ΠΑΝΕΠΙΣΤΗΜΙΟ ΙΩΑΝΝΙΝΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

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

ASSIGNMENT 9:

Peer-to-peer Systems

Due: June 17, 2005

Διδάσκουσα: Ε. Πιτουρά

ΜΑΡΓΑΡΙΤΗ ΣΠΥΡΙΔΟΥΛΑ

ΠΕΡΙΕΧΟΜΕΝΑ

Peer - to – Peer Systems

Περίληψη

Κατανεμημένα συστήματα που στηρίζονται στην ομότιμη και εθελοντική συμπεριφορά των κόμβων που τα απαρτίζουν έγιναν ευρέως γνωστά ως peer-to-peer συστήματα. Το δίκτυο που σχηματίζουν είναι χτισμένο πάνω στην υπάρχουσα υποδομή του διαδικτύου και παρέχουν κοινή χρήση πόρων (αποθηκευτικό χώρο, υπολογιστική ισχύς, κλπ.) ή διαμοιράζουν δεδομένα. Η μη απαίτηση για κεντρικό εξυπηρέτη, η δυνατότητα απευθείας επικοινωνίας μεταξύ των κόμβων (υπολογιστών που μετέχουν), η scalability, η αυτοδιοργάνωση, η αυτονομία και η ανωνυμία και δυναμικότητα (peers join and leave) είναι χαρακτηριστικά που κάνουν τα συστήματα αυτά δημοφιλή, και ελκυστικά ακόμη και στον εμπορικό κόσμο.

Στην παρούσα εργασία το ενδιαφέρον εστιάζεται στον τρόπο οργάνωσης των peer-to-peer συστημάτων, τις αρχιτεκτονικές που βασίζουν την δόμησή τους, τους μηχανισμούς για εντοπισμό και ανάκτηση της επιθυμητής πληροφορίας και στις στρατηγικές για δημιουργία και διατήρηση αντιγράφων έτσι ώστε τα συστήματα να είναι αποδοτικότερα.

1. Εισαγωγή

Τα τελευταία χρόνια με την ραγδαία εξάπλωση του Internet μια ακόμη ανάγκη αναφύεται: Η ανάγκη για κοινή χρήση υπολογιστικών πόρων (αποθήκευση, υπολογιστική ισχύ, κλπ.). Έτσι συγκροτούνται δίκτυα από πολλούς υπολογιστές με σκοπό την ανταλλαγή αρχείων, συνήθως μουσικά κομμάτια (Napster, Gnutella), ή για ανταλλαγή εγγράφων (Freenet), ή για κατανεμημένο υπολογισμό (seti@home) ή και για παροχή κατανεμημένων υπηρεσιών. Τέτοια δίκτυα είναι γνωστά ως peer-to-peer (P2P) δίκτυα.

Σε ένα peer-to-peer δίκτυο κάθε κόμβος που μετέχει είναι ισότιμος με κάθε άλλο και μπορεί να ενεργήσει είτε σαν πελάτης (client) είτε σαν εξυπηρέτης (server). Κινητήρια δύναμη για ανάπτυξη εφαρμογών peer-to-peer αποτελεί η αποκεντρικοποιημένη και κατανεμημένη δομή τέτοιων συστημάτων που δεν απαιτούν διαχείριση και συντήρηση, οικονομικές αξιώσεις ή άλλους νομικούς περιορισμούς. Οι κόμβοι προσαρμόζονται, αυτοδιοργανώνονται καθώς εισέρχονται ή αποχωρούν από το σύστημα, ικανοποιώντας την ιδιότητα της κλιμάκωσης και της ανοχής στις αποτυχίες[1, 2]. Οι λειτουργίες του είναι κατανεμημένες στους κόμβους που μετέχουν σε ένα τέτοιο σύστημα, όπου εκατομμύρια διαφορετικοί χρήστες μπορούν να είναι παρόντες ταυτόχρονα.

Ακολουθεί ο ορισμός ενός peer-to-peer συστήματος όπως δίνεται από τους Androutsellis-Theotokis, Spinellis [1]:

> *"Peer-to-peer συστήματα είναι κατανεμημένα συστήματα που αποτελούνται από διασυνδεδεμένους κόμβους, ικανούς να αυτοδιοργανώνονται σε τοπολογίες δικτύου με σκοπό την κοινή χρήση πόρων όπως περιεχόμενα, κύκλους μηχανής, χώρο αποθήκευσης, και εύρος, ικανά να προσαρμόζονται στις αποτυχίες και στις παροδικές μετακινήσεις κόμβων ενώ διατηρούν προσβάσιμη συνδετικότητα και*

εκτελούνται χωρίς την απαίτηση για μεσολάβηση ή υποστήριξη ενός καθολικού κεντρικού εξυπηρέτη ή αρχής".

Οι πρώτες κατανεμημένες εφαρμογές (SMTP, FTP) εμφανίζονται στα τέλη της δεκαετίας του '80, αρχές '90 και αποτελούν τον πρόδρομο των peer to peer συστημάτων [1].

Στο τέλος της δεκαετίας του '90 το internet είναι «κοινός τόπος» και η αναπτυσσόμενη τεχνολογία επιτρέπει την ανάδυση εφαρμογών μέσω των οποίων οι χρήστες ανταλλάσσουν αρχεία. Το Napster [3, 18] είναι η πρώτη χαρακτηριζόμενη peer-to-peer εφαρμογή, όπου κεντρικοί διακομιστές διατηρούν ευρετήρια για το που βρίσκονται τα αρχεία που ο χρήστης αναζητά, και τα οποία μπορεί να κατεβάσει (download) απευθείας από την θέση που βρίσκονται αποθηκευμένα. Μετεξέλιξη του Napster αποτελεί η Gnutella [3] όπου οι χρήστες τώρα συνδέονται μεταξύ τους για την εύρεση των επιθυμητών αρχείων.

Πολλές άλλες peer-to-peer εφαρμογές εμφανίζονται την ίδια περίοδο είτε για την ανταλλαγή μουσικών αρχείων είτε ακόμη και για την αξιοποίηση των χαμένων κύκλων της CPU (sofi@home). Τόσο η επιστημονική κοινότητα όσο και ο εμπορικός κόσμος δείχνουν έντονο ενδιαφέρον για τα peer-to-peer συστήματα τα οποία όχι μόνο γίνονται αποδεκτά αλλά υιοθετούνται ευρέως διαμοιράζοντας αρχεία ή παρέχοντας κατανεμημένο υπολογισμό.

Στο υπόλοιπο αυτής της εργασίας, θα γίνει λόγος για την πιο βασική λειτουργία των peer-to-peer συστημάτων, την *διαχείριση ερωτήσεων, και* τις αρχιτεκτονικές που προτάσσονται για το χειρισμό των λειτουργιών αναζήτησης, δρομολόγησης, εντοπισμού πληροφορίας, που υποβάλλονται στο peer-to-peer σύστημα. Η δεύτερη ενότητα αναφέρεται στην αναζήτηση γενικά και η τρίτη ενότητα στις αρχιτεκτονικές αυτών των συστημάτων. Στην τέταρτη ενότητα, γίνεται παρουσίαση μεθόδων που χρησιμοποιούνται για την επεξεργασία ερωτήσεων - αναζήτηση πληροφορίας. Ένα άλλο σημαντικό θέμα που μελετάται στην πέμπτη ενότητα είναι που, πόσα και ποια αντίγραφα θα δημιουργηθούν και θα ενημερωθούν-διατηρηθούν. Στην συνέχεια διατυπώνονται απόψεις που αφορούν την ασφάλεια, (Security, privacy, anonymity, trust**)** και τέλος αναφέρονται μερικά συμπεράσματα για τα peer-to-peer συστήματα και τις εφαρμογές τους.

2. Αναζήτηση

Τα peer-to-peer συστήματα είναι από την φύση τους κατανεμημένα και η γενική τους χρήση είναι ο διαμοιρασμός αρχείων (file sharing). Το κλειδί για την χρησιμότητά τους, αλλά και μια κύρια πρόκληση από σχεδιαστική άποψη είναι η τεχνική που χρησιμοποιείται για την *αναζήτηση* και *ανάκτηση* των επιθυμητών δεδομένων. Το πρόβλημα εστιάζει:

- ***στον εντοπισμό των δεδομένων***: ο μηχανισμός που χρησιμοποιείται για το εντοπισμό του επιθυμητού δεδομένου (ή αντικειμένου)

- ***στην δρομολόγηση της ερώτησης:*** η στρατηγική που καθορίζει σε πόσους και ποιους γείτονες θα σταλεί μια ερώτηση που φθάνει σε έναν κόμβο και δεν μπορεί να απαντηθεί από τον ίδιο.

Η αποτελεσματικότητα της τεχνικής αναζήτησης για ένα συγκεκριμένο σύστημα εξαρτάται από τις ανάγκες της εφαρμογής. Παραδοσιακά οι ερωτήσεις που υποστηρίζονται βασίζονται σε ένα αναγνωριστικό *(ID)* ή σε μια λέξη *κλειδί* ή σε μια *κανονική έκφραση*. Πρόσφατα έρευνες προτάσσουν τεχνικές για υποστήριξη πολύπλοκων ερωτήσεων που αφορούν *ομάδα δεδομένων (range queries) ή* περισσότερα από ένα γνωρίσματα *(multi-attribute queries).*

Η στρατηγική που χρησιμοποιείται για τον εντοπισμό και την ανάκτηση της πληροφορίας είναι κρίσιμος παράγοντας στα peer-to-peer συστήματα αφού επηρεάζει την αποτελεσματικότητα, την ικανότητα κλιμάκωσης, την ανοχή και την προσαρμοστικότητα σε αποτυχίες, την αυτό-διατήρηση (nodes join and leave) και εξαρτάται από την τοπολογία του overlay δικτύου, την δόμησή του και την αρχιτεκτονική του. Σε επόμενη ενότητα θα γίνει εκτενής αναφορά σε μεθόδους που εφαρμόζονται για τον χειρισμό ερωτήσεων.

3. Αρχιτεκτονική

Οι κόμβοι, (πρόκειται για προσωπικούς υπολογιστές, σταθμούς εργασίας, κλπ.) που μετέχουν σε ένα peer-to-peer σύστημα σχηματίζουν ένα δίκτυο επικάλυψης (*overlay network*) πάνω από την υπάρχουσα υποδομή του διαδικτύου. Διασυνδέονται, επικοινωνούν και ανταλλάσσουν πληροφορίες μεταξύ τους σε τοπολογίες ανεξάρτητα από το δίκτυο υποδομής (IP network) διατηρώντας την αυτονομία τους[4]. Η αρχιτεκτονική του δικτύου επηρεάζει τον μηχανισμό δρομολόγησης μηνυμάτων αναζήτησης, την απόδοση, την ικανότητα κλιμάκωσης, την προσαρμοστικότητα - ανοχή σε σφάλματα, κλπ. και στοχεύει στην υποστήριξη λειτουργιών όπως διαμοιρασμό αρχείων (*file sharing*), κατανεμημένο υπολογισμό (*distributed computing*), επικοινωνία – συνεργασία μεταξύ των χρηστών (*collaboration network*)

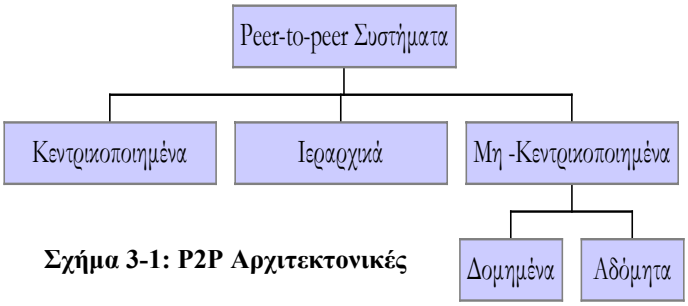
Υπάρχουν διάφορες αρχιτεκτονικές για τον σχηματισμό του overlay δικτύου[1,8]:

3.1. Κεντρικοποιημένα peer-to-peer συστήματα

Στις κεντρικοποιημένες αρχιτεκτονικές υπάρχει ένας κεντρικός εξυπηρέτης (Directory Server) στον οποίο απευθύνουν οι κόμβοι τις ερωτήσεις τους για να πληροφορηθούν που βρίσκονται οι επιθυμητές πληροφορίες (π.χ Napster). Μια τέτοια αρχιτεκτονική αν και είναι αρκετά αποδοτική, δεν έχει την ιδιότητα της κλιμάκωσης ενώ έχει ενιαίο σημείο της αποτυχίας (bottleneck).

3.2. Ιεραρχικά

Οι κόμβοι οργανώνονται σε ιεραρχική δομή όπως γίνεται με τους DNS στο διαδίκτυο. Στα ιεραρχικά peer-to-peer συστήματα εισάγεται ή έννοια



Σχήμα 3-1: P2P Αρχιτεκτονικές

των “super-peers” (FastTrack). Η δομή τους μπορεί να είναι κεντρικοποιημένη ή μη.

3.3. Μη Κεντρικοποιημένα peer-to-peer συστήματα

Μια άλλη κατηγορία αρχιτεκτονικών είναι οι μη – κεντρικοποιημένες όπου οι κόμβοι συγκροτούν το overlay δίκτυο είτε *δομημένα* ακολουθώντας κανόνες για τον σχηματισμό του δικτύου, είτε *αδόμητα* όπου δεν υπάρχει ούτε κεντρικό directory ούτε ακριβείς οδηγίες για τον σχηματισμό τοπολογίας του δικτύου και την τοποθέτηση των περιεχομένων.

> ***Δομημένα:*** Στα δομημένα peer-to-peer συστήματα οι κόμβοι οργανώνονται σε δομημένο γράφο για το σχηματισμό του overlay δικτύου. Στα δεδομένα αντιστοιχίζεται ένα κλειδί και η τοποθέτηση τους στους κόμβους γίνεται με προκαθορισμένο τρόπο έτσι ώστε να διευκολύνεται η αναζήτησή τους και να επιτυγχάνεται η κλιμάκωση. Η τοποθέτηση των αρχείων [8] στα *χαλαρά δομημένα* συστήματα (Freenet) βασίζεται στην εκτίμηση (on hints) για το που μπορεί να βρεθεί η αναζητούμενη πληροφορία. Στα *αυστηρά δομημένα* συστήματα τόσο η δόμηση του overlay δικτύου όσο και η τοποθέτηση των αρχείων είναι σαφώς καθορισμένη.
>
> Ο εντοπισμός ενός αντικειμένου (δεδομένου) από μια εφαρμογή στα δομημένα συστήματα γίνεται σε μικρό αριθμό βημάτων (network hops), υπό την απαίτηση βέβαια να διατηρείται ένας μικρός πίνακας δρομολόγησης σε κάθε κόμβο.
>
> Παραδείγματα τέτοιων συστημάτων αποτελούν τα: Content Addressable Network (CAN) [5], Chord [6], Tapestry [15], Pastry [14], Kademlia [16] και Viceroy [17].
>
> ***Αδόμητα:*** Στα συστήματα αυτά δεν υπάρχει καμιά δομή στο overlay δίκτυο και τα περιεχόμενα τοποθετούνται σε κόμβους στο δίκτυο χωρίς γνώση της τοπολογίας ή άλλης συσχέτισης με αυτό. Τα μη δομημένα συστήματα είναι κατάλληλα σε περιπτώσεις όπου μεγάλο πλήθος κόμβων μετέχει παροδικά στο δίκτυο χωρίς όμως αποδοτικούς μηχανισμούς αναζήτησης, κλιμάκωσης, διαθεσιμότητας. Υποστηρίζουν καλύτερα πολύπλοκες ερωτήσεις σε σχέση με τα δομημένα [13].
>
> Αδόμητα peer-to-peer δίκτυα είναι: Napster [18], Gnutella [22], FastTrack[19] KaZaA [20], BitTorrent [21], κ.α.

Στη συνέχεια περιγράφονται ενδεικτικά μερικά ευρέως γνωστά peer-to-peer συστήματα: το Napster, η Gnutella, το CAN και το Chord. Τα δυο πρώτα αντιπροσωπεύουν τα αδόμητα συστήματα ενώ τα δύο επόμενα τα δομημένα συστήματα.

3.4. Napster

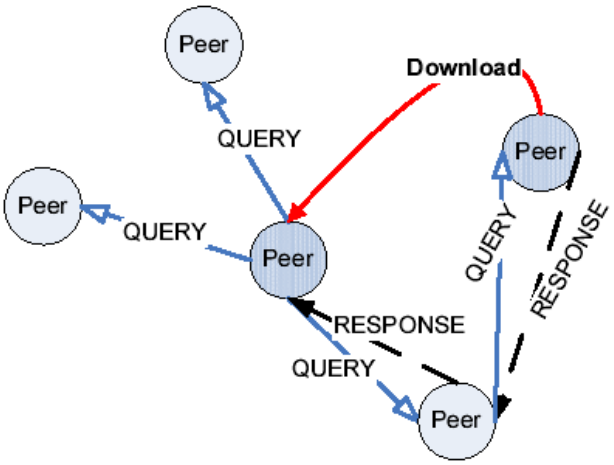
Το Napster έκανε την εμφάνισής του το 1999 και είναι το πιο γνωστό peer-to-peer σύστημα που χρησιμοποιήθηκε για ανταλλαγή μουσικών αρχείων. Η φιλοσοφία της λειτουργίας του ήταν απλή. Σε έναν κεντρικό διακομιστή φιλοξενούνταν μια βάση δεδομένων με καταχωρίσεις ευρετηρίου για την θέση αποθήκευσης των μουσικών αρχείων. Ο χρήστης απεύθυνε εκεί την ερώτηση για το αρχείο που επιθυμούσε, ο

διακομιστής του απαντούσε που θα βρει αντίγραφο και στη συνέχεια, ο χρήστης το κατέβαζε απευθείας από τον κόμβο που διατηρούσε αντίγραφο. Νομικοί περιορισμοί επέβαλαν το σταμάτημα λειτουργίας του Napster (2000), όμως το πνεύμα του διαχέεται στα μετέπειτα peer-to-peer συστήματα.

3.5. Gnutella

Η Gnutella είναι ένα από τα πιο δημοφιλή peer-to-peer δίκτυα για διαμοιρασμό αρχείων (file sharing). Η λειτουργία του στηρίζεται σε ένα μικρό πρόγραμμα (μόλις 100K) που οι κόμβοι εγκαθιστούν για να έχουν την δυνατότητα δικτύωσης και που ουσιαστικά αποτελεί το πρωτόκολλο βάση του οποίου γίνεται η ανταλλαγή των δεδομένων από κόμβο σε κόμβο (από PC σε PC).

Η λειτουργία του Gnutella έχει ως εξής: Το δίκτυο απαρτίζεται από κόμβους (χρήστες) που έχουν εγκαταστήσει το λογισμικό του client και συνδέονται μεταξύ τους χωρίς συγκεκριμένη δομή. Αρχικά, για να συνδεθεί κανείς στο δίκτυο, αρκεί να εντοπίσει έναν κόμβο που συμμετέχει ήδη στο Gnutella. Εκείνος τον ενημερώνει με μια λίστα (την δική του) από άλλους κόμβους που μετέχουν στο δίκτυο. Ο κόμβος στέλνει τις ερωτήσεις του για αναζήτηση σε όλους τους ενεργά συνδεδεμένους κόμβους – συνήθως μέχρι πέντε – με εκείνον (flooding). Η ερώτηση εφόσον δεν απαντηθεί, προωθείται σε γειτονικούς κόμβους. Αν το επιθυμητό αρχείο εντοπισθεί σε περισσότερους κόμβους, τότε ο αιτών κόμβος μπορεί να κατεβάσει το αρχείο σε τμήματα από διαφορετικούς κόμβους, απευθείας[22]. Όταν ο κόμβος αποσυνδέεται η λίστα των κόμβων που ήταν ενεργά συνδεδεμένοι σε αυτόν αποθηκεύεται τοπικά για μελλοντική χρήση.



Σχήμα 3-2: Η λειτουργία της Gnutella

Η Gnutella λειτουργεί σαν ένα πρωτόκολλο ερωτήσεων. Για να το πετύχει αυτό χρησιμοποιεί πακέτα μηνυμάτων πέντε διαφορετικών τύπων (έκδοση 0.4).

- ping: για τον εντοπισμό κόμβων στο δίκτυο
- pong: απάντηση στο μήνυμα ping
- query: αναζήτηση αρχείου
- query hit: απάντηση στο ερώτημα της αναζήτησης
- push: κατέβασμα (download) του αιτούμενου αρχείου

Για τον περιορισμό των μηνυμάτων που προκύπτουν από την δρομολόγηση των ερωτήσεων με την μέθοδο της πλημμύρας και τον τερματισμό της αναζήτησης χρησιμοποιείται ένα πεδίο στο μήνυμα που στέλνεται, το πεδίο Time – To – Live

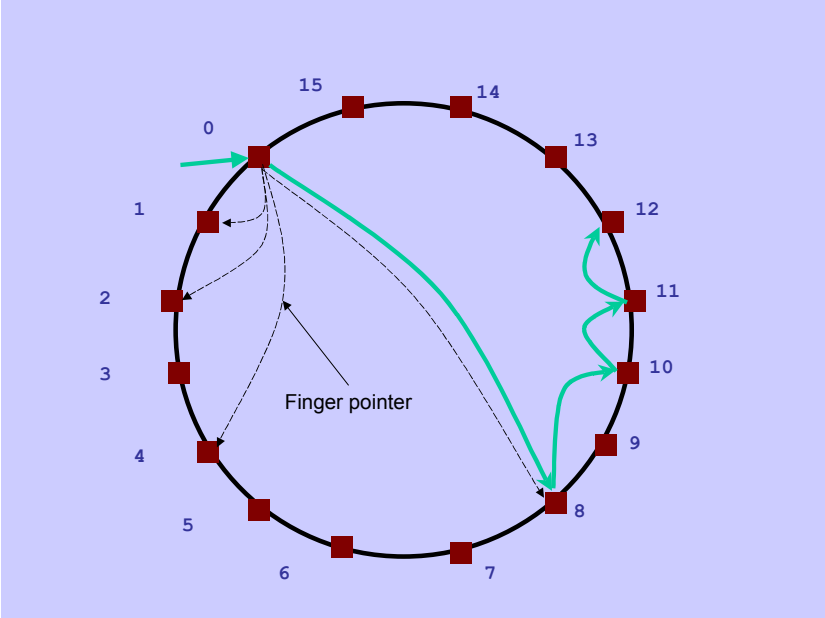
(TTL). Κάθε φορά που στέλνεται ένα μήνυμα από έναν κόμβο η τιμή του πεδίου μειώνεται κατά ένα. Ένας κόμβος που λαμβάνει την ερώτηση την προωθεί αν το πεδίο TTL έχει τιμή μεγαλύτερη του 0 και δεν έχει ξαναδεί το μήνυμα.

Ο μηχανισμός δρομολόγησης δημιουργεί υπερφόρτωση του δικτύου με μηνύματα και σπαταλά τους δικτυακούς πόρους.

3.6. Το σύστημα Chord

Το Chord [6] είναι ένα κατανεμημένο πρωτόκολλο για τον εντοπισμό δεδομένων σε ένα peer-to-peer σύστημα που αναπτύχθηκε από ομάδα του MIT και παρουσιάσθηκε το 2001 (Sigcomm conference). Βασίζεται στη λειτουργία: δεδομένου ενός κλειδιού, αυτό αντιστοιχίζεται σε έναν κόμβο, όπου αποθηκεύεται το ζεύγος κλειδί/τιμή. Πρόκειται για ένα σύστημα απλό στην κατασκευή του, ανεκτικό στις αλλαγές του peer-to-peer δικτύου και εγγυάται την εύρεση των δεδομένων σε χρόνο O(logN) όπου N το πλήθος των κόμβων στο δίκτυο.

Το Chord μπορεί να λειτουργήσει σε ένα δυναμικό περιβάλλον όπου οι κόμβοι εισέρχονται και αποχωρούν από το σύστημα αυθαίρετα με την απαίτηση όμως κάθε κόμβος να αποθηκεύει τμήμα της πληροφορία για επιτυχή δρομολόγηση.



Σχήμα 3-3: Το σύστημα Chord όπου φαίνονται τα finger του κόμβου 0

Τόσο οι κόμβοι όσο και τα δεδομένα που έχουν μοναδικό m-bit (160 bits) αναγνωριστικό (ID) οργανώνονται σε έναν εικονικό δακτύλιο. Το ID του κόμβου κατακερματίζεται από την IP του, και το ID του αντικειμένου (δεδομένο) κατακερματίζεται από το όνομά του και έτσι προκύπτει η θέση τους πάνω στο δακτύλιο (δακτύλιος των 0 έως 2^m-1 θέσεων). Οι κόμβοι κατέχουν πληροφορία για τον προηγούμενο και τον επόμενο κόμβο στο δακτύλιο. Ο κάθε κόμβος είναι υπεύθυνος για τα αντικείμενα που είναι μεταξύ του προηγούμενου κόμβου και του ίδιου.

Οι βασικές λειτουργίες που μπορούν να πραγματοποιηθούν σε ένα τέτοιο σύστημα είναι η εισαγωγή νέου κόμβου (*join*), η αποθήκευση και ανάκτηση δεδομένου (*store & retrieve)* και η αποχώρηση ενός κόμβου (*leave*).

*Εισαγωγή κόμβου (Join)**:*** Όταν ένας κόμβος *n* επιθυμεί να εισέλθει στο σύστημα κατακερματίζει την IP διεύθυνσή του για να προκύψει το αναγνωριστικό του και με κάποια διαδικασία (εξωτερικό μηχανισμό) μαθαίνει το αναγνωριστικό ενός κόμβου *n'* που ήδη ανήκει στο Chord. Ο νέος κόμβος χρησιμοποιεί τον *n'* κόμβο για να προσθέσει τον εαυτό του στο δίκτυο Chord και να αρχικοποιήσει την κατάσταση του που περιλαμβάνει και την μεταφορά κλειδιών από τον επόμενό του κόμβο, αν αυτά αντιστοιχίζονται τώρα σε αυτόν (ο κόμβος *n* είναι τώρα ο επόμενος τους). Η τελευταία διαδικασία απαιτεί το πολύ O(1/N) μετακινήσεις κλειδιών.

Αποχώρηση κόμβου (leave): Όταν ένας κόμβος αποχωρεί από το σύστημα, τότε τα κλειδιά που είχε στην ευθύνη του αντιστοιχίζονται στον επόμενό του κόμβο. Η διαδικασία της ενημέρωσης καθώς οι κόμβοι έρχονται και φεύγουν από το σύστημα απαιτεί $O(\log^2 N)$ μηνύματα.

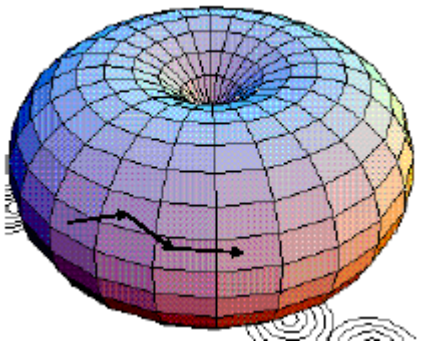
Εισαγωγή Δεδομένων: Ο κόμβος που επιθυμεί να αποθηκεύσει ένα αντικείμενο στο σύστημα εφαρμόζει μια συνάρτηση κατακερματισμού στο όνομα του αντικείμένου και προκύπτει το αναγνωριστικό του. Το νέο αντικείμενο αντιστοιχίζεται (και αποθηκεύεται) στον κόμβο που έχει ID ίσο με το ID του, αν υπάρχει ή του αμέσως επόμενου αν δεν υπάρχει.

Για την ***ανάκτηση*** του αντικειμένου χρησιμοποιείται παρόμοια διαδικασία. Εφαρμόζεται η συνάρτηση κατακερματισμού και προκύπτει η θέση του αντικειμένου και δρομολογείται η ανάκτησή του.

Η μόνη πληροφορία που είναι απαραίτητη στο σύστημα Chord για επιτυχή δρομολόγηση είναι η γνώση του επομένου. Όμως ένα τέτοιο σχήμα δεν διατηρεί την ιδιότητα της κλιμάκωσης. Για το λόγο αυτό κάθε κόμβος διατηρεί ένα τμήμα πληροφορίας για τους άλλους κόμβους βελτιώνοντας έτσι την απόδοση του αλγορίθμου. Η πληροφορία που διατηρεί ο κάθε κόμβος είναι ένας πίνακας m εγγραφών, finger table, και περιέχει τους κόμβους που βρίσκονται σε απόσταση 2^0, 2^1, 2^2, , …, 2^{m-1} από αυτόν. Οι ερωτήσεις τώρα δρομολογούνται μέσω του finger table και η αναζήτηση ολοκληρώνεται σε O(logN) το πολύ hops.

3.7. Το σύστημα CAN

Το σύστημα CAN (Content Addressable Network) [5] είναι ένα ακόμη δομημένο σύστημα που ο σχηματισμός του overlay δικτύου βασίζεται σε κατανεμημένους πίνακες κατακερματισμού. Το CAN αναπτύχθηκε από ομάδα του πανεπιστημίου του Berkeley και παρουσιάσθηκε το 2001 (Sigcomm conference).

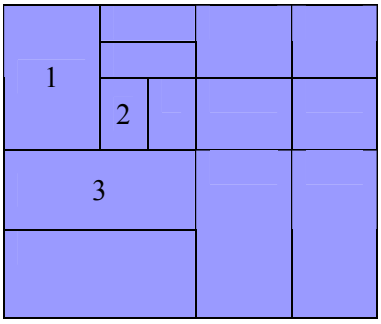


Σχήμα 3-4:Ο d-διάστατος χώρος CAN, *Πηγή 2002, K. Aberer, EPFL-SSC, Laboratoire*

Η σχεδίασή του είναι επίσης απλή και κατανοητή. Βασίζεται στον εικονικό

καρτεσιανό χώρο d-διαστάσεων. Για κάθε διάσταση υπάρχει μια συνάρτηση κατακερματισμού. Ο χώρος διαμερίζεται και αντιστοιχίζεται στους κόμβους που μετέχουν στο σύστημα. Το τμήμα του χώρου που κατέχει ένας κόμβος καλείται *zone*. Τα αντικείμενα (δεδομένα) αντιστοιχίζονται σε σημεία στο χώρο από μια συνάρτηση κατακερματισμού και αποθηκεύονται στους κόμβους που κατέχουν το zone που ανήκουν τα σημεία, ως ζεύγη κλειδί/τιμή (K, V).

Οι κόμβοι αποθηκεύουν τμήμα του κατανεμημένου πίνακα κατακερματισμού και πληροφορία για τους άμεσα γείτονές τους. Δύο κόμβοι θεωρούνται γείτονες αν οι d-1 διαστάσεις τους εκτεινόμενες συμπίπτουν και εφάπτονται στην άλλη διάσταση.



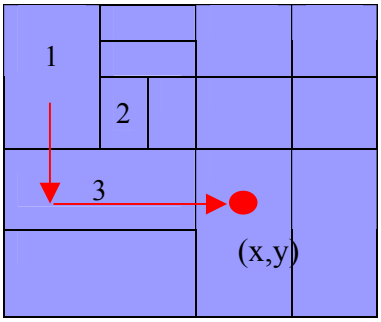
Σχήμα 3-5: CAN σε χώρο 2 διαστάσεων

Δρομολόγηση: Ένα CAN μήνυμα περιλαμβάνει τις συντεταγμένες προορισμού. Έτσι δρομολογείται από έναν "greedy" αλγόριθμο προς τον κόμβο που βρίσκεται πιο κοντά στον προορισμό του. Το μέσο μήκος μονοπατιού για ένα χώρο *d* διαστάσεων με *n* ίσες zones είναι $(d/4)(n^{1/d})$ hops, η δε πληροφορία για τους γείτονες που διατηρεί κάθε κόμβος είναι *2d*.

Εισαγωγή κόμβου (Join): Η σύνδεση ενός νέου κόμβου γίνεται μέσω της διαδικασίας bootstrap. Ο νεοεισερχόμενος κόμβος εντοπίζει έναν κόμβο που ανήκει ήδη στο σύστημα CAN. Με την χρήση του μηχανισμού δρομολόγησης, επιλέγει ένα τυχαίο σημείο P στο χώρο και στέλνει ένα μήνυμα συνένωσης στον κόμβο που καλύπτει το zone όπου ανήκει το P και ο οποίος μπορεί να διαμεριστεί. Ο νέος κόμβος αναλαμβάνει το μισό zone ενώ το άλλο μισό το αναλαμβάνει ο προκάτοχος του. Οι γείτονες κόμβοι ενημερώνονται έτσι ώστε να συμπεριλάβουν και τον νέο κόμβο.

Αποχώρηση κόμβου: Όταν ένας κόμβος αποχωρεί από το σύστημα τότε το zone που καταλαμβάνει συνενώνεται ή αναλαμβάνεται από κάποιον γείτονα – αν δεν είναι δυνατή η συνένωση - όπως επίσης και το τμήμα του πίνακα κατακερματισμού που διατηρούσε ο κόμβος που αποχώρησε.

Εισαγωγή αντικειμένου: Για την εισαγωγή ενός αντικειμένου (K, V) με κλειδί K και τιμή V στο σύστημα CAN έχει ως εξής: Ο κόμβος που επιθυμεί να εισάγει το αντικείμενο εφαρμόζει τις συναρτήσεις κατακερματισμού στο κλειδί K για την εύρεση των συντεταγμένων στο χώρο, σημείο P. Στη συνέχεια δρομολογείται στο σημείο αυτό και αποθηκεύεται το ζεύγος (K, V) στον κόμβο που κατέχει το zone.



Σχήμα 3-6: Δρομολόγηση στο CAN

Το σύστημα CAN έχει μηχανισμούς για την αποχώρηση κόμβου, την ανάκαμψη και την συντήρηση. Σε εθελούσια αποχώρηση ο κόμβος ενημερώνει και παραδίδει την πληροφορία που κατέχει σε έναν από τους γείτονές του. Οι κόμβοι ανταλλάσσουν περιοδικά μηνύματα για ανίχνευση-ενημέρωση

αλλαγών που έχουν προκύψει (συντεταγμένες zone, γείτονες, συντεταγμένες γειτόνων).

Οι δημιουργοί του CAN προτείνουν μια σειρά από βελτιώσεις που περιλαμβάνουν αύξηση του αριθμού των διαστάσεων ή χρήση πολλών "realities" (καρτεσιανοί χώροι) για να μειωθεί το μήκος του μονοπατιού και καλύτερη ανοχή σε αποτυχίες, καλύτερες μετρικές δρομολόγησης που λαμβάνουν υπόψη την πραγματική τοπολογία του δικτύου και τον χρόνο *RTT (Round – Trip - Time)* για βελτίωση του latency και overloading coordinate zones (πολλοί κόμβοι μοιράζονται το ίδιο zone) για όλα τα παραπάνω πλεονεκτήματα (μείωση του μήκους του μονοπατιού, καλύτερη ανοχή σε αποτυχίες, βελτίωση latency).

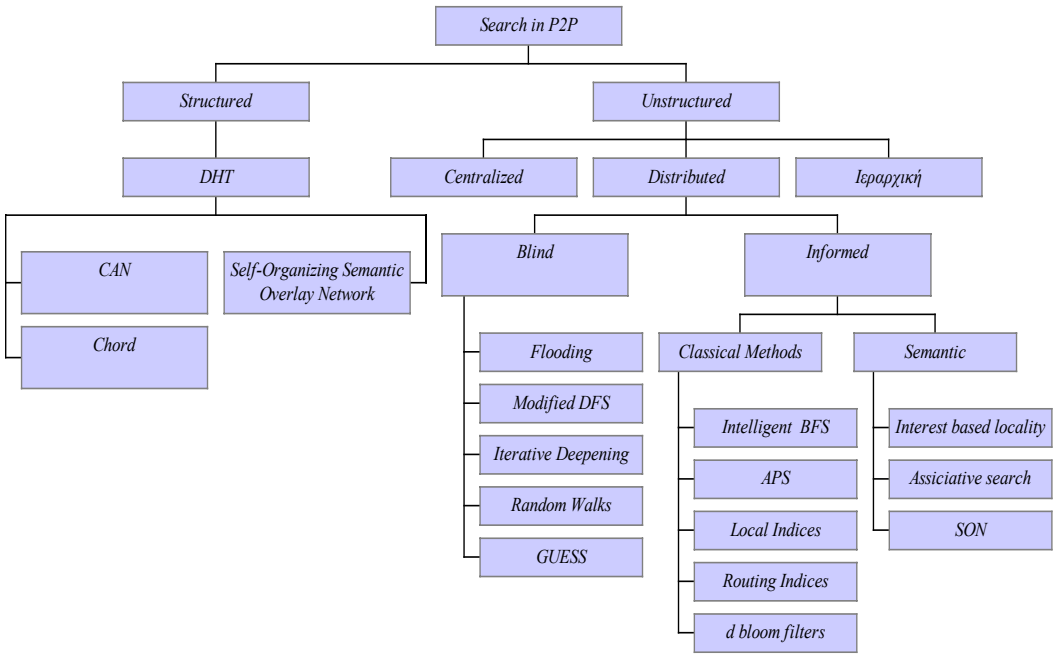
4. Επεξεργασία ~Διαχείριση ερωτήσεων

Ένα κύριο σημείο μελέτης στα peer-to-peer συστήματα είναι η διαχείριση ερωτήσεων που υποβάλλονται στο σύστημα και αποσκοπούν [1, 7, 8]:

- στον εντοπισμό της επιθυμητής πληροφορίας (αρχείου, ή τμήμα του)
- την δρομολόγηση άλλων ερωτήσεων ή την προώθηση ενημερώσεων
- τον εντοπισμό κόμβων για σύνδεση (join)

Η αναζήτηση γίνεται είτε με βάση κάποιο κλειδί (*key*) που αντιστοιχεί στο προς αναζήτηση δεδομένο, είτε με βάση λέξεις κλειδιά (*keyword*) για μεμονωμένα δεδομένα ή για περιοχή δεδομένων (range, multi-attribute queries).

Κλασσικές μεθόδους αναζήτησης όπως κεντρικοποιημένη, ιεραρχική και πλημμύρα εφαρμόζονται και εδώ και εμφανίζονται τόσο στα δομημένα όσο και στο μη δομημένα peer-to-peer συστήματα [8], ενώ προτάσσονται επεκτάσεις ή



Σχήμα 4-1: Στρατηγικές αναζήτησης στα peer to peer δίκτυα

βελτιστοποιήσεις αυτών. Οι μέθοδοι αναζήτησης διακρίνονται επίσης σε *τυφλές* (blind) όπου δεν υπάρχει καθόλου πληροφορία για την αναζήτηση και σε *πληροφορημένες* (informed) όπου υπάρχει είτε κεντρική είτε κατανεμημένη υπηρεσία πληροφόρησης [9].

4.1. Κεντρικοποιημένη αναζήτηση [1]

Μια κεντρική υπηρεσία καταλόγου διατηρεί πληροφορίες τόσο για τα δεδομένα όσο και για τους κόμβους που τα διαθέτουν (είναι αποθηκευμένα σε αυτούς - Napster). Με άλλα λόγια σε μια κεντρική βάση δεδομένων καταχωρούνται μια τιμή κλειδί για το στοιχείο (π.χ. τίτλος τραγουδιού) και η θέση (ο κόμβος) που το στοιχείο βρίσκεται αποθηκευμένο. Αφού εντοπιστεί η επιθυμητή πληροφορία, οι εμπλεκόμενοι κόμβοι επικοινωνούν μεταξύ τους απευθείας για την ανάκτηση της πληροφορίας. Αν η πληροφορία είναι διαθέσιμη σε περισσότερες πηγές τότε επιλέγεται η "καταλληλότερη" ως αναφορά το κόστος, την ταχύτητα, την διαθεσιμότητα σύμφωνα με τις ανάγκες του χρήστη [1].

Η αναζήτηση με τον τρόπο αυτό είναι ταχύτατη – επιτυγχάνεται σε ένα βήμα – όμως δεν παύει η κεντρικοποιημένη δομή της, να αποτελεί περιορισμό στην κλιμάκωση και κεντρικό σημείο αποτυχίας για τα συστήματα αυτά.

4.2. Ιεραρχική[2]

Βασίζεται στο παραδοσιακό σχήμα που χρησιμοποιείται στο Internet για τον εντοπισμό μιας IP διεύθυνσης στο διαδίκτυο με την χρήση Domain Name System (DNS). Στα peer-to-peer συστήματα ειδικοί κόμβοι, οι superpeers, απαρτίζουν μια ιεραρχική δομή (FastTrack's/KaZaA). Η αναζήτηση αρχίζει από την κορυφή της ιεραρχίας και διασχίζει ένα μονοπάτι μέχρι τον κόμβο που περιέχει το στοιχείο (επιθυμητή πληροφορία). Όμως δεν υπάρχει καμιά εγγύηση ότι το στοιχείο θα βρεθεί.

Μια αναζήτηση μπορεί να ολοκληρωθεί σε O(logN) βήματα. Όμως μια αποτυχία ή μια αποχώρηση ενός κόμβου μπορεί να δημιουργήσει σοβαρά προβλήματα ιδιαίτερα αν βρίσκεται ψηλά στην ιεραρχία.

4.3. Τυφλή Αναζήτηση[8, 9, 1]

Η αναζήτηση γίνεται με τη μέθοδο της πλημμύρας (flooding). Όταν δηλαδή ζητείται το δεδομένο Χ τότε ο κόμβος κοιτάει πρώτα την τοπική του βάση. Αν το βρει επιστρέφει το δεδομένο, διαφορετικά προωθεί την ερώτηση στους γείτονές του (π.χ. Gnutella). Αυτή η μέθοδος αναζήτησης δεν εγγυάται ότι το δεδομένο θα βρεθεί και σπαταλά πόρους τους συστήματος (εύρος, υπολογιστική ισχύς κύκλους μηχανής, κλπ.). Πρέπει δε να υπάρχουν μηχανισμοί για την αποφυγή κύκλων, και το σταμάτημα της αναζήτησης μετά από κάποιο αριθμό hops. Οι λύσεις που προτείνονται για το πρόωρο σταμάτημα της αναζήτησης [8] είναι η χρήση της παραμέτρου *TTL* (Time to Live) και *Expanding Ring*.

Η παράμετρος *TTL* τίθεται σε μια τιμή αρχικά, συνήθως 7, και μειώνεται κατά ένα καθώς περνά η ερώτηση από κόμβο σε κόμβο. Η διάδοση των μηνυμάτων σταματά όταν αυτή η τιμή μηδενιστεί.

Εναλλακτικά, αντί για προκαθορισμένη τιμή της παραμέτρου TTL, επιλέγεται η διαδοχική αύξησή της εφόσον δεν βρεθεί η ζητούμενη πληροφορία. Η μέθοδος αυτή

είναι γνωστή ως *Expanding Ring*. Αρχικά, η αναζήτηση ξεκινά με ένα μικρό TTL. Αν δεν βρεθεί αποτέλεσμα, αρχίζει νέα αναζήτηση με αυξημένο TTL. Η διαδικασία επαναλαμβάνεται έως ότου βρεθεί το αποτέλεσμα.

Η τυφλή αναζήτηση με χρήση TTL για μια τυπική τιμή του και με C συνδέσεις κατά μέσο όρο ανά κόμβο (4 ή 5 συνδέσεις συνήθως) παράγει μηνύματα που διακινούνται στο δίκτυο:

$$2 \cdot \sum_{i=0}^{TTL} C \cdot (C-1)^i$$

Για να μειωθεί αυτή η πληθώρα μηνυμάτων προτείνονται παραλλαγές του μηχανισμού της πλημμύρας που επιχειρούν αποδοτικότερη διαχείριση ερωτήσεων σε συστήματα που ο αριθμός των κόμβων είναι μεγάλος[8,1]:

- *Modified –BFS:* Η ερώτηση προωθείται από τον κόμβο σε τυχαίο υποσύνολο των γειτόνων του.
- *Iterative Deepening*: Η αναζήτηση γίνεται σε καθορισμένο βάθος (TTL, hops). Αν το αντικείμενο δεν βρεθεί τότε η ερώτηση επαναλαμβάνεται με νέες τιμές για την παράμετρο βάθους.
- *Random Walks[8]*: Για την μείωση της πληθώρας μηνυμάτων στο δίκτυο προτείνεται επίσης η χρήση ενός ή περισσοτέρων περιπατητών (walker). Ένας περιπατητής επιλέγει τυχαία έναν γείτονα σε κάθε βήμα της αναζήτησης. Για καλύτερα αποτελέσματα αναζήτησης μπορούν να χρησιμοποιηθούν περισσότεροι από ένας περιπατητές (για παράδειγμα *k*). Για να σταματήσουν την αναζήτηση οι περιπατητές χρησιμοποιούνται τεχνικές όπως TTL και checking. Σύμφωνα με την πρώτη, ο περιπατητής σταματά μετά από ορισμένο αριθμό βημάτων ενώ σύμφωνα με την δεύτερη περιοδικά γίνεται έλεγχος αν δόθηκε απάντηση.

4.4. Πληροφορημένη αναζήτηση

Στα peer-to-peer συστήματα προτάθηκαν διάφορες λύσεις για βελτιστοποίηση του τρόπου αναζήτησης [7,8,9]. Οι κόμβοι που μετέχουν στο peer-to-peer σύστημα διατηρούν τοπικά ή κατανεμημένα και σχηματίζουν ένα είδος δομής για το overlay δίκτυο ή ένα επίπεδο πάνω από το overlay δίκτυο. Η δομή αυτή χρησιμοποιείται για την δρομολόγηση των ερωτήσεων ώστε να αποφευχθούν τα μειονεκτήματα της πλημμύρας.

- ***Δομημένα συμμετρικά σχήματα αναζήτησης:*** *(*CAN, Chord, Kademlia, Pastry, Viceroy) Τα συστήματα αυτά είναι δομημένα δηλαδή η θέση των κόμβων και των δεδομένων πάνω στο σύστημα είναι συγκεκριμένη και ο προσδιορισμός της γίνεται με βάση ένα κατανεμημένο πίνακα κατακερματισμού (Distributed Hash Table -DHT). Τόσο οι κόμβοι όσο και τα δεδομένα πρέπει να έχουν ένα μοναδικό αναγνωριστικό. Τα κάθε δεδομένο (ή αντικείμενο) αντιστοιχίζεται και αποθηκεύεται σε έναν κόμβο με βάση μια συνάρτηση κατακερματισμού, επιτυγχάνοντας εν μέρει και εξισορρόπηση φορτίου (load balance).

  Κατά την διαδικασία της αναζήτησης, η θέση του επιθυμητού δεδομένου εντοπίζεται με μια συνάρτηση κατακερματισμού και η ερώτηση προωθείται

στους κόμβους που είναι *πιο κοντά* με βάση κάποια *συνάρτηση απόστασης* (distance function) και λαμβάνοντας υπόψη τον *πίνακα δρομολόγησης(routing indices)* που κάθε κόμβος διατηρεί τοπικά.

Το σύστημα Chord διατηρεί μια skiplist (ή finger tables) δομή και η αναζήτηση ολοκληρώνεται σε O(logN) βήματα αν υπάρχουν N συνδεδεμένοι κόμβοι. Στο CAN η αναζήτηση γίνεται με προώθηση της ερώτησης στον προορισμό (γνωστό εκ των προτέρων) με χρήση greedy αλγορίθμους. Τα συστήματα Kaldemia, Pastry και Tapestry διατηρούν δεντρική δομή (tree) και η αναζήτηση απαιτεί $O(\log_{2^b} N)$ όπου b παράμετρος του αλγορίθμου που συνήθως έχει τιμή 4 και N όπως και παραπάνω οι συνδεδεμένοι κόμβοι. Το Viceroy peer-to-peer σύστημα έχει δομή πεταλούδας (butterfly) και η αναζήτηση διαρκεί το πολύ O(logN) βήματα.

Semantic Overlay Network [23]: Μια άλλη προσέγγιση βασιζόμενη στους DHTs και επεκτείνοντας την ιδέα του CAN προτείνει την τοποθέτηση των δεδομένων στο χώρο όχι με βάση το ID τους αλλά με βάση το περιεχόμενό τους. Έτσι επιτυγχάνεται ένα είδος εννοιολογικής ομαδοποίησης των δεδομένων σε ένα overlay δίκτυο (Semantic Overlay Network). Για κάθε έγγραφο (document) δημιουργείται ένα διάνυσμα (με την βοήθεια του Latent Semantic Index - LSI) και με βάση αυτό τοποθετείται στον CAN χώρο είτε το ίδιο είτε ο δείκτης του.

Όταν υποβάλλεται μια ερώτηση σε κάποιο κόμβο, τότε παράγεται ένα εννοιολογικό διάνυσμα ερώτησης και δρομολογείται στο overlay δίκτυο με την μέθοδο της πλημμύρας σε μια ακτίνα r. Όλοι οι κόμβοι που λαμβάνουν το ερώτημα ελέγχουν τοπικά για το έγγραφο που ταιριάζει καλύτερα και απαντούν στον αιτούντα κόμβο.

Το μέσο μήκος του μονοπατιού δρομολόγησης είναι όπως και στο CAN ($d/4 \cdot n^{\frac{1}{d}}$) όπου *d* το πλήθος των διαστάσεων και *n* το πλήθος των κόμβων. Όμως όσο το πλήθος των διαστάσεων αυξάνει τα αποτελέσματα δεν είναι τα επιθυμητά γι αυτό και προτάσσονται δυο βελτιώσεις: *Rolling Index* και *Content – Direct Search*. Η χρήση του Rolling Index επιλύει το πρόβλημα την ασυμφωνία μεταξύ του εννοιολογικού και του CAN χώρου ως προς τις διαστάσεις με την δημιουργία rotate vectors που βασίζεται στην εκτίμηση για αποτελεσματική διαμέριση του CAN. Η Content – Direct Search χρησιμοποιείται για καθοδηγούμενη αναζήτηση έτσι ώστε να μειωθεί ο αριθμός των επισκεπτόμενων κόμβων.

- ***Routing Indices (RIs)*** [7]: Πρόκειται για έναν μηχανισμό κατανεμημένου ευρετηρίου που σε κάθε κόμβο αποθηκεύει ένα μικρό τμήμα του. Σκοπός των RIs είναι να δείξουν την κατεύθυνση που βρίσκεται το αντικείμενο και όχι την πραγματική του θέση. Όταν υποβάλλεται μια ερώτηση σε έναν κόμβο, εκείνος κοιτάζει την λίστα των γειτόνων του και την προωθεί σε αυτόν με την μεγαλύτερη "goodness" τιμή. Η έννοια "goodness" μπορεί να θεωρηθεί ως ο αριθμός των σχετικών με το ζητούμενο αντικείμενο αρχείων. Τρία εναλλακτικά σχήματα παρουσιάζονται:
    - *compound RI*: Για κάθε πιθανό μονοπάτι συναθροίζεται η "goodness" για το αντικείμενο και προκύπτουν οι εγγραφές του ευρετηρίου.

- *hop-count RI*: Διατηρούνται αρχεία συνάθροισης για ένα ορισμένο αριθμό από hops.
- *exponential RI*: Τα αποτελέσματα του RI πίνακα προκύπτουν ως αποτέλεσμα μιας φόρμουλας κόστους πάνω στο hop-count RI.

- ***Εννοιολογική προσέγγιση***: Η βασική ιδέα αυτής της οργάνωσης είναι απλή [10, 11, 12]: αν ένας κόμβος διαθέτει περιεχόμενο (π.χ. έγγραφο, μουσικό κομμάτι, κλπ.) που ενδιαφέρει και κάποιον άλλον κόμβο, τότε είναι πολύ πιθανό να έχει επίσης και άλλα περιεχόμενα που ενδιαφέρουν επίσης τον άλλον. Μπορεί λοιπόν να δημιουργηθεί ένα δίκτυο μέσω ενός πρωτοκόλλου αυτοδιοργάνωσης.

  Μερικές τέτοιες προσεγγίσεις είναι:

  - *Interest – Based locality*
  - *Associative Search*
  - *Semantic Overlay Network(SONs)*

Interest – Based locality[10]: Σύμφωνα με την προσέγγιση αυτή οι κόμβοι που έχουν κοινά ενδιαφέροντα δημιουργούν απευθείας συνδέσεις μεταξύ τους, shortcuts, τις οποίες και χρησιμοποιούν για τον εντοπισμό δεδομένων. Τα shortcuts παρέχουν μια χαλαρή δομή πάνω από το υπάρχον overlay δίκτυο (Gnutella). Αν μια αναζήτηση μέσω αυτών (shortcuts) αποτύχει τότε χρησιμοποιείται το overlay δίκτυο. Για παράδειγμα έστω ότι ένας κόμβος Χ αναζητά τα αρχεία file1, file2, file3 και σε κάποιους κόμβους εντοπίζει κάποια από αυτά, ενώ στον κόμβο Υ εντοπίζει και τα τρία αρχεία. Σκοπός είναι το σύστημα να εντοπίζει τέτοιους κόμβους και να κατεβάζει τα αρχεία που επιθυμεί απευθείας από αυτούς.

Το ερώτημα είναι πως θα ανακαλυφθούν οι κόμβοι αυτοί για να μπουν στην shortcuts λίστα του κόμβου και πως θα επιλεγούν μεταξύ άλλων οι καταλληλότεροι. Η ανακάλυψη των shortcuts όταν ένας κόμβος εισέρχεται στο σύστημα για πρώτη φορά γίνεται μέσω ερωτήσεων από τον κόμβο προς τους γείτονές με την μέθοδο της πλημμύρας. Από αυτά που βρίσκει επιλέγει κάποια τυχαία και τα προσθέτει στην λίστα του. Η λίστα αυτή αποθηκεύεται από τον κάθε κόμβο που δεσμεύει μνήμη για το σκοπό αυτό. Καθώς οι κόμβοι έρχονται και φεύγουν στο σύστημα και τα ενδιαφέροντα μεταβάλλονται, τα περιεχόμενα της λίστας ανανεώνονται και ενημερώνονται.
Η επιλογή των shortcuts που θα προστεθούν στη λίστα γίνεται από τον κόμβο που την διατηρεί με συνδυασμό κριτηρίων όπως η πιθανότητα «προμήθειας» περιεχομένου, latency, μήκος μονοπατιού, διαθεσιμότητα εύρους, ποσότητα περιεχομένων, φόρτο shortcuts.

Associative Search[11]: Όπως και στην προηγούμενη περίπτωση, έτσι και εδώ, οι κόμβοι οργανώνονται ή ομαδοποιούνται ένα επίπεδο πάνω από το υφιστάμενο peer-to-peer σύστημα (Gnutella, FastTrack). Η ομαδοποίηση των κόμβων σε μια χαλαρή τοπολογία γίνεται με βάση τα όμοια εννοιολογικά δεδομένα με στόχο να απαντά αποδοτικά σε ερωτήσεις για αυτά καθώς και να υποστηρίζει partial match ερωτήσεις ή ακόμη και την εύρεση σπανίων δεδομένων.

Η αναζήτηση είναι καθοδηγούμενη (*guided search*) καθώς η ερώτηση προωθείται στους συναφείς κόμβους που σχηματίζουν κάποιο σύνολο. Το σύνολο, που ονομάζεται *guide rule*, σχηματίζεται από κόμβους που ικανοποιούν κάποια συνθήκη. Μια κατηγορία guide rule είναι το *possession rule* όπου η συνθήκη που πρέπει να ικανοποιείται είναι η παρουσία ενός δεδομένου στο τοπικό ευρετήριο. Οι κόμβοι μπορούν να μετέχουν σε διαφορετικά – περισσότερα του ενός guide rule. Ουσιαστικά η αναζήτηση εκμεταλλεύεται τις συνδέσεις μεταξύ των κόμβων και προωθεί τις ερωτήσεις σε συναφείς "κοινότητες" κόμβων. Χρησιμοποιούνται δυο αλγόριθμοι για την καθοδηγούμενη αναζήτηση: ο αλγόριθμος *RAPIER* και ο αλγόριθμος *GAS*.

> ***RAPIER(Random Possession Rule):*** Ο αλγόριθμος βασίζεται σε μια απλή επαναληπτική στρατηγική: Επιλέγει ένα τυχαίο στοιχείο από το τοπικό ευρετήριο και εφαρμόζει "blind" αναζήτηση στους κόμβους που ανήκουν στο "possession rule" σε προκαθορισμένο βάθος. Αν δεν βρεθεί το δεδομένο η αναζήτηση γίνεται στο υφιστάμενο δίκτυο (π.χ Gnutella) όπως αυτό την υποστηρίζει (τυφλή αναζήτηση).
>
> ***GAS (Greedy Guide Rule):*** Ο αλγόριθμος GAS είναι μια βελτιστοποιημένη έκδοση του προηγουμένου αλγορίθμου (Rapier). Ενώ πριν η επιλογή του στοιχείου από την λίστα γινόταν τυχαία, τώρα λαμβάνεται υπόψη η συνεισφορά του καθενός (πιθανότητα) στην επιτυχή αναζήτηση. Έτσι ο κανόνας καθοδήγησης που επιλέγεται είναι αυτός που έχει την μεγαλύτερη πιθανότητα για επιτυχή αναζήτηση.

Semantic Overlay Network[12]: Τα Semantic Overlay Network στηρίζονται στη λογική ότι θα είναι αποδοτικότερο η δρομολόγηση των ερωτήσεων να γίνεται μόνο προς τους κόμβους που είναι πιο πιθανό να έχουν απαντήσεις. Για να επιτευχθεί αυτό οι κόμβοι με εννοιολογικά όμοια περιεχόμενα σχηματίζουν cluster. Η κατηγοριοποίηση γίνεται ιεραρχικά και σχηματίζουν ένα overlay δίκτυο σε ανώτερο επίπεδο, με τους κόμβους να καταλαμβάνουν θέσεις στην ιεραρχία ανάλογα με το πλήθος των δεδομένων (files) που κατέχουν αλλά και την εννοιολογική τους σημασία. Ο σχηματισμός αυτός δεν αποκλείει τα λάθη, δηλαδή τις εσφαλμένες κατηγοριοποιήσεις όταν η πληροφορία δεν είναι αρκετή.

Όταν μια ερώτηση εισάγεται σε ένα τέτοιο σύστημα, αρχικά θα πρέπει και η ίδια να κατηγοριοποιηθεί και στη συνέχεια να προωθηθεί στο αντίστοιχο συστατικό (τμήμα της ιεραρχίας) για να απαντηθεί. Αν το αποτέλεσμα δεν είναι αρκετό (η απάντηση δεν βρέθηκε, ή δεν βρέθηκαν όλα τα δεδομένα) τότε η ερώτηση προωθείτε σε ανώτερα επίπεδα.

4.5. Σύγκριση μεθόδων αναζήτησης

	Δομημένα (Structured)		**Μη Δομημένα (Unstructured)**			
	CAN	**Chord**	**Napster**	**Gnutella**	**FreeNet**	**FastTrack /KaZaA**
Architecture	d-dimension ID coordination space	Ring	Central	General Graph, Flat & Ad-Hoc Network	General Graph	General graph
Parameter	***N***- number of nodes ***d*** number of dimensions	***N***- number of nodes	None	None	None	None
Search Protocol	(Key, Value) maps point in space	Matching Key with node ID	Centralized	Query Flooding	Key, Descriptive text search	Super Peers
Time Complexity	$O(dn^{1/d})$	O(logN)	O(1)	Unbounded – No guarantee to locate data TTL limits, if data located	Hops - to -Live limits	-
Space Complexity	2d	logN	Constant	Constant	Constant	Centralized
Peers join & leaves	2d	$(\log N)^2$	Constant	Constant	Constant	Centralized
Functionality	DHT based		Centralized	Flat topology	Loosely DHT	peers connected to Super-Peers

Πίνακας 1: Σύγκριση μεταξύ δομημένων και αδόμητων p2p δικτύων

Γενικά ο εντοπισμός δεδομένων στα peer-to-peer συστήματα απαιτεί μεγάλο εύρος αν η αναζήτηση γίνεται μέσω κεντρικού server (Napster), υψηλό overhead αν γίνει αναζήτηση flooding (Gnutella) και υψηλό κόστος ενημέρωσης αν διατηρούνται πολλά αντίγραφα των δεδομένων για πιο σύντομη αναζήτηση – ανάκτηση.

Στα συστήματα που βασίζονται σε πίνακες κατακερματισμού η αναζήτηση απαιτεί $O(\log N)$ βήματα κατά μέσο όρο με την προϋπόθεση ότι υπάρχει εξισορρόπηση φορτίου και ομοιόμορφη κατανομή ερωτήσεων. Στα ιεραρχικά συστήματα η δρομολόγηση ερωτήσεων απαιτεί περίπου $O(\log N)$ βήματα αν το δέντρο της ιεραρχίας είναι ισορροπημένο (balanced).

Μετά την παρουσίαση των αλγορίθμων αναζήτησης το συμπέρασμα είναι ότι κλειδί για μία μέθοδο αναζήτησης είναι ο μικρός αριθμός κόμβων που προσπελαύνονται όσο πιο γρήγορα είναι δυνατό και με το λιγότερο δυνατό overhead. Σε αυτό συμβάλλουν και οι προτασσόμενες βελτιστοποιήσεις που περιλαμβάνουν προσαρμοστικές μεθόδους για τον τερματισμό των αλγορίθμων, την απάλειψη

κύκλων στην δρομολόγηση μηνυμάτων, και στη μικρή αύξηση του αριθμού των κόμβων που επισκέπτεται ο αλγόριθμος σε κάθε βήμα.

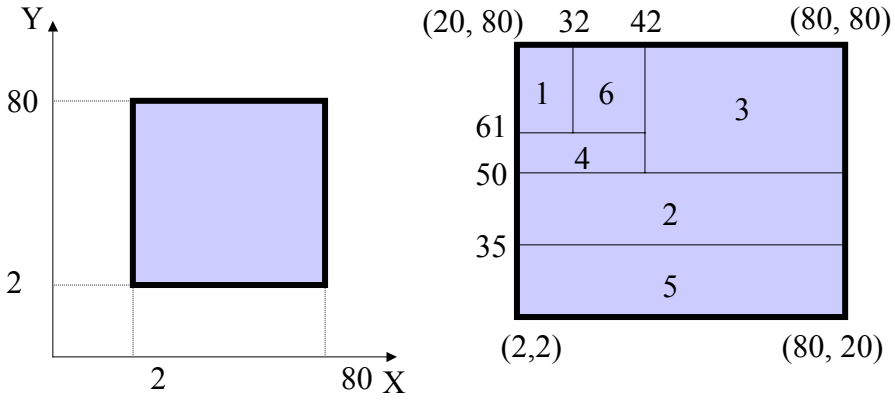
4.6. Προχωρημένες ερωτήσεις – αναζητήσεις

Οι σημερινές εφαρμογές απαιτούν την απάντηση και σε πιο πολύπλοκες ερωτήσεις σε σχέση με αυτά που αναφέρθηκαν παραπάνω, όπου η αναζήτηση γίνεται με βάση κάποιο όνομα ή κάποιο κλειδί. Μια πολύπλοκη ερώτηση μπορεί να αφορά αναζήτηση *εύρους δεδομένων* (range queries), για παράδειγμα εργαζόμενοι ηλικίας 20 έως 35 ετών, *πολλαπλά γνωρίσματα* (multiple attributes) για παράδειγμα ηλικία εργαζομένων, ύψος αμοιβής, ή και άλλες συνδυαστικές ερωτήσεις. Αυτού του είδους οι ερωτήσεις ενδεχομένως να απαιτούν καθολική γνώση του συστήματος, για παράδειγμα στατιστικές πληροφορίες και η απάντησή τους επιτυγχάνεται με συλλογή αποτελεσμάτων με παραδοσιακές τεχνικές απλών κατανεμημένων ερωτήσεων.

4.6.1. Ερωτήσεις διαστήματος (Range Queries)

Το θέμα αυτό διαπραγματεύεται και το [24]. Οι συγγραφείς του προτείνουν μια επέκταση του πολυδιάστατου CAN σε ένα σύστημα διαμοιρασμού δεδομένων γενικού σκοπού. Οι κόμβοι συνεργάζονται για να απαντήσουν σε ερωτήσεις εύρους (range queries).

Το μοντέλο λειτουργίας βασίζεται στο CAN και κάθε γνώρισμα (attribute) αντιστοιχίζεται σε έναν εικονικό χώρο δυο διαστάσεων με βάση το πεδίο ορισμού. Για παράδειγμα το πεδίο ορισμού [a,b] αντιστοιχίζεται στο χώρο με συντεταγμένες (a, a) έως (b, b). Οι *ενεργοί* κόμβοι είναι κάτοχοι τμήματος του χώρου (zone) ανάλογα με τα δεδομένα που διατηρούν ενώ υπάρχουν και *παθητικοί* κόμβοι που δεν μετέχουν στην διαμέριση.



Σχήμα 4-2: Αντιστοίχηση Domain στο χώρο CAN και διαμέριση σε zone

Η δρομολόγηση ερωτήσεων διαστήματος γίνεται όπως και στο CAN και η αποτελεσματικότητα αυξάνεται με την χρήση ενδιάμεσων προσωρινών αποτελεσμάτων (caching) από προηγούμενες ερωτήσεις. Κάθε κόμβος διατηρεί έναν πίνακα δρομολόγησης με πληροφορίες για τους γείτονές του (IP, συντεταγμένες). Αρχικά η ερώτηση υποβάλλεται στο target zone (το zone στο οποίο ανήκει το σημείο που προκύπτει από την συνάρτηση κατακερματισμού). Αν το επιθυμητό αποτέλεσμα υπάρχει τοπικά τότε στέλνεται στον αιτούντα κόμβο. Αν υπάρχει δείκτης σε κόμβο που αποθηκεύει τα δεδομένα τότε η ερώτηση προωθείται σε αυτόν, διαφορετικά γίνεται προώθηση της ερώτησης στους γείτονες κόμβους που γεωγραφικά βρίσκονται πιο κοντά στο σημείο απάντησης. Αν οι ερωτήσεις αφορούν περισσότερα από ένα γνωρίσματα, τότε για το καθένα δημιουργείται ξεχωριστός εικονικός χώρος (CAN).

Το μήκος του μονοπατιού δρομολόγησης είναι $O(\log\sqrt{n})$ για εξ ίσου διαμερισμό του χώρου.

4.6.2. Multi-dimensional queries (MURK~SCRAP)

Για την δρομολόγηση και απάντηση ερωτήσεων πολλαπλού διαστήματος προτάσσεται μια στρατηγική [28] βασιζόμενη στους πίνακες κατακερματισμού και σε ερωτήσεις απλού γνωρίσματος. Πρόκειται για τις μεθόδους SCRAP και MURK.

Η μέθοδος SCRAP (Space-filling Curves with Range Partitioning) διαμερίζει τα δεδομένα σε δύο βήματα: πρώτα τα αντιστοιχίζει σε μια διάσταση και μετά τα διαμερίζει δυναμικά. Οι multidimensional range ερωτήσεις δρομολογούνται ως εξής: μετατρέπονται σε 1-d σύνολο ερωτήσεων μέσω Space-filling Curves και κάθε μια δρομολογείται στον κατάλληλο κόμβο για να απαντηθεί με μια κλασσική μέθοδο (π.χ. Chord).

Η μέθοδος MURK διαμερίζει τα δεδομένα όπως αυτά κατανέμονται στο χώρο σε "rectangles" με ίσο φορτίο και το καθένα αντιστοιχεί σε έναν κόμβο. Η διασύνδεση των κόμβων γίνεται όπως στο CAN μόνο που πρέπει να υπάρχουν συνδέσεις μεταξύ των κόμβων έτσι ώστε μια multidimensional range query να μπορεί να σταλεί σε όλους τους σχετικούς κόμβους. Η δρομολόγηση γίνεται πάνω από το πλέγμα που προκύπτει με greedy αλγορίθμους προς τον κόμβο που ελαχιστοποιεί την Manhattan απόσταση.

Το κόστος αναζήτησης στην πρώτη μέθοδο αυξάνει λογαριθμικά με την αύξηση των κόμβων ενώ στην δεύτερη περίπτωση είναι $\Theta(\sqrt{n})$

4.6.3. Mercury

Το Mercury [29] είναι ένα ακόμη πρωτόκολλο για την υποστήριξη ερωτήσεων πολλαπλών γνωρισμάτων (multiple attribute). Η οργάνωση του Mercury έχει δυο κύρια συστατικά: a) για κάθε attribute μιας multiple-attribute ερώτησης δημιουργεί ένα λογικό σύνολο κόμβων του δικτύου που καλείται *routing hub* b) οργανώνει τους κόμβους που μετέχουν στο routing hub σε *δακτύλιο* και τα δεδομένα τοποθετούνται συνεχόμενα πάνω σε αυτό το δακτύλιο.

Μια ερώτηση περνά σε ένα attribute hub για κάθε γνώρισμα για να απαντηθεί.

4.6.4. Mutant Query

Ένα πλαίσιο εργασίας για τον χειρισμό κατανεμημένων ερωτήσεων [30] προτείνει την χρήση mutant query plan. Το mutant query plan είναι ένα αλγεβρικό γραφικό πλάνο ερώτησης κωδικοποιημένο σε XML που αναφέρεται στις τοποθεσίες των δεδομένων (URLs) και στα ονόματα των πηγών που παρέχουν τα δεδομένα (URNs). Οι κόμβοι που απαρτίζουν το δίκτυο χρησιμοποιούν ιεραρχίες για την κατηγοριοποίηση των δεδομένων και μπορούν να παρέχουν ή και να ζητούν δεδομένα.

Μια αναζήτηση εξελίσσεται ως εξής: Ένα πλάνο ερώτησης με δεντρική δομή σχηματίζεται στον client . Αυτό καθώς περνά από server σε server συγκεντρώνει επιμέρους αποτελέσματα και μειώνεται. Όταν υπολογιστεί πλήρως, τότε το αποτέλεσμα σε XML μορφή, επιστρέφει στον client.

4.6.5. Παράδειγμα

Έστω ότι ένας κόμβος (χρήστης) αναζητά CD που κοστίζουν λιγότερο από 10€ στην περιοχή του Portland. Ας δούμε πως απαντάτε η ερώτηση αυτή αν χρησιμοποιηθεί μια από τις μεθόδους που περιγράφονται στην 4.6 ενότητα, κάνοντας τις εξής υποθέσεις:
Ο χρήστης αναζητά κάποια τραγούδια που δημοσιοποιούνται σε κάποια λίστα από τους παροχείς.
Οι τιμές των CDs κυμαίνονται από 2€ ως 80€ (π.χ. συλλογές). Οι περιοχές αναζήτησης (πόλεις) μπορεί να είναι ανά τον κόσμο. Για χάρη απλότητας, δεν εμφανίζονται στα σχήματα ή εμφανίζονται ελάιστες.

Range Query

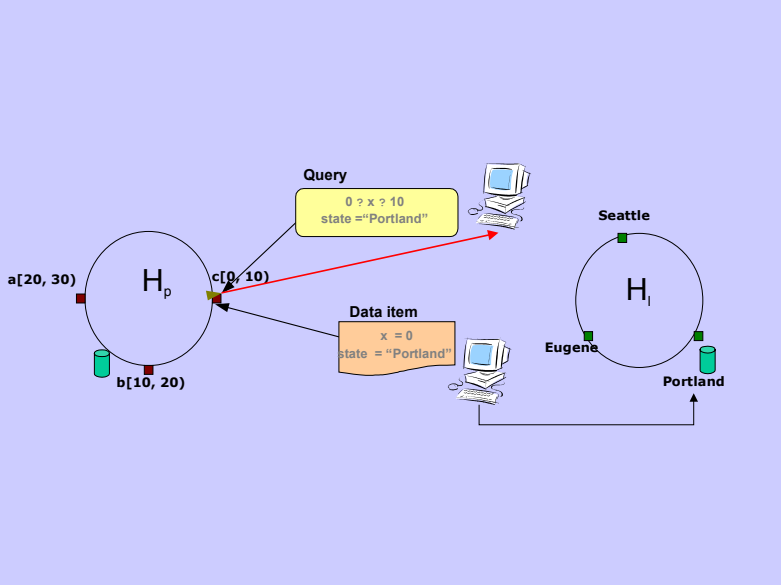
Σύμφωνα με την λειτουργία του πρωτοκόλλου, όταν υπάρχουν *n* γνωρίσματα στην ερώτηση τότε αυτή αντιστοιχίζεται σε έναν 2n εικονικό χώρο. Έτσι η παραπάνω ερώτηση θα μετασχηματισθεί στην ερώτηση διαστήματος (0, 10), (Portland, Portland) στο χώρο τεσσάρων διαστάσεων. Οι πρώτες δύο διαστάσεις αφορούν το πρώτο γνώρισμα (τιμή) ενώ οι δυο επόμενες το δεύτερο γνώρισμα. Επομένως μπορεί να εξετασθεί σαν ανεξάρτητος διαστατός χώρος για απλότητα.

Έστω ότι ο κόμβος 1 (Σχήμα 4.2) υποβάλλει το ερώτημα και ο κόμβος 5 διατηρεί αντίγραφα για εύρος τιμών (2,30). Η ερώτηση δρομολογείται στο zone του 5 και επιλέγεται το αποτέλεσμα –που θα μπορούσε να το αποθηκεύει και από προηγούμενο ερώτημα - και στέλνεται σαν απάντηση στον κόμβο 1. Η αντίστοιχη διαδικασία γίνεται και για το δεύτερο μέρος του ερωτήματος. Η αντιστοίχηση γίνεται πάλι στο διδιάστατο χώρο και το ερώτημα υποβάλλεται με ίδιο πάνω και κάτω όριο (Portland, Portland). Ο συνδυασμός των δύο στων χώρο των 4 διαστάσεων επιστέφεται ως απάντηση στον χρήστη του κόμβου.

Mercury

Η ερώτηση παραδίδεται σε ένα H_α Hub όπου α είναι τα γνωρίσματα που χρησιμοποιούνται στην ερώτηση. Εκεί, εσωτερικά το ένα τμήμα δρομολογείται στα H_p και H_l hubs που απεικονίζουν τα διαστήματα τιμών για το κόστος του CD το

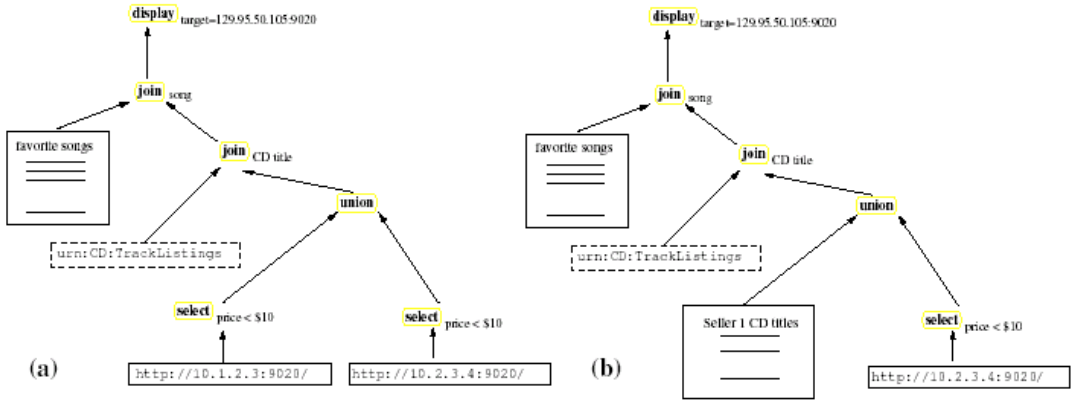
πρώτο (*p: price*) και την περιοχή (*l: location*) το δεύτερο στους κόμβους όπως φαίνεται στο σχήμα 4.3. Το αποτέλεσμα επιστρέφεται στον κόμβο που έκανε την ερώτηση.



Σχήμα 4-3: Εξέλιξη ερώτησης σε Mercury

Mutant query plan

Μια υπηρεσία (online track-listing service) συνδέει τον χρήστη με τις λίστες των προς πώληση CDs. Δημιουργείται ένα mutant πλάνο ερώτησης που περιλαμβάνει τους τελεστές select, join, και τον ψευδό-τελεστή *display και καθορίζει το στόχο – κόμβο που θα απαντήσει. Η αναζήτηση γίνεται για δύο* URNs: urn:ForSale:Portland-CDs και urn:CD:TrackListings όπως φαίνεται στο παρακάτω σχήμα.



Σχήμα 4-4: Η αναζήτηση με χρήση mutant query

Multi-dimensional queries

SCRAP: Έστω ότι τα γνωρίσματα *price* και *location* απεικονίζονται σε 4-bit κώδικα. Από τις δυο διαστάσεις, με παρεμβολή των κωδικών προκύπτει μια διάσταση με 8bits (π.χ. $a_3a_2a_1a_0$ για το πρώτο γνώρισμα και για το δεύτερο γνώρισμα προκύπτει $a_3b_3a_2$

$b_2a_1b_1a_0b_0$). Στη συνέχεια εκτελείται κάποιος well known αλγόριθμος space-filling curve για αντιστοίχηση ερωτήσεων. Το αποτέλεσμα που επιστρέφεται μπορεί να περιέχει και εσφαλμένες απαντήσεις λόγω της αντιστοίχησης σε μια διάσταση.

MURK: Η ερώτηση υποβάλλεται σε ένα κόμβο. Αυτός την στέλνει σε όλους τους κόμβους που έχουν σχετικά δεδομένα (CD >10€, location=Portland) επαναληπτικά μέχρι να συλλέξει την απάντηση. Η προώθηση της ερώτησης γίνεται με greedy στρατηγικές που βασίζονται στην ελάχιστη Manhattan απόσταση από τον προορισμό (αντίστοιχο "rectangle").

5. Διατήρηση αντιγράφων (Content Caching, Replication)

Η διαθεσιμότητα των περιεχομένων, η βελτίωση της απόδοσης και η αντοχή σε προσπάθειες λογοκρισίας στα peer-to-peer συστήματα βασίζεται στην ύπαρξη αντιγράφων. Αντίγραφα των αντικειμένων τοποθετούνται σε περισσότερους κόμβους έτσι ώστε οι αιτήσεις να ικανοποιούνται αποτελεσματικότερα με μικρότερο κόστος (path length, latency, κλπ.) ή ακόμη και για να εντοπίζονται σπάνια δεδομένα. Άμεσα σχετιζόμενο θέμα που προκύπτει με την διατήρηση αντιγράφων είναι το πρόβλημα της ενημέρωσης για την διατήρηση της συνέπειας των δεδομένων. Όπως αναλύεται στη συνέχεια, κλασσικές προσεγγίσεις ή νέες προτάσεις διαχειρίζονται το ζήτημα αυτό.

Υπάρχουν δύο μηχανισμοί για την δημιουργία αντιγράφων, *replication* και *caching*. Αν και ο στόχος τους είναι κοινός ως προς την αποδοτικότητα του συστήματος και την διαθεσιμότητα των δεδομένων έχουν μια σημαντική διαφορά. Τα αντικείμενα που αποθηκεύονται με τον μηχανισμό replication αποθηκεύονται στον κόμβο σε μια βοηθητική μνήμη, ενώ με τον μηχανισμό caching αποθηκεύονται σε κάποια προσωρινή μνήμη.

> **Caching**: Δημιουργία των αντιγράφων από τους peers καθώς κατεβάζουν δεδομένα στο δίσκος τους για να αυξήσουν την διαθεσιμότητα, ακόμη και όταν πολλοί προσπελαύνουν τα ίδια περιεχόμενα από διαφορετικές πηγές[10]. Η μέθοδος αυτή αναφέρεται και ως παθητική αντιγραφή (passive replication)[13]. Τα δεδομένα αποθηκεύονται προσωρινά στην cache και δεν υπάρχει εγγύηση για το χρονικό διάστημα που θα παραμείνουν εκεί, αφού μπορεί να αντικατασταθούν από πιο πρόσφατα δεδομένα από λειτουργίες ανάκτησης.
>
> **Replication**: Τα δεδομένα αντιγράφονται σε ένα κόμβο ακόμη και αν ο ίδιος δεν τα έχει ζητήσει. Μια στρατηγική [8] που χρησιμοποιείται από την Gnutella προτείνει την "*owner replication*" όπου μόνο ο κόμβος που αναζητά το αντικείμενο το αποθηκεύει. Μια άλλη στρατηγική [8] που χρησιμοποιείται από συστήματα όπως το Freenet προτείνει την "*path replication*" όπου όταν γίνει επιτυχή αναζήτηση, οι ενδιάμεσοι κόμβοι κατά μήκος του μονοπατιού (κόμβος αίτησης –κόμβος απόκρισης) διατηρούν αντίγραφο του δεδομένου.

5.1. Στρατηγική δημιουργίας αντιγράφων

Δεδομένου ότι κάθε σύστημα δομημένο ή αδόμητο, επιλέγει την στρατηγική δημιουργίας αντιγράφων τα ερωτήματα που τίθενται είναι[8]:

- Τι θα αντιγραφεί;
- Που θα αποθηκευτεί το αντίγραφο;
- Πόσα αντίγραφα πρέπει να δημιουργηθούν;

Μια εύκολη απάντηση στην πρώτη ερώτηση είναι δημιουργούνται αντίγραφα των πιο δημοφιλή δεδομένων ενώ μπορεί να υιοθετηθεί και η παραδοσιακή προσέγγιση των κατανεμημένων συστημάτων που δημιουργεί και διατηρεί αντίγραφα σε αναλογία των αναγνώσεων/εγγραφών (read/write). Σε μερικές περιπτώσεις δημιουργούνται αντίγραφα όχι των δεδομένων αλλά τμήματος του ευρετηρίου [8].

Η τοποθέτηση των αντιγράφων στο peer-to-peer δίκτυο είναι επίσης ένα ζήτημα προς μελέτη. Συνήθως η τοποθέτηση γίνεται έτσι ώστε να επιτυγχάνεται εξισορρόπηση φορτίου (load balancing) και τοπικότητα αναφοράς (locality).

Το πλήθος των αντιγράφων συνήθως ακολουθεί κάποια κατανομή (8) ή καθορίζεται από τον αλγόριθμο του συστήματος. Θεωρητικά[1, 8] το πλήθος των αντιγράφων που δημιουργούνται μπορεί να είναι:

- *Uniform*: το πλήθος των αντιγράφων είναι ανεξάρτητο από την κατανομή της ερώτησης
- *Proportional*: το πλήθος των αντιγράφων είναι ανάλογο προς τις ερωτήσεις (πιο πολλά αντίγραφα για τα δημοφιλή δεδομένα)
- *Square*-Root: Δίνει το βέλτιστο πλήθος αντιγράφων (p) που είναι ανάλογο του τετραγώνου του ποσοστού των ερωτήσεων (q_r): $p \propto \sqrt{q_r}$.

Ένα σημαντικό πρόβλημα που ανακύπτει με την χρήση αντιγράφων είναι η διαδικασία της ενημέρωσης ο συγχρονισμός και η διατήρηση της συνέπειας αυτών.

5.1.1. Αντιγραφή σε αδόμητα συστήματα

Στα αδόμητα peer-to-peer δίκτυα όπως το Napster και η Gnutella η διατήρηση αντιγράφων βασίζεται σε ένα παθητικό και χωρίς έλεγχο μηχανισμό που εξαρτάται κυρίως από την δημοτικότητα των δεδομένων (files). Άλλα συστήματα, όπως Freenet [32], Publius [31] χρησιμοποιούν ελεγχόμενες στρατηγικές διατήρησης αντιγράφων καθώς το αποτέλεσμα αντιγράφεται στους ενδιάμεσους κόμβους καθώς δρομολογείται στον κόμβο που υπέβαλε την ερώτηση. Στο Oceanstore διατηρούνται δυο επίπεδα αντιγράφων, που αποφεύγουν να τοποθετούν αντίγραφα σε περιοχές με μεγαλύτερη πιθανότητα αποτυχίας.

5.1.2. Αντίγραφα σε δομημένα συστήματα

Στο σύστημα ***Chord[5]*** δημιουργούνται αντίγραφα σε περισσότερους του ενός κόμβους, με μια διαδικασία που επαφίεται στο λογισμικό και βασίζεται στη λίστα διαδόχων (successor-list). Έτσι για το δεδομένο με κλειδί key δημιουργείται αντίγραφο στους k επόμενους κόμβους από τον κόμβο του key. Η διαδικασία

δημιουργίας αντιγράφων εκτελείται όταν οι διάδοχοι κόμβοι εισέρχονται ή αποχωρούν από το σύστημα.

Στο σύστημα ***CAN[6]*** αντίγραφα δημιουργούνται είτε με τον μηχανισμό replication είτε με τον μηχανισμό caching.

Replication: Αν υπάρχει μεγάλος αριθμός αιτήσεων για κάποιο δεδομένο τότε αυτό αντιγράφεται από τους γείτονες του κόμβου που το έχει. Επίσης η διαδικασία αντιγράφων επιτυγχάνεται με:

- Διατήρηση πολλαπλών ανεξάρτητων χώρων συντεταγμένων (realities), όπου κάθε κόμβος διατηρεί διαφορετική ζώνη και γείτονες: αντίγραφα δημιουργούνται σε κάθε χώρο.
- Υπερφόρτωση ζωνών: κάθε ζώνη ανατίθεται σε περισσότερους του ενός κόμβους. Τα δεδομένα που αντιστοιχούν σε μία ζώνη σε όσους κόμβους κατέχουν τη ζώνη.
- Δημιουργία αντιγράφων με χρήση διαφορετικών συναρτήσεων κατακερματισμού

Caching: Κάθε κόμβος διατηρεί σε μια cache μνήμη τα δεδομένα που προσπέλασε πρόσφατα. Έτσι αν φθάσει μια ερώτηση για κάποιο δεδομένο, πρώτα εξετάζει αν αυτό είναι αποθηκευμένο στην cache του. Αν πράγματι υπάρχει, απαντά ο ίδιος στην ερώτηση παρέχοντας το δεδομένο, διαφορετικά προωθεί την ερώτηση.

Στο CAN τα αντίγραφα έχουν κάποια "διάρκεια ζωής" (time-to-live) και στη συνέχεια πρέπει να ανανεωθούν.

Στο Pastry διατηρούνται αντίγραφα σε k γειτονικούς κόμβους ενώ η ίδια λογική εφαρμόζεται και στο σύστημα Tapestry. Το LAR [27] πρωτόκολλο αντιγραφής συνδυάζει το πνεύμα δημιουργίας αντιγράφων στα DHTs συστήματα και στα ιεραρχικά συστήματα, δημιουργώντας αντίγραφα σε γειτονικούς κόμβους, όταν ο φόρτος ενός κόμβου ξεπεράσει κάποιο όριο. Μια γενίκευση του LAR πρωτοκόλλου, *app-cache* προτείνει την δημιουργία σταθερού αριθμού αντιγράφων "κοντά" σε κάθε αρχείο.

5.2. Ενημέρωση

Ο μηχανισμός διαχείρισης αντιγράφων είναι στενά συνδεδεμένος με τον μηχανισμό της ενημέρωσης και εξαρτάται από το περιβάλλον της εφαρμογής και το peer-to-peer σύστημα.

5.2.1. Ενημέρωση σε αδόμητα συστήματα

Στα αδόμητα συστήματα υπάρχουν διαφορετικές προσεγγίσεις για τον χειρισμό των ενημερώσεων. Στο Napster και στο FastTrack που διατηρούν κεντρικά ευρετήρια, όταν γίνεται ενημέρωση των δεδομένων ουσιαστικά γίνεται αλλαγή από τον διακομιστή χωρίς να γίνεται καμιά ενημέρωση στα αντίγραφα που διατηρούν οι κόμβοι. Αυτό έχει ως αποτέλεσμα την ύπαρξη διαφορετικών εκδόσεων με το ίδιο αναγνωριστικό. Στα μη κεντρικοποιημένα συστήματα όπως η Gnutella δεν εφαρμόζεται καμιά πολιτική για την ενημέρωση των αντιγράφων.

Στο FreeNet χρησιμοποιείται ευριστική τεχνική για την ενημέρωση των αντιγράφων, χωρίς όμως εγγύηση όμως ότι θα ενημερωθούν όλα τα αντίγραφα. Μια ενημέρωση από τον "κάτοχο" των δεδομένων δρομολογείται προς τα πίσω στη διαδρομή που κρατήθηκαν τα αντίγραφα κατά την ανάκτηση των δεδομένων.

Κάθε ενημέρωση στο OceanStore δημιουργεί μια νέα έκδοση του αντικειμένου εφαρμόζοντας ένα Byzantine πρωτόκολλο.

Για αποτελεσματικότερους τρόπους ενημέρωσης χρησιμοποιήθηκαν οι επιδημικοί αλγόριθμοι και προτάθηκαν προσεγγίσεις όπως ο Push&Pull αλγόριθμος [25]. Οι κόμβοι που διατηρούν αντίγραφα αποτελούν ένα λογικό τμήμα του χώρου των δεδομένων και είναι συνδεδεμένοι μεταξύ τους ("γνωστοί") ως ένα βαθμό. Η ενημέρωση γίνεται σε δύο φάσεις, μέσω του μηχανισμού *διάδοσης της φήμης (rumor spreading).*

> Η πρώτη φάση, ***φάση push***, ξεκινά από τον κόμβο που εισάγει την αλλαγή και αφού ενημερώσει το αντίγραφό το προωθεί σε ένα υποσύνολο από τους "γνωστούς" του γείτονες, οι οποίοι με την σειρά τους διαδίδουν (την φήμη) το νέο σε υποσύνολο των γνωστών τους γειτόνων, παρόμοια με την πλημμύρα με παράμετρο TTL.
>
> Η δεύτερη φάση, ***φάση pull***, ξεκινά από έναν κόμβο που θέλει να ενημερώσει το αν το αντίγραφό του. Η ανάγκη αυτή μπορεί να οφείλεται σε δυο λόγους: είτε γιατί ο κόμβος ήταν εκτός δικτύου, είτε γιατί έλαβε μια pull αίτηση και δεν είναι σίγουρος ότι έχει το πιο πρόσφατο αντίγραφο.

Οι δύο φάσεις είναι ακολουθιακές, αλλά μπορεί να επικαλυφθούν στην συνέχεια.

5.2.2. Ενημέρωση σε δομημένα συστήματα

Στα δομημένα συστήματα συνήθως τα αντίγραφα τοποθετούνται στους ενδιάμεσους κόμβους που βρίσκονται πάνω στο μονοπάτι της διαδρομής μεταξύ του κόμβου που υποβάλλει το ερώτημα και του κόμβου που έχει το δεδομένο. Η τακτική αυτή αναφέρεται ως Path Caching with Expiration (PCX) [26] αφού τα δεδομένα που γίνονται cache θα πρέπει κάποια στιγμή να ανανεωθούν. Η διαδικασία της ενημέρωσης απαιτεί γνώση για το ποιες καταχωρήσεις είναι αναγκαίο να ενημερωθούν όπως επίσης αν υπάρχει ενδιαφέρον για ενημέρωση (για παράδειγμα δεν υφίστανται πια).

Μια στρατηγική για ασύγχρονη ενημέρωση, που μπορεί να εφαρμοσθεί τόσο σε δομημένα όσο και σε αδόμητα συστήματα, αποτελεί το πρωτόκολλο Ελεγχόμενης Διάδοσης ενημερώσεων (Controlled Update Propagation - CUP) [26]. Το CUP δημιουργεί προσωρινά αντίγραφα (cache) των καταχωρήσεων του ευρετηρίου, καθώς απαντά σε ερωτήσεις αναζήτησης. Στη συνέχεια διαδίδει τις ενημερωμένες καταχωρήσεις του ευρετηρίου για διατήρηση αυτών των caches.

Το CUP διατηρεί δύο κανάλια: ένα για την προώθηση των μηνυμάτων ερώτησης και ένα για προώθηση μηνυμάτων ενημέρωσης ασύγχρονα. Οι ερωτήσεις διασχίζουν το μονοπάτι προς τον κάτοχο κόμβο του δεδομένου ενώ οι ενημερώσεις κάνουν την αντίστροφη διαδρομή. Η ενημέρωση μπορεί να αφορά την διαγραφή, την προσθήκη ή την ανανέωση δεδομένων.

Εφαρμόζοντας PCX στρατηγική ενημέρωσης επιτυγχάνεται εξισορρόπηση φορτίου ερωτήσεων για δημοφιλή δεδομένα, μείωση latency, και ανακούφιση από hot spots. Το πρωτόκολλο CUP, όπως ισχυρίζονται οι δημιουργοί του μειώνει κατά 2 έως 300 φορές το overhead του δικτύου.

6. Ασφάλεια, Ανωνυμία, Έλεγχος Πρόσβασης

Σε αυτή την ενότητα αναπτύσσονται θέματα σχετικά με την ασφάλεια, ανωνυμία, εμπιστοσύνη και έλεγχο πρόσβασης στα peer-to-peer συστήματα.

Η *ασφάλεια* [1] είναι ιδιαίτερη απαίτηση στα peer-to-peer συστήματα, καθώς υπάρχει ανάγκη για διαθεσιμότητα, μυστικότητα, εμπιστοσύνη, ακεραιότητα, αυθεντικότητα. Εξ αιτίας της αυτονομίας και της ανοικτής δομής τους τα συστήματα αυτά είναι ιδιαίτερα ευάλωτα σε επιθέσεις.

Το ενδιαφέρον επικεντρώνεται στην ασφαλή αποθήκευση δεδομένων και στην ασφαλή δρομολόγηση[13]. Η ασφαλή αποθήκευση με την χρήση παραδοσιακών και νέων πρωτοκόλλων και αλγορίθμων κρυπτογραφίας (ψηφιακές υπογραφές, multi-key encryption, firewalls) με σκοπό να χτισθεί εμπιστοσύνη μεταξύ των κόμβων και των κοινόχρηστων αντικειμένων. Η ασφαλή δρομολόγηση επιτυγχάνεται με τεχνικές ελέγχου αυθεντικότητας και ακεραιότητας, την ασφαλή αντιστοίχηση ID, κλπ.

Το χαρακτηριστικό της *ανωνυμίας* επιτρέπει στους χρήστες να χρησιμοποιούν το peer-to-peer σύστημα διατηρώντας όμως την ανωνυμία τους, αποφεύγοντας έτσι τυχόν νομικού περιορισμούς (πρόβλημα που αντιμετώπισε το Napster). Σύμφωνα με τον Diegledine (et al. 2000)[1,13] η ανωνυμία αναφέρεται στο συγγραφέα (ή εκδότη), στην ταυτότητα του κόμβου και του αντικειμένου και στις λεπτομέρειες της ερώτησης.

Για την επίτευξη της ανωνυμίας χρησιμοποιούνται τεχνικές[1,13] όπως multicasting, Covert paths, ανώνυμες συνδέσεις κλπ.

Ο έλεγχος πρόσβαση, η πιστοποίηση, η διαχείριση ταυτοτήτων είναι θέματα που δεν έχει δοθεί ιδιαίτερη σημασία [1]. Αφού το περιβάλλον είναι κατανεμημένο είναι δυνατό οι ίδιες φυσικές οντότητες να εμφανίζονται με διαφορετικές ταυτότητες. Τελικά ο έλεγχος πρόσβασης και η πιστοποίηση προκύπτει από την κατανεμημένη δομή όπου η ευθύνη και ο έλεγχος περνά στους κόμβους.

7. Σύνοψη

Τα peer-to-peer συστήματα είναι πραγματικότητα και ώριμα για να παρέχουν ένα άλλο τρόπο σχεδίασης δικτυακών εφαρμογών. Απαρτίζονται από διασυνδεδεμένους κόμβους κατανεμημένους ανά τον κόσμο και σχηματίζουν ένα overlay δίκτυο. Αυτοδιοργανώνονται σε ευέλικτες τοπολογίες δικτύου με σκοπό την κοινή χρήση πόρων (δεδομένα, αποθήκευση, κύκλους μηχανής). Η ευρεία αποδοχή τους και η πληθώρα εφαρμογών τους βασίζεται στην αυτοδιοργάνωση σε ένα δυναμικό περιβάλλον όπου οι κόμβοι έρχονται και φεύγουν, στην δυνατότητα για άμεση επικοινωνία μεταξύ των χρηστών, την ιδιότητα της κλιμάκωσης, στην αυτονομία και ανωνυμία.

Τα peer-to-peer συστήματα διακρίνονται σε δύο μεγάλες κατηγορίες: τα δομημένα συστήματα που η οργάνωσή τους στηρίζεται κυρίως σε κατανεμημένους πίνακες κατακερματισμού και στα αδόμητα συστήματα όπου δεν υπάρχει καμιά συσχέτιση μεταξύ της θέσης των δεδομένων και της τοπολογίας του δικτύου.

Δύο κύρια ζητήματα μελετούνται στην παρούσα εργασία, χωρίς φυσικά να εξαντλούνται τα θέματα που σχετίζονται με τα peer-to-peer συστήματα , η αναζήτηση της επιθυμητής πληροφορίας και η δημιουργία αντιγράφων.

Η αναζήτηση ή εντοπισμός της επιθυμητής πληροφορίας βασίζεται σε κλασικές προσεγγίσεις όπως κεντρικοποιημένη, ιεραρχική, πλημμύρα ή σε μεθοδικές στρατηγικές που εκμεταλλεύονται την τοπολογία του δικτύου και την τοπική πληροφορία όπως DHTs ή σε αυθαίρετη επιλογή υποσυνόλου για αναζήτηση όπως

Η εύρεση αποδοτικών και ευέλικτων αλγορίθμων για τον εντοπισμό της πληροφορίας είναι ένα θέμα ερευνητικά ανοικτό για τα peer-to-peer συστήματα. Άμεσα συνδεδεμένο είναι και το ζήτημα της δημιουργίας αντιγράφων έτσι ώστε το σύστημα αν είναι ανθεκτικό στις αποτυχίες, αξιόπιστο, ασφαλές και με κλιμακούμενη απόκριση.

8. Αναφορές

[1] S. Androutsellis-Theotokis, D. Spinellis, *"A Survey of Peer-to-Peer Content Distribution Technologies*", *ACM Computing Surveys, pp 335-371, December 2004.*

[2] H. Balakrishnan, M. Frans Kaashoek, D. Karger, R. Morris, and I. Stoica, "*Looking Up Data in P2P Systems*," Communications of the ACM, 46(2), February 2003.

[3] Computer για όλους, *"Peer To Peer Computing*", Αριθμός Τεύχους: 202, 1/6/2001

[4] X. Zhang, **Q. Zhang**, Z. Zhang, G. Song and W. Zhu, "*A Construction of Locality-Aware Overlay Network: mOverlay and its performance*", *IEEE JSAC Special Issue on Recent Advances on Service Overlay Networks*, Jan. 2004.

[5] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, "*A scalable content addressable network*," in *Proceedings of the ACM SIGCOMM*, 2001, pp. 161–172.

[6] 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 SIGCOMM 2001.

[7] A. Crespo and H. Garcia-Molina. "*Routing indices for peer-to-peer systems"* Submitted to ICDCS 2002, October 2001.

[8] Q. LV, P. Cao, E. Cohen, K. Li, and S. Shenker. Search and replication in unstructured peer-to-peer networks. In *Proc. 16th ACM International Conference on Supercomputing (ICS'02)*, June 2002.

[9] D. Tsoumakos and N. Roussopoulos, "*A Comparison of Peer-to-Peer Search Methods*", in WebDB03, 2003

[10] K. Sripanidkulchai, B. Maggs, H. Zhang, "*Efficient Content Location Using Interest-Based Locality in Peer-to-Peer Systems"*, *IEEE INFOCOM'03*

[11] E. Cohen, A. Fiat, H. Kaplan, *"Associative Peer to Peer Networks: Harnessing Latent Semantics"*, IEEE INFOCOM 2003

[12] A. Crespo and H. Garcia-Molina. *"Semantic overlay networks for p2p systems"*, Technical report, Computer Science Department, Stanford University, October 2002

[13] D. S. Milojicic, V. Kalogeraki, R. Lukose, K. Nagaraja, J. Pruyne, B. Richard, S. Rollins, Z. Xu, "*Peer-to-Peer Computing",* HP Laboratories 2002

[14] A. Rowstron and P. Druschel, "*Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems*," in *Proceedings of the Middleware*, 2001.

[15] B. Y. Zhao, L. Huang, J. Stribling, S. C. Rhea, A. D. Joseph, and J. D. Kubiatowicz, "*Tapestry: A resilient global-scale overlay for service deployment*," *IEEE Journal on Selected Areas in Communications*, vol. 22, no. 1, pp. 41–53, January 2004.

[16] P. Maymounkov and D. Mazieres, "*Kademlia: A peer-to-peer information system based on the xor metric*," in *Proceedings of the IPTPS*, Cambridge, MA, USA, February 2002, pp. 53–65.

[17] D. Malkhi, M. Naor, and D. Ratajczak, "*Viceroy: a scalable and dynamic emulation of the butterfly*," in *Proceedings of the ACM PODC 2002*, Monterey, CA, USA, July 2002, pp. 183–192.

[18] *Napster*. http://www.napster.com/

[19] (2001) *Fasttrack* http://www.fasttrack.nu/

[20] (2001) *Kazaa*. http://www.kazaa.com/

[21] (2003) *Bittorrent*. http://bitconjurer.org/BitTorrent/

[22] http://en.wikipedia.org/wiki/Gnutella

[23] C. Tang, Z. Xu, and S. Dwarkadas, "*Peer-to-Peer Information Retrieval Using Self-Organizing Semantic Overlay Networks*". In *ACM SIGCOMM 2003*, Aug. 2003.

[24] O. D. Sahin, A. Gupta D. Agrawal A. El Abbadi, *"A Peer-to-peer Framework for Caching Range Queries"*, In *Proc. 20th Int. Conf. on Data Eng*. (ICDE), 2004.

[25] K. Datta, A. Hauwirth, M. Aberer, "*Updates in Highly Unreliable, Replicated Peer-to-Peer Systems*", ICDCS04]

[26] M. Roussopoulos and M. Baker, "*CUP: Controlled Update Propagation in Peer-to-Peer Networks*." *The Proceedings of the 2003 Annual USENIX Technical Conference,* June 2003.

[27] V. Gopalakrishnan, B. Silaghi, B. Bhattacharjee, P. Keleher, *"Adaptive Replication in Peer-to-Peer Systems*", Department of Computer Science University of Maryland, College Park, ICDCS 2004

[28] P. Ganesan, B. Yang, H. Garcia-Molina, *"One Torus to Rule them All: Multi-dimensional Queries in P2P"*, WebDB 2004, June 2004.

[30] A. Bharambe, M. Agrawal, S. Seshan, "*Mercury: Supporting Scalable Multi-Attribute Range Queries*", In Proceedings of the SIGCOMM'04, USA

[31] V. Papadimos, D. Maier, K. Tufte Distributed Query Processing and Catalogs for Peer-to-Peer Systems, Proceedings of the 2003 CIDR Conference, USA, April 2004

[32] M. Waldman, A. D. Rubin, and L. F. Cranor, "Publius: a robust, tamperevident, censorship-resistant, web publishing system," in *Proceedings of the Ninth USENIX Security Symposium*, Denver, CO, USA, 2000.

[33] Clarke, O. Sandberg, B. Wiley, and T. W. Hong. (1999) Freenet: A distributed anonymous information storage and retrieval system. http://freenetproject.org/ freenet.pdf