Μελέτη της δυναμικής συμπεριφοράς ενός P2P συστήματος

Διονύσης Χατζημαρκάκης

27 Ιανουαρίου 2009

Περιεχόμενα

Κεφάλαιο 1

Εισαγωγή

Τα υπολογιστικά και επικοινωνιακά περιβάλλοντα που βασίζονται στο internet σήμερα είναι πολύ πιο περίπλοκα και χαοτικά από τα κλασσικά κατανεμημένα συστήματα κυρίως γιατί δεν έχουν κάποια κεντρική οργάνωση και ιεραρχικό έλεγχο. Έχει γίνει μεγάλη έρευνα και υπάρχει πολύ ενδιαφέρον στην ανάπτυξη των Peer-to-peer συστημάτων γιατί παρέχουν ένα καλό υπόστρωμα για τη δημιουργία μεγάλου εύρους διαμοιρασμού πληροφορίας, κατανομής των περιεχομένων και application-level multicast applications.Τα peer-to-peer δίκτυα είναι κατανεμημένα συστήματα από τη φύση τους χωρίς ιεραρχική οργάνωση ή κεντρικό έλεγχο. Οι Peers διαμορφώνουν κατανεμημένα δίκτυα που οργανώνονται αυτόματα και εδραιώνονται πάνω στο internet , προσφέροντας ένα σύνολο απο χαρακτηριστικά. Τα βασικότερα είναι η ρωμαλαία αρχιτεκτονική δρομολόγησης ευρείας περιοχής, η επαρκής αναζήτηση δεδομένων, η επιλογή των κοντινότερων peers ,ο άφθονος αποθηκευτικός χώρος, η μονιμότητα, η ιεραρχική ονομασία, η εμπιστευτικότητα και η πιστοποίηση, η ανωνυμία, η τεράστια δυνατότητα αυξομοίωσης του δικτύου και η ανεκτικότητα σε σφάλματα. Τα συστήματα αυτά φτάνουν πέρα απο τις υπηρεσίες που προσφέρουν τα client-server συστήματα γιατί έχουν συμμετρία στους ρόλους καθώς ένας client μπορεί να είναι και server. Αυτό συμβαίνει γιατί ένας peer πληρώνει τη συμμετοχή του στη συνολική κοινότητα πληροφοριών προσφέροντας πρόσβαση στους πόρους και τα δεδομένα που έχει. Επίσης τα συστήματα αυτά επιτρέπουν σε άλλα συστήματα να έχουν πρόσβαση στους πόρους του κάθε peer και υποστηρίζουν resource sharing που απαιτεί ανεκτικότητα σε σφάλματα, αυτόματη οργάνωση αλλά και ιδιότητες massive scalability. Όπως γνωρίζουμε και απ το [1] υπάρχουν δύο κατηγορίες P2P συστημάτων:

- Unstructured

  Σε αυτη τη κατηγορία η τοπολογία του συστήματος δεν έχει σχέση με τον τρόπο που διανέμονται τα δεδομένα στους peers και χρησιμοποιούνται πιο συχνά στο σημερινό internet για διάφορους λόγους. Οι κυριότεροι είναι ότι δεν απαιτείται κάποιος κεντρικός server , είναι ταχύτερα για δεδομένα που ζητούνται συχνά, είναι self-scaling και όσο πιο πολλοί peers μπαίνουν στο σύστημα τόσο πιο ικανή καθίσταται η δυνατότητα download. Επίσης πρέπει να σημειώσουμε ότι ένα unstructed σύστημα συντίθεται από peers που μπαίνουν στο δίκτυο με μερικούς χαλαρούς κανόνες χωρίς παραπάνω γνώση της τοπολογίας του συστήματος.

- Structured

  Σε αυτή τη κατηγορία η τοπολογία του P2P συστήματος είναι αυστηρά ελεγχόμενη και τα δεδομένα αποθηκεύονται όχι σε τυχαίους κόμβους - peers αλλά σε συγκεκριμένες θέσεις που κάνουν τα queries πιο πρακτικά. Μερικά structed P2P συστήματα χρησιμοποιούν DHT-Distributed Hash Tables όπου τοποθετούνται πληροφορίες για τα δεδομένα και το που βρίσκονται, δηλαδή ποιοι peers είναι υπεύθυνοι για τα αντίστοιχα δεδομένα.

Σύμφωνα με το ίδιο άρθρο τα structured P2P συστήματα χρησιμοποιούν κάποιο Distributed Hash Table (DHT) σαν υπόστρωμα στο οποίο τοποθετείται η πληροφορία για το που βρίσκονται τα δεδομένα στους διάφορους peers. Αυτό γίνεται εφικτό αφού κάθε data object έχει ένα μοναδικό κλειδί και κάθε peer είναι υπεύθυνος να γνωρίζει που βρίσκονται τα δεδομένα με το συγκεκριμένο κλειδί. Αυτά τα συστήματα έχουν την ιδιότητα να αναθέτουν συνέχεια ομοιόμορφα τυχαία NodeIDs στο σύνολο των peers μέσα σε ένα μεγάλο εύρος αναγνωριστικών. Τα data objects αντιστοιχίζονται σε μοναδικά κλειδιά επιλεγμένα από τον ίδιο χώρο κλειδιών. Τα κλειδιά αυτά με τη σειρά τους αντιστοιχίζονται από το πρωτόκολο του δικτύου σε κάποιους υπάρχοντες online peers. Ο κάθε peer διατηρεί ένα μικρό routing table που αποτελείται από τα NodeIDs των γειτόνων και τις διευθύνσεις IP. Οι ερωτήσεις αναζήτησης καθώς και η δρομολόγηση των μηνυμάτων προωθούνται σε peers με έναν προοδευτικό τρόπο όπου τα NodeIDs έρχονται όλο και πιο κοντά στο κλειδί για το οποίο γίνεται η αναζήτηση. Στη θεωρία τα DHT συστήματα μπορούν να εγγυηθούν ότι κάθε data object μπορεί να βρεθεί σε $O(logN)$ βήματα κατά μέσο όρο, όπου N είναι ο αριθμός των peers στο σύστημα. Παρόλο που τα structured P2P συστήματα μπορούν να βρουν αποτελεσματικά σπάνια αντικείμενα, στη πράξη είναι πολύ πιο ακριβά απο τα unstructured P2P συστήματα για δεδομένα που ζητούνται συχνά.

1.1 Με τι ασχοληθήκαμε

Το αντικείμενο της εργασίας μας είναι η μελέτη της δυναμικής συμπεριφοράς ενός συστήματος P-grid. Το P-grid στη γενική του μορφή όπως γνωρίζουμε από το [2] και το [3] είναι μια τοπολογία που χρησιμοποιείται στα κατανεμημένα συστήματα. Ένας peer θα θεωρήσουμε ότι είναι ένας υπολογιστής, δηλαδή ένα μηχάνημα κάποιου online χρήστη. Σε ένα τέτοιο σύστημα ο κάθε peer αντιστοιχίζεται σε ένα φύλλο ενός δέντρου τύπου trie. Το trie είναι ένα δυαδικό δέντρο αναζήτησης όπου ο κάθε κόμβος του έχει ως αναγνωριστικό ένα string που αποτελείται απο 0 και 1 ανάλογα αν είναι αριστερό ή δεξί παιδί αντίστοιχα. Κάθε φύλλο του trie αντιπροσωπεύει ένα τμήμα του συνολικού χώρου δεδομένων και για το κάθε ένα φύλλο πρέπει να υπάρχει κάποιος peer να είναι υπεύθυνος γι΄ αυτό.

Στο σύστημα P-grid όπως περιγράφεται στη βασική του μορφή κάθε peer αντιπροσωπεύεται από ένα φύλλο του trie. Αυτό πρακτικά σημαίνει ότι ο κάθε peer θα πρέπει να γνωρίζει που βρίσκονται εκείνα τα δεδομένα που έχουν ώς data key - κλειδί που ανήκει στον χώρο δεδομένων που του έχει ανατεθεί. Ως κλειδί θεωρείται εκείνο το αναγνωριστικό κάθε αντικειμένου δεδομένων που έχει δημιουργηθεί χρησιμοποιώντας μια συγκεκριμένη hash function. Στο συγκεκριμένο σύστημα τα κλειδιά των αντικειμένων αυτών αντιστοιχίζονται σε έναν N-διάστατο χώρο και περισσότερες λεπτομέρειες που σχετίζονται με την τοπολογία P-grid θα μπορούσε να βρει κανείς στο [4]. Σε αυτό το σημείο πρέπει να πούμε ότι η βασική μορφή του P-grid όπως αναφέρεται στο παραπάνω άρθρο αλλά και στο [3] είναι διαφορετική από αυτή που μελετήσαμε στην παρούσα έρευνα. Στη βασική μορφή του P-grid θεωρείται ότι το κάθε φύλλο του trie αντιπροσωπεύει έναν peer. Κάτι τέτοιο δεν ισχύει στην παρούσα έρευνα καθώς τα φύλλα του trie αλλά και όλο αυτό το σύστημα θα βασίζεται στην ιδέα της virtual τοπολογίας των nodes. Δηλαδή οι nodes είναι virtual και όχι πραγματικοί peers . Οι πραγματικοί peers θα αναλαμβάνουν να είναι υπεύθυνοι για περισσότερα από ένα φύλλα-nodes του trie που αυτά με τη σειρά τους θα είναι υπεύθυνα να γνωρίζουν που βρίσκονται τα δεδομένα που αντιστοιχίζονται σε ένα κλειδί το οποίο θα ανήκει στο τμήμα του N-διάστατου χώρου δεδομένων για το οποίο είναι υπεύθυνα. Επίσης πρέπει να πούμε ότι όταν ένας peer επιθυμεί να βγεί εκτός του συστήματος, αναλαμβάνει να δώσει το φύλλο ή

τα φύλλα για τα οποία ήταν υπεύθυνος σε κάποιον άλλο online peer.Έτσι στην παρούσα έρευνα δεν χάνονται δεδομένα και αυτό είναι ένα βασικό πλεονέκτημα στην αξιοπιστία της αναζήτησης δεδομένων έναντι της βασικής μορφής του P-grid που περιγράφεται στο [2]. Καθώς βγαίνουν εκτός του συστήματος οι peers κάποιοι άλλοι peers πολλαπλασιάζουν τα φύλλα για τα οποία είναι υπεύθυνοι αφού κανένα φύλλο δεν επιτρέπεται να χαθεί καθώς αποχωρούν peers από το σύστημα. Με αυτή την παραδοχή όταν κάποιος νέος peer εισέρχεται στο σύστημα δεν είναι απαραίτητο να γίνει split κάποιο φύλλο αφού είναι πιθανό ο υπεύθυνος peer εκείνου του φύλλου που θα έκανε split να είναι υπεύθυνος για πολλά φύλλα του trie και άρα να δώσει ένα από αυτά στον νεο peer που θέλει να γίνει μέλος.

Επίσης με τον όρο δυναμική συμπεριφορά ενός τέτοιου συστήματος εννοούμε τη μελέτη του κόστους και την αξιολόγηση του συστήματος για διαδικασίες που σχετίζονται με τις εισόδους ή εξόδους των peers προς και από το σύστημα. Όλες αυτές οι διαδικασίες δρομολογούνται και ολοκληρώνονται από peers που επικοινωνούν μεταξύ τους μέσω internet στέλνοντας τα απαραίτητα μηνύματα ο ένας στον άλλο. Για να γίνει κάποιος νέος peer μέλος ενός τέτοιου συστήματος πρέπει να στείλει κάποια μηνύματα σε άλλους peers που είναι ήδη online στο σύστημα και κάποιοι από αυτούς πρέπει να στείλουν μηνύματα σε άλλους προκειμένου να ενημερωθούν για την είσοδο του νέου peer. Ομοίως για να βγει εκτός συστήματος κάποιος peer που είναι ήδη μέλος, θα πρέπει να στείλει μηνύματα σε κάποιους άλλους peers προκειμένου να ολοκληρωθεί η έξοδός του και να βρει κάποιους peers που θα αναλάβουν να είναι υπεύθυνοι για τα φύλλα του. Όλα αυτά τα μηνύματα τα μετρήσαμε και καταλήξαμε σε κάποια βασικά συμπεράσματα που μας δείχνουν μια ολοκληρωμένη εικόνα του κόστους των διαδικασιών εισόδου και εξόδου των peers προς και από το σύστημα καθώς αυξάνεται συγχρόνως ο αριθμός των online peers. Μετρώντας όλα αυτά τα μηνύματα και χρησιμοποιώντας τα κατάλληλα κριτήρια αξιολόγησης αποφανθήκαμε για το πόσο καλά προσαρμόζεται το σύστημα καθώς αυξάνεται σε αριθμό από online peers και πόσο γρήγορη είναι η δρομολόγηση των διαδικασιών εισόδου και εξόδου από πλευράς κόστους, δηλαδή μηνυμάτων.

Τα αποτελέσματα όπως θα δούμε μας οδηγούν σε ενδιαφέροντα συμπεράσματα. Όταν γίνεται είσοδος ενός νέου peer επιλέγουμε αρχικά κάποιο φύλλο του trie προκειμένου να δώσει μέρισμα του συνολικού χώρου δεδομένων στον νέο peer που θέλει να γίνει μέλος του συστήματος P-grid. Όπως θα δούμε στη συνέχεια όπου θα αναλύσουμε τα αποτελέσματα με μεγαλύτερη λεπτομέρεια ο τρόπος επιλογής εκείνου του αρχικού φύλλου επηρεάζει άμεσα την απόδοση του συστήματος αλλά και την ταχύτητα ολοκλήρωσης των διαδικασιών εισόδου για τους νέους peers. Επίσης θα δούμε ότι για τις διαδικασίες εξόδου των peers από το σύστημα, όταν το ποσοστό του χρόνου παραμονής των peers στο δίκτυο γίνει ίσο ή και μεγαλύτερο από το ποσοστό του χρόνου που βρίσκεται εκτός δικτύου, τότε τα μηνύματα που στέλνονται για κάθε έξοδο είναι σχεδόν ίδια ακόμα κι αν το ποσοστό χρόνου που βρίσκονται online οι peers γίνει πολύ μεγαλύτερο από το ποσοστό χρόνου που βρίσκονται offline. Επίσης θα δούμε ότι το ίδιο ισχύει όχι μόνο για τις διαδικασίες εξόδου των peers αλλά και για τις διαδικασίες εισόδου. Θα παρατηρήσουμε ακόμα ότι όταν δημιουργείται ένα μεγαλύτερο trie στέλνονται περισσότερα μηνύματα για το ίδιο πλήθος από online peers παρόλο που ο κάθε peer γίνεται υπεύθυνος για περισσότερα φύλλα. Τέλος θα δούμε ότι η απόδοση του συστήματος είναι πολύ απογοητευτική για τις διαδικασίες εξόδου κάτι όπως θα δούμε στη συνέχεια οφείλεται στη διαμόρφωση των συνδέσμων που πρέπει να διατηρούν τα φύλλα μεταξύ τους.

1.1.1 Τι μετρήσαμε στα πειράματά μας

Όπως γνωρίζουμε από το [5] είναι πολύ σημαντικό να μετρήσουμε τον μέγιστο αριθμό διεργασιών που μπορεί να εξυπηρετήσει ένα P2P σύστημα χωρίς να υπερφορτωθεί κάποιος peer. Το μέγεθος αυτό ονομάζεται maximum throughput και στη δική μας έρευνα οι διεργασίες για τις οποίες μελετάται είναι οι διαδικασίες εισόδου και εξόδου των peers. Πρέπει επίσης να επισημάνουμε ότι είναι η πρώτη φορά που μελετάται αυτό το μέγεθος για αυτού του είδους τις διαδικασίες αφού ως τώρα είχε χρησιμοποιηθεί μόνο για διεργασίες που αφορούν την αναζήτηση δεδομένων. Σύμφωνα με το ίδιο άρθρο ο μέγιστος αυτός αριθμός διεργασιών που μπορεί να εξυπηρετήσει το σύστημα πρέπει να αυξάνεται καθώς αυξάνεται και ο αριθμός των online peers. Όταν δεν συμβαίνει αυτό θα μπορούμε να λέμε ότι το

σύστημα δεν είναι scalable και δεν προσαρμόζεται καλά καθώς αυξάνεται ο αριθμός των online peers.

Για να περιγράψουμε τα επόμενα δύο κριτήρια που χρησιμοποιήσαμε για να αξιολογήσουμε το σύστημά μας θα ήταν εύλογο να περιγράψουμε πρώτα τι είναι το process tree όπως αναφέρεται αναλυτικά και στο [5]. Στην έρευνα που πραγματοποιήσαμε οι διεργασίες εισόδου ή εξόδου αντιπροσωπεύουν τα processes. Κάθε process p μπορεί να αναπαρασταθεί με ένα δέντρο του οποίου οι κόμβοι είναι peers και οι ακμές είναι μηνύματα που στέλνονται από γονιούς σε παιδιά. Η ρίζα του δέντρου είναι ο peer στον οποίο άρχισε η διαδικασία. Όταν μια τέτοια διαδικασία επισκεφτεί έναν peer u τότε αυτός ο peer την επεξεργάζεται τοπικά και στη συνέχεια την προωθεί σε ένα πλήθος γειτόνων στέλνοντας σε κάθε έναν από αυτούς ένα μήνυμα. Αυτοί οι peers γίνονται παιδιά του u στο process tree. Ο κάθε peer μπορεί να εμφανιστεί περισσότερες από μια φορά σε ένα τέτοιο process tree. Επίσης ο κάθε κόμβος στο process tree μιας διαδικασίας p έχει ένα βάρος και όλα τα βάρη έχουν άθροισμα 1. Το κάθε ένα από αυτά τα βάρη αντιπροσωπεύει το ποσοστό της δουλειάς που πραγματοποιείται στο συγκεκριμένο βήμα. Περισσότερες λεπτομέρειες μπορεί να βρει κανείς στο [5].

Τώρα που ορίσαμε τι είναι το process tree είναι πολύ εύκολο να ορίσουμε δύο ακόμα κριτήρια που χρησιμοποιήσαμε στη παρούσα εργασία. Σύμφωνα με το [5] είναι επίσης σημαντικό να μελετήσουμε τον μέγιστο αριθμό των καθυστερήσεων που απαιτούνται για να ολοκληρωθεί μια διεργασία εισόδου ή εξόδου, γνωστό με τον όρο latency. Ως καθυστέρηση στα πειράματα που πραγματοποιήσαμε θεωρήσαμε την αποστολή ενός μηνύματος σε κάποιον peer. Για τις διαδικασίες εισόδου αυτό το μήνυμα είναι εκείνο με το οποίο ο νέος peer ζητά να γίνει μέλος του συστήματος και το οποίο προωθείται από κάποιους online peers με σκοπό να φτάσει στον προορισμό του. Δηλαδή πρακτικά το latency μας δείχνει το μέγιστο αριθμό προωθήσεων του αρχικού μηνύματος σε δίαφορους online peers έως ότου φτάσει το μήνυμα σε εκείνον που θα αναλάβει να κάνει τον νέο peer μέλος του συστήματος P-grid. Για μια διαδικασία εξόδου αυτό το μέγεθος είναι πάντα 1. Αυτό συμβαίνει γιατί ο peer ο οποίος αρχίζει τη διαδικασία εξόδου του, στέλνει ένα μήνυμα σε κάθε έναν απο τους online peers που αναλαμβάνει κάποιο από τα φύλλα του. Το μήνυμα αυτό στέλνεται απ΄ ευθείας από τον peer που επιθυμεί να βγεί εκτός του συστήματος χωρίς να μεσολαβούν άλλοι peers , σε ένα μόνο βήμα. Δηλαδή στο process tree θα υπάρχουν πολλές ακμές που θα ξεκινούν από τη ρίζα και θα καταλήγουν η κάθε μία σε έναν μόνο κόμβο, κάτι που σημαίνει ότι το ύψος του θα είναι 1.

Τέλος χρησιμοποιήσαμε το μέγεθος traffic το οποίο μας δείχνει πόσα μηνύματα απαιτούνται για να ολοκληρωθεί μια διαδικασία εισόδου ή εξόδου καθώς μεταβάλεται ο αριθμός των online peers στο σύστημα. Θα δούμε ότι τα μηνύματα αυτά μεταβάλλονται διαφορετικά για τις διαδικασίες εξόδου και διαφορετικά για τις διαδικασίες εισόδου. Στο process tree αυτό το μέγεθος αντιπροσωπεύεται από το πλήθος των ακμών του δέντρου.

Κεφάλαιο 2

Σχετικές εργασίες

Ο τομέας στον οποίο ασχοληθήκαμε είναι σχετικός περισσότερο με τα structed P2P συστήματα τα οποία χρησιμοποιούν τοπολογίες όπως είναι το CAN - Content Addressable Network, το chord,το Tapestry κλπ... Στη συνέχεια θα περιγράψουμε στη βασική της μορφή τη τοπολογία P-grid όπως περιγράφεται στο [2] και θα δούμε κάποια σχετικά structured P2P συστήματα όπως το CAN και το Chord. Έτσι θα μπορούμε να κατανοήσουμε καλύτερα στη συνέχεια για το πρωτόκολο που μελετήσαμε, το τρόπο λειτουργίας και δρομολόγησης των διαδικασιών εισόδου και εξόδου των χρηστών προς και από το σύστημα.

2.1 P-Grid

Η τοπολογία P-Grid στη γενική της μορφή ερευνά πως θα μπορούσε να δημιουργηθεί μια δομή πρόσβασης σε ένα περιβάλλον που αποτελείται από μεγάλο αριθμό αναξιόπιστων χρηστών χωρίς κεντρικό έλεγχο, που να μπορεί ακόμα να παρέχει ένα καλό επίπεδο αξιοπιστίας για αναζήτηση και να αυξομειώνεται(scale) καλά για ένα μεγάλο αριθμό από peers και σε αποθηκευτικό χώρο και σε κόστος επικοινωνίας.

Προκειμένου το σύστημα να αποκτήσει scalability χρησιμοποιείται η προσέγγιση των δέντρων(tree structures). Για να φτιάξουμε και να χρησιμοποιήσουμε αυτές τις δομές αναζήτησης χρησιμοποιούμε randomized αλγόριθμους που βασ΄ιζονται αποκλειστικά σε τοπικές αλληλεπιδράσεις μεταξύ των peers. Η ιδέα είναι ότι δημιουργούμε τυχαίες συναντήσεις μεταξύ των peers οι οποίοι συμμετέχουν επιτυχώς στο χώρο αναζήτησης και διατηρούν αρκετή πληροφορία προκειμένου να μπορούν να επικοινωνήσουν με άλλους peers για αποτελεσματικές απαντήσεις μελλοντικών ερωτήσεων. Η κατανεμημένη δομή πρόσβασης που προκύπτει ονομάζεται P-Grid. Η έρευνα αυτή έχει διευκρινίσει ότι μια τέτοια προσέγγιση είναι εφικτή. Με τη μόνη υπόθεση ότι η κατανομή πληροφορίας δεν είναι ασύμμετρη, είναι αρκετό να χρησιμοποιήσουμε δυαδικά δέντρα αναζήτησης όπου τα κλειδιά για τα δεδομένα είναι μοναδικά για κάποια περιοχή δεδομένων.

Τα κύρια χαρακτηριστικά του P-Grid είναι τα εξής:

- Είναι τελείως αποκεντρωμένα, δεν υπάρχουν κεντρικές υποδομές, όλοι οι peers μπορούν να είναι χρήσιμοι σαν ένα σημείο εισόδου στο σύστημα και όλες οι αλληλεπιδράσεις είναι αυστηρώς τοπικές.
- Χρησιμοποιούνται randomized αλγόριθμοι για την κατασκευή της δομής πρόσβασης, για τις ενημερώσεις των δεδομένων και για τις αναζητήσεις που γίνονται και που είναι ρωμαλαίες απέναντι στις αποτυχίες των nodes.
- Κλιμακώνεται πολύ καλύτερα από αντίστοιχα συστήματα για μεγάλο αριθμό από nodes και τα συνολικά data items παρουσιάζονται στο δίκτυο εξ ίσου σημαντικά για όλους τους nodes λαμβάνοντας υπ όψην και τον αποθηκευτικό χώρο και το κόστος επικοινωνίας.

Στο σημείο αυτό θα πρέπει να περιγράψουμε τη δομή πρόσβασης και τον τρόπο κατασκευής του P-Grid έτσι ώστε να μπορούμε στη συνέχεια να περιγράψουμε με ακρίβεια την έρευνα που κάναμε και με ποιές βελτιώσεις και αλλαγές καταλήξαμε στα αποτελέσματα που προέκειψαν.

1. Δομή πρόσβασης του P-Grid:

   - Ο χώρος αναζήτησης χωρίζεται διαδοχικά σε διαστήματα- intervals της μορφής $I(k), k \in K$. Ο κάθε peer αναλαμβάνει την ευθύνη για ένα διάστημα αναζήτησης $I(k)$. Κάθε κλειδί ανταποκρίνεται σε ένα μονοπάτι στο δυαδικό δέντρο αναζήτησης και θα λέμε ότι ένας peer είναι υπεύθυνος για το μονοπάτι $I(k)$.
   - Το να αναλάβει την ευθύνη για ένα διάστημα $I(k)$ κάποιος peer σημαίνει ότι ο peer αυτός μπορεί να παρέχει στο σύστημα τις διευθύνσεις όλων των peers οι οποίοι έχουν ένα αντικείμενο πληροφορίας - data item με κλειδί k που να ανήκει στο διάστημα $I(k)$, πχ. $val(k_{query}) \in I(k)$.
   - Για κάθε πρόθεμα k_l του k μήκους $l, l = 1, 2...length(k)$ ένας peer A διατηρεί δείκτες σε άλλους peers, οι οποίοι έχουν το ίδιο πρόθεμα μήκους l αλλά διαφορετική τιμή στη θέση $l+1$ του κλειδιού για το οποίο είναι υπεύθυνοι. Αυτοί οι δείκτες σε άλλους peers ονομάζονται δείκτες του A στο επίπεδο $l + 1$. Χρησιμοποιούνται για να δρομολογήθούν οι αιτήσεις για αναζήτηση κλειδιών με ίδιο πρόθεμα k_l και που ίσως μετά από αυτό το πρόθεμα το κλειδί που αναζητείται δεν ταυτίζεται με το κλειδί για το οποίο είναι υπεύθυνοι.
   - Η αναζήτηση μπορεί να ξεκινήσει από έναν οποιονδήποτε peer.

2. Κατασκευή του P-Grid:

   Καθώς δεν υπάρχει κεντρικός έλεγχος η κατασκευή του P-Grid γίνεται χρησιμοποιώντας αποκλειστικά τοπικές αλληλεπιδράσεις μεταξύ των peers. Σύμφωνα με το [2] η γενική ιδέα είναι ότι όποτε δύο peers συναντιώνται χρησιμοποιούν τη δυνατότητα να κάνουν μια ανανέωση και έναν νέο ορισμό των σχέσεων που μπορεί να έχουν με άλλους peers . Σε αυτό τη σημείο δεν μας ενδιαφέρει γιατί μπορεί να συναντηθούν δύο peers. Μερικοί λόγοι μπορεί να είναι ότι κάποιος νέος peer εισήλθε στο σύστημα, μια αναζήτηση για την οποία έπρεπε να συναντηθούν αυτοί οι δύο peers κλπ ...
   Όπως και νά χει οι peers συναντιώνται συχνά με τυχαίο τρόπο και οι διαδικασία έχει ώς εξής:

   - Αρχικά υπάρχει ο χώρος δεδομένων αδιάσπαστος και κανένας peer δεν υπάρχει στο σύστημα.
   - Ο πρώτος peer που θα εισέλθει στο σύστημα θα γίνει υπεύθυνος όλου του χώρου δεδομένων.
   - Όταν εισέλθει ο δεύτερος peer στο σύστημα, θα επικοινωνήσει αρχικά με τον αρχικό peer , που είναι υπεύθυνος για όλο το χώρο δεδομένων και θα αποφασίσουν να χωρίσουν αυτό τον χώρο σε δύο ίσα κομμάτια από τα οποία το ένα θα το πάρει ο νέος peer. Επίσης πρέπει ο καθένας να κρατήσει ένα εισερχόμενο link - δείκτη του άλλου το οποίο θα μας δείχνει ότι ο άλλος node τον έχει ως γείτονα, καθώς επίσης και ένα εξερχόμενο link - δείκτη το οποίο θα μας δείχνει ότι και αυτός τον έχει ως γείτονα. Έτσι και οι δύο μαζί θα γνωρίζουν ο ένας τον άλλο και θα μπορούν να καλύψουν και τα δύο μισά του χώρου δεδομένων για αναζήτηση. Τα φύλλα που δημιουργούνται καθώς εισέρχονται νέοι peers αντιπροσωπεύουν τμήματα του χώρου δεδομένων και θα αναφερόμαστε σε αυτά ως nodes ή φύλλα του trie.
   - Στη συνέχεια ο τρίτος peer που θα πρέπει να γίνει μέλος του συστήματος θα διαλέξει με κάποιο τρόπο ένα node από τα δύο που υπάρχουν. Αυτός που θα επιλεγεί θα πρέπει να γίνει split και να δώσει στον νέο peer το ένα μισό, δηλαδή το ένα από τα δύο φύλλα που θα προκύψουν. Επίσης πρέπει τα δύο νέα φύλλα που δημιουργήθηκαν να αποθηκεύσουν τα απαραίτητα εισερχόμενα και εξερχόμενα links. Αυτά είναι τα links που είχε το φύλλο που έκανε split προσθέτοντας ακόμα ένα εισερχόμενο και ένα εξερχόμενο link για τη μεταξύ σχέση των δύο νέων φύλλων. Καθώς αυτή η διαδικασία συνεχίζεται το σύστημα P-Grid όλο και μεγαλώνει σε αριθμό από peers οι οποίοι είναι υπεύθυνοι για

διαφορετικές περιοχές δεδομένων, δηλαδή διαφορετικά φύλλα, τα οποία αντιπροσωπεύονται με διαφορετικά κλειδιά που κρατάνε οι peers.

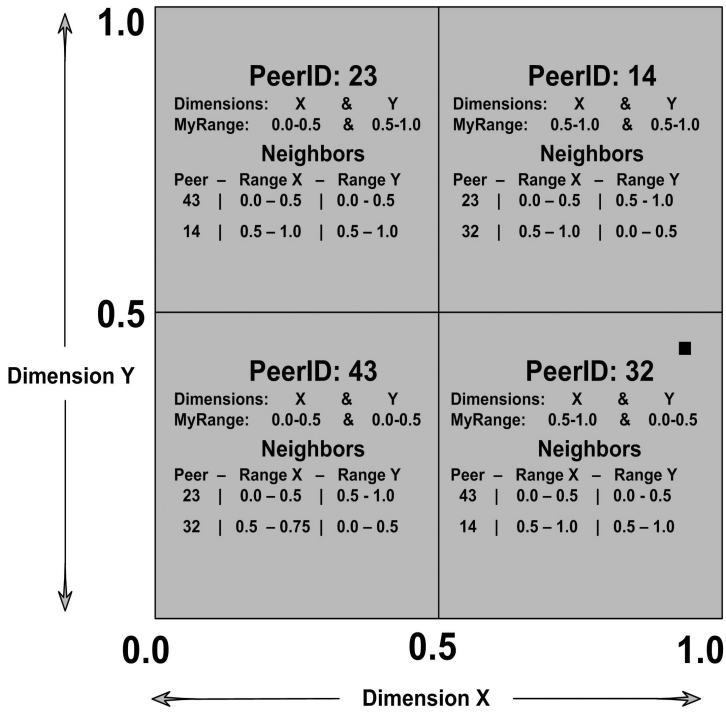
Μια σχετική έρευνα που δείχνει ότι είναι αρκετά υποσχόμενες οι τοπολογίες που χρησιμοποιούν δέντρα όπως και το P-grid περιγράφεται στο [6]. Σύμφωνα με αυτό, τα δίκτυα που βασίζονται σε δέντρα τύπου trie όπως είναι και το P-grid έχουν ένα βασικό πλεονέκτημα: η διάμετρος δρομολόγησης είναι πάντα λογαριθμική και δεν υποφέρουν από ενδογενής συνωστισμό ακόμα κι αν αλλάξει κατά πολύ το σχήμα ή το ύψος του trie. Ως διάμετρος δρομολόγησης ενός δέντρου θεωρείται η μέγιστη απόσταση που υπάρχει ανάμεσα στα δύο πιο μακρινά φύλλα. Έτσι η απόσταση αυτή θεωρείται ότι είναι τα βήματα από το ένα φύλλο στο άλλο εώς ότου φτάσουμε στον προορισμό. Στο ίδιο άρθρο αναφέρεται ότι βασιζόμενοι σε αυτό το συμπέρασμα οι επιστήμονες μπόρεσαν να αναπτύξουν ένα πλαίσιο εργασίας που μπορεί να χειριστεί γενικά τα προβλήματα των αναζητήσεων προσαρμοσμένο μέσω μιας συνάρτησης διαμελισμού του χώρου. Το αξιοπρόσεκτο χαρακτηριστικό αυτού του πλαισίου είναι ότι αποσυνδέει τη δρομολόγηση των μηνυμάτων με τα προβλήματα της αναζήτησης. Στο ίδιο άρθρο προτείνεται μια νέα στρατηγική που σκοπό έχει να εξισοροπήσει το load [1] σε ασσύμετρα προβλήματα αναζήτησης χρησιμοποιώντας περιττή επανάληψη πληροφορίας μέσω της κατάλληλης επιλογής του χώρου αναζήτησης.

2.2 Content Addressable Network (CAN)

Το CAN είναι ένα κατανεμημένο αποκεντρωμένο P2P σύστημα το οποίο είναι σχεδιασμένο να μπορεί να αυξομειώνεται, να είναι ανεκτικό σε σφάλματα και να οργανώνεται αυτόματα. Όπως γνωρίζουμε ήδη από το [7] και το [8] το αρχιτεκτονικό του σχέδιο είναι ένα εικονικό πολυδιάστατο καρτεσιανό διάστημα συντεταγμένων. Αυτό το d-διάστατο διάστημα συντεταγμένων είναι απόλυτα λογικό καθώς ο χώρος αυτός χωρίζεται δυναμικά για τους peers του συστήματος έτσι ώστε ο κάθε peer να είναι υπεύθυνος για τη δική του ξεχωριστή περιοχή που είναι τμήμα του συνολικού χώρου. Ένας peer διατηρεί ένα routing table όπου κρατάει τις διευθύνσεις IP και τις εικονικές συντεταγμένες τις ζώνης για την οποία είναι υπεύθυνος ο κάθε γείτονας. Ένα μήνυμα του CAN περιέχει τις συντεταγμένες του προορισμού. Έτσι όταν ένα μήνυμα αρχίσει τη δρομολόγηση από έναν οποιονδήποτε peer τότε χρησιμοποιώντας τις συντεταγμένες των γειτόνων, ο peer προωθεί το μήνυμα προς τον προορισμό στέλνοντας το μήνυμα σε εκείνον τον γείτονα που είναι πιο κοντά στις συντεταγμένες του προορισμού. Η απόδοση για τη δρομολόγηση στο CAN είναι $O(d.N^{1/d})$. Ο εικονικός χώρος συντεταγμένων χρησιμοποιείται για να αποθηκευτούν τα ζευγάρια $\langle key, value \rangle$ ως εξής: Για να καταχωρήσουμε το ζευγάρι $\langle K, V \rangle$ το κλειδί K αντιστοιχίζεται σε ένα σημείο R στο χώρο χρησιμοποιώντας μια ομοιόμορφη hash function. Το πρωτόκολο αναζήτησης προκειμένου να ανακτήσει μια καταχώρηση που ανταποκρίνεται στο κλειδί K χρησιμοποιεί την ίδια hash function σε κάθε peer για να αντιστοιχίσει το κλειδί K στο σημείο R και στη συνέχεια να ανακτήσει τη τιμή V από το σημείο R. Αν το σημείο R δεν ανήκει στη περιοχή του peer που κάνει την ερώτηση ή των γειτόνων του, τότε η ερώτηση πρέπει να προωθηθεί χρησιμοποιώντας το πρωτόκολο του CAN εώς ότου φτάσει σε εκείνον τον peer που να έχει την ζητούμενη περιοχή στην οποία ανήκει το σημείο R. Βλέπουμε δηλαδή ότι τα routing tables των peers χρησιμοποιούνται προκειμένου να έχουμε μια επαρκής δρομολόγηση μεταξύ τυχαίων σημείων στο συνολικό χώρο συντεταγμένων. Στο σχήμα 2.1 βλέπουμε μια εικόνα που θα μπορούσε να έχει μια τοπολογία CAN δύο διαστάσεων με 4 online peers καθώς και τα routing tables που θα πρέπει να διατηρούν οι peers προκειμένου να μπορούν να δρομολογήσουν ένα μήνυμα προς οποιονδήποτε προορισμό.

Σύμφωνα με το [7] όταν ένας νέος peer θέλει να γίνει μέλος του συστήματος πρέπει να αποκτήσει με κάποιο τρόπο το δικό του μερίδιο από το συνολικό χώρο συντεταγμένων. Αυτό γίνεται εφικτό αν κάνουμε split, δηλαδή αν χωρίσουμε σε δύο ίσα κομμάτια τη περιοχή κάποιου υπάρχοντος peer και στη συνέχεια δώσουμε το ένα μισό στον καινούργιο peer διατηρώντας το άλλο μισό στον αρχικό peer της περιοχής. Το σύστημα CAN διατηρεί ένα σχετικο DNS πεδίο ορισμού στο οποίο υπάρχουν ονόματα που αντιστοιχούν στις διευθύνσεις IP των peers που υπάρχουν

[1] Η δυνατότητα ομοιόμορφου διαμοιρασμού της απαιτούμενης εργασίας στους peers



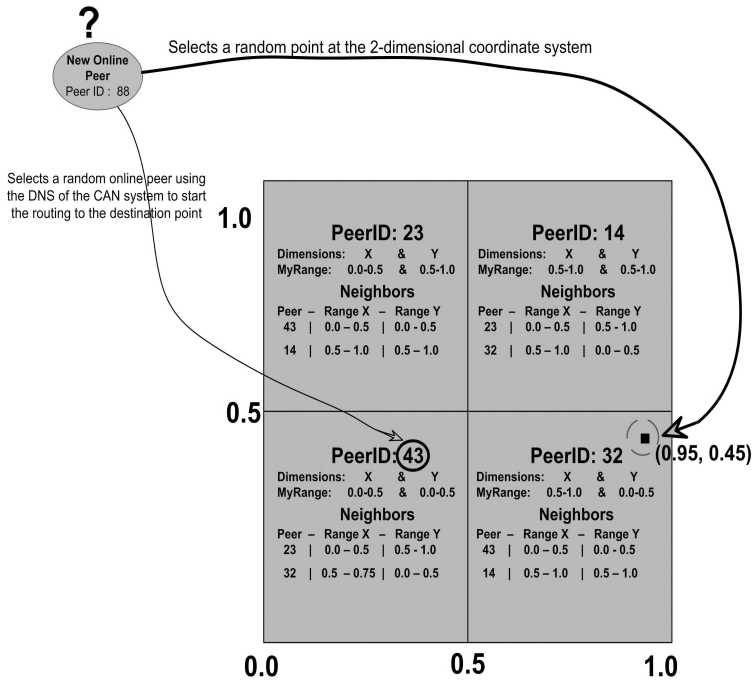
Σχήμα 2.1: Κατάσταση ενός συστήματος CAN δύο διαστάσεων με 4 online peers.

ήδη online στο σύστημα. Προκειμένου κάθε φορά να βρεθεί εκείνη η περιοχή που θα γίνει split ο νέος peer χρησιμοποιεί το DNS για να βρει την διεύθυνση IP ενός τυχαίου αρχικού peer που θα τον ονομάζουμε bootstrap. Ο bootstrap παρέχει κάποιες IP διευθύνσεις κάποιων τυχαίων peers που είναι σίγουρο ότι είναι online στο σύστημα. Ο νέος peer επιλέγει τυχαία ένα σημείο R και στέλνει στον bootstrap μια αίτηση προκειμένου να γίνει μέλος (=JOIN request) της περιοχής στην οποία ανήκει το σημείο R. Κάθε peer χρησιμοποιεί τον μηχανισμό δρομολόγησης του CAN εώς ότου φτάσει η αίτηση στον peer ο οποίος είναι υπεύθυνος για την περιοχή στην οποία ανήκει το σημείο R. Αυτός ο peer θα χωρίσει την περιοχή του σε δύο ίσα κομμάτια και θα αναθέσει το ένα μισό στον νέο peer. Αφού αποκτήσει τη δική του περιοχή ο νέος peer , μαθαίνει τις διευθύνσεις και τις συντεταγμένες της περιοχής των γειτόνων του και προσθέτει ακόμα τον peer που έκανε split την περιοχή του ως γείτονα. Η εισαγωγή ενός νέου peer επηρεάζεται μόνο από έναν μικρό αριθμό από peers ο οποίος εξαρτάται μόνο από τη διάσταση του χώρου συντεταγμένων και είναι ανεξάρτητος από τον συνολικό αριθμό των online peers. Στο σχήμα βλέπουμε πως αρχίζει μια διαδικασία εισόδου ενός νέου peer στο σύστημα CAN.

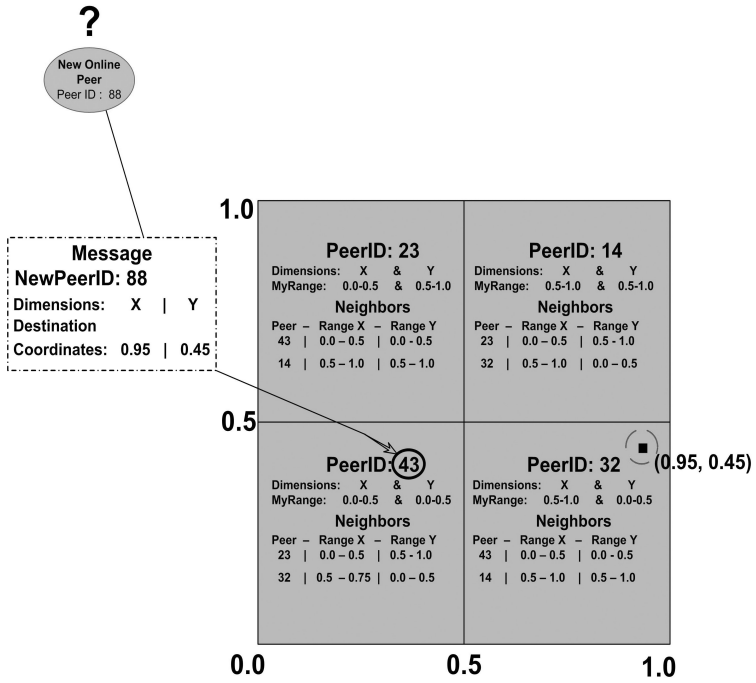
Όταν ένας peer βγαίνει offline πρέπει με κάποιο τρόπο να αναλάβει κάποιος άλλος online peer να είναι υπεύθυνος για την περιοχή του. Σύμφωνα με το [7] ο πιο απλός και φυσιολογικός τρόπος για να γίνει αυτό είναι μέσω κάποιου γείτονα. Αν η περιοχή κάποιου γείτονα μπορεί να ενωθεί με τη περιοχή του peer που έφυγε και να προκύψει μια έγκυρη μοναδική περιοχή τότε αυτό γίνεται. Αν όχι τότε τη περιοχή αυτή την αναλαμβάνει ο γείτονας που έχει τη μικρότερη περιοχή για την οποία να είναι υπεύθυνος. Στα σχήματα 2.6 και 2.7 βλέπουμε ένα παράδειγμα για την κάθε περίπτωση που μπορεί να συμβεί. Στο παράδειγμα που περιγράψαμε για τη διαδικασία εισόδου θα υποθέσουμε ότι μπήκαν ακόμα δύο peers , ο 36 και ο 78 και η εικόνα του συστήματος θα είναι όπως στο σχήμα 2.6.

Έστω τώρα ότι ο χρήστης 36 βγήκε offline. Βλέπουμε στο σχήμα 2.6 τη περίπτωση όπου η περιοχή του peer 36 που αποχωρεί από το δίκτυο μπορεί να ενωθεί (να γίνει merged) με τη περιοχή κάποιου γείτονα, δηλαδή στη περίπτωσή μας με τη περιοχή του peer 14.

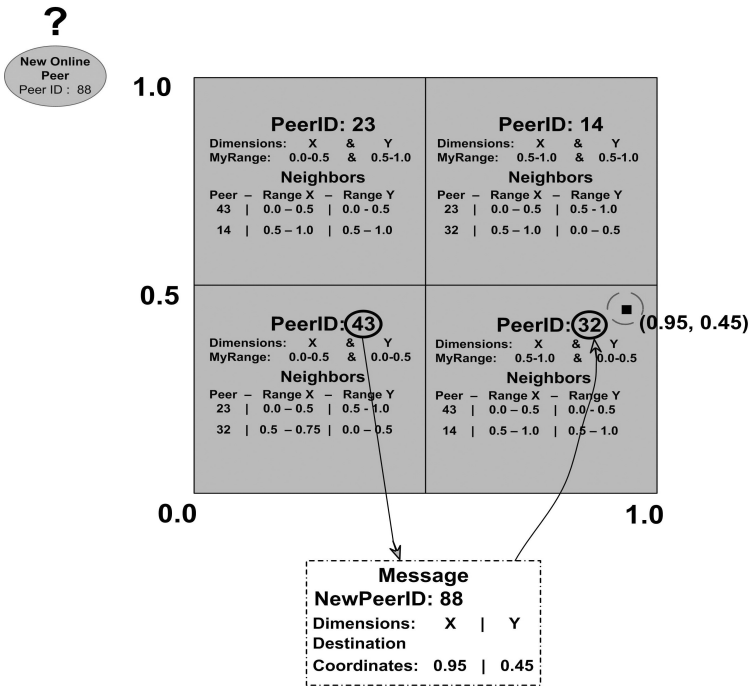
Σύμφωνα με το ίδιο άρθρο αν η περιοχή του peer που βγήκε offline δεν μπορεί να ενωθεί με τη περιοχή κάποιου γείτονα πρέπει κάποιος από τους γείτονες να αναλάβει τη περιοχή και στο σχήμα 2.7 βλέπουμε μια τέτοια περίπτωση.Επιλέγεται



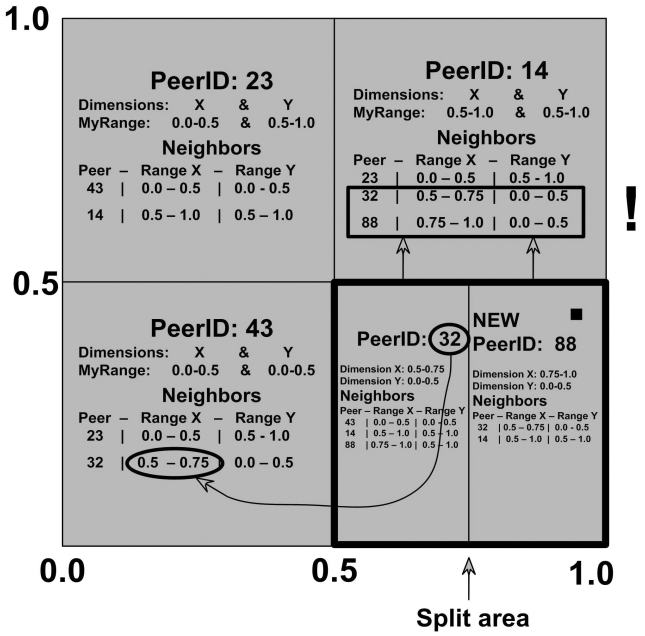
Σχήμα 2.2: Έναρξη εισαγωγής ενός νέου peer στο σύστημα CAN δύο διαστάσεων με 4 online peers. Ο νέος peer επιλέγει τυχαία ένα σημείο στο χώρο και έναν τυχαίο online peer από τον οποίο θα αρχίσει η δρομολόγηση προς το σημείο προορισμό.



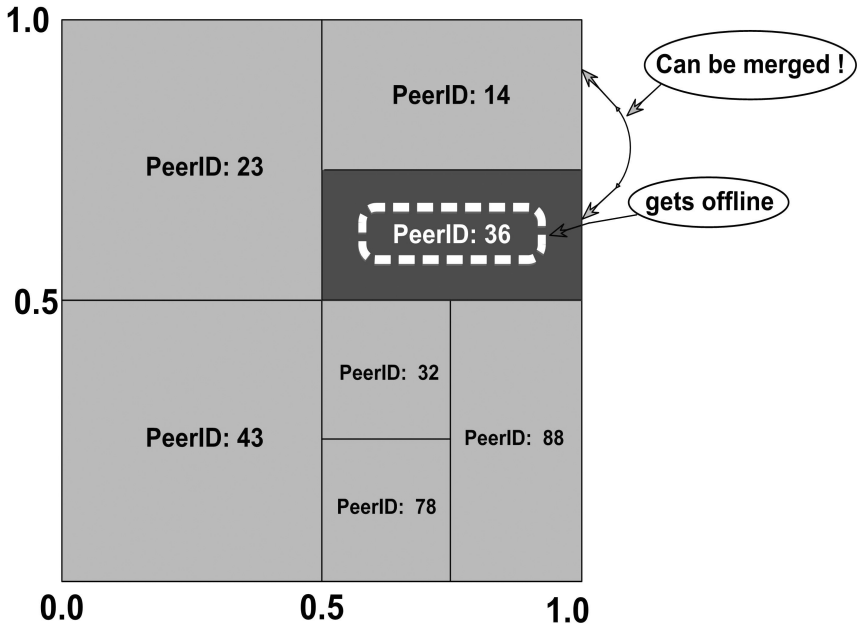
Σχήμα 2.3: Ο νέος peer στέλνει ένα μήνυμα στον τυχαίο peer 43 στο οποίο μήνυμα θα περιγράφονται οι συντεταγμένες του σημείου προορισμού στο οποίο πρέπει να φτάσει το μήνυμα καθώς και άλλα στοιχεία όπως είναι η IP διεύθυνση του νέου peer.



Σχήμα 2.4: Ο peer 43 ελέγχει ποιος από τους γείτονές του έχει εύρος περιοχής τέτοιο ώστε να περιέχει το σημείο προορισμό ή να είναι τουλάχιστον πιο κοντά στο ζητούμενο σημείο. Στη συνέχεια προωθεί το μήνυμα σε εκείνον τον γείτονα.



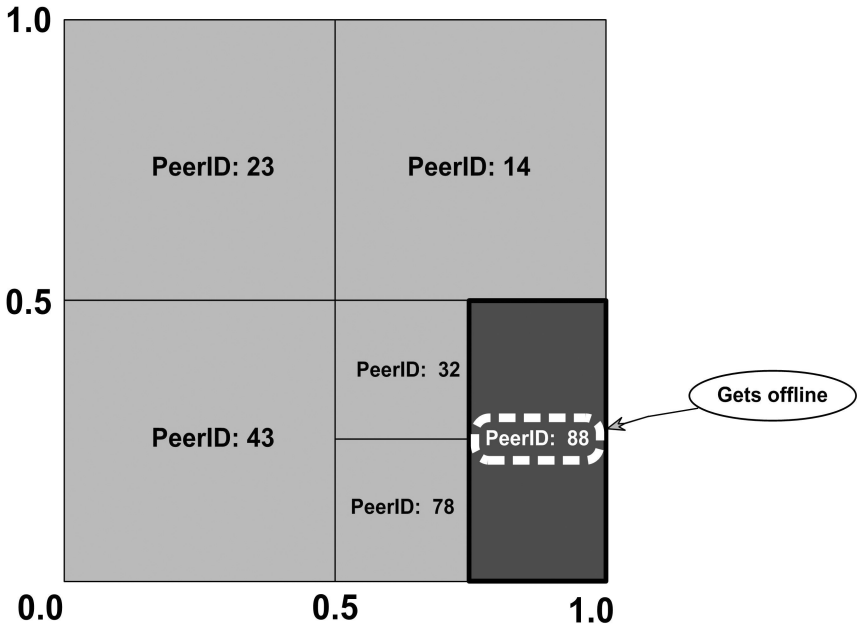
Σχήμα 2.5: Ο peer 43 έχει χωρίσει πλέον τη περιοχή του και έχει δώσει το ένα μισό στον νέο peer 88. Ενημερώθηκαν οι γείτονες 14 και 43 για αυτήν την αλλαγή και πλέον ο peer 88 έχει τη δική του περιοχή.



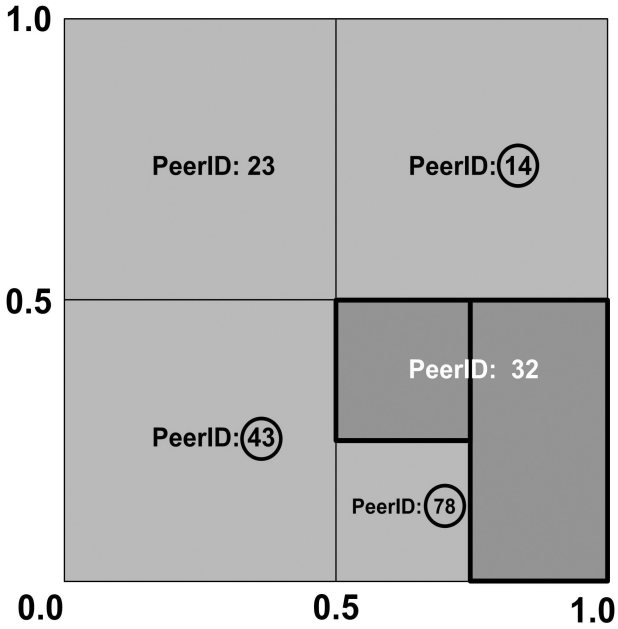
Σχήμα 2.6: Ο peer 36 έχει πλέον φύγει από το σύστημα και σε αυτή τη περίπτωση μπορεί να ενωθεί η περιοχή του με τη περιοχή του χρήστη 14.

πάντα εκείνος ο γείτονας του οποίου η περιοχή είναι η μικρότερη σε σχέση με των υπολοίπων. Αν τύχει και δύο γείτονες έχουν ίδιο μέγεθος περιοχής τότε επιλέγεται κάποιος τυχαίος. Έστω λοιπόν στα παράδειγμά μας ότι βγαίνει offline ο peer 88 και επιλέγεται ο γείτονας 32 για να αναλάβει τη περιοχή του 88.

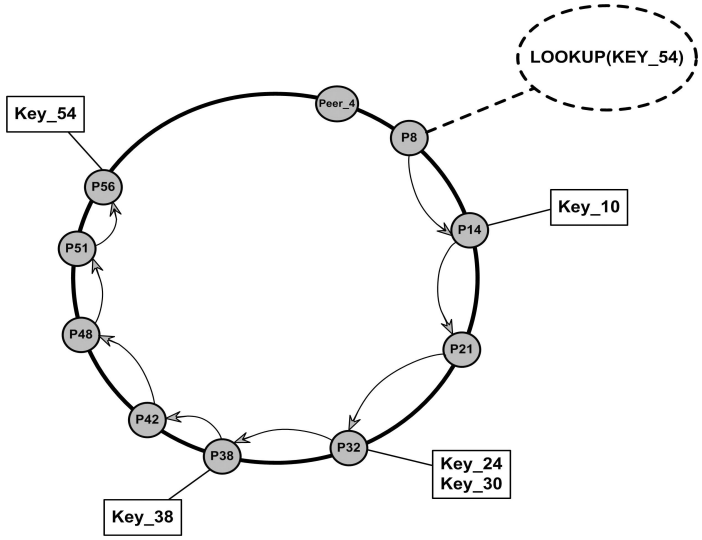
Περιγράψαμε μέσα σε γενικές γραμμές τις βασικές λειτουργίες εισόδου και εξόδου των peer του συστήματος CAN που είναι ένα P2P σύστημα αρκετά σχετικό με το σύστημα P-Grid που υλοποιήσαμε. Σύμφωνα με το [7] το σύστημα CAN έχει ένα βασικό πλεονέκτημα: όταν φεύγει ένας peer τα μηνύματα που θα στείλουν οι γείτονες προκειμένου να αναλάβουν τη περιοχή του είναι πολύ λίγα και εξαρτώνται αποκλειστικά και μόνο από τη διάσταση του συστήματος συντεταγμένων και όχι από το μέγεθος του δικτύου. Ας δούμε στη συνέχεια ένα ακόμα P2P σύστημα που είναι αρκετά σχετικό με την έρευνα που πραγματοποιήσαμε, το σύστημα chord.



Σχήμα 2.7: Ο peer 88 έχει πλέον φύγει από το σύστημα και σε αυτή τη περίπτωση δεν μπορεί να γίνει merge δηλαδή ένωση με τη περιοχή κάποιου γείτονα.



Σχήμα 2.8: Ο peer 32 είναι πλέον υπεύθυνος για δύο διαφορετικές περιοχές, τη δική του και τη περιοχή του peer 88 που βγήκε offline.

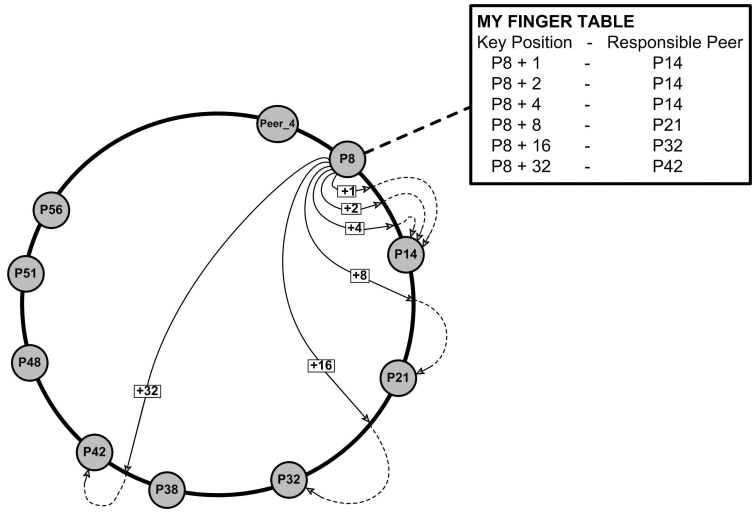


Σχήμα 2.9: Ενα σύστημα chord που αποτελείται από δέκα peers και πέντε data keys. Βλέπουμε το μονοπάτι που ακολουθείται από μια ερώτηση που ξεκίνησε στον peer 8 για την αναζήτηση του κλειδιού 54.

2.3 Chord

Σύμφωνα με το [8] και [9] το σύστημα P2P chord είναι μια τοπολογία για P2P συστήματα η οποία χρησιμοποιεί συνεχής διασπάσεις (hashing) προκειμένου να αντιστοιχίσει τους κόμβους πάνω σε ένα m-bit κυκλικό χώρο. Ο κάθε κόμβος αντιστοιχίζεται σε ένα όνομα-identifier. Κάθε identifier a αντιστοιχίζεται στον κόμβο ο οποίος έχει τον μικρότερο identifier από εκείνους που είναι μεγαλύτεροι ή ίσοι του a στον κυκλικό χώρο. Αυτές οι συνεχείς διασπάσεις είναι σχεδιασμένες έτσι ώστε να επιτρέπουν στους peers να μπαίνουν και να βγαίνουν από το δίκτυο με ελάχιστο κόστος.Αυτό το αποκεντρωμένο σύστημα τείνει να ισσοροπεί το load στο σύστημα αφού ο κάθε peer διατηρεί σχεδόν τον ίδιο αριθμό από κλειδιά και γίνεται ελάχιστη κίνηση κλειδιών καθώς οι peers εισέρχονται και εξέρχονται από το σύστημα. Σε μια σταθερή κατάσταση, για N αριθμό από peers, κάθε peer διατηρεί πληροφορίες για τη δρομολόγηση μηνυμάτων για περίπου $O(logN)$ άλλους peers. Αυτό είναι πολύ χρήσιμο αλλά η απόδοση μειώνεται όταν τα δεδομένα θεωρούνται πλέον παλιά και έχουν λήξει. Οι συναρτήσεις για hashing αναθέτουν peers και κλειδιά σε έναν m-bit κυκλικό χώρο. Κάθε identifier των peers επιλέγεται κάνοντας hashing την IP διεύθυνσή του και ένα identifier - κλειδί παράγεται κάνοντας hashing το κλειδί του data. Το μήκος ενός identifier m πρέπει να είναι αρκετά μεγάλο για να ελαχιστοποιήσουμε την πιθανότητα το hashing ενός κλειδιού να παράγει τον ίδιο identifier με κάποιο άλλο. Οι identifiers είναι ταξινομημένοι πάνω στον κυκλικό χώρο modulo 2m. Κάθε κλειδί k ανατίθεται στον πρώτο peer του οποίου ο identifier είναι ίσος ή ακολουθείται από k στον χώρο των identifiers. Αυτός ο peer λέγεται successor peer του κλειδιού k και γράφεται ως $successor(k)$. Αν οι identifiers αναπαριστώνται ως έναν κύκλο από αριθμούς 0 εώς το $2m-1$, τότε ο $successor(k)$ είναι ο πρώτος peer ακολουθώντας τη φορά του ρολογιού απο το κλειδί k. Για να διατηρήθεί η αντιστοίχιση μετά τα συνεχόμενα hashing όταν ένας peer n γίνεται μέλος του συστήματος, συγκεκριμένα κλειδιά που είχαν ανατεθεί στον successor του n τώρα πρέπει να ανατεθούν στον n. Όταν ο peer n βγει εκτός του συστήματος Chord, όλα τα κλειδιά που του είχαν ανατεθεί πρέπει να ξαναεπιστρέψουν στον successor του n. Γι΄ αυτό οι peers εισέρχονται και εξέρχονται από το σύστημα σε χρόνο $(logN)^2$. Δεν χρειάζονται άλλες αλλαγές για ανάθεση κλειδιών σε άλλους peers. Στο παρακάτω σχήμα υπάρχει ένας κύκλος Chord όπου $m = 6$. Σε αυτό το συγκεκριμένο παράδειγμα που περιγράφηκε και στο [8] υπάρχουν 10 peers που έχουν αποθηκεύσει 5 κλειδιά. Ο successor του identifier 10 είναι ο peer 14, έτσι το κλειδί 10 θα βρίσκεται στον peer 14. Παρόμοια αν ένας peer που πρόκειται να γίνει μέλος έχει ως identifier το 26, θα πάρει το κλειδί που έχει ως identifier το 24 από τον χρήστη 32.

Κάθε peer στο σύστημα Chord ξέρει πως να επικοινωνήσει με τον δικό του



Σχήμα 2.10: Ενα σύστημα chord που αποτελείται από δέκα peers και πέντε data keys. Βλέπουμε το μονοπάτι που ακολουθείται από μια ερώτηση που ξεκίνησε στον peer 8 για την αναζήτηση του κλειδιού 54.

successor στο συνολικό κυκλικό χώρο. Οι ερωτήσεις αναζήτησης περιέχουν την αντιστοίχιση του κλειδιού και του peer που πρέπει να είναι υπεύθυνος γι΄ αυτό το κλειδί. Στο σχήμα 2.9 βλέπουμε μια αναζήτηση στην οποία ο peer 8 ψάχνει να βρει το κλειδί 54. Η ερώτηση περνάει από κάθε peer του κύκλου με identifier μεταξύ 8 και 56. Η απάντηση επιστρέφεται κατά μήκος τις ίδιας διαδρομής αλλά ανάστροφα.

Σύμφωνα με το ίδιο άρθρο αφού το m είναι ο αριθμός των bits στο χώρο κλειδιών/NodeIDs , κάθε peer n διατηρεί έναν πίνακα δρομολόγησης (routing table) ο οποίος έχει m εγγραφές και ονομάζεται finger table. Η i -οστή εγγραφή του πίνακα ενός peer n περιέχει την ταυτότητα του πρώτου peer s που ακολουθείται κατά $n+2^{i-1}$ θέσεις στον κυκλικό χώρο. Για παράδειγμα $s = successor(n+2^{i-1})$, όπου $1 \leq i \leq m$. Ο peer s είναι ο i -οστός finger του peer $n(s = n.finger[i])$. Μια εγγραφή στο finger table περιέχει και τον identifier για το σύστημα Chord και την διεύθυνση IP του σχετικού peer. Στο σχημα 2.10 φαίνεται το finger table του peer 8 όπου m=6 bits. Η πρώτη εγγραφή δείχνει στον peer 14 αφού $14 = successor(8 + 2^{1-1})$. Η τελευταία εγγραφή δείχνει όπως ήταν αναμενόμενο στον identifier 42 καθώς ισχύει ότι $42 = successor(8 + 2^{6-1} = successor(40))$, δηλαδή ο 42 είναι ο πρώτος identifier που διαδέχεται το κλειδί 40. Με αυτό το τρόπο οι peers αποθηκεύουν πληροφορίες για έναν μικρό αριθμό από peers και ξέρουν περισσότερα για διαδοχικούς peers που βρίσκονται κοντά τους στον κυκλικό χώρο. Επίσης ένα finger table δεν περιέχει αρκετή πληροφορία για να αποφασίσει αμέσως ποιος είναι ο successor ενός αμφιλεγόμενου κλειδιού k. Για παράδειγμα ο peer 8 δεν μπορεί να αποφασίσει από μόνος του ποιος είναι ο successor του κλειδιού 34 γιατί ο successor αυτού του κλειδιού είναι ο 38 ο οποίος δεν βρίσκεται στο finger table του peer 8.

Όταν ένας peer γίνεται μέλος του συστήματος, σύμφωνα με το [8] και το [9] τα finger tables κάποιων peers τα οποία περιέχουν δείκτες σε successors πρέπει να αλλάξουν. Είναι σημαντικό οι δείκτες προς τους successors να είναι έγκυροι και πρόσφατοι κάθε στιγμή γιατί διαφορετικά δεν μπορεί να εγγυηθεί η αποτελεσματική αναζήτηση. Το πρωτόκολο Chord χρησιμοποιεί ένα πρωτόκολο σταθεροποίησης το οποίο εκτελείται περιοδικά στο παρασκήνιο προκειμένου να ενημερώνει τα finger tables. Η ορθότητα του πρωτοκόλου βασίζεται στο γεγονός ότι ο κάθε peer είναι πάντα ενήμερος για τους successors που πρέπει να ξέρει. Όταν ένας peer δεν είναι πλέον έγκυρος είναι πιθανό κάποιος να μην ξέρει τον νέο successor και να μην μπορεί να τον μάθει. Για να αποφύγουμε αυτή τη κατάσταση, ο κάθε peer διατηρεί μια λίστα από successors μεγέθους r , η οποία λίστα περιέχει τους πρώτους r successors. Όταν ένας successor peer δεν ανταποκρίνεται τότε ο peer ο οποίος τον ψάχνει επικοινωνεί με τον επόμενο peer στη λίστα των successors. Αν υποθέσουμε ότι η αποτυχία κάποιου peer συμβαίνει με πιθανότητα p τότε η πιθανότητα να αποτύχουν όλοι οι peers της λίστας είναι p^r. Αυξάνοντας τον αριθμό r κάνουμε αυτόματα το σύστημα πιο ευέλικτο, σταθερό και ανθεκτικό σε σφάλματα.

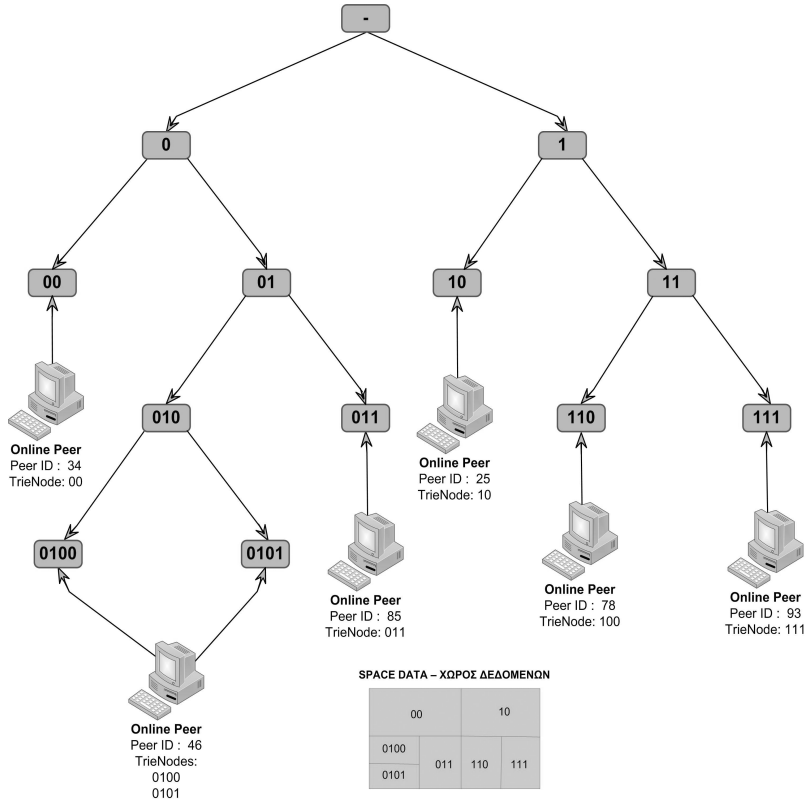
Κεφάλαιο 3

Αναλυτική περιγραφή του πρωτοκόλου

Το αντικείμενο της εργασίας μας ήταν να προσομοιώσουμε ένα βελτιωμένο πρωτόκολο P-Grid σε ένα περιβάλλον όπου οι χρήστες πραγματοποιούν χρονικά τυχαίες εισόδους και εξόδους προς και από το δίκτυο εντελώς δυναμικά και με τυχαίο τρόπο. Θα δούμε πως συμπεριφέρεται το σύστημα για διάφορα κριτήρια που μας είναι χρήσιμα για να αποφανθούμε πόσο καλά προσαρμόζεται το σύστημα για τις διαδικασίες εισόδου και εξόδου των peers , πόσο γρήγορα γίνεται μια διαδικασία εισόδου ή εξόδου στο σύστημα και πόσα μηνύματα απαιτούνται να στείλουν οι peers μεταξύ τους για μια τέτοια διαδικασία. Αυτά τα μηνύματα είναι απαραίτητα προκειμένου να επικοινωνήσουν για να αποφανθούν οι peers για τα links που πρέπει να τους συνδέουν αλλά και για τα δεδομένα για τα οποία πρέπει ο καθένας να είναι υπεύθυνος.

Ας πάρουμε όμως τα πράγματα από την αρχή και ας δούμε πως ξεκινάει και από τι αποτελείται η προσομοίωση που πραγματοποιήσαμε. Αρχικά έχουμε ένα σύνολο δεδομένων τα οποία θα υποθέσουμε ότι διατηρούν οι peers. Τα δεδομένα που χρησιμοποιήσαμε στη πράξη είναι ένα σύνολο από σημεία στον δισδιάστατο χώρο που όλα μαζί αποτελούν τον γεωγραφικό χώρο της Ελλάδας στον χάρτη. Το κάθε σημείο στον δισδιάστατο χώρο αποτελείται από δύο συντεταγμένες (x, y). Ο συνολικός χώρος αναζήτησης είναι ένα σύστημα συντεταγμένων δύο διαστάσεων μεγέθους $N * N$ το οποίο σύστημα περιέχει όλα τα σημεία. Κάθε σημείο αυτού του δισδιάστατου χώρου θεωρούμε ότι είναι ένα αντικείμενο δεδομένων και αντιστοιχίζεται σε ένα κλειδί k.

Το P-grid όπως έχουμε ήδη περιγράψει είναι μια τοπολογία που αποτελείται από ένα δυαδικό δέντρο αναζήτησης, δηλαδή ένα trie. Το trie θα το χρησιμοποιήσουμε για να αντιστοιχίσουμε τα δεδομένα αλλά και το σύνολο του δισδιάτατου χώρου δεδομένων στα φύλλα του. Όταν θα λέμε ότι δύο φύλλα του trie είναι υπεύθυνα για όλο το χώρο δεδομένων θα εννοούμε ότι έχουμε χωρίσει τον χώρο των δεδομένων μας σε δύο ίσους δισδιάστατους χώρους και ότι το ένα φύλλο είναι υπεύθυνο για το ένα μισό και το άλλο φύλλο για το άλλο μισό. Έτσι όταν ένα φύλλο του trie αναζητεί ένα κλειδί που αντιστοιχεί σε κάποιο αντικείμενο δεδομένων, δηλαδή τελικά κάποιο σημείο του δισδιάστατου χώρου, θα θεωρούμε ότι έχει βρει το κλειδί όταν επικοινωνήσει με εκείνο το φύλλο του trie που θα είναι υπεύθυνο για το τμήμα του δισδιάστατου χώρου στο οποίο ανήκει το κλειδί. Το κάθε φύλλο του trie θα πρέπει να διατηρεί δείκτες προς κάποια άλλα φύλλα όπως έχουμε περιγράψει ήδη στο κεφάλαιο 2.1. Εκείνα τα φύλλα θα είναι οι γείτονες του εκάστοτε φύλλου και θα λέμε ότι ένα φύλλο διατηρεί εξερχόμενο link προς κάποιο άλλο φύλλο. Καθώς μεγαλώνει το trie γίνονται split διάφορα φύλλα προκειμένου να δημιουργηθούν νέα φύλλα. Το κάθε φύλλο του trie θα πρέπει να γνωρίζει ποια άλλα φύλλα το έχουν ως γείτονα έτσι ώστε όταν γίνει split να τα ενημερώσει και να αντικαταστήσουν το φύλλο που έκανε split και που το είχαν στον πίνακα των εξερχομένων links, με κάποιο από τα δύο νέα φύλλα που δημιουργήθηκαν από το split. Άρα το κάθε φύλλο του trie πρέπει να διατηρεί δύο ήδη γειτόνων: τα φύλλα που έχει ως γείτονες και τα φύλλα που το έχουν ως γείτονα. Θα θεωρήσουμε ότι αυτά βρίσκονται σε δύο πίνακες, των πίνακα των εξερχομένων links και τον πίνακα των εισερχομένων links αντίστοιχα. Όταν λέμε ότι τα φύλλα πρέπει να γνωρίζουν κάποια άλλα

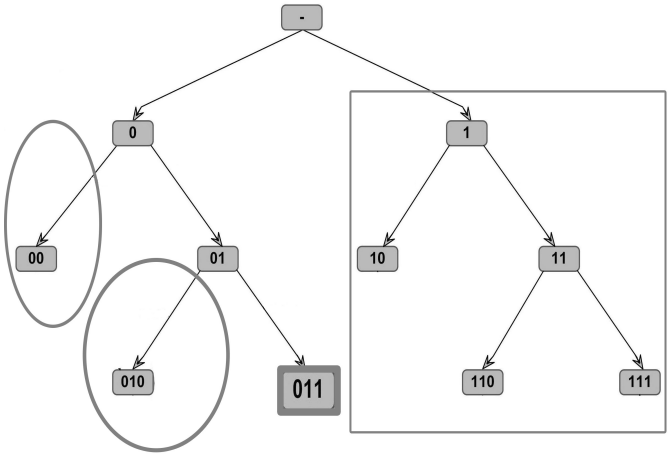


Σχήμα 3.1: Παράδειγμα ενός συτήματος P-grid που αποτελείται από 5 online χρήστες και με ένα trie που αποτελείται από 6 φύλλα.

φύλλα είναι σαν να λέμε ότι οι διάφορες περιοχές του δισδιάστατου χώρου πρέπει να γνωρίζουν κάποιες άλλες περιοχές του χώρου δεδομένων. Έτσι όταν αρχίσει κάποια αναζήτηση σε κάποιο τμήμα του χώρου δεδομένων θα μπορούμε να δρομολογήσουμε την αναζήτηση προς το τμήμα του χώρου δεδομένων στον οποίο περιέχεται το σημείο που αναζητούμε χρησιμοποιώντας αυτές τις σχέσεις μεταξύ των τμημάτων του χώρου. Σε επίπεδο trie αυτό σημαίνει ότι όταν η αναζήτηση κάποιου κλειδιού που αντιπροσωπεύει ένα σημείο του χώρου δεδομένων, αρχίσει σε κάποιο φύλλο του trie, τότε το φύλλο θα πρέπει να δρομολογήσει αυτήν την αναζήτηση μέσω των γειτόνων του προς εκείνο το φύλλο που είναι υπεύθυνο για τον χώρο στον οποίο ανήκει το σημείο που αντιπροσωπεύεται από το κλειδί που αναζητείται.

Σε αυτό το σημείο είμαστε έτοιμοι να περιγράψουμε με ποιο τρόπο συνδέονται όλα τα παραπάνω με τους πραγματικούς χρήστες. Κάθε χρήστης υποθέτουμε ότι έχει έναν υπολογιστή, ένα μηχάνημα δηλαδή με το οποίο συνδέεται στο internet. Ο κάθε υπολογιστής θεωρούμε ότι είναι ένας peer για το σύστημά μας. Πρακτικά όταν κάποιος peer θέλει να γίνει μέλος ενός συστήματος P-grid, σημαίνει ότι θέλει να χρησιμοποιήσει τους πόρους του συστήματός του προκειμένου να πληρώσει τη συνεισφορά του αλλά και να μπορεί να πραγματοποιεί αναζητήσεις δεδομένων χρησιμοποιώντας το σύστημα P-grid. Οι peers συνδέονται με το trie άμεσα αφού ο καθένας απο αυτούς πρέπει να αναλάβει να αποθηκεύσει ένα ή περισσότερα φύλλα. Όταν αποκτήσει κάποιο φύλλο του trie και γίνει μέλος του συστήματος θα μπορεί να αναζητήσει οποιοδήποτε αντικείμενο δεδομένων. Αυτό θα μπορεί να το κάνει χρησιμοποιώντας το φύλλο ή τα φύλλα για τα οποία θα είναι υπεύθυνος. Η αναζήτηση κάποιου κλειδιού θα αρχίζει σε κάποιο φύλλο για το οποίο θα είναι υπεύθυνος και θα δρομολογείται μέσω των εξερχομένων links σε άλλα φύλλα που θα ανήκουν πιθανόν σε άλλους peers εώς ότου φτάσει σε εκείνο το φύλλο που θα είναι υπεύθυνο για την περιοχή του χώρου δεδομένων στην οποία θα ανήκει το κλειδί που αναζητείται.

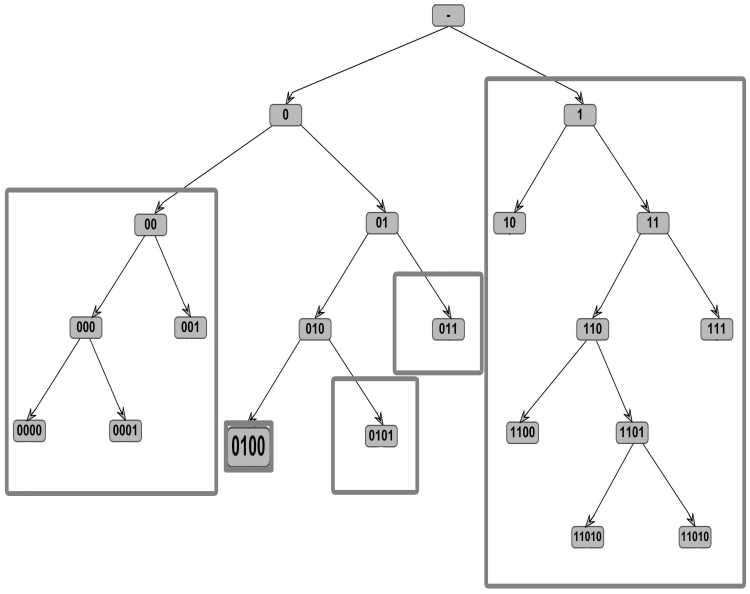
Όταν κάποιος peer επιθυμεί να βγει εκτός του συστήματος θα πρέπει να αρχίσει τη διαδικασία εξόδου του. Όταν γίνει αυτή η αίτηση για έξοδο θα πρέπει το πρωτόκολο να αναλάβει να βρει online peers που θα αναλάβουν τα φύλλα για τα οποία ήταν υπεύθυνος ο peer που επιθυμεί να βγει εκτός. Το πρωτόκολο θα βρει



Σχήμα 3.2: Υποδέντρα προς τα οποία πρέπει να διατηρεί κάποιο εξερχόμενο link το φύλλο 011.

τυχαίους online peers που θα αναλάβουν να είναι υπεύθυνοι για κάθε ένα από τα φύλλα που είχε ο peer που επιθυμεί να βγει εκτός. Ο κάθε ένας απο τους τυχαίους online peers που αναλαμβάνουν κάποιο φύλλο πρέπει να ενημερώνει τους υπεύθυνους peers των γειτόνων του φύλλου προκειμένου να τους ενημερώσουν ότι το φύλλο πλέον έχει νέο υπεύθυνο peer. Πιο αναλυτικά η διαδικασία εξόδου περιγράφεται στο κεφάλαιο 3.3.

Ας ξεκινήσουμε όμως να περιγράφουμε με μεγαλύτερη λεπτομέρεια θέματα που αφορούν τη τοπολογία του P-grid. Στο σχήμα 3.2 μπορούμε να παρατηρήσουμε πρακτικά αυτό που περιγράφεται στο [2] για τη περιγραφή των γειτόνων-links που πρέπει να διατηρούν τα φύλλα του trie στη τοπολογία του P-Grid. Για κάθε πρόθεμα k_l του k μήκους $l, l = 1, 2...length(k)$ ένα φύλλο A διατηρεί δείκτες προς άλλα φύλλα τα οποία έχουν το ίδιο πρόθεμα μήκους l αλλά διαφορετική τιμή στη θέση $l + 1$ του κλειδιού τους. Δηλαδή στη πράξη αυτό σημαίνει ότι το φύλλο 011 πρέπει να έχει κάποιο εξερχόμενο link προς κάποιο φύλλο στο υποδέντρο 1*, στο υποδέντρο 00* και στο υποδέντρο 010*.



Σχήμα 3.3: Τα υποδέντρα που είναι μέσα στα πλαίσια δηλώνουν αυτά προς τα οποία πρέπει να διατηρεί κάποιο εξερχόμενο link το φύλλο 0100.

Στο σχήμα 3.3 παρατηρούμε ότι το φύλλο 0100 πρέπει σύμφωνα με το [2] να διατηρεί ένα εξερχόμενο link προς κάθε ένα από τα εξής υποδέντρα:

- 1*
- 00*
- 011*
- 0101*

Με αυτό το τρόπο παρατηρούμε ότι τα φύλλα του trie διατηρούν γείτονες σε διάφορα υποδέντρα και άρα μπορούν με ένα προοδευτικό τρόπο να προωθούν την αναζήτηση κάποιου κλειδιού προς άλλα φύλλα που θα είναι πιο κοντά σε εκείνο το φύλλο που θα αντιπροσωπεύει την περιοχή στην οποία ανήκει το κλειδί ή το φύλλο που αναζητούμε.

3.1 Δημιουργία του δέντρου αναζήτησης

Θα περιγράψουμε τώρα πως αναπτύσσεται ένα trie καθώς εισέρχονται στο σύστημα νέοι peers οι οποίοι πρέπει να αναλάβουν να αποθηκεύσουν κάποιες περιοχές δεδομένων δηλαδή κάποια φύλλα του trie. Όταν αρχικά ο πρώτος peer εισέλθει στο σύστημα, αναλαμβάνει να είναι υπεύθυνος για όλο τον χώρο δεδομένων. Αυτό σημαίνει ότι δεν υπάρχει κανένα φύλλο στο trie και ότι όλα τα δεδομένα θα γνωρίζει που βρίσκονται μόνο αυτός ο αρχικός peer. Όταν στη συνέχεια ένας νέος peer θελήσει να γίνει μέλος του συστήματος θα πρέπει αρχικά να συναντήσει κάποιον τυχαίο online peer και να του ζητήσει με κάποιο τρόπο να γίνει μέλος του συστήματος. Αυτή η έναρξη της διαδικασίας εισόδου θεωρούμε ότι θα γίνει αν ο νέος peer στείλει ένα μήνυμα σε εκείνον τον τυχαίο peer ζητώντας του να αρχίσει η διαδικασία προκειμένου να γίνει μέλος και να του ανατεθεί κάποιο φύλλο. [1] Με αυτό το τρόπο όταν κάποιος νέος peer επιθυμεί να γίνει μέλος του συστήματος, το πρωτόκολο θα τον υποχρεώσει να στείλει ένα μήνυμα σε κάποιον τυχαίο online

[1] Σύμφωνα με το [10] γνωρίζουμε ότι υπάρχουν πρωτόκολλα που εκτελούνται παράλληλα με ένα πρωτόκολο P2P και μπορούν κάθε στιγμή να μας δώσουν την IP διεύθυνση κάποιου τυχαίου online peer.

peer με το οποίο μήνυμα θεωρείται ότι κάνει αίτηση για να γίνει μέλος ο νέος peer. Αυτός ο τυχαίος online peer επιλέγει με τη σειρά του κάποιο τυχαίο φύλλο για το οποίο είναι υπεύθυνος προκειμένου να αρχίσει η διαδικασία εισόδου. Αυτό το φύλλο θα το ονομάζουμε Bootstrap.

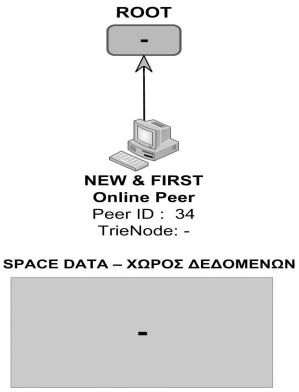
Στη συνέχεια το πρωτόκολο θα πρέπει να αποφασίσει με κάποιο τρόπο ποιο θα είναι το φύλλο-προορισμός στο οποίο θα πρέπει να φτάσει η αίτηση, δηλαδή το μήνυμα με το οποίο γίνεται η αίτηση για νέα εισαγωγή. Αυτό το φύλλο-προορισμός θα το ονομάσουμε Mate. Προσοχή όμως, πρέπει να διευκρινίσουμε ότι η δρομολόγηση προς τον προορισμό θα αρχίσει από το τυχαίο φύλλο Bootstrap. Στη συνέχεια μέσω μιας επαναλαμβανόμενης διαδικασίας ο υπεύθυνος peer κάθε φύλλου προωθεί το μήνυμα της αίτησης στον υπεύθυνο peer του γείτόνα του φύλλου που είναι πιο κοντά, δηλαδή που έχει μεγαλύτερο κοινό πρόθεμα σαν κλειδί με αυτό του προορισμού. Οι γείτονες αυτοί είναι εκείνα τα φύλλα προς τα οποία διατηρούνται εξερχόμενα links. Δηλαδή χρησιμοποιούμε των πίνακα των εξερχομένων links προκειμένου να προωθηθεί το αρχικό μήνυμα προς peers που είναι υπεύθυνοι για φύλλα που είναι πιο κοντά προς στο φύλλο Mate .[2] Πρέπει να τονίσουμε ότι πρακτικά τα μηνύματα που προωθούνται στέλνονται μεταξύ των υπεύθυνων peers των φύλλων του trie αφού αυτοί αντιπροσωπεύουν πραγματικούς χρήστες που είναι συνδεδεμένοι στο δίκτυο. Όταν το μήνυμα για αίτηση νέου μέλους φτάσει στον peer που είναι υπεύθυνος για τον Mate τότε θεωρούμε ότι το μήνυμα έφτασε στον προορισμό του. Προτού συνεχίσουμε όμως θα πρέπει να εξηγήσουμε ότι η επιλογή του Mate στη προσομοίωση που πραγματοποιήσαμε έγινε με δύο διαφορετικούς τρόπους:

Volume Balanced: Το πρωτόκολο επιλέγει ένα τυχαίο σημείο του χώρου δεδομένων και στη συνέχεια βρίσκει ποιο φύλλο είναι υπεύθυνο για τη περιοχή στην οποία ανήκει εκείνο το σημείο. Αυτό το φύλλο το ονομάζουμε Mate. Ο υπεύθυνος peer αυτού του φύλλου είναι αυτός με τον οποίο πρέπει να επικοινωνήσει τελικά ο νέος peer. Με αυτό το τρόπο δημιουργείται ένα πιο ισοζυγισμένο trie αφού εκείνα τα φύλλα που έχουν αναλάβει μεγαλύτερη περιοχή του χώρου δεδομένων είναι πιο πιθανό να επιλεγούν και άρα να διασπατούν για κάποια νέα είσοδο. Οπότε αυτή η μέθοδος τείνει να δημιουργεί ίσου μεγέθους περιοχές και κατά συνέπεια ένα πιο ισορροπημένο trie.

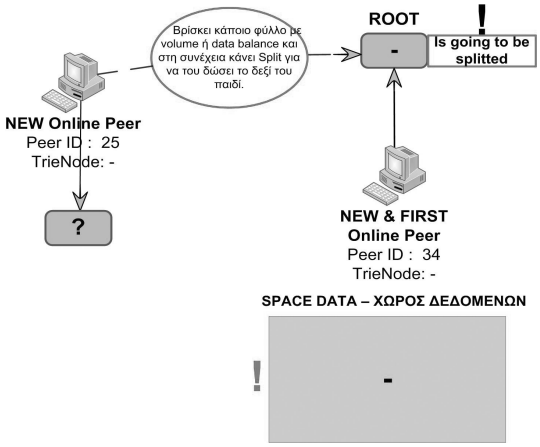
Data Balanced: Το πρωτόκολο επιλέγει ένα τυχαίο κλειδί που αντιπροσωπεύει ένα από τα data items που υπάρχουν στο σύστημα. Στη συνέχεια βρίσκει ποιο φύλλο είναι υπεύθυνο για τη περιοχή που ανήκει το τυχαίο data item και αυτό το φύλλο θα το ονομάζουμε Mate. Ο υπεύθυνος peer του Mate είναι αυτός με τον οποίο πρέπει να επικοινωνήσει τελικά ο νέος peer. Με αυτό το τρόπο το trie δεν γίνεται ισοροπημένο αλλά τείνει να επιλέγει τα φύλλα που τυχαίνει να έχουν περισσότερα data items στη περιοχή τους και να διασπάει τις περιοχές αυτών συνεχώς με αποτέλεσματα να επιλέγονται με μεγάλη συχνότητα η ίδιες περιοχές και να διασπώνται σε μικρότερες ενώ συγχρόνως να υπάρχουν άλλες περιοχές του χώρου δεδομένων που είναι πολύ μεγαλύτερες και απλά τυχαίνει να μην περιέχονται σε αυτές data items.

Στη συνέχεια πρέπει να εξηγήσουμε τι ακριβώς γίνεται όταν φτάσει το αρχικό μήνυμα στον προορισμό του. Μετά από την επαναλαμβανόμενη προώθηση του αρχικού μηνύματος από τον υπεύθυνο peer του Bootstrap εως τον υπεύθυνο peer του Mate αρχίζει ένα νέο στάδιο της διαδικασίας εισόδου όπου πρέπει να δώσουμε ένα φύλλο στον νέο peer. Ο υπεύθυνος peer του Mate πρέπει σε αυτό το σημείο να ελέγξει αν έχει κι άλλα φύλλα εκτός από τον Mate για τα οποία να είναι υπεύθυνος. Αν έχει κι άλλα φύλλα τότε αναθέτει στον νέο peer να είναι υπεύθυνος για τον Mate. Στη συνέχεια πρέπει ο νέος peer να στείλει ένα μήνυμα σε κάθε έναν από τους υπεύθυνους peers των φύλλων που υπάρχουν είτε στον πίνακα εισερχομένων είτε στον πίνακα εξερχομένων links. Αυτά τα μηνύματα στέλνονται προκειμένου

[2] Όπως θα δούμε στο κεφάλαιο 4 πραγματοποιήσαμε μια παραλλαγή του αρχικού πρωτοκόλου που περιγράφουμε κατά την οποία χρησιμοποιήσαμε τον πίνακα εξερχομένων και τον πίνακα εισερχομένων links συγχρόνως προκειμένου το πρωτόκολο να βρει τους peers που είναι υπεύθυνοι για φύλλα που είναι πιο κοντά προς στο φύλλο Mate .



Σχήμα 3.4: Ο πρώτος peer μπαίνει στο σύστημα και αναλαμβάνει να είναι υπεύθυνος για όλο το χώρο δεδομένων.



Σχήμα 3.5: Ο 2ος peer μπαίνει online.

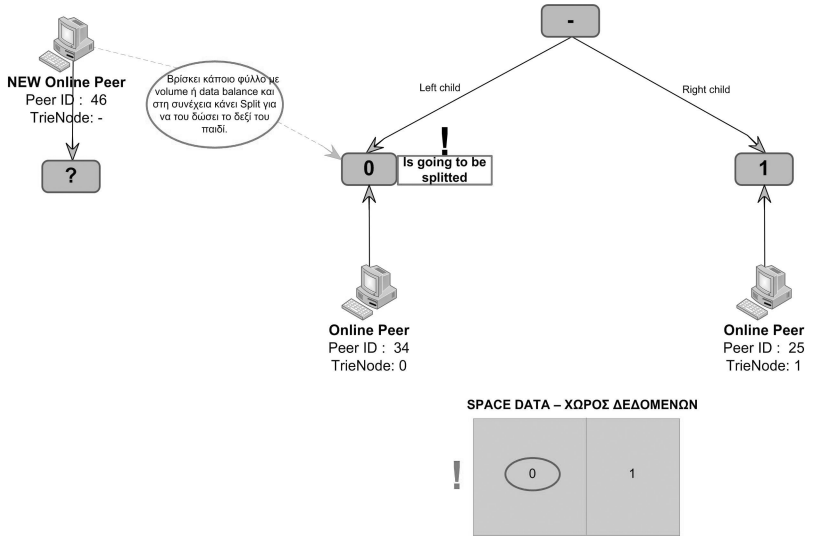
ο νέος peer να ενημερώσει για την αλλαγή που έγινε τους υπεύθυνους peers των φύλλων που γνωρίζουν αλλά και τον φύλλων που γνωρίζει ο Mate.

Επίσης αν ο υπεύθυνος peer του Mate δεν έχει άλλα φύλλα για τα οποία να είναι υπεύθυνος θα πρέπει να αρχίσει η διαδικασία split του Mate προκειμένου να δημιουργηθεί ένα ακόμα φύλλο. Απο τη διαδικασία split του Mate θα προκύψουν δύο νέα φύλλα, τα παιδιά του Mate. Θεωρήσαμε τη σύμβαση ότι το αριστερό παιδί θα το αναλάβει ο peer που ήταν υπεύθυνος για τον Mate πριν το split ενώ το δεξί παιδί θα το αναλάβει ο νέος peer για τον οποίο γίνεται η διαδικασία. Το γεγονός ότι έγινε split ο Mate πρέπει να γνωστοποιηθεί σε εκείνα τα φύλλα που είτε έχουν γείτονα τον Mate είτε ο Mate τα έχει ως γείτονες. Αυτό γίνεται από τον υπεύθυνο peer του Mate στέλνοντας ένα μήνυμα σε κάθε έναν από τους υπεύθυνους peers των φύλλων που υπάρχουν είτε σαν εισερχόμενα είτε σαν εξερχόμενα links του Mate. Θεωρούμε ότι θα στείλει ένα μήνυμα σε εκείνο τον peer που έχει είτε ένα είτε περισσότερα φύλλα που ανήκουν είτε στον πίνακα εισερχομένων είτε στον πίνακα εξερχομένων links του Mate. Περισσότερες λεπτομέρειες για τα μηνύματα που στέλνονται θα δούμε στο κεφάλαιο 3.2.3.

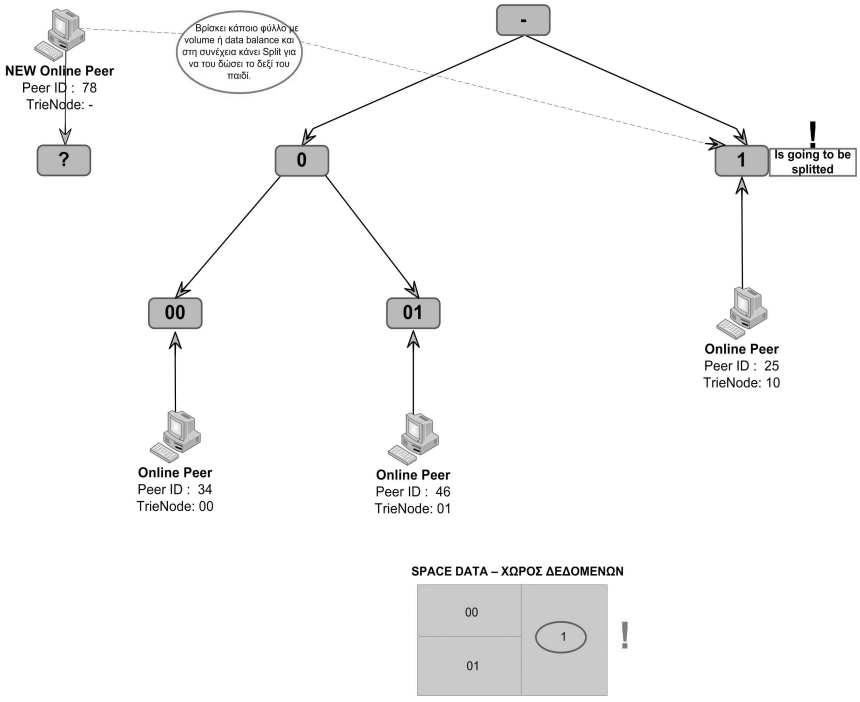
3.1.1 Παράδειγμα εκτέλεσης

Στο σχήμα 3.4 παρατηρούμε τον πρώτο peer που μπαίνει online και θέλει να γίνει μέλος του συστήματος P-Grid. Αναλαμβάνει να είναι υπεύθυνος για όλο τον χώρο δεδομένων αφού δεν υπάρχουν άλλα φύλλα και είναι ο μοναδικός peer που υπάρχει στο σύστημα.

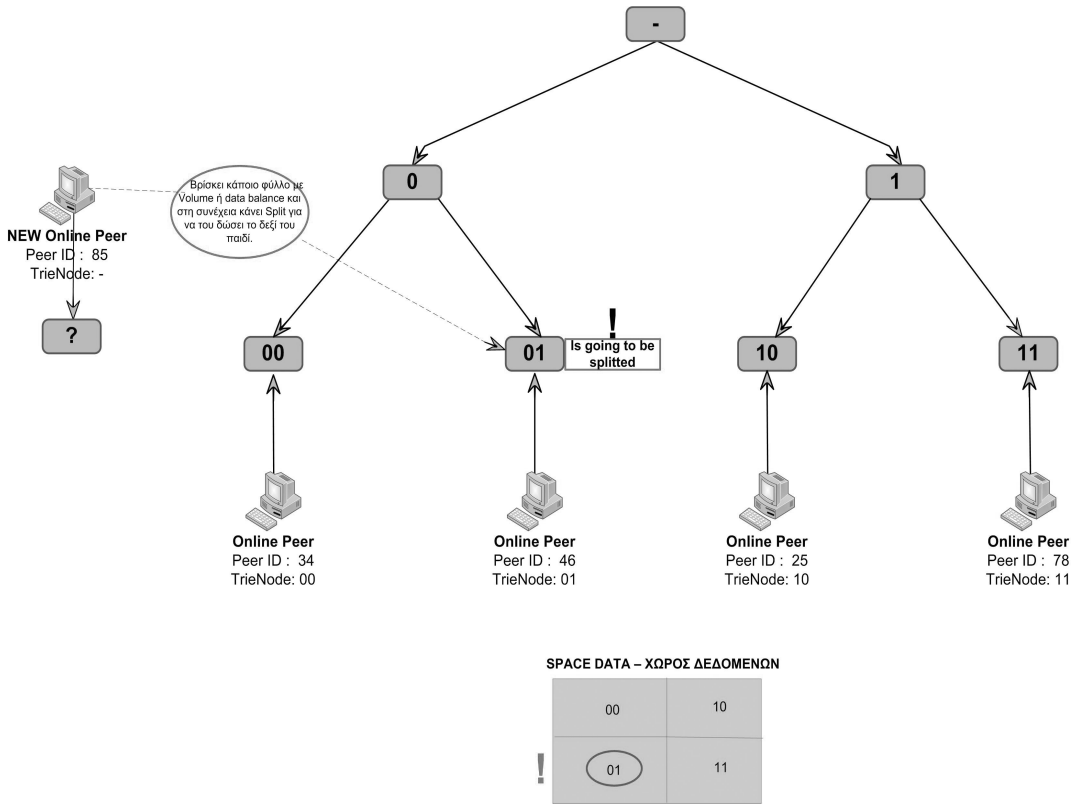
Στα σχήματα 3.6 ως και 3.12 βλέπουμε πως αυξάνονται τα φύλλα του trie καθώς νέοι peers εισέρχονται στο σύστημα....



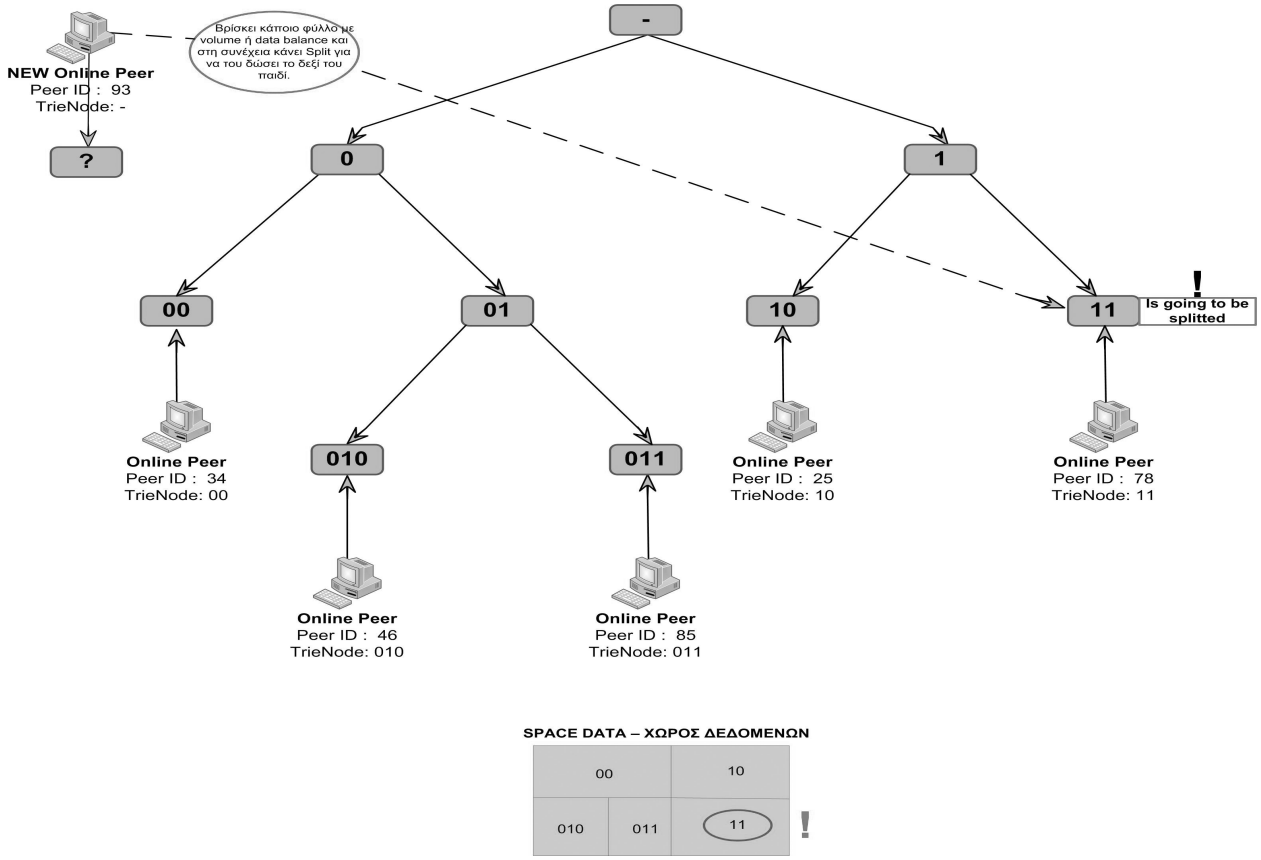
Σχήμα 3.6: Ο 3ος peer μπαίνει online.



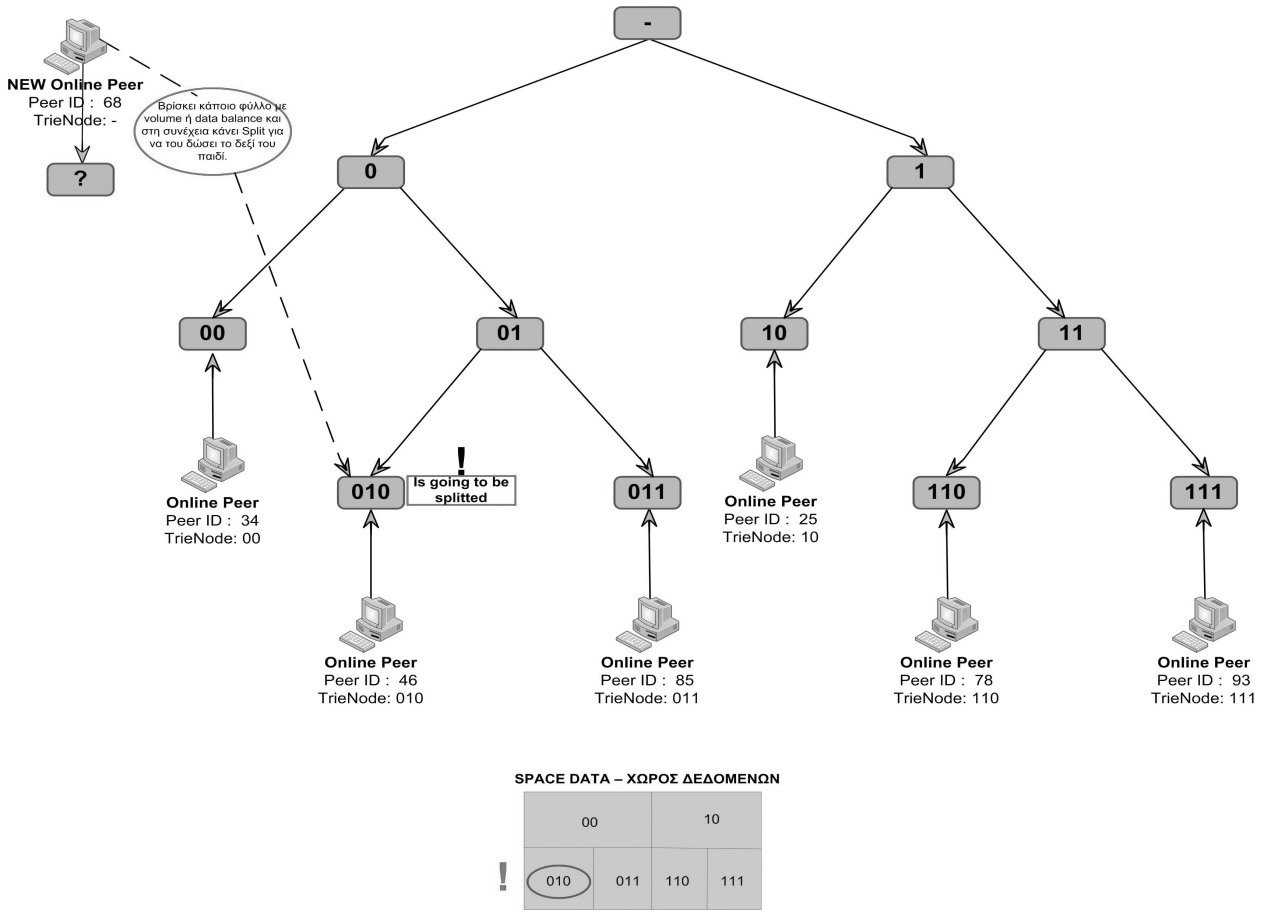
Σχήμα 3.7: Ο 4ος peer μπαίνει online.



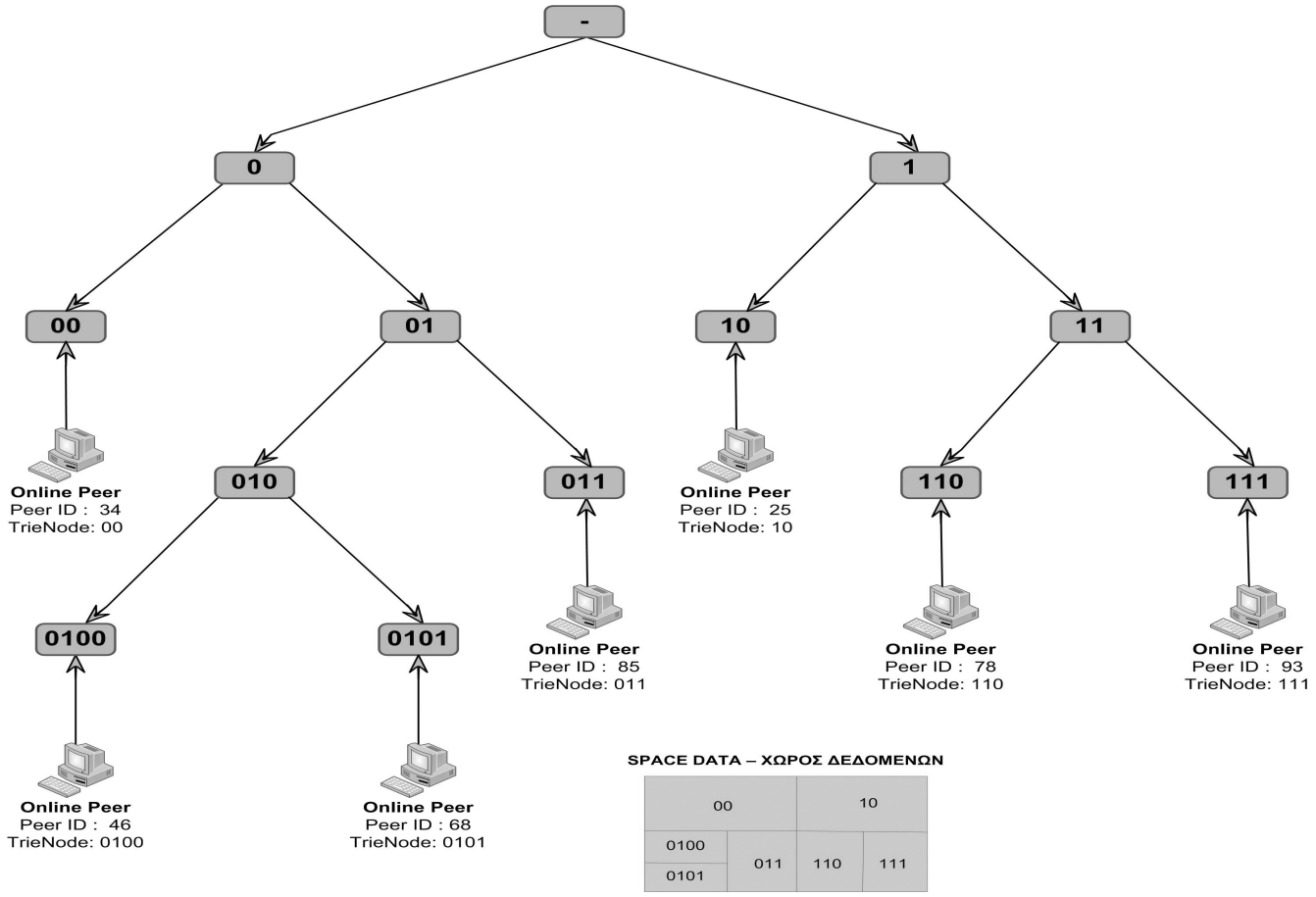
Σχήμα 3.8: Ο 5ος peer μπαίνει online.



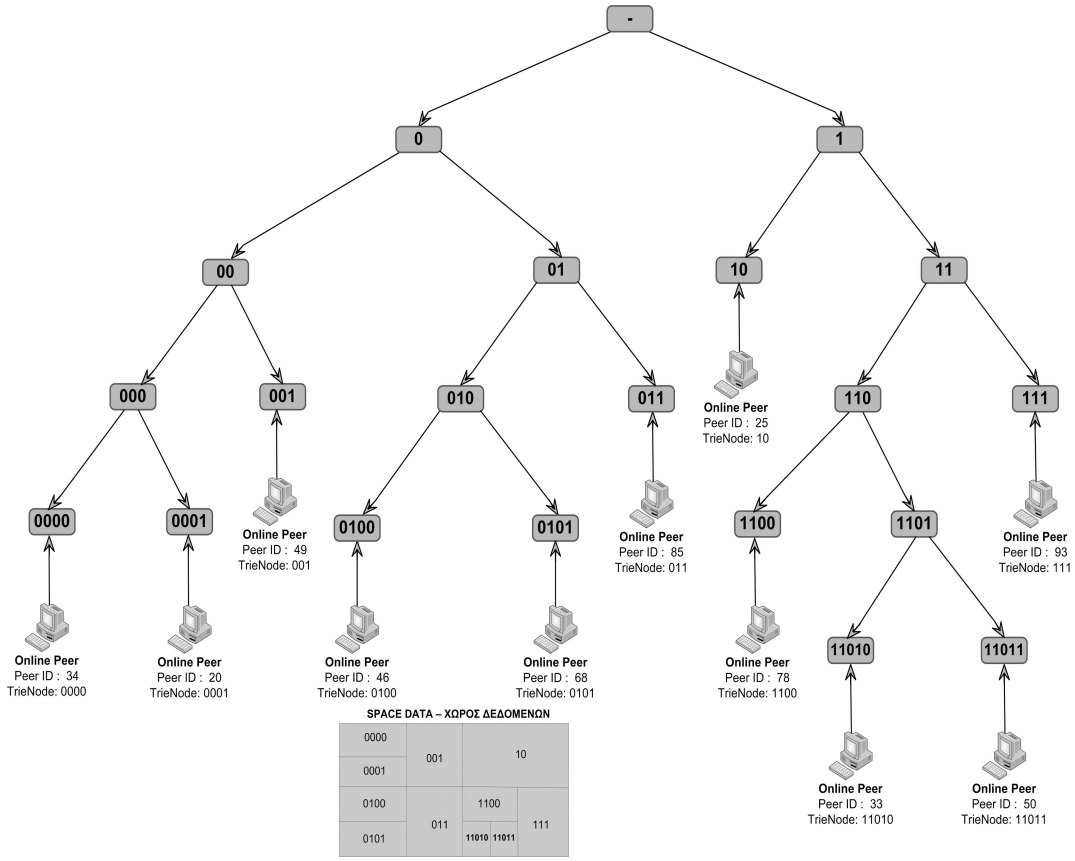
Σχήμα 3.9: Ο 6ος peer μπαίνει online.



Σχήμα 3.10: Ο 7ος peer μπαίνει online.



Σχήμα 3.11: Ο 8ος peer μπαίνει online.

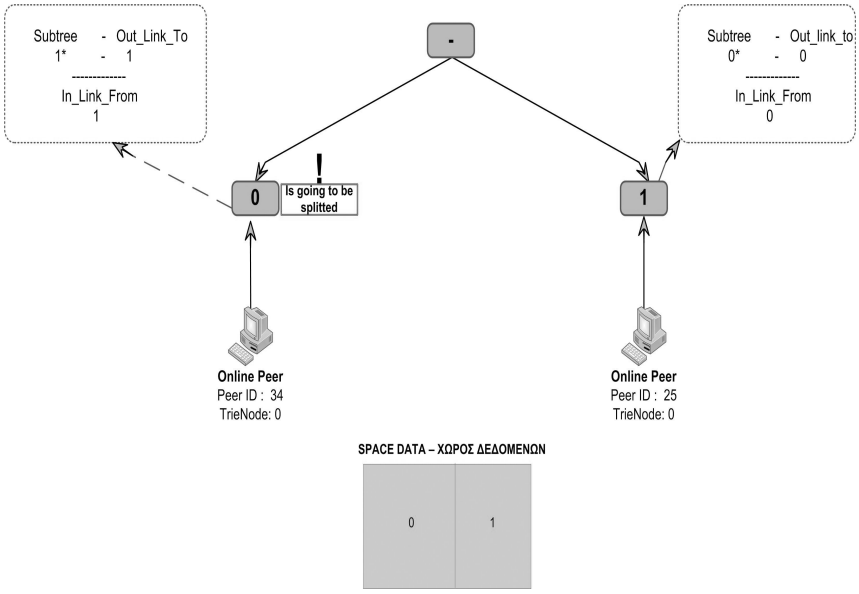


Σχήμα 3.12: Το σύστημα με 11 online peers.

Όπως είδαμε στο παραπάνω παράδειγμα για κάθε εισαγωγή ενός νέου peer στο σύστημα γίνεται μια ακόμα διάσπαση του χώρου δεδομένων. Έτσι το κάθε φύλλο του trie καλύπτει κάποιο κομμάτι του χώρου και όλα μαζί μπορούν να αντιπροσωπεύσουν το σύνολο του χώρου δεδομένων. Αυτή είναι η βασική ιδέα για τη δημιουργία ενός συστήματος P-Grid. Όπως έχουμε ήδη αναφέρει ο κάθε peer μπορεί να είναι υπεύθυνος για πολλά φύλλα του trie και όχι μόνο για ένα. Τα περισσότερα φύλλα τα αποκτά ένας peer μόνο όταν κάποιος άλλος peer που είναι ήδη online στο σύστημα αρχίσει τη διαδικασία εξόδου του όπως άλλωστε έχουμε περιγράψει στο κεφάλαιο 1.1. Στο παράδειγμα που περιγράψαμε έγιναν συνεχείς εισαγωγές νέων peers χωρίς ενδιάμεσα να μεσολαβήσει κάποια διδικασία εξόδου. Για να γίνει πιο κατανοητή η δουλειά μας αυτό θα γίνει στο κεφάλαιο 3.3 όπου θα έχουμε ήδη περιγράψει με μεγάλη λεπτομέρεια τις διαδικασίες εισόδου των peers στο σύστημα, τον τρόπο δημιουργίας των εισερχομένων και εξερχομένων links, τη συνάρτηση exchange που θα χρησιμοποιηθεί προκειμένου να ενημερώνονται οι peers για τυχόν νέα φύλλα, καθώς και τα μηνύματα που στέλνονται μεταξύ των peers προκειμένου να πραγματοποιηθούν όλα τα παραπάνω.

3.1.2 Κατασκευή των πινάκων γειτνίασης

Περιγράψαμε το τρόπο διάσπασης του κάθε φύλλου του trie αλλά δεν έχουμε εξηγήσει το τρόπο κατασκευής των πινάκων γειτνίασης που χρησιμοποιούνται για την δρομολόγηση ενός μηνύματος προς τον προορισμό του. Όπως έχουμε ήδη εξηγήσει στην αρχή του κεφαλαίου ένα φύλλο του trie διατηρεί δύο ήδη συνδέσμων-links: Τα εισερχόμενα links τα οποία συνδέουν κάποιο φύλλο με άλλα φύλλα τα οποία το έχουν ως γείτονα και τα εξερχόμενα links που συνδέουν ένα φύλλο με άλλα φύλλα που πρέπει να γνωρίζει με το τρόπο που περιγράψαμε στην αρχή του κεφαλαίου. Πρέπει να διευκρινίσουμε ότι για την πραγματοποίηση της δρομολόγησης ενός μηνύματος προς τον προορισμό του, αρχικά χρησιμοποιήσαμε μόνο τον πίνακα των εξερχομένων links. Όπως γνωρίζουμε κάποια πρωτόκολα διατηρούν πίνακες που ονομάζονται routing tables και που χρησιμοποιούνται όταν γίνεται κάποια αναζήτηση για την προώθηση μηνυμάτων προς τον προορισμό τους. Για τη δική



Σχήμα 3.13: Πίνακες γειτνίασης για τα πρώτα 2 φύλλα του trie.

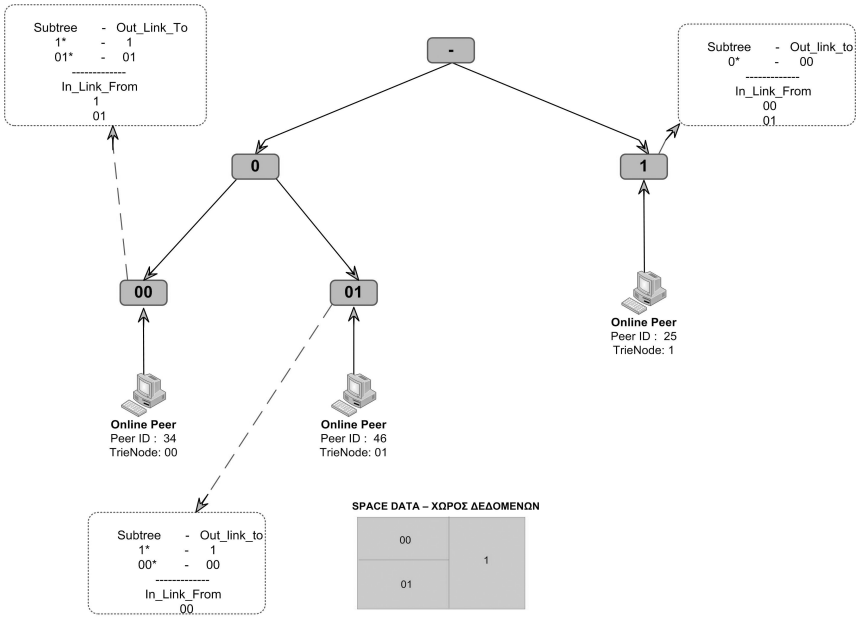
μας εργασία θα μπορούσε να πει κάποιος ότι το routing table είναι οι γείτονες ενός φύλλου ή όπως έχουμε εξηγήσει ο πίνακας των εξερχομένων links του φύλλου και σε αυτά τα links θα βασιστεί η δρομολόγηση κάποιου μηνύματος προς τον προορισμό του.

Στο σχήμα 3.13 μπορούμε να δούμε πιο ξεκάθαρα αυτό που συμβαίνει με τα εισερχόμενα και εξερχόμενα links των φύλλων καθώς γίνονται νέα splits. Όταν έγινε το πρώτο split της ρίζας του trie το αριστερό φύλλο 0 που δημιουργήθηκε πρέπει να αποθηκεύσει ένα εξερχόμενο link προς το δεξί φύλλο 1 στον πίνακα εξερχομένων links. Ομοίως πρέπει το φύλλο 1 να αποθηκεύσει ένα εξερχόμενο link προς το αριστερό φύλλο 0 στον πίνακα εξερχομένων links. Πρέπει επίσης το κάθε ένα από τα παιδιά να γνωρίζουν ποια άλλα φύλλα τα έχουν ως γείτονες και αυτό το κάνουμε διατηρώντας έναν πίνακα εισερχομένων links για κάθε φύλλο. Όταν στη συνέχεια γίνει split το φύλλο 0 όπως φαίνεται στο σχήμα 3.14 θα πρέπει τα φύλλα που θα προκύψουν να κληρονομήσουν τα ίδια εξερχόμενα links του φύλλου 0, δηλαδή τον 1 και να προσθέσουν ακόμα ένα εξερχόμενο αλλά και ένα εισερχόμενο link προς τον αδερφό τους. Δηλαδή το φύλλο 00 θα έχει τελικά εξερχόμενα links προς το φύλλο 1 και το φύλλο 01. Επίσης το φύλλο 01 θα έχει εξερχόμενα links προς το φύλλο 1 και το φύλλο 00. Συγχρόνως πρέπει να αλλάξει το εισερχόμενο link που είχε προς το φύλλο 0 ο πίνακας εισερχομένων links του φύλλου 1 και να μπει στη θέση του κάποιος τυχαίος από τα παιδιά του 0, δηλαδή από τα φύλλα 00 και 01. Αυτό πρέπει να γίνει γιατί το φύλλο 1 θα πρέπει να διατηρεί εξερχόμενα links πάντα προς κάποιο τυχαίο φύλλο του υποδέντρου 0* και άρα αφού το φύλλο 0 έκανε split δεν είναι πλέον φύλλο του trie και άρα δεν αντιπροσωπεύει κάποια περιοχή δεδομένων. Άρα όποιος το είχε ως γείτονα, δηλαδή το έχει σαν εξερχόμενο link πρέπει να ενημερωθεί και να το αντικαταστήσει με κάποιο από τα παιδιά του. Τέλος πρέπει να αλλάξει το φύλλο 0 που υπήρχε στον πίνακα εισερχομένων links του φύλλου 1 και να μπουν στη θέση του και τα δύο παιδιά του 0 αφού έχουν το 1 στον πίνακα των εξερχομένων links. Η διαδικασία που περιγράψαμε επαναλαμβάνεται συνεχώς καθώς γίνονται split τα φύλλα του trie.

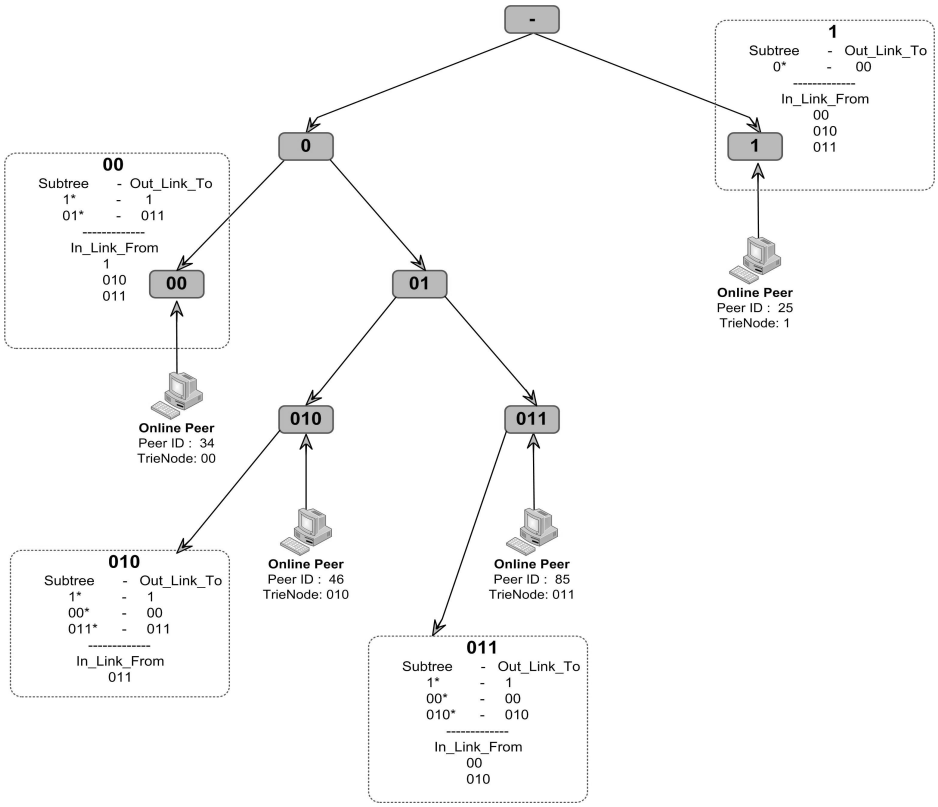
Στα σχήματα 3.14 ώς και 3.17 βλέπουμε πως δημιουργούνται οι πίνακες γειτνίασης για τα εισερχόμενα και εξερχόμενα links προς τα διάφορα φύλλα και χρησιμοποιώντας το παραπάνω σκεπτικό για την οργάνωση των πινάκων εισερχομένων και εξερχομένων links του κάθε φύλλου. Με διακεκομμένες γραμμές δείχνουμε κάποια παραδείγματα συνδέσμων που έχουν τα διάφορα φύλλα μεταξύ τους.

3.1.3 Exchange Neighbors

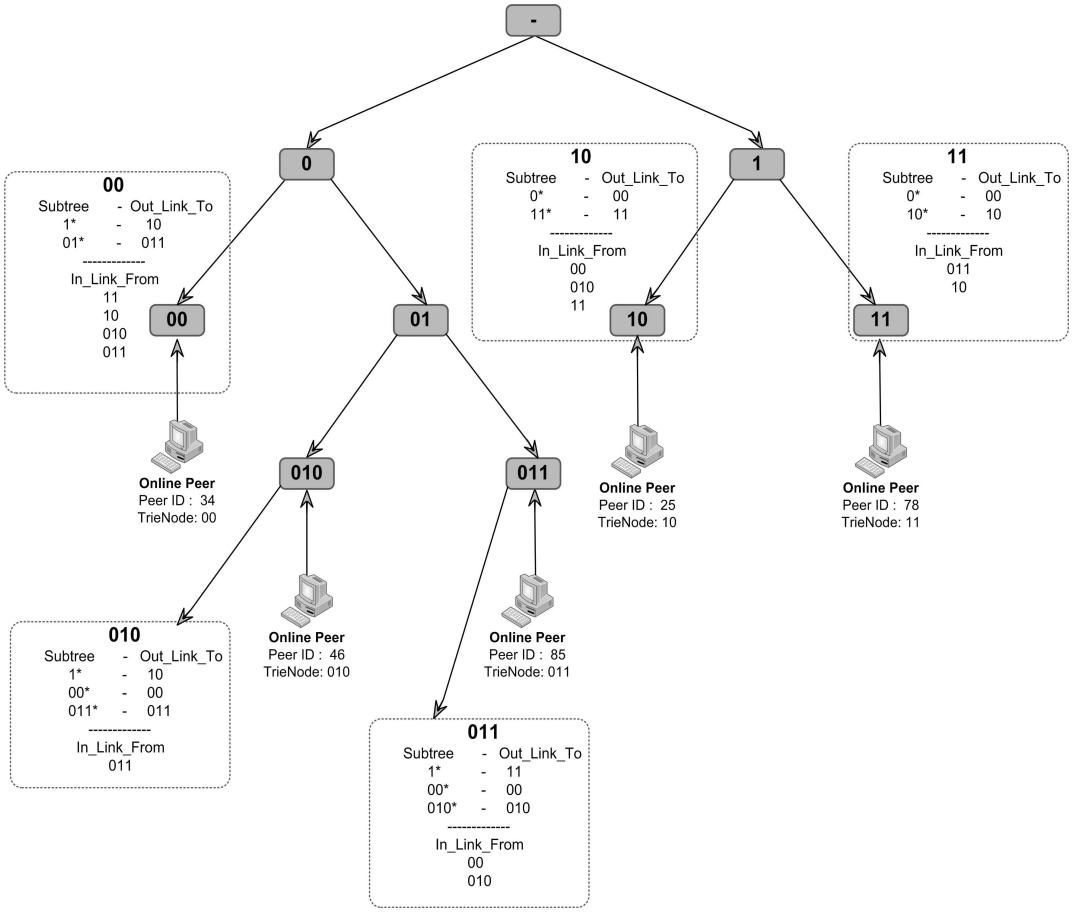
Τώρα που έχουμε πλέον περιγράψει τον τρόπο που διατηρούμε τους πίνακες γειτνίασης με τα εισερχόμενα και εξερχόμενα links προς τα διάφορα υποδέντρα πρέπει



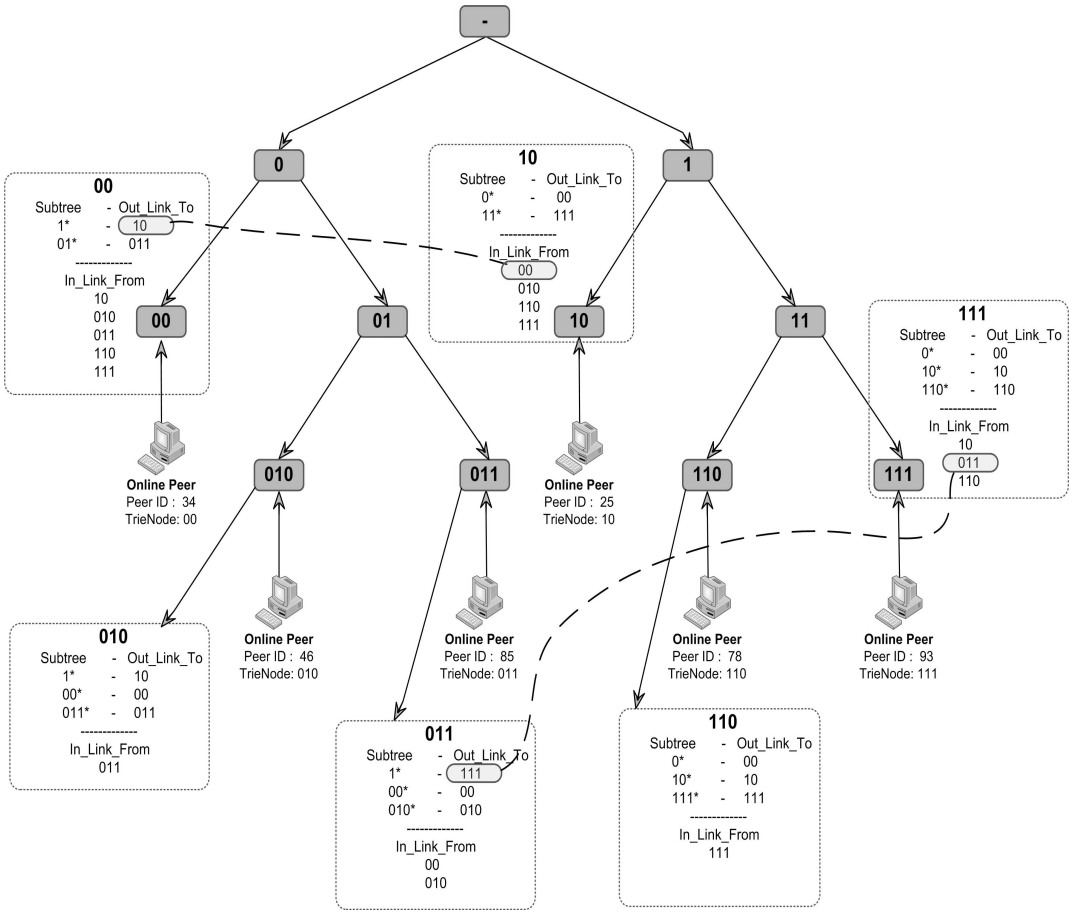
Σχήμα 3.14: Πίνακες γειτνίασης για τα πρώτα 3 φύλλα του trie.



Σχήμα 3.15: Πίνακες γειτνίασης για τα πρώτα 4 φύλλα του Trie.



Σχήμα 3.16: Πίνακες γειτνίασης για τα πρώτα 5 φύλλα του Trie.



Σχήμα 3.17: Πίνακες γειτνίασης για τα πρώτα 6 φύλλα του Trie.

να εξηγήσουμε τη συνάρτηση exchange που πραγματοποιήσαμε σύμφωνα με το [2]. Το κάθε φύλλο πρέπει να είναι ενήμερο για νέα πιθανά splits που έχουν γίνει σε υποδέντρα που μπορεί να διατηρεί γείτονες ενώ πρέπει συγχρόνως να έχει επιλέξει πάντα τυχαία φύλλα προς τα οποία θα διατηρεί links. Αυτός είναι ο σκοπός της συνάρτησης exchange και καλείται κάθε φορά που συναντιόνται δύο peers. Σε αυτό το σημείο δεν μας ενδιαφέρει για ποιο λόγο μπορεί να συναντηθούν. Η συνάρτηση exchange αναγκάζει τα δύο φύλλα να ανταλλάξουν τα εξερχόμενα links με τυχαίο τρόπο και να επαναπροσδιορίζουν τους πίνακες εξερχομένων links που διατηρούν προς τα διάφορα υποδέντρα. Με αυτό το τρόπο, τα δύο φύλλα για τα οποία εκτελείται η συνάρτηση exchange, μαθαίνουν και ενημερώνονται για νέα φύλλα που μπορεί να έχουν δημιουργηθεί σε υποδέντρα προς τα οποία διατηρούν ήδη εξερχόμενα links και απλά δεν τους είναι γνωστά. Ας δούμε όμως με ένα ακόμη παράδειγμα πιο αναλυτικά τη λειτουργία αυτής της συνάρτησης.

Έστω ότι τα δύο φύλλα των peers που συναντιώνται είναι το φύλλο 10011011 και το 10010101. Θα χρησιμοποιούμε τη τελεία (.) για να διαχωρίσουμε το κοινό πρόθεμα του pathid τους με το υπόλοιπο.

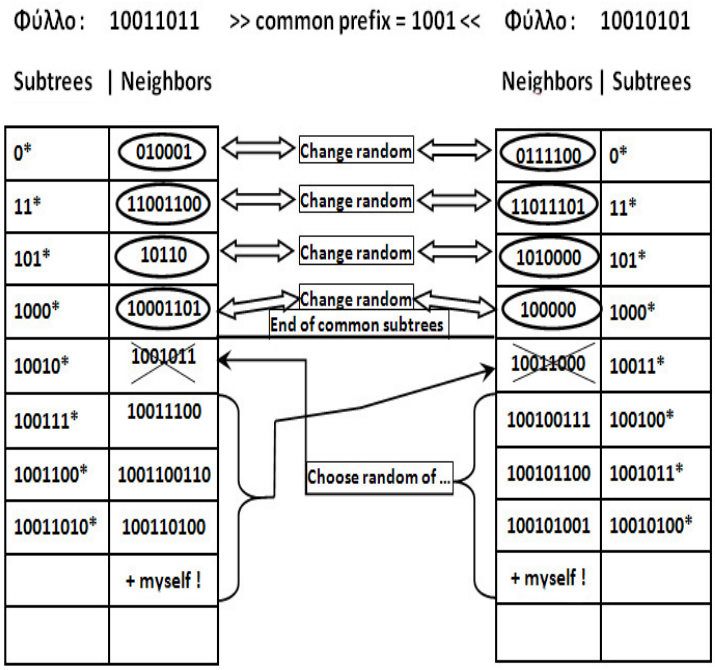
- Αρχικά ελέγχουμε αν τα δύο φύλλα έχουν κοινό πρόθεμα στα pathid τους. Το κοινό τους πρόθεμα παρατηρούμε ότι είναι το 1001.

- Για κάθε ένα από τα υποδέντρα του κοινού τους προθέματος στα οποία πρέπει να διατηρούν εξερχόμενα links , δηλαδή για τα 1000*, 101*, 11*, 0* ανταλάσσουν links γι΄ αυτά τα υποδέντρα. Αυτό γίνεται με τυχαίο τρόπο. Δηλαδη για το κάθε φύλλο και κάθε υποδέντρο προς το οποίο πρέπει να έχει εξερχόμενο link , διαλέγουμε ένα τυχαίο φύλλο ανάμεσα στα δύο που υπήρχαν ήδη σαν εξερχόμενα links στα δύο αυτά φύλλα για τα οποία γίνεται η ανταλλαγή. Δηλαδή μπορεί τελικά το νέο εξερχόμενο link να είναι προς το ίδιο φύλλο με αυτό που είχε πριν. Αν το εξερχόμενο link μετά την τυχαία επιλογή είναι τελικά το link του άλλου φύλλου, δηλαδή αλλάξει, τότε ο υπεύθυνος peer του φύλλου για το οποίο γίνεται η αλλαγή θα στείλει ένα μήνυμα στον υπεύθυνο peer του νέου γείτονα αλλά και του παλιού. Πρέπει δηλαδή να ενημερωθούν οι υπεύθυνοι peers του νέου φύλλου προς τον οποίο είναι το εξερχόμενο link αλλά και του παλιού φύλλου για την αλλαγή που έγινε και πλέον δεν το έχει ως γείτονα.

- Στη συνέχεια για το επόμενο υποδέντρο του διαφορετικού pathid τους δηλαδή για τον πρώτο το υποδέντρο 1001.0* και για τον δεύτερο το υποδέντρο 1001.1*, θα επιλέξουμε κάποιο τυχαίο από τα φύλλα των εξερχόμενων links του άλλου που ανήκουν σε εκείνο το υποδέντρο. Δηλαδή για το υποδέντρο 1001.0* του node 10011011 θα επιλέξουμε κάποιο τυχαίο από τα φύλλα που έχει το άλλο φύλλο στα υποδέντρα 1001.00*, 1001.011*, 1001.0100*, 1001.0101. Παρατηρήστε ότι το τελευταίο υποψήφιο δέντρο από το οποίο μπορεί να πάρει εξερχόμενο link είναι ο ίδιος ο node που συναντά. Ομοίως για τον node 10010101 θα πρέπει να επιλέξει σαν εξερχόμενο link προς το υποδέντρο 1001.1* κάποιο φύλλο από αυτά προς τα οποία έχει εξερχόμενα links ο άλλος node στα υποδέντρα 1001.11*, 1001.100*, 1001.1010*, 1001.1011. Ομοίως τώρα αν το νέο εξερχόμενο link είναι προς κάποιο διαφορετικό από το φύλλο που είχε πριν, πρέπει ο υπεύθυνος peer του φύλλου για το οποίο γίνεται η αλλαγή να στείλει ένα μήνυμα στους υπεύθυνους peers του νέου φύλλου-γείτονα αλλά και του παλιού που πλέον δεν είναι γείτονας.

- Καθώς γίνονται τα παραπάνω και αλλάζουμε συνεχώς τον πίνακα εξερχομένων links των δύο φύλλων πρέπει συγχρόνως να διατηρούμε τους πίνακες εισερχομένων links των nodes που έχουν εισερχόμενα links προς τους δύο nodes για τους οποίους εκτελείται η exchange ή αλλιώς των nodes που ήταν γείτονες και άλλαξαν ή έγιναν πλέον γείτονες μετά την ανταλλαγή.

- Επίσης πρέπει να εξηγήσουμε ότι στέλνεται μήνυμα μόνο μια φορά για τον κάθε διαφορετικό υπεύθυνο peer των εκάστοτε παλιών ή νέων φύλλων εξερχομένων links για τα διάφορα υποδέντρα που γίνονται οι ανταλλαγές. Δηλαδή ένα μήνυμα θα πάρει κάποιος που είναι υπεύθυνος peer κάποιου φύλλου προς το οποίο διατηρείται εξερχόμενο link και που άλλαξε ακόμα κι αν αλλάξουν κι άλλα εξερχόμενα links προς nodes που έχουν τον ίδιο υπεύθυνο peer.

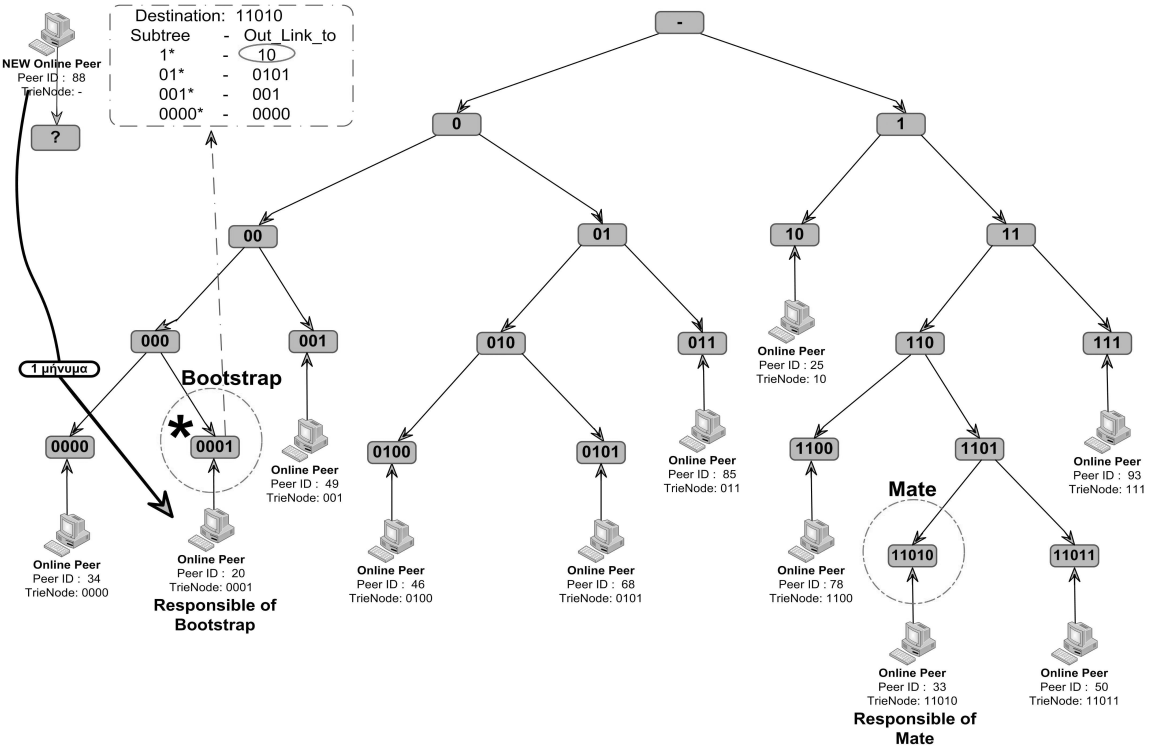


Σχήμα 3.18: Συνάρτηση exchange ανάμεσα στα φύλλα 10011011 και 10010101.

Τα μηνύματα που στέλνονται για όλες αυτές τις διαδικασίες αντιπροσωπεύουν ρεαλιστικά την πραγματικότητα και σκοπό έχουν να αντιπροσωπεύσουν τον πραγματικό αριθμό μηνυμάτων που απαιτούνται για να ολοκληρωθούν όλες αυτές οι αλλαγές των σχέσεων μεταξύ των φύλλων του trie. Στο σχήμα 3.18 βλέπουμε πιο αναλυτικά τις αλλαγές που κάνει η συνάρτηση exchange που μόλις περιγράψαμε για τα φύλλα 10011011 και 10010101 κάποιων δύο peers που συναντιόνται. Στο αμέσως επόμενο κεφάλαιο θα δούμε πότε εκτελείται η συνάρτηση exchange στη πράξη καθώς και τα μηνύματα που περιγράψαμε ότι πρέπει να στείλουν οι υπεύθυνοι peers των δύο φύλλων στους διάφορους γείτονες που μπορεί να αλλάξουν μετά από αυτή την ανταλλαγή των εξερχομένων links.

3.2 Αναλυτική περιγραφή της διαδικασίας εισόδου νέων μελών

Έχουμε ήδη περιγράψει στα κεφάλαια 3.1.1 και 3.1.2 με ποιο τρόπο αντιστοιχούμε τον χώρο δεδομένων στα φύλλα και τα φύλλα στους online peers καθώς και τη σχέση που πρέπει να έχουν τα φύλλα του trie μεταξύ τους προκειμένου να μπορούν να αντιπροσωπεύσουν όλο τον χώρο δεδομένων γρήγορα και αποτελεσματικά. Για να ολοκληρώσουμε τη διαδικασία κατασκευής του συστήματος P-Grid θα πρέπει ακόμα να περιγράψουμε τι είδους μηνύματα και σε ποιους πρέπει να σταλθούν όταν ένας νέος peer εισέρχεται στο σύστημα. Επίσης πρέπει ακόμα να περιγράψουμε πως ξεκινάει και πως δρομολογείται η διαδικασία εύρεσης του φύλλου που μπορεί να κάνει split προκειμένου να δώσει ένα μέρισμα από τον χώρο δεδομένων, δηλαδή ένα φύλλο στον νέο peer. Όλα αυτά που έχουμε ήδη περιγράψει θα γίνουν πιο κατανοητά στη συνέχεια όπου θα δούμε στη πράξη πως αρχίζει και πως δρομολογείται μια διαδικασία εισόδου μέσω των εξερχομένων links των φύλλων που θα συμμετέχουν αλλά συγχρόνως θα περιγράψουμε και τα μηνύματα που στέλνονται μεταξύ των υπεύθυνων peers των φύλλων που συμμετέχουν στη διαδικασία. Τέλος θα δούμε στη πράξη τη συνάρτηση exchange που εκτελείται όταν συναντιόνται δύο φύλλα καθώς και τη διαδικασία split του φύλλου που θα αναλάβει να δώσει κάποιο φύλλο στον νέο peer.



Σχήμα 3.19: Έναρξη δρομολόγησης για την εισαγωγή του νέου peer 88.

3.2.1 Βήμα 1ο - Έναρξη δρομολόγησης για νέα εισαγωγή

Αρχικά ένας νέος peer που θέλει να γίνει μέλος του συστήματος P-Grid θα επικοινωνήσει όπως έχουμε ήδη περιγράψει σε κάποιο τυχαίο online peer. Το πρωτόκολο αναλαμβάνει να βρεί έναν τυχαίο online peer και υποχρεώνει τον νέο peer να στείλει ένα μήνυμα σε αυτόν με το οποίο μήνυμα θα θεωρήσουμε ότι του ζητάει να τον κάνει μέλος. Αυτός ο τυχαίος online peer επιλέγει με τη σειρά του κάποιο τυχαίο φύλλο για το οποίο είναι υπεύθυνος προκειμένου να αρχίσει η διαδικασία εισόδου από αυτό. Αυτό το φύλλο θα το ονομάζουμε Bootstrap. Στη συνέχεια το πρωτόκολο θα πρέπει να αποφασίσει είτε με volume balance είτε με data balance ποιο θα είναι το φύλλο-προορισμός και άρα ποιος θα είναι ο υπεύθυνος peer του φύλλου αυτού. Σε αυτόν τον peer θα πρέπει να φτάσει το αρχικό μήνυμα με το οποίο ζητάει ο νέος peer να γίνει μέλος. Το φύλλο-προορισμό θα το ονομάσουμε Mate. Προσοχή όμως, πρέπει να διευκρινίσουμε ότι η δρομολόγηση προς τον προορισμό θα αρχίσει από το τυχαίο φύλλο Bootstrap. Στη συνέχεια μέσω μιας επαναλαμβανόμενης διαδικασίας ο υπεύθυνος peer κάθε φύλλου προωθεί το μήνυμα της αίτησης στον υπεύθυνο peer του φύλλου-γείτονα που είναι πιο κοντά, δηλαδή που έχει μεγαλύτερο κοινό πρόθεμα σαν κλειδί με αυτό του προορισμού. Οι γείτονες αυτοί είναι εκείνα τα φύλλα προς τα οποία διατηρούνται εξερχόμενα links. Δηλαδή χρησιμοποιούμε των πίνακα των εξερχομένων links προκειμένου να προωθηθεί το αρχικό μήνυμα προς peers που είναι υπεύθυνοι για φύλλα που είναι πιο κοντά στο φύλλο Mate.

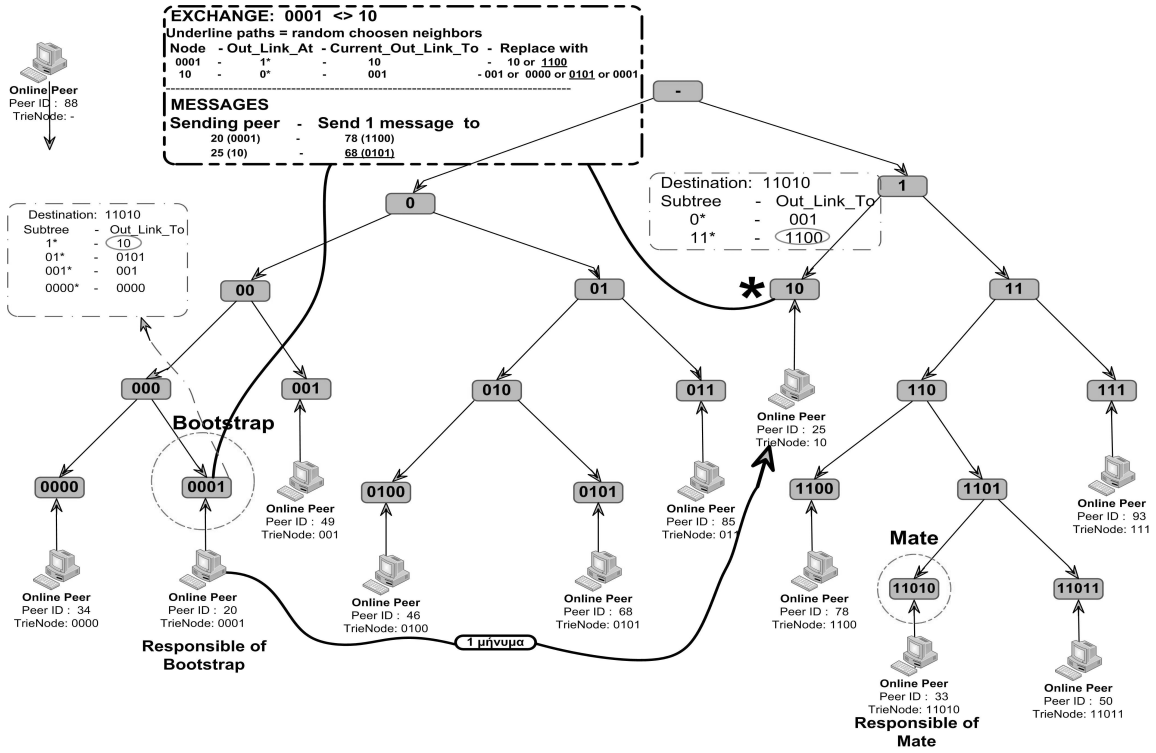
Για να γίνει όμως πιο κατανοητή η διαδικασία δεν γίνεται να μην την περιγράψουμε με ένα αναλυτικό παράδειγμα που θα περιλαμβάνει εικόνες με όλες τις λεπτομέρειες βήμα προς βήμα. Όπως βλέπουμε στο σχήμα 3.19 αρχικά ο νέος peer που επιθυμεί να γίνει μέλος του συστήματος θα στείλει ένα μήνυμα στον υπεύθυνο peer του Bootstrap. Στη συνέχεια αυτό το μήνυμα θα δρομολογηθεί προς τον προορισμό του σύμφωνα με τη διαδικασία που περιγράφεται αμέσως μετά.

3.2.2 Βήμα 2ο - Προώθηση του μηνύματος σε γείτονες

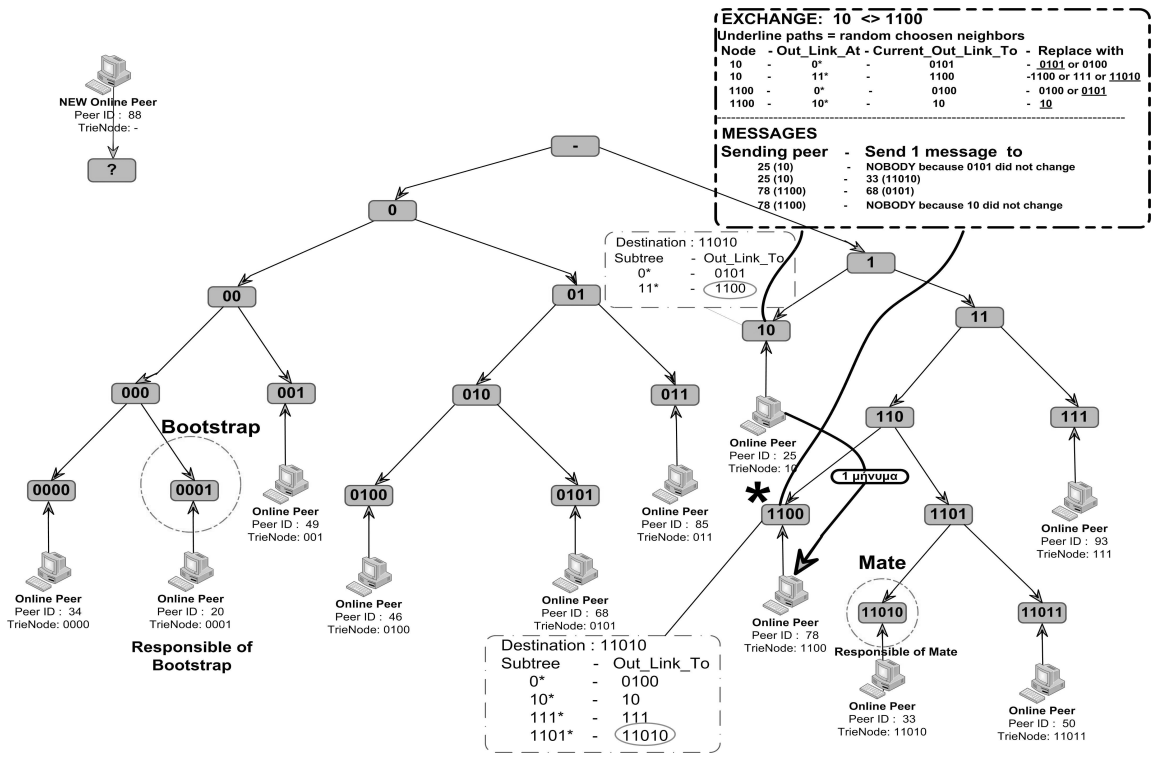
Στο επόμενο βήμα πρέπει ο Bootstrap να επιλέξει ένα φύλλο από αυτά που υπάρχουν στον πίνακα των εξερχόμενων links ώστε να προωθηθεί η αίτηση για νέα εισαγωγή σε κάποιο φύλλο που θα είναι πιο κοντά στον προορισμό μας, δηλαδή

τον Mate. Με (*) θα συμβολίζουμε το σημείο στο οποίο βρίσκεται η εκτέλεσή της διαδικασίας. Αρχικά θα συγκρίνει ο Bootstrap το pathid του Mate στον οποίο θέλουμε να φτάσουμε και θα επιλέξει εκείνον τον γείτονα του πίνακα εξερχομένων links που έχει μεγαλύτερο common prefix με το pathid του προορισμού μας, δηλαδή όπως φαίνεται και στο σχήμα 3.20 θα επιλέξει τον 10. Αρχικά ο peer 20 πρέπει να στείλει ένα μήνυμα στον peer 25 αφού εκείνος είναι υπεύθυνος για το φύλλο 10, προκειμένου να συνεχίσει η διαδικασία εύρεσης του Mate από το φύλλο 10 που σίγουρα είναι πιο κοντά στον προορισμό μας. Σε αυτό ακριβώς το σημείο εκτελείται η συνάρτηση exchange ανάμεσα στα δύο φύλλα 0001 και 10 με το τρόπο που περιγράψαμε στο προηγούμενο κεφάλαιο. Παρατηρούμε ότι τα δύο φύλλα που συναντιόνται δεν έχουν κοινό common prefix και άρα το φύλλο 0001 θα αλλάξει το εξερχόμενο link που έχει προς το υποδέντρο 1* με κάποιο από τα φύλλα 10 που έχει τη παρούσα στιγμή και το φύλλο 1100 προς το οποίο διατηρεί εξερχόμενο link το φύλλο 10 στο υποδέντρο 11*. Επίσης το φύλλο 10 θα αλλάξει το φύλλο προς το οποίο διατηρεί εξερχόμενο link στο υποδέντρο 0* με κάποιο τυχαίο φύλλο από τα 0101,001,0000 και 0001. Όλα αυτά τα μηνύματα είναι σημαντικά και μας ενδιαφέρουν στη μελέτη που κάναμε. Είναι μηνύματα που προσομειώνουμε και που στέλνονται μέσω internet προκειμένου να γίνει μέλος του συστήματος ένας νέος peer.

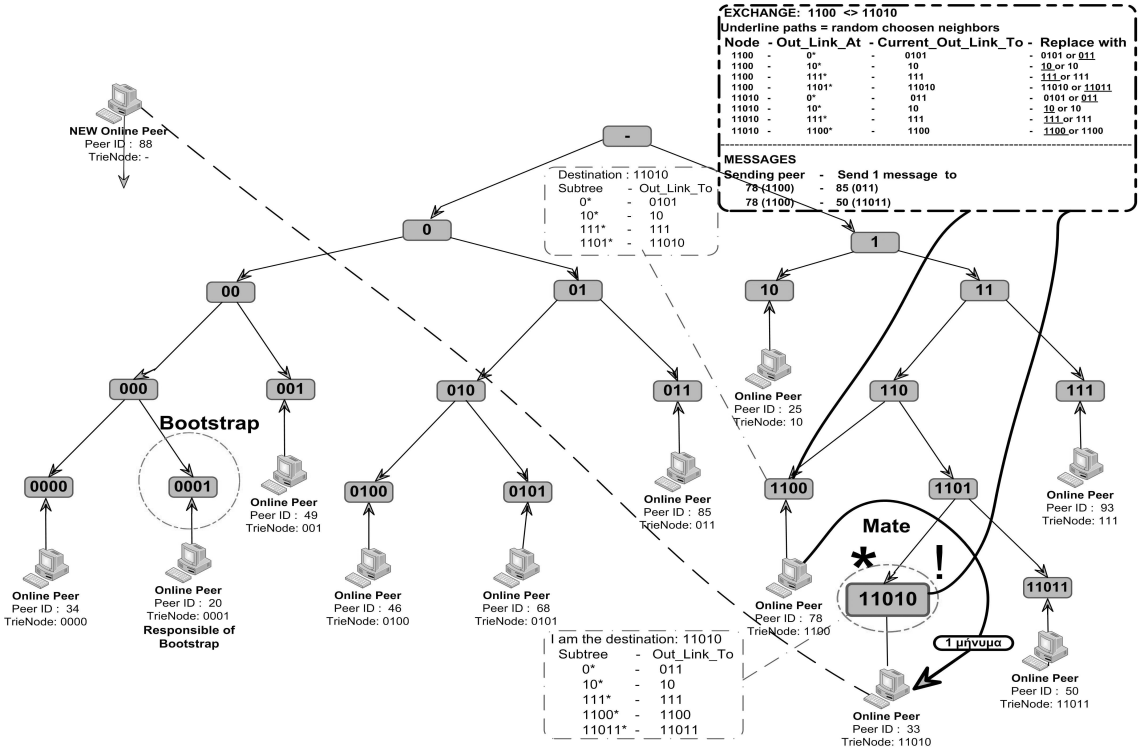
Ας επανέλθουμε όμως στο παράδειγμά μας και να δούμε τη συνέχεια της δρομολόγησης για την αίτηση εισαγωγής του νέου peer 88. Αφού στείλει το απαιτούμενο μήνυμα ο υπεύθυνος peer του Bootstrap στον peer 25 ο οποίος είναι υπεύθυνος για το φύλλο 10, η διαδικασία θα συνεχιστεί με την ίδια λογική από το φύλλο 10 αυτή τη φορά. Το φύλλο 10 ελέγχει όλα τα εξερχόμενα links που διατηρεί τα διάφορα φύλλα και συγκρίνει το pathid του καθενός με αυτό του προορισμού. Πρέπει να επιλέξει εκείνο το εξερχόμενο link που οδηγεί στο φύλλο που έχει το πιο μεγάλο κοινό πρόθεμα με τον προορισμό, δηλαδή όπως φαίνεται και στο σχήμα 3.21 το φύλλο 10 πρέπει να προωθήσει το μήνυμα στο φύλλο 1100. Αυτό σημαίνει ότι ο χρήστης 25 θα πρέπει να στείλει μήνυμα στον χρήστη 78. Ανάμεσα στα φύλλα 10 και 1100 εκτελείται όμοια η συνάρτηση exchange με τη διαφορά ότι τώρα τα δύο φύλλα έχουν κοινό πρόθεμα το 1. Τα μηνύματα στέλνονται με τον τρόπο που έχουμε εξηγήσει και γίνεται ξανά ανταλλαγή των γειτόνων που διατηρούνται προς τα ανάλογα υποδέντρα. Η διαδικασία επαναλαβάνεται και τα επόμενα σχήματα εξηγούν αναλυτικά πως θα φτάσει το αρχικό μήνυμα αίτησης για νέα εισαγωγή στον peer 33 που είναι ο ζητούμενος peer , καθώς είναι υπεύθυνος για το φύλλο Mate που είναι ο προορισμός μας.



Σχήμα 3.20: Προώθηση του μηνύματος στο φύλλο 10.



Σχήμα 3.21: Προώθηση του μηνύματος στο φύλλο 1100.



Σχήμα 3.22: Προώθηση του μηνύματος στο φύλλο 11010 που είναι και ο τελικός προορισμός.

3.2.3 Βήμα 3ο - Άφιξη στο προορισμό και τέλος διαδικασίας

Όταν το μήνυμα φτάσει στον προορισμό μας το επόμενο βήμα είναι ο peer 33 να ελέγξει αν έχει ένα ή περισσότερα φύλλα για τα οποία να είναι υπεύθυνος. Αν έχει περισσότερα τότε θα δώσει στον νέο peer τον ίδιο τον Mate. Αν είναι όμως υπεύθυνος μόνο για ένα φύλλο όπως συμβαίνει σε αυτό το παράδειγμα, τότε πρέπει να αρχίσει η διαδικασία split του 11010.

Όπως είδαμε στη προηγούμενη παράγραφο όταν ένα φύλλο κάνει split, τα παιδιά του κληρονομούν τον πίνακα εξερχομένων links του φύλλου που έκανε split. Αυτά όμως τα εξερχόμενα links είναι προς φύλλα για τα οποία είναι υπεύθυνοι κάποιοι online peers. Για να μιλήσουμε όμως σε επίπεδο peers οι οποίοι επικοινωνούν μεταξύ τους πρέπει να θεωρήσουμε ότι στέλνονται μηνύματα μέσω internet. Αυτό πρακτικά σημαίνει ότι για όλη αυτή τη διαδικασία, πέρα από τα μηνύματα που περιγράψαμε, πρέπει να σταλθούν και άλλα μηνύματα από τον υπεύθυνο peer του Mate. Ο υπεύθυνος peer του Mate πρέπει να ενημερώσει τους υπεύθυνους peers των φύλλων προς τα οποία διατηρεί εξερχόμενα ή και εισερχόμενα links ο Mate ότι ο Mate κάνει split.

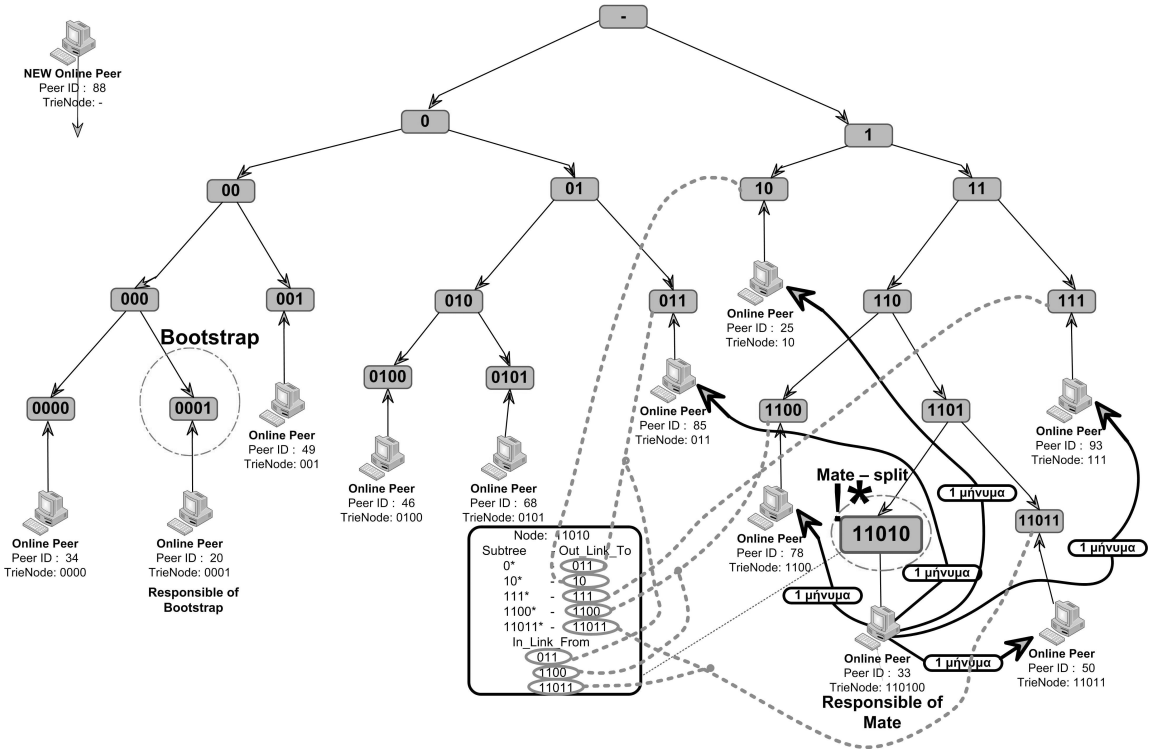
Προκειμένου να αντιπροσωπεύσουμε ρεαλιστικά το πλήθος και το είδος των μηνυμάτων που στέλνονται από τον υπεύθυνο peer του Mate υποθέτουμε ότι ισχύουν τα εξής:

- Όταν κάνει split:

  Ο υπεύθυνος peer του Mate, δηλαδή ο peer 33 πρέπει να στείλει μήνυμα σε όλους τους peers που είναι υπεύθυνοι για τα φύλλα προς τα οποία διατηρεί εξερχόμενα links αλλά και για τα φύλλα προς τα οποία διατηρεί εισερχόμενα links ο Mate, προκειμένου να τους ενημέρωσει ότι ο Mate κάνει split.

- Όταν ΔΕΝ κάνει split:

  Αν τελικά ο peer 33 είναι υπεύθυνος για περισσότερα από ένα φύλλα και όχι μόνο για τον Mate, τότε ο Mate δεν θα κάνει split και απλά θα στείλει ο νέος peer μήνυμα στους peers που είναι υπεύθυνοι για τα φύλλα προς τα οποία διατηρεί εξερχόμενα links αλλά και για τα



Σχήμα 3.23: Ο υπεύθυνος peer του Mate θα στείλει μήνυμα στους υπεύθυνους peers των φύλλων προς τα οποία διατηρεί εισερχόμενα και εξερχόμενα links προκειμένου να ενημερωθούν για το split του Mate.

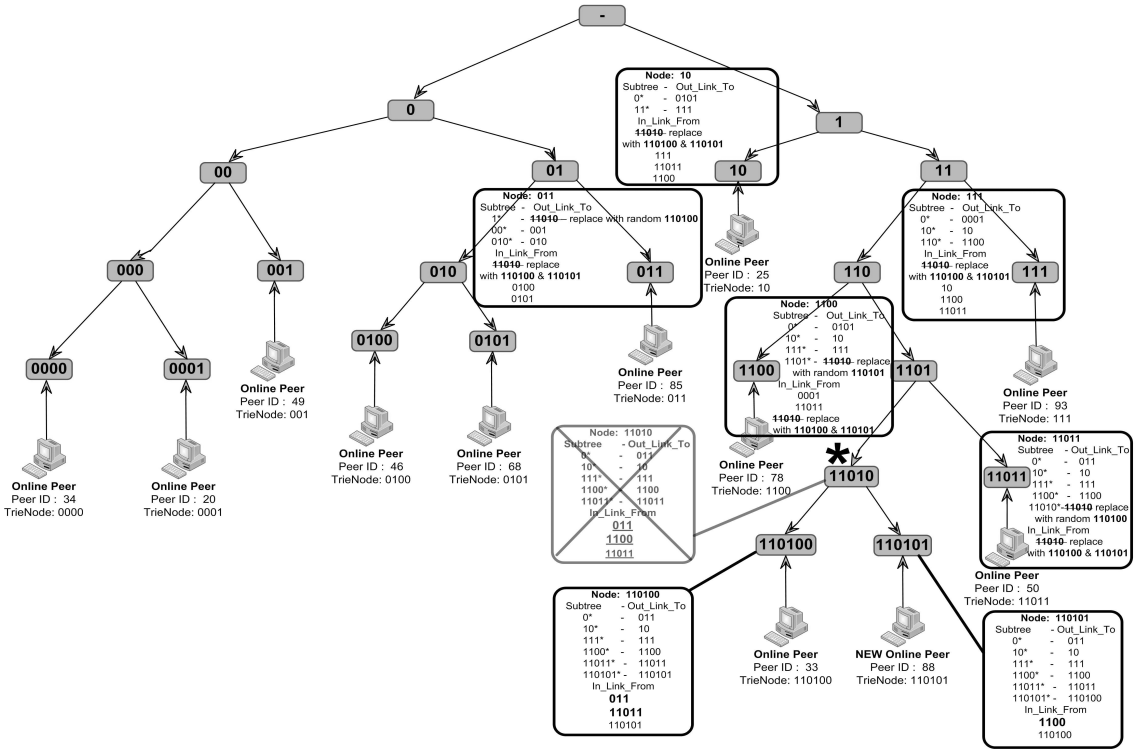
φύλλα προς τα οποία διατηρεί εισερχόμενα links ο Mate. Έτσι θα τους ενημερώσει ότι τώρα θα είναι αυτός ο υπεύθυνος peer του Mate και όχι ο 33.

- Αν τελικά πρέπει να γίνει split εξηγήσαμε ότι ο peer 33 θα στείλει μηνύματα στους peers που είναι υπεύθυνοι για τα φύλλα προς τα οποία διατηρεί εξερχόμενα links αλλά και για τα φύλλα προς τα οποία διατηρεί εισερχόμενα links ο Mate. Αυτό όμως θα γίνει μόνο σε περίπτωση που τα φύλλα προς τα οποία διατηρείται εισερχόμενο ή εξερχόμενο link ΔΕΝ έχουν σαν υπεύθυνο peer τον 33. Αυτό πρέπει να ισχύει αφού δεν θα είχε νόημα να ενημερώσει ένας peer τον εαυτό του ότι κάποιο από τα φύλλα για τα οποία είναι υπεύθυνος θα κάνει split.

- Επίσης αν για ένα φύλλο διατηρεί και εισερχόμενο και εξερχόμενο link ο Mate, τότε θα στείλουμε ένα και μόνο μήνυμα στον υπεύθυνο peer αυτού του φύλλου και όχι δύο ξεχωριστά μηνύματα.

- Τέλος μόνο ένα μήνυμα θα σταλθεί και σε κάποιον peer ο οποίος είναι υπεύθυνος για πολλά φύλλα προς τα οποία μπορεί να διατηρεί είτε εισερχόμενα είτε εξερχόμενα links ο Mate.

Στο σχήμα 3.23 φαίνονται τα μηνύματα που πρέπει να στείλει ο υπεύθυνος του Mate καθώς και τα εισερχόμενα και εξερχόμενα links που διατηρεί ο Mate.

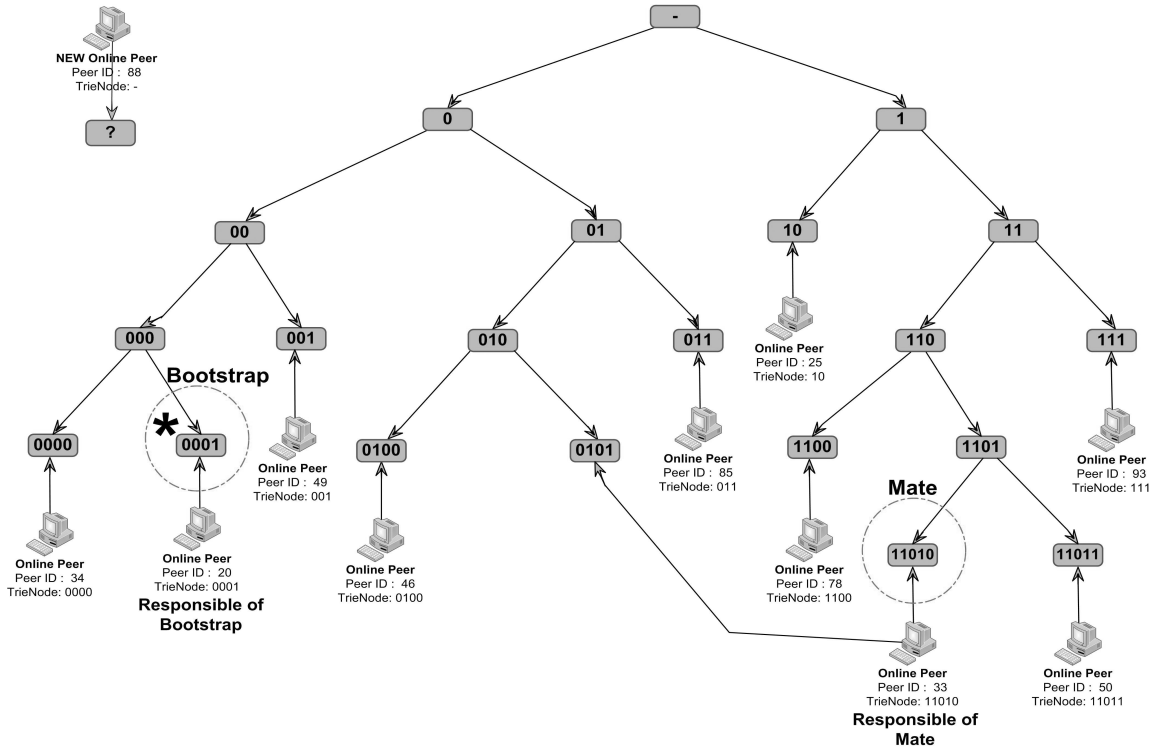


Σχήμα 3.24: Αλλαγές που πρέπει να γίνουν στους πίνακες γειτνίασης κάποιων φύλλων εξάιτίας του split που έγινε. Ο σκοπός των μηνυμάτων του σχήματος 3.23 είναι να αντιληφθούν οι σχετικοί peers τις αλλαγές που φαίνονται στο παρόν σχήμα λόγω του split που έγινε.

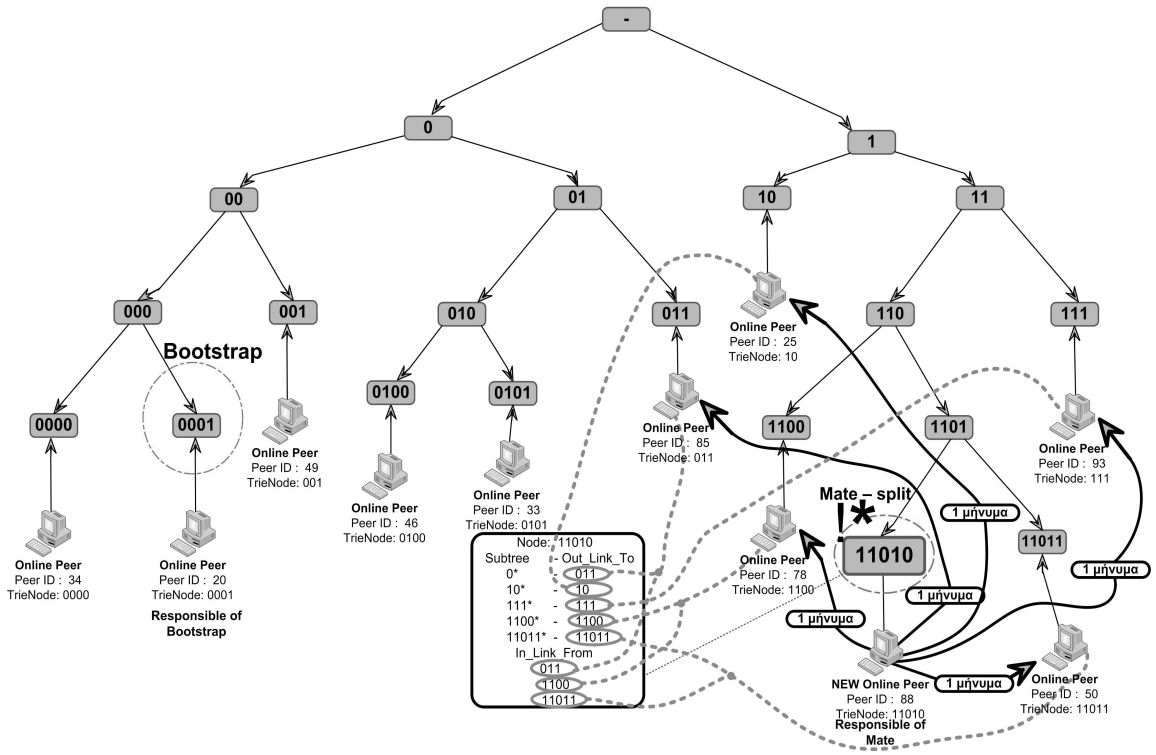
Σε αυτό το σημείο πρέπει να πούμε ότι αν ο υπεύθυνος peer του Mate είχε και άλλα φύλλα για τα οποία να είναι υπεύθυνος, τότε δεν θα γινόταν split. Τότε απλά ο νέος peer θα γινόταν υπεύθυνος peer του Mate. Θα έστελνε τα ίδια ακριβώς μηνύματα που έστειλε και στη περίπτωση του split ο peer 33. Δηλαδή θα ενημέρωνε με το ίδιο πλήθος μηνυμάτων τους υπεύθυνους peers των φύλλων προς τα οποία διατηρεί εισερχόμενα και εξερχόμενα links ο Mate, μόνο που αυτή τη φορά θα τα έστελνε ο νέος peer 88 και όχι ο 33. Στα σχήματα 3.25 και 3.26 βλέπουμε τη περίπτωση όπου ο Mate δεν θα γίνει split αφού ο υπεύθυνος peer του Mate έχει και άλλα φύλλα για τα οποία είναι υπεύθυνος και έτσι θα αναθέσει στον νέο peer να είναι υπεύθυνος για τον Mate χωρίς να γίνει κάποιο split.

Μέχρι τώρα έχουμε εξηγήσει πώς δημιουργείται από το μηδέν ένα σύστημα P-Pgrid , πως χωρίζουμε τον χώρο των δεδομένων που αντιστοιχεί σε κάθε φύλλο του trie , ποια είναι τα εισερχόμενα και εξερχόμενα links που πρέπει να διατηρεί το κάθε φύλλο του trie καθώς επίσης και με ποιο τρόπο ανταλλάσουν εξερχόμενα links δύο φύλλα που μπορεί να έρθουν σε επαφή. Επίσης εξηγήσαμε σε επίπεδο πραγματικών peers ποια είναι τα ελάχιστα απαιτούμενα μηνύματα που πρέπει να στείλουν μεταξύ τους προκειμένου να γίνει ένας νέος peer μέλος του συστήματος και τέλος εξηγήσαμε σε περίπτωση που το φύλλο-προορισμός γίνει split ποιοι peers πρέπει να ενημερωθούν με μηνύματα που θα στείλει ο υπεύθυνος peer του Mate για την περίπτωση που γίνει split ή ο νέος χρήστης για την περίπτωση που δεν γίνει split.

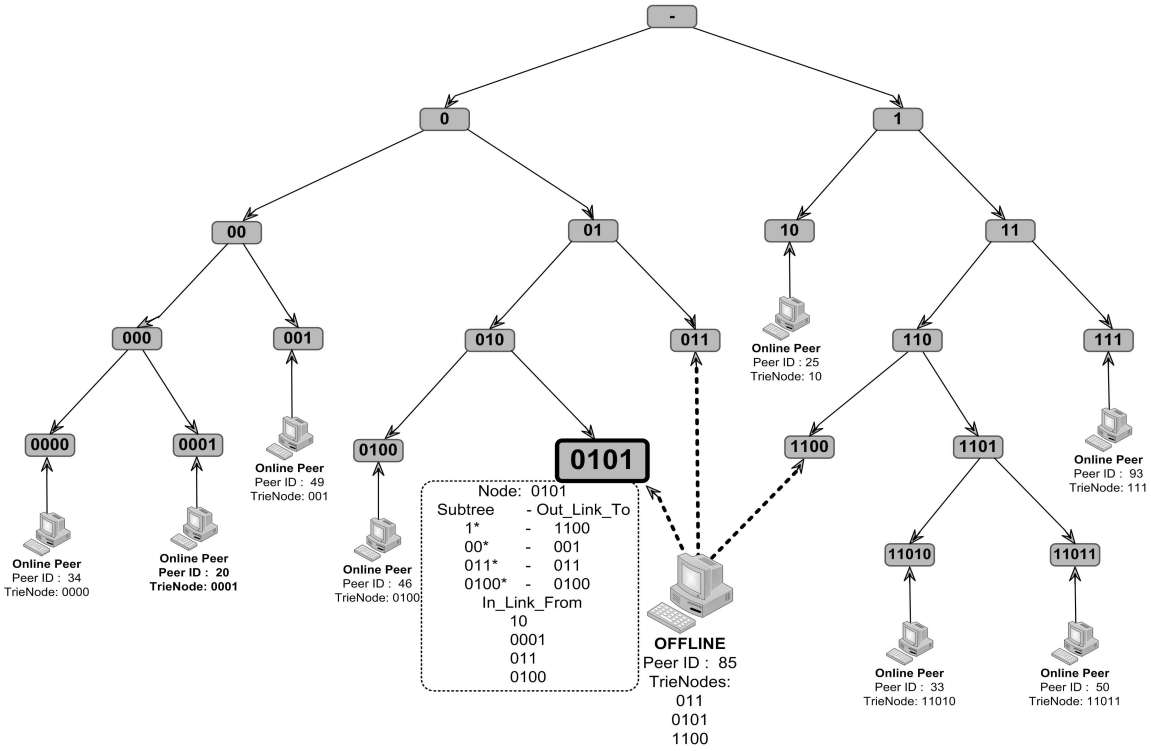
Σε αυτό το σημείο έχουμε ολοκληρώσει την αναλυτική περιγραφή τις διαδικασίας εισόδου όπως πραγματοποιήθηκε στα πειράματά μας. Στο επόμενο κεφάλαιο θα περιγράψουμε τη διαδιακασία εξόδου ενός peer που επιθυμεί να βγει εκτός συστήματος, πως πρέπει να διαμορφωθεί το trie και οι γείτονες του κάθε φύλλου ώστε να έχουμε σταθερότητα και αξιοπιστία στην αναζήτηση δεδομένων, ποιος θα αναλάβει να είναι υπεύθυνος για το φύλλο ή τα φύλλα που είχε ο peer που επιθυμεί να βγει εκτός και το σημαντικότερο τι είδους μηνύματα στέλνονται μεταξύ των peers προκειμένου να γίνουν όλες αυτές οι εργασίες.



Σχήμα 3.25: Άφιξη στον προορισμό 11010 στη περίπτωση όπου ο peer 33 είναι υπεύθυνος για δύο φύλλα, το 0101 και το 11010.



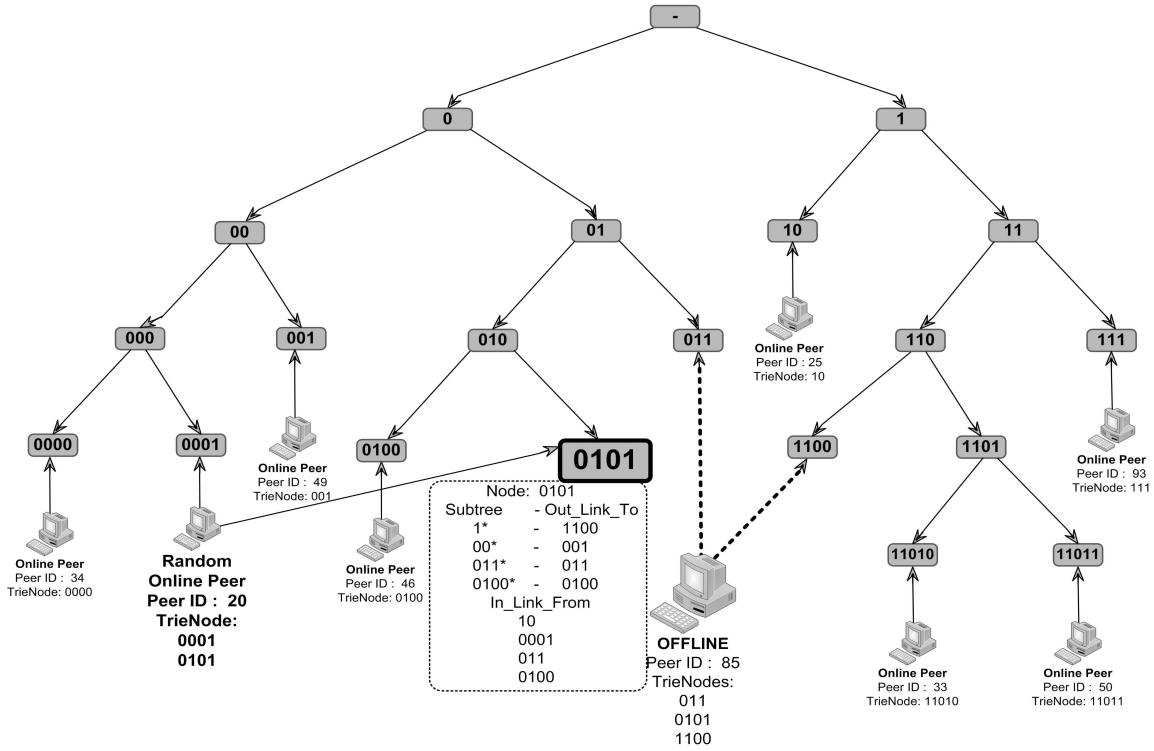
Σχήμα 3.26: Ο νέος peer 88 αναλαμβάνει να είναι υπεύθυνος για το φύλλο 11010 και ενημερώνει όλα τα φύλλα των εισερχόμενων και εξερχόμενων links.



Σχήμα 3.27: Ο peer 85 επιθυμεί να βγει εκτός συστήματος. Πρέπει κάποιοι peers να αναλάβουν τα φύλλα του.

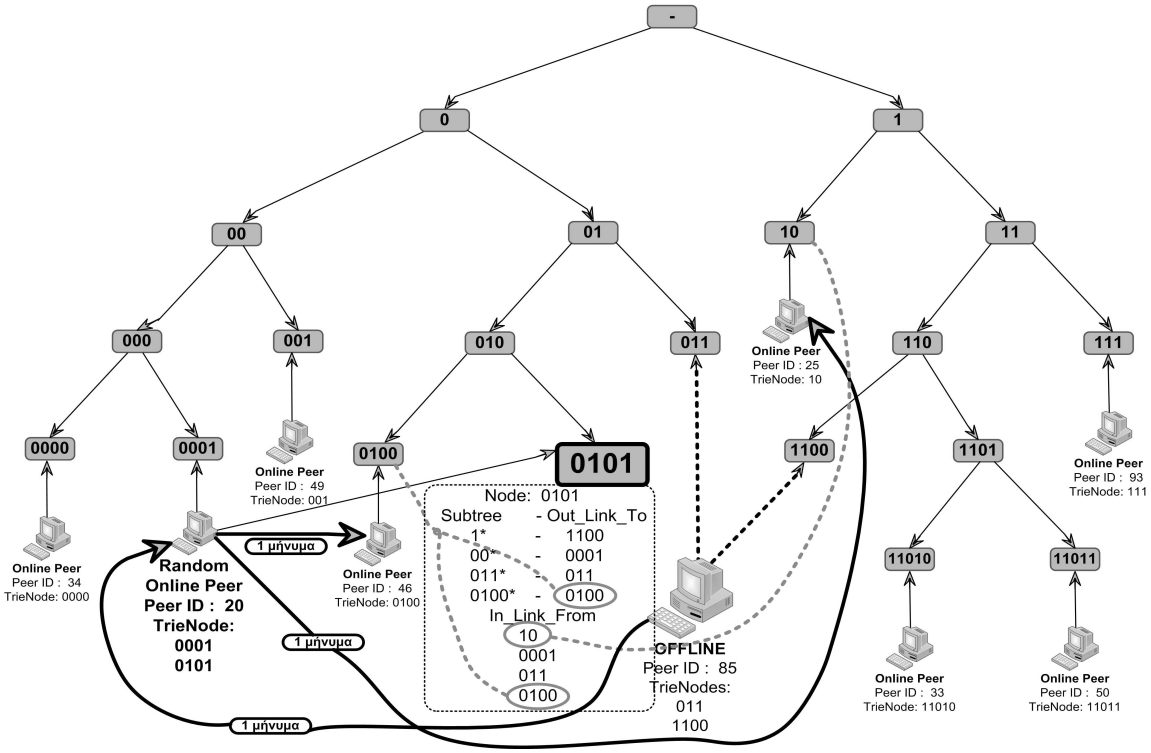
3.3 Αναλυτική περιγραφή της διαδικασίας εξόδου από το σύστημα

Για να μπορέσουμε να περιγράψουμε με ακρίβεια τη διαδικασία εξόδου ενός peer που είναι ήδη online στο σύστημα P-grid θα ήταν εύλογο να το κάνουμε χρησιμοποιώντας το trie του παραδείγματος που χρησιμοποιήσαμε στη προηγούμενη παράγραφο. Ας υποθέσουμε ότι πλέον ο peer 85 είναι υπεύθυνος για τα φύλλα 011, 0101 και 1100 και ότι οι peers 68 και 78 είχαν βγεί εκτός συστήματος κάποια στιγμή δίνοντας στον 85 τα φύλλα για τα οποία ήταν υπεύθυνοι. Αυτό το κάνουμε γιατί έτσι θα καλύψουμε την περίπτωση όπου ο peer που επιθυμεί να βγεί εκτός συστήματος θα έχει πολλά φύλλα και όχι μόνο ένα. Στο σχήμα 3.27 βλέπουμε το σύστημα όπως το περιγράψαμε.



Σχήμα 3.28: Το φύλλο 0101 το αναλαμβάνει ο τυχαίος online peer 20.

Ας υποθέσουμε τώρα ότι ο peer 85 επιθυμεί να βγει εκτός συστήματος. Μόλις κάνει αυτή την αίτηση για έξοδο ο peer 85 θα πρέπει το πρωτόκολο να αναλάβει να βρει online peers που θα αναλάβουν να είναι υπεύθυνοι για τα φύλλα του. Αρχικά το πρωτόκολο επιλέγει κάποιον τυχαίο online peer και του δίνει το ένα από τα φύλλα που είχε ο peer 85. Έστω ότι ο πρώτος τυχαίος peer είναι ο 20 και ότι το πρώτο φύλλο για το οποίο πρέπει να βρεθεί κάποιος peer που θα το αναλάβει είναι το 0101. Προκείμενου να δώσει ο peer 85 το φύλλο 0101 στον peer 20 θα υποθέσουμε ότι στη πράξη πρέπει ο peer 85 να στείλει ένα μήνυμα στον peer 20 με το οποίο θα του δίνει το φύλλο 0101. Επίσης ένα μήνυμα θα έστελνε ο peer 85 ακόμα κι αν ο 20 αναλάμβανε περισσότερα από ένα φύλλα κατά τη διάρκεια αυτής της διαδικασίας.



Σχήμα 3.29: Παρατηρούμε τα μηνύματα που πρέπει να σταλθούν προκειμένου ο peer 20 να γίνει υπεύθυνος για το φύλλο 0101 και να το γνωστοποιήσει στους υπεύθυνους peers των γειτόνων του.

Στο σχήμα 3.29 μπορούμε να δούμε αναλυτικά τα μηνύματα που πρέπει να στείλει ο peer 20 και την σχέση που έχουν εκείνοι οι peers με τα φύλλα προς τα οποία διατηρούσε είτε εισερχόμενο είτε εξερχόμενο link το φύλλο 0101. Με έντονα βέλη και γραμμές βλέπουμε να μηνύματα που θα στείλει ο peer 20 και με διακεκομένες γραμμές τα αντίστοιχα links που υπάρχουν μεταξύ των peers και των φύλλων προς τα οποία διατηρεί είτε εισερχόμενο είτε εξερχόμενο link το φύλλο 0101. Βλέπουμε δηλαδή με πιο τρόπο ενημερώνει ο τυχαίος online peer 20 τους υπεύθυνους peers των γειτονικών φύλλων. Πρέπει δηλαδή όποια φύλλα έχει ως γείτονες και όποια φύλλα έχουν ως γείτονα το φύλλο 0101 να ενημερωθούν για την αλλαγή του υπεύθυνου peer και πιο συγκεκριμένα να ενημερωθούν οι υπεύθυνοι peers των φύλλων αυτών. Δηλαδή όπως φαίνεται και στον πίνακα εξερχομένων links του φύλλου 0101, θα πρέπει ο peer 20 να στείλει μήνυμα στους υπεύθυνους peers των φύλλων 1100, 0001, 011, 0100 και 10 που είναι τα φύλλα προς τα οποία διατηρεί εξερχόμενα και εισερχόμενα links το φύλλο 0101.

Προσέξτε ότι το φύλλο 1100 έχει υπεύθυνο peer τον 85 που επιθμεί να βγεί εκτός συστήματος. Άρα ο εκάστοτε τυχαίος online peer δεν θα είχε νόημα να στείλει μήνυμα στον peer που επιθμεί να βγει εκτός συστήματος αφού αυτός είναι εκείνος που δίνει τα φύλλα του σε τυχαίους online peers. Ας ονομάσουμε K το εκάστοτε φύλλο που πρόκειται να πάρει ο κάθε τυχαίος online peer και ας δούμε προσεκτικά πότε πρέπει ο τυχαίος online peer να στείλει μήνυμα αλλά και σε ποιους:

- Το φύλλο προς το οποίο διατηρεί είτε εισερχόμενο είτε εξερχόμενο link το K πρέπει να έχει σαν υπεύθυνο peer κάποιον διαφορετικό peer από εκείνον για τον οποίο γίνεται η διαδικασία. Δηλαδή στο παράδειγμά μας θα πρέπει το κάθε φύλλο προς το οποίο διατηρεί είτε εισερχόμενο είτε εξερχόμενο link το φύλλο 0101 να έχει υπεύθυνο peer κάποιον διαφορετικό από τον peer 85. Τα φύλλα που ΔΕΝ τηρούν αυτές τις προϋποθέσεις είναι το 1100 και το 011 το οποίο τυχαίνει να ανήκει και σε εισερχόμενο και σε εξερχόμενο link του φύλλου 0101. Άρα αυτό σημαίνει ότι για αυτά τα φύλλα δεν θα στείλει μήνυμα στους υπεύθυνους peers ο 20.

- Το φύλλο προς το οποίο διατηρεί είτε εισερχόμενο είτε εξερχόμενο link το K πρέπει να έχει σαν υπεύθυνο peer κάποιον διαφορετικό peer και όχι τον ίδιο. Δηλαδή στο παράδειγμά μας για να στείλει μήνυμα στους αντίστοιχους peers ο peer 20, θα πρέπει τα φύλλα προς τα οποία διατηρεί είτε εισερχόμενο είτε εξερχόμενο link το φύλλο 0101 να μην έχουν υπεύθυνο peer τον 20. Το φύλλο που δεν τηρεί αυτή τη προϋπόθεση είναι το 0001. Αυτό είναι λογικό καθώς δεν θα είχε νόημα να στείλει ο peer 20 μήνυμα στον εαυτό του για να τον ενημερώσει για ο,τιδήποτε.

- Αν το φύλλο K διατηρεί και εισερχόμενο και εξερχόμενο link προς ένα φύλλο M τότε θα στείλουμε ένα και μόνο μήνυμα στον υπεύθυνο peer του M και όχι δύο ξεχωριστά μηνύματα.

- Επίσης μόνο ένα μήνυμα θα στείλει ο peer 20 και σε κάποιον peer ο οποίος έχει πολλά φύλλα τα οποία μπορεί να ανήκουν είτε σε εισερχόμενα είτε σε εξερχόμενα links του K.

Άρα λοιπόν σύμφωνα με τα παραπάνω ο peer 20 θα στείλει μήνυμα:

- Στον peer 46 που είναι υπεύθυνος για το φύλλο 0100.

- Στον peer 25 που είναι υπεύθυνος για το φύλλο 10.

Η διαδικασία αυτή θα συνεχιστεί για όλα τα φύλλα για τα οποία ήταν υπεύθυνος ο peer 85 που έκανε την αίτηση για έξοδο. Δηλαδή όπως βλέπουμε στο σχήμα 3.30 θα βρεθεί και πάλι ένας νέος τυχαίος online peer , έστω ο 93, ο οποίος θα αναλάβει να είναι υπεύθυνος για το φύλλο 011. Αρχικά ο peer 85 θα στείλει ένα μήνυμα στον 93 προκειμένου να πάρει το φύλλο 011. Στη συνέχεια ο peer 93 θα στείλει ένα μήνυμα σε κάθε έναν από τους online peers οι οποίοι είναι υπεύθυνοι των φύλλων προς τα οποία διατηρείται είτε εισερχόμενο είτε εξερχόμενο link και τα οποία πληρούν τις παραπάνω προϋποθέσεις για την αποστολή μηνύματος. Δηλαδή όπως μπορούμε να δούμε και στο σχήμα 3.30 θα στείλει μήνυμα στους εξής online peers:

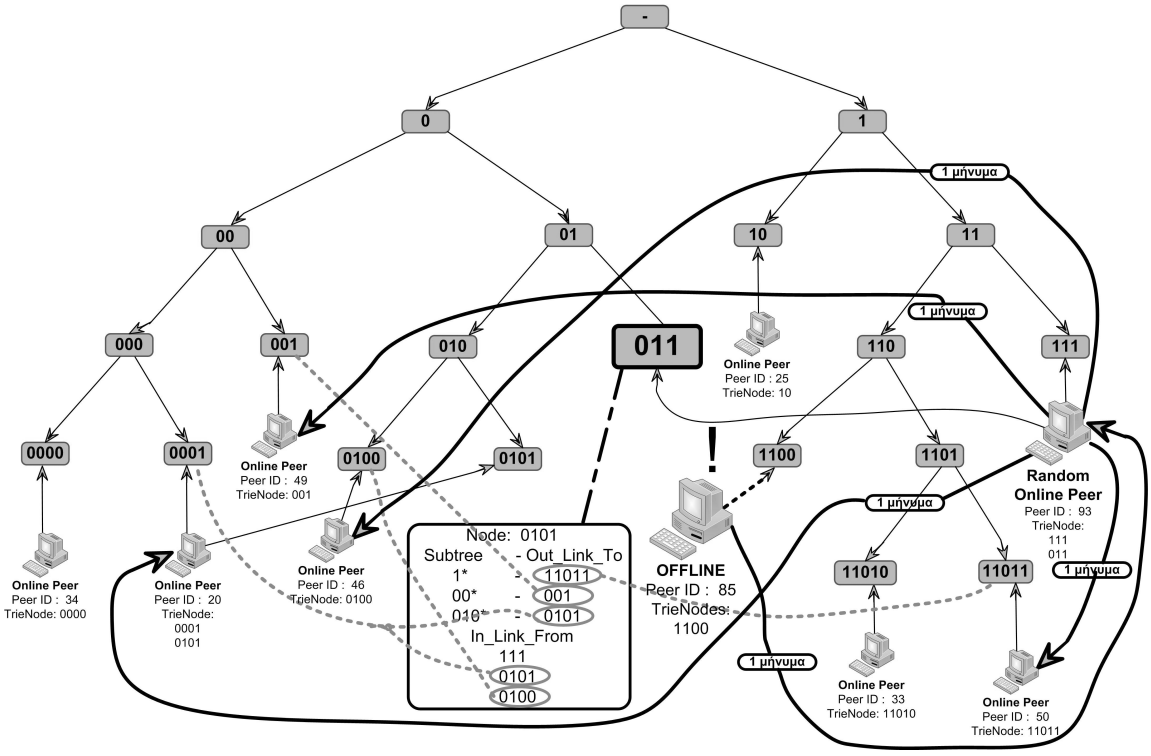
- Στον peer 50 που είναι υπεύθυνος για το φύλλο 11011.

- Στον peer 49 που είναι υπεύθυνος για το φύλλο 001.

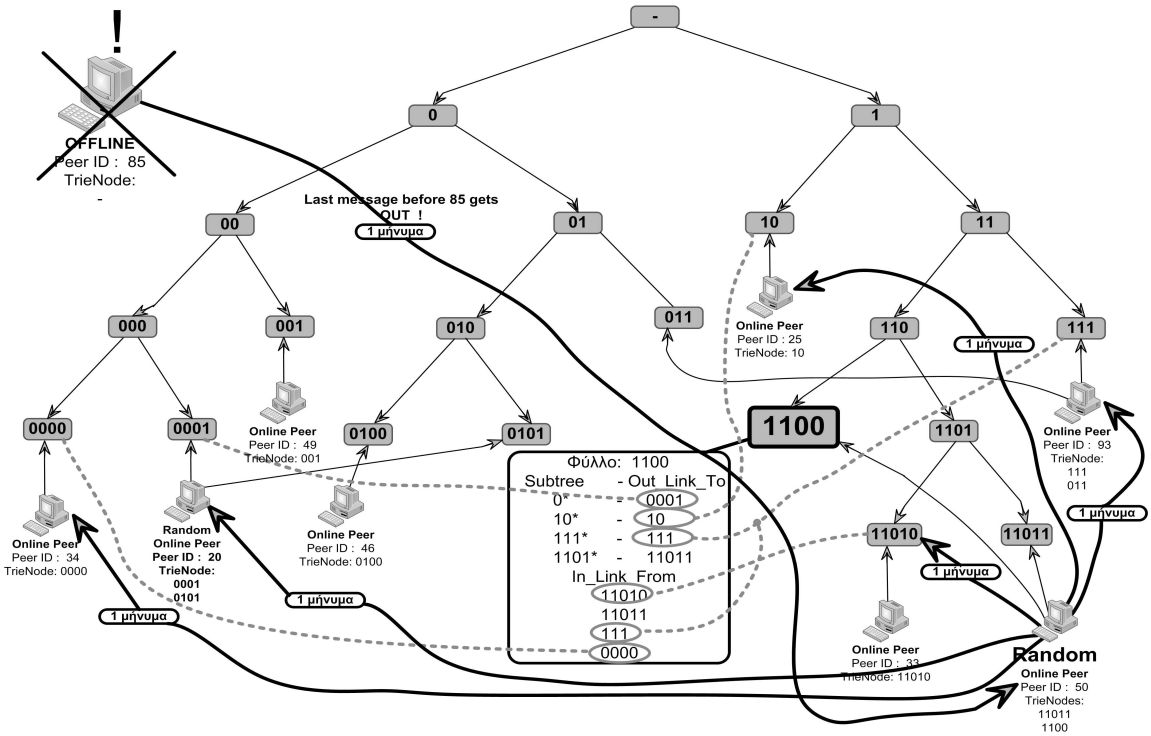
- Στον peer 46 που είναι υπεύθυνος για το φύλλο 0100.

- Στον peer 20 που είναι υπεύθυνος για το φύλλο 0101.

Και τέλος βλέπουμε στο σχήμα 3.31 ότι ο τυχαίος online peer αυτή τη φορά είναι ο 50, ο οποίος θα αναλάβει να είναι υπεύθυνος για το φύλλο 1100. Πλέον ο peer 85 δεν είναι υπεύθυνος για κανένα φύλλο του trie και άρα δεν είναι μέλος του συστήματος οπότε μπορούμε να πούμε ότι η διαδιακασία εξόδου του έχει ολοκληρωθεί.



Σχήμα 3.30: Βλέπουμε τα μηνύματα που πρέπει να σταλθούν προκειμένου ο peer 93 να γίνει υπεύθυνος για το φύλλο 011 και να το γνωστοποιήσει στους υπεύθυνους peers των γειτόνων του.



Σχήμα 3.31: Βλέπουμε τα μηνύματα που πρέπει να σταλθούν προκειμένου ο peer 50 να γίνει υπεύθυνος για το φύλλο 1100 και να το γνωστοποιήσει στους υπεύθυνους peers των γειτόνων του.

Κεφάλαιο 4

Πειράματα - Μελέτη αποτελεσμάτων

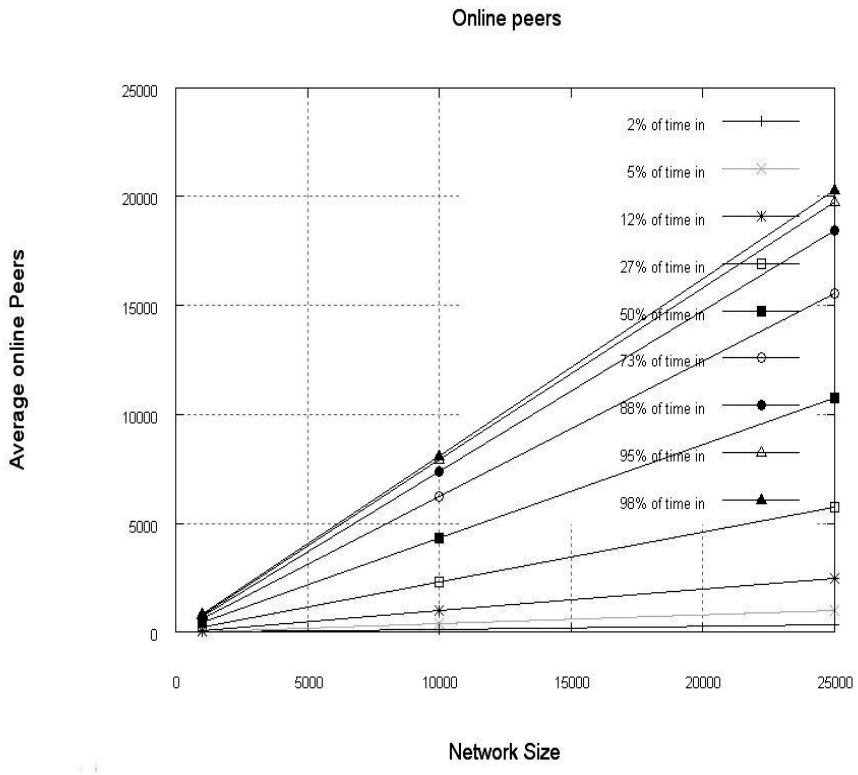
Η εργασία μας επικεντρώνεται στις διαδικασίες εισόδου και εξόδου των peers και σε αυτό το σημείο πρέπει να περιγράψουμε με ποιο τρόπο πραγματοποιήθηκαν αυτές οι χρονικά τυχαίες εισόδοι και εξόδοι τυχαίων peers. Αρχικά θεωρήσαμε ότι έχουμε ένα πλήθος N από peers οι οποίοι θα συμμετέχουν στα πειράματά μας. Τα πειράματα που πραγματοποιήσαμε έγιναν για 1000, 10000, 25000, 75000 και 100000 peers οι οποίοι εισέρχονται και εξέρχονται από το σύστημα σε τυχαίες χρονικές στιγμές. Κάθε ένας από αυτούς τους peers υποθέσαμε ότι θα πραγματοποιεί ένα πλήθος $2r$ εισόδων και εξόδων μαζί. Αυτό που μας ενδιαφέρει και χρειαζόμαστε προκειμένου να πραγματοποιούμε τυχαίες εισόδους και εξόδους κάποιου χρήστη είναι το ποσοστό του χρόνου που θα βρίσκεται εντός και εκτός του συστήματος. Δηλαδή μας ενδιαφέρει το κλάσμα $\frac{duration-online}{duration-offline}$. Αυτό το κλάσμα θεωρήσαμε ότι παίρνει τιμές που ακολουθούν την εκθετική κατανομή με μέσους όρους $\frac{1}{e^4}, \frac{1}{e^3}, \frac{1}{e^2}, \frac{1}{e^1}, 1, e^1, e^2, e^3, e^4$. Ας δούμε με ένα απλό παράδειγμα πως χρησιμοποιήσαμε αυτά τα κλάσματα χρόνου στη πράξη. Έστω ότι έχουμε τη περίπτωση όπου $\frac{duration-online}{duration-offline} = \frac{1}{e^2}$.Αυτό σημαίνει ότι ο κάθε peer βρίσκεται κατά μέσο όρο 1 μονάδα χρόνου [1] online στο σύστημα και επομένως αφού περάσει αυτό το χρονικό διάστημα θα βγεί offline. Στη συνέχεια θα παραμείνει offline για e^2 χρονικές μονάδες. Αφού περάσει κατά μέσο όρο αυτό το διάστημα θα ξαναγίνει online και η διαδικασία επαναλαμβάνεται εώς ότου πραγματοποιήσει $2r$ εισόδους και εξόδους. Στα πειράματα που πραγματοποιήσαμε έγιναν 10 εισόδοι και 10 εξόδοι στο σύστημα για τον κάθε peer. Πρέπει να εξηγήσουμε ότι ως πείραμα θα θεωρήσουμε την εκτέλεση του πρωτοκόλου για ένα συγκεκριμένο πλήθος από peers , ένα συγκεκριμένο κλάσμα χρόνου εντός του συστήματος προς το χρόνο εκτός του συστήματος και μια επανάληψη. Καθώς κατασκευάζαμε τις τυχαίες χρονικά εισόδους και εξόδους του κάθε peer διατηρούσαμε μια μεταβλητή η οποία ήταν η συνολική διάρκεια των εισόδων και εξόδων εκείνου του peer. Δηλαδή πάνω σε αυτή τη διάρκεια του κάθε peer προσθέτονταν τα χρονικά διαστήματα όπου παρέμενε online ή offline ο peer. Κάθε εγγραφή που αφορούσε είσοδο ή έξοδο κάποιου peer είχε σαν στοιχεία τον χρόνο που έγινε, το αναγνωριστικό του peer για τον οποίο πρόκειται κλπ. Αφού φτιάξαμε όλες τις εγγραφές εισόδου και εξόδου όλων των peers ταξινομήσαμε τις εγγραφές αυτές κατά αύξοντα χρόνο που έγιναν. Το ταξινομημένο ως προς τον χρόνο σύνολο αυτών των εγγραφών εισόδου ή εξόδου των peers είναι η σημαντικότερη μεταβλητή των πειραμάτων μας και χρησιμοποιήθηκε προκειμένου να προσομοιώσουμε στη πράξη τις τυχαίες χρονικά εισόδους και εξόδους που συνδέονται αντίστοιχα με τυχαίους peers.

Αφού πλέον έχουμε περιγράψει με ποιο τρόπο πραγματοποιήθηκαν οι τυχαίες χρονικές στιγμές εισόδου και εξόδου θα ήταν πιο πρακτικό να αναφερόμαστε στα κλάσματα του χρόνου εντός προς τον χρόνο εκτός του συστήματος με ποσοστά %. Έτσι για τα παραπάνω κλάσματα που περιγράψαμε θα αναφερόμαστε αντίστοιχα

[1] Ως μονάδα χρόνου θεωρούμε οποιαδήποτε μονάδα. Σε αυτή τη φάση δεν μας ενδιαφέρει αν είναι ώρες, μέρες ή κάτι άλλο αφού θα είναι ίδια μονάδα για κάθε κλάσμα χρόνου και άρα το κάθε κλάσμα θα είναι καθαρός αριθμός. Θεωρήσαμε απλούστερο να θεωρούμε πάντα ως τη μονάδα το χρονικό διάστημα ενός από τα δύο. Δηλαδή είτε ο χρόνος που είναι online είτε ο χρόνος που είναι offline στο σύστημα να είναι 1.

στα ποσοστά 2 %, 5 %, 12 %, 27 %, 50 %, 73 %, 88 %, 95 %, 98 % όπου οι χρήστες βρίσκονται online στο σύστημα. Στο σχήμα 4.1 παρατηρούμε το πλήθος των online peers κατά τη διάρκεια των πειραμάτων μας σε σχέση με το μέγεθος του δικτύου. Μέγεθος δικτύου είναι το πλήθος των peers που συμμετείχαν σε κάθε πείραμα. Όπως έχουμε ήδη αναφέρει τα πειράματα μας έγιναν για 1.000, 10.000, 25.000, 75.000 και 100.000 peers. Στο ίδιο σχήμα παρατηρούμε ότι όταν το ποσοστό του χρόνου που βρίσκεται online ο κάθε peer στο σύστημα γίνει 98 % του χρόνου, το πλήθος των online peers δεν είναι το ανάλογο για κάθε μέγεθος δικτύου. Αυτό ισχύει για όλα τα ποσοστά του χρόνου και θα εξηγήσουμε στη συνέχεια γιατί συμβαίνει κάτι τέτοιο. Ας πάρουμε το δίκτυο για 100.000 peers και τη γραφική παράσταση για το 98 %. Αυτό σημαίνει ότι θα έπρεπε το 98 % των 100.000 peers να είναι κατά μέσο όρο online. Στο ίδιο σχήμα παρατηρούμε ότι το πλήθος των online peers για αυτό το ποσοστό των 100.000 peers είναι περίπου 80.000. Αυτός ο αριθμός είναι μικρότερος από το 98 % των 100.000, δηλαδή του 98.000. Άρα βλέπουμε ότι λείπουν 18.000 online peers από αυτούς που αναλογούν σε αυτό το ποσοστό του χρόνου. Θα καταλάβουμε αμέσως όμως γιατί συμβαίνει αυτό αν σκεφτούμε πως πραγματοποιήσαμε την προσομοίωσή μας. Τον αριθμό των online peers τον μετρήσαμε σε κάθε διαδικασία εισόδου ή εξόδου και ο συνολικός μέσος όρος προέκυψε από όλες αυτές τις μετρήσεις. Αρχικά υποθέσαμε ότι οι 100.000 peers που συμμετείχαν σε κάθε πείραμα ήταν offline. Οπότε αρχικά όλοι οι peers θα πραγματοποιήσουν 100.000 διαδικασίες εισόδου και άρα το πλήθος των online peers θα αυξάνεται ξεκινώντας απ΄το μηδέν. Στη συνέχεια θα υπάρχει ένα χρονικό διάστημα όπου οι peers θα πραγματοποιούν εξόδους και εισόδους ανάλογα με τα ποσοστά χρόνου. Σε εκείνο το διάστημα το 98 % των peers ήταν online στο σύστημα. Επίσης μετά από αυτό το χρονικό διάστημα οι peers θα πραγματοποιήσουν τη τελευταία τους έξοδο. Δηλαδή θα πραγματοποιηθούν 100.000 διαδικασίες εξόδου εώς ότου όλοι οι peers βρεθούν εκτός δικτύου και επομένως οι online peers φτάσουν στο μηδέν. Σε αυτό το διάστημα ο μέσος αριθμός των online peers θα μειωθεί και αυτό δεν γίνεται να μην επηρεάσει τον μέσο αριθμό των online peers όλου του πειράματος. Οι τελευταίες αυτές έξοδοι των peers που είναι 100.000 καθώς και οι πρώτες είσοδοι που ήταν πάλι 100.000 μείωσαν τον αναμενόμενο μέσο όρο των 98.000 online peers στους 80.000. Το ίδιο συμβαίνει για όλα τα μεγέθη δικτύου και για όλα τα ποσοστό του χρόνου για τα οποία πραγματοποιήθηκαν τα πειράματά μας.

Τέλος πρέπει να πούμε ότι για κάθε πλήθος N από peers και για κάθε ένα ποσοστό χρόνου από τα παραπάνω πραγματοποιήσαμε 10 επαναλήψεις του πειράματος όπου κάθε πείραμα είχε διαφορετικές τυχαίες εισόδους και εξόδους των peers. Έτσι πήραμε τις μέσες τιμές των 10 επαναλήψεων προκειμένου να οδηγηθούμε σε αξιόπιστα αποτελέσματα που θα βασίζονται σε πολλές επαναλήψεις του ίδιου πειράματος. Πρέπει επίσης να επισημάνουμε ότι πραγματοποιήσαμε κάποιες εκτελέσεις του πρωτοκόλου που περιγράψαμε η οποίες σχετίζονται με την λειτουργία ή όχι της συνάρτησης exchange. Έτσι πραγματοποιήσαμε εκτελέσεις του πρωτοκόλου για τις περιπτώσεις όπου η συνάρτηση exchange χρησιμοποιήθηκε με πιθανότητα 1/3, με πιθανότητα 1/9 καθώς και την περιίπτωση όπου δεν χρησιμοποιήθηκε καθόλου. Αυτή η τελευταία περίπτωση όπως θα δούμε στη συνέχεια έφερε τα καλύτερα αποτελέσματα. Τέλος σκεφτήκαμε να πραγματοποιήσουμε μια παραλλαγή του πρωτοκόλου η οποία αφορά την προώθηση του αρχικού μηνύματος για αίτηση νέου μέλους που είδαμε αναλυτικά στο κεφάλαιο 3.2.2. Σκεφτήκαμε ότι όταν ένα φύλλο του trie ελέγχει τα εξερχόμενα links που διατηρεί προκειμένου να βρει εκείνον τον γείτονα που έχει το μεγαλύτερο κοινό πρόθεμα με αυτό του προορισμού, θα μπορούσαμε να ελέγξουμε και τα εισερχόμενα links για ακριβώς τον ίδιο σκοπό. Δηλαδή θα μπορούσαμε να χρησιμοποιήσουμε ως routing table εκτός από τον πίνακα των εξερχομένων links και τον πίνακα των εισερχομένων links κάτι που όπως θα δούμε στη συνέχεια βελτιώνει την απόδοση του συστήματος για το latency. Επίσης στις εκτελέσεις αυτού του είδους δεν χρησιμοποιήθηκε καθόλου η συνάρτηση exchange αφού αυτό όπως ήδη αναφέραμε έφερε καλύτερα αποτελέσματα για όλα τα μεγέθη που μετρήθηκαν.



Σχήμα 4.1: Παρατηρούμε τον μέσο όρο των online peers σε σχέση με το μέγεθος του δικτύου ανάλογα με τα ποσοστά χρόνου που ακολουθούν οι peers.

4.1 Μελέτη του χρόνου των διαδικασιών εισόδου και εξόδου

Σε αυτό το σημείο πρέπει να εξηγήσουμε πως συνδέεται ο χρόνος κατά τον οποίο βρίσκονται εντός η εκτός συστήματος οι peers με την απόδοση του συστήματος. Όπως θα δούμε σε επόμενη παράγραφο το Maximum throughput είναι το σημαντικότερο χαρακτηριστικό του συστήματος αφού μας δείχνει πρακτικά πως αλλάζει ο μέγιστος αριθμός διαδικασιών εισόδων ή/και εξόδων που μπορεί να εξυπηρετήσει το σύστημα στη μονάδα του χρόνου χωρίς να υπερφορτωθεί κάποιος peer. Περισσότερες λεπτομέρειες για τα αποτελέσματα του Maximum throughput θα δούμε στη παράγραφο 4.2.

Κάθε peer που συμμετείχε στο πείραμα θεωρούμε ότι πραγματοποιεί r εισόδους και r εξόδους. Έστω ότι T_{in} είναι ο μέσος χρόνος που παραμένει εντός συστήματος και T_{out} ο μέσος χρόνος που παραμένει εκτός συστήματος ο κάθε peer. Άρα η συνολική διάρκεια της συμμετοχής του κάθε peer στο πείραμα είναι $T_{total} = r(T_{in} + T_{out})$. Είναι πολύ σημαντικό να πούμε ότι κατά μεσό όρο αυτή η διάρκεια δεν ισχύει μόνο για κάθε peer αλλά είναι η διάρκεια του συνολικού πειράμματος κάθε φορά. Ο κάθε peer συμμετέχει για χρόνο $r(T_{in} + T_{out})$ κάτα μέσο όρο αλλά μέσα σε αυτό το χρόνο θα πραγματοποιηθούν και όλες οι διαδικασίες εισόδου και εξόδου όλων των peers. Οι χρονικές στιγμές εισόδου ή εξόδου του κάθε peer είναι τυχαία κατανεμημένες μέσα σε αυτό το χρονικό διάστημα. Αν στο πείραμα συμμετείχαν N peers τότε τα συνολικά procceses του πειράματος θα είναι $N \cdot 2r$. Επομένως μπορούμε να πούμε ότι το σύστημα εξυπηρετεί συνολικά κατά μέσο όρο $N \cdot 2r$ διαδικασίες εισόδου και εξόδου σε χρόνο $T_{total} = r(T_{in} + T_{out})$, δηλαδή οι διαδικασίες εισόδου και εξόδου που εξυπηρετεί το σύστημα ανα μονάδα χρόνου θα είναι:

$$\frac{\#Processes}{TotalDuration} = \frac{N \cdot 2r}{r(T_{in} + T_{out})} = \frac{2 \cdot N}{T_{in}(1 + \frac{T_{out}}{T_{in}})}$$

Το κλάσμα στο οποίο καταλήξαμε είναι ο αριθμός των εισόδων και των εξόδων που εξυπηρετεί το σύστημα στη μονάδα του χρόνου όταν οι διάρκεια εντός και εκτός του συστήματος για τον κάθε peer είναι T_{in} και T_{out} αντίστοιχα. Το κλάσμα αυτό

είναι η θεωρητική προσέγγιση για το πλήθος των διαδικασιών που εξυπηρετεί το σύστημα στη μονάδα του χρόνου και θα πρέπει να είναι μικρότερο ή ίσο από το Maximum throughput που μετρήσαμε για το συγκεκριμένο πείραμα κάθε φορά. Δηλαδή θα πρέπει να ισχύει:

$$\frac{2 \cdot N}{T_{in}(1 + T_{out}/T_{in})} \leq \frac{\Lambda_{max}}{E[s]}$$

, όπου $E[s]$ είναι ο μέσος χρόνος που απαιτείται για τη λήψη ενός μηνύματος από τον κάθε peer [2].

Αν την ίδια ανισότητα την λύσουμε ως προς T_{in} θα γίνει:

$$T_{in} \geq \frac{2N \cdot E[s]}{\Lambda_{max}(1 + T_{out}/T_{in})}$$

Παρατηρώντας την παραπάνω ανισότητα μπορούμε να πούμε ότι το δεξί μέλος μας δείχνει ποιος είναι ο ελάχιστος χρόνος που μπορεί να παραμείνει κάποιος peer εντός του συστήματος προκειμένου το σύστημα να μπορεί ακόμα να εξυπηρετεί το maximum αριθμό διαδικασιών εισόδου και εξόδου στη μονάδα του χρόνου όταν η σχέση που διατηρούν ο χρόνος εκτός του συστήματος με τον χρόνο εντός του συστήματος είναι T_{out}/T_{in}.

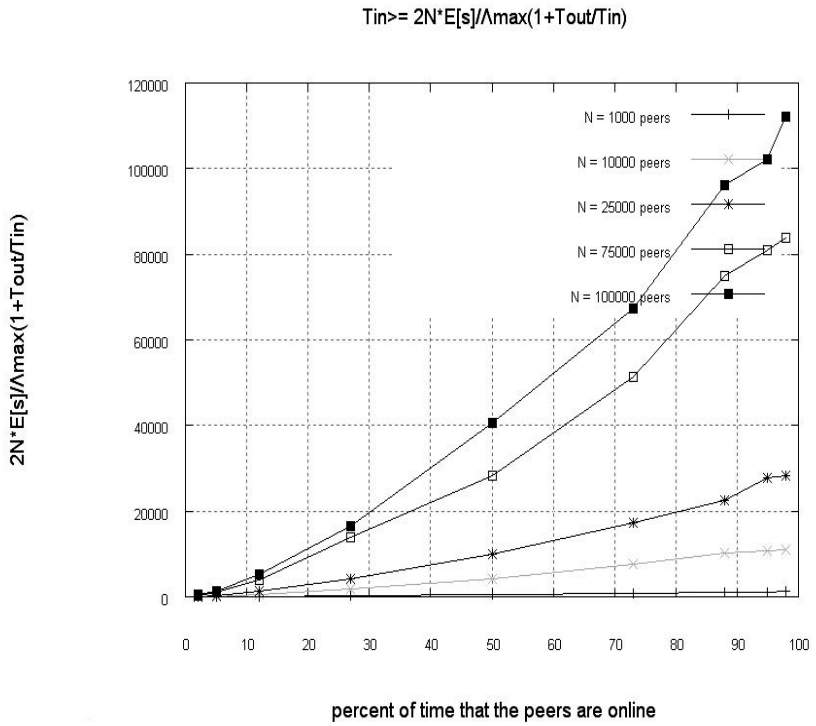
Στο σχήμα 4.2 φαίνεται πως αλλάζει το ελάχιστο T_{in} που θα μπορούσε να έχει κάποιος peer έτσι ώστε το σύστημα να μπορεί ακόμα να εξυπηρετεί το maximum αριθμό διαδικασιών εισόδου ή εξόδου στη μονάδα του χρόνου ενώ στο σχήμα 4.24 (σελιδα 61******) παρατηρούμε το Λ_{max} για τις αντίστοιχες διαδικασίες εισόδου και εξόδου. Στο σχήμα 4.2 παρατηρούμε ότι όταν αυξάνεται το ποσοστό του χρόνου που βρίσκονται εντός συστήματος οι peers αλλά και όταν αυξάνεται το πλήθος των peers που συμμετείχαν στο πείραμα, αυξάνεται και ο ελάχιστος χρόνος που θα πρέπει να παραμείνει ο κάθε peer μέσα στο σύστημα.

Στο σχήμα 4.3 παρατηρούμε πως αλλάζει ο ελάχιστος χρόνος που θα πρέπει να παραμείνει κάποιος peer εντός του συστήματος σε σχέση με τους online peers που υπάρχουν κάθε φορά. Με απλά λόγια παρατηρούμε τον ελάχιστο χρόνο που πρέπει να παραμείνει κάθε peer στο σύστημα όταν το πλήθος των online peers που υπάρχουν εκείνη τη στιγμή είναι αυτό που φαίνεται στον οριζόντιο άξονα.[3] Καθώς αυξάνονται οι online peers στο σύστημα αυξάνεται και το ελάχιστο T_{in} που θα πρέπει να τηρεί ο κάθε peer. Επίσης παρατηρούμε ότι για το ίδιο πλήθος από online peers , με την αύξηση του ποσοστού του χρόνου που βρίσκονται online οι peers στο σύστημα αυξάνεται και το ελάχιστο T_{in}. Αυτό όπως θα δούμε στη συνέχεια έχει άμεση σχέση με το γεγονός ότι τα μηνύματα που λαμβάνει ο πιο φορτωμένος peer αυξάνονται όταν αυξάνεται και το ποσοστού του χρόνου που βρίσκονται online οι peers , δηλαδή το κλάσμα T_{in}/T_{out}. Επομένως στη παραπάνω σχέση του T_{in} θα υπάρχει ένα μικρότερο Λ_{max} και ένα μικρότερο T_{out}/T_{in} καθώς αυξάνεται το κλάσμα T_{in}/T_{out} που καθορίζει το ποσοστό του χρόνου που βρίσκονται εντός συστήματος οι peers. Με αυτό το τρόπο επαληθεύεται η βασική παρατήρηση του σχήματος 4.3 ότι όταν το ποσοστό του χρόνου εντός του συστήματος αυξάνεται για τον κάθε peer, τότε αυξάνεται και ο ελάχιστος χρόνος που θα πρέπει να παραμείνει online προκειμένου να σταλθούν τα απαραίτητα μηνύματα που απαιτούνται για την είσοδό του στο σύστημα.

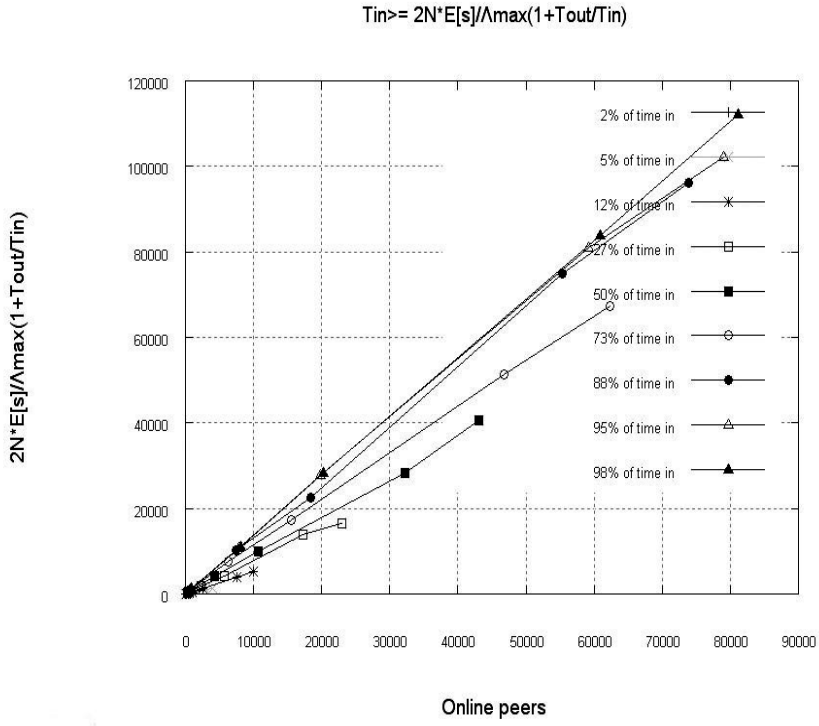
Επομένως γνωρίζουμε πλέον με ποιο τρόπο συνδέονται τα χρονικά διαστήματα κατά τα οποία οι peers βρίσκονται εντός και εκτός του συστήματος με το Λ_{max} καθώς και τη σχέση που διατηρούν αυτά τα μεγέθη μεταξύ τους. Στη συνέχεια

[2] Στη πράγραφο 4.3 θα δούμε αναλυτικά πως υπολογίσαμε το Maximum throughput. Σύμφωνα με το [5] γνωρίζουμε ότι ισχύει $\Lambda_{max} = \frac{1}{E[s] \cdot max_j \mu_j}$. Στα πειράματά μας θεωρήσαμε ότι $E[s] = 1$ οπότε αν θέλουμε να εκφράσουμε τα αποτελέσματα με πραγματικές μονάδες χρόνου θα μπορούσαμε απλά να διαιρέσουμε το Λ_{max} με το $E[s]$ που θα πρέπει να είναι ένας ρεαλιστικός μέσος χρόνος που απαιτείται για τη λήψη ενός μηνύματος από τον κάθε peer.

[3] Πρέπει να επισημάνουμε σε αυτό το σημείο ότι στην παραπάνω σχέση του T_{in} το μέγεθος Λ_{max} και το κλάσμα T_{out}/T_{in} διατηρούν την ίδια ακριβώς σχέση για κάθε ένα απο τα πειράματα που πραγματοποιήσαμε είτε χρησιμοποιείται η συνάρτηση exchange είτε όχι. Έτσι θεωρήσαμε πιο κατανοητό και απλούστερο να μην αναφερθούμε στα T_{in} όλων των πειραμάτων αφού ο σκοπός της ανάλυσης του χρόνου αυτού δεν επηρεάζεται από τα διαφορετικά πειράματα που υλοποιήσαμε. Έτσι επιλέξαμε τυχαία την περίπτωση όπου η συνάρτηση exchange χρησιμοποιείται και η εκτέλεση του πρωτοκόλου έγινε με volume balance.



Σχήμα 4.2: Παρατηρούμε πως αλλάζει ο ελάχιστος χρόνος που θα μπορούσε να παραμείνει κάποιος peer εντός του συστήματος έτσι ώστε το σύστημα να μπορεί ακόμα να εξυπηρετεί τον maximum αριθμό διαδικασιών εισόδου ή εξόδου στη μονάδα του χρόνου χωρίς ωστόσο να υπερφορτωθεί κάποιος peer σε σχέση με το ποσοστό του χρόνου που βρίσκονται online οι peers. (volume balance)



Σχήμα 4.3: Παρατηρούμε πως αλλάζει ο ελάχιστος χρόνος που θα μπορούσε να παραμείνει κάποιος peer εντός του συστήματος έτσι ώστε το σύστημα να μπορεί ακόμα να εξυπηρετεί τον maximum αριθμό διαδικασιών εισόδου ή εξόδου στη μονάδα του χρόνου χωρίς ωστόσο να υπερφορτωθεί κάποιος peer σε σχέση με τους online peers που υπάρχουν στο σύστημα εκείνη τη στιγμή.(volume balance)

του κεφαλαίου θα δούμε αναλυτικά τα σημαντικότερα κριτήρια για την μελέτη του πρωτοκόλου και θα δούμε ότι το ποσοστό του χρόνου που βρίσκονται online οι peers στο σύστημα συνδέεται άμεσα με το πλήθος των μηνυμάτων που στέλνονται, το πλήθος των φύλλων που δημιουργούνται καθώς και με τον μέγιστο αριθμό διαδικασιών εισόδου ή/ και εξόδου που μπορεί να εξυπηρετήσει το σύστημα.

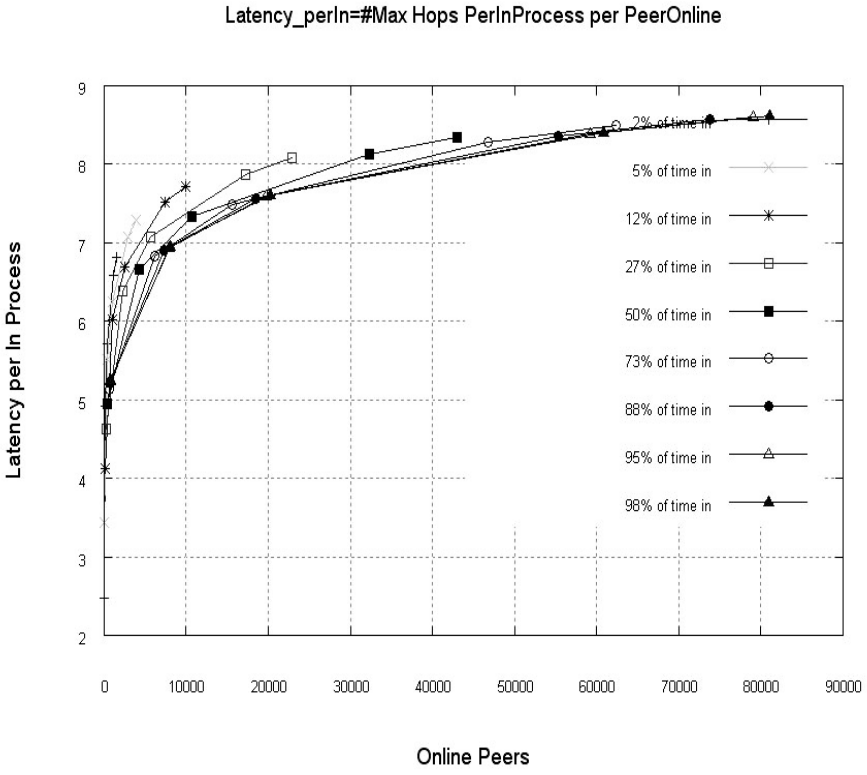
4.2 Latency

Ας δούμε αρχικά τα αποτελέσματα που πήραμε για το latency. Όπως έχουμε ήδη περιγράψει στο κεφάλαιο 1.1.1 το latency είναι εκείνος ο αριθμός που μας δείχνει τον μέγιστο αριθμό προωθήσεων του αρχικού μηνύματος για αίτηση νέας εισαγωγής προς κάποιους online peers έως ότου φτάσει το μήνυμα σε εκείνον που θα αναλάβει να κάνει τον νέο peer μέλος του συστήματος P-grid. Αυτός ο μέγιστος αριθμός προωθήσεων υπολογίστηκε από όλες τις διαδικασίες εισόδου που πραγματοποιήθηκαν για το αντίστοιχο πείραμα κάθε φορά. Το κάθε πείραμα πραγματοποιήθηκε 10 φορές και στη συνέχεια πήραμε τον μέσο όρο των 10 μετρήσεων του μέγιστου αριθμού προωθήσεων που πραγματοποιήθηκαν κάθε φορά. Για μια διαδικασία εξόδου αυτό το μέγεθος είναι πάντα ένα. Αυτό συμβαίνει γιατί ο peer ο οποίος αρχίζει τη διαδικασία εξόδου του, στέλνει ένα μήνυμα σε κάθε έναν απο τους online peers που αναλαμβάνει κάποιο από τα φύλλα του. Το μήνυμα αυτό στέλνεται απ΄ ευθείας από τον peer που επιθυμεί να βγεί εκτός του συστήματος χωρίς να μεσολαβούν άλλοι peers , σε ένα μόνο βήμα. Οπότε πρακτικά βλέπουμε ότι η τιμή του latency μας ενδιαφέρει μόνο για τις διαδικασίες εισόδου.

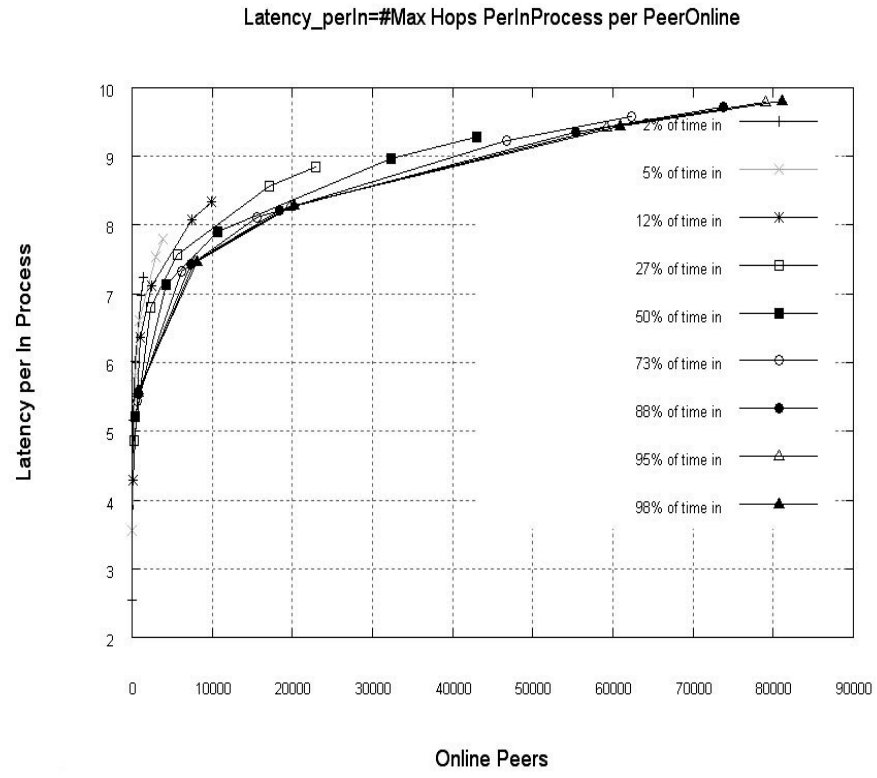
Στην περίπτωση της εισαγωγής ενός νέου peer το αρχικό μήνυμα για αίτηση νέου μέλους προωθείται αρχικά από τον νέο peer που θέλει να γίνει μέλος του συστήματος προς κάποιον τυχαίο online peer. Στη συνέχεια όπως έχουμε περιγράψει στο κεφάλαιο 3.1 και 3.2 εκείνος ο peer επιλέγει ένα τυχαίο φύλλο από αυτά για τα οποία είναι υπεύθυνος και μέσω μιας επαναλαμβανόμενης διαδικασίας ο υπεύθυνος peer κάθε φύλλου προωθεί το μήνυμα της αίτησης στον υπεύθυνο peer του γείτονα εκείνου που είναι πιο κοντά, δηλαδή που έχει μεγαλύτερο κοινό πρόθεμα σαν κλειδί με αυτό του προορισμού. Στο σχήμα 4.4 βλέπουμε τα αποτελέσματα για το latency των πειραμάτων που πραγματοποιήσαμε όταν η επιλογή του φύλλου που θα αναλάβει να δώσει κάποιο φύλλο στον νέο peer γίνει με volume balance ενώ στο σχήμα 4.5 βλέπουμε τα ίδια αποτελέσματα όταν η επιλογή γίνει με data balance.

Παρατηρούμε ότι για όλα τα ποσοστά χρόνου εντός του συστήματος το latency είναι σχεδόν το ίδιο και ειδικότερα για τα ποσοστά που ξεπερνούν το 50%. Σε αυτό το σημείο είναι σημαντικό να δούμε το σχήμα 4.6 όπου παρατηρούμε πως αλλάζει το πλήθος των φύλλων του trie σε σχέση με το μέγεθος του δικτύου για κάθε διαφορετικό ποσοστό χρόνου που είναι online οι peers. Επομένως είναι εύλογο να σκεφτούμε το τρόπο που δημιουργούνται νέα φύλλα σε σχέση με τα ποσοστά χρόνου που βρίσκονται online οι peers. Στην έναρξη κάθε εκτέλεσης υποθέτουμε ότι όλοι οι peers είναι offline, έτσι αρχικά θα υπάρχει μια χρονική περίδο κατά την οποία εισέρχονται όλοι οι peers στο σύστημα για πρώτη φορά ενώ οι διαδικασίες εξόδου που θα μεσολαβούν θα είναι σχεδόν αμελητέες. Τότε το πλήθος των φύλλων του trie αμέσως μετά από αυτή τη χρονική περίοδο θα είναι περίπου όσοι και οι peers που συμμετείχαν στο πείραμα αφού για κάθε είσοδο θα δημιουργείται και ένα φύλλο.

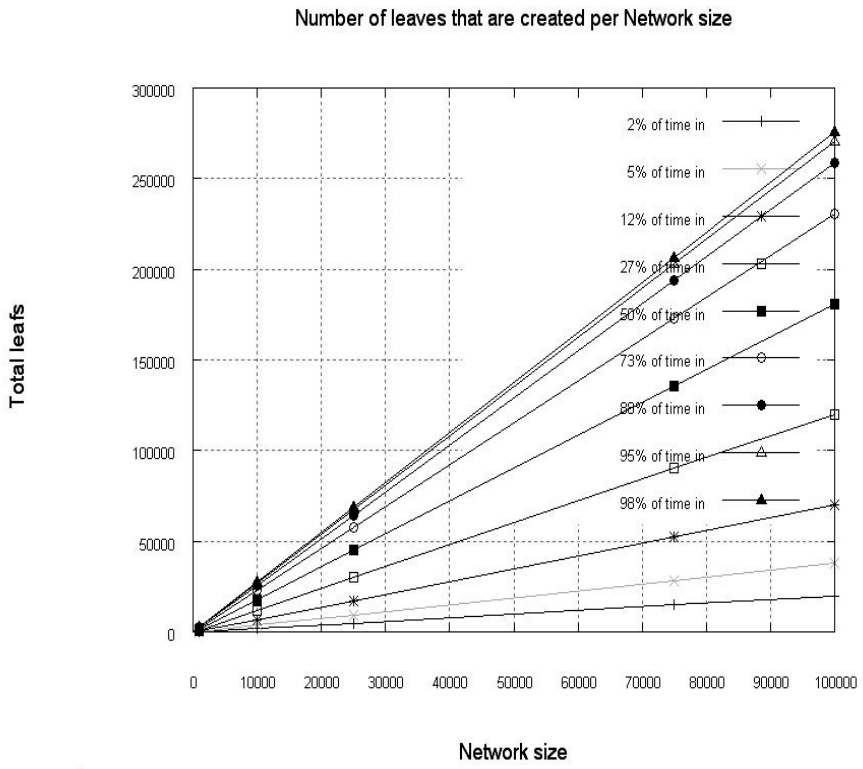
Σύμφωνα με το σχήμα 4.7 παρατηρούμε ότι καθώς αυξάνονται οι online peers αυξάνεται και το πλήθος των φύλλων κάτι που ήταν αναμενόμενο. Επίσης για το ίδιο πλήθος από online peers , όταν το ποσοστό του χρόνου εντός του συστήματος γίνεται μεγαλύτερο τότε το πλήθος των φύλλων δεν αλλάζει πολύ. Παρατηρούμε ότι το πλήθος των φύλλων που δημιουργούνται για τα ποσοστά χρόνου εντός του συστήματος από 2% ως 50 % γίνεται κατά 20-30 % μικρότερο ενώ για τα ποσοστά χρόνου εντός του συστήματος που είναι πάνω απο 50% παρατηρούμε ότι το πλήθος των φύλλων που δημιουργούνται δεν αλλάζει σχεδόν καθόλου. Αυτές οι παρατηρήσεις μας οδηγούν στο συμπέραμσα ότι μετά από τη χρονική περίοδο κατά την οποία εισέρχονται για πρώτη φορά στο σύστημα οι peers, το πλήθος των φύλλων που δημιουργούνται σχετίζεται με το ποσοστό του χρόνου που βρίσκονται εντός συστήματος οι peers με κάποιο συγκεκριμένο τρόπο: Όταν οι peers παραμένουν εντός του συστήματος για λιγότερο χρονικό διάστημα από ότι εκτός, δηλαδή



Σχήμα 4.4: Το latency σε σχέση με τους online peers που βρίσκονται στο σύστημα τη χρονική στιγμή της εισαγωγής του νέου χρήστη.(volume balance)



Σχήμα 4.5: Το latency σε σχέση με τους online peers που βρίσκονται στο σύστημα τη χρονική στιγμή της εισαγωγής του νέου χρήστη.(data balance)



Σχήμα 4.6: Ο αριθμός των φύλλων του trie που έχουν δημιουργηθεί σε σχέση με το πλήθος των peers που συμμετείχαν για κάθε ένα από τα ποσοστά χρόνου που πραγματοποιήθηκαν.

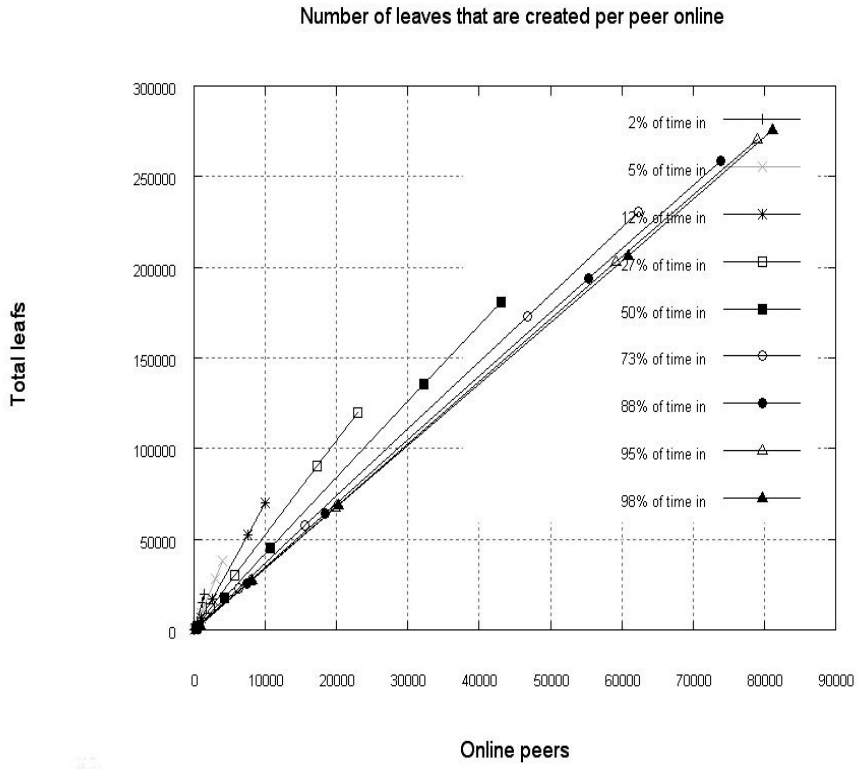
όταν το ποσοστό του χρόνου εντός είναι μικρότερο του 50 %, τότε τα φύλλα που δημιουργούνται είναι πάντα λίγο περισσότερα.

Ας κοιτάξουμε πάλι το σχήμα 4.4 που δείχνει το latency για τις εισόδους των peers. Παρατηρούμε ότι η αλλάγή του latency όταν αλλάζει το ποσοστό του χρόνου που βρίσκονται online οι peers είναι αρκετά μικρή. Επίσης αν συγκρίνουμε τα σχήματα 4.4 και 4.5 όπου βλέπουμε τα αποτελέσματα που προέκυψαν για το latency με volume balance και data balance αντίστοιχα, θα δούμε ότι δεν υπάρχουν σημαντικές διαφορές στη μορφή της γραφικής παράστασης παρά μόνο ότι το latency είναι γενικότερα λίγο μεγαλύτερο στη περίπτωση του data balance. Αυτό σημαίνει ότι όταν χρησιμοποιούμε volume balance κάθε διαδικασία εισόδου πραγματοποιείται πιο γρήγορα αφού το αρχικό μήνυμα φτάνει στον προορισμό του σε λιγότερα βήματα. Θα μπορούσαμε να ισχυριστούμε σε αυτό το σημείο ότι η μέθοδος volume balance για την εύρεση του φύλλου-προορισμού είναι προτιμότερη όταν πρόκειται για το latency. Αυτό πάλι με τη σειρά του μας δείχνει ότι όταν το trie είναι ισορροπημένο τότε το latency βελτιώνεται αρκετά.

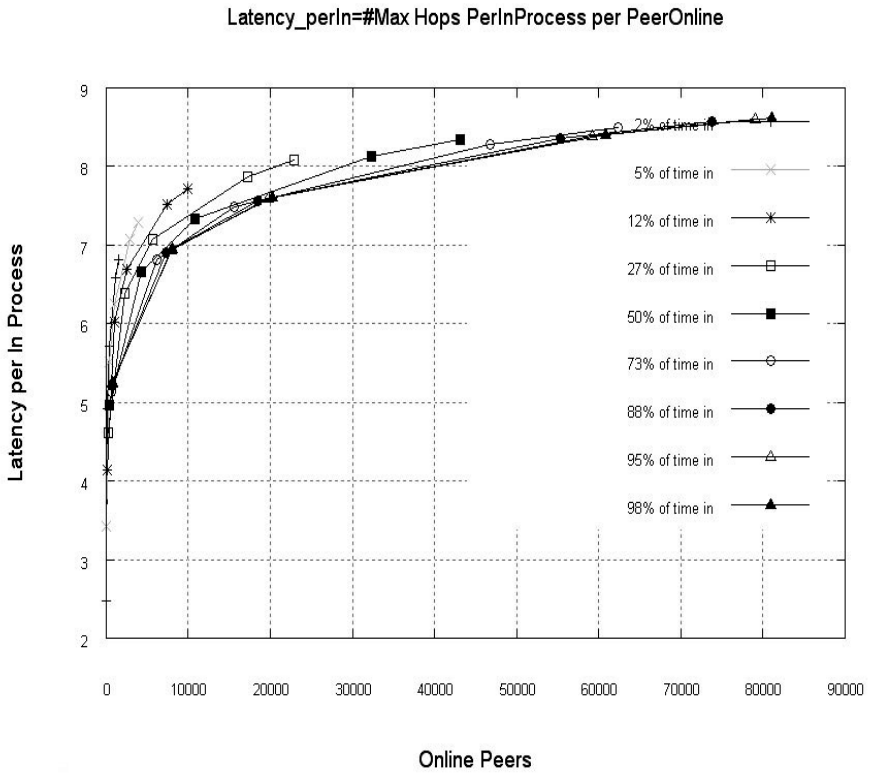
Στο σχήμα 4.8 παρατηρούμε το latency για τις εισόδους των peers όταν η συνάρτηση exchange λαμβάνει μέρος στις διαδικασίες εισόδου με πιθανότητα 1/3, στο σχήμα 4.9 με πιθανότητα 1/9, στο σχήμα 4.10 παρατηρούμε το latency για τις εισόδους όταν δεν χρησιμοποιείται καθόλου η συνάρτηση exchange και τέλος στο σχήμα 4.11 παρατηρούμε τη περίπτωση όπου δεν χρησιμοποιείται η συνάρτηση exchange ενώ συγχρόνως χρησιμοποιούμε ως routing table των πίνακα εισερχομένων και εξερχομένων links μαζί.

Από τα σχήματα 4.8, 4.9, 4.10 παρατηρούμε ότι το latency δεν επηρεάζεται καθόλου από τη χρήση ή όχι της συνάρτησης exchange. Αυτό είναι πολύ σημαντικό αφού ο σκοπός αυτής της συνάρτησης όπως έχουμε ήδη εξηγήσει στο κεφάλαιο 3.1.3 είναι να γίνει πιο γρήγορη η προώθηση του αρχικού μηνύματος για αίτηση νέου μέλους προς τον προορισμό του. Παρατηρούμε λοιπόν ότι η λειτουργία της συνάρτησης exchange δεν έφερε σημαντικά αποτελέσματα στο πρωτόκολο που υλοποιήσαμε αφού είναι ολοφάνερο ότι το latency δεν επηρεάζεται από τη χρήση ή όχι της συνάρτησης αυτής.

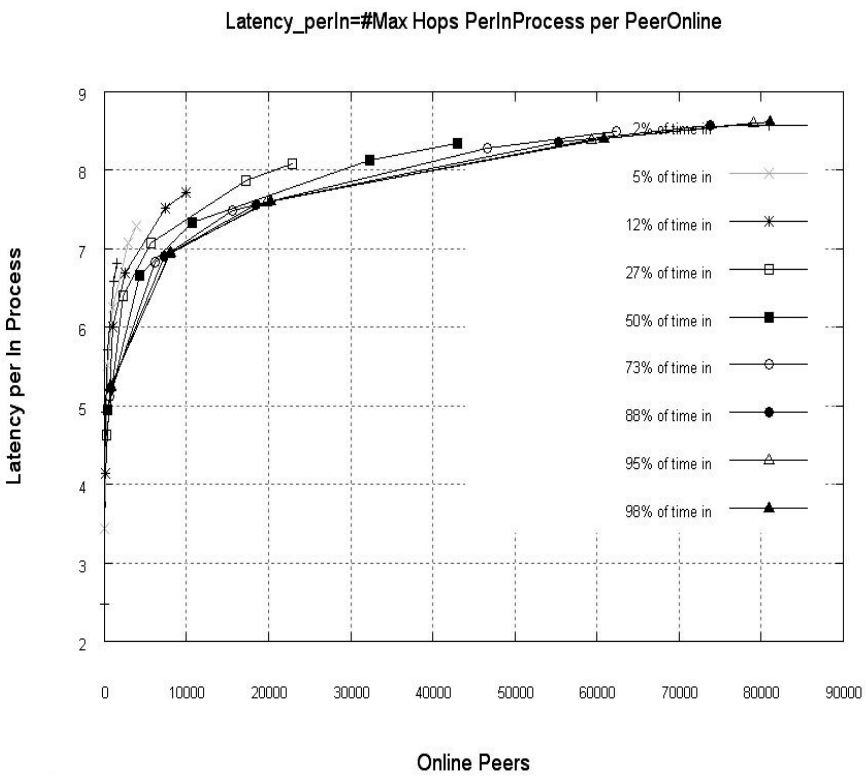
Αντίθετα στο σχήμα 4.11 παρατηρούμε ότι όταν χρησιμοποιούμε τα εισερχόμ-



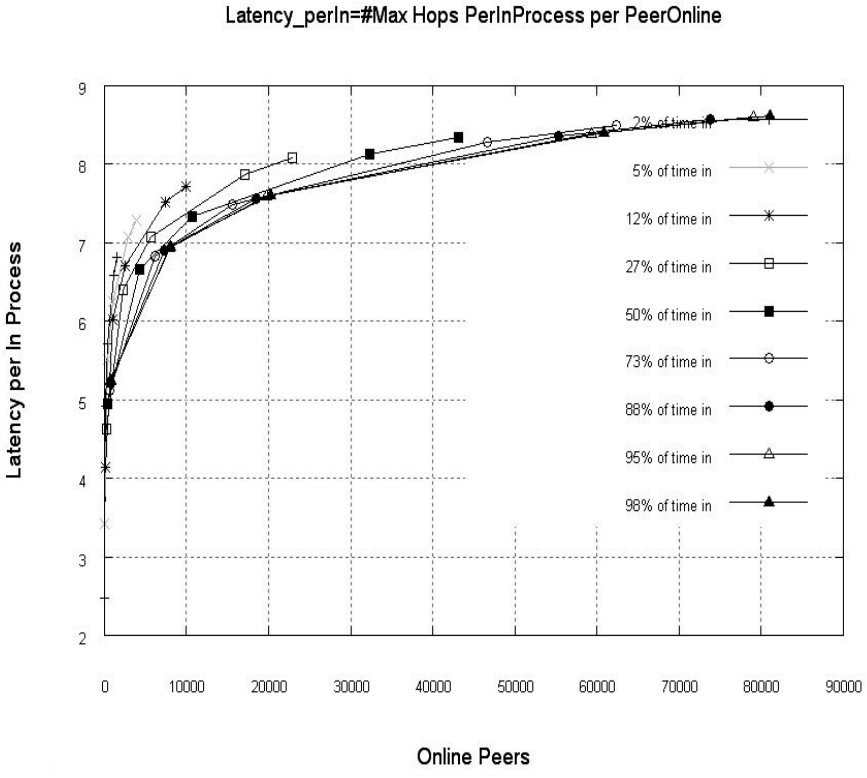
Σχήμα 4.7: Ο αριθμός των φύλλων του trie που έχουν δημιουργηθεί σε σχέση με τους online peers που βρίσκονται στο σύστημα και που είναι υπεύθυνοι για αυτά τα φύλλα.



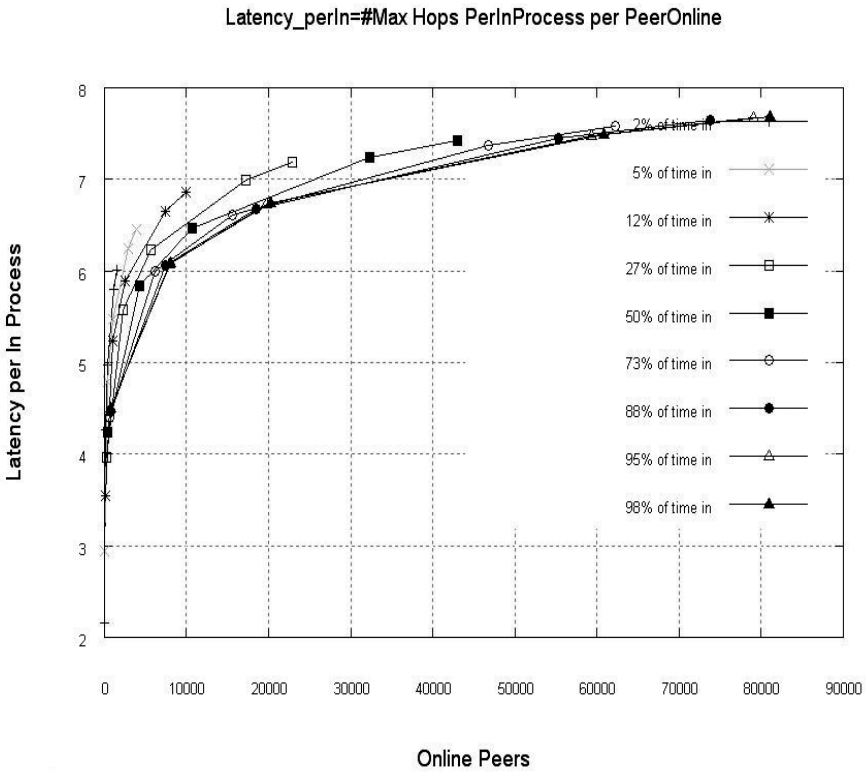
Σχήμα 4.8: Το latency σε σχέση με τους online peers που βρίσκονται στο σύστημα όταν η συνάρτηση exchange χρησιμοποιείται με πιθανότητα 1/3.(volume balance)



Σχήμα 4.9: Το latency σε σχέση με τους online peers που βρίσκονται στο σύστημα όταν η συνάρτηση exchange χρησιμοποιείται με πιθανότητα 1/9.(volume balance)



Σχήμα 4.10: Το latency σε σχέση με τους online peers που βρίσκονται στο σύστημα όταν η συνάρτηση exchange δεν χρησιμοποιείται ποτέ.(volume balance)



Σχήμα 4.11: Το latency σε σχέση με τους online peers που βρίσκονται στο σύστημα όταν η συνάρτηση exchange δεν χρησιμοποιείται ποτέ ενώ συγχρόνως χρησιμοποιούμε ως routing table των πίνακα εισερχομένων και εξερχομένων links μαζί.(volume balance)

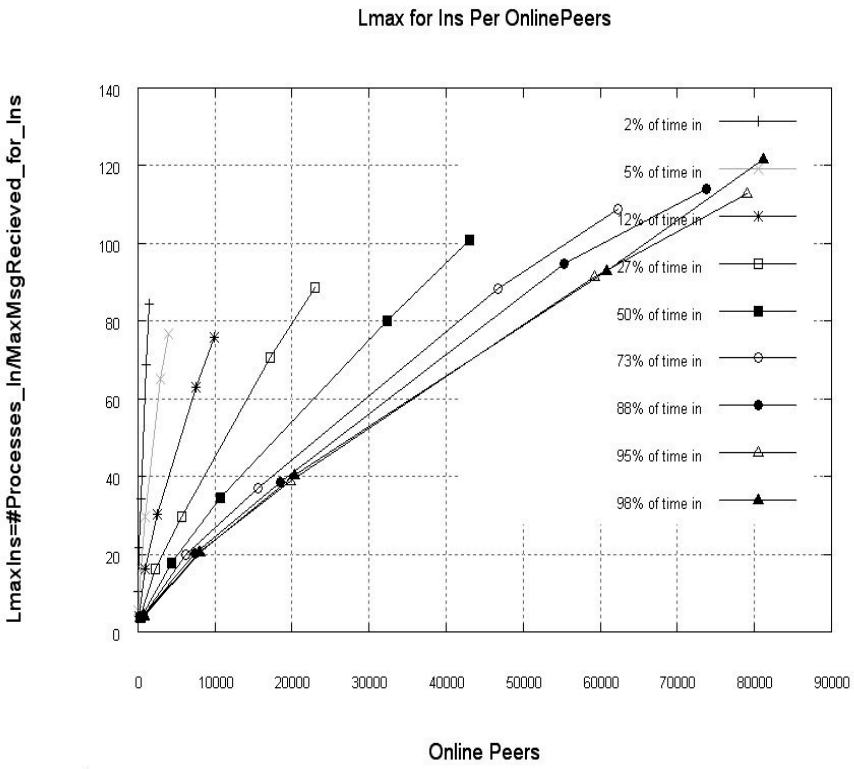
ενα links μαζί με τα εξερχόμενα links προκειμένου να προωθήσουμε το μήνυμα σε κάποιον peer που είναι υπεύθυνος για κάποιο φύλλο που είναι πιο κοντά στο φύλλο προορισμό, τότε το latency μειώνεται κατά ένα μήνυμα με αποτέλεσμα η κάθε διαδικασία εισόδου να γίνεται περίπου 13.3 % πιο γρήγορη. Αυτό είναι πολύ σημαντικό για την έρευνά μας καθώς με μια διαφορετική προσέγγιση του αρχικού πρωτοκόλου που περιγράψαμε βελτιώθηκε αρκετά η ταχύτητα ολοκλήρωσης κάθε διαδικασίας εισόδου.

4.3 Maximum Throughput

Ας δούμε τώρα ένα άλλο κριτήριο με το οποίο ασχοληθήκαμε και σύμφωνα με το [5] είναι από τα πιο σημαντικά για τη μελέτη ενός P2P συστήματος, το Maximum throughput. Το μέγεθος αυτό εκφράζει τον μέγιστο αριθμό διαδικασιών εισόδο ή/και εξόδο που μπορεί να εξυπηρετήσει το σύστημα στη μονάδα του χρόνου και θα αναφερόμαστε σε αυτό με τον όρο Λ_{max} . Όπως έχουμε ήδη ορίσει στην ενότητα 1.1.1, ένα process p στο process tree αντιπροσωπεύει μια διαδικασία εισόδου ή εξόδου. Το Λ_{max} έχει χρησιμοποιηθεί μόνο για διαδικασίες που αφορούν την αναζήτηση δεδομένων και είναι η πρώτη φορά που μελετάται για διαδικασίες εισόδου και εξόδου των peers στα P2P συστήματα. Σύμφωνα με το [5] το Λ_{max} ορίζεται ως:

$$\Lambda_{max} = \frac{1}{E[S] \cdot max_j \mu_j}$$

όπου $max_j \mu_j$ είναι ο αριθμός μηνυμάτων ανά διαδικασία εισόδου ή/και εξόδου που έλαβε ο πιο φορτωμένος peer του συστήματος και $E[S]$ είναι ο μέσος χρόνος που απαιτείται για τη λήψη του κάθε μηνύματος από έναν peer. Στα πειράματά μας υποθέσαμε ότι $E[S] = 1$ για χάρην απλότητας. Ο κάθε peer υποθέσαμε ότι συμμετείχε σε έναν αριθμό R διαδικασιών εισόδου και εξόδου κάθε φορά. Άρα ο αριθμός των μηνυμάτων που έλαβε ο πιο φορτωμένος peer υπολογίζεται



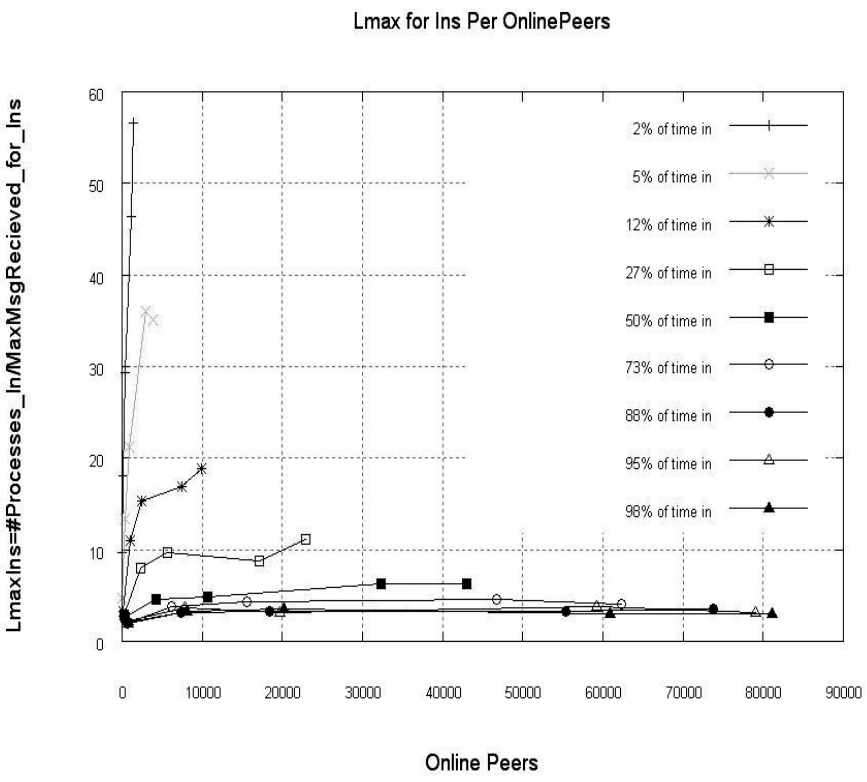
Σχήμα 4.12: Το maximum throughput για διαδικασίες εισόδου σε σχέση με τους online peers που βρίσκονται στο σύστημα.(volume balance)

λαμβάνοντας υπ' όψην τα μηνύματα από όλες τις διαδικασίες εισόδου και εξόδου που πραγματοποίησε. Άρα θα είναι:

$$\Lambda_{max} = \frac{1}{\frac{max_j \mu_j}{R}} = \frac{R}{max_j \mu_j}$$

Πρέπει ακόμα να εξηγήσουμε ότι αν είναι r ο αριθμός εισόδων και r ο αριθμός των εξόδων που πραγματοποιεί ο κάθε peer και N το πλήθος των peers που συμμετείχαν στο κάθε πείραμα τότε ισχύει $R = 2r \cdot N$. Στα πειράματα που πραγματοποιήσαμε υπολογίσαμε το Λ_{max} για κάθε διαφορετικό ποσοστό του χρόνου που είναι online οι peers στο σύστημα. Για κάθε ένα από αυτά τα ποσοστά υπολογίσαμε το Λ_{max} για k = 1.000, 10.000, 25.000, 75.000, 100.000 peers. Επίσης υπολογίσαμε το Λ_{max} για τις διαδικασίες εισόδου και για τις διαδικασίες εξόδου ξεχωριστά αλλά και το συνολικό Λ_{max} που αφορά είτε διαδικασίες εισόδου είτε διαδικασίες εξόδου. Έτσι έχουμε μια πιο ολοκληρωμένη εικόνα του Λ_{max} σε σχέση με τα ήδη των διαδικασιών για τις οποίες μελετήθηκε.

Στο σχήμα 4.12 και 4.13 βλέπουμε πως αλλάζει το Λ_{max} μόνο για διαδικασίες εισόδου σε σχέση με τους online peers που βρίσκονται στο σύστημα για volume balance και data balance αντίστοιχα. Αρχικά θα παρατηρήσουμε ότι όταν χρησιμοποιούμε volume balance το Λ_{max} αυξάνεται πάντα με πολύ ικανοποιητικό ρυθμό. Αυτό σημαίνει ότι καθώς αυξάνονται οι online peers αυξάνεται και η ικανότητα του συστήματος να εξυπηρετεί εισόδους νέων peers που επιθυμούν να γίνουν μέλη του συστήματος. Παρατηρούμε ότι όταν το ποσοστό του χρόνου παραμονής των peers που είναι online στο σύστημα είναι σχετικά μικρότερο από το ποσοστό του χρόνου που βρίσκονται εκτός, το Λ_{max} αυξάνεται με πολύ μεγαλύτερο ρυθμό από ό,τι όταν το ποσοστό του χρόνου που βρίσκονται online στο σύστημα γίνει μεγαλύτερο από 50 %. Επίσης παρατηρούμε ότι όταν το ποσοστό του χρόνου εντός του συστήματος γίνει πάνω από 50%, τότε το Λ_{max} για τις εισόδους δεν διαφέρει πάρα πολύ καθώς βρίσκονται σχεδόν πάνω στην ίδια γραμμή. Επίσης για ποσοστά από 2-50% υπάρχει σημαντική διαφορά του ρυθμού αύξησης του Λ_{max}. Το σημαντικό όμως είναι ότι όσο κι αν είναι το ποσοστό του χρόνου που βρίσκονται online οι peers στο σύστημα, το Λ_{max} για τις εισόδους πάντα αυξάνεται.



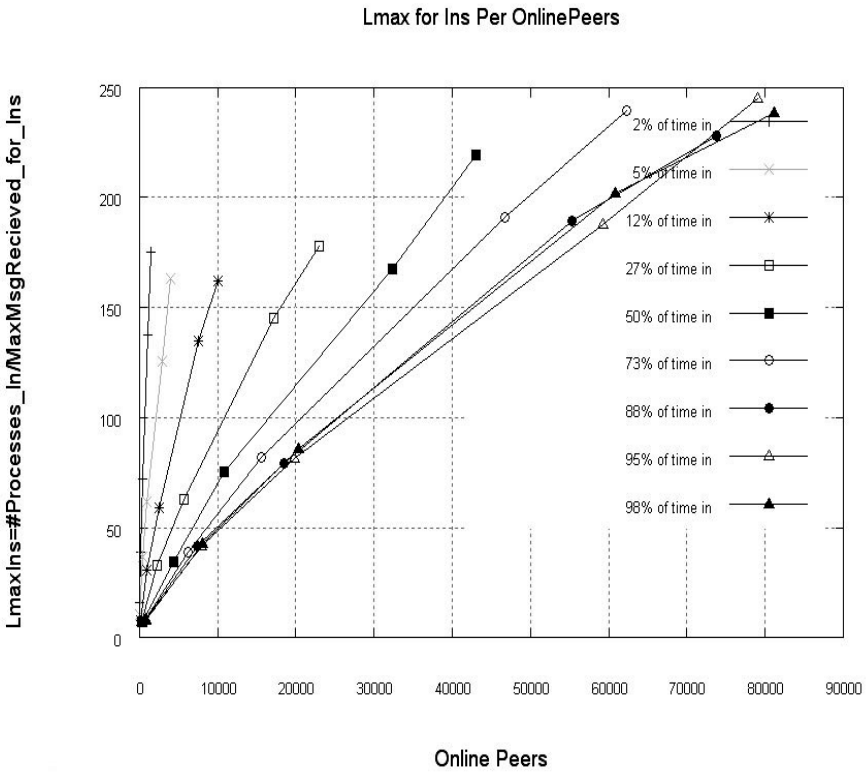
Σχήμα 4.13: Το maximum throughput για διαδικασίες εισόδου σε σχέση με τους online peers που βρίσκονται στο σύστημα.(data balance)

Δυστυχώς δεν μπορούμε να πούμε το ίδιο για το Λ_{max} όταν χρησιμοποιούμε data balance. Όπως φαίνεται και στο σχήμα 4.13 η διαφορά είναι τεράστια και είναι φανερό ότι το Λ_{max} δεν προσαρμόζεται με καθόλου ικανοποιητικό ρυθμό καθώς αυξάνονται οι online peers. Αυτό επαληθεύει κάτι που έχουμε ήδη αναφέρει σε προηγούμενη παράγραφο ότι όταν το trie είναι ισοζυγισμένο, τότε το σύστημα έχει καλύτερη απόδοση για κάθε μέγεθος που μετρήσαμε.

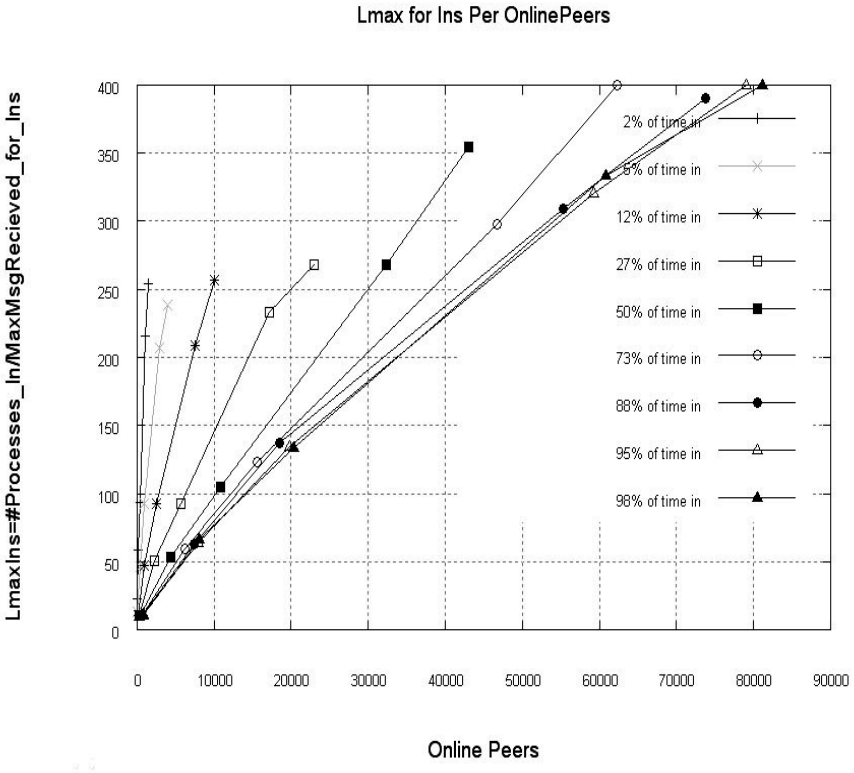
Στο σχήμα 4.14 παρατηρούμε το Λ_{max} για τις εισόδους των peers όταν η συνάρτηση exchange λαμβάνει μέρος στις διαδικασίες εισόδου με πιθανότητα 1/3, στο σχήμα 4.15 με πιθανότητα 1/9, στο σχήμα 4.16 παρατηρούμε το Λ_{max} για τις εισόδους όταν δεν χρησιμοποιείται καθόλου η συνάρτηση exchange και τέλος στο σχήμα 4.17 παρατηρούμε τη περίπτωση όπου δεν χρησιμοποιείται η συνάρτηση exchange ενώ συγχρόνως χρησιμοποιούμε ως routing table των πίνακα εισερχομένων και εξερχομένων links μαζί.

Όπως μπορούμε να παρατηρήσουμε στο σχήμα 4.14 το Λ_{max} για τις διαδικασίες εισόδου είναι περίπου δύο φορές καλύτερο από το αντίστοιχο Λ_{max} του σχήματος 4.12 όπου η συνάρτηση exchange χρησιμοποιείται πάντα. Παρατηρούμε στο σχήμα 4.15 όπου η συνάρτηση exchange χρησιμοποιείται με πιθανότητα 1/9 ότι το Λ_{max} για τις εισόδους γίνεται ακόμα καλύτερο από τις δύο προηγούμενες περιπτώσεις ενώ στο σχήμα 4.16 όπου δεν χρησιμοποιείται καθόλου η συνάρτηση exchange είναι φανερό ότι το Λ_{max} είναι σχεδόν τέσσερις φορές μεγαλύτερο από το Λ_{max} του αρχικού σχήματος 4.12. Τέλος στο σχήμα 4.17 όπου δεν χρησιμοποιείται ποτέ η exchange ενώ συγχρόνως χρησιμοποιούμε τον πίνακα των εισερχομένων και των εξερχομένων links ως routing table παρατηρούμε ότι το Λ_{max} για τις εισόδους γίνεται κατά ένα μικρό ποσοστό καλύτερο από αυτό του σχήματος 4.16. Οπότε είναι φανερό ότι η τελευταία περίπτωση διατηρεί ένα Λ_{max} που είναι πάνω από 4 φορές καλύτερο από το Λ_{max} της βασικής μορφής του πρωτοκόλου που περιγράψαμε.

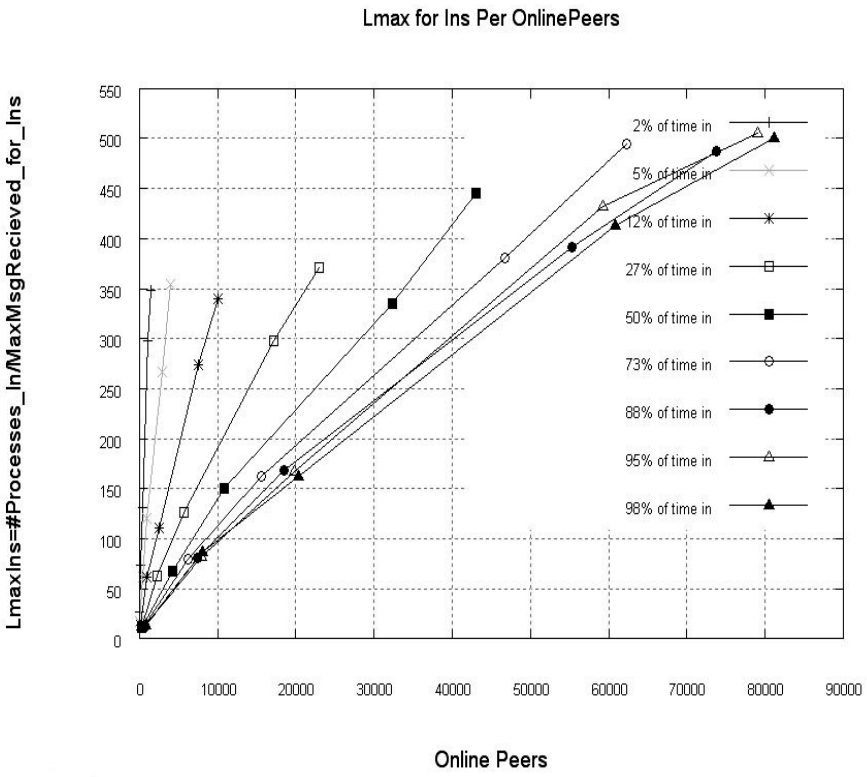
Στα σχήματα 4.18 και 4.19 βλέπουμε το Λ_{max} για τις εξόδους των peers με volume balance και data balance αντίστοιχα. Το Λ_{max} παρατηρούμε ότι δεν αυξάνεται με κάποιο ρυθμό αλλά τείνει να παραμείνει σταθερό σε μια τιμή. Επίσης παρατηρούμε ότι όσο πιο μεγάλο είναι το ποσοστό του χρόνου που βρίσκονται online οι peers τόσο μικρότερο είναι το Λ_{max} Επίσης αν συγκρίνουμε τα δύο σχήματα



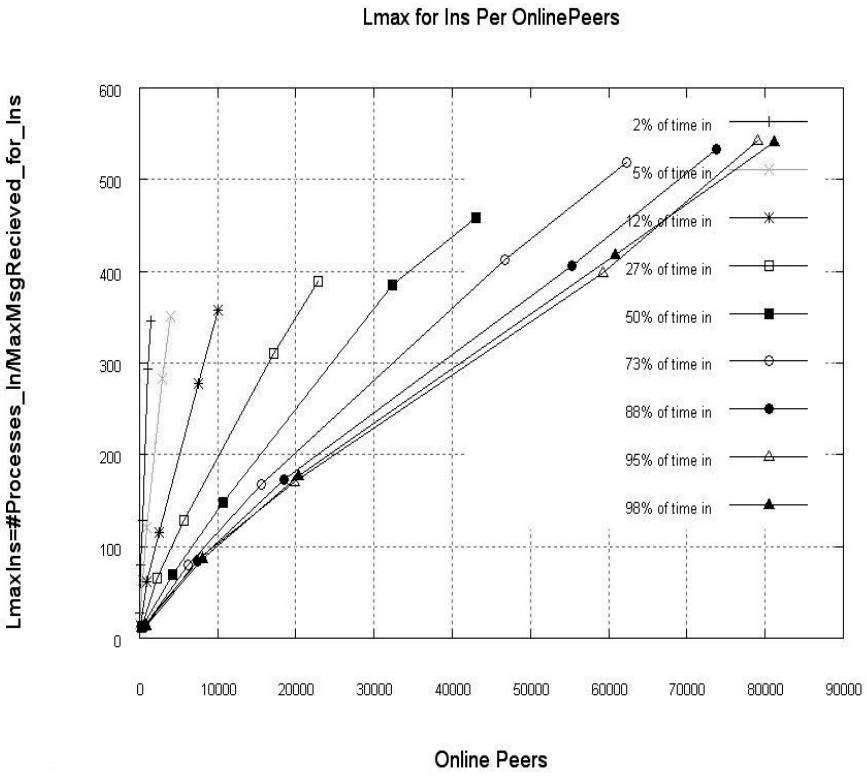
Σχήμα 4.14: Το Maximum throughput για διαδικασίες εισόδου σε σχέση με τους online peers που βρίσκονται στο σύστημα όταν η συνάρτηση exchange χρησιμοποιείται με πιθανότητα 1/3.(volume balance)



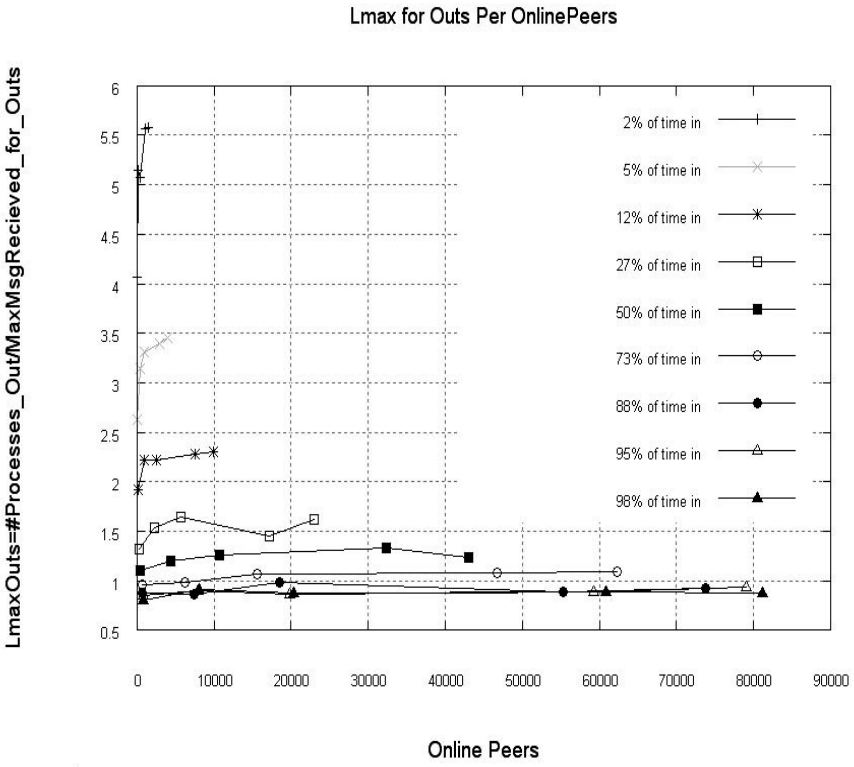
Σχήμα 4.15: Το Maximum throughput για διαδικασίες εισόδου σε σχέση με τους online peers που βρίσκονται στο σύστημα όταν η συνάρτηση exchange χρησιμοποιείται με πιθανότητα 1/9.(volume balance)



Σχήμα 4.16: Το Maximum throughput για διαδικασίες εισόδου σε σχέση με τους online peers που βρίσκονται στο σύστημα όταν η συνάρτηση exchange δεν χρησιμοποιείται ποτέ.(volume balance)



Σχήμα 4.17: Το Maximum throughput για διαδικασίες εισόδου σε σχέση με τους online peers που βρίσκονται στο σύστημα όταν η συνάρτηση exchange δεν χρησιμοποιείται ποτέ ενώ συγχρόνως χρησιμοποιούμε ως routing table των πίνακα εισερχομένων και εξερχομένων links μαζί.(volume balance)



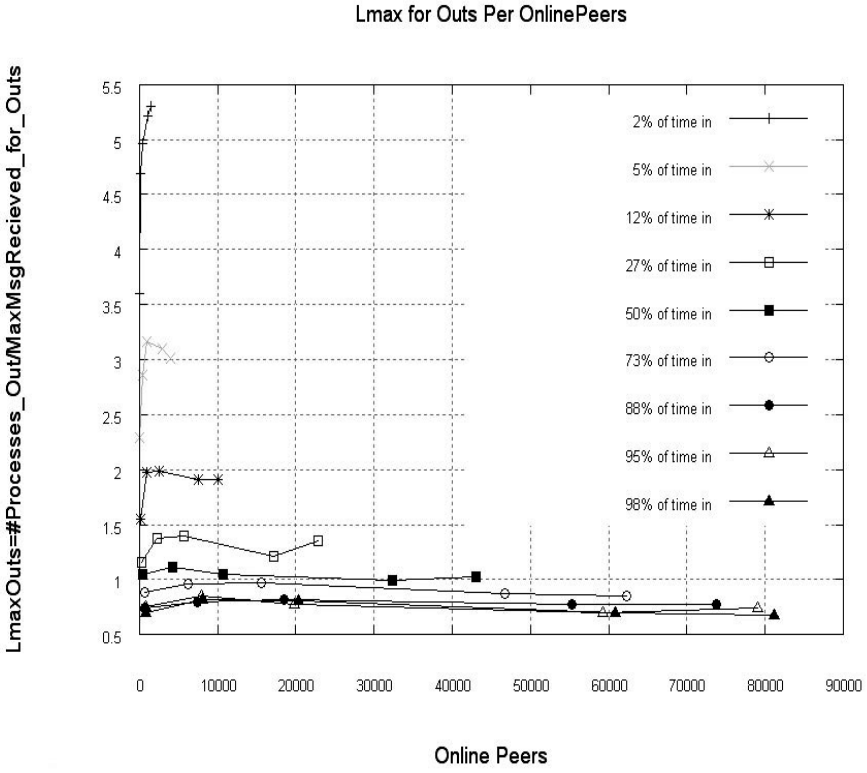
Σχήμα 4.18: Το Maximum throughput για τις διαδικασίες εξόδου σε σχέση με τους online peers που βρίσκονται στο σύστημα.(volume balance)

θα παρατηρήσουμε ότι το Λ_{max} δεν είναι διαφορετικό είτε χρησιμοποιούμε volume balance είτε χρησιμοποιούμε data balance. Πρέπει να θυμίσουμε ότι το Λ_{max} για τις εξόδους των peers υπολογίζεται λαμβάνοντας υπ' όψην όλα τα μηνύματα που στέλνονται μεταξύ των peers όταν κάποιος peer επιθυμεί να βγει offline. Το πλήθος των μηνυμάτων αυτών εξαρτάται από το πλήθος των φύλλων του trie και από το πλήθος των online peers που είναι υπεύθυνοι γι' αυτά τα φύλλα.

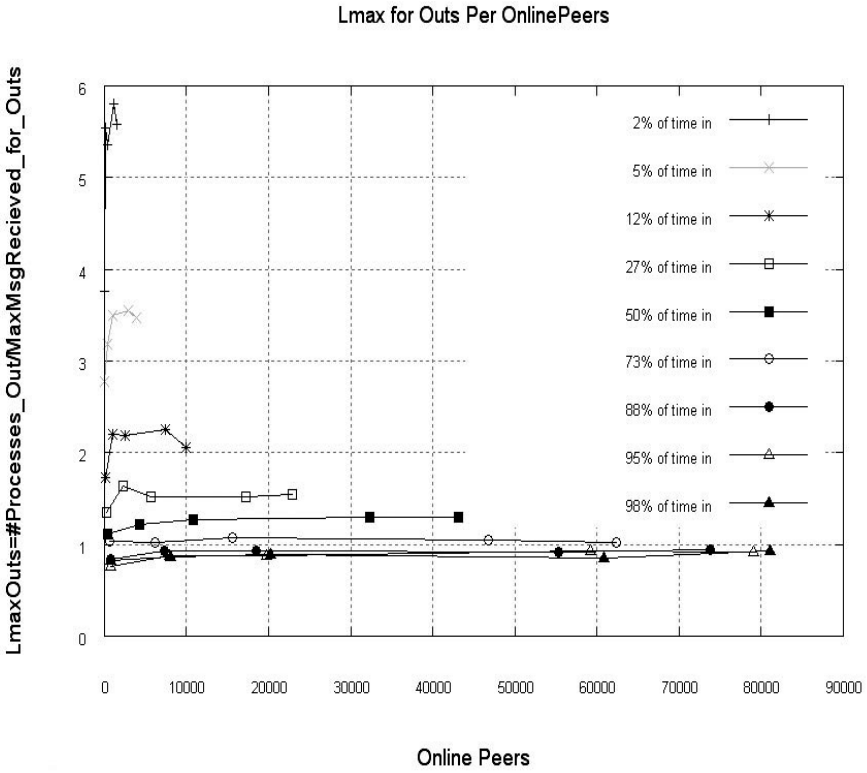
Στο σχήμα 4.20 παρατηρούμε το Λ_{max} για τις εξόδους των peers όταν η συνάρτηση exchange λαμβάνει μέρος στις διαδικασίες εισόδου με πιθανότητα 1/3, στο σχήμα 4.21 με πιθανότητα 1/9, στο σχήμα 4.22 παρατηρούμε το Λ_{max} για τις εξόδους όταν δεν χρησιμοποιείται καθόλου η συνάρτηση exchange και τέλος στο σχήμα 4.23 παρατηρούμε τη περίπτωση όπου δεν χρησιμοποιείται η συνάρτηση exchange ενώ συγχρόνως χρησιμοποιούμε ως routing table των πίνακα εισερχομένων και εξερχομένων links μαζί. Στα σχήματα 4.20 ως 4.23 μπορούμε να παρατηρήσουμε ότι το Λ_{max} των διαδικασιών εξόδου δεν επηρεάζεται σχεδόν καθόλου με τη χρήση ή όχι τις συνάρτησης exchange.

Στη συνέχεια θα παρουσιάσουμε τα αποτελέσματα του Λ_{max} που αφορά το σύνολο των διαδικασιών που πραγματοποιούνται είτε είναι διαδικασίες εισόδου είτε είναι διαδικασίες εξόδου. Στα σχήματα 4.24 ως 4.29 παρατηρούμε το συνολικό Λ_{max} που αφορά όλες τις διαδικασίες για κάθε μια από τις περιπτώσεις που περιγράφονται στο κάθε σχήμα. Παρατηρούμε ότι όπως ήταν αναμενόμενο το συνολικό Λ_{max} επηρεάζεται αρνητικά από τις διαδικασίες εξόδου των peers και έχει ακριβώς την ίδια μορφή με το Λ_{max} που αφορά τις εξόδους των peers με τη διαφορά ότι είναι λίγο καλύτερο καθώς έχει επηρεαστεί από την καλή απόδοση του συστήματος για τις εισόδους των peers.

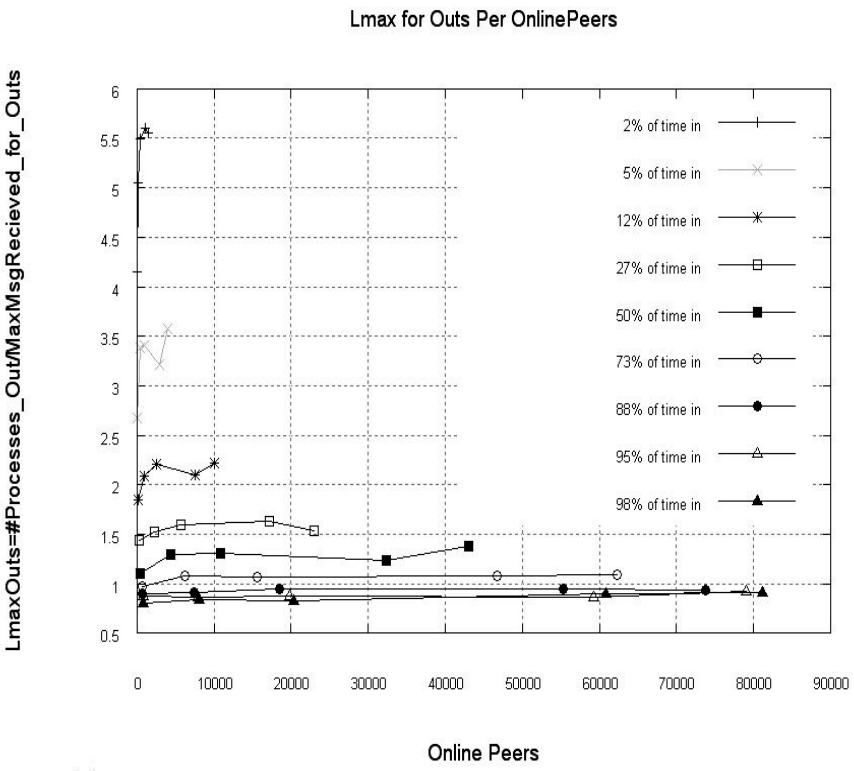
Σε αυτό το σημείο θα ήταν χρήσιμο να καταλήξουμε σε κάποια βασικά συμπεράσματα που προέκυψαν από τα αποτελέσματα του Λ_{max}. Αρχικά πρέπει να πούμε ότι το Λ_{max} που αφορά μόνο τις εισόδους νέων peers στο σύστημα αυξάνεται πάντα κάτι που είναι πολύ ενθαρυντικό για το πρωτόκολο ενώ συγχρόνως δεν συμβαίνει το ίδιο για το Λ_{max} που αφορά τις εξόδους των peers. Θα παρατηρήσουμε επίσης ότι η συνάρτηση exchange λειτουργεί αρνητικά για το Λ_{max} των διαδικασιών εισόδου καθώς τα αποτελέσματα μας δείχνουν ότι όσο πιο σπάνια χρησιμοποιούμε τη συνάρτηση exchange τόσο καλύτερο Λ_{max} προκύπτει για τις



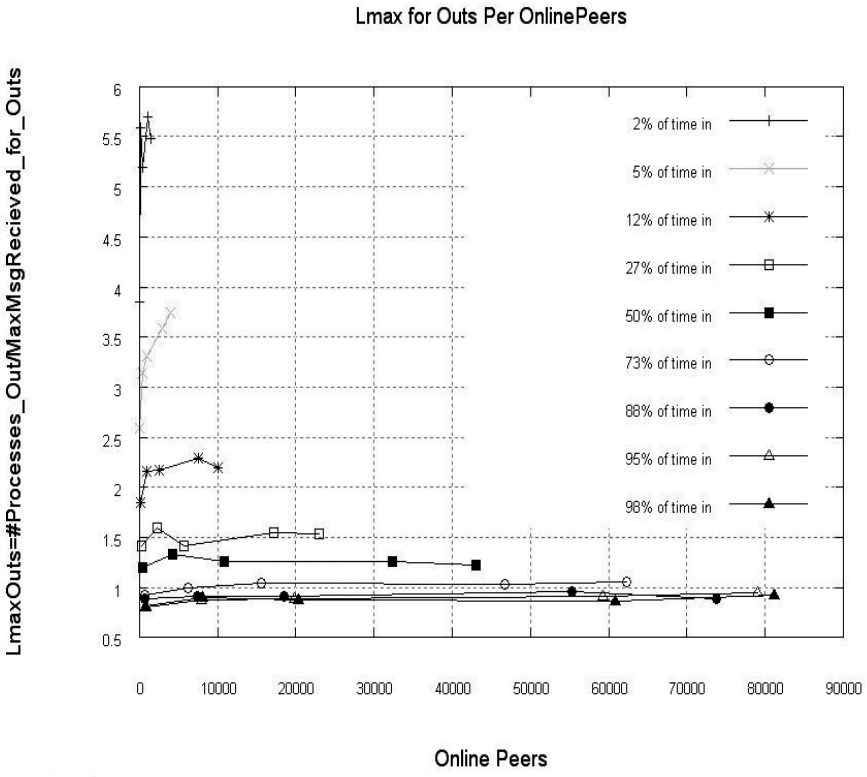
Σχήμα 4.19: Το Maximum throughput για τις διαδικασίες εξόδου σε σχέση με τους online peers που βρίσκονταν στο σύστημα.(data balance)



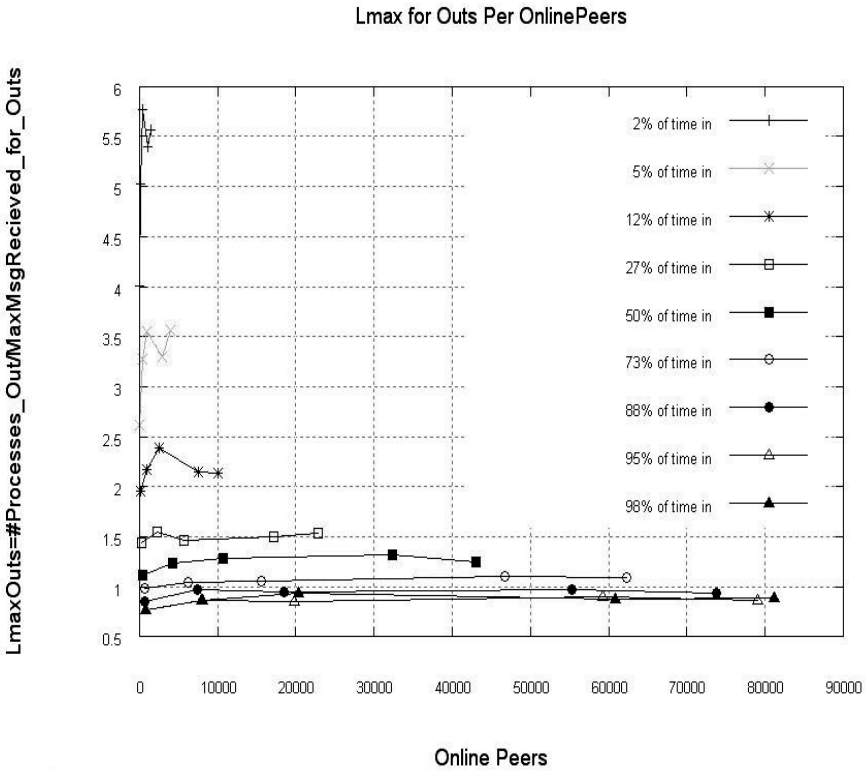
Σχήμα 4.20: Το Maximum throughput για διαδικασίες εξόδου σε σχέση με τους online peers που βρίσκονται στο σύστημα όταν η συνάρτηση exchange χρησιμοποιείται με πιθανότητα 1/3.(volume balance)



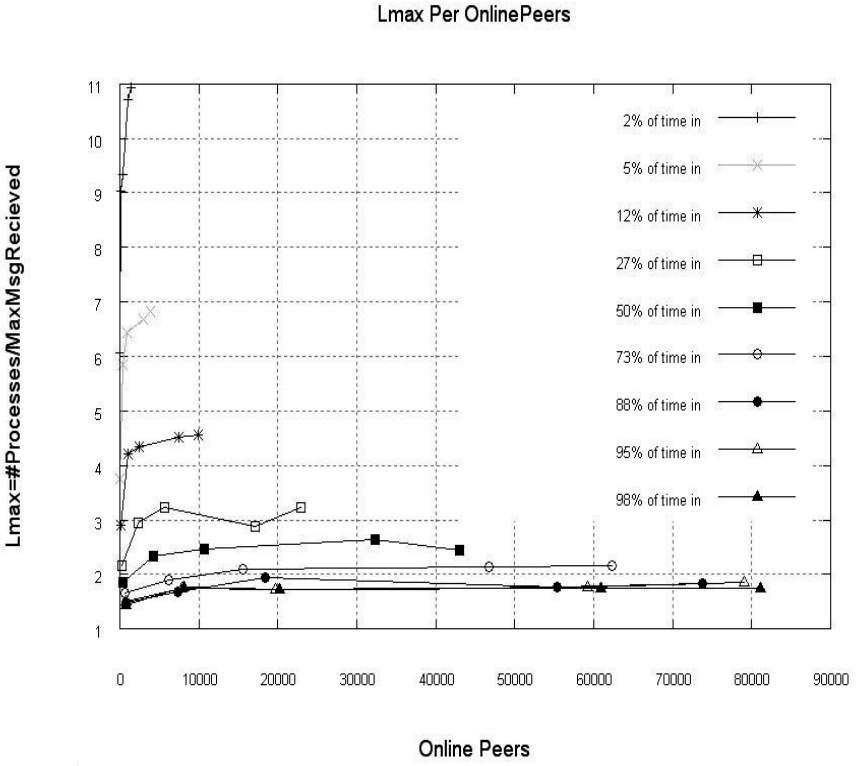
Σχήμα 4.21: Το Maximum throughput για διαδικασίες εξόδου σε σχέση με τους online peers που βρίσκονται στο σύστημα όταν η συνάρτηση exchange χρησιμοποιείται με πιθανότητα 1/9.(volume balance)



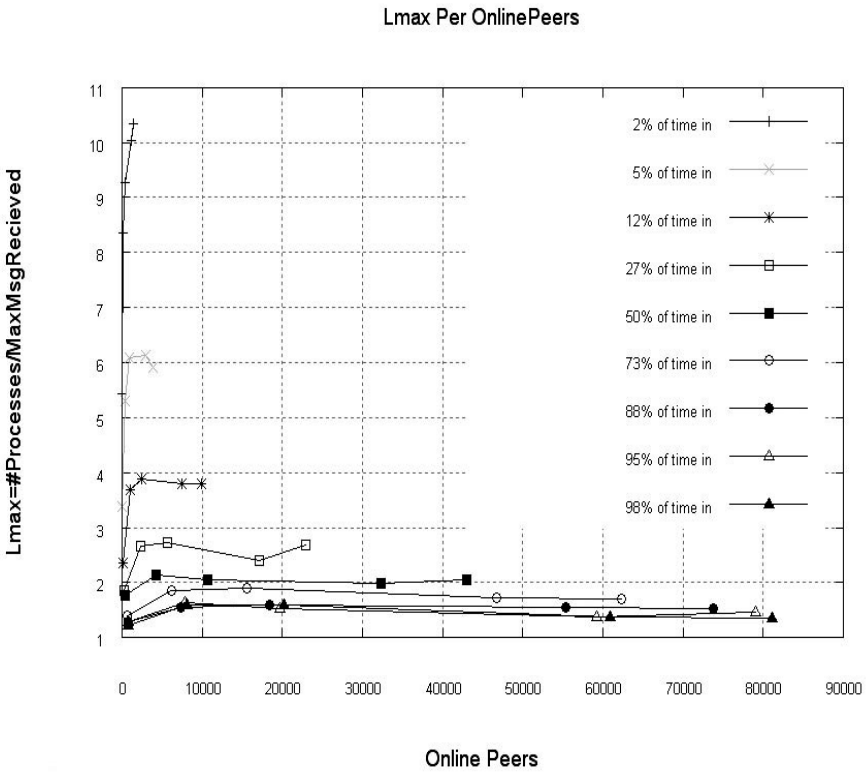
Σχήμα 4.22: Το Maximum throughput για διαδικασίες εξόδου σε σχέση με τους online peers που βρίσκονται στο σύστημα όταν η συνάρτηση exchange δεν χρησιμοποιείται ποτέ.(volume balance)



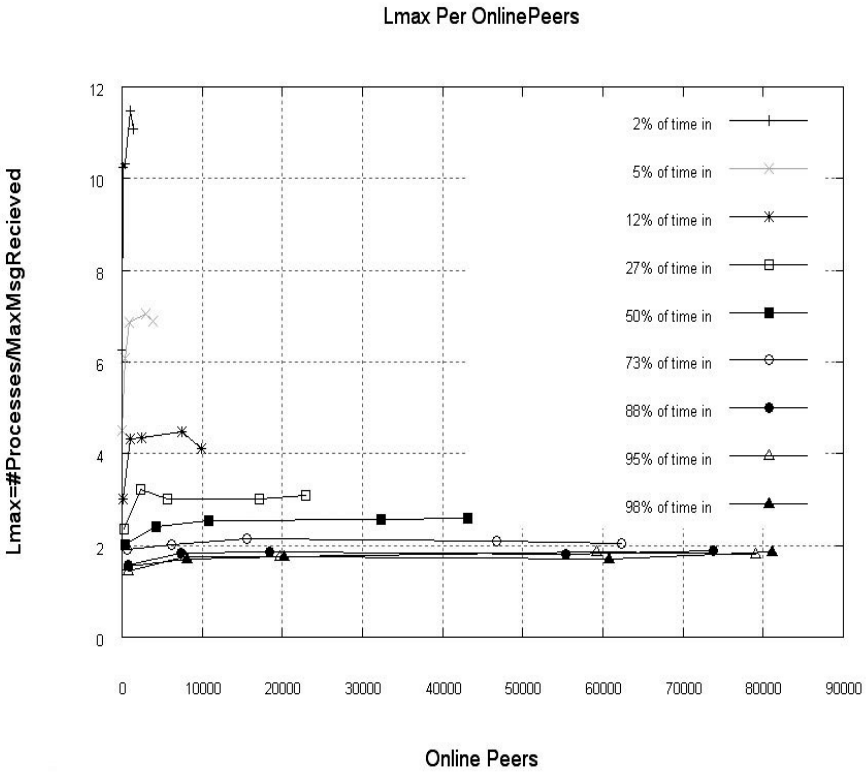
Σχήμα 4.23: Το Maximum throughput για διαδικασίες εξόδου σε σχέση με τους online peers που βρίσκονται στο σύστημα όταν η συνάρτηση exchange δεν χρησιμοποιείται ποτέ ενώ συγχρόνως χρησιμοποιούμε ως routing table των πίνακα εισερχομένων και εξερχομένων links μαζί.(volume balance)



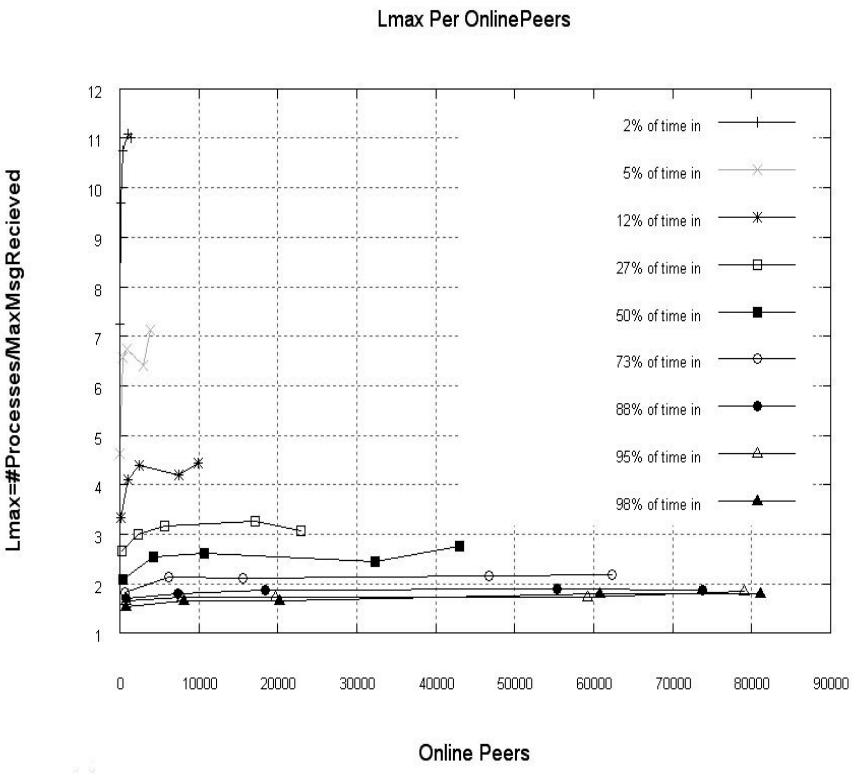
Σχήμα 4.24: Το συνολικό Maximum throughput για τις διαδικασίες εισόδου και εξόδου μαζί σε σχέση με τους online peers που βρίσκονται στο σύστημα.(volume balance)



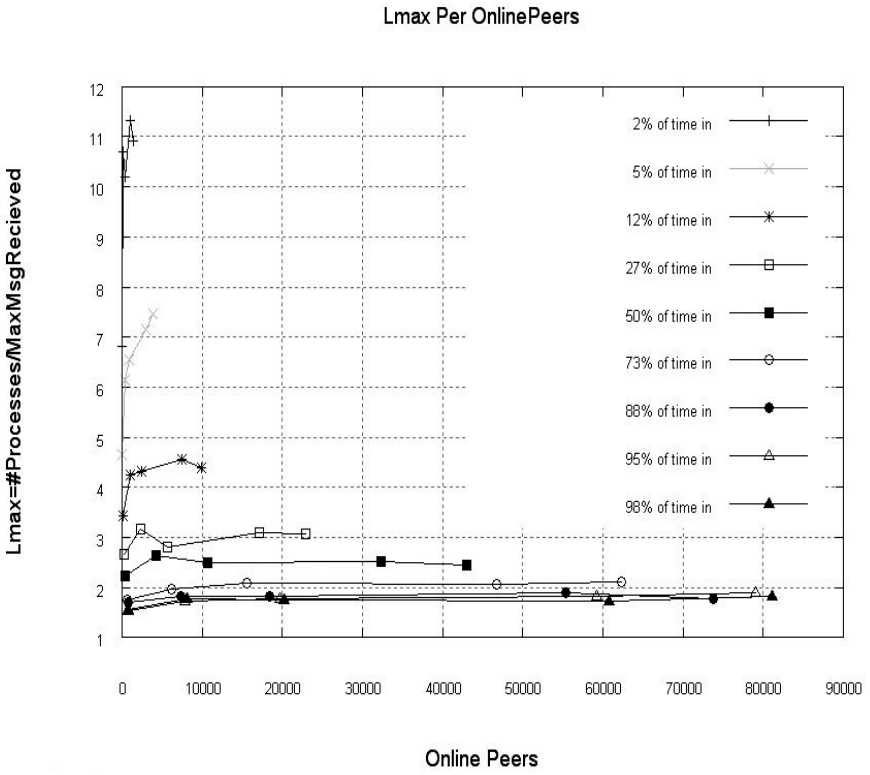
Σχήμα 4.25: Το συνολικό Maximum throughput για τις διαδικασίες εισόδου και εξόδου μαζί σε σχέση με τους online peers που βρίσκονται στο σύστημα.(data balance)



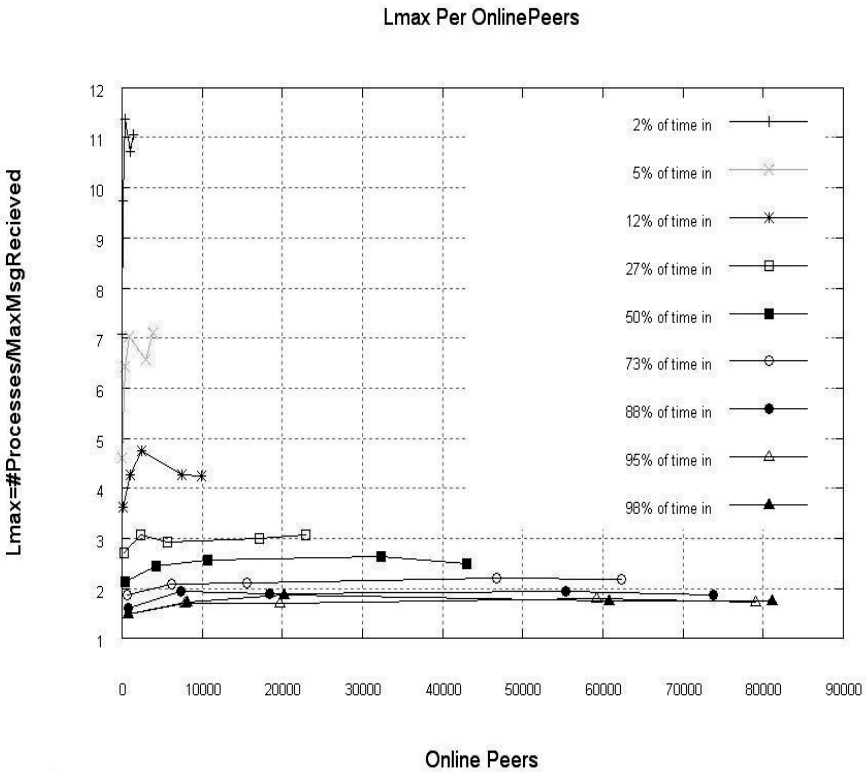
Σχήμα 4.26: Το συνολικό Maximum throughput για τις διαδικασίες εισόδου και εξόδου μαζί σε σχέση με τους online peers που βρίσκονται στο σύστημα όταν η συνάρτηση exchange χρησιμοποιείται με πιθανότητα 1/3.(volume balance)



Σχήμα 4.27: Το συνολικό Maximum throughput για τις διαδικασίες εισόδου και εξόδου μαζί σε σχέση με τους online peers που βρίσκονται στο σύστημα όταν η συνάρτηση exchange χρησιμοποιείται με πιθανότητα 1/9.(volume balance)



Σχήμα 4.28: Το συνολικό Maximum throughput για τις διαδικασίες εισόδου και εξόδου μαζί σε σχέση με τους online peers που βρίσκονται στο σύστημα όταν η συνάρτηση exchange δεν χρησιμοποιείται ποτέ.(volume balance)



Σχήμα 4.29: Το συνολικό Maximum throughput για τις διαδικασίες εισόδου και εξόδου μαζί σε σχέση με τους online peers που βρίσκονται στο σύστημα όταν η συνάρτηση exchange δεν χρησιμοποιείται ποτέ ενώ συγχρόνως χρησιμοποιούμε ως routing table των πίνακα εισερχομένων και εξερχομένων links μαζί.(volume balance)

διαδικασίες εισόδου. Αυτό ήταν αναμενόμενο αφού η συνάρτηση exchange λαμβάνει μέρος κατά τη διάρκεια μιας διαδικασίας εισόδου και αναγκάζει τους peers που συναντιόνται να στείλουν μηνύματα σε άλλους peers όπως έχουμε περιγράψει αναλυτικά στο κεφάλαιο 3.1.3 και 3.2.2. Αυτά τα μηνύματα είναι η αιτία να λάβουν περισσότερα μηνύματα οι peers και άρα τα μηνύματα που έλαβε ο πιο φορτωμένος peer να είναι πολύ περισσσότερα. Παρατηρήσαμε επίσης ότι το Λ_{max} που αφορά μόνο τις εξόδους των peers δεν επηρεάζεται από τη λειτουργία ή όχι της συνάρτησης exchange.

Το συνολικό Λ_{max} που αφορά όλες τις διαδικασίες εισόδου και εξόδου μαζί, παρατηρούμε ότι δεν αυξάνεται και δεν βελτιώνεται καθώς αυξάνονται οι online peers για κάθε μια από τις περιπτώσεις που μελετήσαμε. Παρατηρούμε μονάχα ότι το συνολικό Λ_{max} είναι μερικές μονάδες καλύτερο από το Λ_{max} που αφορά τις εξόδους. Αυτό είναι λογικό αφού παρατηρήσαμε ότι το Λ_{max} για τις εισόδους πάντα αυξάνεται και είναι αρκετά μεγαλύτερο απο το αντίστοιχο Λ_{max} των εξόδων οπότε το συνολικό Λ_{max} είναι αναμενόμενο να είναι ο συνδιασμός των δύο παραπάνω. Επομένως πρακτικά βλέπουμε ότι οι διαδικασίες εισόδου βελτιώνουν το συνολικό Λ_{max} ενώ οι διαδικασίες εξόδου το μειώνουν αφού έχουν σαν αποτέλεσμα ένα πολύ χειρότερο Λ_{max}. Το απογοητευτικό Λ_{max} των εξόδων μας προβλημάτισε αρχικά και μας ανάγκασε να αναρωτηθούμε γιατί συμβαίνει κάτι τέτοιο. Αν κοιτάξουμε το σχήμα 4.30 και το σχήμα 4.36 όπου παρατηρούμε το συνολικό αριθμό μηνυμάτων ανά διαδικασία εισόδου και εξόδου αντίστοιχα θα δούμε ότι σε μια διαδιακασία εξόδου στέλνονται πολλά περισσότερα μηνύματα από ό,τι σε μια διαδικασία εισόδου. Αυτό δεν θα ήταν κάτι αρνητικό παρά μόνο αν μελετήσουμε προσεκτικά τον μέσο όρο μηνυμάτων που έλαβε ο κάθε peer ανά διαδικασία εξόδου καθώς και τον μέγιστο αριθμό μηνυμάτων που έλαβε ο πιο φορτωμένος peer. Παρατηρούμε στο σχήμα 4.36 ότι ο μέσος αριθμός μηνυμάτων ανά διαδικασία εξόδου για ένα πλήθος 75.000 peers και ποσοστό χρόνου εντός του συστήματος 98% ότι είναι περίπου 200 μηνύματα ανά διαδικασία εξόδου. Αυτό σημαίνει ότι όταν κάποιος peer βγεί εκτός συστήματος θα λάβουν ένα μήνυμα περίπου 200 διαφορετικοί peers. Σε αυτό το σημείο σκεφτήκαμε να αναζητήσουμε τη σχέση που έχουν ο μέσος αριθμός μηνυμάτων που στάλθηκαν ανά διαδικασία εξόδου με το πλήθος των μηνυμάτων που έλαβε ο πιο φορτωμένος peer. Αυτό ήταν απαιραίτητο προκειμένου να αποφανθούμε γιατί το Λ_{max} για τις αντίστοιχες εξόδους προκύπτει να είναι ένας αριθμός μικρότερος της μονάδας όπως φαίνεται και στο σχήμα 4.18. Αν ο μέσος αριθμός μηνυμάτων που στέλνονται ανά διαδικασία εξόδου είναι περίπου 200 τότε για $N = 75.000$ το σύστημα εξυπηρετεί κατά μέσο όρο $75.000/200 =\simeq 375$ διαδικασίες εξόδου στη μονάδα του χρόνου. Στο σχήμα 4.18 μπορούμε να δούμε ότι ο μέγιστος αριθμός διαδικασιών εξόδου που μπορεί και εξυπηρετεί το σύστημα στη μονάδα του χρόνου σύμφωνα με τα πειράματα που έγιναν, είναι περίπου 0.9. Παρατηρούμε δηλαδή ότι ο μέγιστος αυτός ρυθμός που προέκυψε από τα πειράματά μας είναι περίπου 416% μικρότερος από τον μέσο όρο που υπολογίσαμε. Δηλαδή ενώ ο μέσος αριθμός μηνυμάτων που λαμβάνει ο κάθε peer ανά διαδιακασία εξόδου είναι περίπου 200, τα αποτελέσματα δείχνουν ότι το πλήθος των μηνυμάτων που έλαβε ο πιο φορτωμένος peer είναι ένας αριθμός πολύ μεγαλύτερος του 200. (περίπου $75.000/0.9 \simeq 83.333$!!!) Αυτό το συμπέρασμα μας έκανε να αναζητήσουμε βαθύτερα την αιτία στην οποία οφείλεται το πολύ μικρό Λ_{max} για τις εξόδους και έτσι έπρεπε να σκεφτούμε τι είδους μηνύματα στέλνονται σε κάθε διαδικασία εξόδου. Όπως έχουμε ήδη περιγράψει στο κεφάλαιο 3.3 όταν κάποιος peer επιθυμεί να βγεί εκτός του συστήματος θα πρέπει για κάθε φύλλο που είναι υπεύθυνος να ελέγξει τα εισερχόμενα και εξερχόμενα links που αυτό διατηρεί. Για κάθε φύλλο με το οποίο διατηρεί ένα ή περισσότερα εισερχόμενα ή εξερχόμενα links στέλνεται ένα μήνυμα στον υπεύθυνο peer εκείνου του φύλλου. Για αυτό το λόγο θεωρήσαμε χρήσιμο να υπολογίσουμε την μέση αλλά και τη μέγιστη τιμή του πλήθους των εισερχομένων και των εξερχομένων links που διατηρεί το κάθε φύλλο του trie. Οι εκτελέσεις έγιναν για 75.000 peers και 98% ποσοστό χρόνου εντός του συστήματος και τα αποτελέσματα ήταν τα εξής:

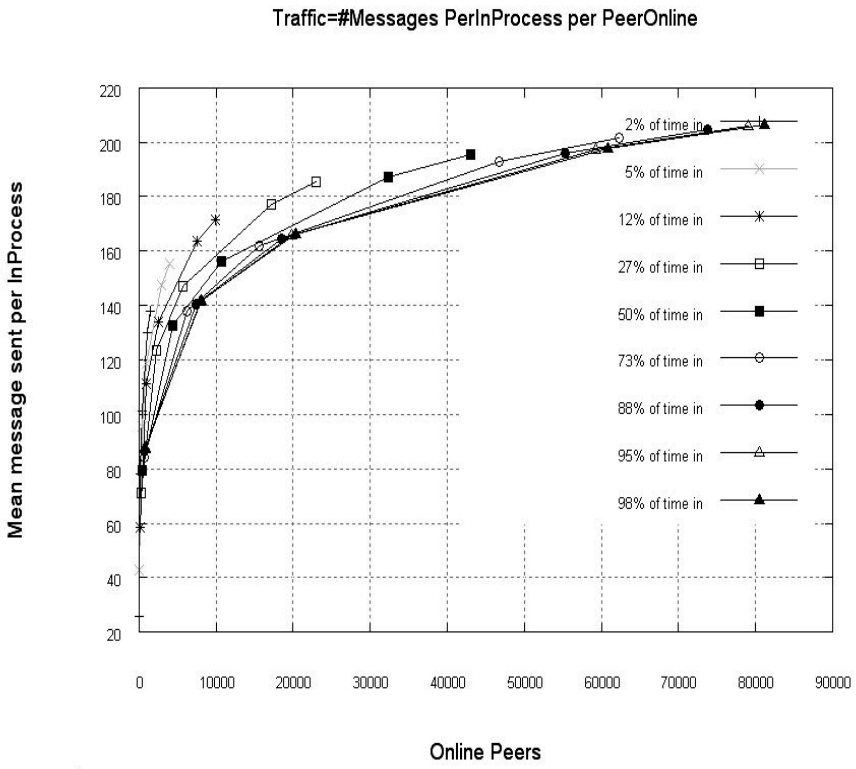
- Με χρήση της συνάρτησης exchange:
  Συνολικά φύλλα: 206393
  Σύνολο εισερχομένων links : 3692140
  Μέσος όρος εισερχομένων links ανά φύλλο: 17.8889

Μέγιστος αριθμός εισερχομένων links : 157
Σύνολο εξερχομένων links : 3692140
Μέσος όρος εξερχομένων links ανά φύλλο: 17.8889
Μέγιστος αριθμός εξερχομένων links : 22

- Χωρίς τη λειτουργία της συνάρτησης exchange:
  Συνολικά φύλλα: 206873
  Σύνολο εισερχομένων links : 3701483
  Μέσος όρος εισερχομένων links ανά φύλλο: 17.8925
  Μέγιστος αριθμός εισερχομένων links : 287
  Σύνολο εξερχομένων links : 3701483
  Μέσος όρος εξερχομένων links ανά φύλλο: 17.8925
  Μέγιστος αριθμός εξερχομένων links : 22

Παρατηρώντας τα παραπάνω αποτελέσματα καταλήγουμε στη βαθύτερη αιτία που το σύστημα δεν έχει ένα καλό Λ_{max} για τις διαδικασίες εξόδου. Ενώ ο μέσος όρος των εισερχομένων links ανά φύλλο είναι περίπου 17,9 και ο maximum αριθμός ήταν 22 δεν συμβαίνει το ίδιο για τα εισερχόμενα links παρόλο που και αυτά έχουν τον ίδιο μέσο όρο. Ενώ η μέση τιμή των εισερχομένων links που μετρήσαμε ήταν 17,9, η μέγιστη τιμή που μετρήθηκε ήταν 157 και 287 για τις δύο παραπάνω περιπτώσεις. Παρατηρούμε ότι όταν λειτουργεί η συνάρτηση exchange ο maximum αριθμός εισερχομένων links είναι αρκετά μικρότερος από ό,τι στη περίπτωση όπου δεν χρησιμοποιείται ενώ συγχρόνως τα μηνύματα που έλαβε ο πιο φορτωμένος peer δεν γίνονται λιγότερα. Επομένως ενώ παρατηρούμε μεγάλη διαφορά του μέγιστου αριθμού των εισερχομένων links στις δύο αυτές περιπτώσεις δεν παρατηρούμε κάποια αντίστοιχη διαφορά στα σχήματα 4.18 και 4.22 όπου φαίνεται το Λ_{max} για τις διαδικασίες εξόδου των δύο περιπτώσεων. Επίσης το Λ_{max} για τις διαδικασίες εξόδου είναι ίδιο είτε χρησιμοποιείται η συνάρτηση exchange είτε όχι. Τα δύο παραπάνω σε συνδιασμό με τα σχήματα 4.36 και 4.40 που είναι τα αντίστοιχα σχήματα για το πλήθος των μηνυμάτων που στέλνονται ανά διαδικασία εξόδου μας οδηγούν στο εξής συμπέρασμα: Ο μέγιστος αριθμός μηνυμάτων που στέλνονται ανά διαδικασία εξόδου είναι πολύ μεγάλος είτε χρησιμοποιείται η συνάρτηση exchange είτε όχι αφού σε κάθε περίπτωση συσσωρεύονται πάρα πολλά εισερχόμενα links σε κάποια φύλλα. Δηλαδή η αιτία τελικά για το απογοητευτικό Λ_{max} των εξόδων θα μπορούσαμε να πούμε ότι οφείλεται στο ότι καθώς λειτουργεί το σύστημα με το πρωτόκολο που περιγράψαμε, συσσωρεύονται πολλά εισερχόμενα links προς κάποια φύλλα με αποτέλεσμα όταν βγαίνουν διάφοροι peers offline ένα μεγάλο ποσοστό από αυτούς να στέλνει μηνύματα προς τον ίδιο υπεύθυνο peer εκείνων των φύλλων που διατηρούν πολλά εισερχόμενα links.

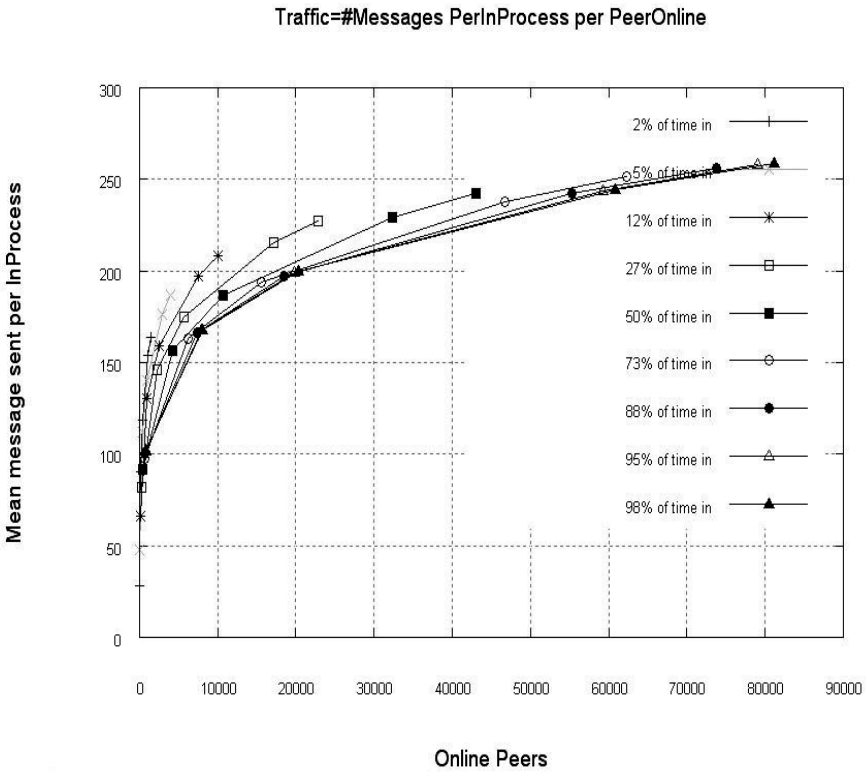
Οι παραπάνω παρατηρήσεις μας δείχουν τελικά ότι όπως κι αν διαμορφωθεί ο maximum αριθμός των εισερχομένων links των φύλλων, πάντα θα διαφέρουν πάρα πολύ από τον μέσο όρο που θα έπρεπε να έχει το κάθε φύλλο. Επομένως ένας τρόπος για να γίνει μια πιο ομοιόμορφη κατανομή των μηνυμάτων που στέλνονται όταν κάποιος peer επιθυμεί να βγει εκτός συστήματος ίσως να ήταν η ανταλαγή φύλλων μεταξύ των peers οι peers μεταξύ τους ανά τακτά χρονικά διαστήματα με τυχαίο τρόπο. Έτσι όταν διάφοροι peers βγαίνουν εκτός του συστήματος δεν θα δέχονταν όλα τα μηνύματα ο ίδιος peer εξ΄ αιτίας κάποιων φύλλων που θα τυχαίνει να διατηρούν πολλά εισερχόμενα links. Αυτό πάλι θα είχε σαν συνέπεια να αυξηθούν τα συνολικά μηνύματα που έλαβαν όλοι οι peers κάτι που δεν γνωρίζουμε πως θα επηρεάσει τα μηνύματα που έλαβε ο πιο φορτωμένος peer. Μια άλλη πιθανή λύση ίσως να ήταν η διάσπαση των φύλλων που ξεπερνούν ένα όριο εξερχομένων links. Με τη συνεχή διάσπαση των φύλλων θα δημιουργούνται νέα φύλλα τα οποία θα κληρονομούν τα εισερχόμενα links του αρχικού οπότε το καθένα θα έχει σίγουρα λιγότερα εισερχόμενα links. Στη συνέχεια θα μπορούσε ο υπεύθυνος peer να αναθέτει τα φύλλα που έχουν δημιουργηθεί σε άλλους peers και έτσι να μην υπάρχει ένας peer που να είναι υπεύθυνος για κάποιο φύλλο που να έχει πάρα πολλά εισερχόμενα links.



Σχήμα 4.30: Το Traffic για τις διαδικασίες εισόδου σε σχέση με τους online peers που βρίσκονται στο σύστημα.(volume balance)

4.4 Traffic

Τέλος θα δούμε τα αποτελέσματα για το traffic των διαδικασιών εισόδου ή/και εξόδου. Το μέγεθος αυτό μας δείχνει πόσα μηνύματα απαιτούνται συνολικά για να ολοκληρωθεί μια διαδικασία εισόδου ή/και εξόδου καθώς μεταβάλεται ο αριθμός των online peers στο σύστημα. Αρχικά θα υπολογίσουμε το traffic που αφορά μόνο τις εισόδους νέων peers.

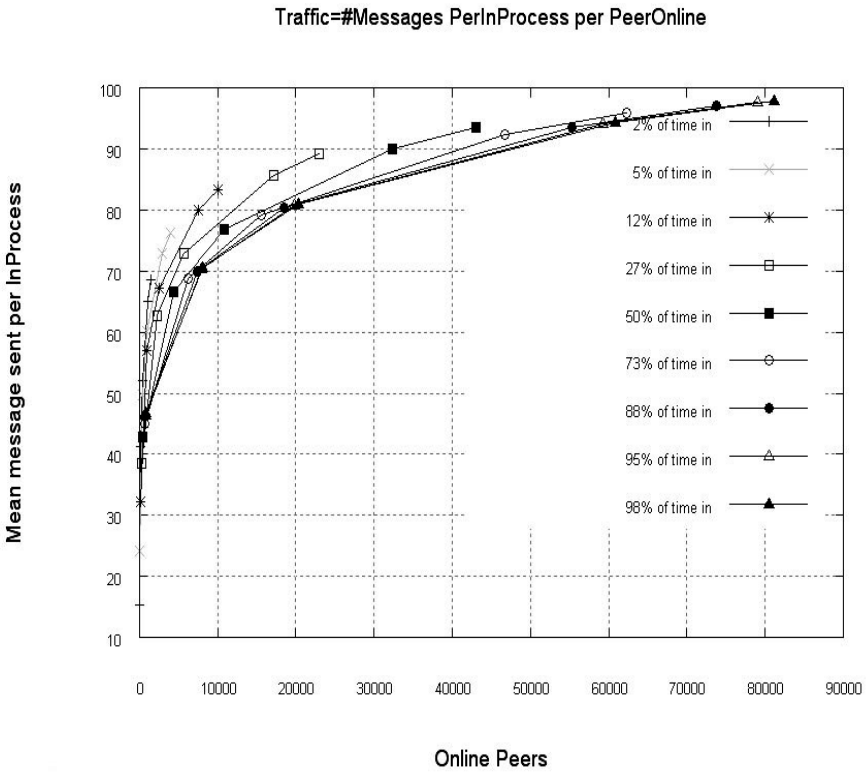


Σχήμα 4.31: Το Traffic για τις διαδικασίες εισόδου σε σχέση με τους online peers που βρίσκονται στο σύστημα.(data balance)

Αρχικά παρατηρούμε στο σχήμα 4.30 και 4.31 ότι και για volume balance και για data balance η μορφή του traffic είναι ίδια και απλά στη περίπτωση του data balance απαιτούνται πάντα περισσότερα μηνύματα. Παρατηρούμε επίσης ότι όταν το ποσοστό του χρόνου που βρίσκονται online οι peers στο σύστημα είναι μικρότερο από ό,τι offline τότε στέλνονται λίγο περισσότερα μηνύματα για το ίδιο πλήθος από online peers όπως ακριβώς παρατηρήσαμε και για το latency. Αυτό έχει άμεση σχέση με το μέγεθος του trie και θα δούμε στη συνέχεια με ποιο τρόπο.

Έστω ότι έχουμε έναν αριθμό N από online peers που είναι υπεύθυνοι για δύο διαφορετικά tries , ένα μεγέθους K και ένα μεγέθους $M < K$. Τα αποτελέσματα στα οποία καταλήξαμε μας αποδεικνύουν ότι στο πρωτόκολο P-grid στέλνονται περισσότερα μηνύματα όταν το trie είναι μεγαλύτερο αφού Όπως έχουμε εξηγήσει στη παράγραφο 4.2 όταν το ποσοστό του χρόνου που βρίσκονται online οιpeers γίνει μικρότερο τότε για ίδιο πλήθος από online peers έχουν δημιουργηθεί περισσότερα φύλλα. Τα αποτελέσματα του σχήματος 4.30 και 4.31 μας αποδεικνύουν δηλαδή ότι για περισσότερα φύλλα στέλνονται περισσότερα μηνύματα. Καθώς αυξάνονται τα φύλλα του trie , για το ίδιο πλήθος από online peers , αυξάνονται οι περιπτώσεις όπου κάποιο φύλλο ενός peer προωθεί ένα μήνυμα πρός κάποιο άλλο φύλλο που έχει τον ίδιο υπεύθυνο peer. Αυτό έχει σαν αποτέλεσμα να μην στέλνεται πρακτικά εκείνο το μήνυμα και άρα να μην το μετράμε. Άρα καθώς αυξάνονται τα φύλλα αυξάνεται και η πιθανότητα να μη καταγράφονται αυτού του είδους τα μηνύματα. Τα αποτελέσματα του traffic των εισόδων αλλά και των εξόδων όπως θα δούμε στη συνέχεια, μας αποδεικνύουν ότι η αύξηση του μεγέθους του trie αναγκάζει τελικά τους peers να στέλνουν περισσότερα μηνύματα μεταξύ τους για κάθε είσοδο και έξοδο παρόλο που ο κάθε peer γίνεται υπεύθυνος για περισσότερα φύλλα.

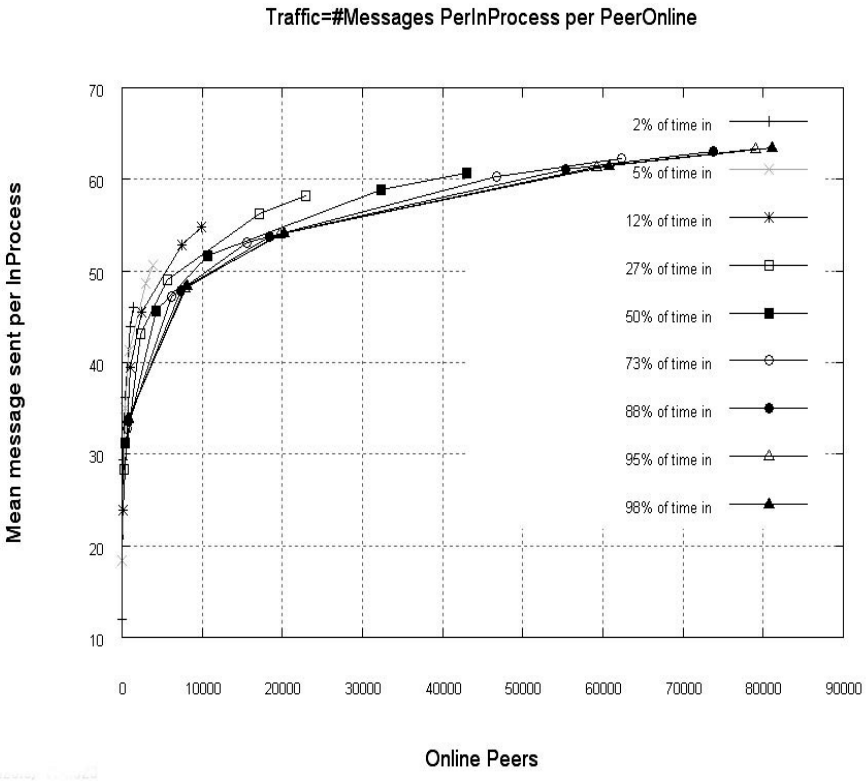
Στο σχήμα 4.32 παρατηρούμε το traffic για τις εισόδους των peers όταν η συνάρτηση exchange λαμβάνει μέρος στις διαδικασίες εισόδου με πιθανότητα 1/3, στο σχήμα 4.33 με πιθανότητα 1/9, στο σχήμα 4.34 παρατηρούμε το traffic για τις εισόδους όταν δεν χρησιμοποιείται καθόλου η συνάρτηση



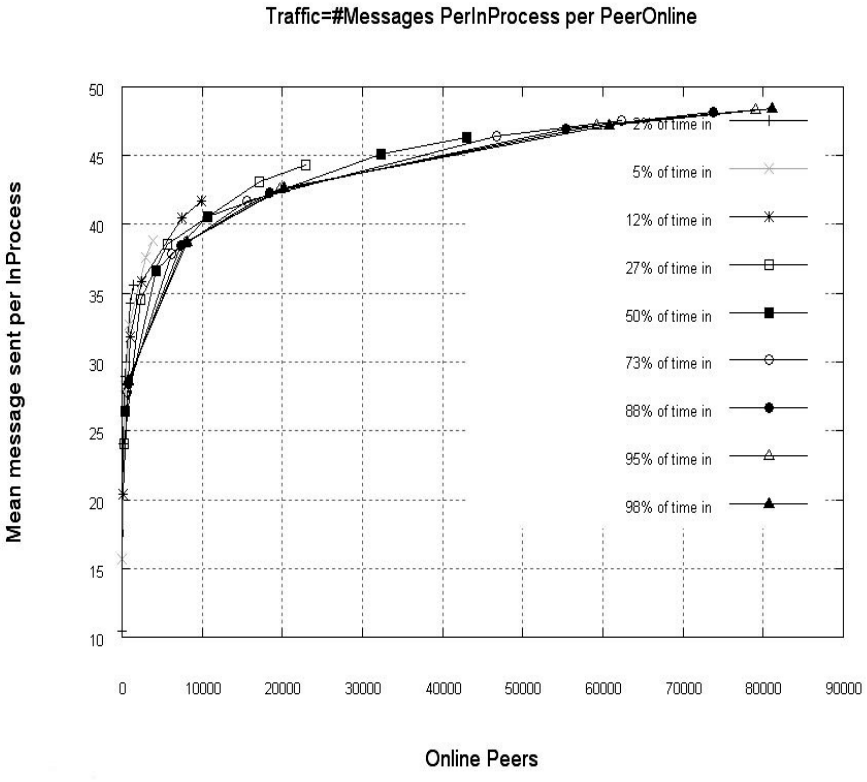
Σχήμα 4.32: Το Traffic για διαδικασίες εισόδου σε σχέση με τους online peers που βρίσκονται στο σύστημα όταν η συνάρτηση exchange χρησιμοποιείται με πιθανότητα 1/3.(volume balance)

exchange και τέλος στο σχήμα 4.35 παρατηρούμε τη περίπτωση όπου δεν χρησιμοποιείται η συνάρτηση exchange ενώ συγχρόνως χρησιμοποιούμε ως routing table των πίνακα εισερχομένων και εξερχομένων links μαζί.

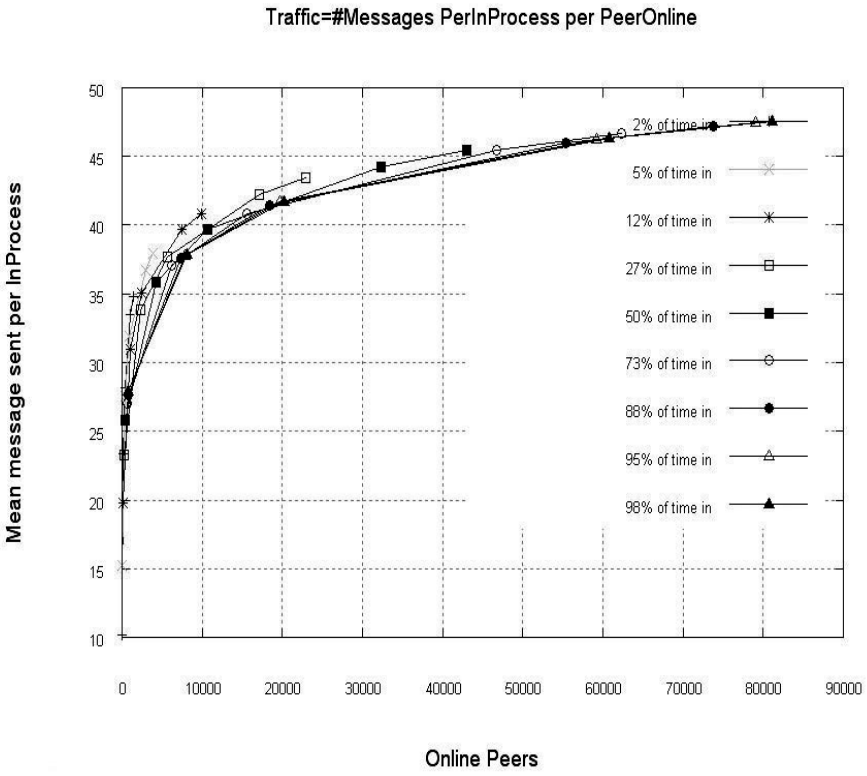
Παρατηρώντας τα σχήματα 4.32 ως 4.35 θα δούμε ότι όσο λιγότερο χρησιμοποιούμε τη συνάρτηση exchange τόσο πιο λίγα μηνύματα στέλνονται ανά διαδιακασία εισόδου. Τα πειράματα του σχήματος 4.34 και του σχήματος 4.35 που δεν χρησιμοποίησαν καθόλου τη συνάρτηση exchange μας δείχνουν ότι στάλθηκαν περίπου τέσσερις φορές λιγότερα μηνύματα ανά διαδικασία εισόδου σε σχέση με τα μηνύματα του σχήματος 4.30 όπου η συνάρτηση exchange χρησιμοποιήθηκε κανονικά. Αυτό ήταν αναμενόμενο αφού τα περισσότερα μηνύματα που έχουν σχέση με τις εισόδους νέων peers είναι αυτά που στέλνονται λόγω της συνάρτησης exchange όπως άλλωστε έγινε φανερό στο κεφάλαιο 3.2 όπου είδαμε στη πράξη τα μηνύματα που απαιτούνται προκειμένου να λάβει μέρος η λειτουργία της συνάρτησης.



Σχήμα 4.33: Το Traffic για διαδικασίες εισόδου σε σχέση με τους online peers που βρίσκονται στο σύστημα όταν η συνάρτηση exchange χρησιμοποιείται με πιθανότητα 1/9.(volume balance)

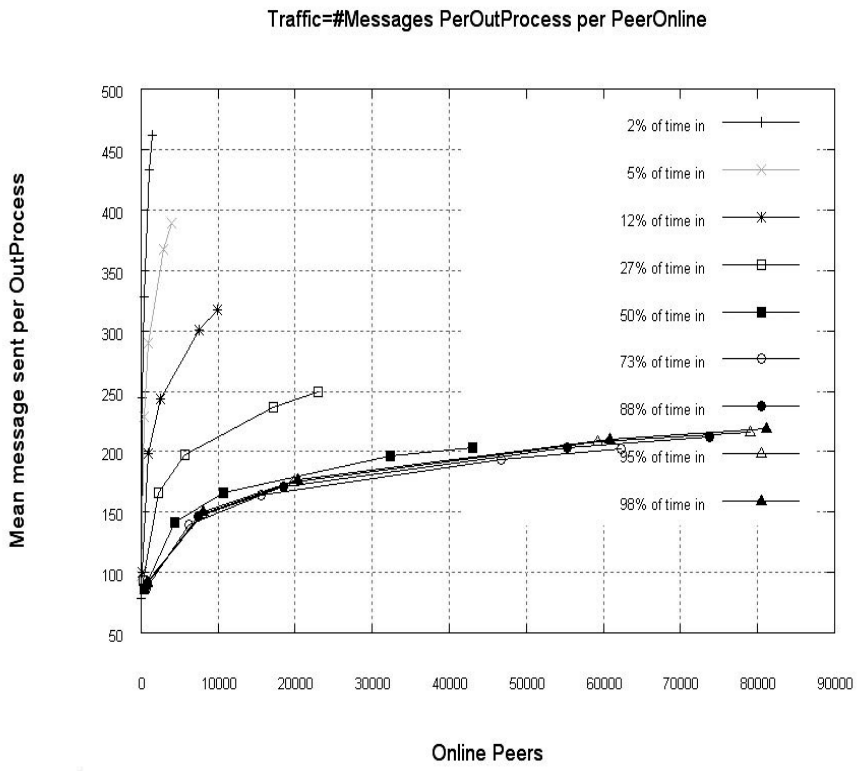


Σχήμα 4.34: Το Traffic για διαδικασίες εισόδου σε σχέση με τους online peers που βρίσκονται στο σύστημα όταν η συνάρτηση exchange δεν χρησιμοποιείται ποτέ.(volume balance)

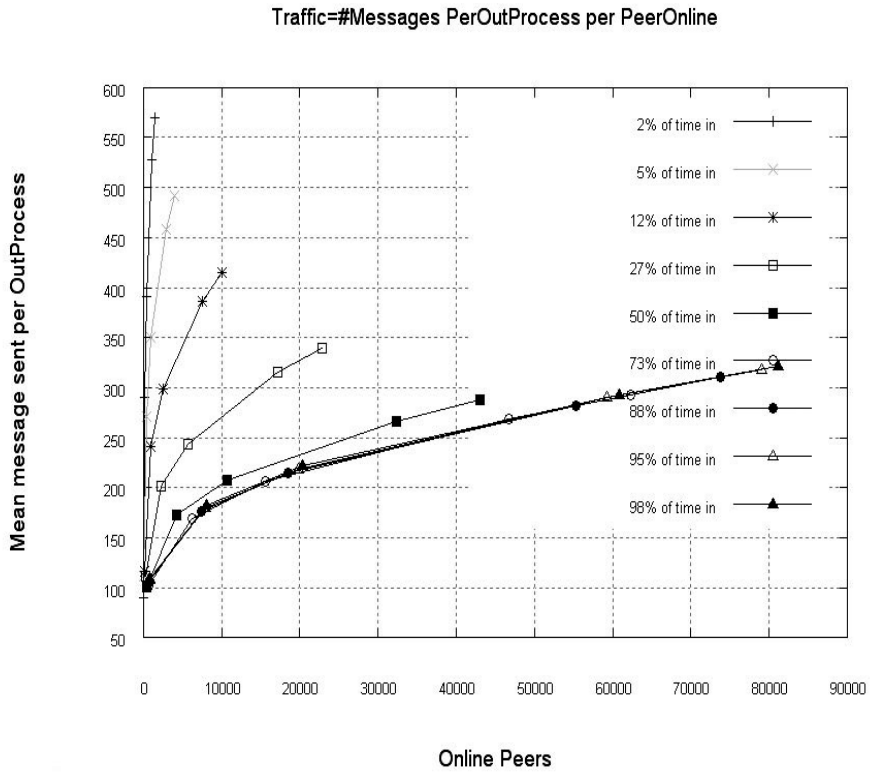


Σχήμα 4.35: Το Traffic για διαδικασίες εισόδου σε σχέση με τους online peers που βρίσκονται στο σύστημα όταν η συνάρτηση exchange δεν χρησιμοποιείται ποτέ ενώ συγχρόνως χρησιμοποιούμε ως routing table των πίνακα εισερχομένων και εξερχομένων links μαζί.(volume balance)

Στα σχήματα 4.36 και 4.37 παρατηρούμε traffic για τις διαδικασίες εξόδου των peers από το σύστημα όταν χρησιμοποιούμε volume balance και data balance αντίστοιχα. Τα μηνύματα αυτά είναι εκείνα που στέλνονται κατά τη διάρκεια μιας διαδικασίας εξόδου προκειμένου κάποιοι peers να αναλάβουν τα φύλλα για τα οποία ήταν υπεύθυνος ο peer που βγήκε offline.



Σχήμα 4.36: Το Traffic για τις διαδικασίες εξόδου σε σχέση με τους online peers που βρίσκονται στο σύστημα.(volume balance)

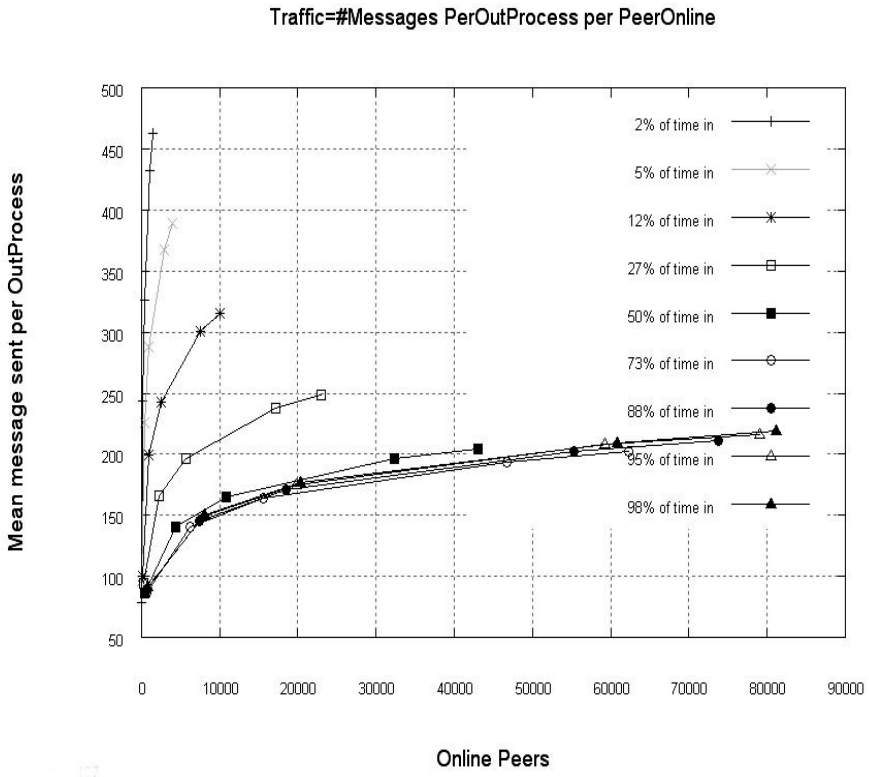


Σχήμα 4.37: Το Traffic για τις διαδικασίες εξόδου σε σχέση με τους online peers που βρίσκονται στο σύστημα.(data balance)

Όπως και στο traffic των διαδικασιών εισόδου νέων peers έτσι και για τις διαδικασίες εξόδου δεν υπάρχει αισθητή διαφορά στη μορφή των γραφικών παραστάσεων για volume balance και data balance. Η διαφορά και πάλι είναι ότι στη περίπτωση του data balance στέλνονται περισσότερα μηνύματα. Επίσης παρατηρούμε ότι όταν το ποσοστό του χρόνου που βρίσκονται online οι peers γίνει μεγαλύτερο του 50 % το πλήθος των μηνυμάτων που στέλνονται για κάθε έξοδο είναι σχεδόν ίδιο για κάθε ένα από ποσοστά 73%, 88%, 95%, 98%. Αυτή η παρατήρηση είναι πολύ σημαντική γιατί μας δείχνει ότι όταν το μέγεθος του trie είναι μικρότερο από μια συγκεκριμένη τιμή τότε τα μηνύματα που στέλνονται ανά διαδιακασία εξόδου είναι σχεδόν ίδια για το ίδιο πλήθος online peers ακόμα κι αν το trie είναι πολύ μικρότετο από εκείνη τη τιμή. Να θυμίσουμε ότι στο κεφάλαιο 4.2 παρατηρήσαμε το σχήμα 4.7 και συμπεράναμε ότι όταν το ποσοστό του χρόνου που βρίσκονται online οι peers γίνει μικρότερο, τότε για ίδιο πλήθος από online peer έχουν δημιουργθεί περισσότερα φύλλα. Στο σχήμα 4.30 παρατηρήσαμε ότι στέλνονται σχεδόν τα ίδια μηνύματα ανά διαδικασία εισόδου ακόμα κι αν το trie γίνει πολύ μεγαλύτερο. Τώρα παρατηρούμε ότι για τις διαδικασίες εξόδου δεν συμβαίνει το ίδιο αφού για τα ποσοστά 2%, 5%, 12%, 27% και 50% η διαφορά είναι τεράστια. Δηλαδή για μικρότερα ποσοστά χρόνου στέλνονται πολύ περισσότερα μηνύματα ανά διαδικασία εξόδου κατά μέσο όρο. Αν κοιτάξουμε το σχήμα 4.18 που είδαμε στην ανάλυση του Λ_{max} για τις εξόδους θα δούμε ότι για τα πολύ μικρά ποσοστά χρόνου εντός του συστήματος έχουμε πολύ μεγαλύτερα και διαφορετικά Λ_{max}. Δηλαδή ενώ στέλνονται περισσότερα μηνύματα κατά μέσο όρο όταν έχουμε μικρότερα ποσοστά χρόνου (και άρα μεγαλύτερο trie) παρατηρούμε ότι το Λ_{max} γίνεται πολύ καλύτερο. Αυτή η παρατήρηση είναι πολύ σημαντική και μας δείχνει ότι όταν έχει φτιαχτεί ένα μεγαλύτερο trie, τα συνολικά μηνύματα που στέλνονται ανά διαδικασία εξόδου είναι περισσότερα αλλά τα μηνύματα που έλαβε ο πιο φορτωμένος peer είναι λιγότερα από ότι αν είχε δημιουργηθεί ένα μικρότερο trie.

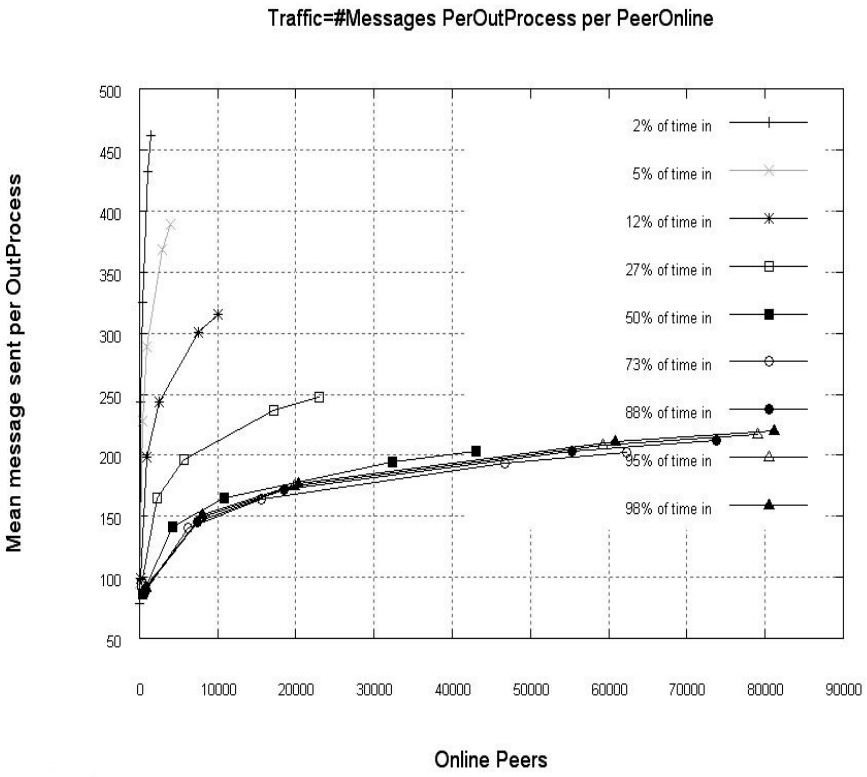
Στο σχήμα 4.38 παρατηρούμε το traffic για τις εξόδους των peers όταν η συνάρτηση exchange λαμβάνει μέρος στις διαδικασίες εισόδου με πιθανότητα 1/3, στο σχήμα 4.39 με πιθανότητα 1/9, στο σχήμα 4.40 παρατηρούμε το traffic για τις εξόδους όταν δεν χρησιμοποιείται καθόλου η συνάρτηση exchange και τέλος στο σχήμα 4.41 παρατηρούμε τη περίπτωση όπου δεν χρησιμοποιείται η συνάρτηση exchange ενώ συγχρόνως χρησιμοποιούμε ως routing table των πίνακα εισερχομένων και εξερχομένων links μαζί.

Αν κοιτάξουμε τα σχήματα 4.38 ως 4.41 θα παρατηρήσουμε ότι το πλήθος των μηνυμάτων που στέλνονται ανά διαδικασία εξόδου δεν επηρεάζεται καθόλου από τη συνάρτηση exchange. Οι γραφικές παραστάσεις για τις εξόδους είναι ακριβώς οι ίδιες και είναι φανερό ότι το πλήθος των μηνυμάτων ανά έξοδο δεν επηρεάζεται καθόλου από την exchange.

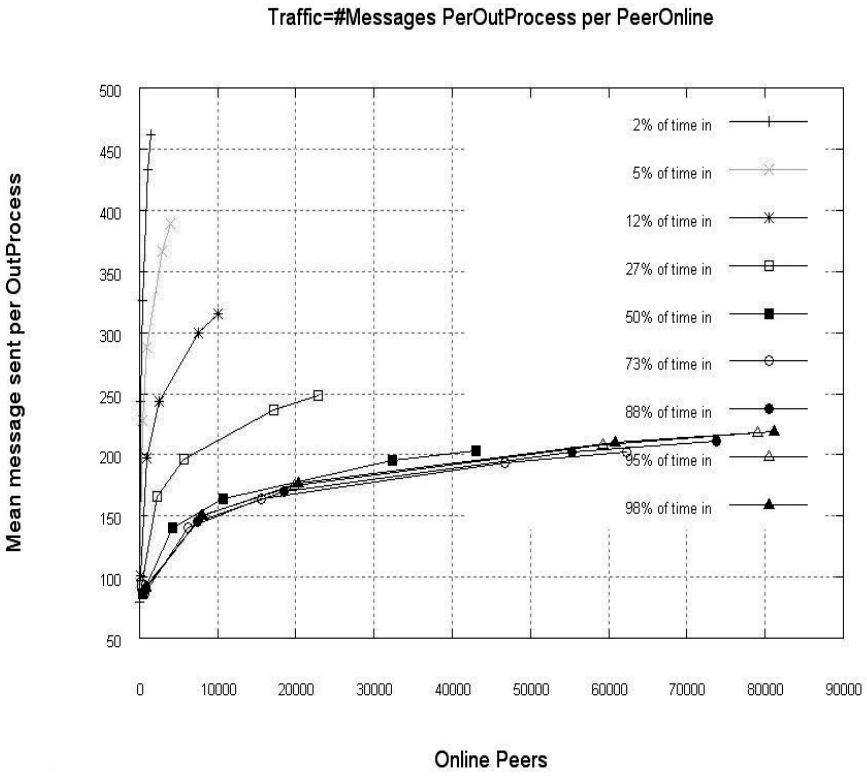


Σχήμα 4.38: Το Traffic για διαδικασίες εξόδου σε σχέση με τους online peers που βρίσκονται στο σύστημα όταν η συνάρτηση exchange χρησιμοποιείται με πιθανότητα 1/3.(volume balance)

Σχήμα 4.39: Το Traffic για διαδικασίες εξόδου σε σχέση με τους online peers που βρίσκονται στο σύστημα όταν η συνάρτηση exchange χρησιμοποιείται με πιθανότητα 1/9.(volume balance)



Σχήμα 4.40: Το Traffic για διαδικασίες εξόδου σε σχέση με τους online peers που βρίσκονται στο σύστημα όταν η συνάρτηση exchange δεν χρησιμοποιείται ποτέ.(volume balance)



Σχήμα 4.41: Το Traffic για διαδικασίες εξόδου σε σχέση με τους online peers που βρίσκονται στο σύστημα όταν η συνάρτηση exchange δεν χρησιμοποιείται ποτέ ενώ συγχρόνως χρησιμοποιούμε ως routing table των πίνακα εισερχομένων και εξερχομένων links μαζί.(volume balance)

Κεφάλαιο 5

Ανασκόπηση και μελλοντική δουλειά

5.1 Ανασκόπηση

Η εργασία μας έφτασε στο τέλος της αφού ήδη ολοκληρώσαμε τους αρχικούς στόχους μας και είδαμε τα αποτελέσματα των πειραμάτων μας. Εξηγήσαμε αναλυτικά τα χαρακτηριστικά του πρωτοκόλου που μελετήσαμε αλλά και της προσομοίωσης που πραγματοποιήσαμε για τα πειράματά μας. Μετρήσαμε τα μηνύματα που στέλνονται για κάθε διαδικασία εισόδου και εξόδου και είδαμε πως διαμορφώνεται το σύστημα καθώς αυξάνεται σε αριθμό από online peers αλλά και με ποιο τρόπο επηρεάζεται το κόστος για κάθε νέα είσοδο ή έξοδο από το σύστημα. Αυτό είναι σημαντικό γιατί μετρώντας όλα αυτά τα μηνύματα και χρησιμοποιώντας τα απαραίτητα κριτήρια αξιολόγησης για το σύστημα P-grid αποφανθήκαμε για το πόσο καλά προσαρμόζεται το σύστημα καθώς αυξάνεται σε μέγεθος και πόσο γρήγορη είναι η δρομολόγηση των διαδικασιών εισόδου και εξόδου των peers από πλευράς μηνυμάτων. Όπως είδαμε στην ανάλυση των αποτελεσμάτων το σύστημα είναι καλύτερο οταν χρησιμοποιούμε volume balance και όχι data balance για όλα τα μεγέθη που μετρήσαμε. Επίσης είδαμε ότι και για τις διαδικασίες εξόδου και για τις διαδικασίες εισόδου των peers από και προς το σύστημα, όταν το ποσοστό του χρόνου που βρίσκονται online οι peers γίνει μεγαλύτερο του 50 % τότε τα μηνύματα που στέλνονται για κάθε έξοδο και είσοδο αντίστοιχα είναι σχεδόν ίδια για κάθε ένα από τα ποσοστά 73%, 88%, 95% και 98%. Είδαμε τη μορφή των γραφικών παραστάσεων για τα βασικότερα κριτήρια που αφορούν τέτοιου είδους διαδικασίες και παρατηρήσαμε ότι το Maximum throughput παραμένει σχεδόν σταθερό για τα πειράματα που πραγματοποιήθηκαν με data balance κάτι που είναι πολύ απογοητευτικό αφού ίσως περιμέναμε να αυξάνεται τουλάχιστον με κάποιο μικρό ρυθμό. Επίσης είδαμε ότι το maximum throughput για τις διαδικασίες εισόδο αυξάνεται πάντα καθώς αυξάνονται οι online peers και ότι η συνάρτηση exchange λειτουργεί πολύ αρνητικά για το Maximum throughput που αφορά τις εισόδους. Επίσης είναι σημαντικό να επισημάνουμε ότι το απογοητευτικό maximum throughput για τις εξόδους επηρεάζει πάρα πολύ το συνολικό maximum throughput και οφείλεται κυρίως στα πολλά μηνύματα που δέχονται κάποιοι peers οι οποίοι είναι υπεύθυνοι για φύλλα τα οποία διατηρούν πάρα πολλά εισερχόμενα links σε σχέση με τον μέσο όρο. Όπως έχουμε ήδη αναφέρει στο κεφάλαιο 4.3 το πρόβλημα αυτό θα μπορούσε να λυθεί με κάποιες τυχαίες ανταλλαγές φύλλων μεταξύ των peers ανά τακτά χρονικά διαστήματα καθώς αυτό θα δημιουργούσε μια πιο ομοιόμορφη κατανομή των μηνυμάτων που λαμβάνουν οι peers.

5.2 Προβλήματα και μελλοντική δουλειά

Πρέπει να εξηγήσουμε σε αυτό το σημείο ότι υπάρχουν κάποιες ειδικές περιπτώσεις που δεν έχουμε λάβει υπ όψην οι οποίες μπορεί να προκύψουν καθώς

γίνεται κάποια διαδικασία εισόδου ή εξόδου. Όπως έχουμε ήδη εξηγήσει ο σκοπός της εργασίας μας ήταν να αποφανθούμε για τη συμπεριφορά του συστήματος σε επίπεδο μηνυμάτων που σχετίζονται με διαδικασίες εισόδου και εξόδου των peers. Θεωρήσαμε δεδομένο ότι οι peers είναι πάντα αξιόπιστοι όπως και το δίκτυο internet που χρησιμοποιούν και δεν λάβαμε υπ'οψην τα προβλήματα που μπορεί να προκύψουν καθώς πραγματοποιούνται αυτές οι διαδικασίες όταν δεν συμβαίνουν οι παραπάνω παραδοχές.

- Υποθέσαμε ότι οι χρήστες μπορούν οποιαδήποτε χρονική στιγμή να βγουν εκτός του συστηματος P-grid. Θεωρήσαμε δεδομένο ότι αυτή τη διαδικασία εξόδου τη πραγματοποιεί ο ιδιος ο peer και υπό φυσιολογικές συνθήκες στέλνει ένα μήνυμα σε κάθε έναν από τους online peers που θα αναλάβει να είναι υπεύθυνος για κάποιο από τα φύλλα του. Κάποιος peer όμως μπορεί να βγει ξαφνικά offline όχι πάντα επειδή το επιθυμεί. Μπορεί για παράδειγμα να έγινε κάποια ξαφνική βλάβη στο δίκτυο internet το οποίο χρησιμοποιεί και να βρεθεί εκτός του συστήματος P-grid ξαφνικά και απροειδοποίητα. Σε αυτη τη περίπτωση δεν θα γίνει γνωστό σε κανέναν peer αυτή η ξαφνική έξοδος και η προσομοίωσή μας δεν θα μπορούσε να συνεχιστεί. Το πρόβλημα αυτό θα μπορούσε να λυθεί αν όλοι peers ζητούσαν ανα τακτά χρονικά διαστήματα από άλλους peers να τους ενημερώσουν γα την κατάστασή τους και για το αν εξακολουθούν να βρίσκονται εντός του συστήματος P-grid. Έτσι χρησιμοποιώντας τα εισερχόμενα και εξερχόμενα links θα μπορούσαν όλοι να μάθουν για κάποια ξαφνική προβληματική έξοδο κάποιου άλλου peer που θα γνωρίζουν.
- Υποθέτωντας ότι όλοι οι peers είναι πάντα online απορρίπτουμε αμέσως μια περίπτωση που αφορά τις διαδικασίες εισόδου ενός νέου peer. Καθώς προωθείται προς τον προορισμό το αρχικό μήνυμα για αίτηση μιας νέας εισαγωγής θα ήταν πιθανό ο υπεύθυνος peer κάποιου φύλλου που μπορεί να μεσολαβεί σε αυτή τη διαδικασία να βγεί ξαφνικά offline ή να έχει ξεκινήσει ήδη τη διαδικασία εξόδου και να μην έχει ολοκληρωθεί ακόμη. Αυτή είναι μια προβληματική περίπτωση και θα μπορούσε κάποιος να σκεφτεί εύκολα κάποιες πιθανές λύσεις για μια τέτοια κατάσταση. Μια λύση θα ήταν να ακυρώσουμε τη διαδικασία και να την επαναλάβουμε από την αρχή. Επίσης θα μπορούσε να σταματήσει προσωρινά η διαδικασία εισόδου εώς ότου ολοκληρωθεί η διαδικασία εξόδου εκείνου του peer που μεσολαβεί και χρειάζεται προκειμένου να φτάσει το αρχικό μήνυμα στον προορισμό του.
- Επίσης μια προβληματική περίπτωση θα ήταν κατά τη διαδικασία split κάποιου φύλλου προκειμένου να δώσει στον νέο peer μερίδιο από τον χώρο δεδομένων. Καθώς γίνεται η διαδικασία split του Mate όπως περιγράψαμε, θα ήταν πιθανό να βρεθούν ξαφνικά offline ο υπεύθυνος peer του Mate , ο νέος peer για τον οποίο γίνεται η διαδικασία η ακόμα και οι δύο. Σε αυτή τη περίπτωση θα είχαμε τον Mate η ακόμα χειρότερα δύο νεα φύλλα που προέκυψαν από το split του Mate, για τα οποία δεν υπάρχει κανένας peer που να είναι υπεύθυνος. Συγχρόνως μπορεί οι γείτονες του Mate που θα έχει κάνει ήδη split να μην προλάβουν να ενημερωθούν για το ότι ο Mate έχει κάνει split και τη θέση του έχουν πάρει δύο νέα φύλλα. Σε αυτή τη περίπτωση η λύση δεν θα μπορούσε να είναι κάποια που να μην λαμβάνει υπ όψην τους γείτονες του Mate. Όπως αναφέραμε παραπάνω οι peers θα μπορούσαν να ζητούν ενημέρωση της κατάστασης κάποιων άλλων peers ανα τακτά χρονικά διαστήματα. Αν γινόταν αυτό, κάποιος υπεύθυνος peer κάποιου γείτονα του Mate, θα ζητούσε να ενημερωθεί από τον υπεύθυνο του Mate για το αν εξακολουθεί να βρίσκεται online. Όταν θα έβλεπε ότι αυτό δεν ισχύει θα μπορούσε να αναλάβει αυτός να είναι υπεύθυνος για τον Mate. Αφού γινόταν αυτός ο νέος υπεύθυνος peer του Mate θα έπρεπε να ελέγξει αν ο Mate εξακολουθεί να είναι φύλλο του trie ή αν έχει κάνει εσφαλμένα split. Αν συμβαίνει κάτι τέτοιο

θα έπρεπε είτε να αναλάβει τα δύο νέα φύλλα που δημιουργήθηκαν είτε να τα συνενώσει ώστε να επιστρέψουμε στην αρχική κατάσταση όπου ο Mate είναι φύλλο του trie. Στη συνέχεια ο νέος υπεύθυνος peer θα έπρεπε να ενημερώσει τους υπεύθυνους peers των φύλλων που ανήκουν στα εισερχόμενα ή εξερχόμενα links του Mate.

– Ακόμη μια πιθανή προβληματική περίπτωση θα μπορουσε να συμβεί κατά τη διαδικασία εξόδου κάποιου peer. Αρχικά ο peer που επιθυμεί να βγεί εκτός του συστήματος βρίσκει τυχαίους online peers για να αναλάβουν να είναι υπεύθυνοι για τα φύλλα του. Αφού επιλέξει κάποιον peer για να αναλάβει κάποιο από τα φύλλα του, θα ήταν πιθανό εκείνος ο peer να βρεθεί εκείνη τη στιγμή offline από το σύσημα, πριν αναλάβει το φύλλο για το οποίο επιλέχτηκε. Σε αυτή τη περίπτωση θα υπήρχαν δύο διαδικασίες εξόδου, μια φυσιολογική και μια προβληματική η οποίες εξαρτώνται η μια από την άλλη. Όταν ο peer που βγήκε offline με τον φυσιολογικό τρόπο καταλάβει ότι ο peer που επέλεξε για να του δώσει κάποιο φύλλο έχει πρόβλημα και δεν ανταποκρίνεται, τότε θα μπορούσε απλά να επιλέξει κάποιον άλλο και η διαδικασία να εξελιχθεί φυσιολογικά.

Περιγράψαμε λοιπόν κάποιες βασικές περιπτώσεις στις οποίες οι διαδικασίες εισόδου ή εξοδου θα έπρεπε να δρομολογηθούν με διαφορετικό και πιο ευέλικτο τρόπο που θα τους επέτρεπε να είναι πάντα αξιόπιστες και να ολοκληρωθούν επιτυχώς ανεξάρτητα από τα προβλήματα που πορεί να προέκυπταν.

Μια μελλοντική εργασία που θα μπορούσε να βοηθήσει αυτή την έρευνα, θα ήταν η μελέτη αυτών των προβληματικών καταστάσεων με σκοπό να μάθουμε ποιες είναι η πιο γρήγορες και με λιγότερο κόστος λύσεις για αυτά τα προβλήματα αλλά και το πως αλλάζει η απόδοση του συστήματος για τις διάφορες εργασίες που μπορεί να εκτελεί. Μπορούμε ακόμα να προσθέσουμε ότι το γέγονος ότι το Λ_{max} για τις εξόδους δεν είναι ικανοποιητικό μπορεί να βελτιωθεί σημαντικά αν σκεφτούμε ένα τρόπο για να κάνουμε πιο ομοιόμορφη την κατανομή των εισερχομένων links των φύλλων του trie. Επίσης θα μπορούσαμε ακόμα να μελετήσουμε το πως επηρεάζονται οι ερωτήσεις που μπορεί να πραγματοποιούνται στο σύστημα, όταν συγχρόνως γίνονται είσοδοι και έξοδοι των peers με φυσιολογικό ή όχι τρόπο. Έτσι θα μπορούσαμε να έχουμε έναν πιο ρεαλιστικό κόσμο στον οποίο μελετάμε το σύστημα και θα παρατηρούσαμε τη συμπεριφορά του για τις διάφορες εργασίες που μπορεί να εκτελεί έχοντας να αντιμετωπίσει συγχρόνως προβλήματα που είναι πιθανό να προκύψουν λόγω της αναξιοπιστίας των χρηστών ή και του δικτύου.

Bibliography

[1] E. K. Lua, J. Crowcroft, M. Pias, R. Sharma, and S. Lim, IEEE Communications Surveys & Tutorials **7** (2005).

[2] K. Aberer, P-grid: A self-organizing access structure for p2p information systems, in *In CoopIS*, pp. 179–194, 2001.

[3] K. Aberer *et al.*, SIGMOD Record **32** (2003).

[4] A. Datta, M. Hauswirth, R. John, R. Schmidt, and K. Aberer, Range queries in trie-structured overlays, in *Proceedings of the Fifth IEEE International Conference on Peer-to-Peer Computing (P2P)*, pp. 57–66, IEEE Computer Society, 2005.

[5] S. Blanas, Contention-based performance evaluation of multidimensional range search in peer-to-peer networks.

[6] M. Argyrioy, V. Samoladas, and S. Blanas, Grasp: Generalized range search in peer-to-peer networks.

[7] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Schenker, A scalable content-addressable network, in *Proceedings of the 2001 conference on applications, technologies, architectures, and protocols for computer communications (SIGCOMM)*, pp. 161–172, ACM Press, 2001.

[8] E. K. Lua, J. Crowcroft, M. Pias, R. Sharma, and S. Lim, IEEE Communications Survey and Tutorial **7**, 72 (2005).

[9] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, Chord: A scalable peer-to-peer lookup service for internet applications, in *Proceedings of the 2001 conference on applications, technologies, architectures, and protocols for computer communications (SIGCOMM)*, pp. 149–160, ACM Press, 2001.

[10] M. Jelasity, S. Voulgaris, R. Guerraoui, A.-M. Kermarrec, and M. van Steen, ACM Trans. Comput. Syst. **25**, 8 (2007).