Πανεπιστήμιο Ιωαννίνων
Σχολή Θετικών Επιστημών
Τμήμα Πληροφορικής

Topics in Database Systems: Data Management in Peer-to-Peer Systems

Άρθρο Βιβλιογραφικής Επισκόπησης

Χριστοδουλίδου Μαρία

Ιωάννινα, 17 Ιουνίου 2005

Πίνακας περιεχομένων

Πίνακας εικόνων

1 Περίληψη

Κατά την τελευταία δεκαετία, η ανάπτυξη που γνώρισε το διαδίκτυο ήταν μεγάλη και αναπτύχθηκαν πολλές εφαρμογές. Η πιο πρόσφατη είναι οι εφαρμογές Συστημάτων Ομότιμων Κόμβων (**peer-to-peer** ή **p2p**) διαμοιρασμού δεδομένων, οι οποίες επιτρέπουν στους χρήστες, που ονομάζονται peers, να διαμοιράζονται δεδομένα διαμέσου του διαδικτύου. Η ραγδαία ανάπτυξή τους συνεχίζεται και δείχνει ότι πρόκειται να παίξουν σημαντικό ρόλο στο μέλλον.

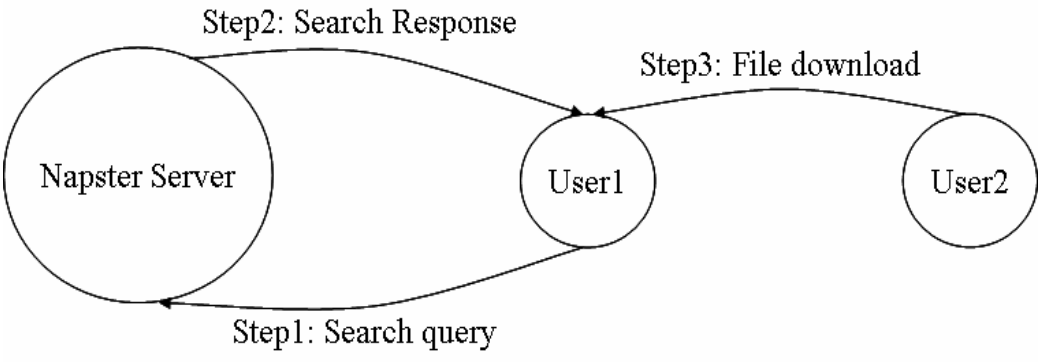
Στην εργασία αυτή παρουσιάζονται οι κατηγορίες στις οποίες διακρίνονται τα P2P συστήματα, καθώς, επίσης, και κάποιοι βασικοί εκπρόσωποί τους που έχουν προταθεί και μελετηθεί έως τις μέρες μας, καθένας από τους οποίους εμφανίζει διάφορα πλεονεκτήματα και μειονεκτήματα, ανάλογα με τον τρόπο με τον οποίο πραγματοποιείται η αποθήκευση και η διαχείριση των δεδομένων. Έμφαση δίνεται στις μεθόδους αναζήτησης δεδομένων που χρησιμοποιούνται. Ακόμη, παρουσιάζονται κάποιες τεχνικές δημιουργίας και ενημέρωσης αντιγράφων, που βελτιώνουν την ανοχή των συστημάτων σε σφάλματα και αυξάνουν τη διαθεσιμότητα των δεδομένων. Στη συνέχεια, αναφερόμαστε σε πρόσφατες τεχνικές που αναπτύχθηκαν και έχουν ως σκοπό να κάνουν αποδοτικότερη την αναζήτηση ομαδοποιώντας κόμβους με συσχετιστικά όμοια δεδομένα. Τέλος, δεν παραλείπουμε να αναφερθούμε σε τεχνικές υλοποίησης ερωτήσεων διαστήματος, για μονοδιάστατα, αλλά και πολυδιάστατα δεδομένα. Καταλήγοντας, εξάγουμε κάποια συμπεράσματα για τα P2P συστήματα, τα πλεονεκτήματα τους και τις εφαρμογές στις οποίες μπορούν να χρησιμοποιηθούν.

2 Εισαγωγή

Όπως είναι γνωστό, στο διαδίκτυο αρχικά χρησιμοποιήθηκε το αρχιτεκτονικό μοντέλο του εξυπηρετούμενου – εξυπηρέτη (client - server). Σε αυτό, τα δεδομένα αποθηκεύονται στους εξυπηρέτες. Για την εύρεση κάποιας πληροφορίας ο χρήστης (εξυπηρετούμενος) εντοπίζει τον εξυπηρέτη που κατέχει την επιθυμητή πληροφορία και κάνει μια αίτηση για να την αποκτήσει. Κατόπιν, ο εξυπηρέτης ανταποκρίνεται παρέχοντας, συνήθως, την ίδια την πληροφορία. Τις περισσότερες φορές υπάρχει μικρός αριθμός εξυπηρετών που παρέχουν υπηρεσίες στους υπόλοιπους χρήστες. Οι εξυπηρέτες αυτοί είναι συνήθως διαθέσιμοι για μεγάλο χρονικό διάστημα, ενώ οι εξυπηρετούμενοι παραμένουν συνδεδεμένοι για λίγο.
Στα P2P συστήματα η παραπάνω αρχιτεκτονική εγκαταλείπεται. Οι χρήστες δρουν και ως εξυπηρετούμενοι και ως εξυπηρέτες, δηλαδή αποθηκεύουν, ανακτούν και συνεισφέρουν στην εύρεση πληροφοριών που αναζητούνται σε ένα δυναμικό σύστημα. Επίσης, η αποθήκευση και διαχείριση των δεδομένων γίνεται με διαφορετικό τρόπο. Κάθε φορά που ένας νέος κόμβος εισέρχεται στο σύστημα, τα αρχεία που επιθυμεί να διαμοιραστεί είναι αμέσως διαθέσιμα στους υπόλοιπους χρήστες. Η πληροφορία του συστήματος, λοιπόν, αποτελείται από τις επιμέρους πληροφορίες που βρίσκονται αποθηκευμένες σε κάθε κόμβο [19].
Το μοντέλο αυτό, όμως, γεννά δύο σημαντικά ζητήματα: ποιος μηχανισμός αναζήτησης θα εφαρμοστεί προκειμένου να είναι δυνατή η αποτελεσματική εύρεση των επιθυμητών πληροφοριών και πώς θα γίνει τελικά η μεταφορά των δεδομένων [10]. Στα P2P συστήματα προτείνονται δύο λύσεις: κεντρικοποιημένη και αποκεντρικοποιημένη αναζήτηση.
Τα κεντρικοποιημένα συστήματα (Napster [13]), αποδεικνύονται ιδιαίτερα ευάλωτα καθώς υπάρχει μοναδικό σημείο αποτυχίας, κατάρρευση του οποίου μπορεί να οδηγήσει σε κατάρρευση ολόκληρου του συστήματος. Επίσης, η απόδοσή τους είναι μικρή όταν το πλήθος των κόμβων αυξηθεί πολύ.
Τα μειονεκτήματα αυτά έρχονται να καλύψουν αποκεντρικοποιημένα συστήματα, όπως είναι τα δίκτυα Gnutella [12], Freenet [11](αδόμητα, όπου η σύνδεση των κόμβων ακολουθεί χαλαρούς κανόνες), τα Chord [14], CAN [20] (δομημένα, όπου οι κόμβοι συνδέονται με αυστηρό τρόπο και δημιουργείται ένα ιδεατό δίκτυο για την πραγματοποίηση της δρομολόγησης) και άλλα. Φυσικά, σε κάθε ένα από αυτά χρησιμοποιούνται διαφορετικές τεχνικές αποθήκευσης και διαχείρισης των δεδομένων.
Στην εργασία αυτή παρουσιάζονται οι βασικές κατηγορίες P2P συστημάτων και κάποια συστήματα που, κατά καιρούς, έχουν προταθεί. Κατόπιν, εστιάζουμε σε ορισμένες τεχνικές δημιουργίας αντιγράφων. Τέλος, αναφερόμαστε σε πρόσφατες τεχνικές που προτείνουν τη χρήση πληροφοριών σχετικών με τα δεδομένα που διαθέτουν οι κόμβοι για τη βελτίωση της απόδοσης και συζητούμε τι μπορεί να συμβεί στο μέλλον.

3 Κεντρικοποιημένα συστήματα

Στις αρχές του 1999, ο **Shawn Fanning**, φοιτητής στο Northeastern University των Η.Π.Α., δημιούργησε το **Napster** [13], που έμελλε να αποτελέσει κινητήριο μοχλό της εξέλιξης του φαινόμενου που ονομάζεται **Συστήματα Ομότιμων Κόμβων**.



Εικόνα 1 Κεντρικοποιημένο P2P σύστημα

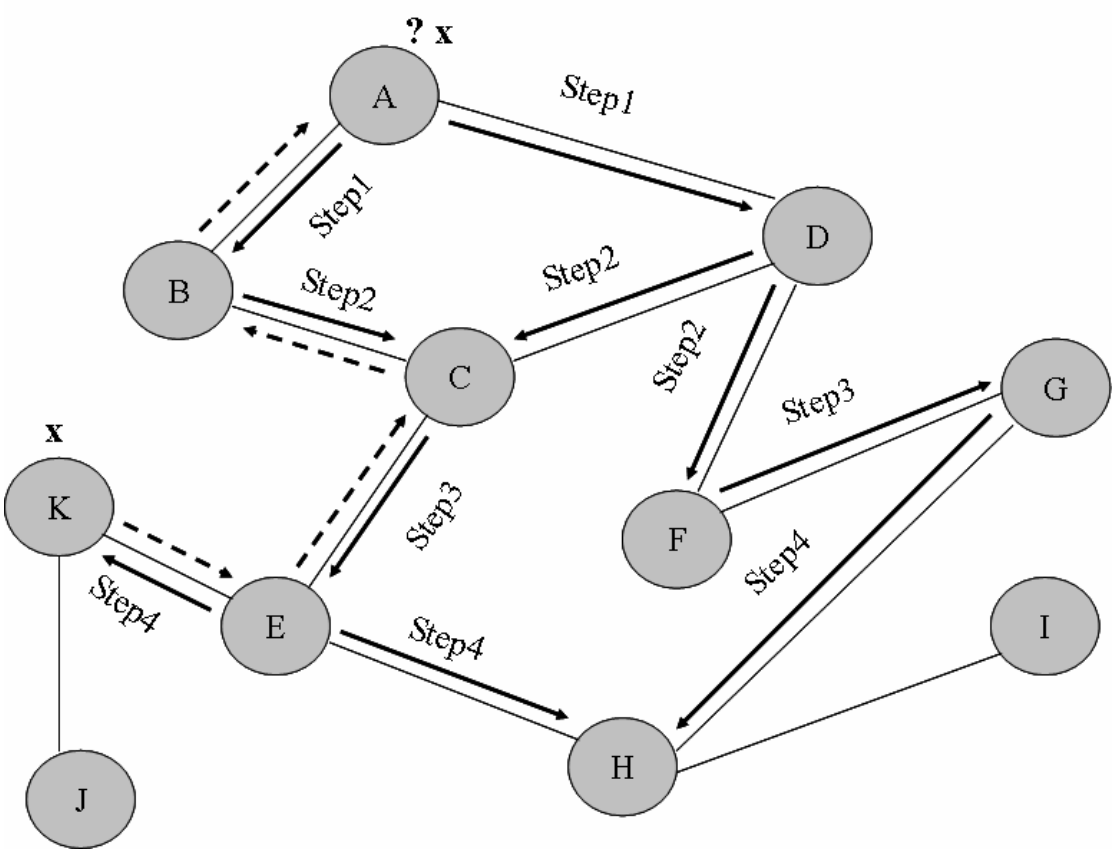
Το Napster παρουσιάζει κεντρικοποιημένη αρχιτεκτονική. Τα διαμοιραζόμενα αρχεία όλων των συνδεδεμένων κόμβων βρίσκονται καταγεγραμμένα στο μοναδικό κεντρικό εξυπηρέτη (**server**). Κάθε χρήστης έπρεπε να συνδεθεί στον εξυπηρέτη του Napster και να δηλώσει ποια αρχεία του επιθυμεί να μοιραστεί. Τα αρχεία αυτά καταχωρούνταν στον κεντρικό κατάλογο. Ο κατάλογος αυτός ενημερωνόταν περιοδικά από το λογισμικό των χρηστών που συνδέονταν στο σύστημα και αποσυνδέονταν από αυτό, με αποτέλεσμα να υπάρχουν, πιθανώς, άκυρα δεδομένα. Οι ερωτήσεις των χρηστών που ήταν συνδεδεμένοι με τον εξυπηρέτη του Napster στέλνονταν στο **Napster Web site**. Η απάντηση, που επιστρεφόταν από τον εξυπηρέτη μετά από αναζήτηση στον κεντρικό κατάλογο, ήταν μία λίστα των κόμβων που είχαν το επιθυμητό αρχείο. Στη συνέχεια, ο εξυπηρέτης επέτρεπε τη σύνδεση μεταξύ των κόμβων (αυτού που έκανε την αίτηση και αυτού που κατείχε το αρχείο) και το αρχείο μεταφερόταν απ' ευθείας. Ο τρόπος με τον οποίο λειτουργούσε το Napster φαίνεται στην Εικόνα 1. Ο χρήστης 1 ρωτά τον εξυπηρέτη, εκείνος του απαντά ότι ο χρήστης 2 κατέχει την πληροφορία που αναζητά και η μεταφορά γίνεται απ'ευθείας μεταξύ των δύο χρηστών.

Όπως είπαμε, το Napster αποτελεί κεντρικοποιημένο μοντέλο συστήματος ομοτίμων. Η ύπαρξη ενός κεντρικού καταλόγου επιτρέπει στους χρήστες να αναζητούν και να βρίσκουν τα επιθυμητά αρχεία αποτελεσματικά, δηλαδή εξετάζοντας μόνο τον κατάλογο αυτό και όχι δομές που θα έπρεπε να διατηρούνται σε άλλους κόμβους του συστήματος και στις οποίες η πρόσβαση θα ήταν πιο αργή λόγω δρομολόγησης, αλλά και αποτελεί το βασικότερο μειονέκτημα καθώς αποτυχία αυτού οδηγεί σε κατάρρευση ολόκληρου του συστήματος.

Στον κεντρικοποιημένο χαρακτήρα του Napster οφείλονται και τα νομικά προβλήματα που οδήγησαν στο κλείσιμό του. Τα μειονεκτήματα του Napster έρχονται να καλύψουν νέα αποκεντρικοποιημένα συστήματα.

4 Μη κεντρικοποιημένα, αδόμητα συστήματα

Σε αυτά τα συστήματα δεν υπάρχει κεντρικός κατάλογος, επομένως μοναδικό σημείο αποτυχίας, αλλά ούτε και αυστηρή δομή. Η **Gnutella** [12] είναι ένα χαρακτηριστικό παράδειγμα. Αναπτύχθηκε στις αρχές του 2000 και χρησιμοποιήθηκε κυρίως για την ανταλλαγή .mp3 αρχείων. Στη Gnutella η αναζήτηση πραγματοποιείται μέσω ενός **κατανεμημένου μηχανισμού** που χρησιμοποιεί τους κόμβους του συστήματος για να εντοπίσει την επιθυμητή πληροφορία.



Εικόνα 2 Το αποκεντρικοποιημένο σύστημα Gnutella

4.1 Πώς δουλεύει η Gnutella

Οι κόμβοι στο σύστημα ονομάζονται **servents**, καθώς δρουν και ως εξυπηρέτες (**SERVers)** και ως εξυπηρετούμενοι (**cliENTS)**. Για να εισέλθει ένας νέος κόμβος (έστω ο κόμβος Α), δεν απαιτείται σύνδεση με κάποιον κεντρικό εξυπηρέτη, αλλά ο κόμβος συνδέεται αρχικά (μέσω **TCP/IP** σύνδεσης) με έναν ήδη συνδεδεμένο στο σύστημα κόμβο (έστω το Β) και του ανακοινώνει την παρουσία του μέσω ενός μηνύματος, Υπάρχουν διευθύνσεις (π.χ. **gnutellahosts.com**) όπου μπορεί κανείς να βρει διευθύνσεις κόμβων που είναι σχεδόν πάντα διαθέσιμοι στο σύστημα. Στη συνέχεια, ο Β ενημερώνει τους γείτονές του ότι ο Α έχει συνδεθεί στο σύστημα. Η διαδικασία αυτή συνεχίζεται με τους γειτονικούς κόμβους του Β, που ενημερώνουν τους δικούς τους γείτονες. Κάθε νέος κόμβος στον οποίο φθάνει η ανακοίνωση κατ' αυτόν τον τρόπο ενημερώνει στη συνέχεια τους δικούς του γείτονες, έως ότου εξαντληθεί το περιθώριο (**Time-To-Live** ή **TTL**), που ορίζεται με το πρωταρχικό μήνυμα. Ο μετρητής TTL μειώνεται κατά ένα κάθε φορά που η ύπαρξη του

νέου κόμβου ανακοινώνεται σε ένα επίπεδο κόμβων που ήδη υπάρχουν στο σύστημα. Όταν λάβει μηδενική τιμή, η μετάδοση του μηνύματος σταματά. Όσο μεγαλύτερο είναι το TTL, τόσοι περισσότεροι κόμβοι θα ενημερωθούν για την παρουσία του νέου κόμβου.
Αφού λοιπόν ο νέος χρήστης συνδεθεί στο σύστημα, μπορεί να αναζητήσει πληροφορίες μέσω μηνυμάτων. Κάθε μήνυμα χαρακτηρίζεται από τυχαίο αναγνωριστικό και φέρει ένα μετρητή TTL συνήθως με τιμή 7. Ο κόμβος αποστέλλει το μήνυμα αρχικά στους γείτονές του. Το μήνυμα, στη συνέχεια, προωθείται από κάθε κόμβο που το λαμβάνει στους δικούς του γείτονες μέχρι να εξαντληθεί το TTL ή να βρεθεί κόμβος που κατέχει το δεδομένο που αναζητείται, οπότε η απάντηση, που περιέχει τα στοιχεία που είναι απαραίτητα για την απ' ευθείας μεταφορά του δεδομένου, προωθείται προς τα πίσω στον αρχικό κόμβο. Ένα παράδειγμα φαίνεται στην Εικόνα 2, όπου ο κόμβος Α αναζητά το δεδομένο Χ που κατέχει ο Κ. Ο Α ρωτά τους γείτονές του αρχικά, αυτοί ρωτούν με τη σειρά τους δικούς τους και η διαδικασία συνεχίζεται, ώσπου ρωτιέται ο Κ. Στη συνέχεια, η απάντηση προωθείται προς τα πίσω ακολουθώντας το αντίστροφο μονοπάτι προς τον Α.
Πρέπει να αναφέρουμε, επίσης, ότι καθώς το σύστημα αλλάζει δυναμικά, οι συνδέσεις των κόμβων δεν είναι απολύτως αξιόπιστες και πρέπει να ελέγχονται με περιοδικά μηνύματα. Φυσικά, τα πολλά μηνύματα που ανταλλάσσονται μεταξύ των κόμβων δημιουργούν συμφόρηση, που αποτελεί βασικό μειονέκτημα.
Το σύστημα Gnutella μοιάζει με **power-law δίκτυο**. Σε τέτοιου είδους δίκτυα, οι περισσότεροι κόμβοι έχουν λίγες συνδέσεις, ενώ λίγοι κόμβοι συνδέονται με πολλούς άλλους, επομένως μπορεί να αφαιρεθεί μεγάλος αριθμός κόμβων χωρίς να χαθεί η συνεκτικότητά του. Αυτή η ιδιότητα αποτελεί σημαντικό πλεονέκτημα, αλλά και μειονέκτημα, καθώς το καθιστά ευάλωτο σε καλά σχεδιασμένες επιθέσεις που θα είχαν ως στόχο υψηλά συνδεδεμένους κόμβους.

4.2 Μέθοδοι αναζήτησης

Για τα Αδόμητα Συστήματα Ομότιμων Κόμβων έχουν αναπτυχθεί πολλές μέθοδοι αναζήτησης που παρουσιάζουν διάφορα πλεονεκτήματα και μειονεκτήματα [6], [9], [18].

4.2.1 Τυφλές μέθοδοι αναζήτησης

Στις μεθόδους αυτές οι κόμβοι δε διαθέτουν πληροφορίες για την τοποθεσία των δεδομένων στο σύστημα. Μερικές είναι οι ακόλουθες:

- **Πλημμύρα**: κάθε κόμβος αποστέλλει μια ερώτηση σε όλους τους γείτονές του μέχρι να εξαντληθεί το TTL ή να ικανοποιηθεί (Gnutella). Στη μέθοδο αυτή δημιουργείται συμφόρηση λόγω των πολλών μηνυμάτων που αποστέλλονται, ενώ είναι δύσκολη και η ανεύρεση μη δημοφιλών δεδομένων.
- **Modified-BFS** [6]: ο κόμβος επιλέγει τυχαία μόνο ένα μέρος των γειτονικών του για να προωθήσει την ερώτηση, με αποτέλεσμα να μειώνεται ο αριθμός των μηνυμάτων που κινούνται στο σύστημα.
- **Expanding ring**: ο κόμβος ξεκινά την αποστολή της ερώτησης με μικρό TTL. Αν η αναζήτηση αποτύχει, ξεκινά νέα με αυξημένο TTL. Η διαδικασία συνεχίζεται μέχρι να ικανοποιηθεί η ερώτηση. Η μέθοδος αποδίδει καλύτερα όταν δημιουργούνται αντίγραφα των δημοφιλών δεδομένων.
- **Random walks** [6]: ο κόμβος αποστέλλει k μηνύματα σε k τυχαίους γείτονές του. Κάθε ένα ακολουθεί διαφορετικό μονοπάτι, προωθούμενο από έναν ενδιάμεσο κόμβο σε κάποιον τυχαίο γείτονα. Οι ερωτήσεις, που ονομάζονται **walkers**, επιτυγχάνουν όταν βρεθεί το δεδομένο ή αποτυγχάνουν όταν εξαντληθεί το TTL ή όταν ανακαλύψουν, μέσω περιοδικών μηνυμάτων στον κόμβο που εκκίνησε, την

ερώτηση ότι έχει ικανοποιηθεί. Με τη μέθοδο αυτή μειώνεται σημαντικά ο αριθμός των μηνυμάτων.

- **GUESS** [6]: οι κόμβοι διακρίνονται σε ultrapeers και leaf-nodes. Κάθε ultrapeer συνδέεται με άλλους ultrapeers και ένα σύνολο leaf-nodes. Η αναζήτηση πραγματοποιείται με τους ultrapeers να επικοινωνούν με άλλους ultrapeers και αυτούς να ρωτούν όλους τους leaf-nodes τους μέχρι να βρεθεί ικανοποιητικός αριθμός δεδομένων.
- **Gnutella2** [6]: όταν ένας super-peer λάβει μια ερώτηση από ένα φύλλο, την αποστέλλει στα άλλα φύλλα και στους γειτονικούς super-peers μέχρι να ικανοποιηθεί. Γειτονικοί super-peers ανταλλάσσουν πληροφορίες για να αποφευχθεί άσκοπη επικοινωνία.

4.2.2 Μέθοδοι αναζήτησης με πληροφορία

Στις μεθόδους αυτές υπάρχει κάποια πληροφορία που διευκολύνει την αναζήτηση. Μερικές από αυτές είναι οι εξής:

- **Intelligent-BFS** [6]: οι κόμβοι διατηρούν πληροφορίες για προηγούμενες απαντήσεις που έλαβαν από τους γείτονές τους και προωθούν μια νέα ερώτηση σε εκείνους από τους γειτονικούς κόμβους που είχαν απαντήσει σε περισσότερες ερωτήσεις στο παρελθόν. Όταν βρεθεί το δεδομένο, η απάντηση προωθείται προς τα πίσω και ενημερώνονται οι διατηρούμενες πληροφορίες.
- **APS** [6]: κάθε κόμβος διατηρεί πληροφορίες για κάθε δεδομένο που ζήτησε από κάθε γειτονικό. Οι πληροφορίες αυτές αποτελούν την πιθανότητα ενός γείτονα να επιλεγεί ως επόμενος στην προώθηση μιας ερώτησης. Κατά την αναζήτηση αναπτύσσονται k ανεξάρτητα μονοπάτια (**walkers**) και η προώθηση γίνεται βάσει των πληροφοριών. Οι αντίστοιχες πιθανότητες αυξάνονται ή μειώνονται αν ένας walker επιτύχει ή αποτύχει, αντίστοιχα.
- **Local indices** [6]: κάθε κόμβος δεικτοδοτεί τα δεδομένα όλων των κόμβων σε ακτίνα r από αυτόν και απαντά ερωτήσεις εκ μέρους τους. Η αναζήτηση πραγματοποιείται όπως στον BFS, αλλά οι κόμβοι επικοινωνούν με άλλους κόμβους έως κάποιο βάθος. Φυσικά, απαιτείται ανανέωση των πληροφοριών όταν εισέρχονται ή αποχωρούν κόμβοι με χρήση πλημμύρας με TTL = r.
- **Routing indices (RIs)** [1]: σε αυτή τη μέθοδο τα δεδομένα καταχωρούνται σε κατηγορίες. Κάθε κόμβος γνωρίζει πόσα δεδομένα κάθε κατηγορίας μπορεί να ανακτήσει μέσω των γειτονικών του. Οι ερωτήσεις προωθούνται στον 'καλύτερο' κόμβο, όπως προκύπτει από μια συνάρτηση αξιολόγησης, μέχρι να ανακτηθεί ικανοποιητικός αριθμός δεδομένων.

  Διακρίνουμε τρία είδη ευρετηρίων δρομολόγησης: α) **Compound Routing Indices (CRIs)** – οι πληροφορίες που διατηρούνται σε κάθε κόμβο αφορούν όλα τα δεδομένα που μπορούν να βρεθούν μέσω αυτού. Η δημιουργία και η συντήρησή τους γίνεται μέσω μηνυμάτων που ανταλλάσσουν οι κόμβοι κάθε φορά που δημιουργείται μια νέα σύνδεση ή αποθηκεύεται ένα νέο δεδομένο στο σύστημα. β) **Hop-count Routing Indices** – κάθε κόμβος διατηρεί πληροφορίες για δεδομένα που μπορεί να ανακτήσει μέσω των γειτονικών του έως ένα μέγιστο πλήθος βημάτων (*horizon*), δηλαδή λαμβάνεται και η απόσταση που πρέπει να διανυθεί προκειμένου να ικανοποιηθεί μια ερώτηση. γ) **Exponentially aggregated RIs** – και εδώ διατηρούνται πληροφορίες για τα δεδομένα που μπορούν να ανακτηθούν μέσω κόμβων σε κάποια απόσταση, αλλάζει όμως η αξιολόγηση των γειτόνων, αφού κάθε κόμβος αξιολογείται βάσει της απόστασής του, αλλά και του πλήθους των δεδομένων που μπορεί να παράσχει.

Δυνατή είναι και η οπισθοδρόμηση προκειμένου να ικανοποιηθεί μια ερώτηση. Η αρχικοποίηση και η ενημέρωση των πληροφοριών γίνεται μέσω πλημμύρας, ενώ μειονέκτημα αποτελεί πιθανή ανακρίβεια των πληροφοριών λόγω συσχετισμένων κατηγοριών, λάθος καταμετρήσεων ή κύκλων στο δίκτυο.

- **Distributed Resource Location Protocol** [6]: κάθε κόμβος που δε γνωρίζει τίποτα για τη θέση ενός δεδομένου προωθεί την ερώτηση σε κάθε γειτονικό με μια πιθανότητα. Αν το δεδομένο βρεθεί, ακολουθείται το μονοπάτι προς τα πίσω και κάθε κόμβος σε αυτό αποθηκεύει τη θέση του δεδομένου για να διευκολυνθούν επόμενες αναζητήσεις. Είναι προφανές ότι αρχικά απαιτούνται πολλά μηνύματα για την εύρεση δεδομένων, όσο όμως οι ερωτήσεις αυξάνονται, οι αναζητήσεις πραγματοποιούνται πιο αποδοτικά.

4.3 Δημιουργία και συντήρηση αντιγράφων

Για να αυξήσουμε τη διαθεσιμότητα των δεδομένων, έτσι ώστε οι αναζητήσεις να πραγματοποιούνται αποδοτικότερα, και για να βελτιώσουμε την ανοχή του συστήματος σε σφάλματα μπορούμε να δημιουργήσουμε αντίγραφα της πληροφορίας για την ανεύρεση δεδομένων ή των ίδιων των δεδομένων σε περισσότερους κόμβους. Πολλές είναι οι μέθοδοι που υπάρχουν [9], [18].

4.3.1 Μέθοδοι δημιουργίας αντιγράφων

Ανάλογα με το πού τοποθετούνται τα αντίγραφα, διακρίνουμε τις μεθόδους:

- **Owner replication**: όταν μια αναζήτηση είναι επιτυχής, δημιουργείται αντίγραφο του δεδομένου στον κόμβο που το ζήτησε αρχικά. Χρησιμοποιείται στη Gnutella.
- **Path replication**: το δεδομένο αποθηκεύεται σε όλους τους κόμβους στο μονοπάτι μεταξύ του κόμβου που το ζήτησε και εκείνου που το παρείχε. Τείνει να δημιουργεί αντίγραφα τοπολογικά κοντά στο ιδεατό δίκτυο. Χρησιμοποιείται στο Freenet.
- **Random replication**: επιλέγουμε τυχαία κάποιους από τους κόμβους στους οποίους προωθήθηκε η ερώτηση και δημιουργούμε αντίγραφα.

Ανάλογα με τον αριθμό των αντιγράφων που επιθυμούμε, διακρίνουμε τις μεθόδους:

- **Uniform replication**: δημιουργούμε τον ίδιο αριθμό αντιγράφων για κάθε δεδομένο του συστήματος. Πρόκειται για την απλούστερη τεχνική. Προφανώς, το γεγονός ότι δημιουργούμε αντίγραφα ακόμη και για δεδομένα που αναζητούνται σπάνια την καθιστά μη αποδοτική.
- **Proportional replication**: για κάθε δεδομένο δημιουργείται αριθμός αντιγράφων ανάλογος της πιθανότητας αναζήτησής του. Με άλλα λόγια, όσο πιο δημοφιλές είναι ένα δεδομένο, τόσο πιο πολλά αντίγραφά του δημιουργούνται με αποτέλεσμα να πραγματοποιούνται αποδοτικότερα οι αναζητήσεις. Η μέθοδός αυτή καθιστά την αναζήτηση ευκολότερη για δημοφιλή δεδομένα και δυσκολότερη για λιγότερο δημοφιλή, αλλά και οδηγεί σε εξισορρόπηση φορτίου.
- **Square-root replication**: για κάθε δεδομένο δημιουργείται αριθμός αντιγράφων ανάλογος της τετραγωνικής ρίζας της πιθανότητας αναζήτησής του. Αποδίδει καλύτερα από την παραπάνω μέθοδο, αφού το μέγεθος της αναζήτησης γίνεται βέλτιστο.

Για τον υπολογισμό καλύτερων εκτιμήσεων των πιθανοτήτων αναζήτησης δεδομένων μπορεί να χρησιμοποιηθεί η μέθοδος **Replication with probe memory**: οι κόμβοι διατηρούν το συνολικό αριθμό πρόσφατων ερωτήσεων για κάθε δεδομένο σε ολόκληρο το μονοπάτι αναζήτησης και ανάλογα με αυτόν υπολογίζουν καλύτερες εκτιμήσεις των πιθανοτήτων αναζήτησης.

4.3.2 Ενημέρωση αντιγράφων

Στα περισσότερα συστήματα, τα δεδομένα σπάνια αλλάζουν, επομένως δεν είναι απαραίτητη η συχνή ενημέρωσή τους. Υπάρχουν, όμως, και εφαρμογές όπου τα δεδομένα τροποποιούνται συχνά, καθώς το σύστημα αλλάζει δυναμικά με την πάροδο του χρόνου, χωρίς οι κόμβοι να μπορούν να έχουν μια συνολική εικόνα του συστήματος. Στην περίπτωση αυτή, τα δεδομένα και τα αντίγραφά τους πρέπει να διατηρούνται ενημερωμένα για να παρέχονται οι σωστές πληροφορίες στους χρήστες.
Στη συνέχεια περιγράφεται ένας μηχανισμός ενημέρωσης που βασίζεται στη *διάδοση φήμης* (*rumor spreading*) και αποτελεί τροποποιημένη εκδοχή υπαρχόντων αλγορίθμων πλημμύρας [3].

Αλγόριθμος ενημέρωσης

Push-phase: η ενημέρωση προωθείται από τον κόμβο που διαθέτει το τροποποιημένο αντίγραφο σε ένα μέρος των κόμβων που διαθέτουν παλαιότερη έκδοση του δεδομένου, τους οποίους γνωρίζει. Η διαδικασία συνεχίζεται ανάλογα, με τους κόμβους που λαμβάνουν την ενημέρωση να την προωθούν περαιτέρω, όπως στη μέθοδο της πλημμύρας.
Pull-phase: κόμβοι που έχουν αποσυνδεθεί από το σύστημα και επανασυνδέονται, κόμβοι που δεν έχουν λάβει ενημερώσεις για μεγάλο χρονικό διάστημα ή έχουν λάβει μηνύματα ενημέρωσης, αλλά δεν είναι σίγουροι ότι έχουν λάβει την τελευταία έκδοση κάποιου δεδομένου, στη φάση αυτή ρωτούν κάποιους από τους κόμβους που κατέχουν το συγκεκριμένο δεδομένο για να αποκτήσουν την πιο πρόσφατη έκδοσή του.

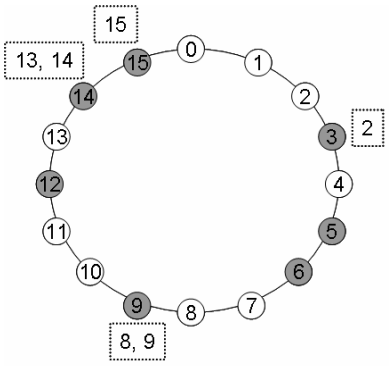
Άλλες μέθοδοι ενημέρωσης αντιγράφων περιλαμβάνουν αλγορίθμους διάδοσης φημών με κάποια τυχαιότητα. Επίσης, κάποια P2P συστήματα εφαρμόζουν κάποιες άλλες τεχνικές για να αντιμετωπίσουν το πρόβλημα. Πιο συγκεκριμένα, στο Napster [13], όπου διατηρείται ένας κατάλογος στον κεντρικό εξυπηρέτη με τα δεδομένα που είναι διαθέσιμα στο σύστημα, αν κάποιο δεδομένο τροποποιηθεί, κόμβοι που διατηρούν αντίγραφά του δεν ενημερώνονται. Απλώς εξαρτάται από το χρήστη να ανακαλύψει την πιο πρόσφατη έκδοση του δεδομένου που αναζητά. Ανάλογα αντιμετωπίζεται το πρόβλημα στη Gnutella [12].
Στο Freenet [11] δημιουργούνται αντίγραφα των δεδομένων κατά την αναζήτηση σε όσους κόμβους βρίσκονται στο μονοπάτι μεταξύ του κόμβου που αναζητά ένα δεδομένο και εκείνου που το παρέχει. Σε περίπτωση τροποποίησης, προωθούνται μηνύματα ενημέρωσης με ευριστικό τρόπο, αλλά τίποτα δεν εγγυάται ότι οι τροποποιήσεις θα γίνουν γνωστές σε όλους του κατόχους του συγκεκριμένου δεδομένου.
Στο OceanStore, μηνύματα ενημέρωσης δημιουργούν νέες εκδόσεις δεδομένων. Για τη διατήρηση της συνέπειας, ο κόμβος που τροποποιεί κάποιο δεδομένο ενημερώνει κάποιους από τους κατόχους του, οι οποίοι εκτελούν ένα πρωτόκολλο για να συμφωνήσουν ποια είναι η τελευταία έκδοση του δεδομένου και στη συνέχεια διασπείρουν τις πληροφορίες περαιτέρω στο σύστημα.

4.3.3 Διαγραφή αντιγράφων

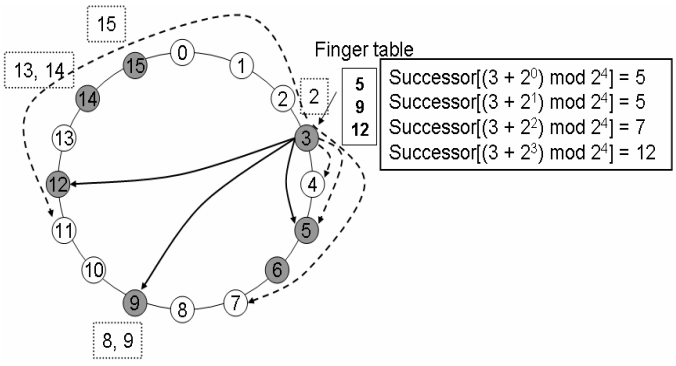
Καθώς ο χώρος αποθήκευσης στους κόμβους του συστήματος είναι περιορισμένος, αντίγραφα διαγράφονται και νέα δημιουργούνται με την πάροδο του χρόνου. Μπορούμε να διαγράψουμε τυχαία ένα αντίγραφο, να χρησιμοποιήσουμε τον αλγόριθμο First In First Out (FIFO) ή να αντιστοιχίσουμε συγκεκριμένη διάρκεια ζωής σε κάθε δεδομένο. Οι αλγόριθμοι Least Recently Used (LRU) και Least Frequently Used (FLU) δε χρησιμοποιούνται καθώς επιθυμούμε τεχνικές ανεξάρτητες από το ρυθμό αναζήτησης των δεδομένων και την ταυτότητά τους [9], [18].

5 Μη κεντρικοποιημένα, δομημένα συστήματα

Στα συστήματα αυτά, η τοπολογία του δικτύου ακολουθεί αυστηρούς κανόνες. Πιο συγκεκριμένα, οι κόμβοι τοποθετούνται σε συγκεκριμένες θέσεις ενός ιδεατού δικτύου και τα δεδομένα τοποθετούνται ανάλογα σε συγκεκριμένους θέσεις του δικτύου αυτού, έτσι ώστε να πραγματοποιούνται αποδοτικά οι αναζητήσεις. Επίσης, στα περισσότερα από αυτά, το μέγεθος των απαιτούμενων πληροφοριών για τη δρομολόγηση είναι περιορισμένο, ενώ εγγυώνται ότι και το πλήθος των βημάτων αναζήτησης δεν είναι μεγάλο.
Στη συνέχεια περιγράφονται δύο από τα γνωστότερα δομημένα συστήματα, το Chord και το CAN.



Εικόνα 3 Ο δακτύλιος του Chord για m = 4.

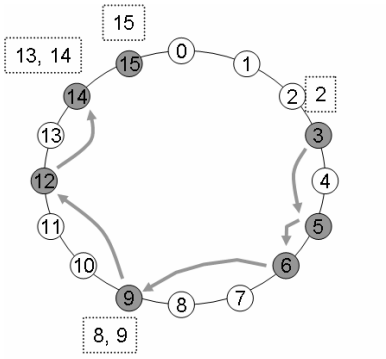


Εικόνα 4 Δημιουργία του πίνακα δεικτών.

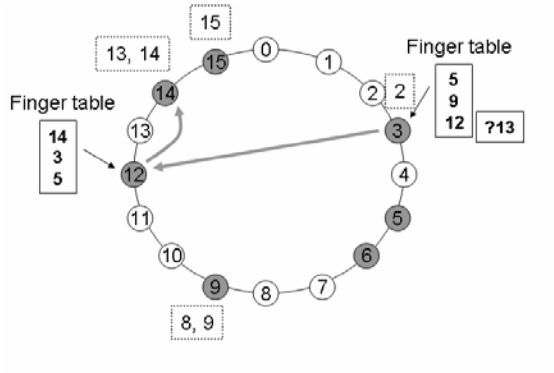
5.1 CHORD

Στο Chord [14], όλοι οι κόμβοι κατανέμονται ομοιόμορφα σε ένα δακτύλιο (**identifier circle** ή **Chord ring**) και τα δεδομένα του συστήματος κατανέμονται ομοιόμορφα στους υπάρχοντες κόμβους. Πιο συγκεκριμένα, το Chord αντιστοιχίζει ένα **κλειδί** (αναγνωριστικό που αποτελείται από m bits, πλήθος που επιλέγεται έτσι ώστε να είναι αμελητέα η πιθανότητα να υπάρξει σύγκρουση μεταξύ των αναγνωριστικών) τόσο στους κόμβους, όσο και στα δεδομένα του συστήματος χρησιμοποιώντας συνεπή κατακερματισμό (**consistent hashing**). Οι κόμβοι διατάσσονται στο δακτύλιο με βάση το υπόλοιπο της διαίρεσης του κλειδιού τους με 2^m , ενώ ένα δεδομένο φυλάσσεται στον κόμβο εκείνο που έχει το κοντινότερο μεγαλύτερο κλειδί από το κλειδί του και ονομάζεται διάδοχος κόμβος (**successor node**). Στην Εικόνα 3 βλέπουμε το δακτύλιο του Chord για m = 4. Στο σύστημα έχουν εισέλθει οι κόμβοι 3, 5, 6, 9, 12, 14 και 15, ενώ το δεδομένο 2 αποθηκεύεται στον κόμβο 3 (διάδοχο του 2), τα δεδομένα 8 κα9 9 στον 9, τα δεδομένα 13 και 14 στον 14, ενώ το δεδομένο 15 στον κόμβο 15, που υπάρχει πραγματικά.
Κάθε κόμβος στο Chord γνωρίζει, εκτός από το γειτονικό του κόμβο, και μερικούς ακόμη προκειμένου να πραγματοποιεί αποτελεσματικά αναζητήσεις κλειδιών στο σύστημα. Πιο συγκεκριμένα, κάθε κόμβος γνωρίζει όλους τους άλλους κόμβους που έχουν το κοντινότερο μεγαλύτερο κλειδί από κλειδιά τα οποία απέχουν από το δικό του κλειδί κατά αυξανόμενες δυνάμεις του 2, διατηρώντας την IP διεύθυνσή τους σε έναν πίνακα, που ονομάζεται πίνακας δεικτών (**finger table**). Επίσης, για να απλοποιηθούν οι μηχανισμοί εισαγωγής και αποχώρησης από το σύστημα, κάθε κόμβος διατηρεί και ένα δείκτη στον προηγούμενό του στο δακτύλιο (**predecessor**). Στην Εικόνα 4 βλέπουμε τους πραγματικούς δείκτες που διατηρεί ο κόμβος 4 στο δακτύλιο, ενώ με διακεκομμένες βλέπουμε που θα οδηγούσαν οι δείκτες αν υπήρχαν όλοι οι κόμβοι στο σύστημα.

Με βάση τα παραπάνω, είναι εύκολο να αποδειχθεί ότι σε ένα σταθερό σύστημα N κόμβων κάθε κόμβος γνωρίζει τα στοιχεία άλλων O(logN) κόμβων και πραγματοποιεί την αναζήτηση ενός κλειδιού μέσω O(logN) μηνυμάτων σε άλλους κόμβους.
Δύο είναι τα στοιχεία που πρέπει να προσέξουμε: κάθε κόμβος διατηρεί πληροφορίες για μερικούς μόνο άλλους κόμβους που τον ακολουθούν στο δακτύλιο και ο πίνακας δεικτών του δεν έχει όλες εκείνες τις πληροφορίες που χρειάζονται για να βρούμε απ' ευθείας τον κόμβο στον οποίο αντιστοιχεί ένα τυχαίο κλειδί.
Το Chord υποστηρίζει, επίσης, την εισαγωγή και τη διαγραφή κόμβων από το σύστημα και εγγυάται ότι λίγα δεδομένα μετακινούνται σε διαφορετικό κόμβο και στις δύο περιπτώσεις. Πιο συγκεκριμένα, όταν ένας νέος κόμβος εισέρχεται στο σύστημα, κάποια από τα κλειδιά που πριν είχαν ανατεθεί στο διάδοχό του πρέπει τώρα να ανατεθούν στον ίδιο. Ανάλογα, όταν ένας κόμβος εγκαταλείπει το σύστημα, όλα τα κλειδιά του πρέπει να ανατεθούν στο διάδοχό του. Αυτές είναι οι μόνες μετακινήσεις κλειδιών που απαιτούνται για να διατηρηθεί η συνέπεια.
Στη συνέχεια περιγράφεται αναλυτικότερα το πρωτόκολλο του δικτύου Chord.



Εικόνα 5 Το μονοπάτι που ακολουθεί μια ερώτηση για το κλειδί 13 από τον κόμβο 3 όταν χρησιμοποιείται ο απλός αλγόριθμος.



Εικόνα 6 Το μονοπάτι που ακολουθεί μία ερώτηση για το κλειδί 13 από τον κόμβο 3 όταν χρησιμοποιείται ο γρηγορότερος αλγόριθμος.

5.1.1 Αλγόριθμοι αναζήτησης κλειδιού

Ένας αλγόριθμος που θα μπορούσε να χρησιμοποιηθεί στο Chord για τον εντοπισμό κλειδιών είναι ο εξής: κάθε κόμβος επικοινωνεί με το διάδοχό του μέχρι να ικανοποιηθεί η ερώτηση. Στην Εικόνα 5 φαίνεται το μονοπάτι που θα ακολουθούσε μία ερώτηση στην περίπτωση αυτή από τον κόμβο 3 για το κλειδί 13. Το κλειδί ευρίσκεται μετά από 5 βήματα.
Ο αλγόριθμος αυτός είναι **απλός**, αλλά και **αργός**, γι' αυτό το Chord χρησιμοποιεί έναν πιο αποτελεσματικό αλγόριθμο.
Για την υλοποίηση αυτού του αλγορίθμου, κάθε κόμβος διατηρεί έναν πίνακα δρομολόγησης με το πολύ $\log 2^m = m$ καταχωρήσεις (**finger table**). Η **i-οστή** καταχώρηση του πίνακα αυτού για κάποιο κόμβο k περιέχει την ταυτότητα (αναγνωριστικό και IP διεύθυνση) του επόμενου κόμβου από τον κόμβο $\mathbf{k + 2^{(i-1)} \bmod m}$, όπου $1 \le i \le m$, **(i-οστός δείκτης ή i-th finger)**.
Κατά τον έλεγχο που πραγματοποιείται στον κόμβο k, εξετάζεται αν το κλειδί ανήκει στο διάστημα μεταξύ του k και του διαδόχου του. Αν όντως ισχύει, ο διάδοχος του k είναι ο κόμβος στον οποίο βρίσκεται το προς αναζήτηση κλειδί. Στην αντίθετη περίπτωση, ο k αναζητά στον πίνακα δεικτών τον κόμβο που έχει το κλειδί που είναι το **αμέσως μεγαλύτερο** από το κλειδί που αναζητείται και εκεί κατευθύνει την ερώτηση. Η διαδικασία

συνεχίζεται μέχρι να βρεθεί ο κόμβος στον οποίο βρίσκεται αποθηκευμένο το κλειδί που αναζητείται.
Στην Εικόνα 6 φαίνεται το μονοπάτι που ακολουθεί η προηγούμενη ερώτηση με χρήση αυτού του αλγορίθμου. Το κλειδί ευρίσκεται μόλις σε δύο βήματα.

5.1.2 Εισαγωγή νέων κόμβων στο σύστημα

Σε ένα δυναμικό σύστημα στο οποίο κόμβοι εισέρχονται και αποχωρούν οποιαδήποτε στιγμή, οι πίνακες δεικτών πρέπει να διατηρούνται ενημερωμένοι και τα κλειδιά να διατηρούνται στους σωστούς κόμβους, έτσι ώστε οι αναζητήσεις να πραγματοποιούνται αποδοτικά. Για αυτό το λόγο, κάθε φορά που ένας νέος κόμβος n εισέρχεται στο σύστημα πρέπει:

- να βρίσκει τον προηγούμενό του στο δακτύλιο και να αρχικοποιεί τον πίνακα δεικτών
- να ενημερώνονται οι πίνακες δεικτών των άλλων κόμβων και
- να μεταφέρονται στον n τα κλειδιά για τα οποία είναι πλέον υπεύθυνος.

5.1.3 Σταθεροποίηση του συστήματος

Σε ένα δυναμικό σύστημα όπου κόμβοι εισέρχονται ή αποχωρούν από αυτό, δεν αρκούν μόνο τα παραπάνω για να διατηρηθεί η συνέπεια των δομών δρομολόγησης. Γι' αυτό το λόγο χρησιμοποιείται ένα **πρωτόκολλο σταθεροποίησης**.
Κάθε κόμβος του δικτύου περιοδικά 'ρωτάει' το διάδοχό του ποιος είναι ο προηγούμενός του στο δακτύλιο για να ελέγξει μήπως πρέπει να αλλάξει διάδοχο. Επίσης, ο νέος διάδοχος ενημερώνεται για την ύπαρξη του νέου κόμβου. Επιπλέον, περιοδικά κάθε κόμβος ενημερώνει και τον πίνακα δεικτών.
Σε περίπτωση που πραγματοποιηθεί μια ερώτηση πριν να ολοκληρωθούν οι ρουτίνες σταθεροποίησης, μπορεί να καθυστερήσει ή και να αποτύχει. Αποδεικνύεται, όμως, ότι όσο ο χρόνος που χρειάζεται για την ενημέρωση των διαδόχων είναι μικρότερος από το χρόνο που περνάει για να διπλασιαστεί το σύστημα σε μέγεθος, η αναζήτηση πραγματοποιείται σε **O(logN)** χρόνο, όσο και όταν οι ρουτίνες έχουν ολοκληρωθεί.
Σημειώνεται ότι για να είναι πιο αξιόπιστο το δίκτυο, κάθε κόμβος διατηρεί μια λίστα μεγέθους r με τους προηγούμενους r διαδόχους του **(successor list)**. Έτσι, αν ο διάδοχος ενός κόμβου δεν ανταποκρίνεται, δοκιμάζεται ο **επόμενος** από τη λίστα. Για να διασπαστεί ο δακτύλιος τώρα, πρέπει να αποτύχουν ταυτόχρονα όλοι οι κόμβοι της λίστας, γεγονός με πολύ μικρή πιθανότητα να συμβεί.

Συμπερασματικά, μπορούμε να πούμε ότι το **Chord** κατορθώνει με εύκολο και αποτελεσματικό τρόπο, δοθέντος ενός κλειδιού, να εντοπίσει τον κόμβο που είναι υπεύθυνος για αυτό. Όταν το σύστημα παραμένει σταθερό, κάθε κόμβος διατηρεί πληροφορίες για **O(logN) κόμβους** και πραγματοποιεί αναζητήσεις μέσω **O(logN) μηνυμάτων**. Ανάλογα είναι τα αποτελέσματα και όταν το σύστημα αλλάζει δυναμικά.

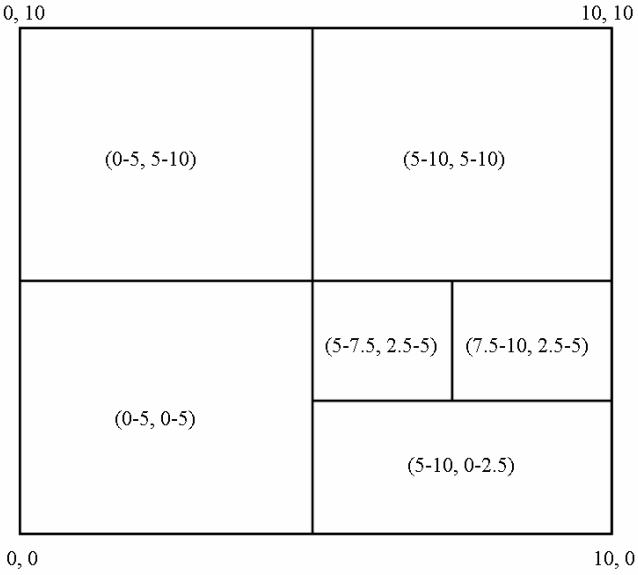
5.2 A Content-Addressable Network (CAN)

Στο **CAN** [20] υλοποιείται ένας μηχανισμός δεικτοδότησης που μπορεί να εφαρμοστεί σε δίκτυα μεγάλου μεγέθους.

5.2.1 Βασικά χαρακτηριστικά

Το **CAN** μας θυμίζει τους **πίνακες κατακερματισμού**, όπου οι βασικές λειτουργίες που επιτελούνται είναι η **εισαγωγή**, η **αναζήτηση** και η **διαγραφή** ζευγών (κλειδί, αξία). Κάθε ένας από τους κόμβους που ανήκουν σε αυτό κατέχει ένα κομμάτι του πίνακα

κατακερματισμού, που ονομάζεται ζώνη (**zone**) και διατηρεί πληροφορίες για ορισμένους άλλους κόμβους. Οι ερωτήσεις κλειδιών δρομολογούνται μέσω των κόμβων προς εκείνον τον κόμβο του οποίου η ζώνη περιέχει το συγκεκριμένο κλειδί.



Εικόνα 7 Ο χώρος συντεταγμένων του CAN.

5.2.2 Σχεδιασμός του CAN

Το **CAN** σχεδιάζεται γύρω από έναν εικονικό **d-διάστατο** χώρο καρτεσιανών συντεταγμένων πάνω σε ένα **d-torus**. Ο χώρος αυτός διαμοιράζεται σε όλους τους κόμβους του συστήματος, έτσι ώστε κάθε ένας να κατέχει μία ζώνη, και χρησιμοποιείται για την αποθήκευση ζευγών (κλειδί, αξία). Στην Εικόνα 7 βλέπουμε ένα διδιάστατο χώρο συντεταγμένων με 5 κόμβους, καθένας από τους οποίους κατέχει μια ζώνη. Στην εικόνα φαίνονται και τα όρια των ζωνών. Η **αποθήκευση** πραγματοποιείται αντιστοιχίζοντας το κλειδί σε τυχαίο σημείο P στο χώρο συντεταγμένων χρησιμοποιώντας μια συνάρτηση κατακερματισμού. Το αντίστοιχο ζεύγος (κλειδί, αξία) αποθηκεύεται, κατόπιν, στον κόμβο του οποίου η ζώνη περιέχει το P. Για την ανάκτηση του κλειδιού, ο κόμβος που ενδιαφέρεται εφαρμόζει την ίδια συνάρτηση κατακερματισμού για να βρει το P. Αν αυτό περιέχεται σε ζώνη που δεν ανήκει στον ίδιο τον κόμβο που αναζητά το κλειδί ή στους γείτονές του, η ερώτηση δρομολογείται μέσω του συστήματος στον κόμβο στον οποίο ανήκει το σημείο P, με τρόπο που περιγράφεται στη συνέχεια.
Για τη δρομολόγηση, κάθε κόμβος διατηρεί πληροφορίες για κόμβους γειτονικούς στο χώρο.

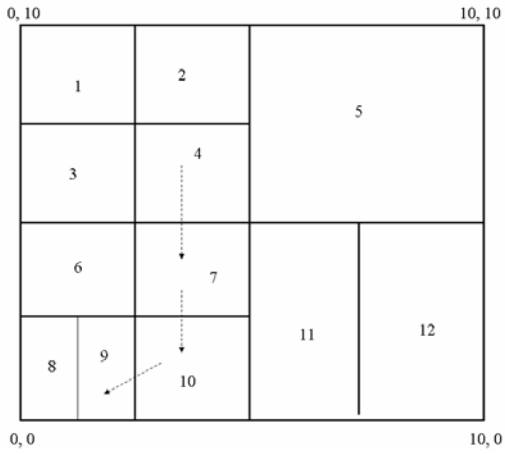
Δρομολόγηση

Η **δρομολόγηση** πραγματοποιείται ακολουθώντας την **ευθεία γραμμή** από την πηγή στον προορισμό, διαμέσου του χώρου συντεταγμένων. Στην Εικόνα 8 βλέπουμε ένα παράδειγμα δρομολόγησης από τον κόμβο 4 στον 9.
Κάθε κόμβος διατηρεί έναν **πίνακα δρομολόγησης**, όπου κρατά τις IP διευθύνσεις και τις συντεταγμένες των άμεσων γειτόνων του στο χώρο. Σε ένα d-διάστατο χώρο, δύο κόμβοι **γειτνιάζουν** αν οι συντεταγμένες τους συμπίπτουν εν μέρει σε d-1 διαστάσεις και συνορεύουν σε μία διάσταση.
Τα μηνύματα που κινούνται στο CAN περιέχουν τις συντεταγμένες του προορισμού τους. Έτσι, κάθε φορά που ένα μήνυμα φτάνει σε έναν κόμβο, προωθείται προς τον προορισμό του μέσω του γειτονικού με τις κοντινότερες συντεταγμένες στις συντεταγμένες του προορισμού του.

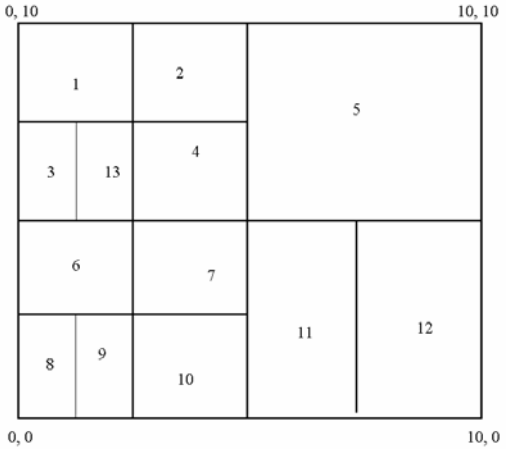
Σε ένα d-διάστατο χώρο που χωρίζεται σε n ζώνες, το μέσο μήκος του μονοπατιού δρομολόγησης είναι $\mathbf{(d/4)(n^{1/d})}$ και κάθε κόμβος έχει **2d γείτονες**. Με άλλα λόγια, το μέσο μήκος του μονοπατιού είναι $\mathbf{O(n^{1/d})}$.
Παρατηρούμε ότι μεταξύ δύο σημείων στο χώρο υπάρχουν πολλά **διαφορετικά μονοπάτια**. Επομένως, ακόμη και αν ένας από τους γείτονες ενός κόμβου αποτύχει, η δρομολόγηση μπορεί να συνεχιστεί με τον επόμενο διαθέσιμο κόμβο από τον πίνακα δρομολόγησης.
Αν , όμως, ένας κόμβος χάσει όλους τους γείτονές του και η συνέπεια του συστήματος δεν έχει αποκατασταθεί, η δρομολόγηση που περιγράφηκε μπορεί να αποτύχει. Στην περίπτωση αυτή, ο συγκεκριμένος κόμβος μπορεί να εκτελέσει μια αναζήτηση επεκτεινόμενου δακτυλίου για να εντοπίσει έναν άλλο κόμβο, κοντινότερο στον προορισμό και, κατόπιν, να αποστείλει σε αυτόν την ερώτηση. Από εκεί η δρομολόγηση συνεχίζεται με τον τρόπο που περιγράφηκε.

Κατασκευή του CAN

Όταν φτάνει ένας νέος κόμβος στο σύστημα, πρέπει να του ανατεθεί μια ζώνη. Αυτό επιτυγχάνεται με τη βοήθεια ενός άλλου κόμβου, ο οποίος χωρίζει τη δική του ζώνη στα δύο, κρατάει τη μισή και παραδίδει την υπόλοιπη στο νέο κόμβο. Η διαδικασία αυτή γίνεται σε τρία βήματα



Εικόνα 8 Παράδειγμα δρομολόγησης από τον κόμβο 4 στον 9.



Εικόνα 9 Ο χώρος συντεταγμένων μετάτην είσοδο του κόμβου 13.

Εκκίνηση

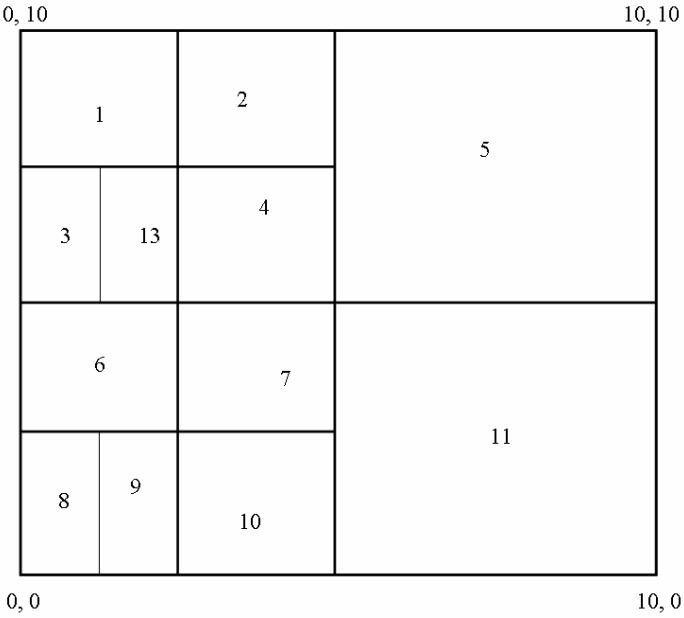
Ένας κόμβος που επιθυμεί να εισέλθει στο σύστημα πρέπει να γνωρίζει την IP διεύθυνση ενός ήδη συνδεδεμένου στο σύστημα κόμβου. Αυτό είναι το πρώτο βήμα για την είσοδο.

Εύρεση της ζώνης

Ο νέος κόμβος επιλέγει ένα τυχαίο σημείο P στο χώρο και αποστέλλει ένα μήνυμα συμμετοχής με προορισμό αυτό. Το μήνυμα δρομολογείται όπως περιγράφηκε παραπάνω, από τον πρωταρχικά γνωστό κόμβο στον κόμβο του οποίου η ζώνη περιέχει το P. Στη συνέχεια, ο κάτοχος της ζώνης αυτής τη χωρίζει σε δύο ίσα τμήματα και αναθέτει ένα από αυτά στο νέο κόμβο. Η διάσταση στην οποία θα γίνει ο διαχωρισμός επιλέγεται βάσει κατάταξης των διαστάσεων που επιτρέπει τη συγχώνευση ζωνών όταν κάποιοι κόμβοι αποτύχουν ή αποχωρήσουν οικειοθελώς. Τα ζεύγη (κλειδί, αξία) που αντιστοιχούν στη νέα ζώνη ανατίθενται στο νέο κόμβο.

Ενημέρωση των δομών δρομολόγησης

Ο νέος κόμβος μαθαίνει τους γείτονές του από τον προηγούμενο κάτοχο της ζώνης. Αυτός και οι γείτονές του με τη σειρά τους ενημερώνουν τους δικούς τους πίνακες μέσω περιοδικών μηνυμάτων. Σημειώνεται ότι η προσθήκη νέων κόμβων επηρεάζει μόνο **Ο(αριθμός διαστάσεων)** υπάρχοντες κόμβους.
Στην Εικόνα 9 βλέπουμε πώς αλλάζει ο χώρος συντεταγμένων της Εικόνας 8 με την είσοδο του κόμβου 13 και τη διαίρεση της ζώνης του 3 στα δύο.



Εικόνα 10 Αποχώρηση κόμβου και συγχώνευση ζωνών.

Αποχώρηση κόμβων

Κατά την οικιοθελή αποχώρηση, η ζώνη και τα ζεύγη (κλειδί, αξία) της ζώνης του αποχωρήσαντος κόμβου παραδίδονται σε έναν από τους γείτονες του κόμβου που αναχώρησε. Αν είναι δυνατή η **συγχώνευση**, τότε πραγματοποιείται, αλλιώς η ζώνη ανατίθεται προσωρινά στο γειτονικό κόμβο που κατέχει τη **μικρότερη ζώνη**. Στην Εικόνα 10 βλέπουμε να αποχωρεί ο κόμβος 12 και η ζώνη που κατείχε πριν να συνενώνεται με τη ζώνη του κόμβου 11.
Κατά την αποτυχία, που γίνεται αντιληπτή μέσω περιοδικών μηνυμάτων, η χωρίς κάτοχο ζώνη ανατίθεται στο γειτονικό κόμβος με τη μικρότερη ζώνη.
Φυσικά, όπως προκύπτει, είναι δυνατόν ο χώρος συντεταγμένων να κατακερματιστεί σε περισσότερες μικρότερες ζώνες και ένας κόμβος να αποκτήσει πλέον τον έλεγχο δύο ζωνών. Για να αποφευχθεί μεγαλύτερος κατακερματισμός του χώρου, στο παρασκήνιο τρέχει ένας αλγόριθμος επανάθεσης ζωνών για να κατέχει κάθε κόμβος μία και μόνο ζώνη.

5.2.3 Βελτιώσεις στο σχεδιασμό

Έχουν προταθεί και βελτιώσεις στο σχεδιασμό, όπως: αύξηση του πλήθους των διαστάσεων του χώρου συντεταγμένων, χρήση πολλαπλών χώρων, της τοπολογίας του δικτύου για τη διαμόρφωση του CAN, χρήση πολλαπλών συναρτήσεων κατακερματισμού ή τεχνικών κρυφής αποθήκευσης και δημιουργίας αντιγράφων, ανάθεση μιας ζώνης σε πολλούς κόμβους και ομοιόμορφος διαμοιρασμός του χώρου, κ.ά.
Οι βελτιώσεις αυτές μπορούν να οδηγήσουν σε μείωση του μήκους μονοπατιού ή της καθυστέρησης αναζήτησης, σε αύξηση της ανοχής σε αποτυχίες και της διαθεσιμότητας των δεδομένων και σε πιο ομοιόμορφο διαμοιρασμό του χώρου συντεταγμένων.

5.3 Δημιουργία αντιγράφων (Replication)

Όπως και στα αδόμητα συστήματα, έτσι και στα δομημένα, για να αυξήσουμε τη διαθεσιμότητα των δεδομένων και να έχουμε αποδοτικότερες αναζητήσεις μπορούμε να δημιουργήσουμε αντίγραφα της πληροφορίας για την ανεύρεση δεδομένων ή των ίδιων των δεδομένων σε περισσότερους κόμβους. Στη συνέχεια αναφερόμαστε σε τεχνικές δημιουργίας αντιγράφων στα συστήματα Chord και CAN.

Μέθοδοι δημιουργίας αντιγράφων

Chord

Για να εγγυηθούμε την επιτυχία των αναζητήσεων, οι κόμβοι μπορούν να διατηρούν λίστες με διαδοχικούς successors στο δακτύλιο (successor lists). Οι λίστες αυτές μπορούν να ενημερώνονται περιοδικά. Επίσης, μπορούν να δημιουργηθούν αντίγραφα των ίδιων των δεδομένων χρησιμοποιώντας διαφορετικές συναρτήσεις κατακερματισμού ή ακόμη και να διατηρούνται δεδομένα που πρόσφατα αναζητήθηκαν σε κρυφές μνήμες (cache).

CAN

Η χρήση κρυφής μνήμης και πολλαπλών συναρτήσεων κατακερματισμού χρησιμοποιούνται και στο CAN. Επίσης, η χρήση πολλαπλών πραγματικοτήτων μπορεί να επιτρέψει την αντιγραφή του πίνακα κατακερματισμού σε κάθε πραγματικότητα για να γίνει αποδοτικότερη η αναζήτηση. Το ίδιο μπορεί να γίνει και αν έχουμε υπερφόρτωση ζωνών. Τέλος, αν ένας κόμβος δέχεται πολλές ερωτήσεις για κάποιο δεδομένο, μπορεί να δημιουργήσει αντίγραφά του στους γείτονές του (*hot spot replication*) για να μειωθεί ο φόρτος και να έχουμε εξισορρόπηση φορτίου.

6 Ομαδοποίηση με βάση το περιεχόμενο

Ένας τρόπος για να γίνει αποδοτικότερη η αναζήτηση σε P2P συστήματα είναι να ομαδοποιήσουμε τους κόμβους βάσει των δεδομένων που διαθέτουν ή αυτών που ζητούν. Στη συνέχεια περιγράφονται τρεις τέτοιες προσπάθειες που έχουν γίνει.

6.1 Τοπικότητα βάσει ενδιαφερόντων (Interest-Based Locality)

Έχει παρατηρηθεί ότι όταν ένας κόμβος παρέχει ένα δεδομένο που ζητά κάποιος άλλος, τότε πιθανότατα διαθέτει και άλλα δεδομένα που ενδιαφέρουν τον κόμβο αυτό (**interest-based locality**). Αυτή την αρχή εκμεταλλευόμαστε δημιουργώντας επιπλέον συνδέσεις (**shortcuts**) [15] κόμβων πάνω από το σύστημα Gnutella, διατηρώντας ταυτόχρονα τον απλό και πλήρως αποκεντρικοποιημένο χαρακτήρα της.

6.1.1 Δημιουργία των shortcuts

Όταν ένας κόμβος εισέρχεται στο σύστημα, δε γνωρίζει τίποτα για τα ενδιαφέροντα των άλλων κόμβων και εντοπίζει το πρώτο δεδομένο που επιθυμεί μέσω πλημμύρας. Τότε επιλέγουμε τυχαία έναν ή περισσότερους από τους κόμβους που του παρέχουν το δεδομένο και τους τοποθετούμε σε μια 'shortcut-list'. Κατά τις επόμενες αναζητήσεις, ο κόμβος ρωτάει με τη σειρά όλους τους κόμβους της λίστας μέχρι να λάβει απάντηση. Μόνο αν δε βρει αυτό που αναζητά με τον παραπάνω τρόπο χρησιμοποιείται πλημμύρα, που οδηγεί στη δημιουργία νέων shortcuts.

6.1.2 Επιλογή και ενημέρωση των shortcuts

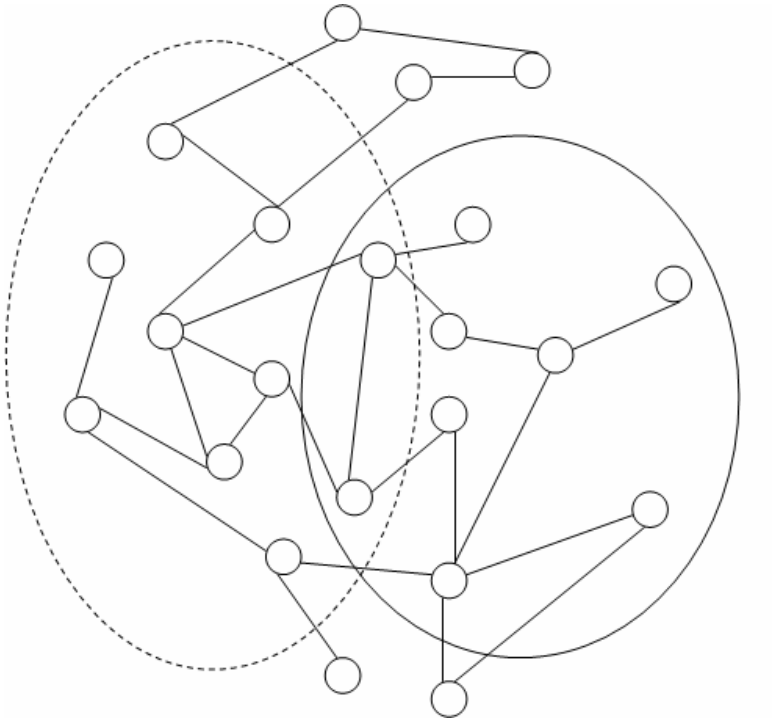
Η σειρά με την οποία χρησιμοποιούνται τα shortcuts της λίστας μπορεί να γίνει βάσει της πιθανότητας να διαθέτει ένας κόμβος δεδομένα ή του πλήθους αυτών, του latency του shortcut, του διαθέσιμου εύρους ζώνης και άλλων. Φυσικά, οι πληροφορίες ανανεώνονται κάθε φορά που ο κόμβος αποκτά ένα νέο δεδομένο. Επιπλέον, καθώς ο χώρος που αφιερώνεται στη διατήρηση πληροφοριών για τα shortcuts είναι περιορισμένος, τα εισάγουμε και τα εξάγουμε από τη λίστα βάσει της χρησιμότητάς τους.

6.2 Συσχετιστική Αναζήτηση (Associative Search)

Δύο ιδιαίτερα επιθυμητά χαρακτηριστικά στα P2P συστήματα είναι η ικανότητα να υποστηρίζουν την εύρεση λιγότερο δημοφιλών δεδομένων καθώς, επίσης, και partial-match ερωτήσεις, δηλαδή ερωτήσεις που περιέχουν λέξεις κλειδιά ή τυπογραφικά λάθη. Τις ιδιότητες αυτές έχουν τα συσχετιζόμενα δίκτυα [8] που περιγράφονται στη συνέχεια.

6.2.1 Συσχετιζόμενα δίκτυα (associative overlays)

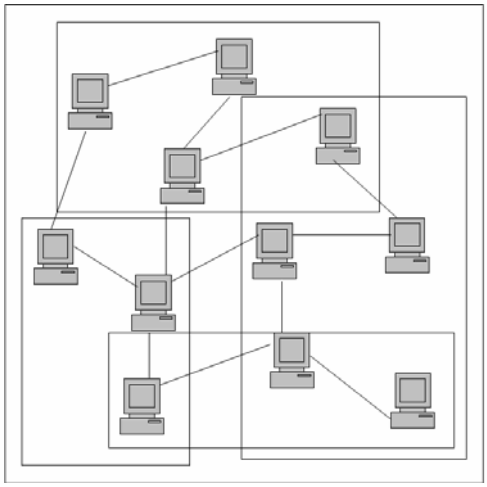
Στο σύστημα ορίζονται κανόνες (**guide rules**), οι οποίοι είναι σύνολα κόμβων που ικανοποιούν κάποιο κατηγόρημα., π.χ. διαθέτουν σημασιολογικά όμοια δεδομένα. Τα σύνολα δεν είναι απαραίτητα ξένα μεταξύ τους, επομένως ένας κόμβος μπορεί να ανήκει σε περισσότερους από έναν κανόνες. Στην Εικόνα 11 βλέπουμε δύο κανόνες.
Όταν ένας κόμβος αναζητά ένα δεδομένο, ρωτά αρχικά τους κόμβους που ανήκουν σε έναν από τους κανόνες στους οποίους ο ίδιος συμμετέχει. Αν η ερώτηση δεν ικανοποιηθεί, μπορεί να δοκιμαστεί και νέος κανόνας. Σημειώνεται ότι η αναζήτηση σε έναν κανόνα είναι



Εικόνα 11 Δύο κανόνες.

τυφλή, καθώς κάθε κόμβος διατηρεί μια λίστα για κάθε κανόνα στον οποίο ανήκει, με ορισμένους από τους άλλους κόμβους που συμμετέχουν στον ίδιο κανόνα, με αποτέλεσμα να μοιάζουν οι κανόνες με μικρά αδόμητα συστήματα

Για να συμμετάσχει ένας κόμβος σε έναν κανόνα αναζητά αρχικά τυφλά σε ολόκληρο το σύστημα τα δεδομένα που ο ίδιος κατέχει και ανακαλύπτει κανόνες με δημοφιλή σχετικά δεδομένα. Επόμενες αναζητήσεις τού δίνουν την ευκαιρία να συμμετάσχει και σε κανόνες με πιο σπάνια δεδομένα.



Εικόνα 12 Σχηματισμός SONs.

6.3 Semantic Overlay Networks (SONs)

Καθώς η προώθηση των ερωτήσεων σε όλους τους κόμβους του συστήματος (π.χ. μέσω πλημμύρας) δεν ευνοεί την κλιμάκωση, κρίνεται αποδοτικότερο να δρομολογούνται μόνο σε κόμβους που είναι πιο πιθανό να έχουν απαντήσεις. Η βασική ιδέα είναι να ομαδοποιούμε

τους κόμβους βάσει των δεδομένων που κατέχουν και να προωθούμε τις ερωτήσεις μόνο σε σχετικές ομάδες. Οι ομάδες καλούνται SONs [2] και μπορούν να επικαλύπτονται. Στην Εικόνα 12 βλέπουμε ένα παράδειγμα σχηματισμού τεσσάρων SONs. Παρατηρούμε ότι οι κόμβοι μπορούν να ανήκουν σε περισσότερα από ένα SONs.

6.3.1 Δημιουργία και χρήση των SONs

Αρχικά, προτείνεται μια ιεραρχία ταξινόμησης, που αποτελεί τη βάση του συστήματος. Ουσιαστικά, πρόκειται για ένα δένδρο εννοιών, που γνωστοποιείται σε κάθε νέο κόμβο. Για κάθε έννοια θα σχηματιστεί και ένα SON.
Τα δεδομένα κάθε νέου κόμβου κατηγοριοποιούνται σε ένα ή περισσότερα φύλλα του δένδρου. Για να είναι καλή η κατηγοριοποίηση πρέπει τα αρχεία κάθε κατηγορίας να ανήκουν σε μικρό αριθμό κόμβων, οι κόμβοι να έχουν αρχεία σε μικρό αριθμό κατηγοριών και η κατηγοριοποίηση να γίνεται με όσο το δυνατόν λιγότερα λάθη.
Ένας κόμβος συμμετέχει σε ένα SON ανάλογα με τα δεδομένα που κατέχει (η συντηρητική στρατηγική τον τοποθετεί σε κάποιο SON έστω κι αν κατέχει ένα μόνο δεδομένο της συγκεκριμένης έννοιας, ενώ η επιθετική μόνο αν κατέχει σημαντικό αριθμό δεδομένων της έννοιας).
Όταν ένας κόμβος δημιουργεί μια ερώτηση, την κατηγοριοποιεί και, στη συνέχεια, την αποστέλλει στα κατάλληλα SONs. Εκεί η προώθηση συνεχίζεται με κάποια μέθοδο (π.χ. πλημμύρα) μέχρι να ικανοποιηθεί.

6.3.2 Χρήση μεθόδων ανάκτησης πληροφοριών

Το σχήμα που περιγράφηκε παραπάνω μπορεί να χρησιμοποιηθεί για την υλοποίηση υπηρεσιών ανάκτησης πληροφοριών [5]. Πιο συγκεκριμένα, οι κόμβοι ενός SON σχηματίζουν μια μηχανή αναζήτησης (*pSearch engine*) και οι κόμβοι που επιθυμούν να τη χρησιμοποιήσουν επικοινωνούν με αυτούς. Το πλήθος κόμβων της μηχανής μεταβάλλεται ανάλογα με το φόρτο που έχει και η είσοδος και η έξοδος από αυτή πραγματοποιείται εύκολα.

Οργάνωση

Οι κόμβοι του συστήματος οργανώνονται σε ένα CAN, ενώ η συσχέτιση των δεδομένων γίνεται με τις τεχνικές *Vector Space Model* (VSM) και *Latent Semantic Indexing* (LSI). Η VSM τεχνική αναπαριστά δεδομένα και ερωτήσεις ως διανύσματα όρων. Κάθε στοιχείο τους αντιστοιχεί στη σπουδαιότητα που έχει ο όρος και υπολογίζεται βάσει στατιστικού σχήματος. Κατά την ανάκτηση πληροφοριών, τα δεδομένα ταξινομούνται βάσει της ομοιότητας των διανυσμάτων τους με το διάνυσμα της ερώτησης και επιστρέφονται εκείνα με τη μεγαλύτερη ομοιότητα. Η LSI μειώνει τη διάσταση των διανυσμάτων που έχουν προκύψει, δίνοντας *συσχετιστικά διανύσματα* (*semantic vectors*). Το σημαντικό είναι ότι δεδομένα που δε μοιράζονται κοινούς όρους μπορούν να συσχετιστούν.
Τα δεδομένα αποθηκεύονται σε CAN διάστασης όσης και η διάσταση των συσχετιστικών διανυσμάτων, βάσει αυτών. Οι ερωτήσεις μετατρέπονται, επίσης, σε συσχετιστικά διανύσματα και δρομολογούνται με το γνωστό τρόπο του CAN. Όταν καταλήξουν στον προορισμό τους, μεταφέρονται μέσω πλημμύρας σε γειτονικούς κόμβους στο χώρο για να επιστραφούν και δεδομένα ίσως λιγότερο σχετικά με αυτά που αναζητήθηκαν και να ικανοποιηθεί η ερώτηση.

7 Ερωτήσεις διαστήματος

Παρόλο που τα περισσότερα δομημένα συστήματα που χρησιμοποιούν Κατανεμημένους Πίνακες Κατακερματισμού οδηγούν σε εξισορρόπηση φορτίου και εγγυώνται λογαριθμικό πλήθος βημάτων κατά τη δρομολόγηση με σχετικά μικρού μεγέθους διατηρούμενη πληροφορία, δεν είναι δυνατή η υλοποίηση ερωτήσεων διαστήματος. Αυτό οφείλεται στο γεγονός ότι η τιμή κατακερματισμού ενός διαστήματος δε σχετίζεται με τις αντίστοιχες τιμές κατακερματισμού των ίδιων των τιμών εντός του διαστήματος. Στη συνέχεια περιγράφονται κάποια συστήματα που υποστηρίζουν την υλοποίηση ερωτήσεων διαστήματος.

7.1 Mercury

Πρόκειται για ένα πρωτόκολλο που επιτρέπει την πραγματοποίηση ερωτήσεων διαστήματος πολλαπλών γνωρισμάτων [4]. Κάθε ερώτηση είναι μια σύζευξη διαστημάτων κάποιων γνωρισμάτων. Ακολουθεί η περιγραφή του πρωτοκόλλου.

float x-συντεταγμένη = 50 float y-συντεταγμένη = 100 string Παίκτης = 'Γκάλης' string Ομάδα = 'Άρης' int Σκορ = 32	float x-συντεταγμένη > 34 float y-συντεταγμένη < 53 string Παίκτης = 'Γ*' int Σκορ = '*'

Εικόνα 13 Παράδειγμα δεδομένου και ερώτησης στο Mercury.

7.1.1 Μοντέλο δεδομένων και ερωτήσεων

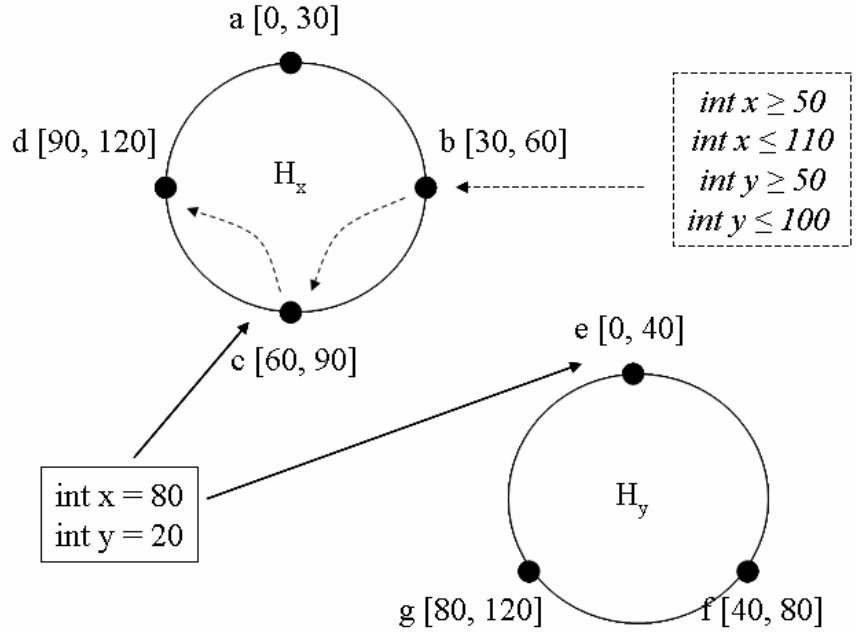
Ένα δεδομένο αναπαρίσταται από μια λίστα πεδίων της μορφής (***τύπος, γνώρισμα, τιμή***). Αναγνωρίζονται οι τύποι ακεραίων και πραγματικών αριθμών, χαρακτήρων και συμβολοσειρών.

Μια σύζευξη κατηγορημάτων της μορφής (***τύπος, γνώρισμα, τελεστής, τιμή***) αποτελεί μια ερώτηση. Υποστηρίζονται οι τελεστές <, >, ≤, ≥ και =, ενώ για τις συμβολοσειρές επιτρέπονται και οι τελεστές *πρόθεμα* και *επίθεμα*. Η σύζευξη επιτυγχάνεται με πολλαπλές ξεχωριστές ερωτήσεις. Στην Εικόνα 13, βλέπουμε ένα δεδομένο και μια ερώτηση. Το δεδομένο περιλαμβάνει τα γνωρίσματα *x-συντεταγμένη* και *y-συντεταγμένη* πραγματικού τύπου, τα γνωρίσματα *παίκτης* και *ομάδα* τύπου συμβολοσειράς και το γνώρισμα *σκορ* ακέραιου τύπου, με τιμές 50, 100, 'Γκάλης', 'Άρης' και 32, αντίστοιχα. Η ερώτηση αναζητά όλα τα δεδομένα των οποίων η τιμή της *x-συντεταγμένης* είναι μεταξύ 34 και 53, το γνώρισμα *παίκτης* ξεκινά από 'Γ', ενώ δε διευκρινίζεται η επιθυμητή τιμή για το σκορ.

7.1.2 Δρομολόγηση

Οι κόμβοι του συστήματος διαχωρίζονται σε ομάδες που καλούνται *κέντρα γνωρισμάτων* (*attribute hubs*). Τα κέντρα αυτά είναι λογικά και κάθε κόμβος μπορεί να ανήκει σε περισσότερα από ένα. Κάθε κέντρο γνωρίσματος είναι υπεύθυνο για ένα από τα γνωρίσματα που ορίζονται στο σύστημα. Οι κόμβοι κάθε κέντρου οργανώνονται σε ένα δακτύλιο και κάθε ένας από αυτούς είναι υπεύθυνος για ένα συνεχόμενο εύρος τιμών του γνωρίσματος. Κάθε κόμβος υπεύθυνος για κάποιο εύρος τιμών ικανοποιεί όλες τις ερωτήσεις που αφορούν

το συγκεκριμένο εύρος και διατηρεί όλα τα δεδομένα με τιμές σε αυτό το διάστημα. Η ανάθεση των διαστημάτων στους κόμβους γίνεται κατά την είσοδό τους στο σύστημα.



Εικόνα 14 Αποθήκευση δεδομένων και δρομολόγηση ερωτήσεων.

Αποθήκευση δεδομένων και δρομολόγηση ερωτήσεων

Μια ερώτηση μεταφέρεται μόνο σε ένα από τα κέντρα γνωρισμάτων του συστήματος, που σχετίζεται με κάποιο από τα γνωρίσματα που ορίζονται στην ερώτηση. Στο κέντρο αυτό, η ερώτηση μεταφέρεται σε όλους εκείνους τους κόμβους που πιθανώς διαθέτουν επιθυμητά αποτελέσματα.

Τα δεδομένα, τα οποία αποτελούνται από γνωρίσματα με συγκεκριμένες τιμές, αποθηκεύονται στους κατάλληλους κόμβους των αντίστοιχων κέντρων γνωρισμάτων, δηλαδή στους κόμβους που είναι υπεύθυνοι για τα κατάλληλα διαστήματα τιμών. Σημειώνεται ότι δεν είναι απαραίτητη η διατήρηση αντιγράφων των δεδομένων σε όλα τα κέντρα γνωρισμάτων που τους αντιστοιχούν. Αντίθετα, μπορεί ένας κόμβος σε ένα κέντρο να διατηρεί αυτούσιο κάποιο δεδομένο, ενώ οι κόμβοι άλλων κέντρων να διατηρούν ένα δείκτη στο δεδομένο αυτό.

Μέσα σε κάποιο κέντρο γνωρίσματος, μια ερώτηση κατευθύνεται στον πρώτο κόμβο του δακτυλίου που την ικανοποιεί, δηλαδή είναι υπεύθυνος για διάστημα που περιέχει κάποιο μέρος του εύρους τιμών που ορίζεται στην ερώτηση. Όσο ο επόμενος κόμβος στο δακτύλιο εξακολουθεί να καλύπτει άλλο μέρος του εύρους που αναζητείται, η ερώτηση προωθείται, αλλιώς σταματά. Στην Εικόνα 14, βλέπουμε ένα σύστημα όπου δημιουργούνται δύο κέντρα H_x και H_y, για τα γνωρίσματα x και y συντεταγμένων αντικειμένων. Και τα δυο γνωρίσματα έχουν τιμές μεταξύ 0 και 120. Στο κέντρο H_x συμμετέχουν οι κόμβοι a, b, c και d, ενώ στο H_y συμμετέχουν οι e, f και g. Στους κόμβους αυτούς ανατίθενται τα διαστήματα τιμών που φαίνονται στο σχήμα. Το δεδομένο <*int x = 80, int y = 20*> αποθηκεύεται και στα δύο κέντρα, αφού ορίζονται τιμές για τα γνωρίσματα που αφορούν και τα δυο, στους κόμβους c και e, οι οποίοι είναι υπεύθυνοι για τα διαστήματα [60, 90] και [0, 40], αντίστοιχα. Έστω ότι δημιουργείται η ερώτηση <*int x ≥ 50, int x ≤ 110, int y ≥ 50, int y ≤ 100*>. Έστω ότι μεταφέρεται αρχικά στον κόμβο b του κέντρου H_x. Από εκεί περνά και επεξεργάζεται από τους κόμβους c και d, καθένας από τους οποίους είναι υπεύθυνος για μέρος του εύρους που ορίζεται στην ερώτηση.

Σημειώνεται ότι για να είναι δυνατή η δρομολόγηση, κάθε κόμβος συνδέεται όχι μόνο με τον επόμενο και τον προηγούμενό του στο δακτύλιο κάθε κέντρου όπου συμμετέχει, αλλά διατηρεί και μια σύνδεση με όλα τα υπόλοιπα κέντρα του συστήματος.
Για να πραγματοποιείται αποδοτικότερα η δρομολόγηση μέσα στα κέντρα γνωρισμάτων, κάθε κόμβος διατηρεί k μακρινές ακμές, εκτός από τις συνδέσεις με τον προηγούμενο και τον επόμενό του στο δακτύλιο. Η δημιουργία τους γίνεται βάσει κάποιας κατανομής. Κατά τη δρομολόγηση μιας ερώτησης, λοιπόν, ένας κόμβος την προωθεί σε εκείνον από τους γείτονές του που είναι υπεύθυνος για διάστημα κοντινότερο στο διάστημα που αναζητά. Υποθέτοντας ότι τα διαστήματα για τα οποία είναι υπεύθυνοι οι κόμβοι ενός δακτυλίου είναι ομοιόμορφα, αποδεικνύεται ότι το αναμενόμενο πλήθος βημάτων που απαιτούνται για τη δρομολόγηση σε ένα δακτύλιο είναι $O((1/k) * \log^2 n)$, όπου n είναι το πλήθος των κόμβων του δακτυλίου. Επίσης, καθώς κατά τη δρομολόγηση πραγματοποιείται το πολύ ένα βήμα για τη μετάβαση από ένα κέντρο σε ένα άλλο, το συνολικό πλήθος βημάτων εξακολουθεί να είναι $O((1/k) * \log^2 n)$.

7.1.3 Συνδέσεις και αποχωρήσεις κόμβων

Συνδέσεις νέων κόμβων

Για να εισέλθει ένας νέος κόμβος στο σύστημα, πρέπει να γνωρίζει ένα ήδη συνδεδεμένο σε αυτό κόμβο. Στη συνέχεια, από τον κόμβο αυτό μαθαίνει ποια κέντρα υπάρχουν και έναν αντιπρόσωπο κάθε κέντρου. Κατόπιν, επιλέγει τυχαία το κέντρο στο οποίο θα εισέλθει, επικοινωνεί με τον κόμβο που γνωρίζει σε αυτό, έστω m, αναλαμβάνει το μισό διάστημα που κατέχει αυτός και γίνεται πλέον μέλος του κέντρου.
Έπειτα, μαθαίνει από τον m ποιοι είναι οι γείτονές του στο δακτύλιο (προηγούμενος και επόμενος στο δακτύλιο, μακρινές ακμές και συνδέσεις με άλλα κέντρα). Ο νέος κόμβος μπορεί να δημιουργήσει τις δικές του μακρινές ακμές σύμφωνα με μια κατανομή και νέες συνδέσεις σε άλλα κέντρα μέσω τυχαίων περιπάτων.

Αποχωρήσεις

Όταν ένας κόμβος αποχωρήσει από το σύστημα, οι συνδέσεις που διατηρούσε πρέπει να αποκατασταθούν. Έτσι, οι κόμβοι διατηρούν μια μικρή λίστα με κόμβους που ακολουθούν το διάδοχό τους στο δακτύλιο και τη χρησιμοποιούν αν καταστραφεί μια τέτοια σύνδεση.
Οι μακρινές ακμές ανακατασκευάζονται περιοδικά μόνο όταν το σύστημα έχει αλλάξει δραματικά.
Για την επιδιόρθωση καταστραμμένων συνδέσεων με άλλα κέντρα γνωρισμάτων, οι κόμβοι χρησιμοποιούν εφεδρικές συνδέσεις που διατηρούν. Αν και αυτές δεν είναι διαθέσιμες, μπορούν να χρησιμοποιηθούν οι αντίστοιχες συνδέσεις του προκατόχου ή του διαδόχου ή να ζητηθούν πληροφορίες από τον κόμβο μέσω του οποίου έγινε η αρχική σύνδεση στο σύστημα.

7.2 Πολυδιάστατες ερωτήσεις σε P2P συστήματα

Για την υλοποίηση τέτοιου είδους ερωτήσεων, αρχικά κατανέμουμε τα δεδομένα του συστήματος σε ένα σύνολο κόμβων, επιτρέποντας την εισαγωγή και την αποχώρηση κόμβων, καθώς, επίσης, και την αποθήκευση και διαγραφή δεδομένων. Έπειτα, μπορούμε να ορίσουμε μια κατάλληλη στρατηγική για να κατευθύνουμε τις ερωτήσεις στους κατάλληλους κόμβους. Στη συνέχεια περιγράφονται δύο τεχνικές που μπορούν να χρησιμοποιηθούν [17].

7.2.1 SCRAP

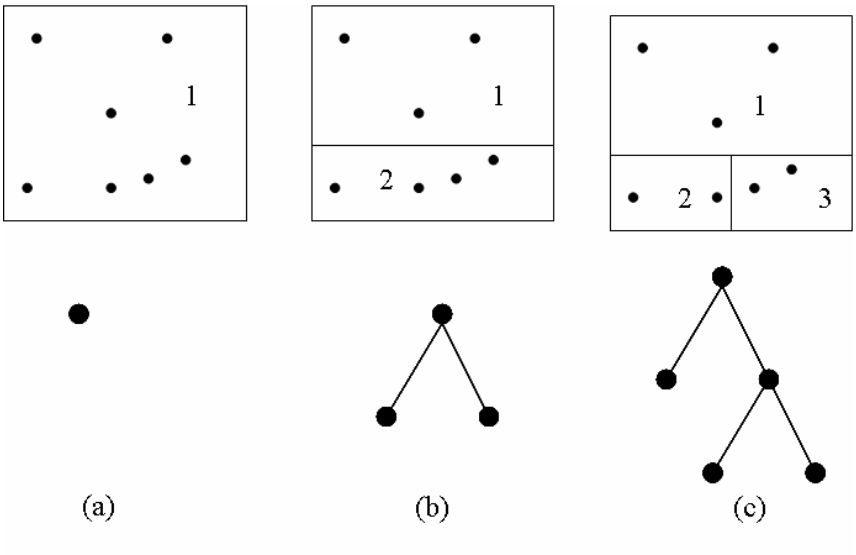
Στην προσέγγιση αυτή, αρχικά αναπαριστούμε τα πολυδιάστατα δεδομένα σε μία διάσταση, χρησιμοποιώντας π.χ. την τεχνική της z-κατάταξης. Έστω, λοιπόν, για παράδειγμα διδιάστατα δεδομένα που αποτελούνται από δύο γνωρίσματα τεσσάρων bits, X και Y. Ένα διδιάστατο δεδομένο <x, y> μετατρέπεται σε δεδομένο 8 bits εναλλάσσοντας τα bits των x και y γνωρισμάτων. Έτσι, το διδιάστατο δεδομένο <0100, 0101> αντιστοιχεί στο μονοδιάστατο δεδομένο <00110001>.
Κατόπιν, οι κόμβοι του συστήματος. αναλαμβάνουν τα δεδομένα κάποιου διαστήματος που έχει προκύψει μετά τη z-κατάταξη. Ο διαμοιρασμός των δεδομένων αλλάζει καθώς κόμβοι εισέρχονται και αποχωρούν. Πιο συγκεκριμένα, όταν ένας κόμβος εισέρχεται, αναλαμβάνει τη μισή ζώνη κάποιου ήδη υπάρχοντος κόμβου, ενώ όταν κάποιος αποχωρεί, ένας από τους γειτονικούς του στην κατάταξη αναλαμβάνει τη ζώνη του.

Δρομολόγηση

Για να ικανοποιηθεί μια πολυδιάστατη ερώτηση, πρέπει να αποσταλεί σε όλους τους κόμβους που κατέχουν σχετικά δεδομένα. Έτσι, αρχικά η πολυδιάστατη ερώτηση διαστήματος μετατρέπεται σε ένα σύνολο μονοδιάστατων ερωτήσεων, ενώ στη συνέχεια κάθε μία από αυτές αποστέλλεται στους κόμβους των οποίων οι ζώνες καλύπτουν τα διαστήματα που ορίζονται. Σημειώνεται ότι οι τεχνικές που χρησιμοποιούνται για τη μετατροπή της ερώτησης επιστρέφουν μικρά σύνολα ερωτήσεων, ίσως λιγότερο ακριβή από την αρχική ερώτηση.
Η δρομολόγηση στους κατάλληλους κόμβους πραγματοποιείται ένα δίκτυο δρομολόγησης, που καλείται *γράφος Παράλειψης* (*Skip graph*). Πρόκειται για μια κυκλικά συνδεδεμένη λίστα των κόμβων του συστήματος, διατεταγμένη σύμφωνα με τα διαστήματα τιμών τα οποία κατέχουν. Στη λίστα αυτή διατηρούνται επιπλέον δείκτες (skip pointers) για την αποδοτικότερη πραγματοποίηση της δρομολόγησης. Κάθε κόμβος διατηρεί O(logN) δείκτες σε εκθετικά αυξανόμενες αποστάσεις από τον ίδιο στη λίστα, όπου N είναι το πλήθος κόμβων της λίστας, οι οποίοι δημιουργούνται βάσει αλγορίθμου με κάποιες συνθήκες τυχαιότητας. Είναι προφανές ότι για τον εντοπισμό του κόμβου που κατέχει ένα δεδομένο απαιτούνται O(logN) μηνύματα, σε αντίθεση με τα O(N) μηνύματα που απαιτούνται σε μια διπλά συνδεδεμένη λίστα.

7.2.2 MURK

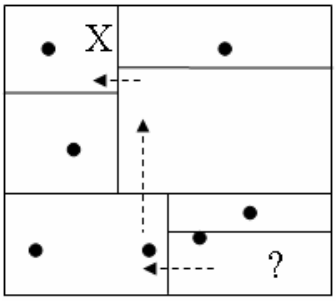
Σε αυτή την προσέγγιση, ο πολυδιάστατος χώρος των δεδομένων διαμοιράζεται σε ορθογώνια, ενώ κάθε κόμβος αναλαμβάνει ένα από αυτά. Ένας τρόπος για να το επιτύχουμε είναι χρησιμοποιώντας kd-δένδρα, στα οποία κάθε φύλλο αναπαριστά ένα ορθογώνιο του χώρου δεδομένων που βρίσκεται στην κατοχή ενός κόμβου. Για παράδειγμα, στην Εικόνα 15 βλέπουμε το διαμοιρασμό ενός διδιάστατου χώρου. Αρχικά, ένας μοναδικός κόμβος, ο 1, υπάρχει στο σύστημα και αναλαμβάνει όλο το χώρο, αντιστοιχώντας σε έναν κόμβο στο kd-δένδρο. Όταν φτάνει ο επόμενος κόμβος, ο 2, ο χώρος διαμοιράζεται σε δυο μέρη με ίσο φορτίο ως προς την πρώτη διάσταση και κάθε ένας αναλαμβάνει από ένα τμήμα. Αυτό οδηγεί σε διάσπαση του μοναδικού κόμβου του δένδρου στα δύο. Καθώς νέοι κόμβοι



Εικόνα 15 Ο διαμοιρασμός του διδιάστατου χώρου.

εισέρχονται, η διαδικασία συνεχίζεται με ανάλογο τρόπο, με διάσπαση υπαρχόντων φύλλων του δένδρου και αντίστοιχων τμημάτων του χώρου δεδομένων. Σημειώνεται ότι κατά το διαμοιρασμό χρησιμοποιούνται κυκλικά οι διαστάσεις.

Όταν ένας κόμβος αποχωρήσει, το τμήμα που κατείχε πρέπει να ανατεθεί σε κάποιον άλλο. Αν ο αδελφικός του κόμβος είναι φύλλο του δένδρου, όπως για παράδειγμα στην Εικόνα 15 (b) όπου ο κόμβος 2 αποχωρεί και ο αδελφικός του 1 είναι φύλλο του δένδρου, το τμήμα αναλαμβάνει ο κόμβος που απομένει, δηλαδή ο 1. Αν, όμως, ο αδελφικός του είναι εσωτερικός κόμβος του δένδρου, όπως για παράδειγμα στην Εικόνα 15 (c), όπου αποχωρεί ο 1, ένα φύλλο χαμηλότερου επιπέδου του αδελφικού του υποδένδρου, έστω ο 2, παραδίδει το τμήμα του στο δικό του αδελφό, τον 3, και αναλαμβάνει ο ίδιος το τμήμα του αποχωρήσαντος κόμβου.



Εικόνα 16 Δρομολόγηση στο MURK.

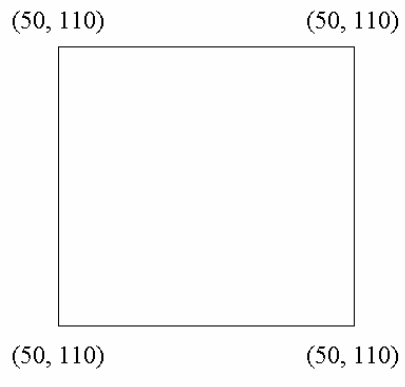
Δρομολόγηση

Για να είναι δυνατή η δρομολόγηση ερωτήσεων, δημιουργούμε συνδέσεις μεταξύ 'γειτονικών' κόμβων, δηλαδή κόμβων που γειτνιάζουν στο χώρο δεδομένων, όπως γίνεται για παράδειγμα στο CAN, με αποτέλεσμα να δημιουργείται ένα πλέγμα. Η δρομολόγηση πραγματοποιείται με τον ακόλουθο τρόπο: έστω ερώτηση που αναζητά δεδομένο που βρίσκεται στο ορθογώνιο Q. Ορίζουμε την απόσταση Manhattan ενός κόμβου n από το Q ως τη μικρότερη απόσταση οποιουδήποτε σημείου του χώρου που κατέχει ο n από οποιοδήποτε σημείο στο Q. Έτσι, ένας κόμβος προωθεί μια ερώτηση σε εκείνο το γειτονικό του που ελαχιστοποιεί την απόσταση Manhattan από το Q. Στην Εικόνα 16, φαίνεται το μονοπάτι που ακολουθεί μια ερώτηση από το σημείο ? στο σημείο X που αντιστοιχεί στο δεδομένο (50, 110). Όταν καταλήξει σε έναν κόμβο που κατέχει σχετικά δεδομένα, προωθείται σε όσους γείτονες κατέχουν επίσης σχετικά δεδομένα, μέχρι να βρεθούν όλες οι

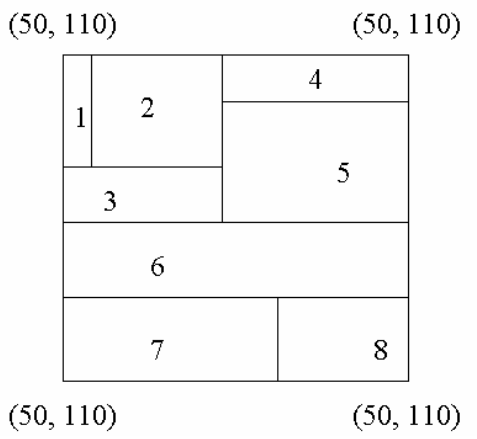
επιθυμητές απαντήσεις. Σημειώνεται ότι οι κόμβοι πρέπει να γνωρίζουν τα όρια του χώρου που κατέχουν οι γειτονικοί τους.

7.3 Διατήρηση αντιγράφων απαντήσεων σε ερωτήσεις διαστήματος

Στη συνέχεια θα περιγραφεί μια μέθοδος για την υλοποίηση ερωτήσεων διαστήματος που βασίζεται στο σύστημα του CAN. Οι απαντήσεις διατηρούνται στους κόμβους που τις ζητούν και χρησιμοποιούνται για την ικανοποίηση μελλοντικών ερωτήσεων [16].



Εικόνα 17 Ο εικονικός χώρος συντεταγμένων.



Εικόνα 18 Ο διαμοιρασμός του χώρου συντεταγμένων.

7.3.1 Μοντέλο συστήματος

Το σύστημα χρησιμοποιεί ένα 2d-διάστατο εικονικό χώρο συντεταγμένων, με τρόπο ανάλογο του CAN. Δεδομένου του πεδίου τιμών [a, b] ενός μονοδιάστατου γνωρίσματος, ο αντίστοιχος χώρος συντεταγμένων είναι το διδιάστατο τετράγωνο που ορίζεται από τις συντεταγμένες (a, a), (b, a), (b, b) και (a, b). Για παράδειγμα, στην Εικόνα 17 βλέπουμε τον εικονικό χώρο συντεταγμένων που αντιστοιχεί στο μονοδιάστατο γνώρισμα με εύρος τιμών [0, 120]. Οι γωνίες του χώρου είναι τα σημεία με συντεταγμένες (0, 0), (120, 0), (120, 120) και (0, 120).

Ο χώρος αυτός διαχωρίζεται σε ορθογώνιες περιοχές, που καλούνται *ζώνες*. Δεν υπάρχει επικάλυψη μεταξύ των ζωνών και καλύπτουν ολόκληρο τον εικονικό χώρο. Μια ζώνη ορίζεται από δύο σημεία $<(x_1, y_1), (x_2, y_2)>$, το κάτω αριστερό και το άνω δεξί. Στην Εικόνα 18 βλέπουμε τον εικονικό χώρο να έχει διαχωριστεί σε 8 ζώνες: ζώνη-1 <(0, 80), (10, 120)>, ζώνη-2 <(10, 80), (55, 120)>, ζώνη-3 <(0, 60), (55, 80)>, ζώνη-4 <(55, 100), (120, 120)>, ζώνη-5 <(55, 60), (120, 100)>, ζώνη-6 <(0, 30), (120, 60)>, ζώνη-7 <(0, 0), (80, 30)> και ζώνη-8 <(80, 0), (120, 30)>.

Κάθε ζώνη ανατίθεται σε έναν κόμβο του συστήματος. Δεν κατέχουν, όμως, όλοι οι κόμβοι απαραίτητα μια ζώνη. Οι κόμβοι που συμμετέχουν στο διαχωρισμό του χώρου και κατέχουν ζώνες ονομάζονται *ενεργοί* (*active*). Οι υπόλοιποι ονομάζονται *παθητικοί* (*passive*). Όλοι οι παθητικοί κόμβοι εγγράφονται στους ενεργούς και κάθε ενεργός κόμβος διατηρεί μια λίστα με τους παθητικούς οι οποίοι έχουν εγγραφεί στον ίδιο.

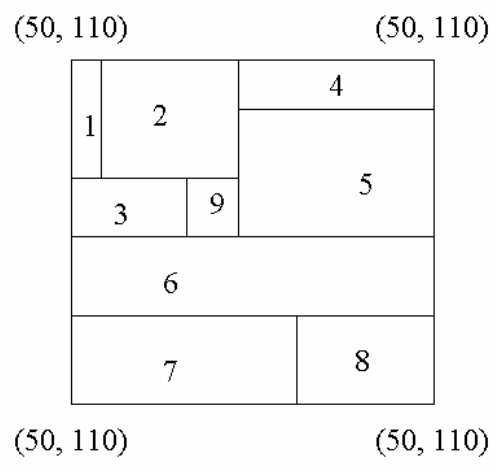
Για τη δρομολόγηση, κάθε ενεργός κόμβος διατηρεί έναν πίνακα με τις IP διευθύνσεις και τις συντεταγμένες ζωνών των γειτονικών τους στον εικονικό χώρο.

Μια ερώτηση διαστήματος εύρους $<q_s, q_e>$ αντιστοιχίζεται στο σημείο (q_s, q_e) του χώρου συντεταγμένων (*σημείο στόχος - target point*). Ο κόμβος στου οποίου τη ζώνη περιέχεται το

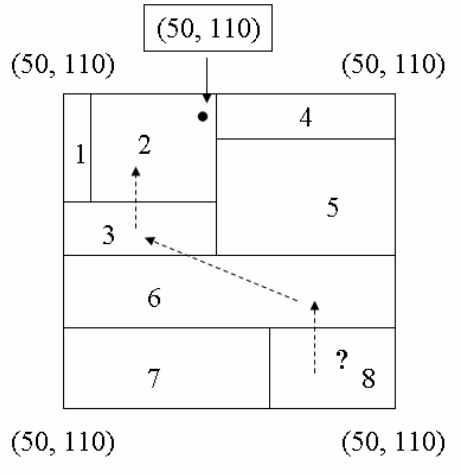
σημείο αυτό θα αποθηκεύσει το αποτέλεσμα της αναζήτησης. Και ο κόμβος, όμως, που έκανε την ερώτηση μπορεί να διατηρήσει ένα αντίγραφο της πληροφορίας.

7.3.2 Συντήρηση ζωνών

Αρχικά, ολόκληρος ο χώρος συντεταγμένων ανατίθεται στον μοναδικό ενεργό κόμβο (*data source*) του συστήματος. Με την πάροδο του χρόνου, ο διαμερισμός του χώρου αλλάζει δυναμικά καθώς ζώνες χωρίζονται και ανατίθενται σε νέους κόμβους.
Μία ζώνη διαιρείται είτε επειδή πρέπει να απαντά σε πάρα πολλές ερωτήσεις διαστήματος, είτε επειδή έχει μεγάλο φόρτο ερωτήσεων δρομολόγησης, αφού μεγάλες ζώνες είναι πιθανότερο να βρεθούν στο μονοπάτι ερωτήσεων. Στην πρώτη περίπτωση, η ζώνη χωρίζεται κατά τέτοιο τρόπο ώστε να κατανέμονται ομοιόμορφα οι αποθηκευμένες απαντήσεις, αλλά και ο χώρος της ζώνης, ενώ στη δεύτερη χωρίζεται ως προς τη μεγαλύτερη διάσταση για να μειωθεί ο φόρτος. Στην Εικόνα 19 βλέπουμε πώς αλλάζει ο χώρος της Εικόνας 18 όταν διαιρείται η ζώνη-4 παράλληλα στον y-άξονα και ένας νέος κόμβος αναλαμβάνει τη ζώνη-9.



Εικόνα 19 Είσοδος νέου κόμβου.



Εικόνα 20 Δρομολόγηση ερωτήσεων.

7.3.3 Δρομολόγηση ερωτήσεων

Όταν δημιουργείται μια ερώτηση διαστήματος, αρχικά αναζητούνται αποτελέσματα στο σημείο στόχο. Έτσι, η ερώτηση δρομολογείται από τον κόμβο που την εκκινεί περνώντας από γειτονικές ζώνες μέχρι να καταλήξει στο στόχο. Η ερώτηση προωθείται από κάθε κόμβο σε εκείνον το γειτονικό που κατέχει τη ζώνη με τις κοντινότερες συντεταγμένες στο στόχο. Στην Εικόνα 20, εκκινείται η ερώτηση (50, 110) στη ζώνη 8. Από εκεί προωθείται μέσω της 6 και της 3 στη 2, όπου και ικανοποιείται. Σημειώνεται πως ερωτήσεις γίνονται και από τους παθητικούς κόμβους.
Αποδεικνύεται πως το μέσο μήκος μονοπατιού αναζήτησης σε ένα ομοιόμορφα διαμοιρασμένο χώρο είναι $O(\sqrt{n})$, όπου N είναι το πλήθος των ζωνών του συστήματος.
Αν δε βρεθούν αποτελέσματα στο στόχο, η ερώτηση προωθείται πάνω και αριστερά στους γειτονικούς, οι οποίοι πιθανώς να διατηρούν αποτελέσματα από προηγούμενες ερωτήσεις.
Σημειώνεται ότι η δρομολόγηση μπορεί να γίνει αποδοτικότερη αν ρωτιέται κάθε κόμβος στο μονοπάτι προς το στόχο για διατηρημένα αποτελέσματα, αν οι κόμβοι κατά την είσοδό τους στο σύστημα εκτελούν κάποιες ερωτήσεις για να αποκτήσουν αντίγραφα ορισμένων δεδομένων, οπότε και θα απαντούν μελλοντικές ερωτήσεις άμεσα ή αν χρησιμοποιηθούν οι βελτιώσεις που αναφέρθηκαν στο σχεδιασμό του CAN.

Τέλος, το σχήμα που προτείνεται μπορεί να γενικευτεί σε ερωτήσεις διαστήματος δεδομένων n διάστασης χρησιμοποιώντας χώρο συντεταγμένων 2n διαστάσεων με ανάλογο τρόπο.

7.4 Κατανεμημένη επεξεργασία ερωτήσεων και κατάλογοι

Στη συνέχεια περιγράφουμε πώς μπορούμε να δημιουργήσουμε κατανεμημένους καταλόγους με κάποια ιεραρχία για να πραγματοποιείται αποδοτικά η δεικτοδότηση δεδομένων και η δρομολόγηση ερωτήσεων [21].

7.4.1 Μεταλλαγμένες ερωτήσεις

Έστω ότι έχουμε ένα κατανεμημένο σύστημα στο οποίο τα δεδομένα περιγράφονται σε XML. Οι ερωτήσεις που πραγματοποιούνται βρίσκουν πιθανώς απαντήσεις σε διάφορους κόμβους του συστήματος. Απαιτείται, λοιπόν, ένας κατανεμημένος μηχανισμός δρομολόγησης για να εντοπίζονται οι κόμβοι που κατέχουν τα δεδομένα που αναζητούνται.
Ένα *πλάνο μεταλλαγμένης ερώτησης* (*mutant query plan, MQP*) είναι ένας γράφος μιας αλγεβρικής ερώτησης σε XML, που πιθανώς περιέχει αυτούσια δεδομένα σε XML, αναφορές σε τοποθεσίες πηγών πληροφοριών (URLs) και ασαφή ονόματα πηγών πληροφοριών (URNs). Σε κάθε MQP ορίζεται ένας στόχος: μια δικτυακή διεύθυνση όπου θα αποσταλεί το αποτέλεσμα όταν θα ικανοποιηθεί το MQP. Ένα MQP ξεκινά από έναν κόμβο και προωθείται σε άλλους συγκεντρώνοντας μέρος της απάντησης, μέχρι να ικανοποιηθεί πλήρως από κάποιο δεδομένο σε XML μορφή και να επιστραφεί η απάντηση στον κόμβο από τον οποίο ξεκίνησε.
Κάθε κόμβος που προωθεί ένα MQP μπορεί να το *μεταλλάξει* αναλύοντας ένα URN σε ένα ή περισσότερα URLs ή ένα URL σε κάποιο δεδομένο. Επίσης, μπορεί να μειώσει το MQP ικανοποιώντας μέρος του πλάνου με παροχή μέρους της απάντησης που αναζητείται.

7.4.2 Κατανεμημένοι κατάλογοι

Για την ανάλυση των URNs σε URLs χρησιμοποιούνται κατάλογοι που διατηρούνται τοπικά στους κόμβους. Η οργάνωση των δεδομένων σε καταλόγους μπορεί να γίνει βάσει των γνωρισμάτων τους, για παράδειγμα βάσει της τοποθεσίας τους στο δίκτυο. Η κατηγοριοποίηση μπορεί να είναι απλή ή ιεραρχική. Στη δεύτερη περίπτωση, δημιουργούμε πολύ-ιεραρχικά πεδία ονομάτων. Για παράδειγμα, μια πολυθρόνα μπορεί να κατηγοριοποιηθεί ως 'Επιπλα/Καρέκλες' (δύο επίπεδα κατηγοριοποίησης. Με ανάλογο τρόπο μπορούν να κατηγοριοποιηθούν και οι κόμβοι διαδραματίζοντας διαφορετικό ρόλο στο σύστημα για να διευκολύνουν τη δρομολόγηση.
Στο σύστημα που περιγράφεται, η δρομολόγηση πραγματοποιείται κατά τέτοιο τρόπο ώστε οι κόμβοι να μπορούν να ικανοποιούν μερικώς μια ερώτηση βάσει τοπικά διατηρούμενων πληροφοριών, χωρίς να έχουν συνολική γνώση του συστήματος.

8 Συμπεράσματα

Στην εργασία αυτή παρουσιάσαμε ορισμένα βασικά P2P συστήματα, δίνοντας έμφαση στις μεθόδους αποθήκευσης της πληροφορίας και αναζήτησης που χρησιμοποιούνται. Στον Πίνακα 1 βλέπουμε τα κύρια στοιχεία των συστημάτων αυτών. Παρατηρούμε ότι στα αδόμητα συστήματα, όπου υπάρχει χαλαρή σύνδεση των κόμβων και διατηρούνται λίγες πληροφορίες για άλλους κόμβους (Gnutella) ή καμία πληροφορία (Napster), δεν υπάρχει καμία εγγύηση για τις αναζητήσεις, ούτε ως προς το πλήθος βημάτων που απαιτούνται, ούτε ως προς την επιτυχία τους, είτε υπάρχουν στο σύστημα τα δεδομένα που αναζητούμε, είτε όχι. Η εύρεση μη δημοφιλών δεδομένων γίνεται δύσκολα, όμως, δυνατή είναι η πραγματοποίηση ερωτήσεων διαστημάτων και ερωτήσεων που βασίζονται σε λέξεις κλειδιά. Από την άλλη πλευρά, τα δομημένα συστήματα απαιτούν τη δημιουργία ενός αυστηρά δομημένου ιδεατού δικτύου, όπου κάθε κόμβος τοποθετείται με συγκεκριμένο τρόπο και γνωρίζει κάποιους 'γειτονικούς'. Συνήθως, οι απαιτήσεις σε μνήμη για τη διατήρηση πληροφοριών δρομολόγησης είναι περιορισμένες. Τα συστήματα αυτά εγγυώνται αποδοτικές αναζητήσεις ακόμη κι αν αλλάζει η σύστασή τους δυναμικά, ενώ και η εύρεση μη δημοφιλών δεδομένων γίνεται εύκολα. Δύσκολη, όμως, είναι η πραγματοποίηση ερωτήσεων διαστήματος. Για το σκοπό αυτό έχουν προταθεί πολλά συστήματα που εκμεταλλεύονται τα θετικά στοιχεία των δομημένων συστημάτων, ενώ καθιστούν δυνατή και την υλοποίηση ερωτήσεων διαστήματος χρησιμοποιώντας διάφορες τεχνικές.

	Αρχιτεκτονική	**Hashing**	**Πληροφορίες**	**Δρομολόγηση**	**Απόδοση**
Napster	Κεντρικοποιημένο, αδόμητο	Όχι	Κεντρικός κατάλογος	Napster web site	
Gnutella	Μη κεντρικο-Ποιημένο, αδόμητο	Όχι	Άμεσοι γείτονες	Πλημμύρα	
Chord	Μη κεντρικο-Ποιημένο, δομημένο	Ναι	Finger table O(logn)	Διάδοχος κόμβος	O(logn)
CAN	Μη κεντρικο-Ποιημένο, δομημένο	Ναι	Άμεσοι γείτονες στο χώρο O(d)	Ζώνη με το κλειδί	O(d(n1/d))

Πίνακας 1 Βασικά στοιχεία Δομημένων και Αδόμητων Συστημάτων Ομότιμων Κόμβων.

Αναφερθήκαμε, επίσης, σε τεχνικές δημιουργίας και συντήρησης αντιγράφων δεδομένων, καθώς η χρήση τους μπορεί να αυξήσει σημαντικά τη διαθεσιμότητα των δεδομένων, επομένως και την απόδοση, αλλά και την ανοχή σε σφάλματα. Φυσικά, εξακολουθούν να αναζητούνται πιο αποδοτικές μέθοδοι για τη συντήρηση των αντιγράφων, έτσι ώστε να παρέχονται οι σωστές πληροφορίες στους χρήστες σε ένα σύστημα που αλλάζει δυναμικά με την πάροδο του χρόνου.

Κατόπιν, παρουσιάσαμε κάποιες προσπάθειες που γίνονται για να βελτιωθεί η απόδοση των συστημάτων κατευθύνοντας τις αναζητήσεις σε κόμβους που πιστεύεται ότι πιθανώς διαθέτουν τις ζητούμενες πληροφορίες, εκμεταλλευόμενοι συσχετισμούς δεδομένων και στοιχεία για ερωτήσεις που έχουν πραγματοποιηθεί στο παρελθόν.

Η εξέλιξη των P2P εφαρμογών ήταν αναμφισβήτητα σημαντική. Φαίνεται, όμως, πως το μέλλον διαγράφεται ακόμη πιο λαμπρό. Καθώς επιτεύγματα και άλλων τομέων των

επιστήμης αρχίζουν να δίνουν λύσεις σε προβλήματα που μέχρι τώρα ήταν δυσεπίλυτα, τα P2P συστήματα γίνονται συνεχώς αποδοτικότερα και μας προετοιμάζουν πως πρόκειται να παίξουν πολύ σημαντικό ρόλο στο μέλλον της επιστήμης και της τεχνολογίας.

9 Αναφορές

[1] A. Crespo and H. Garcia-Molina Routing Indexes Proc. of the Intrenational Confrence on Distributes Systems, (ICDCS) 2002

[2] A. Crespo and H. Garcia-Molina Semantic Overlay Networks for P2P Systems

[3] A. Datta, M. Hauswirth, K. Aberer Updates in Highly Unreliable, Replicated Peer-to-Peer Systems, The 23rd International Conference on Distributed Computing Systems, May 19-22, 2003, Providence, Rhode Island, USA.

[4] A. R. Bharambe, M. Agrawal, S. Seshan: Mercury: Supporting Scalable Multi-Attribute Range Queries. SIGCOMM 2004: 353-366

[5] C. Tang, Z. Xu and S. Dwarkadas, Peer-to-Peer Information Retrieval Using Self-Organizing Semantic Overlay Networks, SIGCOMM 03

[6] D. Tsoumakos, N. Roussopoulos: A Comparison of Peer-to-Peer Search Methods. WebDB 2003: 61-66

[7] E. Adar, B. Huberman. Free Riding on Gnutella, *First Monday*, October 2000.

[8] E. Cohen, A. Fiat, H. Kaplan Associative Search in Peer to Peer Networks: Harnessing Latent Semantics Infocom 03

[9] E. Cohen, S. Shenker, Replication Strategies in Unstructured Peer-to-Peer Networks, *SIGCOMM 02*

[10] H. Balakrishnan, M. Frans Kaashoek, D. R. Karger, R. Morris, I. Stoica, Looking up Data in P2P Systems. CACM 46(2): 43-48 (2003)

[11] http://freenetproject.org/ *Freenet*

[12] http://gnutella.wego.com. *Gnutella*

[13] http://napster.com *Napster*

[14] I. Stoica, R. Morris, D. Karger, M. Frans Kaashoek, and H. Balakrishnan, *Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications*, ACM SIGCOMM 2001, San Diego, CA, August 2001, pp. 149-160.

[15] K. Sripanidkulchai, B. Maggs, and H. Zhang, Efficient Content Location Using Interest-Based Locality in Peer-to-Peer Systems. Infocom 03

[16] O. D. Sahin, A. Gupta, D. Agrawal, A. E. Abbadi: A Peer-to-Peer Framework for Caching Range Queries. ICDE 2004: 165-176

[17] P. Ganesan, B. Yang, H. Garcia-Molina: One Torus to Rule Them All: Multidimensional Queries in P2P Systems. WebDB 2004: 19-24

[18] Q. L. Pei Cao, E. Cohen, K. Li, Scott Shenker Search and Replication in Unstructured Peer-to-peer Networks, International Conference on Supercomputing: 84-95, 2002

[19] S. Ratnasamy. A Scalable Content-Addressable Network. PhD. *Thesis, University of California, Berkeley, Fall 2002*

[20] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. *A Scalable Content-Addressable Network.* In Proc. ACM SIGCOMM 2001, San Diego, CA, August 2001

[21] V. Papadimos, D. Maier, K. Tufte: Distributed Query Processing and Catalogs for Peer-to-Peer Systems. CIDR 2003