ΣΥΣΤΗΜΑΤΑ ΟΜΟΤΙΜΩΝ ΚΟΜΒΩΝ

ΣΤΑΜΚΟΠΟΥΛΟΣ ΚΩΝΣΤΑΝΤΙΝΟΣ

ΠΕΡΙΛΗΨΗ

Τα τελευταία χρόνια έχει γίνει μεγάλη έρευνα για τα Συστήματα Ομότιμων Κόμβων (P2P systems). Με τον όρο αυτό εννοούμε μία κατηγορία υπολογιστικών συστημάτων που χρησιμοποιούνται για να διεκπεραιώσουν μία εργασία με την κατανομή της σε πολλές υπολογιστικές μονάδες (κόμβους). Πρόκειται για ιδεατά δίκτυα υπολογιστών όπου ο κάθε κόμβος είναι αυτόνομος και οι κόμβοι είναι ισότιμοι μεταξύ τους. Υπάρχουν πολλά είδη διαφορετικών P2P συστημάτων το καθένα με διαφορετικές ιδιότητες. Οι δύο μεγάλες κατηγορίες P2P συστημάτων είναι τα κεντρικοποιημένα και τα μη-κεντρικοποιημένα. Τα τελευταία χωρίζονται σε δύο μεγάλες υποκατηγορίες τα δομημένα και τα μη-δομημένα. Μια ειδική κατηγορία μη-κεντρικοποιημένων P2P συστημάτων είναι τα σημασιολογικά ιδεατά δίκτυα.

Στην επισκόπηση αυτή θα περιγραφούν συνοπτικά οι βασικές ιδιότητες των P2P συστημάτων όλων αυτών των κατηγοριών και κυρίως των μη-κεντρικοποιημένων, οι μέθοδοι κατασκευής τους, οι τεχνικές αναζήτησης και τρόποι που υπάρχουν για δημιουργία και ενημέρωση αντιγράφων.

1. ΕΙΣΑΓΩΓΗ

Ένα P2P σύστημα είναι μια συλλογή από αυτόνομους κόμβους που συνεργάζονται για να διεκπεραιώσουν μία μεγάλη εργασία. Όλοι αυτοί οι κόμβοι αποτελούν ένα κατανεμημένο σύστημα, ένα ιδεατό δίκτυο και διαμοιράζονται πόρους, ανταλλάσσουν αρχεία μεταξύ τους ή συμμετέχουν στην επίλυση ενός μεγάλου προβλήματος. Για το λόγο ότι τα P2P συστήματα αποτελούνται από αυτόνομους κόμβους, από διαφορετικούς δηλαδή χρήστες, δεν είναι απαραιτήτως και αξιόπιστα. Επειδή μιλάμε για ιδεατά δίκτυα, δύο γειτονικοί κόμβοι δεν είναι απαραίτητα και γείτονες στο φυσικό δίκτυο, δεν υπάρχει δηλαδή απαραίτητα φυσικό μέσο που ενώνει άμεσα αυτούς τους δύο κόμβους.

Υπάρχουν δύο μεγάλες κατηγορίες P2P συστημάτων, τα κεντρικοποιημένα και τα μη-κεντρικοποιημένα. Στα πρώτα υπάρχει ένας συγκεντρωτικός κατάλογος τοποθετημένος σε έναν ή περισσότερους κόμβους (π.χ. Napster [10]). Κάποιος κόμβος για να βρει ένα αντικείμενο απευθύνεται σε αυτούς τους κόμβους οι οποίοι του επιστρέφουν την θέση του

αντικειμένου στο δίκτυο και στη συνέχεια ο κόμβος επικοινωνεί απευθείας με τον κόμβο που έχει το αντικείμενο. Η κατασκευή όπως και η αναζήτηση σε τέτοια κεντρικοποιημένα συστήματα είναι τετριμμένη: όταν κάποιος κόμβος θέλει να εισαχθεί στο δίκτυο απλά δημοσιεύει τα αντικείμενά του στον κεντρικό κατάλογο και για την αναζήτηση απαιτείται απλώς μία ερώτηση στον κεντρικό κατάλογο. Για το λόγο αυτό τα κεντρικοποιημένα P2P συστήματα δεν θα αναλυθούν περισσότερο. Παρόλο όμως την απλότητα τους, αυτά τα συστήματα πάσχουν από δυνατότητα κλιμάκωσης και έχουν ένα σημείο αποτυχίας (single point of failure).

Η επισκόπηση θα περιγράψει συνοπτικά τη δεύτερη μεγάλη κατηγορία, τα μη-κεντρικοποιημένα P2P συστήματα. Σε αυτά δεν υπάρχει κάποιος κεντρικός κατάλογος με την πληροφορία για την τοποθεσία των αντικειμένων. Οι δύο μεγάλες υποκατηγορίες που υπάρχουν εδώ είναι τα δομημένα και τα μη-δομημένα P2P συστήματα. Στα δομημένα υπάρχει μια δομή στην τοπολογία του δικτύου, είτε χαλαρή (π.χ. Freenet [13]), είτε αυστηρή (π.χ. CAN [1], Chord [2]). Σε μια χαλαρή τοπολογία η τοποθεσία ενός αντικειμένου προσδιορίζεται από “hints”, ενώ σε μια αυστηρή τοπολογία μπορεί να προσδιοριστεί επακριβώς, κάνοντας την απάντηση ερωτήσεων πολύ αποτελεσματική. Όμως στα δομημένα συστήματα οι κόμβοι χάνουν κατά κάποιο τρόπο την ιδιότητα της αυτονομίας τους γιατί αναγκάζονται να αποθηκεύουν ξένα αντικείμενα. Στα μη-δομημένα συστήματα οι κόμβοι εισάγονται στο δίκτυο με τυχαίο τρόπο ή σχεδόν με τυχαίο τρόπο ακολουθώντας κάποιους χαλαρούς κανόνες. Η τοπολογία που προκύπτει είναι τυχαία στην πρώτη περίπτωση ή έχει κάποιες συγκεκριμένες ιδιότητες στην δεύτερη αλλά η τοποθεσία των αντικειμένων δεν μπορεί να προσδιοριστεί με βάση αυτή την τοπολογία και συνήθως η αναζήτησή τους γίνεται με τυφλή αναζήτηση, προωθώντας την ερώτηση από κόμβο σε κόμβο. Μια άλλη ειδική κατηγορία μη-κεντρικοποιημένων P2P συστημάτων είναι τα σημασιολογικά ιδεατά δίκτυα όπου οι κόμβοι οργανώνονται έτσι ώστε αυτοί που έχουν σημασιολογικά όμοια αντικείμενα να βρίσκονται στην ίδια περιοχή του ιδεατού δικτύου.

Στην επισκόπηση αυτή θα περιγραφούν πως κατασκευάζονται μη κεντρικοποιημένα P2P συστήματα, αλγόριθμοι αναζήτησης που αυτά χρησιμοποιούν, μερικές βασικές ιδιότητες όπως αυτή της κλιμάκωσης (η δυνατότητα τα συστήματα να μπορούν να κρατάνε τις ιδιότητές τους ενώ μεγαλώνουν σε μέγεθος), θα αναφερθούν θέματα επίδοσης και επίσης τρόποι για την δημιουργία και την ενημέρωση αντιγράφων.

Το υπόλοιπο της επισκόπησης οργανώνεται ως εξής: στην Ενότητα 2 περιγράφονται τα δομημένα P2P συστήματα δίνοντας δύο αντιπροσωπευτικά παραδείγματα τέτοιων, το CAN και το Chord. Στην Ενότητα 3 περιγράφονται τα μη-δομημένα P2P συστήματα, ενώ στην Ενότητα 4 τα σημασιολογικά ιδεατά δίκτυα. Στην Ενότητα 5 αναφέρονται μερικά πρωτόκολλα και αλγόριθμοι για τον χειρισμό πολυδιάστατων ερωτήσεων εύρους και τέλος στην Ενότητα 6 αναφέρονται κάποια συμπεράσματα και ο επίλογος της επισκόπησης.

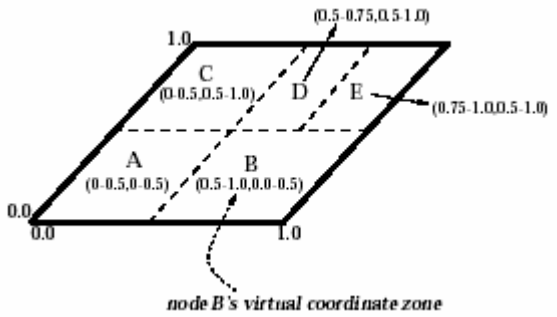
2. ΔΟΜΗΜΕΝΑ P2P ΣΥΣΤΗΜΑΤΑ

Τα δομημένα P2P συστήματα κατασκευάζονται έτσι ώστε να έχουν μία προκαθορισμένη τοπολογία ώστε οι αναζητήσεις να γίνονται σε λίγο χρόνο. Για να γίνει αυτό χρησιμοποιούνται πίνακες κατακερματισμού (Distributed Hash Tables, DHT) οι οποίοι αντιστοιχίζουν κλειδιά σε τιμές. Η θέση ενός αντικειμένου Α στο δίκτυο βρίσκεται κατακερματίζοντας το κλειδί του Α. Αυτό που πληρώνουμε όμως για να έχουμε αποτελεσματική αναζήτηση είναι μεγαλύτερο κόστος κατασκευής του δικτύου. Στη συνέχεια της ενότητας αναλύονται δύο χαρακτηριστικά παραδείγματα τέτοιων συστημάτων.

2.1. Το δίκτυο CAN (Content-Addressable Network)

Περιγραφή

Το CAN [1] είναι ένας μηχανισμός δεικτοδότησης αντικειμένων. Αυτό γίνεται ως εξής: θεωρούμε ένα εικονικό σύστημα συντεταγμένων d-διαστάσεων οργανωμένο σε d-torus. Ο χώρος αυτός διαμοιράζεται δυναμικά στους κόμβους του συστήματος κι έτσι κάθε κόμβος αναλαμβάνει μία ζώνη του χώρου. Ένα ζεύγος κλειδί-τιμή *(K, V)* αντιστοιχίζεται στο σημείο *P* του χώρου χρησιμοποιώντας μία ομοιόμορφη συνάρτηση κατακερματισμού και αποθηκεύεται στον κόμβο ο οποίος είναι υπεύθυνος για την ζώνη εκείνη που περιέχει το σημείο *P*. Ο κάθε κόμβος επίσης αποθηκεύει μια λίστα με τους γείτονές του καθώς και τα όρια των ζωνών που αυτοί καλύπτουν. Στο Σχήμα 1, φαίνεται ένα παράδειγμα ενός δισδιάστατου χώρου που είναι χωρισμένος σε πέντε ζώνες. Στο συγκεκριμένο σχήμα η ζώνη *A* είναι υπεύθυνη για τα σημεία (x,y) με $0 \leq x \leq 0.5$ και $0 \leq y \leq 0.5$.



Σχήμα 1: Ένας δισδιάστατος χώρος χωρισμένος σε πέντε ζώνες

Κατασκευή

Ο προς εισαγωγή κόμβος *A* επιλέγει τυχαία ένα σημείο *P* του χώρου. Ο κόμβος *B* που είναι υπεύθυνος για το σημείο *P* διαιρεί την ζώνη του σε δύο ίσα κομμάτια και ο καθένας από τους *A* και *B* αναλαμβάνει το ένα από τα δύο κομμάτια. Τα ζεύγη που αντιστοιχίζονται στην ζώνη

που μόλις ανέλαβε ο *A* μεταφέρονται από τον *B* στον *A*. Στην περίπτωση αποχώρησης του κόμβου *A*, εθελοντικής ή μη, υπάρχουν αλγόριθμοι που ανακατανέμουν την ζώνη του *A* στους γείτονές του.

Αναζήτηση

Στην περίπτωση που κάποιος κόμβος *A* ψάχνει για ένα αντικείμενο με κλειδί *K*, εφαρμόζει την ομοιόμορφη συνάρτηση κατακερματισμού στο *K*, οπότε προκύπτει το σημείο *P* του ζεύγους *(K, V)*. Αν το *P* δεν ανήκει στη ζώνη του *A* αλλά σε έναν άλλο κόμβο *B*, τότε η αίτηση δρομολογείται σε εκείνο το γείτονα του *A* ο οποίος είναι πιο κοντά στον *B* όσον αφορά την απόσταση στο σύστημα συντεταγμένων. Το μήκος του μονοπατιού από τον *A* στον *B* είναι κατά μέσο όρο $(d/4)\times N^{1/d}$, όπου *N* ο αριθμός των κόμβων. Κάθε κόμβος κρατάει κατά μέσο όρο *2d* γείτονες. Τα δύο παραπάνω αποτελέσματα δείχνουν την δυνατότητα κλιμάκωσης του CAN.

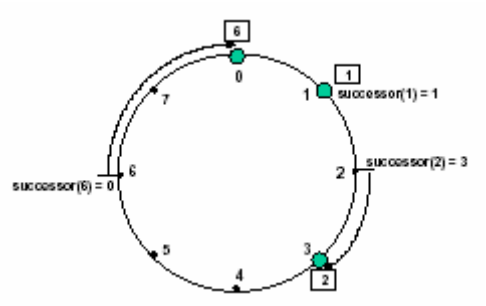
Βελτιώσεις

Για την μείωση του χρόνου απόκρισης και του μήκους μονοπατιού υπάρχουν κάποιες βελτιώσεις που μπορούν να γίνουν στο CAN. Αυτές είναι α) η χρήση πολλών χώρων συντεταγμένων, β) η χρήση πολλών συναρτήσεων κατακερματισμού και γ) η υπερφόρτωση ζωνών. Μια ζώνη είναι υπερφορτωμένη όταν ανατίθεται σε περισσότερους από έναν κόμβους.

2.2. Το δίκτυο Chord

Περιγραφή

Στο Chord [2], στους κόμβους και στα κλειδιά ανατίθεται ένα αναγνωριστικό μήκους *m bit*. Οι κόμβοι σε ένα δίκτυο Chord οργανώνονται πάνω σε έναν δακτύλιο 2^m θέσεων $[0..2^m-1]$(όσα και τα διαφορετικά αναγνωριστικά που μπορούν να υπάρξουν). Κάθε κόμβος που εισάγεται στο δίκτυο καταλαμβάνει μία από τις 2^m θέσεις. Ένας δακτύλιος του Chord με 8 θέσεις φαίνεται στο Σχήμα 2.



Σχήμα 2: Δακτύλιος του Chord

Κατασκευή

Η τοποθέτηση ενός κόμβου ή ενός αντικειμένου στον δακτύλιο του Chord, δηλαδή η εύρεση του αναγνωριστικού τους, γίνεται κατακερματίζοντας την IP διεύθυνση ή το κλειδί τους αντίστοιχα, με μία "consistent" συνάρτηση κατακερματισμού. Μία "consistent" συνάρτηση κατακερματισμού εγγυάται ότι με πολύ μεγάλη πιθανότητα όλοι οι κόμβοι θα αναλάβουν τον ίδιο αριθμό κλειδιών. Η ανάθεση των κλειδιών στους κόμβους γίνεται ως εξής: αφού βρεθεί το αναγνωριστικό του κλειδιού, αυτό ανατίθεται στον κόμβο που βρίσκεται στη θέση του δακτυλίου με αυτό το αναγνωριστικό. Αν δεν υπάρχει κόμβος σε εκείνη τη θέση, τότε το κλειδί ανατίθεται στον κόμβο που έχει την ακριβώς επόμενη θέση στο δακτύλιο. Αν υποθέσουμε ότι στο δίκτυο συμμετέχουν N κόμβοι τότε κάθε κόμβος κρατάει πληροφορίες δρομολόγησης (δείκτες) για άλλους $O(logN)$ κόμβους και επιλύει τις αναζητήσεις σε $O(logN)$ "hops". Για κάθε κόμβο, αυτοί οι δείκτες δείχνουν σε επόμενους από αυτόν κόμβους οι οποίοι βρίσκονται σε αποστάσεις που ισούνται με τις δυνάμεις του δύο. Σε μία εισαγωγή ή μία αποχώρηση ενός κόμβου οι πληροφορίες δρομολόγησης πρέπει να κρατούνται συνεπείς. Αυτό επιτυγχάνεται με ανταλλαγή $O(log^2N)$ μηνυμάτων. Τα τρία παραπάνω αποτελέσματα υποδεικνύουν την καλή κλιμάκωση που έχει ένα δίκτυο Chord. Επίσης για λόγους αξιοπιστίας και ανοχής σε πολλαπλές αποτυχίες, κάθε κόμβος κρατάει και μια λίστα με μερικούς άμεσους διαδόχους (που βρίσκονται συνεχόμενα στον δακτύλιο).

Αναζήτηση

Η θέση ενός αντικειμένου στο δακτύλιο του Chord βρίσκεται κατακερματίζοντας το κλειδί του αντικειμένου. Αν σε εκείνη τη θέση δεν υπάρχει κάποιος κόμβος (δεν έχει κατακερματιστεί κάποιος κόμβος) τότε ξέρουμε ότι το αντικείμενο βρίσκεται αποθηκευμένο στον πρώτο κόμβο που θα συναντήσουμε προχωρώντας στον δακτύλιο με τη φορά του ρολογιού. Οπότε ακολουθώντας τους δείκτες η αίτηση μεταφέρεται προς το κόμβο που κατέχει το αντικείμενο σε $O(logN)$ *"hops"* όπως αναφέρθηκε και προηγουμένως. Βέβαια αυτό δεν είναι τετριμμένο για το πώς γίνεται γιατί δεν γνωρίζουμε ακριβώς τον επόμενο από την θέση κόμβο, οπότε ψάχνουμε πρώτα τον προηγούμενο και μετά μεταβαίνουμε στον επόμενο αυτού που θα είναι κι ο ζητούμενος. Για παράδειγμα, στο Σχήμα 2, έστω ο κόμβος *3*

αναζητά ένα αντικείμενο που έχει κατακερματιστεί στη θέση *6* του δακτυλίου. Τότε επειδή δεν υπάρχει κόμβος στη θέση *6* πρέπει να βρούμε τον επόμενο κόμβο της θέσης *6* (τον κόμβο *0*). Η εύρεση του κόμβου *0* γίνεται ως εξής: αρχικά ο κόμβος *3* ψάχνει τον προηγούμενο κόμβο της θέσης 6. Αυτό γίνεται με την βοήθεια των δεικτών δρομολόγησης. Το αποτέλεσμα της αναζήτησης είναι ο κόμβος *2*, οπότε μεταβαίνοντας στον επόμενο κόμβο του κόμβου *2* έχουμε το αποτέλεσμα.

2.3. Δημιουργία αντιγράφων

Σε πολλές περιπτώσεις για λόγους διαθεσιμότητας ενός αντικείμενου, ή λόγους αποσυμφόρησης ενός κόμβου και γενικότερα λόγους απόδοσης, είναι απαραίτητη η δημιουργία πολλών αντιγράφων ενός αντικείμενου σε διαφορετικούς κόμβους. Το αντίτιμο βέβαια που πληρώνουμε είναι το κόστος για την διατήρηση της συνέπειας των αντιγράφων. Για τα δομημένα συστήματα που αναφέρθηκαν, υπάρχουν οι παρακάτω τρόποι δημιουργίας αντιγράφων. Φυσικά εκτός από αυτούς, σε όλες τις περιπτώσεις μπορεί να εφαρμοστεί “caching” με το οποίο κάποιος κόμβος κρατάει πληροφορίες από κόμβους που του απαντάνε ώστε να τις χρησιμοποιήσει σε επόμενες ερωτήσεις.

CAN

Στο συγκεκριμένο δίκτυο, μπορούν να χρησιμοποιηθούν οι βελτιώσεις που αναφέρθηκαν για τη δημιουργία αντιγράφων. Πιο συγκεκριμένα, στη βελτίωση με τις πολλές πραγματικότητες μπορούμε να έχουμε ένα αντίγραφο σε κάθε πραγματικότητα, όπως και στη βελτίωση με τις πολλαπλές συναρτήσεις κατακερματισμού όπου τοποθετείται ένα αντίγραφο σε κάθε διαφορετικό κόμβο που δίνουν οι διαφορετικές συναρτήσεις. Επίσης στην περίπτωση της υπερφόρτωσης των ζωνών μπορούμε να τοποθετήσουμε αντίγραφα σε κάθε έναν από τους κόμβους που αναλαμβάνουν μία συγκεκριμένη ζώνη. Τέλος, αν ένα αντικείμενο είναι πολύ δημοφιλές, τότε μπορούμε να δημιουργήσουμε αντίγραφα σε κόμβους γειτονικούς του κόμβου που το περιέχει. Έτσι δημιουργείται μία περιοχή του CAN που περιέχει τα ίδιο αντίγραφο.

Chord

Στο Chord αυτό που μπορεί να γίνει είναι να τοποθετηθούν αντίγραφα ενός αντικείμενου σε όλους τους διάδοχους κόμβους που βρίσκονται στην λίστα που κρατάει ο κόμβος του αντικείμενου.

Ενημέρωση αντιγράφων

Όταν γίνεται μία αλλαγή σε κάποιο γνήσιο αντίγραφο ενός αντικειμένου, τότε πρέπει να ενημερωθούν όλοι οι κόμβοι που κατέχουν κάποιο αντίγραφο αυτού. Στα δύο δομημένα συστήματα που αναφέρθηκαν οι θέσεις των αντιγράφων είναι συγκεκριμένες κι έτσι η ενημέρωση τους μπορεί να γίνει πολύ εύκολα.

Στο CAN, η θέση των αντιγράφων βρίσκεται σύμφωνα με τον τρόπο που δημιουργήθηκαν. Δηλαδή, αν χρησιμοποιήθηκαν πολλαπλές πραγματικότητες, θα υπάρχει ένα αντίγραφο σε κάθε πραγματικότητα, αν χρησιμοποιήθηκαν πολλαπλές συναρτήσεις κατακερματισμού τότε τα αντίγραφα θα υπάρχουν στα ξεχωριστά σημεία που δίνουν αυτές οι συναρτήσεις. Στην περίπτωση της δημιουργίας αντιγράφων σε υπερφορτωμένες ζώνες, τα αντίγραφα βρίσκονται σε όλους τους κόμβους που είναι υπεύθυνοι για την υπερφορτωμένη ζώνη. Τέλος, στην περίπτωση δημιουργίας μιας περιοχής με αντίγραφα του ίδιου αντικειμένου, τα αντίγραφα βρίσκονται όλα στην συγκεκριμένη περιοχή και είναι εύκολο να εντοπιστούν.

Στο Chord, οι κόμβοι που κατέχουν αντίγραφα είναι αυτοί που υπάρχουν στην λίστα των άμεσων διαδόχων.

3. ΜΗ-ΔΟΜΗΜΕΝΑ P2P ΣΥΣΤΗΜΑΤΑ

Στα μη-δομημένα συστήματα, οι κόμβοι εισάγονται στο ιδεατό δίκτυο με τυχαίο τρόπο, ή σχεδόν με τυχαίο τρόπο ακολουθώντας κάποιους «χαλαρούς» κανόνες. Οπότε σε αυτά τα συστήματα, η θέση των αντικειμένων δεν ελέγχεται από το σύστημα και δεν είναι γνωστή. Έτσι το σύστημα δεν μπορεί να εγγυηθεί για την επιτυχία μιας αναζήτησης. Ένα παράδειγμα μη-δομημένου P2P συστήματος είναι το “Gnutella” [11]. Στα συστήματα αυτά, αν κάποιος κόμβος ψάχνει ένα αντικείμενο ρωτάει τους γείτονές του, αυτοί τους δικούς τους κ.ο.κ.. Αυτό βέβαια αυξάνει πολύ το φόρτο του δικτύου γιατί παράγονται πάρα πολλά μηνύματα και για αυτό το λόγο σημαντικό ρόλο παίζει η δημιουργία αντιγράφων. Στο κεφάλαιο αυτό θα αναφερθούν οι στρατηγικές αναζήτησης που υπάρχουν και τρόποι για δημιουργία και ενημέρωση αντιγράφων.

3.1. Στρατηγικές αναζήτησης

Οι στρατηγικές αναζήτησης σε ένα μη-δομημένο P2P δίκτυο μπορούν να χωριστούν σε δύο κατηγορίες, στις στρατηγικές αναζήτησης χωρίς πληροφορία ή αλλιώς τυφλές αναζητήσεις και στις στρατηγικές αναζήτησης με πληροφορία. Στην πρώτη κατηγορία μία ερώτηση απλά διαδίδεται από κόμβο σε κόμβο μέχρι να ικανοποιηθεί ενώ στην δεύτερη η επιλογή του

κόμβου στον οποίο θα διαδοθεί μία ερώτηση εξαρτάται από κάποια πληροφορία που αποθηκεύεται στον κόμβο για την θέση του αντικείμενου που αναζητάτε. Στη συνέχεια παρατίθενται οι σημαντικότερες στρατηγικές αναζήτησης των δύο κατηγοριών.

Τυφλή αναζήτηση

Πλημμύρα ή BFS

Ο αλγόριθμος της πλημμύρας [4] είναι ο πιο απλός αλλά και ο πιο δαπανηρός όσον αναφορά την κατανάλωση εύρους ζώνης του δικτύου από τα μηνύματα. Δουλεύει ως εξής: ο κόμβος που ψάχνει ένα αντικείμενο στέλνει την αίτηση σε όλους τους γείτονες του, αυτοί με τη σειρά τους σε όλους τους δικούς τους γείτονες κ.ο.κ.. Αυτό συνεχίζεται μέχρι ένα συγκεκριμένο όριο TTL (Time To Live) που συνήθως είναι ίσο με επτά. Είναι προφανές ότι αν ο γράφος του ιδεατού δικτύου περιέχει κύκλους, που είναι πολύ συνηθισμένο στα P2P συστήματα, τότε υπερφορτώνεται το δίκτυο και πολλοί κόμβοι επεξεργάζονται πολλές φορές τα ίδια μηνύματα. Ένα μειονέκτημα επίσης της πλημμύρας είναι ότι δεν μπορεί να βρει εύκολα μη δημοφιλή αντικείμενα. Από την άλλη το πλεονέκτημα της πλημμύρας είναι ότι μπορεί να βρει όλα τα πιθανά αποτελέσματα, αν δεν υπάρξει περιορισμός στο TTL.

Expanding Ring ή Iterative Deepening

Ο αλγόριθμος αυτός [3, 4]είναι μια παραλλαγή της πλημμύρας. Αυτό που γίνεται σε αυτόν τον αλγόριθμο είναι ότι εκτελούνται διαδοχικές πλημμύρες με διαφορετικό TTL το οποίο αυξάνεται από πλημμύρα σε πλημμύρα. Ο αλγόριθμος δουλεύει καλά όταν υπάρχει κάποια συνθήκη τερματισμού για τον αριθμό των αποτελεσμάτων και επαρκεί μια πλημμύρα σε μικρή έκταση.

Random Walks

Σύμφωνα με αυτήν την στρατηγική αναζήτησης [3, 4] ο κόμβος που αναζητά ένα αντικείμενο στέλνει k μηνύματα σε k τυχαία επιλεγμένους γείτονές του. Το καθένα από αυτά τα k μηνύματα ακολουθεί ένα μονοπάτι με το να επιλέγεται κάθε φορά ένας τυχαίος κόμβος για να προωθηθεί. Έτσι προκύπτουν k τυχαίοι περίπατοι. Το πότε θα σταματήσει ένας τυχαίος περίπατος μπορεί να καθοριστεί από ένα TTL ή ελέγχοντας επαναληπτικά τον κόμβο-πηγή της ερώτησης για το αν έχει ικανοποιηθεί κάποια συνθήκη τερματισμού. Τα πλεονεκτήματα αυτού του αλγορίθμου είναι ότι περιορίζει κατά πολύ το φόρτο του δικτύου (το πολύ k × TTL μηνύματα) και ότι κάνει κατά κάποιο τρόπο εξισορρόπηση του φορτίου αφού επιλέγονται τυχαία οι κόμβοι. Από την άλλη πλευρά, τα μειονεκτήματα του είναι ότι η επιτυχία της αναζήτησης εξαρτάται σημαντικά από την τοπολογία του δικτύου και την τυχαία επιλογή και

ότι ερωτήσεις για δημοφιλή αντικείμενα αντιμετωπίζονται με τον ίδιο ακριβώς τρόπο όπως και ερωτήσεις για μη-δημοφιλή αντικείμενα.

Αναζήτηση με πληροφορία

Intelligent-BFS

Στην προσέγγιση αυτή [4], κάθε κόμβος κρατάει εγγραφές (ερώτηση-γείτονας) για τις πιο πρόσφατα απαντημένες ερωτήσεις ώστε να κάνει μία διαβάθμιση των γειτόνων του ως προς την καταλληλότητά τους να εξυπηρετήσουν την ερώτηση. Ένας κόμβος για να δρομολογήσει μια ερώτηση σε κάποιον από τους γείτονές του, συσχετίζει πρώτα των ερώτηση με πιο παλιές που έχουν απαντηθεί και τη δρομολογεί ανάλογα με την αντίστοιχη διαβάθμιση που έχει κατασκευάσει. Ο αλγόριθμος αν και δίνει καλή ακρίβεια και έχει καλή επίδοση σε αναχωρήσεις και εισαγωγές κόμβων, η ακρίβεια εξαρτάται ισχυρά από την υπόθεση ότι οι κόμβοι ειδικεύονται σε συγκεκριμένα αντικείμενα.

APS (Adaptive Probabilistic Search)

Στην συγκεκριμένη στρατηγική αναζήτησης [4] κάθε κόμβος αποθηκεύει ένα ευρετήριο όπου υπάρχουν εγγραφές για κάθε αντικείμενο που έχει ζητηθεί για κάθε γείτονα. Μία τέτοια εγγραφή εκφράζει την πιθανότητα να επιλεγεί ο συγκεκριμένος γείτονας σε μελλοντική ερώτηση για το συγκεκριμένο αντικείμενο. Η αναζήτηση γίνεται με τυχαίους περίπατους, επιλέγοντας κάθε φορά τον ποιο πιθανό κόμβο για να δρομολογηθεί ο καθένας από αυτούς, σύμφωνα με τις εγγραφές. Σε περίπτωση επιτυχίας ή αποτυχίας της αναζήτησης οι εγγραφές στα ευρετήρια πρέπει να ενημερώνονται ανάλογα. Όσον αφορά την επίδοση του αλγορίθμου είναι πολύ καλή και σχετικά με την ακρίβεια αλλά και με το φορτίο που επιβαρύνει το δίκτυο.

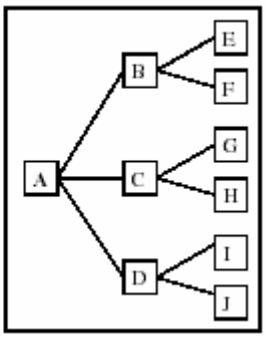
Local Indices

Στην συγκεκριμένη στρατηγική [4] κάθε κόμβος κρατάει ευρετήριο για όλα τα αντικείμενα που βρίσκονται σε μια περιοχή γύρω από τον ίδιο κι έτσι μπορεί να απαντήσει για λογαριασμό αυτών. Η αναζήτηση γίνεται με τρόπο παρόμοιο της πλημμύρας, μόνο που δεν απαιτείται όλοι οι κόμβοι να επεξεργάζονται την ερώτηση κάτι που είναι και το πλεονέκτημά του. Ο αριθμός τον μηνυμάτων που παράγονται είναι ανάλογος με αυτόν της πλημμύρας, αλλά η ακρίβειά του και το ποσοστό επιτυχίας είναι πολύ υψηλά.

Routing Indices (RI)

Η ιδέα στην συγκεκριμένη προσέγγιση [5] είναι να κρατούνται στους κόμβους ευρετήρια δρομολόγησης (Routing Indices, RI) τα οποία βοηθάνε στη λήψη της απόφασης για το ποιος είναι ο καταλληλότερος γειτονικός κόμβος για να δρομολογηθεί η ερώτηση. Για να γίνει αυτό οι γειτονικοί κόμβοι διαβαθμίζονται με βάση το πόσο καλοί είναι για να απαντήσουν την ερώτηση, ή αλλιώς με βάση το "goodness". Το μέγεθος αυτών των ευρετηρίων είναι αρκετά μικρό γιατί είναι ανάλογο του αριθμού των γειτόνων και όχι ανάλογο του αριθμού των αρχείων. Μια υπόθεση που γίνεται είναι ότι τα αρχεία μπορούν να χαρακτηριστούν και να ταξινομηθούν σε κατηγορίες με βάση το θέμα που περιέχουν. Κάθε κόμβος στο ευρετήριο του έχει καταχωρήσεις για κάθε έναν από τους γείτονές του οι οποίες εκφράζουν προσεγγίσεις για τον αριθμό των αρχείων κάθε κατηγορίας που μπορεί να ανακτήσει αν ακολουθήσει τον συγκεκριμένο γείτονα. Μία ερώτηση μπορεί να δρομολογηθεί στους καταλληλότερους γείτονες παράλληλα ή ακολουθιακά. Στο Σχήμα 3, φαίνεται ένα δίκτυο και το RI (ευρετήριο δρομολόγησης) του κόμβου *A*. Τα αρχεία θεωρούνται ότι ανήκουν σε τέσσερις κατηγορίες: databases (DB), networks (N), theory (T) και languages (L). Παρατηρώντας την πρώτη γραμμή του RI του *A*, για παράδειγμα, βλέπουμε τον αριθμό των αντικειμένων που μπορούν να ανακτηθούν αν ακολουθήσουμε τον κόμβο *B* (συνολικά 100 αντικείμενα, 20 της κατηγορίας DB, 10 της κατηγορίας T και 30 της κατηγορίας L).

Path	# docs	Documents with topics: DB	N	T	L
B	100	20	0	10	30
C	1000	0	300	0	50
D	200	100	0	100	150



Σχήμα 3: Το RI του κόμβου A

Υπάρχουν τρεις διαφορετικές προσεγγίσεις για τη δομή των ευρετηρίων και τους αλγορίθμους που χρησιμοποιούνται: "Compound RI", "Hop-count RI" και "Exponential RI". Η πρώτη προσέγγιση είναι ο απλοϊκός αλγόριθμος. Τα ευρετήρια που κρατούνται είναι όπως περιγράφτηκαν στην προηγούμενη παράγραφο ενώ η εύρεση του "goodness" ενός κόμβου γίνεται με βάση τον εξής τύπο:

$$\text{NumberofDocuments} \times \mathbf{\Pi}_i \, \text{CRI}(s_i)/\text{NumberofDocuments},$$

όπου $\text{CRI}(s_i)$ είναι η καταχώρηση του κόμβου για την κατηγορία s_i και NumberofDocuments είναι ο συνολικός αριθμός των αντικειμένων που μπορούν να ανακτηθούν ακολουθώντας τον συγκεκριμένο κόμβο. Για παράδειγμα, (σύμφωνα με στο Σχήμα 3) στην ερώτηση που αφορά

αρχεία για “databases” και “languages” η τιμή του “goodness” για τον κόμβο B είναι 100 × (20/100) × (30/100). Αυτήν η προσέγγιση δεν λαμβάνει υπόψη της τον αριθμό των μεταβάσεων από κόμβο σε κόμβο (hops) που απαιτούνται για την εύρεση των αποτελεσμάτων. Με την παραλλαγή “Hop-count RI” κρατούνται ευρετήρια για κάθε “hop” ξεχωριστά (μέχρι κάποιον ορίζοντα *H*), όπου ξεπερνιέται αυτό το πρόβλημα. Ένα “Hop-count RI” κρατάει εκτιμήσεις για το πόσα αντικείμενα μπορεί να βρει από τους γείτονές του σε 1 “hop”, 2 “hops”…, *H* “hops” Τότε όμως απαιτείται περισσότερος αποθηκευτικός χώρος για τα ευρετήρια. Με την δεύτερη παραλλαγή (Exponentially RI) αυτό διορθώνεται κρατώντας μία μόνο προσέγγιση για κάθε γείτονα για κάθε κατηγορία, λαμβάνοντας περισσότερο υπόψη τις εκτιμήσεις των κοντινότερων κόμβων. Αυτό γίνεται με βάση ένα εκτιμητή που συναθροίζει τα αποτελέσματα ενός “Hop-count” ευρετηρίου δίνοντας μεγαλύτερο βάρος στις εκτιμήσεις των κόμβων που βρίσκονται πιο κοντά. Με το τον τρόπο αυτό μειώνεται η ακρίβεια των ευρετηρίων.

Η χρήση RI μειώνει σημαντικά των αριθμό των μηνυμάτων σε σχέση με την πλημμύρα. Σε περίπτωση όμως εισαγωγών ή αποχωρήσεων κόμβων τα ευρετήρια πρέπει να ενημερώνονται. Γι αυτό η στρατηγική αυτή δεν είναι κατάλληλη για πολύ δυναμικά δίκτυα. Μια βελτίωση πάνω σε αυτό μπορεί να είναι οι περιοδικές ενημερώσεις των ευρετηρίων ανά μεγάλα διαστήματα χάνοντας σε ακρίβεια. Επίσης αυτό που πρέπει να προσεχθεί είναι η ύπαρξη κύκλων στο δίκτυο ώστε να μην υπολογίζονται λανθασμένες τιμές στις εκτιμήσεις των ευρετηρίων. Τέλος για να δουλεύει καλά αυτή η στρατηγική πρέπει και να μην γίνονται σοβαρά λάθη στις ταξινομήσεις των αρχείων στις κατηγορίες.

3.2. Δημιουργία αντιγράφων

Όπως αναφέρθηκε και στην προηγούμενη ενότητα για αποτελεσματικότερη αναζήτηση σε μη-δομημένα συστήματα είναι απαραίτητη η δημιουργία αντιγράφων. Στις δύο επόμενες υπό-ενότητες θα αναλυθεί το θέμα αυτό αρχικά από θεωρητική σκοπιά και στη συνέχεια πως η θεωρία μπορεί να εφαρμοστεί στην πράξη [3].

Θεωρητική ανάλυση

Η απόδοση της αναζήτησης εξαρτάται εκτός από την δημιουργία αντιγράφων και από την τοπολογία του δικτύου και από την κατανομή των ερωτήσεων. Έστω ότι υπάρχει μία συγκεκριμένη τοπολογία και ότι ο αριθμός των αντικειμένων του δικτύου συνολικά είναι *m* και q_i είναι η δημοτικότητα του αντικειμένου *i* ($\sum_{i=1,m} q_i = 1$). Ένα ερώτημα είναι πόσα αντίγραφα θα πρέπει να φτιαχτούν για κάθε αντικείμενο. Υπάρχουν τρεις πιθανές απαντήσεις: καταρχήν θεωρούμε ότι ο συνολικός αριθμός των αντικειμένων είναι *R* και ότι r_i είναι ο αριθμός των κόμβων που αντιγράφεται το αντικείμενο *i*, ($\sum_{i=1,m} r_i = R$). Τότε α)

Uniform: $r_i = R/m$, β) *Proportional:* $r_i \propto q_i$ και γ) *Square-root:* $r_i \propto q_i^{1/2}$. Η σύγκριση των τριών αυτών επιλογών γίνεται με βάση δύο μετρικών, το μέσο μήκος αναζήτησης *A* ($A = \sum_{i=1,m} q_i \times A_i$ με $Ai = n/ri$ όπου *n* ο αριθμός των κόμβων) και το βαθμό αξιοποίησης των αντιγράφων *U* ($U = \sum_{i=1,m} r_i \times U_i/R = 1$ με $Ui = R \times qi/ri$) που αποτυπώνει την εξισορρόπηση του φορτίου. Για τις δύο πρώτες επιλογές το μέσο μήκος αναζήτησης είναι ακριβώς το ίδιο ($A_{uniform} = A_{proportional}$). Διαφέρουν όμως στην αξιοποίηση των αντιγράφων. Στην "Uniform" επιλογή ο βαθμός αξιοποίησης των αντιγράφων είναι ανάλογος με την κατανομή των ερωτήσεων ενώ στην "Proportional" επιλογή όλα τα αντίγραφα έχουν τον ίδιο βαθμό αξιοποίησης, οπότε και βέλτιστη κατανομή του φορτίου. Επίσης στην "Proportional" επιλογή το μέσο μήκος αναζήτησης (A_i) διαφέρει για τα δημοφιλή (μικρό Ai) και τα μη-δημοφιλή αντικείμενα (μεγάλο Ai). Στην τρίτη επιλογή "Square-root", λαμβάνουμε βέλτιστο μέσο μήκος αναζήτησης A, οπότε πρέπει να προτιμάται όταν μπορεί να υλοποιηθεί. Στο Σχήμα 4 [3] είναι συγκεντρωμένα όλα τα παραπάνω.

	Uniform	Proportional	Square-Root
A	$\rho^{-1}m$	$\rho^{-1}m$	$\rho^{-1}(\sum_i \sqrt{q_i})^2$
r_i	R/m	$q_i R$	$R\sqrt{q_i}/\sum_j \sqrt{q_j}$
$A_i = n/r_i$	$\rho^{-1}m$	$(\rho q_i)^{-1}$	$\rho^{-1}\sum_j \sqrt{q_j}/\sqrt{q_i}$
$U_i = Rq_i/r_i$	$q_i m$	1	$\sqrt{q_i}\sum_j \sqrt{q_j}$

Σχήμα 4: Σύγκριση μεταξύ των τριών στρατηγικών δημιουργίας αντιγράφων

Πρακτική ανάλυση

Από την πλευρά της εφαρμογής μιας στρατηγικής δημιουργίας αντιγράφων υπάρχουν δύο εύκολοι τρόποι: "owner replication" και "path replication". Με τον πρώτο τρόπο, ένα αντικείμενο αντιγράφεται στον κόμβο που το ζήτησε μετά από μια επιτυχή αναζήτηση, ενώ με τον δεύτερο το αντικείμενο αντιγράφεται σε όλους τους κόμβους που ανήκουν στο μονοπάτι από όπου επιστρέφει η απάντηση. Ένα σύστημα που χρησιμοποιεί τυχαίους περιπάτους και "path replication" ισοδυναμεί με την "Square-root" επιλογή, μόνο που δεν κάνει καλή διασπορά των αντιγράφων. Για την αντιμετώπιση αυτού του προβλήματος, υπάρχει και ένας άλλος τρόπος, δυσκολότερος στην υλοποίηση, για τη δημιουργία αντιγράφων, ο "random replication", όπου δημιουργούνται αντίγραφα σε τυχαίους κόμβους του μονοπατιού και όχι σε ολόκληρο το μονοπάτι.

3.3. Ενημέρωση αντιγράφων

Ενώ η δημιουργία αντιγράφων βελτιώνει την απόδοση ενός P2P συστήματος, όπως αναφέρθηκε, το αντίτιμο είναι η διατήρηση της συνέπειας των αντιγράφων. Δηλαδή, όταν το

γνήσιο αντίγραφο ενός αντικειμένου μεταβληθεί τότε πρέπει να ενημερωθούν όλα τα υπόλοιπα αντίγραφα που υπάρχουν. Η ενημέρωση των αντιγράφων μπορεί να είναι σύγχρονη όπου όλα τα αντίγραφα ενημερώνονται ταυτόχρονα (είναι εγγυημένη η συνέπεια των αντιγράφων), ή ασύγχρονη όπου τα αντίγραφα μπορούν να είναι σε ασυνεπή μορφή για κάποιο χρονικό διάστημα. Η επιλογή του τρόπου με τον οποίο θα γίνει η ενημέρωση εξαρτάται από την εφαρμογή. Για παράδειγμα, σε μία βάση δεδομένων μιας αεροπορικής εταιρίας απαιτείται η ενημέρωση να είναι σύγχρονη. Σε αντίθεση, σε ένα P2P σύστημα για ανταλλαγή μουσικών αρχείων η ενημέρωση των αντιγράφων μπορεί να είναι ασύγχρονη.

Σημαντικός παράγοντας στην ενημέρωση αντιγράφων είναι ο ρυθμός με τον οποίο ανανεώνονται τα αντικείμενα αλλά και ρυθμός αποχώρησης των κόμβων από το σύστημα. λόγω κακής κατάστασης του δικτύου ή λόγω εθελούσιας αποχώρησης. Επίσης, σε ένα μη-δομημένο P2P σύστημα δεν υπάρχει ολική γνώση του συστήματος και το ποσοστό των κόμβων που είναι συνδεδεμένοι μία δεδομένη χρονική στιγμή είναι πολύ μικρό. Όλοι αυτοί οι παράμετροι πρέπει να λαμβάνονται υπόψη κατά την ενημέρωση των αντιγράφων.

Δύο μέθοδοι που παρομοιάζονται με την διάδοση μιας φήμης και έχουν τα χαρακτηριστικά αυτής είναι η *"push"* και η *"pull"* . Με την μέθοδο "push", ο κόμβος που κάνει μία ενημέρωση διαδίδει αυτό το γεγονός (φήμη) στους γειτονικούς του, με σκοπό να ενημερωθούν όσοι κόμβοι έχουν το συγκεκριμένο αντίγραφο. Με την μέθοδο "pull", ένας κόμβος που κατέχει ένα αντίγραφο επικοινωνεί με τον κόμβο που κατέχει το αυθεντικό αντίγραφο για να διαπιστώσει αν έχει την σωστή έκδοση του αντιγράφου. Στην συνέχεια παρουσιάζεται ένας υβριδικός αλγόριθμος που χρησιμοποιεί και τις δύο παραπάνω μεθόδους για την ενημέρωση αντιγράφων.

Υβριδικός push/pull αλγόριθμος

Στον αλγόριθμο αυτό [13], για την ενημέρωση των αντιγράφων, εκτελούνται διαδοχικά δύο φάσεις, η "push" και η "pull", οι οποίες μπορεί να επικαλύπτονται στο χρόνο. Στην "push" φάση, μία ενημέρωση ενός αντικειμένου διαδίδεται με έναν τρόπο αντίστοιχο της πλημμύρας. Η διαφορά της "push" διάδοσης από την πλημμύρα είναι ότι σε ένα κόμβο η "push" αίτηση προωθείται μόνο σε ένα υποσύνολο των "responsible" κόμβων, αυτών των κόμβων δηλαδή, που κατέχουν ένα αντίγραφο του συγκεκριμένου αντικειμένου και όχι στο σύνολο των γειτονικών κόμβων όπως στην πλημμύρα. Για την αποφυγή λήψης περιττών μηνυμάτων από έναν κόμβο για την ίδια ενημέρωση εφαρμόζονται οι εξής τεχνικές: α) η διάδοση της ενημέρωσης γίνεται μόνο σε έναν τυχαίο υποσύνολο f_r των γειτονικών "responsible" κόμβων, όπως αναφέρθηκε, β) η φήμη διαδίδεται σε καθέναν από τους κόμβους του υποσυνόλου f_r με κάποια πιθανότητα *PF* και γ) μία "push" αίτηση περιέχει μία λίστα *R* με τους κόμβους που τους έχει σταλεί ήδη η "push" αίτηση, ώστε να μην σταλούν επιπλέον μηνύματα σε αυτούς τους κόμβους. Ένας κόμβος που προωθεί μία "push" αίτηση προσθέτει στην λίστα *R* που

έλαβε και τους κόμβους στους οποίους έστειλε ο ίδιος “push” αιτήσεις. Η ρύθμιση των παραμέτρων f_r και *PF* σε έναν κόμβο μπορεί να γίνει με την καταμέτρηση των διπλών μηνυμάτων που λαμβάνει καθώς και με την αξιολόγηση του μεγέθους της λίστας *R*. Όσο περισσότερες φορές δεχθεί κάποιος κόμβος το ίδιο μήνυμα τόσο περισσότερο πρέπει να μειώσει τις παραμέτρους *fr* και *PF*. Το ίδιο πρέπει να κάνει και στην περίπτωση που λάβει μία “push” αίτηση με μεγάλη λίστα *R*, κάτι που σημαίνει ότι η φήμη έχει διαδοθεί σε πολλούς κόμβους, οπότε πρέπει να μειώσει τις συγκεκριμένες παραμέτρους. Αποδεικνύεται ότι ο η “push” φάση συγκλίνει και ότι όλα τα αντίγραφα τελικά ενημερώνονται. Ένας κόμβος σταματάει να διαδίδει μία φήμη όταν συνυπολογίζοντας όλες τις παραμέτρους διαπιστώσει ότι έχουν ενημερωθεί όλα τα αντίγραφα με μεγάλη πιθανότητα.

Κόμβοι που έχουν αποσυνδεθεί και ανασυνδέονται στο δίκτυο, κόμβοι που έχουν να λάβουν κάποια ενημέρωση για μεγάλο χρονικό διάστημα ή κόμβοι που δέχονται κάποια “pull” αίτηση μπαίνουν στην φάση “pull” για να συγχρονιστούν με το σύστημα και να αναθεωρήσουν την εγκυρότητα των αντιγράφων τους. Στη φάση αυτή ένας κόμβος κάνει τις εξής ενέργειες: επικοινωνεί με κόμβους που κατέχουν τα συγκεκριμένα αντίγραφα και αναζητά το πιο πρόσφατα ενημερωμένο. Μία βελτίωση της “pull” φάσης είναι κάποιος κόμβος που εισέρχεται στο σύστημα να μην στέλνει αμέσως “pull” αιτήσεις αλλά να περιμένει για κάποιο χρονικό διάστημα για να λάβει κάποια “push” αίτηση. Αποδεικνύεται ότι η “pull” φάση συγκλίνει και μάλιστα γρηγορότερα από την “push”. Παρόλα αυτά όμως η “pull” παράγει πιο πολλά μηνύματα στο δίκτυο γιατί μπορεί να στείλει στον ίδιο κόμβο το ίδιο μήνυμα πολλές φορές.

4. ΣΗΜΑΣΙΟΛΟΓΙΚΑ ΙΔΕΑΤΑ ΔΙΚΤΥΑ

Στα σημασιολογικά ιδεατά δίκτυα οι κόμβοι που συμμετέχουν έχουν ένα είδος οργάνωσης. Οργανώνονται σε ομάδες κόμβων με κοινά ενδιαφέροντα και περιεχόμενο ώστε να είναι αποτελεσματικότερη η αναζήτηση και κυρίως η αναζήτηση μη-δημοφιλών αντικειμένων και η απάντηση “partial-match” ερωτήσεων (ερωτήσεις με τυπογραφικά λάθη ή ερωτήσεις που περιέχουν ένα υποσύνολο από λέξεις κλειδιά). Στο συγκεκριμένο κεφάλαιο θα περιγραφούν τέσσερις διαφορετικές προσεγγίσεις τέτοιων δικτύων.

1η Προσέγγιση. Associative Search in Peer- to-Peer Networks

Περιγραφή

Το κύριο συστατικό στην προσέγγιση αυτή [6] είναι η καθοδηγούμενη αναζήτηση (guided search), κάτι μεταξύ τυφλής αναζήτησης και αναζήτησης με DHT. Σε αυτήν την αναζήτηση, η ερώτηση δρομολογείται σε μία ομάδα κόμβων με σημασιολογικά όμοιο περιεχόμενο και στη συνέχεια μέσα σε αυτή την ομάδα γίνεται τυφλή αναζήτηση. Αυτή η ομάδα κόμβων ονομάζεται “guide rule” και όλοι οι κόμβοι που ανήκουν σε αυτήν ικανοποιούν κάποιο κατηγόρημα. Μία απλή περίπτωση κατηγορήματος είναι η παρουσία ενός συγκεκριμένου αντικειμένου, οπότε έχουμε “possession rules”. Διαισθητικά, οι κόμβοι που διαμοιράζονται σπάνια αντικείμενα είναι πιο πιθανό να ικανοποιούν ο ένας τις αιτήσεις του άλλου και αυτό ακριβώς είναι που εκμεταλλεύεται η χρήση “possession rules”. Επίσης με την χρήση των “possession rules” είναι εύκολο το παράλληλο “downloading” αντικειμένων από όλους τους κόμβους που συμμετέχουν στο “possession rule”, αφού πολύ εύκολα μπορούν να βρεθούν όλοι οι κόμβοι που περιέχουν ένα συγκεκριμένο αντικείμενο (βρίσκονται στο ίδιο “possession rule”).

Στρατηγικές Αναζήτησης

Ο αλγόριθμος RAPIER

Ο αλγόριθμος χρησιμοποιεί “possession rules” και εργάζεται ως εξής: επιλέγει τυχαία μία περιοχή ενδιαφέροντος που ψάχνει και εκτελεί τυφλή αναζήτηση μέσα στο “possession rule” αυτού του ενδιαφέροντος. Αυτό επαναλαμβάνεται έως ότου βρεθεί ο επιθυμητός αριθμός αποτελεσμάτων. Ο αλγόριθμος συγκρίνεται με τον “URAND” και “PRAND” οι οποίοι δεν χρησιμοποιούν κανενός είδους ομαδοποίηση των κόμβων. Ο πρώτος επιλέγει τυχαία με την ίδια πιθανότητα έναν κόμβο για να δρομολογήσει την ερώτηση, ενώ ο δεύτερος δρομολογεί την ερώτηση σε κόμβους με βάση μία πιθανότητα η οποία είναι ανάλογη του αριθμού των αντικειμένων που κατέχει. Η σύγκριση γίνεται με βάση το μέσο μήκος της αναζήτησης και το ποσοστό επιτυχίας και αποδεικνύεται ότι ο RAPIER είναι καλύτερος από τους άλλους, κυρίως όταν οι κόμβοι ανήκουν σε μικρό αριθμό από “guide rules”

Ο αλγόριθμος GAS

Ο “GAS” είναι μία βελτιστοποίηση του “RAPIER” και διαφέρει μόνο σε ένα σημείο. Αντί να επιλέγει τυχαία “possession rules” όπως ο “RAPIER” επιλέγει με βάση μία διαβάθμιση που κάνει για το ποιος “possession rule” είναι πιο κατάλληλος. Την διαβάθμιση την κάνει με βάση στατιστικά που κρατάει από προηγούμενες ερωτήσεις. Όπως είναι φυσικό ο “GAS” έχει

μικρότερο μέσο μήκος αναζήτησης, δηλαδή αριθμό κόμβων που επισκέπτεται, αλλά έχει το επιπλέον κόστος της ενημέρωσης των στατιστικών.

2η Προσέγγιση. Efficient Content Location Using Interest-Based Locality in Peer-to-Peer Systems

Περιγραφή

Σε αυτό το άρθρο [7] προτείνεται ένας τρόπος για την βελτίωση του μη-δομημένου συστήματος "Gnutella" ώστε να αντιμετωπιστεί η αδυναμία του στην δυνατότητα κλιμάκωσης. Αυτό που προτείνεται είναι η οργάνωση των κόμβων σε μια δομή πάνω από το δίκτυο του "Gnutella" η οποία βασίζεται στα ενδιαφέροντα των κόμβων. Η ιδέα στην δημιουργία μιας τέτοιας δομής βασίζεται στην αρχή ότι αν ένας κόμβος έχει ένα συγκεκριμένο αντικείμενο που ενδιαφέρει κάποιον άλλο κόμβο, είναι πολύ πιθανό ότι θα έχει και άλλα αντικείμενα που θα τον ενδιαφέρουν. Ο τρόπος που δημιουργούνται οι ομάδες κοινού ενδιαφέροντος είναι με την δημιουργία επιπλέον συνδέσεων (shortcuts), οπότε προκύπτει ένα άλλο ιδεατό δίκτυο πάνω από το "Gnutella". Κάθε κόμβος συντηρεί επομένως και μία λίστα συνδέσεων. Η δομή αυτή των συντομεύσεων μπορεί να εφαρμοστεί και σε δομημένα P2P συστήματα.

Δημιουργία και χρήση των συντομεύσεων

Κατά την εισαγωγή του ένας κόμβος εκτελεί μία πλημμύρα για να βρει άλλους κόμβους με το ίδιο περιεχόμενο. Ένας ή περισσότεροι κόμβοι που έχουν το ίδιο περιεχόμενο μπορούν να μπουν στην λίστα των συνδέσεων του. Από τα πειράματα έχει δειχθεί ότι με την προσθήκη πέντε συντομεύσεων σε κάθε αναζήτηση έχουμε την καλύτερη απόδοση. Στη συνέχεια σε επόμενες αναζητήσεις, οι κόμβοι που απαντάνε μπαίνουν επίσης στη λίστα των συντομεύσεων. Αν αυτή η λίστα είναι γεμάτη μία καινούρια συντόμευση πρέπει να βγάλει από τη λίστα μία άλλη, την λιγότερο χρήσιμη με βάση κάποια διαβάθμιση. Διαβάθμιση των συντομεύσεων μπορεί να γίνει με βάση την πιθανότητα επιτυχίας του κόμβου, το "latency" ή το "bandwidth" του μονοπατιού, το φόρτο της συντόμευσης, το ποσό του περιεχομένου ή ενός συνδυασμού όλων αυτών. Στη λίστα των συντομεύσεων ενός κόμβου μπορούν να μπουν επίσης και οι συντομεύσεις των γειτόνων (συντομεύσεων).

Κατά την αναζήτηση ενός αντικειμένου, ένας κόμβος εξετάζει πρώτα στους κόμβους που δείχνουν οι συντομεύσεις του κι αν δεν βρει ικανοποιητικό αριθμό απαντήσεων τότε καταφεύγει στην μέθοδο της πλημμύρας στο "Gnutella" δίκτυο. Τα πειράματα δείχνουν ότι ο αριθμός των μηνυμάτων μιας αναζήτησης μειώνεται σημαντικά με τη χρήση συντομεύσεων σε σχέση με τα μηνύματα που ανταλλάσσονται στο "Gnutella" χωρίς τη χρήση

συντομεύσεων. Επίσης με τη χρήση των συντομεύσεων μπορούν να βρεθούν δημοφιλή και μη-δημοφιλή αντικείμενα πολύ γρήγορα και αποτελεσματικά με συμμετοχή λίγων κόμβων στην διαδικασία της αναζήτησης.

3η Προσέγγιση. Semantic Overlay Networks for P2P Systems

Περιγραφή

Και εδώ [8] η ιδέα είναι να ομαδοποιηθούν οι κόμβοι σε σημασιολογικά ιδεατά δίκτυα (SONs) έτσι ώστε οι ερωτήσεις να προωθούνται μόνο στις σχετικές κατηγορίες με αποτέλεσμα τη μείωση του φόρτου στο φυσικό δίκτυο. Κάθε κόμβος συνδέεται με έναν μικρό αριθμό κόμβων. Μία σύνδεση είναι μία τριάδα (*ni, nj, l*), όπου *ni* και *nj* είναι οι συνδεδεμένοι κόμβοι και *l* είναι το όνομα της κατηγορίας. Το σύνολο των τριάδων με το ίδιο *l* αποτελούν ένα SON.

Δημιουργία και χρήση των SONs

Η υπόθεση που γίνεται είναι ότι υπάρχει μια προκαθορισμένη ιεραρχία κατηγοριών η οποία έχει δομή δένδρου. Επίσης υπάρχουν ταξινομητές που ταξινομούν ερωτήσεις και αντικείμενα στις κατηγορίες της ιεραρχίας. Κατά την εισαγωγή, ο κόμβος πρέπει να εισαχθεί σε ένα ή περισσότερα SONs. Σε ποια από τα SONs θα εισαχθεί εξαρτάται από τα αντικείμενα που έχει. Τα αντικείμενα αυτά ταξινομούνται σε κατηγορίες και ο κόμβος εισάγεται στα SONs των αντίστοιχων κατηγοριών. Η εισαγωγή του κόμβου σε ένα SON μπορεί να γίνει με δύο κριτήρια: είτε αν ο κόμβος έχει απλά ένα αντικείμενο της κατηγορίας του SON, είτε αν ο κόμβος έχει έναν σημαντικό αριθμό από αντικείμενα της κατηγορίας του SON. Πολύ σημαντικό παράγοντα παίζει με πόση ακρίβεια ταξινομούν οι ταξινομητές τα αντικείμενα. Μία σωστή ταξινόμηση αντιστοιχεί τα αντικείμενα σε φύλλα της ιεραρχίας, ενώ μία λάθος τα αντιστοιχεί σε ενδιάμεσους κόμβους.

Μία αναζήτηση ενός αντικειμένου γίνεται ταξινομώντας την ερώτηση σε μια κατηγορία της ιεραρχίας. Η ιδανικότερη περίπτωση είναι να μην υπάρξει λάθος στην ταξινόμηση και η ερώτηση να ταξινομηθεί σε κάποιο φύλλο της ιεραρχίας. Τότε η αναζήτηση γίνεται στο αντίστοιχο SON του συγκεκριμένου φύλλου. Αν ο αριθμός των αποτελεσμάτων δεν είναι ικανοποιητικός, η αναζήτηση συνεχίζεται και στα επόμενα SONs που αντιστοιχούν στους κόμβους της ιεραρχίας από το φύλλο προς τη ρίζα. Στη χειρότερη περίπτωση που η ερώτηση ταξινομηθεί στην ρίζα της ιεραρχίας, διερευνώνται όλα τα SONs. Αν η ερώτηση ταξινομηθεί σε κάποιον ενδιάμεσο κόμβο της ιεραρχίας τότε διερευνούνται όλα τα SONs που είναι απόγονοι και πρόγονοι στην ιεραρχία.

4η Προσέγγιση. Peer-to-Peer Information Retrieval Using Self-Organizing Semantic Overlay Networks

Η συγκεκριμένη προσέγγιση [9] χρησιμοποιεί τις τεχνικές VSM (Vector Space Model) και LSI (Latent Semantic Indexing) ώστε να γίνεται αντιστοίχιση των αντικειμένων και των ερωτήσεων σε σημασιολογικά διανύσματα, δηλαδή σε σημεία ενός καρτεσιανού χώρου. Τέτοια διανύσματα που διαφέρουν λίγο μεταξύ τους αντιστοιχούν σε αντικείμενα (ή ερωτήσεις) που διαφέρουν λίγο μεταξύ τους. Σκοπός είναι τα αντικείμενα να τοποθετηθούν σε ένα δίκτυο CAN και τα σημασιολογικά όμοια αντικείμενα να βρίσκονται κοντά μεταξύ τους. Για να γίνει ως σαν κλειδί για την τοποθέτηση των αντικειμένων στο CAN χρησιμοποιείται το σημασιολογικό διάνυσμα. Μια ερώτηση αντιστοιχίζεται και αυτή σε ένα διάνυσμα οπότε προκύπτει η θέση της στο δίκτυο CAN κι έτσι δρομολογείται με το γνωστό τρόπο σε εκείνο το σημείο. Η αναζήτηση έπειτα γίνεται στον κόμβο που κατέχει το συγκεκριμένο σημείο και σε μια περιοχή γύρω από αυτόν.

Ένα δύσκολο σημείο στην υλοποίηση αυτού του αλγορίθμου είναι η επιλογή της διάστασης του CAN. Το ιδανικότερο θα ήταν η διάσταση του CAN να ήταν ίση με την διάσταση των σημασιολογικών διανυσμάτων η οποία όμως είναι πολύ μεγάλη. Η πολύ μεγάλη διάσταση όμως δεν βολεύει στο CAN γιατί δεν υπάρχουν τόσο πολύ κόμβοι, τα σημασιολογικά διανύσματα δεν κατανέμονται ομοιόμορφα στο χώρο και γιατί η αναζήτηση γίνεται πολύπλοκη. Τα προβλήματα αυτά υπερνικούνται με τη βοήθεια τεχνικών που μειώνουν την διάσταση του προβλήματος και τον χώρο αναζήτησης. Για την μείωση της διάστασης χρησιμοποιείται η τεχνική των «περιστρεφόμενων σημασιολογικών διανυσμάτων» ενώ η μείωση του χώρου αναζήτησης γίνεται με διατήρηση ευρετηρίων με κόμβους που απαντήσανε σε προηγούμενες ερωτήσεις.

5. ΣΥΝΘΕΤΕΣ ΕΡΩΤΗΣΕΙΣ ΣΕ P2P

Στην ενότητα αυτή θα περιγραφούν τρόποι για τη δημιουργία P2P συστημάτων τα οποία είναι σε θέση να απαντάνε σε σύνθετες ερωτήσεις, όπως είναι οι πολυδιάστατες ερωτήσεις και οι ερωτήσεις εύρους. Επίσης θα περιγραφεί ένας τρόπος για κατανεμημένη επεξεργασία ερωτήσεων οι οποίες περιέχουν δομημένη πληροφορία.

5.1. Πολυδιάστατες Ερωτήσεις και Ερωτήσεις Εύρους

Με τον όρο πολυδιάστατες ερωτήσεις εννοούμε ερωτήσεις που αφορούν περισσότερα του ενός γνωρίσματα ενός αντικειμένου και με τον όρο ερωτήσεις εύρους εννοούμε ερωτήσεις

που αφορούν ένα εύρος τιμών κι όχι μια συγκεκριμένη τιμή. Για παράδειγμα, αν υπήρχαν αντικείμενα με δύο γνωρίσματα *x* και *y* τότε μια ερώτηση εύρους δύο διαστάσεων θα ήταν της μορφής: [50≤x≤150, 150≤y≤250], όπου τα ζητούμενα αντικείμενα είναι αυτά που το *x* γνώρισμά τους παίρνει τιμές από 50 έως 150 και το *y* από 150 έως 250. Στη συνέχεια της ενότητας θα περιγραφούν πρωτόκολλα και αλγόριθμοι που υποστηρίζουν τέτοιου είδους ερωτήσεις.

Το Πρωτόκολλο Mercury

Περιγραφή

Το Mercury [16], είναι ένα πρωτόκολλο που υποστηρίζει πολυδιάστατες ερωτήσεις και ερωτήσεις εύρους. Κάθε ερώτηση θεωρείται σαν την σύνδεση πολλών εύρων τιμών ενός ή περισσότερων γνωρισμάτων. Το βασικό συστατικό του πρωτοκόλλου είναι τα *"hubs"*. Ένα "hub" είναι μία συλλογή από κόμβους που συνδέονται μεταξύ τους. Για κάθε γνώρισμα δημιουργείται ένα "hub". Όταν ένα αντικείμενο εισάγεται στο σύστημα τότε αντιγράφεται σε όλα τα "hubs" ή απλά τοποθετείται σε όλα τα "hubs" ένας δείκτης προς το αντικείμενο για εξοικονόμηση χώρου. Μια ερώτηση για να απαντηθεί δρομολογείται ακριβώς σε ένα "hub". Οι κόμβοι σε ένα "hub" οργανώνονται σε ένα κυκλικό ιδεατό δίκτυο με κάθε κόμβο να αναλαμβάνει ένα υποσύνολο του εύρους τιμών του συγκεκριμένου γνωρίσματος. Έτσι τα αντικείμενα τοποθετούνται συνεχόμενα σε αυτό το κύκλο ανάλογα με την τιμή τους. Ένας κόμβος μπορεί να συμμετέχει σε περισσότερα του ενός "hubs" ανάλογα με τα αντικείμενα που κατέχει.

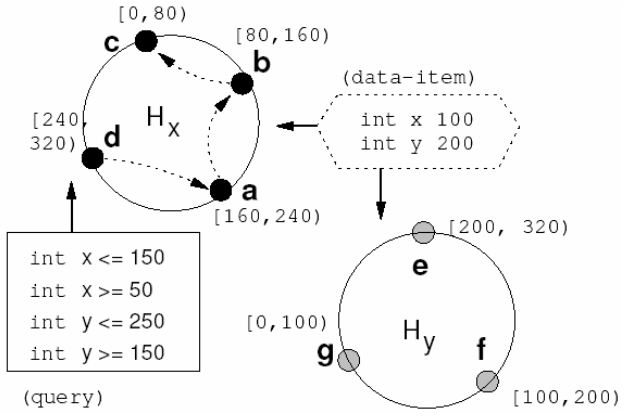
Κάθε αντικείμενο αναπαριστάται ως μια λίστα από εγγραφές, όπου κάθε εγγραφή είναι της μορφής [τύπος – γνώρισμα – τιμή]. Η λίστα επομένως έχει τόσες εγγραφές όσες και τα γνωρίσματα του αντικειμένου. Μία ερώτηση αναπαριστάται ως μία συνένωση κατηγορημάτων τα οποία είναι εγγραφές της μορφής [τύπος – γνώρισμα – τελεστής – τιμή]

Κατασκευή – Αναζήτηση

Ένα αντικείμενο καθώς εισάγεται στο σύστημα πρέπει να αντιγραφεί σε όλα τα "hubs". Για παράδειγμα, στο Σχήμα 5, το αντικείμενο με *x*=100 και *y*=200 εισάγεται και στο "hub" του *x* (H_x) αλλά και στο "hub" του *y* (H_y). Πιο συγκεκριμένα, εισάγεται στους κόμβους *b* και *e* γιατί το εύρος που αυτοί καλύπτουν περιέχουν τις τιμές των *x* και *y* του αντικειμένου αντίστοιχα. Οι τέσσερις κόμβοι του H_x μοιράζονται ομοιόμορφα το εύρος τιμών [0, 320) για το *x*, ενώ οι τρεις κόμβοι του H_y μοιράζονται ομοιόμορφα το εύρος τιμών [0, 320) για το *y*.

Μία ερώτηση δρομολογείται ακριβώς σε ένα "hub" το οποίο αντιστοιχεί σε ένα από τα γνωρίσματα της ερώτησης και στη συνέχεια προωθείται σε εκείνους τους κόμβους του "hub"

στους οποίους μπορούν να βρεθούν κάποια αποτελέσματα. Ο τρόπος λειτουργίας της αναζήτησης φαίνεται στο Σχήμα 5. Η ερώτηση περιέχει τέσσερις εγγραφές-κατηγορήματα οι οποίες συνοψίζονται στην εξής έκφραση: [50≤x≤150, 150≤y≤250]. Η ερώτηση δρομολογείται στον H_x αρχικά και συγκεκριμένα στον κόμβο *d* και από εκεί στη συνέχεια στους κόμβους *b* και *c*, οι οποίοι είναι υπεύθυνοι για το εύρος τιμών [0, 160) που είναι υπερσύνολο του εύρους τιμών [50, 150] του γνωρίσματος *x* της ερώτησης.



Σχήμα 5: Παράδειγμα ερώτησης εύρους με δύο διαστάσεις X και Y

Η δρομολόγηση μέσα σε ένα "hub" μπορεί να γίνει ακολουθιακά, προωθώντας την ερώτηση από κόμβο σε κόμβο όπως βρίσκονται στον κύκλο. Αυτό απαιτεί για κάθε κόμβο, μόνο δείκτες στον προηγούμενο και στον επόμενο κόμβο του στον κύκλο. Για αποτελεσματικότερη αναζήτηση μπορούν να χρησιμοποιηθούν *k* επιπλέον δείκτες προς άλλους κόμβους. Τέλος, εκτός από αυτούς τους δείκτες ένας κόμβος πρέπει να διατηρεί και δείκτες προς όλα τα "hubs" που υπάρχουν.

Στην αποτελεσματικότητα της αναζήτησης σημαντικό ρόλο παίζει η κατανομή που θα έχουν τα αντικείμενα στους κόμβους των "hubs". Έτσι με μια ομοιόμορφη κατανομή μπορούμε να λάβουμε πολυπλοκότητα αναζήτησης ίση με $O((1/k)\times log^2 N)$ (χρησιμοποιώντας k δείκτες σε κάθε κόμβο), η οποία δίνει την δυνατότητα κλιμάκωσης. Όμως, για να επιτευχθεί ομοιόμορφη κατανομή των αντικειμένων στους κόμβους είναι πολύ δύσκολο και απαιτείται εξισορρόπηση του φορτίου. Λόγω της συμμετοχής των αντικειμένων σε όλα τα "hubs" και την ύπαρξη πολλών δεικτών, οι εισαγωγές και οι διαγραφές κόμβων από το σύστημα μπορούν να γίνουν με καλή πολυπλοκότητα μόνο όταν ο αριθμός των γνωρισμάτων είναι σχετικά μικρός.

Πολυδιάστατες Ερωτήσεις Εύρους με τη χρήση του CAN

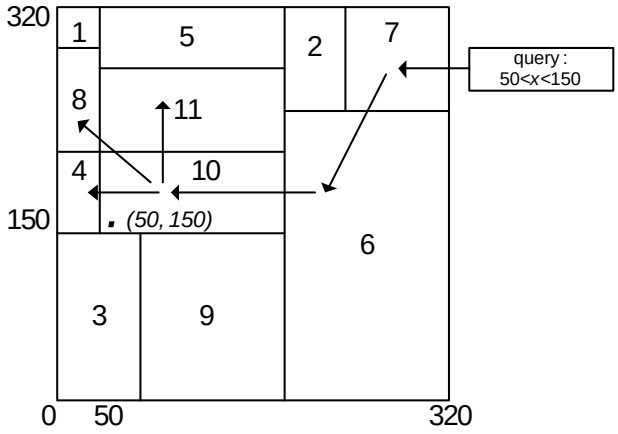
Περιγραφή - Κατασκευή

Στην συγκεκριμένη προσέγγιση [15], για κάθε γνώρισμα κατασκευάζεται ένα σύστημα CAN δύο διαστάσεων. Και στις δύο διαστάσεις αναπαριστάται το εύρος τιμών του συγκεκριμένου γνωρίσματος. Η μία διάσταση αφορά το κάτω όριο μιας ερώτησης εύρους, ενώ η άλλη το πάνω όριο και μία ερώτηση αντιστοιχίζεται σε ένα σημείο του CAN (*target point*). Δηλαδή, η ερώτηση για το γνώρισμα x: $[50 \leq x \leq 150]$, θα αντιστοιχιστεί στο "target point" (50, 150). Η ζώνη που περιέχει το "target point" ονομάζεται "target zone" και ο κόμβος που είναι υπεύθυνος για τη ζώνη αυτή "target node".

Η ιδέα είναι η εξής: θεωρώντας ότι όλα τα αντικείμενα βρίσκονται αποθηκευμένα σε μία βάση δεδομένων με πολλές σχέσεις, να μπορούν αυτά να κατανεμηθούν στο P2P δίκτυο. Κάθε κόμβος καταλαμβάνει συγκεκριμένο υποσύνολο του εύρους τιμών για κάθε γνώρισμα και η τοποθέτησή στο δίκτυο γίνεται με βάση αυτό. Έτσι αρχικά ολόκληρη η βάση δεδομένων βρίσκεται σε έναν αρχικό κόμβο, ο οποίος καταλαμβάνει ολόκληρο τον εικονικό χώρο του CAN και στη συνέχεια κατανέμεται στους υπόλοιπους κόμβους του συστήματος με διάσπαση του χώρου σε μικρότερες ζώνες. Οι κόμβοι που κατέχουν κάποια ζώνη του CAN και επομένως ένα μέρος της βάσης ονομάζονται "active" κόμβοι. Όλοι οι υπόλοιποι ονομάζονται "passive". Όταν κάποιος "active" κόμβος δέχεται πολλές ερωτήσεις ή όταν περνάνε από αυτόν πολλές ερωτήσεις δρομολόγησης, διασπά τη ζώνη του σε δύο ίσα μέρη και το ένα κομμάτι το αναλαμβάνει κάποιος "passive" κόμβος ο οποίος γίνεται πλέον "active".

Αναζήτηση

Οι ερωτήσεις δρομολογούνται και απαντιόνται αποκλειστικά από "active" κόμβους. Η ερώτηση δρομολογείται στον "target node" όπως ακριβώς στο σύστημα CAN (Ενότητα 2.1). Στο Σχήμα 6 φαίνεται η δρομολόγηση της ερώτησης $[50 \leq x \leq 150]$ που αρχικοποιείται στον κόμβο 7. Το "target point" της ερώτησης είναι το (50, 150) και ο "target node" είναι ο 10. Η ερώτηση δρομολογείται στον 10 μέσω του 6. Αν τα αποτελέσματα που περιέχει ο 10 δεν είναι αρκετά, τότε η αναζήτηση μπορεί να προωθηθεί στους κόμβους που βρίσκονται πάνω και αριστερά του 10, δηλαδή, στους 4, 8 και 11 και στη συνέχεια στους 1 και 5. Αυτό συμβαίνει γιατί όλοι οι κόμβοι που βρίσκονται πάνω και αριστερά του "target point" καταλαμβάνουν υπερσύνολα του εύρους τιμών που καταλαμβάνει ο "target node".



Σχήμα 6: Δρομολόγηση ερώτησης εύρους για το γνώρισμα *x*

Η προώθηση των μηνυμάτων προς τους κόμβους που βρίσκονται πάνω και αριστερά από το “target point” φορτώνει τους κόμβους που βρίσκονται προς την πάνω αριστερή γωνία του CAN με πολλά μηνύματα. Αυτό μπορεί να αντιμετωπιστεί με δύο τρόπους: α) η προώθηση να μην γίνεται σε όλους τους κόμβους μέχρι την πάνω αριστερή γωνία αλλά μόνο σε μια περιοχή γύρω από το “target point” και β) να γίνεται “caching” των αποτελεσμάτων για την εκμετάλλευση παλιότερων αποτελεσμάτων με μεγαλύτερο εύρος τιμών. Για παράδειγμα, αν αποθηκευτεί το αποτέλεσμα της ερώτησης [50≤*x*≤150] σε έναν κόμβο, τότε ο κόμβος θα είναι σε θέση να απαντήσει και την ερώτηση [70≤*x*≤100].

Όλα τα παραπάνω ισχύουν και για πολυδιάστατες ερωτήσεις. Αν ο αριθμός των γνωρισμάτων των αντικειμένων που αποθηκεύονται στο σύστημα είναι *n*, τότε δημιουργείται ένα πολυδιάστατο CAN με *2n* διαστάσεις. Οι πολυδιάστατες ερωτήσεις αντιστοιχίζονται σε ένα “target point” αυτού του χώρου *2n* διαστάσεων και η αναζήτηση γίνεται με τον ίδιο τρόπο.

SCRAP: Space-Filling Curves with Range Partitioning

Στην προσέγγιση αυτή [14], τα πολυδιάστατα αντικείμενα κατανέμονται στους κόμβους ενός “skip” γράφου. Στον γράφο αυτό οι κόμβοι οργανώνονται πάνω σε ένα κύκλο με βάση τα εύροι τιμών που καταλαμβάνουν. Ένας κόμβος εκτός από τους δείκτες που κρατάει για τον προηγούμενο και τον επόμενο στον κύκλο, μπορεί να κρατάει και άλλους *O(logN)* δείκτες (*N* είναι ο συνολικός αριθμός των κόμβων), σε κόμβους που βρίσκονται σε αποστάσεις που αυξάνονται εκθετικά. Δηλαδή, ο *i* δείκτης δείχνει στον κόμβο που βρίσκεται σε απόσταση 2^i. Η κατανομή των δεδομένων γίνεται σε δύο βήματα: α) αρχικά τα πολυδιάστατα αντικείμενα απεικονίζονται σε αντικείμενα μίας διάστασης, χρησιμοποιώντας “space-filling curve”

μεθόδους, όπως η “z-ordering”[18] και η “Hilbert curve”[19] και β) τα μονοδιάστατα πλέον αντικείμενα κατανέμονται στους κόμβους του “skip” γράφου σύμφωνα με την τιμή τους.

Για την απάντηση μιας πολυδιάστατης ερώτησης εύρους, αρχικά η ερώτηση μετατρέπεται σε ένα σύνολο από απλές μονοδιάστατες ερωτήσεις εύρους που αφορούν ένα μόνο γνώρισμα με “space-filling curve” μεθόδους και στη συνέχεια κάθε μία από αυτές δρομολογείται σε εκείνους τους κόμβους τους οποίους τα εύροι τιμών τους επικαλύπτονται με τα εύροι τιμών των ερωτήσεων. Η αναζήτηση μίας μονοδιάστατης ερώτησης γίνεται σε χρόνο *O(logN)*, ο συνολικός όμως χρόνος αναζήτησης γίνεται απαγορευτικός για αντικείμενα με πολύ μεγάλη διάσταση, γιατί οι σχετικοί κόμβοι που θα πρέπει να εξεταστούν αυξάνουν με την αύξηση της διάστασης.

MURK: Multi-Dimensional Rectangulation with KD-Trees

Στην συγκεκριμένη προσέγγιση, χώρος διαστάσεων των αντικειμένων διασπάται σε υποχώρους ή αλλιώς υπερκύβους. Για αντικείμενα δύο διαστάσεων (δύο γνωρισμάτων) ο χώρος διασπάται σε ορθογώνια. Κάθε κόμβος διαχειρίζεται ένα τέτοιο ορθογώνιο. Για την πραγματοποίηση ενός τέτοιου διαχωρισμού του χώρου των διαστάσεων χρησιμοποιούνται kd-δένδρα των οποίων τα φύλλα αντιστοιχούνε σε ορθογώνια.

Μιλώντας για αντικείμενα δύο διαστάσεων, η κατασκευή ενός MURK δικτύου γίνεται ως εξής: αρχικά ένας κόμβος διαχειρίζεται ολόκληρο το δισδιάστατο χώρο. Όταν ένας δεύτερος κόμβος εισέρχεται στο σύστημα, ο χώρος κόβεται με τέτοιο τρόπο ώστε τα δύο ορθογώνια που προκύπτουν να έχουν τον ίδιο φόρτο, δηλαδή τον ίδιο αριθμό αντικειμένων. Οι δύο κόμβοι αναλαμβάνουν από ένα ορθογώνιο. Με την άφιξη νέων κόμβων στο σύστημα, ακολουθείται η ίδια διαδικασία, μοιράζοντας κάθε φορά ένα ορθογώνιο σε δύο κόμβους. Κατά την αποχώρηση ενός κόμβου από το δίκτυο το ορθογώνιο που καταλάμβανε ο κόμβος πρέπει να ανακατανεμηθεί στους γειτονικούς κόμβους.

Για την απάντηση μιας πολυδιάστατης ερώτησης εύρους, απαιτείται μία σύνδεση μεταξύ των κόμβων έτσι ώστε η ερώτηση να σταλεί σε όλους τους κόμβους που μπορεί να έχουν αποτελέσματα. Ένας τρόπος για να γίνει αυτό είναι με τη σύνδεση όλων των γειτονικών κόμβων, αυτών δηλαδή που έχουν κοινά όρια στο χώρο, οπότε προκύπτει ένα «πλέγμα». Η δρομολόγηση μέσα στο πλέγμα γίνεται με “greedy” τρόπο επιλέγοντας εκείνο το γείτονα που μας οδηγεί πιο κοντά στο ζητούμενο ορθογώνιο που βρίσκονται οι απαντήσεις. Για την βελτίωση της απόδοσης της αναζήτησης μπορούμε να προσθέσουμε επιπλέον δείκτες δρομολόγησης στο δίκτυο.

5.2. Κατανεμημένη Επεξεργασία Ερωτήσεων

Περιγραφή

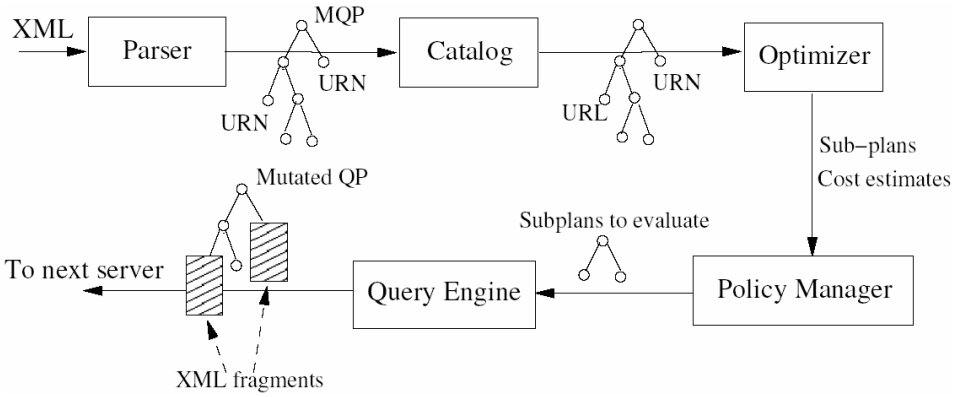
Η ιδέα της συγκεκριμένης προσέγγισης [17], βασίζεται στην κατασκευή πλάνων εκτέλεσης των ερωτήσεων και στη συνέχεια αποτίμησής τους με κατανεμημένο τρόπο. Τα αποτελέσματα μιας ερώτησης συλλέγονται από κόμβο σε κόμβο καθώς το πλάνο της ερώτησης δρομολογείται και αποτιμάται από κόμβο σε κόμβο. Η δρομολόγηση των πλάνων γίνεται με βάση τον αριθμό των αποτελεσμάτων που μπορούν να βρεθούν στους κόμβους και με βάση το κόστος για την ανάκτηση των αποτελεσμάτων. Η ερωτήσεις που μπορούν να απαντηθούν είναι σύνθετες με την έννοια ότι περιέχουν επιπλέον πληροφορίες που εκφράζονται με XML. Επίσης, οι κόμβοι εκφράζουν τα αντικείμενα που προσφέρουν σε XML με βάση ιεραρχιών κατηγοριών και οι ερωτήσεις γίνονται κι αυτές με βάση αυτές τις ιεραρχίες

Mutant Query Plans (MQP)

Όπως αναφέρθηκε, οι ερωτήσεις εκφράζονται με τη χρήση της XML για να μπορούν να έχουν περισσότερη πληροφορία. Ένας τρόπος για να γίνει αυτό είναι τα MQP πλάνα. Ένα MQP πλάνο είναι ένα δένδρο με τελεστές της σχεσιακής άλγεβρας (όπως *join*, *select* και *union*) και τον ψευδοτελεστή *display* ο οποίος προσδιορίζει τον στόχο της ερώτησης. Ένα MQP πλάνο περιέχει α) κομμάτια XML, β) αναφορές σε τοποθεσίες πόρων (URLs) και γ) αναφορές σε αφηρημένα ονόματα πόρων (URNs).

Όταν γίνεται μια ερώτηση σε κάποιον κόμβο δημιουργείται ένα MQP πλάνο που εκφράζει την ερώτηση. Το πλάνο περνάει από κόμβο σε κόμβο μαζεύοντας αποτελέσματα έως ότου αποτιμηθεί σε ένα σταθερό κομμάτι XML. Κάθε κόμβος που δέχεται το MQP πλάνο μπορεί να το επεξεργαστεί με δύο τρόπους: α) να επιλύσει ένα URN σε ένα ή περισσότερα URLs, β) να μειώσει το πλάνο αποτιμώντας ένα υπό-δένδρο του και αντικαθιστώντας τον με τα αποτελέσματα της αποτίμησης.

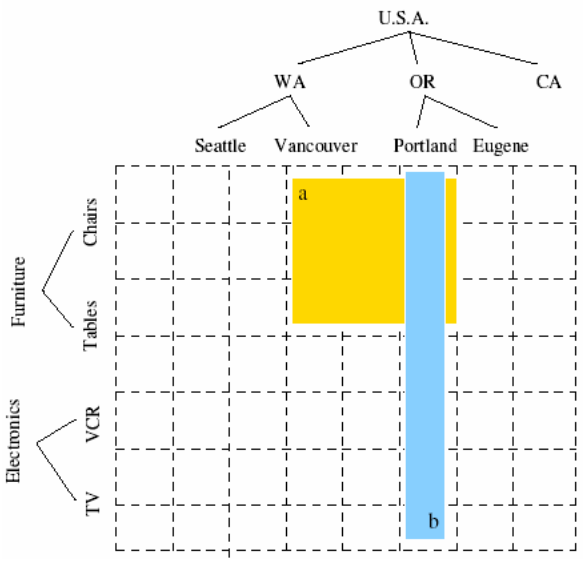
Η επεξεργασία ενός MQP πλάνου φαίνεται στο Σχήμα 7. Αρχικά το πλάνο έρχεται στον κόμβο σε XML μορφή και φτιάχνεται στην μνήμη το αντίστοιχο πλάνο με τη βοήθεια του “Parser”. Στη συνέχεια προσδιορίζεται ποια URNs μπορούν να επιλυθούν με URLs με την χρήση καταλόγων (θα αναλυθεί ο τρόπος στη συνέχεια). Στη συνέχεια ο “Optimizer” βρίσκει ποια υπό-δένδρα μπορούν να αποτιμηθούν τοπικά και με ποιο κόστος. Ο “Policy Manager” αποφασίζει ποια από αυτά τελικά θα αποτιμηθούν τοπικά, στη συνέχεια αποτιμώνται, οπότε προκύπτει το μεταλλαγμένο MQP πλάνο που στέλνεται στον επόμενο κόμβο για να το επεξεργαστεί.



Σχήμα 7: Επεξεργασία ενός MQP πλάνου σε έναν κόμβο

Κατανεμημένοι Κατάλογοι και Ιεραρχίες Κατηγοριών

Στην προηγούμενη ενότητα περιγράφτηκε πως ένας κόμβος επεξεργάζεται ένα MQP πλάνο. Για την αποτίμηση μιας ερώτησης το πλάνο πρέπει να δρομολογηθεί σε πολλούς κόμβους. Η δρομολόγηση πρέπει να γίνεται σε κόμβους που έχουν σχετική πληροφορία με την ερώτηση. Για να γίνει αυτό χρειάζεται ένας κατάλογος που να διατηρεί πληροφορίες για τα αντικείμενα που υπάρχουν στους κόμβους ώστε να επιλύονται τα URNs σε URLs. Για λόγους κλιμάκωσης ο κατάλογος αυτός κατανέμεται μεταξύ των κόμβων. Η κατανομή αυτή γίνεται χρησιμοποιώντας ιεραρχίες κατηγοριών. Μία ιεραρχία κατηγοριών είναι ένα δένδρο με τη ρίζα να είναι η πιο γενική κατηγορία, ενώ οι υπόλοιποι κόμβοι του δένδρου είναι οι διάφορες υποκατηγορίες οργανωμένες στην ιεραρχία του δένδρου με τις πιο ειδικές να βρίσκονται στα φύλλα. Το σύνολο των ιεραρχιών που καλύπτουν ένα πεδίο μιας εφαρμογής αποτελούν ένα “multi-hierarchic namespace”. Έτσι τα αντικείμενα και οι ερωτήσεις εκφράζονται σαν υποσύνολα τέτοιων “namespaces” που αντιστοιχούν σε περιοχές ενδιαφέροντος. Ένα παράδειγμα ενός “namespace” με δύο ιεραρχίες φαίνεται στο Σχήμα 8. Το συγκεκριμένο “namespace” αφορά την πώληση μεταχειρισμένων αντικειμένων στις Η.Π.Α. Η ιεραρχία στο πάνω μέρος του σχήματος κατηγοριοποιεί τα αντικείμενα σε Χώρα, Πολιτεία και Πόλη, ενώ η ιεραρχία στα αριστερά του σχήματος κατηγοριοποιεί ηλεκτρικές συσκευές και έπιπλα. Τα υποσύνολα *a* και *b* αποτελούν δύο διαφορετικές περιοχές ενδιαφέροντος. Για παράδειγμα η περιοχή *a* εκφράζει όλα τα έπιπλα που πουλιόνται στο Vancouver και στο Portland, ενώ η περιοχή *b* εκφράζει όλα τα αντικείμενα που πουλιόνται στο Portland.



Σχήμα 8: Ένα namespace με δύο ιεραρχίες

Ρόλοι των κόμβων

Οι κόμβοι του P2P συστήματος μπορούν να έχουν πολλούς διαφορετικούς ρόλους. Μπορούν να δρουν ως πελάτες που αναζητούν αντικείμενα, ως εξυπηρετητές που παρέχουν κάποια αντικείμενα, ως κόμβοι που φιλοξενούν μέρος του καταλόγου ή ως κόμβοι που διατηρούν ευρετήρια που βοηθούν στην εύρεση κόμβων συγκεκριμένης κατηγορίας.

6. *ΣΥΜΠΕΡΑΣΜΑΤΑ – ΕΠΙΛΟΓΟΣ*

Στην επισκόπηση περιγράφτηκαν συνοπτικά τα P2P συστήματα και ο τρόπος με τον οποίο δουλεύουν. Επίσης αναφέρθηκαν οι τρόποι δημιουργίας και ενημέρωσης αντιγράφων. Τα κεντρικοποιημένα P2P συστήματα ενώ είναι πολύ εύκολα στην υλοποίηση δεν έχουν την ιδιότητα της κλιμάκωσης και είναι εύκολο να μηνυθούν (Napster). Τα μη κεντρικοποιημένα P2P συστήματα αντιθέτως έχουν την ιδιότητα της κλιμάκωσης. Από αυτά, τα δομημένα έχουν πολύ καλούς χρόνους αναζήτησης με μεγαλύτερους χρόνους για εισαγωγή κόμβων ενώ τα μη δομημένα έχουν καλύτερους χρόνους εισαγωγής ενός κόμβου στο δίκτυο αλλά χειρότερους χρόνους αναζήτησης. Επίσης στα δομημένα όλα τα αντικείμενα, δημοφιλή και μη μπορούν να βρεθούν σε συγκεκριμένο χρόνο σε αντίθεση με τα μη δομημένα όπου όμως υποστηρίζονται και “partial-match” ερωτήσεις κάτι που δεν γίνεται στα δομημένα. Τα σημασιολογικά δίκτυα προσπαθούν να δώσουν μία δομή στο δίκτυο με βάση τη σημασιολογία των αντικειμένων ώστε η αναζήτηση να είναι αποτελεσματικότερη ακόμα και για μη δημοφιλή αντικείμενα. Στον Πίνακα 1 βρίσκονται συγκεντρωμένα τα P2P συστήματα που εξετάστηκαν με τα χαρακτηριστικά τους. Στην Ενότητα 5 περιγράφτηκαν αλγόριθμοι και

πρωτόκολλα για απάντηση πολυδιάστατων ερωτήσεων εύρους. Ο σημαντικότερος παράγοντας για την αποτελεσματικότητα αυτών των αλγορίθμων είναι η διάσταση των αντικειμένων. Αυτοί οι αλγόριθμοι βρίσκονται συγκεντρωμένοι στον Πίνακα 2. Ο Πίνακας 3 περιγράφει μερικά ανοιχτά προβλήματα στα P2P συστήματα και τις συνέπειές τους.

Για την κατασκευή ενός P2P συστήματος πρέπει να λαμβάνονται υπόψη όλοι αυτοί οι παράμετροι που αναφέρθηκαν και να γίνεται επιλογή αυτού του συστήματος που ταιριάζει καλύτερα στις απαιτήσεις μας.

Κατηγορίες P2P / Χαρακτηριστικά	**Δομημένα**	**Μη Δομημένα**	**Κεντρικοποιημένα**	**Σημασιολογικά**
Ιδιότητα Κλιμάκωσης	ΝΑΙ	ΟΧΙ	ΟΧΙ	Εξαρτάται από κατηγοριοποίηση
Αυτονομία Κόμβων	ΟΧΙ	ΝΑΙ	ΝΑΙ	ΝΑΙ
Τοπολογία του Δικτύου	Συγκεκριμένη	Ελεύθερη	Κεντρικός Server	Σημασιολογική Ομαδοποίηση
Αριθμός "hops για επιτυχή αναζήτηση	CAN: $d \times N^{(1/d)}$ Chord: logN	Άγνωστος	1 "hop"	Άγνωστος
Πολυπλοκότητα Εισαγωγής/Διαγραφής Κόμβου	CAN: 2d Chord: $(\log N)^2$	Σταθερός χρόνος	Σταθερός χρόνος	Σταθερός χρόνος
Εύρεση δημοφιλών και μη δημοφιλών αντικειμένων	Στον ίδιο χρόνο	Ευκολότερη εύρεση των δημοφιλών	Στον ίδιο χρόνο	Στον ίδιο χρόνο
Υποστήριξη "Partial match" ερωτήσεων	ΟΧΙ	ΝΑΙ	ΝΑΙ	ΝΑΙ
Ανοχή σε Σφάλματα	ΝΑΙ	ΝΑΙ	ΟΧΙ	ΝΑΙ
Ασφάλεια	ΟΧΙ	ΟΧΙ	ΟΧΙ	ΟΧΙ

Πίνακας 1: Χαρακτηριστικά των διάφορων P2P κατηγοριών

Σημείωση: όπου d είναι ο αριθμός των διαστάσεων στο CAN και N ο αριθμός των κόμβων

Αλγόριθμοι	**Βασική ιδέα**	**Ιδιότητα Κλιμάκωσης**
Mercury	Κατασκευή ενός “hub”για κάθε γνώρισμα	Κακή όταν ο αριθμός των γνωρισμάτων είναι μεγάλος
Αλγόριθμος βασισμένος στο CAN	Χρήση ενός δισδιάστατου CAN για κάθε γνώρισμα	Καλή
SCRAP	Μείωση διαστάσεων των αντικειμένων και τοποθέτησή τους σε “skip” γράφο.	Κακή όταν ο αριθμός των γνωρισμάτων είναι μεγάλος
MURK	Κατανομή του χώρου των αντικειμένων στους κόμβους με χρήση kd-δένδρων. Τοποθέτηση των κόμβων σε πλέγμα με και χρήση επιπλέον δεικτών	Καλή όταν τοποθετηθούν στο πλέγμα επιπλέον δείκτες

Πίνακας 2: Σύνοψη των αλγόριθμων για πολυδιάστατες ερωτήσεις εύρους

Προβλήματα στα P2P συστήματα	**Επιπτώσεις**
Ιδεατά δίκτυα ανεξάρτητα από τα φυσικά δίκτυα	Ένα “hop” στο ιδεατό δίκτυο μπορεί να είναι πολύ μεγάλη απόσταση στο φυσικό δίκτυο
Έλλειψη καθολικής γνώσης	Πρέπει να χειριζόμαστε δρομολογήσεις ερωτήσεων βασιζόμενοι σε πιθανότητες
Συχνή ανανέωση του δικτύου: εισαγωγές, διαγραφές	Δύσκολη διατήρηση μιας συγκεκριμένης τοπολογίας, π.χ. στα δομημένα
Μικρή πιθανότητα ένας κόμβος να είναι συνδεδεμένος στο δίκτυο	Δεν υπάρχει εγγύηση για την αξιοπιστία του συστήματος και ότι ένα αντικείμενο θα βρεθεί
Άγνωστη και μεταβαλλόμενη υπολογιστική ισχύς των κόμβων	Δεν μπορούμε να υποθέσουμε μία σταθερή και ισχυρή υποδομή του συστήματος

Πίνακας 3: Ανοιχτά προβλήματα στα P2P και οι επιπτώσεις τους

Αναφορές

[1] S.Ratnasamy, P.Francis, M.Handley, R.Karp and S.Shenker. *A Scalable Content-Addressable Network.* In ACM SIGCOMM'01, August 2001.

[2] I.Stoica, R.Morris, D.Karger, F.Kaashoek, and H.Balakrishnan. *Chord: A scalable peer-to-peer lookup service for internet applications.* In Proceedings of SIGCOMM'2001, 2001

[3] Q.Lv, P.Cao, E.Cohen, K.Li, S.Shenker. *Search and Replication in Unstructured Peer-toPeer Networks.* In ICS'02 June 2002, New York, USA.

[4] D.Tsoumakos, N.Roussopoulos. *A Comparison of Peer-to-Peer Search Methods.* In WebDB, June 2003, San Diego, California.

[5] A.Crespo and H.Garcia-Molina. *Routing Indices for Peer-to-Peer Systems.* In ICDCS, July 2002.

[6] E.Cohen, A.Fiat, H.Kaplan. *Assosiative Search in Peer to Peer Networks: Harnessing Latent Semantics.* IEEE INFOCOM 2003.

[7] K.Sripanidkulchai, B.Maggs, H.Zhang. *Efficient Content Location Using Interest-Based Locality in Peer-to-Peer Systems.* Carnegie Mellon University, Pittsburg, PA 15213.

[8] A.Crespo, H.Molina. *Semantic Overlay Networks for P2P Systems.* In proceedings of the 29th VLDB Conference, Berlin, Germany, 2003.

[9] C.Tang, Z.Xu, S.Dwarkadas. *Peer-to-Peer Information Retrieval Using Self-Organizing Semantic Overlay Networks.* In SIGCOMM'03, August 2003, Karlsruhe, Germany.

[10] *Napster website*. http://www.napster.com

[11] *Gnutella website.* http://www.gnutella.wego.com

[12] *Freenet website.* http://freenet.sourceforge.net

[13] Anwitaman Datta, Manfred Hauswirth, Karl Aberer. *Updates in Highly Unreliable Peer-to-Peer Systems.*

[14] P.Ganesan, B.Yang, H.Garcia-Molina. *One Torus to Rule them. All: Multi-dimensional Queries in P2P Systems*. Seventh International Workshop on the Web and Databases(WebDB 2004), June 17-18, 2004, Paris, France.

[15] O.D.Sahin, A.Gupta, D.Agrawal, A.El Abbadi. *A Peer-to-peer Framework for Caching Range Queries*. Proceedings of the 20th International Conference on Data Engineering (ICDE'04), 2004 IEEE

[16] A.R.Bharambe, M.Agrawal, S.Seshan. *Mercury: Supporting Scalable Multi-Attribute Range Queries*. In SIGCOMM'04, Aug. 30-Sept. 3, 2004,

Portland, Oregon, USA.

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

[18] J.Orenstein and T.Merrett. *A class of data structures for associative searching*. In Proc. PODS, 1984.

[19] H.Jagadish. *Linear clustering of objects with multiple attributes*. In Proceedings. SIGMOD, 1990