i!=link.get(0)&&i!=link.get(1)){
                        for (int j=0;j<augGraph; j++){
                                capReq.addTerm(1,f[k][i][j]);
                                capReq.addTerm(-1,f[k][j][i]);
                        }
                cplex.addEq(capReq,0);
                }
        }
}

//14a2
for (int k=0;k<reqLinksNum;k++){
        IloLinearNumExpr capReq = cplex.linearNumExpr();
        ArrayList<Integer> link = backupLinksToNodes.get(k);
        for (int w=0;w<augGraph;w++){
                capReq.addTerm(1,f[k][link.get(1)][w]);
                capReq.addTerm(-1,f[k][w][link.get(0)]);
        }
        double reCap = 0;
        reCap=link.get(2);
        cplex.addEq(capReq,reCap);
}

//14a3
for (int k=0;k<reqLinksNum;k++){
        IloLinearNumExpr capReq = cplex.linearNumExpr();
        ArrayList<Integer> link = backupLinksToNodes.get(k);
        for (int w=0;w<augGraph;w++){
                capReq.addTerm(1,f[k][link.get(1)][w]);
                capReq.addTerm(-1,f[k][w][link.get(2)]);
        }
        double reCap = 0;
        reCap=-link.get(3);
        cplex.addEq(capReq,reCap);
}
```

Κώδικας 7. 11 Περιορισμοί διατήρησης κύριας ροής κίνησης

```
////////////////////////////////////////////////////////////////////
////////////// Backup Flow Reservation Constraints//////////////////
////////////////////////////////////////////////////////////////////

//14b1
for (int k=0;k<backupLinksNum;k++){
        for (int i=0;i<augGraph; i++){
        IloLinearNumExpr capReq = cplex.linearNumExpr();
        ArrayList<Integer> link = backupLinksToNodes.get(k);
        if (i!=link.get(0)&&i!=link.get(1)){
                for (int j=0;j<augGraph; j++){
                        capReq.addTerm(1,g[k][i][j]);
                        capReq.addTerm(-1,g[k][j][i]);
                        }
                cplex.addEq(capReq,0);
                }
        }
}

//14b2
for (int k=0;k<backupLinksNum;k++){
        IloLinearNumExpr capReq = cplex.linearNumExpr();
        ArrayList<Integer> link = backupLinksToNodes.get(k);
        for (int w=0;w<augGraph;w++){
                capReq.addTerm(1,g[k][link.get(0)][w]);
                capReq.addTerm(-1,g[k][w][link.get(0)]);
        }
        double reCap = 0;
        reCap=link.get(2);
        cplex.addEq(capReq,reCap);
}

//14b3
for (int k=0;k<backupLinksNum;k++){
        IloLinearNumExpr capReq = cplex.linearNumExpr();
        ArrayList<Integer> link = backupLinksToNodes.get(k);
        for (int w=0;w<augGraph;w++){
                capReq.addTerm(1,g[k][link.get(1)][w]);
                capReq.addTerm(-1,g[k][w][link.get(1)]);
        }
        double reCap = 0;
        reCap=-link.get(2);
        cplex.addEq(capReq,reCap);
}
```

Κώδικας 7. 12 Περιορισμός διατήρησης εφεδρικής ροής κίνησης

Οι δομές `reqLinksToNodes` και `backupLinksToNodes` είναι πίνακες κατακερματισμού για τις κύριες και εφεδρικές ζεύξεις αντίστοιχα όπου κλειδί είναι το id της ζεύξης και τιμή ένας πίνακας τριών στοιχείων. Το πρώτο είναι η προέλευση της ζεύξης, το δεύτερο ο προορισμός και το τρίτο το εύρος ζώνης.

```
////////////////////////////////////////////////////////////////////
////////////Node Capacity Constraints/////////////////////////////
////////////////////////////////////////////////////////////////////

//15a
for (int k=0;k<reqLinksNum;k++){
    for (int i=0;i<s_types;i++){
        for (int w : Vs.get(i)){
            for (int m : Vv.get(i)){
                for (int j=0;j<3;j++){
                    double cpuSub=Scost[j][w];
                    cplex.addLe(cplex.prod(Vcost[j][m], x[k][m][w+relReqNodeNum]),cpuSub);
                }
            }
        }
    }
}

//15b
for (int k=0;k<backupLinksNum;k++){
    for (int i=0;i<s_types;i++){
        for (int w : Vs.get(i)){
            for (int m : Vv.get(i)){
                for (int j=0;j<3;j++){
                    double cpuSub=Scost[j][w];
                    cplex.addLe(cplex.prod(Vcost[j][m], y[k][m][w+relReqNodeNum]),cpuSub);
                }
            }
        }
    }
}
```

Κώδικας 7. 13 Περιορισμοί για τη χωρητικότητα κόμβων

```
////////////////////////////////////////////////////////////////////
////////////Link Capacity Constraints/////////////////////////////
////////////////////////////////////////////////////////////////////

//16a
for (int k=0;k<reqLinksNum; k++){
        for (int i=0;i<augGraph; i++){
                for (int j=0;j<augGraph; j++){
                        IloNumExpr flowSum =cplex.prod(x[0][0][0],0);
                        IloLinearNumExpr availBW = cplex.linearNumExpr();
                        flowSum = cplex.sum(cplex.prod(f[k][i][j],1),
                                cplex.prod(f[k][j][i],1));
                        double capacity = augCapTable[i][j];
                        availBW.addTerm(capacity,x[k][i][j]);
                        cplex.addLe(flowSum,availBW);
                }
        }
}

//16b
for (int k=0;k<backupLinksNum; k++){
        for (int i=0;i<augGraph; i++){
                for (int j=0;j<augGraph; j++){
                        IloNumExpr flowSum =cplex.prod(y[0][0][0],0);
                        IloLinearNumExpr availBW = cplex.linearNumExpr();
                        flowSum = cplex.sum(cplex.prod(g[k][i][j],1),
                                        cplex.prod(g[k][j][i],1));

                        double capacity = augCapTable[i][j];
                        availBW.addTerm(capacity,y[k][i][j]);
                        cplex.addLe(flowSum,availBW);
                }
        }
}
//17
for (int i=0;i<augGraph; i++){
        for (int j=0;j<augGraph; j++){
                IloNumExpr expr1 =cplex.prod(x[0][0][0],0);
                IloNumExpr flowSum =cplex.prod(x[0][0][0],0);

                for (int k=0;k<reqLinksNum; k++){
                        flowSum = cplex.sum(cplex.prod(f[k][i][j],1),
                                cplex.prod(f[k][j][i],1));
                        expr1 =cplex.sum(expr1,flowSum);
                }
                for (int k=0;k<backupLinksNum; k++){
                        flowSum = cplex.sum(cplex.prod(g[k][i][j],1),
                                        cplex.prod(g[k][j][i],1));
                        expr1 =cplex.sum(expr1,flowSum);
                }
                double capacity = augCapTable[i][j];

                cplex.addLe(expr1, capacity);
        }
}
```

Κώδικας 7. 14 Περιορισμοί για τη χωρητικότητα ζεύξεων

όπου τα στοιχεία του πίνακα `augCapTable[[i][j]` είναι η διαθέσιμη χωρητικότητα της ζεύξης ij του επαυξημένου υποστρώματος.

Ακολουθούν οι περιορισμοί που αφορούν στην αντιστοίχιση των εικονικών κόμβων και εφεδρικών εικονικών κόμβων σε κόμβους του δικτυακού υποστρώματος.

Υπενθυμίζεται ότι ενώ οι εικονικές ζεύξεις του συνόλου E^V ενώνουν μόνο κόμβους του αρχικού εικονικού αιτήματος, οι εφεδρικές ζεύξεις του συνόλου E^B ενώνουν ένα εφεδρικό κόμβο τον γειτονικό κόμβο του κρίσιμου κόμβου που καλύπτεται από τον εφεδρικό. Ο πίνακας `isAdjToCritical[m]` που συναντάται στους παρακάτω περιορισμούς είναι ένας λογικός πίνακας που παίρνει την τιμή $true$ όταν ο κόμβο m είναι φειτονικός με κάποιον κρίσιμο κόμβο και άρα θα πρόσκειται σε αυτόν εφεδρική ζεύξη.

```
//18a
for (int k=0;k<reqLinksNum;k++){
        for (int i=0;i<s_types;i++){
                for (int w : Vs.get(i)){
                        IloLinearNumExpr capReq = cplex.linearNumExpr();
                        for (int m : Vr.get(i)){
                                capReq.addTerm(1,x[k][m][w+relReqNodeNum]);
                        }
                        cplex.addLe(capReq,1);
                }
        }
}

//18b
for (int k1=0;k1<backupLinksNum;k1++){
        for (int k2=0;k2<reqLinksNum;k2++){
                for (int i=0;i<s_types;i++){
                        for (int w : Vs.get(i)){
                                IloLinearNumExpr capReq = cplex.linearNumExpr();
                                for (int m : Vr.get(i)){
                                        capReq.addTerm(1,y[k1][m][w+relReqNodeNum]);
                                        if(!isAdjToCritical[m])
                                            capReq.addTerm(1,x[k2][m][w+relReqNodeNum]);
                                }
                                for (int m : Vb.get(i)){
                                   capReq.addTerm(1,y[k1][m+reqNodeNum][w+relReqNodeNum]);
                                }
                                cplex.addLe(capReq,1);
                        }
                }
        }
}
```

Κώδικας 7. 15 Αντιστοίχιση κύριου/εφεδρικού εικονικού κόμβου σε μοναδικό κόμβο του δικτυακού υποστρώματος

```
//19a + 20a
for (int k=0;k<reqLinksNum;k++){
        for (int i=0;i<s_types;i++){
                for (int m : Vr.get(i)){
                        IloLinearNumExpr capReq = cplex.linearNumExpr();
                        for (int w : Vs.get(i)){
                                capReq.addTerm(1,x[k][m][w+relReqNodeNum]);
                        }
                        cplex.addEq(capReq,1);
                }
        }
}

for (int k=0;k<reqLinksNum;k++){
        for (int m=0;m<relReqNodeNum;m++){
                IloLinearNumExpr capReq = cplex.linearNumExpr();
                for (int w=relReqNodeNum;w<augGraph;w++){
                        capReq.addTerm(1,x[k][m][w]);
                }
                cplex.addLe(capReq, 1);
        }
}

//19b + 20b + 20c
for (int k=0;k<backupLinksNum;k++){
        for (int i=0;i<s_types;i++){
                for (int m : Vr.get(i)){
                        if(isAdjToCritical[m]){
                                IloLinearNumExpr capReq = cplex.linearNumExpr();
                                for (int w : Vs.get(i)){
                                        capReq.addTerm(1,y[k][m][w+relReqNodeNum]);
                                }
                                cplex.addEq(capReq,1);
                        }
                }
                for (int m : Vb.get(i)){
                        IloLinearNumExpr capReq = cplex.linearNumExpr();
                        for (int w : Vs.get(i)){
                                capReq.addTerm(1,y[k][m+reqNodeNum][w+relReqNodeNum]);
                        }
                        cplex.addEq(capReq,1);
                }
        }
}

for (int k=0;k<backupLinksNum;k++){
        for (int m=0;m<relReqNodeNum;m++){
                IloLinearNumExpr capReq = cplex.linearNumExpr();
                for (int w=relReqNodeNum;w<augGraph;w++){
                        capReq.addTerm(1,y[k][m][w]);
                }
                cplex.addLe(capReq, 1);
        }
}
```

Κώδικας 7. 16 Αντιστοίχιση το πολύ ενός κύριου/εφεδρικού εικονικού κόμβου στον ίδιο κόμβο υποστρώματος

Ο συνδυασμός των παρακάτω περιορισμών είναι ισοδύναμος με τους περιορισμούς 19 και 20 της ενότητας 3. Συγκεκριμένα ο πρώτος εξασφαλίζει ότι οπωσδήποτε ένας εικονικός κόμβος πρέπει να αντιστοιχίζεται σε έναν κόμβο υποστρώματος της ίδιας κατηγορίας και ο δεύτερος ότι σε ένα κόμβο υποστρώματος αντιστοιχίζεται το πολύ ένας εικονικός κόμβος.

```
//21
for (int k=0;k<reqLinksNum;k++){
        for (int m=reqNodeNum;m<relReqNodeNum;m++){
                IloLinearNumExpr capReq = cplex.linearNumExpr();
                for (int w=relReqNodeNum;w<augGraph;w++){
                        capReq.addTerm(1,x[k][m][w]);
                }
                cplex.addEq(capReq,0);
        }
}
```

Κώδικας 7. 17 Όχι κύρια ροή από εφεδρικούς κόμβους

```
//22
for (int k1=0;k1<backupLinksNum;k1++){
        for (int m=0;m<reqNodeNum; m++){
                if(isAdjToCritical[m]){
                        for (int w=relReqNodeNum;w<augGraph;w++){
                                for (int k2=0;k2<reqLinksNum;k2++){
                                        cplex.addEq(x[k2][m][w],y[k1][m][w]);
                                }
                        }
                }
                else{
                        IloLinearNumExpr capReq = cplex.linearNumExpr();
                        for (int w=relReqNodeNum;w<augGraph;w++){
                                capReq.addTerm(1,y[k1][m][w]);
                        }
                        cplex.addEq(capReq,0);
                }
        }
}
```

Κώδικας 7. 18 Οι πίνακες x και y πρέπει να είναι συνεπείς μεταξύ τους για τους κοινούς του εικονικού κόμβους

```
//23a
for (int k=0;k<reqLinksNum;k++){
        for (int i=0;i<augGraph; i++){
                for (int j=0;j<augGraph; j++){
                        IloLinearNumExpr capReq1 = cplex.linearNumExpr();
                        IloLinearNumExpr capReq2 = cplex.linearNumExpr();
                        capReq1.addTerm(1,x[k][i][j]);
                        capReq2.addTerm(1,x[k][j][i]);
                        cplex.addEq(capReq1,capReq2);
                }
        }
}

//16b
for (int k=0;k<backupLinksNum;k++){
        for (int i=0;i<augGraph; i++){
                for (int j=0;j<augGraph; j++){
                        IloLinearNumExpr capReq1 = cplex.linearNumExpr();
                        IloLinearNumExpr capReq2 = cplex.linearNumExpr();
                        capReq1.addTerm(1,y[k][i][j]);
                        capReq2.addTerm(1,y[k][j][i]);
                        cplex.addEq(capReq1,capReq2);
                }
        }
}
```

Κώδικας 7. 19 Τριγωνική συμμετρία πικάκων x, y

```
//24a
for (int k=0;k<reqLinksNum;k++){
        for (int i=0;i<augGraph;i++){
                for (int j=0;j<augGraph;j++){
                        IloNumExpr flowSum =cplex.prod(x[0][0][0],0);
                        IloLinearNumExpr capReq = cplex.linearNumExpr();
                        capReq.addTerm(1,x[k][i][j]);
                        flowSum = cplex.sum(cplex.prod(f[k][i][j],1),
                        cplex.prod(f[k][j][i],1));
                        cplex.addGe(flowSum, capReq);
                }
        }
}

//24b
for (int k=0;k<backupLinksNum;k++){
        for (int i=0;i<augGraph;i++){
                for (int j=0;j<augGraph;j++){
                        IloNumExpr flowSum =cplex.prod(y[0][0][0],0);
                        IloLinearNumExpr capReq = cplex.linearNumExpr();
                        capReq.addTerm(1,y[k][i][j]);
                        flowSum = cplex.sum(cplex.prod(g[k][i][j],1),
                        cplex.prod(g[k][j][i],1));
                        cplex.addGe(flowSum, capReq);
                }
        }
}
```

Κώδικας 7. 20 Οι μεταβλητές x, y τίθενται όταν περνάει ροή από τις αντίστοιχες ζεύξεις

```
//25a
for (int k1=0;k1<reqLinksNum;k1++){
        for (int k2=0;k2<reqLinksNum;k2++){
                for (int i=0;i<relReqNodeNum; i++){
                        for (int j=relReqNodeNum;j<augGraph; j++){
                                IloLinearNumExpr capReq1 = cplex.linearNumExpr();
                                IloLinearNumExpr capReq2 = cplex.linearNumExpr();
                                capReq1.addTerm(1,x[k1][i][j]);
                                capReq2.addTerm(1,x[k2][i][j]);
                                cplex.addEq(capReq1,capReq2);
                        }
                }
        }
}

//25b
for (int k1=0;k1<backupLinksNum;k1++){
        for (int k2=0;k2<backupLinksNum;k2++){
                for (int i=0;i<relReqNodeNum; i++){
                        for (int j=relReqNodeNum;j<augGraph; j++){
                                IloLinearNumExpr capReq1 = cplex.linearNumExpr();
                                IloLinearNumExpr capReq2 = cplex.linearNumExpr();
                                capReq1.addTerm(1,y[k1][i][j]);
                                capReq2.addTerm(1,y[k2][i][j]);
                                cplex.addEq(capReq1,capReq2);
                        }
                }
        }
}
```

Κώδικας 7. 21 Η αντιστοίχιση πρέπει να είναι συνδεδεμένος γράφος

Για την αντιστοίχιση των ζεύξεων έχουμε το τροποποιημένο πρόβλημα πολλαπλών ροών όπως περιεγράφηκε στην ενότητα 3.2.3.1

```
IloNumVar[][][] f_mcf = new IloNumVar[reqLinksNum][][];
for (int k=0;k<reqLinksNum;k++){
       f_mcf[k]=new IloNumVar[subNodeNum][];
       for (int i=0;i<subNodeNum;i++){
           f_mcf[k][i]=cplex1.numVarArray(subNodeNum,0,Double.MAX_VALUE);
       }
}

IloNumVar[][][] g_mcf = new IloNumVar[backupLinksNum][][];
for (int k=0;k<backupLinksNum;k++){
       g_mcf[k]=new IloNumVar[subNodeNum][];
       for (int i=0;i<subNodeNum;i++){
           g_mcf[k][i]=cplex1.numVarArray(subNodeNum,0,Double.MAX_VALUE);
       }
}
```

Κώδικας 7. 22 Μεταβλητέ MCF για ταυτόχρονη αντιστοίχιση κύριας και εφεδρικής ροής

```
//////////////////////////////////////////////////////////////////
/////////////// Objective function ///////////////////////////////
//////////////////////////////////////////////////////////////////
IloLinearNumExpr flows_mcf = cplex1.linearNumExpr();
for (int i=0;i<subNodeNum; i++){
       for (int j=0;j<subNodeNum; j++){
              for (int k =0; k<reqLinksNum; k++) {
                     flows_mcf.addTerm(1.0, f_mcf[k][i][j]);
              }
              for (int k =0; k<backupLinksNum; k++) {
                     flows_mcf.addTerm(1.0, g_mcf[k][i][j]);
              }
       }
}
```

Κώδικα 7. 23 Αντικειμενική Συνάρτηση

```
//flow reservation 1a
for (int k=0;k<reqLinksNum;k++){
        for (int i=0;i<subNodeNum;i++){
                IloLinearNumExpr capReq = cplex1.linearNumExpr();
                if ((o1[k]!=i)&&(d1[k]!=i)){
                        for (int j=0;j<subNodeNum;j++){
                                capReq.addTerm(1,f_mcf[k][i][j]);
                                capReq.addTerm(-1,f_mcf[k][j][i]);
                        }
                }
                cplex1.addEq(capReq,0);
        }
}


//flow reservation 2a
for(int k=0;k<reqLinksNum;k++){
        IloLinearNumExpr capReq = cplex1.linearNumExpr();
                for (int i=0;i<subNodeNum;i++){
                        capReq.addTerm(1,f_mcf[k][o1[k]][i]);
                        capReq.addTerm(-1,f_mcf[k][i][o1[k]]);
                }
                double reCap = 0;
                reCap=tmpRelReqLinks[reqLinksToNodes.get(k).get(0)]
                        [reqLinksToNodes.get(k).get(1)];
                cplex1.addEq(capReq,reCap);
}

//flow reservation 3a
for(int k=0;k<reqLinksNum;k++){
        IloLinearNumExpr capReq = cplex1.linearNumExpr();
        for (int i=0;i<subNodeNum;i++){
                capReq.addTerm(1,f_mcf[k][d1[k]][i]);
                capReq.addTerm(-1,f_mcf[k][i][d1[k]]);                                  }
                double reCap = 0;
                reCap=-tmpRelReqLinks[reqLinksToNodes.get(k).get(0)]
                        [reqLinksToNodes.get(k).get(1)];
                cplex1.addEq(capReq,reCap);
}
```

Κώδικας 7. 24 Διατήρηση κύριας ροής

όπου οι πίνακες `o1[k]` και `d1[k]` δίνουν τον κόμβο του υποστρώματος που έχουν αντιστοιχηθεί τα άκρα της εικονικής ζεύξης `k`, πηγή και προορισμός αντίστοιχα.

```
//flow reservation 1b
for (int k=0;k<backupLinksNum;k++){
        for (int i=0;i<subNodeNum;i++){
                IloLinearNumExpr capReq = cplex1.linearNumExpr();
                if ((o2[k]!=i)&&(d2[k]!=i)){
                        for (int j=0;j<subNodeNum;j++){
                                capReq.addTerm(1,g_mcf[k][i][j]);
                                capReq.addTerm(-1,g_mcf[k][j][i]);
                        }
                }
                cplex1.addEq(capReq,0);
        }
}

//flow reservation 2b
for(int k=0;k<backupLinksNum;k++){
        IloLinearNumExpr capReq = cplex1.linearNumExpr();
        for (int i=0;i<subNodeNum;i++){
                capReq.addTerm(1,g_mcf[k][o2[k]][i]);
                capReq.addTerm(-1,g_mcf[k][i][o2[k]]);
        }
        double reCap = 0;
        reCap=tmpRelReqLinks[backupLinksToNodes.get(k).get(0)]
                [backupLinksToNodes.get(k).get(1)];
        cplex1.addEq(capReq,reCap);
}

//flow reservation 3b
for(int k=0;k<backupLinksNum;k++){
        IloLinearNumExpr capReq = cplex1.linearNumExpr();
        for (int i=0;i<subNodeNum;i++){
                capReq.addTerm(1,g_mcf[k][d2[k]][i]);
                capReq.addTerm(-1,g_mcf[k][i][d2[k]]);
        }
        double reCap = 0;
        reCap=-tmpRelReqLinks[backupLinksToNodes.get(k).get(0)]
                [backupLinksToNodes.get(k).get(1)];
        cplex1.addEq(capReq,reCap);
}
```

Κώδικας 7. 25 Διατήρηση εφεδρικής ροής

όπου οι πίνακες `o2[k]` και `d2[k]` όπως πριν αλλά για τις εφεδρικές εικονικές ζεύξεις

```
//Link constraint
for (int i=0;i<subNodeNum; i++){
        for (int j=0;j<subNodeNum; j++){
                IloNumExpr expr1 =cplex1.prod(f_mcf[0][0][0],0);
                IloNumExpr flowSum =cplex1.prod(f_mcf[0][0][0],0);
                for (int k=0;k<reqLinksNum; k++){
                        flowSum = cplex1.sum(cplex1.prod(f_mcf[k][i][j],1),
                        cplex1.prod(f_mcf[k][j][i],1));
                        expr1 =cplex1.sum(expr1,flowSum);
                }
                flowSum =cplex1.prod(f_mcf[0][0][0],0);
                for (int k=0;k<backupLinksNum; k++){
                        flowSum = cplex1.sum(cplex1.prod(g_mcf[k][i][j],1),
                        cplex1.prod(g_mcf[k][j][i],1));
                        expr1 =cplex1.sum(expr1,flowSum);
                }
                double capacity =augCapTable[relReqNodeNum+i][relReqNodeNum+j];
                cplex1.addLe(expr1,capacity);
        }
}
```

Κώδικας 7. 26 Περιορισμός χωρητικότητας ζεύξεων υποστρώματος

7.2.1.2 Αλγόριθμος με αντιστοίχιση εικονικών κόμβων σε δύο βήματα

Έχοντας τρέξει τον NCM αλγόριθμο αποθηκεύουμε στους πίνακες `x, f, fi` τα αποτελέσματα όπως προέκυψαν μετά την στρογγυλοποίηση της μεταβλητής `x` και την επίλυση του κλασικού προβληματος πολλαπλών ροών για την `fi`.

Οι πίνακες `x, f` θα πρέπει να προσαυξηθούν κατάλληλα με 0 ώστε να ταιριάζουν στις διαστάσει του επαυξημένου με τι αξιόπιστο εικονικό αίτημα δικτυακού υποστρώματος

```
IloNumVar[][][] y = new IloNumVar[backupLinksNum][][];
for (int k=0;k<backupLinksNum;k++){
        y[k]=new IloNumVar[augGraph][];
        for (int i=0;i<augGraph;i++){
                y[k][i]=cplex.numVarArray(augGraph, 0, 1);//, IloNumVarType.Int);
        }
}

IloNumVar[][][] g = new IloNumVar[backupLinksNum][][];
for (int k=0;k<backupLinksNum;k++){
        g[k]=new IloNumVar[augGraph][];
        for (int i=0;i<augGraph;i++){
                g[k][i]=cplex.numVarArray(augGraph, 0, 1000000);
        }
}
```

Κώδικας 7. 27 Μεταβλητές του προβλήματος

```
//////////////////////////////////////////////////////////////////
////////////Objective function: Total amount of bandwidth/////////
//////////////////////////////////////////////////////////////////
IloLinearNumExpr flows = cplex.linearNumExpr();
for (int i=relReqNodeNum;i<augGraph; i++){
        for (int j=relReqNodeNum;j<augGraph; j++){
                for (int k =0; k<backupLinksNum; k++)
                        flows.addTerm(1.0/(augCapTable[i][j]+0.000001), g[k][i][j]);
        }
}
```

Κώδικας 7. 28 Συνολικό Εύρος Ζώνης

```
//////////////////////////////////////////////////////////////////
////////////Objective function: Total amount of capacities//////////
//////////////////////////////////////////////////////////////////
IloLinearNumExpr capacities = cplex.linearNumExpr();
for (int i=0;i<v_types;i++){                    //for each category
        for (int w : Vs.get(i)){                //for each virtual node of the category
                for (int m : Vv.get(i)){        //for each substrate node of the category
                        double total_cap=0;
                        for (int j=0;j<3;j++){  //for each capacity of the category
                                total_cap= total_cap+Vcost[j][m]/(Scost[j][w]+0.000001)));
                        }
                        for (int k=0;k<backupLinksNum;k++)      //for each backup link
                                capacities.addTerm(total_cap,y[k][m][w+relReqNodeNum]);
                }
        }
}
```

Κώδικας 7. 29 Άθροισμα Υπολογιστικών Πόρων

```
//////////////////////////////////////////////////////////////////
////////////Objective function: Total Hops per Virtual Link////////
//////////////////////////////////////////////////////////////////
IloLinearNumExpr hops = cplex.linearNumExpr();
for (int i=relReqNodeNum;i<augGraph; i++){
        for (int j=relReqNodeNum;j<augGrah; j++){
                for (int k=0;k<backupLinksNum;k++)       //for each backup link
                        hops.addTerm(1.0/(augCapTable[i][j]+0.000001), y[k][i][j]);
        }
}
```

Κώδικας 7. 30 Συνολικά Άλματα

Ως προς τους περιορισμούς δεν αλλάζει κάτι σε σχέση με τον προηγούμενο αλγόριθμο. Κρατάμε μόνο τους περιορισμούς που εμπλέκουν τις μεταβλητές y και g.

Στη συνέχεια λύνεται το κλασικό πρόβλημα πολλαπλών ροών για την αντιστοίχιση των εφεδρικών ζεύξεων όπου όμως στον περιορισμό για το διαθέσιμο εύρος ζώνης λαμβάνουμε υπόψιν και την κύρια ροή που έχει ήδη αντιστοιχηθεί:

```
//Link constraint
for (int i=0;i<initSubNodeNum; i++){
        for (int j=0;j<initSubNodeNum; j++){
                IloNumExpr expr1 =cplex1.prod(g_mcf[0][0][0],0);
                double flowSum1 = 0;
                IloNumExpr flowSum2 =cplex1.prod(g_mcf[0][0][0],0);
                for (int k=0;k<reqLinksNum; k++){
                        flowSum1 = flowSum1 + fi[k][i][j] + fi[k][j][i];
                }
                for (int k=0;k<backupLinksNum; k++){
                        flowSum2 = cplex1.sum(cplex1.prod(g_mcf[k][i][j],1),
                        cplex1.prod(g_mcf[k][j][i],1));
                        expr1 =cplex1.sum(expr1,flowSum2);
                }
                double capacity =augCapTable[relReqNodeNum+i][relReqNodeNum+j]-flowSum1;
                cplex1.addLe(expr1,capacity);
        }
}
```

Κώδικας 7. 31 Περιορισμός εύρους ζώνης στο MCF πρόβλημα

7.2.1.3 Αλγόριθμοι με ανοχή σε σφάλματα σε ζεύξεις

Για να εξασφαλίσουμε ανεκτικότητα σε σφάλματα σε ζεύξεις και στους δύο παραπάνω αλγορίθμους θα γίνει η εξής τροποποίηση.

Αρχικά στη φάση της αντιστοίχισης των κόμβων (στο δεύτερο βήμα για το δεύτερο αλγόριθμο) προστίθεται ο παρακάτω περιορισμός που μεριμνά για τα ασύνδετα μονοπάτια:

```
for(int k=0; k<backupLinksNum; k++){
        int src = backupLinksToNodes.get(k).get(0);
        int dest = backupLinksToNodes.get(k).get(1);
        Node node = tmpRelReqNodesList.get(src);
        List<Node> tmpCriticals=((BackupNode) node).getCriticalNodes();
        for(Node critical : tmpCriticals){
                int cid=critical.getId();
                for(int l=0;l<reqLinksNum;l++){
                        if(reqLinksToNodes.get(l).get(0)==cid &&
                             reqLinksToNodes.get(l).get(1)==dest
                        || reqLinksToNodes.get(l).get(1)==cid &&
                            reqLinksToNodes.get(l).get(0)==dest){
                                for (int m=relReqNodeNum;m<augGraph; m++){
                                        for (int w=relReqNodeNum;w<augGraph; w++){
                                                expr=cplex.sum(y[k][m][w],x[l][m][w]);
                                                cplex.addLe(expr,1);
                                        }
                                }
                        }
                }
        }
}
```

Κώδικας 7. 32 Περιορισμός για ασύνδετα μονοπάτια στη φάση αντιστοίχισης κόμβων

Επιπλέον χρειαζόμαστε ένα πίνακα για να αποθηκεύουμε πότε μια ζεύξη υποστρώματος δεν μπορεί να χρησιμοποιηθεί για να περάσει εφεδρική ροή όπως περιγράφεται στην ενότητα 3.3:

```
////////////////////////////////////////////////
///////////////// available backup paths
/**
* available[b][i][j] = 0, i.e. available for flow from virtual
* backup link b, if no flow from virtual a link l, s.t. b is a
* backup for l, is routed via the substrate link (i,j)
*/
double[][][] available =new double [backupLinksNum][substrate.getGraph().getVertexCount()]
                                    [substrate.getGraph().getVertexCount()];

for (int l=0;l<reqLinksNum;l++){
        for (int i=0;i<subNodeNum;i++){
                for (int j=0;j<subNodeNum;j++){
                        if(fi[l][i][j]>0.00001){
                                ArrayList<Integer> a = reqLinksToBackupLinks.get(l);
                                if(a!=null){
                                        for(int b:a){
                                                available[b][i][j]=1;
                                        }
                                }
                        }
                }
        }
}
```

Κώδικας 7. 33 Πίνακας ασύνδετων διαδρομών

Τέλος προσθέτουμε τον αντίστοιχο του (3.31) περιορισμό στην επίλυση του προβλήματος πολλαπλών ροών για την εφεδρική ροή που εξασφαλίζει με τη βοήθεια του παραπάνω πίνακας τις επιθυμητές ασύνδετες διαδρομές.

```
//disjoint working and corresponding backup links constraint
for (int i=0;i<subNodeNum; i++){
        for (int j=0;j<subNodeNum; j++){
                for (int k=0;k<backupLinksNum; k++){
                        if(available[k][i][j]==1){
                                cplex2.addEq(g_mcf[k][j][i],0);
                        }
                }
        }
}
```

Κώδικας 7. 34 Περιορισμός για ασύνδετες διαδρομές στο MCF πρόβλημα

7.2.2 Πλατφόρμες και προγραμματιστικά εργαλεία

Η εφαρμογή αναπτύχθηκε σε Java μέσω του ολοκληρωμένου περιβάλλοντος ανάπτυξης Ecplipse Hellios. Η εφαρμογή έτρεξε σε Java Virtual Machine μέσω του Java Runtime Environment έκδοση 1.6.0 (update 24). Για την επίλυση των προβλημάτων ΓΠ χρησιμοποιήθηκε το λογισμικό CPLEX της IBM έκδοση 12.3.

Τέλος, τα πειράματα έτρεξαν σε 2 εικονικά μηχανήματα με τα εξής χαρακτηριστικά:

- 2-πύρηνος επεξεργαστής Quad-Core AMD Opteron 2.50GHz
- Μνήμη 4GB
- Λειτουργικό σύστημα Windows 7 Professional 64bits