Δημοκριτειο Πανεπιστημιο Θρακης
Τμημα Ηλεκτρολογων Μηχανικων και Μηχανικων Υπολογιστων
Τομεας Λογισμικου και Αναπτυξης Εφαρμογων

Κατανεμημένη αναζήτηση από δίκτυα agents με προστασία της ιδιωτικότητας

Διπλωματικη Εργασια

του

Γεωργίου Ε. Σταματελάτου

Επιβλέπων: Παύλος Σ. Εφραιμίδης
Επ. Καθηγητής Δ.Π.Θ.

Εργαστηριο Προγραμματισμου και Επεξεργασιας Πληροφοριων
Ξάνθη, Απρίλιος 2011

Δημοκρίτειο Πανεπιστήμιο Θράκης
Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών
Τομέας Λογισμικού και Ανάπτυξης Εφαρμογών
Εργαστήριο Προγραμματισμού και Επεξεργασίας Πληροφοριών

Κατανεμημένη αναζήτηση από δίκτυα agents με προστασία της ιδιωτικότητας

Διπλωματικη Εργασια

του

Γεωργίου Ε. Σταματελάτου

Επιβλέπων: Παύλος Σ. Εφραιμίδης
Επ. Καθηγητής Δ.Π.Θ.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την ..η Απριλίου 2011.

(Υπογραφή)	*(Υπογραφή)*	*(Υπογραφή)*
.....	.....	.....
Παύλος Σ. Εφραιμίδης	Βασίλειος Θ. Τσαουσίδης	Βασίλειος Α. Κάτος
Επ. Καθηγητής Δ.Π.Θ.	Καθηγητής Δ.Π.Θ	Επ. Καθηγητής Δ.Π.Θ.

Ξάνθη, Απρίλιος 2011

(Υπογραφή)

.....

Γεωργιος Ε. Σταματελατος

Δημοκρίτειο Πανεπιστήμιο Θράκης
Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών
Τομέας Λογισμικού και Ανάπτυξης Εφαρμογών
Εργαστήριο Προγραμματισμού και Επεξεργασίας Πληροφοριών

Στην οικογένειά μου

Ευχαριστίες

Θα ήθελα καταρχήν να ευχαριστήσω τον καθηγητή κ. Παύλο Εφραιμίδη για την επίβλεψη αυτής της διπλωματικής εργασίας καθώς και για την συνεχή υποστήριξη. Επίσης ευχαριστώ ιδιαίτερα τον υποψήφιο διδάκτορα Γιώργο Δροσάτο για την καθοδήγησή του και την εξαιρετική συνεργασία που είχαμε. Τέλος ευχαριστώ τον συνάδελφο Κώστα Καλοπήτα για τις συμβουλές του πάνω σε θέματα σύνταξης του κειμένου.

Περίληψη

Στην παρούσα εργασία παρουσιάζεται το Quantum, ένα δίκτυο ομότιμων κόμβων για την εκτέλεση κατανεμημένων υπολογισμών μεταξύ αυτόνομων agents (πρακτόρων). Βασικό χαρακτηριστικό του Quantum είναι η προστασία της ιδιωτικότητας των κόμβων που συμμετέχουν σε κάθε κατανεμημένο υπολογισμό. Επιπλέον το Quantum υποστηρίζει τοπολογίες που σχηματίζουν μεγάλο πλήθος agents που επικοινωνούν μέσω του διαδικτύου και παρουσιάζει ανοχή σε προσθήκες/αφαιρέσεις κόμβων. Στη λύση που σχεδιάσαμε, επιλέξαμε μια αποκεντρωμένη αρχιτεκτονική για το δίκτυο και αξιοποιήσαμε τεχνολογίες δικτύων ομότιμων κόμβων(peer-to-peer).

Λέξεις Κλειδιά

Δίκτυα Ομότιμων Κόμβων, Κατανεμημένοι Υπολογισμοί, Ιδιωτικότητα

Abstract

In this paper Quantum is being presented. Quantum is a peer-to-peer network used for distributing computations among independent agents. The main feature of Quantum is the privacy of the peers which participate in each distributed computation. Furthermore, Quantum supports infrastructures which are formed by a mass of agents communicating over the Internet, and is tolerant of adding/removing peers. In the proposed solution we have chosen a decentralized network architecture and exploited peer-to-peer network technologies.

Keywords

P2P Networks, Distributed Computations, Privacy

Περιεχόμενα

Κατάλογος Σχημάτων

Κατάλογος Πινάκων

Κεφάλαιο 1

Εισαγωγή

Στόχος της παρούσας διπλωματικής εργασίας είναι να δημιουργηθεί μια δικτυακή πλατφόρμα που να υποστηρίζει κατανεμημένες εφαρμογές μεταξύ ομάδων χρηστών όπου κάθε χρήστης θα αντιπροσωπεύεται από κάποιο προσωπικό agent. Στις εφαρμογές αυτές οι προσωπικοί agent θα πρέπει να μπορούν να εκτελούν κατανεμημένους υπολογισμούς όπου θα χρησιμοποιούνται (χωρίς όμως να αποκαλύπτονται) προσωπικά δεδομένα που φυλάσσονται στους agents. Ένα σύνολο για παράδειγμα εκατοντάδων ή ακόμη και χιλιάδων προσωπικών agent θα μπορούν να εκτελούν έναν κατανεμημένο υπολογισμό ώστε να βρεθεί ποιος χρήστης βρίσκεται πιο κοντά σε κάποια τοποθεσία. Αφορμή για τη μελέτη τέτοιου είδους προβλημάτων αποτέλεσε το έργο Polis, στο οποίο αναπτύσσονται τεχνολογίες όπου κάθε χρήστης διαχειρίζεται ο ίδιος τα προσωπικά του δεδομένα και η πρόσβαση σε αυτά γίνεται αποκεντρωμένα με συμφωνίες μεταξύ των agents. Η ανάγκη για την εκτέλεση σύνθετων υπολογισμών με τη συμμετοχή μεγάλου πλήθους κόμβων του Polis οδήγησε στην ανάπτυξη του Quantum. Βασικό χαρακτηριστικό του Quantum είναι ότι η λειτουργία του δε βασίζεται σε κεντρικό διακομιστή αλλά η επικοινωνία μεταξύ των χρηστών του είναι τελείως αποκεντρωμένη, γεγονός που ενισχύει την ιδιωτικότητά τους.

Για την οργάνωση των κόμβων/agents σε τοπολογία επιλέχθηκε μια αποκεντρωμένη οργάνωση για το δίκτυο όπου όλοι οι κόμβοι συμμετέχουν ισότιμα. Για το λόγο αυτό μελετήθηκαν και αξιοποιήθηκαν τεχνολογίες δικτύων ομότιμων κόμβων (peer-to-peer). Ο λόγος είναι ότι έτσι εξασφαλίζεται η επεκτασιμότητα (scalability) του δικτύου ενώ μειώνονται οι κίνδυνοι για την ιδιωτικότητα των κόμβων.

Επιπρόσθετα, για λόγους κατανόησης διάφορων εννοιών που συναντώνται σε όλη την έκταση της εργασίας, μελετήθηκαν βασικές έννοιες ασφάλειας υπολογιστικών συστημάτων, όπως ιδιωτικότητα, προσωπικά δεδομένα, αλγόριθμοι hash, ασύμμετρη κρυπτογραφία καθώς επίσης και ομομορφικές ιδιότητες κρυπτογράφησης.

Κεφάλαιο 2

Ασφάλεια

2.1 Ιδιωτικότητα

Η λέξη ιδιωτικότητα αποτελεί την απόδοση στην ελληνική, του αγγλικού όρου privacy. Η έννοια της λέξεως με την μετάφραση της, αποδίδεται ικανοποιητικά στην ελληνική, σε αντίθεση με ορισμένες γλώσσες που χρησιμοποιούν τον όρο στα αγγλικά. Αν και η ρίζα της λέξεως στα ελληνικά είναι αρχαία, η έννοια που περιγράφει είναι σχετικά νέα. Πρέπει να τονιστεί πως το "ιδιωτεύειν", εθεωρείτο ως κοινωνική αναπηρία στην αρχαία Ελλάδα, και "ιδιώτης" χαρακτηριζόταν ο μη μετέχων στα κοινά πολίτης. Ωστόσο, ο σημερινός όρος της ιδιωτικότητας, δεν εμπεριέχει αυτή την αρνητική έννοια.

Ένας τομέας της ιστορικής μελέτης (γενικά, αλλά και της ελληνικής εν προκειμένω), αποτελεί ο "Δημόσιος και ιδιωτικός βίος", το δεύτερο σκέλος του οποίου περιγράφει τον ιδιωτικό τομέα της ζωής των μελετώμενων πολιτισμών. Μέσω αυτού μπορεί να καταστεί σαφές, ότι η αρχαία έννοιας της λέξης ήταν διαφορετική με αυτή της σημερινής, καθώς δεν ήταν το δικαίωμα στην προσωπική ζωή που καυτηρίαζε ο όρος, αλλά η συμμετοχή ή όχι (πέραν της δεδομένης οικογενειακής - προσωπικής ζωής) στο δημόσιο βίο και στο πολιτικό γίγνεσθαι.

Η γέννηση της έννοιας της ιδιωτικότητας (privacy) συναντάται για πρώτη φορά στην εργασία των Samuel D. Warren kai Louis D. Brandeis. Στην εργασία αυτή, οι δύο Αμερικανοί δικηγόροι καθόρισαν την ιδιωτικότητα ως "το δικαίωμα να είσαι μόνος" ("the right to be let alone"). Ο λόγος για αυτή τη δημοσίευση ήταν η ανάπτυξη νέων μορφών τεχνολογίας που συνδέθηκαν με ορισμένες εξελίξεις. Οι φωτογραφίες, για παράδειγμα, που χρησιμοποιούνταν από τον Κίτρινο Τύπο ήταν, σύμφωνα με την άποψη των συντακτών, μια επίθεση στην προσωπική ιδιωτικότητα σύμφωνα με την άποψη του δικαιώματος να είσαι μόνος.

Ο πιο συνήθης ορισμός της ιδιωτικότητας που χρησιμοποιείται σήμερα, είναι αυτός του Alan Westin: "Η ιδιωτικότητα είναι η αξίωση των ατόμων, των ομάδων και των ιδρυμάτων, να αποφασίζουν από μόνοι τους για το πότε, πώς και μέχρι ποιο σημείο οι πληροφορίες που τους αφορούν, θα διαβιβάζονται σε άλλους". Ένας άλλος επίσης γνωστός όρος που χρησιμοποιείται για την προστασία των προσωπικών δεδομένων, εκτός του όρου της ιδιωτικότητας, και είναι επίσης σύμφωνος με τον ορισμό του Westin, είναι αυτός του πληροφοριακού αυτοπροσδιορισμού.

2.2 Προσωπικά δεδομένα

Τα προσωπικά δεδομένα είναι κάθε πληροφορία που αναφέρεται και περιγράφει ένα άτομο, όπως: στοιχεία αναγνώρισης (ονοματεπώνυμο, ηλικία, κατοικία, επάγγελμα, οικογενειακή κατάσταση κλπ.), φυσικά χαρακτηριστικά, εκπαίδευση, εργασία (προϋπηρεσία, εργασιακή συμπεριφορά κλπ.), οικονομική κατάσταση (έσοδα, περιουσιακά στοιχεία, οικονομική συμπεριφορά), ενδιαφέροντα, δραστηριότητες και συνήθειες. Το άτομο (φυσικό πρόσωπο) στο οποίο αναφέρονται τα δεδομένα ονομάζεται υποκείμενο των δεδομένων.

Πέραν του γενικού χαρακτηρισμού ορισμένων δεδομένων ως προσωπικά, υπάρχει διάκριση μιας υποκατηγορίας αυτών που περιγράφονται με τον όρο "Ευαίσθητα προσωπικά δεδομένα". Ευαίσθητα χαρακτηρίζονται τα προσωπικά δεδομένα ενός ατόμου που αναφέρονται στη φυλετική ή εθνική του προέλευση, στα πολιτικά του φρονήματα, στις θρησκευτικές ή φιλοσοφικές του πεποιθήσεις, στη συμμετοχή του σε συνδικαλιστική οργάνωση, στην υγεία του, στην κοινωνική του πρόνοια, στην ερωτική του ζωή, στις ποινικές διώξεις και καταδίκες του, καθώς και στη συμμετοχή του σε συναφείς με τα ανωτέρω ενώσεις προσώπων. Τα ευαίσθητα προσωπικά δεδομένα προστατεύονται από το νόμο με αυστηρότερες ρυθμίσεις από ότι τα απλά προσωπικά δεδομένα.

Εκτός από τις δύο προαναφερθείσες κατηγορίες δεδομένων, οι οποίες περιγράφονται από το νομοθέτη, και θεσμικά υπάγονται υπό την εποπτεία της αρμόδιας ανεξάρτητης αρχής (στην Ελλάδα αυτή είναι η "Αρχή προστασίας δεδομένων προσωπικού χαρακτήρα"), υπάρχει και μια διαφορετική κατηγορία δεδομένων, τα λεγόμενα "Προσωπικά αναγνωρίσιμα δεδομένα" (PII = Personal Identifiable Information). Τα δεδομένα αυτά μπορεί να είναι μεμονωμένα, όπως π.χ. ο αριθμός ταυτότητας, ή συνδυασμοί, όπως π.χ. η ημερομηνία γέννησης και ο ταχυδρομικός κώδικας, και μπορούν να προσδιορίσουν με ακρίβεια ένα άτομο.

Πρέπει να τονιστεί ότι τα PII δεν περιορίζονται μόνο στα προσωπικά δεδομένα. Ένα συχνά αναφερόμενο σχετικό παράδειγμα είναι ότι, το 87% του αμερικανικού πληθυσμού μπορεί να ταυτοποιηθεί από το συνδυασμό του φύλου, της ημερομηνίας γέννησης και του ταχυδρομικού κώδικα της κατοικίας του.

Από τα παραπάνω προκύπτει ένα κρίσιμο συμπέρασμα. Η γνώση μιας ομάδας πληροφοριών, όπως για παράδειγμα το ιστορικό των αναζητήσεων ενός άγνωστου χρήστη μιας μηχανής αναζήτησης, μπορεί να μας οδηγήσει στην εξαγωγή σημαντικών στοιχείων, ακόμη και ευαίσθητων προσωπικών δεδομένων του ατόμου. Δεδομένου ότι μπορεί να γίνει ταυτοποίηση του αρχικά άγνωστου χρήστη από μια ομάδα PII που ίσως εισάγει στις αναζητήσεις του, μπορεί ακολούθως, από τη μορφή και το περιεχόμενο των αναζητήσεων του, να εξαχθούν πληροφορίες για τις πράξεις του αλλά ακόμη και για τις σκέψεις του. Από ένα ανώνυμο σύνολο ερωτημάτων δηλαδή, μπορεί σε πρώτο επίπεδο να υπάρξει αναγνώριση του δημιουργού των ερωτημάτων, και ακολούθως, σε δεύτερο επίπεδο εξόρυξη ευαίσθητων δεδομένων που τον αφορούν.

Τέλος, χάρη στις τελευταίες εξελίξεις στην ανάπτυξη φορητών συσκευών και γενικότερα τεχνολογιών Ubiquitous Computing, έχει δοθεί επιπλέον η δυνατότητα συλλογής "Δυναμικών προσωπικών δεδομένων". Σε αυτή την ειδική κατηγορία προσωπικών δεδομένων βρίσκονται δεδομένα όπως η γεωγραφική θέση (Location) και πληροφορίες που αφορούν τη σωματική

κατάσταση και την υγεία ενός ατόμου, όπως η αρτηριακή πίεση, οι χτύποι της καρδίας και η διάθεση του.

2.3 Βασικές έννοιες της κρυπτογραφίας

2.3.1 Χρήση της κρυπτογραφίας στην ανταλλαγή δεδομένων

Η ανάγκη για την αποστολή ενός μηνύματος από κάποιον αποστολέα σε έναν παραλήπτη, χωρίς να υπάρχει κίνδυνος να διαβαστεί από κάποιον άλλον, δηλαδή αυτό το μήνυμα να αποσταλεί με ασφάλεια, αποτελεί ένα βασικό αντικείμενο της κρυπτογραφίας.

2.3.2 Βασική ιδέα της κρυπτογραφίας

Το αρχικό αυτό μήνυμα είναι το plaintext (ή cleartext, δηλ. καθαρό κείμενο). Η διαδικασία με την οποία στο μήνυμα αυτό αποκρύπτεται η πληροφορία του περιεχόμενου του ονομάζεται κρυπτογράφηση (encryption). Το κρυπτογραφημένο μήνυμα είναι το ciphertext. Η διαδικασία αλλαγής του ciphertext πίσω πάλι στο plaintext ονομάζεται αποκρυπτογράφηση (decryption). (Σημείωση: Σύμφωνα με το πρότυπο ISO 7498-2, χρησιμοποιούνται οι όροι "encipher" και "decipher" αντί για τους όρους "encrypt" και "decrypt", αντίστοιχα.) Όλα αυτά παρουσιάζονται στο Σχήμα 2.1.

Σχήμα 2.1: Κρυπτογράφηση και αποκρυπτογράφηση.

Η επιστήμη που έχει ως αντικείμενο την ασφάλεια των μηνυμάτων είναι η κρυπτογραφία (cryptography), και πραγματοποιείται από τους cryptographers (κρυπτογράφους). Οι Cryptanalysts ασχολούνται με την κρυπτανάλυση (cryptanalysis), το σπάσιμο του ciphertext, δηλαδή να μπορέσουν να δουν τι κρύβεται πίσω από τα κρυπτογραφημένα δεδομένα. Ο κλάδος της επιστήμης που ασχολείται με την κρυπτογραφία και την κρυπτανάλυση, είναι η κρυπτολογία (cryptology) και αυτοί που εξασκούν το επάγγελμα αυτό είναι οι cryptologists. Οι σύγχρονοι cryptologists εκπαιδεύονται γενικά σε θεωρητικό μαθηματικό επίπεδο. Στην πράξη με τον όρο κρυπτογραφία συνήθως αναφερόμαστε συνολικά στην κρυπτολογία.

Το plaintext δηλώνεται με το γράμμα M, από το μήνυμα, ή με το γράμμα P, από το plaintext. Το plaintext μπορεί να είναι ένα ρεύμα (stream) από bits, ένα αρχείο κειμένου, ένα αρχείο εικόνας, ένα ρεύμα ψηφιακής φωνής, μια ψηφιακή βίντεο εικόνα κλπ.. Όσον αφορά σε έναν υπολογιστή, το M είναι απλά δυαδικά δεδομένα. Το plaintext μπορεί να αποτελεί είτε τη μεταφορά είτε την αποθήκευση δεδομένων. Ουσιαστικά δηλαδή το M είναι το μήνυμα που κρυπτογραφείται.

Ας υποθέσουμε τώρα ότι το ciphertext δηλώνεται με το γράμμα C. Είναι επίσης δυαδικά δεδομένα, μερικές φορές του ίδιου μεγέθους με το M και κάποιες άλλες μπορεί και μεγαλύτερου. (Η κρυπτογράφηση όμως σε συνδυασμό με τη συμπίεση, μπορεί να οδηγήσει ώστε

το C να είναι μικρότερο από το M, χωρίς όμως η κρυπτογράφηση να μπορεί να εκτελεστεί απευθείας.) Η συνάρτηση κρυπτογράφησης E, επιδρά πάνω στο M για να παράγει το C. Η μαθηματική αυτή σχέση είναι:

$$E(M) = C$$

Στην αντίστροφη διαδικασία, η συνάρτηση αποκρυπτογράφησης D επιδρά πάνω στο C για να παραγάγει το M:

$$D(C) = M$$

Δεδομένου ότι η ουσία της κρυπτογράφησης και έπειτα της αποκρυπτογράφησης ενός μηνύματος είναι να ανακτηθεί το αρχικό plaintext, η ακόλουθη σχέση αποδεικνύει αυτό:

$$D(E(M)) = M$$

2.3.3 Οι έννοιες Authentication, Integrity και Nonrepudiation

Εκτός από την παροχή της εμπιστευτικότητας, το σύστημα της κρυπτογραφίας καλείται συχνά να παρέχει και άλλες υπηρεσίες όπως:

- **Authentication (Αυθεντικοποίηση).** Πρέπει να είναι σε θέση ο δέκτης να εξακριβώνει εάν το μήνυμα ανήκει πράγματι στον αποστολέα ή ότι ο αποστολέας είναι αυτός που ισχυρίζεται ότι είναι.

- **Integrity (Ακεραιότητα).** Πρέπει να είναι σε θέση ο δέκτης να ελέγχει εάν το μήνυμα δεν έχει τροποποιηθεί κατά τη μεταφορά, ώστε να μην μπορεί κάποιος εισβολέας να αντικαταστήσει το μήνυμα με ένα ψεύτικο και αυτό να φαίνεται ως αληθινό.

- **Nonrepudiation.** Ένας αποστολέας δεν μπορεί αργότερα να αρνηθεί ψευδώς ότι έστειλε το μήνυμα.

Αυτές οι έννοιες είναι ζωτικής σημασίας για την κοινωνική αλληλεπίδραση με τη χρήση υπολογιστών, και είναι ανάλογες με τις διαπροσωπικές αλληλεπιδράσεις. Ορισμένα παραδείγματα όπου μπορούν να εφαρμοστούν οι παραπάνω υπηρεσίες είναι:

- Ότι κάποιος είναι αυτός που λέει ότι είναι (Αυθεντικοποίηση).

- Ότι η άδεια ενός οδηγού, το ιατρικό δίπλωμα, και το διαβατήριο είναι έγκυρα (Υπογραφή).

- Ότι ένα έγγραφο είναι απόλυτα βέβαιο ότι έχει προέλθει από κάποιο πρόσωπο (Nonrepudiation).

- Ότι ένα έγγραφο δεν έχει αλλοιωθεί ή τροποποιηθεί (Ακεραιότητα).

2.3.4 Αλγόριθμοι και κλειδιά της κρυπτογραφίας

Ένας κρυπτογραφικός αλγόριθμος (αποκαλείται και ως cipher), είναι η μαθηματική συνάρτηση που χρησιμοποιείται για την κρυπτογράφηση και την αποκρυπτογράφηση. Γενικά, υπάρχουν δύο σχετικές συναρτήσεις: μια για την κρυπτογράφηση και μια άλλη για την αποκρυπτογράφηση.

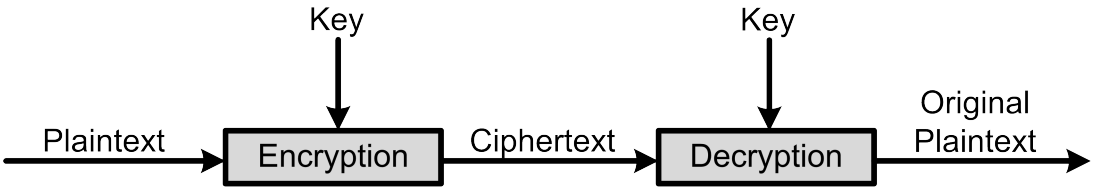
Εάν η ασφάλεια ενός αλγορίθμου είναι βασισμένη στο μυστικό τρόπο λειτουργίας του αλγόριθμου, τότε ένας τέτοιος αλγόριθμος είναι περιορισμένος (restricted). Τέτοιοι περιορισμένοι αλγόριθμοι έχουν μόνο ιστορικό ενδιαφέρον και θεωρούνται ανεπαρκείς για τα σημερινά πρότυπα. Μια μεγάλη ή μεταβαλλόμενη ομάδα χρηστών δεν μπορεί να τους χρησιμοποιήσει, επειδή κάθε φορά που ένας χρήστης αλλάζει ομάδα θα πρέπει να χρησιμοποιήσει και έναν διαφορετικό αλγόριθμο. Επίσης εάν κάποιος αποκαλύψει τυχαία το μυστικό, τότε ο καθένας θα πρέπει να αλλάξει τον αλγόριθμό.

Επιπλέον, οι περιορισμένοι αυτοί αλγόριθμοι δεν επιτρέπουν κανέναν ποιοτικό έλεγχο ή τυποποίηση. Κάθε ομάδα χρηστών πρέπει να έχει έναν μοναδικό αλγόριθμο. Μια τέτοια ομάδα δεν μπορεί να χρησιμοποιήσει έτοιμα προϊόντα υλικού ή λογισμικού, επειδή οποιοσδήποτε θα μπορούσε να αγοράσει το ίδιο προϊόν και να μάθει τον αλγόριθμο, οπότε θα πρέπει να γράψουν οι ίδιοι τους αλγορίθμους και τις εφαρμογές τους. Εάν κανένας στην ομάδα δεν είναι καλός cryptographer, δεν θα μπορούν να ξέρουν εάν ο αλγόριθμος τους είναι ασφαλής.

Παρόλα αυτά τα σημαντικά μειονεκτήματα, οι περιορισμένοι αλγόριθμοι είναι πάρα πολύ δημοφιλείς για εφαρμογές χαμηλής ασφάλειας. Στις περιπτώσεις αυτές οι χρήστες είτε δεν αντιλαμβάνονται είτε δεν ενδιαφέρονται για τα προβλήματα ασφάλειας που υπάρχουν στο σύστημά τους.

Τα σύγχρονα συστήματα κρυπτογραφίας λύνουν αυτό το πρόβλημα με τη χρήση ενός κλειδιού, που δηλώνεται εάς υποθέσουμε με το γράμμα K. Αυτό το κλειδί μπορεί να είναι οποιοδήποτε από ένα μεγάλο αριθμό τιμών. Η περιοχή των πιθανών τιμών του κλειδιού καλείται keyspace. Οπότε τώρα οι διαδικασίες της κρυπτογράφησης και της αποκρυπτογράφησης χρησιμοποιούν αυτό το κλειδί (δηλαδή, εξαρτώνται από αυτό το κλειδί και αυτό το γεγονός δηλώνεται από τον δείκτη K), έτσι οι συναρτήσεις τώρα γίνονται (δείτε το Σχήμα 2.2):

$$E_K(M) = C$$
$$D_K(C) = M$$
$$D_K(E_K(M)) = M$$

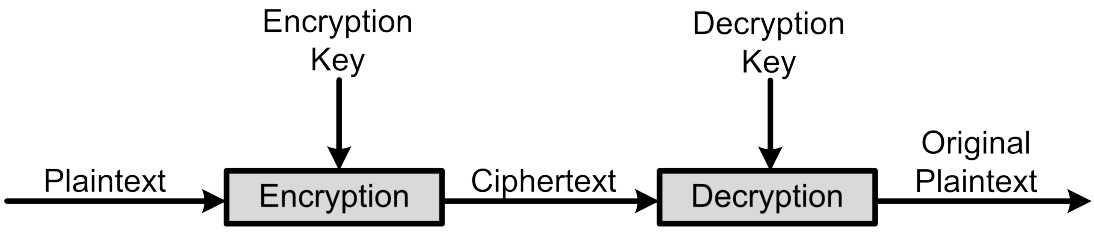


Σχήμα 2.2: Κρυπτογράφηση και αποκρυπτογράφηση με ένα κλειδί.

Μερικοί αλγόριθμοι χρησιμοποιούν διαφορετικά κλειδιά στην κρυπτογράφηση και στην

αποκρυπτογράφηση (δείτε το Σχήμα 2.3). Δηλαδή το κλειδί κρυπτογράφησης, K_1, είναι διαφορετικό από το αντίστοιχο κλειδί αποκρυπτογράφησης, K_2. Σε αυτήν την περίπτωση οι συναρτήσεις γίνονται:

$$E_{K_1}(M) = C$$
$$D_{K_2}(C) = M$$
$$D_{K_2}(E_{K_1}(M)) = M$$



Σχήμα 2.3: Κρυπτογράφηση και αποκρυπτογράφηση με δύο διαφορετικά κλειδιά.

Όλη η ασφάλεια σε αυτούς τους αλγόριθμους είναι βασισμένη στο κλειδί (ή τα κλειδιά) και όχι στις λεπτομέρειες του αλγορίθμου. Αυτό σημαίνει ότι ο αλγόριθμος μπορεί να δημοσιευθεί και να αναλυθεί, οπότε και τα προϊόντα που χρησιμοποιούν τον αλγόριθμο μπορούν να παραχθούν μαζικά. Επίσης, δεν πειράζει εάν κάποιος ξέρει τον αλγόριθμό σας, επειδή δεν ξέρει το ιδιαίτερο κλειδί σας, δεν μπορεί να διαβάσει τα μηνύματά σας. Έτσι λοιπόν, ένα σύγχρονο κρυπτογραφικό σύστημα αποτελείται πλέον από τον αλγόριθμο, όλα τα πιθανά plaintexts, τα ciphertexts και τα κλειδιά.

2.3.5 Συμμετρικοί Αλγόριθμοι

Υπάρχουν δύο γενικοί τύποι αλγορίθμων βασισμένων σε κλειδιά: οι συμμετρικοί αλγόριθμοι και οι αλγόριθμοι δημόσιου-κλειδιού (public-key). Οι συμμετρικοί αλγόριθμοι, οι οποίοι αποκαλούνται μερικές φορές και συμβατικοί αλγόριθμοι, είναι αλγόριθμοι όπου το κλειδί κρυπτογράφησης μπορεί να υπολογιστεί από το κλειδί αποκρυπτογράφησης και αντίστροφα. Στους περισσότερους συμμετρικούς αλγορίθμους το κλειδί κρυπτογράφησης και το κλειδί αποκρυπτογράφησης είναι τα ίδια. Αυτοί οι αλγόριθμοι αποκαλούνται επίσης και αλγόριθμοι μοναδικού-κλειδιού και απαιτούν ο αποστολέας και ο δέκτης να συμφωνούν σχετικά με ένα κλειδί προτού να μπορέσουν να επικοινωνήσουν με ασφάλεια. Η ασφάλεια ενός συμμετρικού αλγορίθμου στηρίζεται στο κλειδί, χωρίς να υπάρχει λόγος απόκρυψης των βασικών σημείων κρυπτογράφησης και αποκρυπτογράφησης. Εφόσον πρέπει να παραμείνει η επικοινωνία μυστική, το κλειδί πρέπει να παραμείνει μυστικό.

Η κρυπτογράφηση και η αποκρυπτογράφηση με έναν συμμετρικό αλγόριθμο δείχνονται από τις σχέσεις:

$$E_K(M) = C$$
$$D_K(C) = M$$

Οι συμμετρικοί αλγόριθμοι μπορούν να διαιρεθούν σε δύο κατηγορίες:

1. Σε αυτούς που το plaintext είναι ένα bit (ή μερικές φορές ένα byte) την κάθε φορά και ονομάζονται αλγόριθμοι ρευμάτων (stream) ή ρεύματα (stream) ciphers.

2. Και σε αυτούς που το plaintext είναι ομάδες από bits. Οι ομάδες των bits καλούνται blocks, και οι αλγόριθμοι αυτοί ονομάζονται αλγόριθμοι block ή blocks ciphers. Για τους σύγχρονους αλγορίθμους υπολογιστών, ένα χαρακτηριστικό μέγεθος blocks είναι 64 bits, αρκετά μεγάλο για να αποτρέψει την ανάλυση και αρκετά μικρό για να είναι εφαρμόσιμο.

2.3.6 Αλγόριθμοι δημόσιου-κλειδιού

Οι αλγόριθμοι δημόσιου-κλειδιού (public-key) (επίσης αποκαλούνται και ασύμμετροι αλγόριθμοι) σχεδιάστηκαν έτσι ώστε το κλειδί που χρησιμοποιείται για την κρυπτογράφηση να είναι διαφορετικό από το κλειδί που χρησιμοποιείται για την αποκρυπτογράφηση. Ακόμη, το κλειδί αποκρυπτογράφησης δεν μπορεί (τουλάχιστον σε λογικό χρονικό διάστημα) να υπολογιστεί από το κλειδί κρυπτογράφησης. Οι αλγόριθμοι αυτοί καλούνται ως "public-key" επειδή το κλειδί κρυπτογράφησης μπορεί να δημοσιοποιηθεί: Ένας άγνωστος μπορεί να χρησιμοποιήσει το κλειδί κρυπτογράφησης για να κρυπτογραφήσει ένα μήνυμα, αλλά μόνο ένα συγκεκριμένο πρόσωπο με το αντίστοιχο κλειδί αποκρυπτογράφησης μπορεί να αποκρυπτογραφήσει το μήνυμα. Σε αυτά τα συστήματα, το κλειδί κρυπτογράφησης καλείται συχνά δημόσιο κλειδί, και το κλειδί αποκρυπτογράφησης καλείται συχνά ιδιωτικό κλειδί. Το ιδιωτικό κλειδί μερικές φορές επίσης καλείται μυστικό (secret) κλειδί, αλλά για να αποφευχθεί η σύγχυση με τους συμμετρικούς αλγόριθμους, αυτή η φράση γενικά δεν χρησιμοποιείται. Η κρυπτογράφηση που χρησιμοποιεί το δημόσιο κλειδί K μπορεί να περιγραφεί με την ακόλουθη συνάρτηση:

$$E_{K_d}(M) = C$$

Η αποκρυπτογράφηση με το ιδιωτικό κλειδί περιγράφεται με τη συνάρτηση:

$$D_{K_i}(C) = M$$

Μερικές φορές, τα μηνύματα κρυπτογραφούνται με το ιδιωτικό κλειδί και αποκρυπτογραφούνται με το δημόσιο κλειδί. Συγκεκριμένα, αυτό χρησιμοποιείται στις ψηφιακές υπογραφές. Παρά την πιθανή σύγχυση, αυτές οι διαδικασίες περιγράφονται με τις συναρτήσεις:

$$E_{K_i}(M) = C$$
$$D_{K_d}(C) = M$$

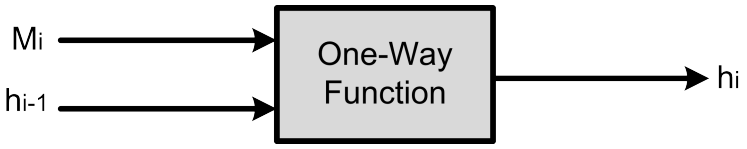
2.4 Αλγόριθμοι κρυπτογραφίας hash και SHA

2.4.1 Μονόδρομες hash συναρτήσεις

Μια hash συνάρτηση είναι μια συνάρτηση που δέχεται σαν είσοδο δεδομένα αυθαίρετου μήκους και τα μετατρέπει σε μονόδρομα δεδομένα (χωρίς να υπάρχει δυνατότητα επαναφοράς στην αρχική τους μορφή). Στην πραγματικότητα, οι μονόδρομες hash συναρτήσεις στηρίζονται

στην ιδέα μιας συνάρτησης συμπίεσης. Αποτελέσματα αυτής της μονόδρομης συνάρτησης είναι μια μεταβλητή hash με μήκος n, για δεδομένη είσοδο μεγαλύτερου μήκους m. Οι είσοδοι της συνάρτησης συμπίεσης είναι ένα block μηνύματος και η έξοδος του προηγούμενου block από το κείμενο (Σχήμα 2.4). Η έξοδος της συνάρτησης είναι το hash όλων των blocks μέχρι εκείνο το σημείο. Δηλαδή το hash του βλοχκ M_i θα είναι:

$$h_i = f(M_i, h_{i-1})$$



Σχήμα 2.4: Μονόδρομη hash συνάρτηση.

Αυτή η hash μεταβλητή, μαζί με το επόμενο block μήνυμα, γίνεται η επόμενη είσοδος της συνάρτησης συμπίεσης. Το hash ολόκληρου του μηνύματος είναι το hash του τελευταίου block.

Η προ-εικόνα (δηλαδή, πριν ξεκινήσει ακόμη η διαδικασία) πρέπει να περιέχει κάποια δυαδική αντιπροσώπευση του μήκους ολόκληρου του μηνύματος. Αυτή η τεχνική ξεπερνά ένα πιθανό πρόβλημα ασφάλειας σε μηνύματα που έχουν ενδεχομένως διαφορετικά μήκη hashing για την ίδια μεταβλητή. Αυτή η τεχνική μερικές φορές καλείται και ως MD-strengthening.

Διάφοροι ερευνητές έχουν εξετάσει σε θεωρητικό επίπεδο ότι εάν η συνάρτηση συμπίεσης είναι ασφαλής, τότε και η μέθοδος hashing αυθαίρετου μήκους με προ-εικόνα είναι επίσης ασφαλής, αλλά τίποτα δεν έχει αποδειχθεί.

2.4.2 SHA hash συναρτήσεις

Η οικογένεια του SHA (Secure Hash Algorithm) είναι ένα σύνολο σχετικών κρυπτογραφικών hash συναρτήσεων. Η συνηθέστερα χρησιμοποιούμενη συνάρτηση της οικογένειας αυτής, ο SHA-1, υιοθετείται σε μια μεγάλη ποικιλία δημοφιλών εφαρμογών και πρωτοκόλλων ασφάλειας, συμπεριλαμβανομένου του TLS, SSL, PGP, SSH, S/MIME, και IPSec. Ο SHA-1 θεωρείται ο διάδοχος του MD5, μιας προηγούμενης, ευρέως χρησιμοποιούμενης hash συνάρτησης. Σε μερικές περιπτώσεις, με ιδιαίτερες απαιτήσεις ασφάλειας, προτείνεται η χρήση του SHA-256 ή μεγαλύτερου. Οι αλγόριθμοι SHA σχεδιάστηκαν από την NSA (National Security Agency) και δημοσιεύθηκαν ως πρότυπα της αμερικανικής κυβέρνησης.

Το πρώτο μέλος της οικογένειας, δημοσιεύτηκε το 1993 και ονομάζεται επίσημα SHA. Εντούτοις, καλείται συχνά ως SHA-0 για να αποφευχθεί η σύγχυση με τους διαδόχους του. Δύο έτη αργότερα δημοσιεύθηκε ο SHA-1, που είναι ο πρώτος διάδοχος του SHA. Από τότε, τέσσερις παραλλαγές του SHA έχουν δημοσιευτεί για να καλύψουν τις αυξανόμενες ανάγκες ασφαλείας: SHA-224, SHA-256, SHA-384, και SHA-512 (μερικές φορές συλλογικά αναφέρονται ως SHA-2). Στο Σχήμα 2.5 φαίνονται τα διάφορα είδη του SHA.

Algorithm	Output size	Internal state size	Block size	Length size	Word size	Passes	Operations	Collision
SHA-0	160	160	512	64	32	80	+,and,or,xor,rotl	Yes
SHA-1	160	160	512	64	32	80	+,and,or,xor,rotl	With flaws
SHA-256/224	256/224	256	512	64	32	64	+,and,or,xor,shr,rotr	No
SHA-512/384	512/384	512	1024	128	64	80	+,and,or,xor,shr,rotr	No

Σχήμα 2.5: Τα διάφορα μεγέθη του SHA.

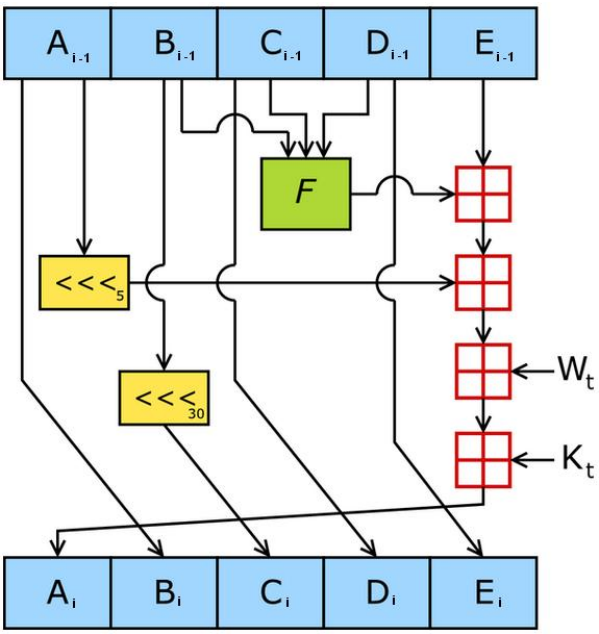
2.4.3 SHA-1 και SHA-2

Ο SHA-1 παράγει μια συγχώνευση 160-bits από ένα μήνυμα με μέγιστο μέγεθος 2^{64} bits, και είναι βασισμένος σε αρχές παρόμοιες με εκείνες που χρησιμοποιήθηκαν από τους αλγόριθμους MD4 και MD5. Ο SHA-1 είναι ασφαλής επειδή έχει ως σκοπό να είναι υπολογιστικά ανέφικτο να ανακτηθεί ένα μήνυμα που αντιστοιχεί σε μια δεδομένη συγχώνευση μηνυμάτων, ή για να βρεθούν δύο διαφορετικά μηνύματα που να παράγουν την ίδια συγχώνευση μηνυμάτων. Οποιαδήποτε αλλαγή συμβεί σε ένα μήνυμα κατά τη μεταφορά, με μεγάλη πιθανότητα θα οδηγήσει σε μια διαφορετική συγχώνευση μηνυμάτων, και ο έλεγχος της υπογραφής θα αποτύχει. Στο Σχήμα 2.6 φαίνεται μία επανάληψη της συνάρτησης συμπίεσης του SHA-1. Εκεί τα A, B, C, D και E είναι λέξεις των 32-bits μιας κατάστασης, το F είναι μια μη γραμμική συνάρτηση, το $<<<_n$ δηλώνει μια αριστερή μετατόπιση κατά n-bits, το + δείχνει την πρόσθεση modulo 2^{32}, το i-1 είναι η παρούσα κατάσταση, ενώ το i η επόμενη κατάσταση, και το K_t είναι μια σταθερά.

Η οικογένεια του SHA-2 χρησιμοποιεί παρόμοια λογική με τον SHA-1 με την διαφορά ότι ανάλογα με την έκδοση του χρησιμοποιεί λέξεις των 32-bits (στον SHA-256 και SHA-224) και των 64-bits (στον SHA-512 και SHA-384). Επιπρόσθετα, πραγματοποιεί μετατόπιση διαφορετικού αριθμού από bits και χρησιμοποιεί διαφορετικές τιμές σταθερών (ανάλογα με την έκδοση), χωρίς όμως η βασική δομή του να διαφέρει ουσιαστικά.

2.4.4 Ασφάλεια του SHA

Επιθέσεις έχουν βρεθεί για τον SHA-0 και τον SHA-1, όπως φαίνεται άλλωστε και στη στήλη "Collision" του Σχήματος 2.5. Καμία επίθεση δεν έχει αναφερθεί ακόμα για τις διάφορες παραλλαγές του SHA-2, αλλά δεδομένου ότι είναι παρόμοιοι με τον SHA-1 οι ερευνητές ανησυχούν. Για αυτό το λόγο αναπτύσσουν νέα, καλύτερα hashing πρότυπα του SHA. Ένα νέο hashing πρότυπο, ο SHA-3, αυτή τη στιγμή βρίσκεται στο στάδιο της ανάπτυξης και αναμένεται να έχει ολοκληρωθεί στο τέλος του 2012.



Σχήμα 2.6: Μια επανάληψη της συνάρτησης συμπίεσης του SHA-1.

2.5 Κρυπτογραφικό Σύστημα ElGamal

Το σύστημα κρυπτογράφησης ElGamal είναι ένας ασύμμετρος αλγόριθμος κρυπτογράφησης δημοσίου κλειδιού και βασίζεται στη βασική ιδέα των Diffie-Hellman. Για πρώτη φορά περιγράφτηκε από τον Taher Elgamal το 1984. Η ασφάλεια της κρυπτογράφησης του ElGamal βασίζεται στο πρόβλημα του Διακριτού Λογάριθμου. Στις ενότητες που ακολουθούν γίνεται αναλυτική περιγραφή του αλγορίθμου.

2.5.1 Αλγόριθμος ElGamal

Ο αλγόριθμος ElGamal αποτελείται από τρία κύρια μέρη: τη δημιουργία κλειδιών, τον αλγόριθμο κρυπτογράφησης, και τον αλγόριθμο αποκρυπτογράφησης.

Δημιουργία κλειδιών

Η διαδικασία που πρέπει να ακολουθηθεί για τη δημιουργία ενός ζεύγους (δημόσιου και ιδιωτικού) κλειδιών είναι η εξής:

1. Επιλέγουμε ένα τυχαίο μεγάλο και πρώτο αριθμό p και ένα πρωταρχικό στοιχείο/γεννήτορα g από το σύνολο ${Z_p}^*$, όπου ${Z_p}^*$ συμβολίζουμε το σύνολο όλων των ακέραιων $\{1, 2, \ldots, p-1\}$, δηλαδή $g^k \neq 1 \ mod \ p$ για όλα τα k μικρότερα του $p-1$.
2. Επιλέγουμε ένα τυχαίο αριθμό a στο διάστημα $1 \leq a \leq p-1$ ως το ιδιωτικό κλειδί.
3. Υπολογίζουμε το $y = g^a \ mod \ p$.
4. Το δημόσιο κλειδί είναι το (p, g, y) και το ιδιωτικό κλειδί είναι το a.

Για την εύρεση του γεννήτορα g θα πρέπει να σημειώσουμε τα εξής:

- Αν ισχύει το $g^k = 1 \ mod \ p$ για κάποιον ακέραιο αριθμό $1 \leq k \leq p-1$, τότε ο αριθμός k υποχρεωτικά διαιρεί τον $p-1$.
- Άρα, αν θέλουμε να ελέγξουμε αν ένας αριθμός g είναι γεννήτορας $mod \ p$, δε χρειάζεται να τον υψώσουμε σε όλες τις δυνάμεις $\{1, 2, \ldots, p-1\}$, αρκεί να τον υψώσουμε στους διαιρέτες του $p-1$.

Αλγόριθμος κρυπτογράφησης

Ο αλγόριθμος που ακολουθείται για την κρυπτογράφηση ενός μηνύματος είναι η εξής:

1. Επιλέγουμε έναν τυχαίο αριθμό r από το σύνολο $\{1, 2, \ldots, p-1\}$.
2. Εκφράζουμε το μήνυμα με έναν ακέραιο αριθμό m από τους $\{1, 2, \ldots, p-1\}$.
3. Υπολογίζουμε το $g = g^r \ mod \ p$ και το $d = my^r \ mod \ p$.
4. Οπότε το ciphertext είναι το $E_k(m, r) = (g, d)$.

Αλγόριθμος αποκρυπτογράφησης

Ο αλγόριθμος που ακολουθείται για την αποκρυπτογράφηση του αρχικού μηνύματος είναι ο εξής:

1. Υπολογίζουμε το g^{-a}, αφού γνωρίζουμε το ιδιωτικό κλειδί.
2. Το αρχικό μήνυμα θα είναι το $m = (g^{-a})d \ mod \ p$.
3. Με άλλα λόγια η ανάκτηση του αρχικού μηνύματος γίνεται με την πράξη $\frac{d}{g^a}$.
4. Οπότε το αρχικό plaintext είναι το $D_k(g, d) = m \ (mod \ p)$.

2.5.2 Ασφάλεια του ElGamal

Ο αντίπαλος που θα επιχειρήσει επίθεση στο κρυπτοσύστημα, θα πρέπει να ανακτήσει το ιδιωτικό κλειδί a, από τη σχέση:

$$y = g^a \ mod \ p$$

γνωρίζοντας τα p, g, y. Θα πρέπει δηλαδή να λύσει το διακριτό λογάριθμο με βάση g. Ωστόσο, θεωρούμε ότι η ασφάλεια του κρυπτοσυστήματος ElGamal βασίζεται στο διακριτό λογάριθμο, διότι η λύση του διακριτού λογάριθμου μπορεί να καθιστά το κρυπτοσύστημα ανασφαλές, αλλά δεν έχει αποδειχθεί το αντίστροφο, ότι δηλαδή η ασφάλεια του κρυπτοσυστήματος στηρίζεται αποκλειστικά στο πρόβλημα του διακριτού λογάριθμου.

Η ύπαρξη του τυχαίου αριθμού r, έχει ως αποτέλεσμα τη δυνατότητα αντιστοίχισης του απλού κειμένου σε $p-1$ κρυπτοκείμενα. Η διαδικασία όπου το απλό κείμενο αναμειγνύεται με

μια τυχαία μεταβλητή, ονομάζεται διαδικασία δημιουργίας συνθηκών τυχαιότητας (randomization process). Το βήμα αυτό το οποίο δεν υπάρχει στο RSA, καθιστά το κρυπτοσύστημα ElGamal ανθεκτικότερο σε επιθέσεις παρόμοιες με αυτές που παρουσιάζονται στο RSA. Βέβαια, η χρήση του τυχαίου αριθμού εισάγει έναν επιπλέον κίνδυνο που οδηγεί σε μια πρόσθετη απαίτηση. Για κάθε μήνυμα που κρυπτογραφείται, θα πρέπει να επιλέγεται διαφορετικός τυχαίος r. Στην περίπτωση που δύο μηνύματα m και m' κρυπτογραφηθούν με τον ίδιο r, τότε για τα αντίστοιχα κρυπτοκείμενα που θα προκύψουν (g, d) και (g', d'), η γνώση του ενός μηνύματος επιτρέπει την ανάκτηση του άλλου από τον λόγο:

$$\frac{d}{d'} = \frac{m \cdot y^r}{m' \cdot y^r} = \frac{m}{m'}$$

Τέλος, όσον αφορά το μέγεθος του p, το κατώτατο όριο που προτείνεται είναι 1024 bits. Γενικά, κατά την κρυπτογράφηση με το κρυπτοσύστημα ElGamal, το μέγεθος των παραμέτρων αποτελεί σημαντικό κριτήριο υλοποίησης, λόγω του αυξημένου χρόνου που απαιτείται για την κρυπτογράφηση (δύο πράξεις ύψωσης σε δύναμη έναντι της μιας στην περίπτωση του RSA), και λόγω της διαστολής του κρυπτοκειμένου. Τα μειονεκτήματα αυτά έχουν σαν αποτέλεσμα να προτιμάται μειωμένο μέγεθος του modulus.

2.6 Ομομορφική κρυπτογράφηση (Homomorphic encryption)

Η Ομομορφική κρυπτογράφηση (Homomorphic encryption) είναι μια μορφή κρυπτογράφησης που μπορεί να εκτελέσει μια συγκεκριμένη αλγεβρική πράξη στο plaintext με την εκτέλεση μιας (ενδεχομένως διαφορετικής) αλγεβρικής πράξης στο ciphertext. Αυτή η ιδιότητα μπορεί να έχει τόσο θετικές όσο και αρνητικές επιπτώσεις σε ένα κρυπτογραφικό σύστημα. Έτσι λοιπόν, το μοντέλο της ομομορφικής κρυπτογράφησης είναι ευάλωτο σε κακόβουλες επιθέσεις από το σχεδιασμό της, με αποτέλεσμα να είναι ακατάλληλο για την ασφαλή μετάδοση δεδομένων. Όμως, η ομομορφική ιδιότητα των διάφορων κρυπτογραφικών συστημάτων μπορεί να χρησιμοποιηθεί για τη δημιουργία ασφαλών εκλογικών συστημάτων, ανθεκτικών hash συναρτήσεων και μοντέλων ιδιωτικής ανάκτησης πληροφοριών (private information retrieval).

Παρακάτω παρουσιάζονται διάφορα αποδοτικά ομομορφικά κρυπτογραφικά συστήματα μαζί με την αντίστοιχη ομομορφική ιδιότητα που παρουσιάζουν:

- **Unpadded RSA:** Εάν το δημόσιο κλειδί είναι (e, n) τότε η κρυπτογράφηση ενός μηνύματος x θα δίνεται από $\mathcal{E}(x) = x^e \ mod \ m$. Τότε η ομομορφική ιδιότητα που παρουσιάζει είναι:

$$\mathcal{E}(x_1) \cdot \mathcal{E}(x_2) = x_1^e x_2^e \ mod \ m = (x_1 x_2)^e \ mod \ m = \mathcal{E}(x_1 \cdot x_2)$$

- **ElGamal:** Εάν το δημόσιο κλειδί είναι (p, g, y) και το ιδιωτικό κλειδί είναι το a, όπου $y = g^a \ mod \ p$, τότε η κρυπτογράφηση ενός μηνύματος x θα δίνεται από $\mathcal{E}(x) =$

(g^r, my^r), όπου r ένας τυχαίος αριθμός στο σύνολο $\{1, 2, \ldots, p-1\}$. Τότε η ομομορφική ιδιότητα που παρουσιάζει είναι:

$$\mathcal{E}(x_1) \cdot \mathcal{E}(x_2) = (g^{r_1}, x_1 \cdot y^{r_1})(g^{r_2}, x_2 \cdot y^{r_2}) = (g^{r_1+r_2}, (x_1 \cdot x_2)y^{r_1+r_2}) = \mathcal{E}(x_1 \cdot x_2)$$

- **Goldwasser-Micali:** Εάν το δημόσιο κλειδί είναι modulus m και ένα τετραγωνικό μη-υπόλοιπο (quadratic non-residue) x, τότε η κρυπτογράφηση ενός bit b θα δίνεται από $\mathcal{E}(b) = r^2 x^b \ mod \ m$. Τότε η ομομορφική ιδιότητα που παρουσιάζει είναι:

$$\mathcal{E}(b_1) \cdot \mathcal{E}(b_2) = r_1^2 x^{b_1} r_2^2 x^{b_2} = (r_1 r_2)^2 x^{b_1+b_2} = \mathcal{E}(b_1 \oplus b_2)$$

όπου $\oplus$ δηλώνει ένα επιπλέον modulo 2 (π.χ. αποκλειστικό ή (XOR».

- **Benaloh:** Εάν το δημόσιο κλειδί είναι modulus m και μια βάση g με ένα blocksize r, τότε η κρυπτογράφηση ενός μηνύματος x θα δίνεται από $\mathcal{E}(b) = g^x u^r \ mod \ m$. Τότε η ομομορφική ιδιότητα που παρουσιάζει είναι:

$$\mathcal{E}(x_1) \cdot \mathcal{E}(x_2) = (g^{x_1} u_1^r)(g^{x_2} u_2^r) = g^{x_1+x_2}(u_1 u_2)^r = \mathcal{E}(x_1 + x_2 \ mod \ r)$$

- **Paillier:** Εάν το δημόσιο κλειδί είναι modulus m και μια βάση g, τότε η κρυπτογράφηση ενός μηνύματος x θα δίνεται από $\mathcal{E}(x) = g^x r^m \ mod \ m^2$. Τότε η ομομορφική ιδιότητα που παρουσιάζει είναι:

$$\mathcal{E}(x_1) \cdot \mathcal{E}(x_2) = (g^{x_1} r_1^m)(g^{x_2} r_2^m) = g^{x_1+x_2}(r_1 r_2)^m = \mathcal{E}(x_1 + x_2 \ mod \ m)$$

Τέλος, αξίζει να αναφερθεί ότι υπάρχει μια πρόσφατη εργασία του Craig Gentry [14] που παρουσιάζει μια πλήρης ομομορφική κρυπτογράφηση (fully homomorphic encryption). Αυτή η πλήρης ομομορφική κρυπτογράφηση, παρουσιάζει ταυτόχρονα τόσο την ιδιότητα της πρόσθεσης όσο και του πολλαπλασιασμού με αποτέλεσμα να είναι σε θέση να μπορεί να πραγματοποιήσει σύνθετες πράξεις πάνω στα ciphertexts.

Κεφάλαιο 3

Δίκτυα Ομότιμων Κόμβων

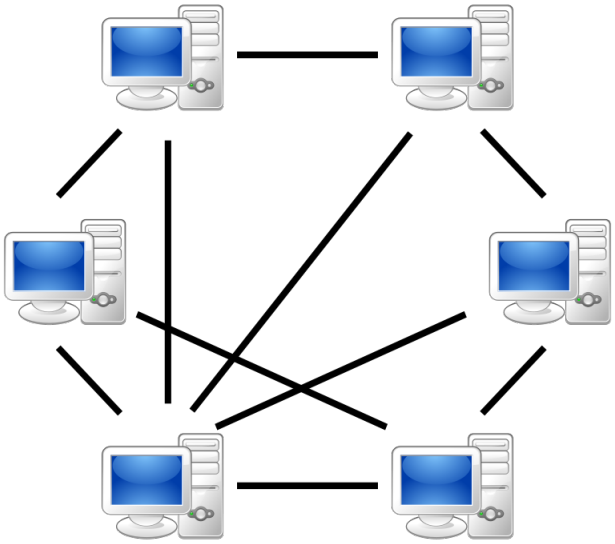
Στο κεφάλαιο αυτό παρουσιάζονται αναλυτικά η έννοια και τα χαρακτηριστικά των δικτύων ομότιμων κόμβων. Περιγράφεται το περιβάλλον στο οποίο λειτουργούν, οι κατηγορίες στις οποίες ταξινομούνται καθώς και τα πλεονεκτήματα τα οποία οδήγησαν στην έρευνα και την εφαρμογή τους.

3.1 Η έννοια του δικτύου ομότιμων κόμβων

Ένα δίκτυο ομότιμων κόμβων (στο εξής P2P) είναι ένα λογικό δίκτυο που επιτρέπει στους κόμβους, που ανήκουν σε αυτό, να μοιράζονται ορισμένους από τους πόρους τους ισοδύναμα. Το δίκτυο αυτό χρησιμοποιεί την επεξεργαστική ισχύ, τον αποθηκευτικό χώρο, το εύρος ζώνης καθώς και άλλες υπηρεσίες (πχ. εκτυπωτές) των κόμβων, που συμμετέχουν σε αυτό. Οι πόροι αυτοί είναι απευθείας προσβάσιμοι από οποιονδήποτε συμμετέχοντα του δικτύου χωρίς τη μεσολάβηση ενδιάμεσων οντοτήτων. Έτσι, οι κόμβοι ενός P2P δικτύου αποτελούν τόσο παροχείς όσο και καταναλωτές υπηρεσιών, σε αντίθεση με αυτούς της παραδοσιακής αρχιτεκτονικής πελάτη-διακομιστή (client-server), σύμφωνα με την οποία οι διακομιστές παράγουν ενώ οι πελάτες καταναλώνουν. Συχνά, έννοιες όπως αποκεντροποιημένο (decentralized) και αυτορυθμιζόμενο (self-organizing) χρησιμοποιούνται για να περιγράψουν P2P δίκτυα, υπό την έννοια ότι ανεξάρτητοι κόμβοι οργανώνονται σε ένα λογικό δίκτυο χωρίς την παρουσία κεντρικού συντονισμού.

Κάθε κόμβος που συμμετέχει στο P2P δίκτυο "τρέχει" το απαραίτητο λογισμικό με το οποίο μπορεί να γίνει η λογική σύνδεση του με τους υπολοίπους. Συχνά, το λογισμικό αυτό αναφέρεται ως πράκτορας (στο εξής agent). Στο μοντέλο που περιγράφεται στην παρούσα εργασία η έννοια του agent θα χρησιμοποιηθεί για την περιγραφή της οντότητας που αποτελείται από τον κόμβο και το λογισμικό του.

Ένα τυπικό παράδειγμα P2P δικτύου παρουσιάζεται στο Σχήμα 3.1, ενώ η αντίστοιχη τοπολογία αρχιτεκτονικής πελάτη-διακομιστή φαίνεται στο Σχήμα 3.2.



Σχήμα 3.1: Τυπική αρχιτεκτονική P2P δικτύου

3.2 Ιστορικά στοιχεία

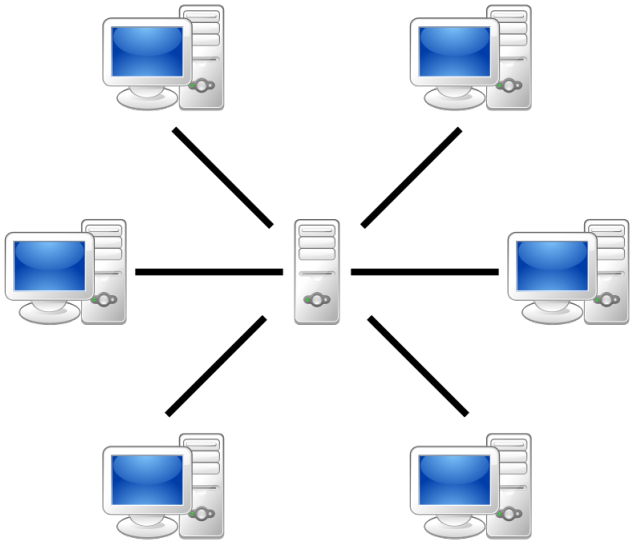
3.2.1 Εξέλιξη των P2P δικτύων

Η διαμοίραση αρχείων πριν την εμφάνιση των P2P

Από τα πρώτα χρόνια της λειτουργίας του διαδικτύου, γεννήθηκε η ανάγκη στους χρήστες παγκοσμίως, να ανακαλύψουν μια μέθοδο, ώστε να μπορούν ανταλλάσσουν μεταξύ τους πληροφορίες και δεδομένα. Η αρχή έγινε το 1979 με το Usenet, που αρχικά δημιουργήθηκε ως ένα δίκτυο με το οποίο θα μπορούσαν να μεταδίδονται νέα και πληροφορίες ανάμεσα στους φοιτητές των πανεπιστημίων University of North Carolina και Duke University, ενώ στη συνέχεια έγινε εφικτή και η ανταλλαγή μικρών αρχείων με αρκετούς, όμως, περιορισμούς. Τα επόμενα χρόνια, για την ανταλλαγή αρχείων μεταξύ των χρηστών του διαδικτύου, αναπτύχθηκαν τα πρωτόκολλα ftp τα οποία επέτρεπαν ανταλλαγή αρχείων μέσα από ένα μοντέλο server-client.

Η εμφάνιση του Napster ως πρώτου P2P δικτύου

Η μεγάλη επανάσταση όμως στο διαμοιρασμό αρχείων έγινε το 1999 όταν εμφανίστηκε το πρώτο P2P δίκτυο, το περίφημο πλέον Napster, δημιουργός του οποίου ήταν ο, μόλις δεκαοκτώ τότε ετών, Shawn Fanning. Αυτή η πρωτοποριακή τεχνολογία επέτρεπε σε χρήστες από όλο τον κόσμο να αναζητούν και να "κατεβάζουν" τραγούδια, σε μορφή MP3, τα οποία ήταν



Σχήμα 3.2: Τυπική αρχιτεκτονική πελάτη-διακομιστή

αποθηκευμένα σε υπολογιστές άλλων χρηστών της υπηρεσίας. Για να είναι εφικτή η αναζήτηση των αρχείων αυτών, το Napster χρησιμοποιούσε έναν κεντρικό server, στον οποίο ήταν αποθηκευμένοι οι τίτλοι όλων των τραγουδιών, τα οποία οι χρήστες της υπηρεσίας διέθεταν για διαμοιρασμό, αλλά και οι ηλεκτρονικές διευθύνσεις των χρηστών αυτών. Όταν κάποιος χρήστης πραγματοποιούσε μια αναζήτηση, ο server εντόπιζε τους χρήστες που διέθεταν το επιθυμητό μουσικό κομμάτι και στην συνέχεια δημιουργούσε μια σύνδεση μεταξύ του υπολογιστή του χρήστη που έκανε την αναζήτηση και ενός εκ των υπολογιστών των χρηστών που διέθεταν το τραγούδι, ώστε να πραγματοποιηθεί ο διαμοιρασμός του αρχείου.

Το Napster αμέσως κατηγορήθηκε πως παραβίαζε τα πνευματικά δικαιώματα των δημιουργών των τραγουδιών και πως χρησιμοποιούταν για την παράνομη ανταλλαγή προστατευμένων αρχείων. Για τον λόγο αυτό, έπειτα από δικαστική μάχη με την αμερικάνικη ένωση δισκογραφικών (Recording Industry Association of America - RIAA), αναγκάστηκε το 2001 να αναστείλει την λειτουργία του. Σήμερα λειτουργεί ως υπηρεσία επί πληρωμή.

Gnutella

Τον Μάρτιο του 2000 εμφανίστηκε το πρώτο αποκεντρωτικό P2P δίκτυο, το Gnutella. Δημιουργοί του ήταν οι Justin Frankel και Tom Pepper, της εταιρίας Nullsoft. Η μεγάλη διαφοροποίηση σε σχέση με το Napster ήταν πως στο Gnutella απουσίαζε ο κεντρικός εξυ-

πηρετητής και πλέον κάθε κόμβος θεωρούταν ισάξιος με τους υπόλοιπους. Λόγω όμως των νομικών προβλημάτων που αντιμετώπισε εκείνη την περίοδο το Napster, τα οποία και οδήγησαν τελικά στο κλείσιμο του, η Nullsoft αποφάσισε πολύ γρήγορα να διακόψει τη λειτουργία του Gnutella. Παρόλ' αυτά, πολλοί προγραμματιστές, εμπνευσμένοι από το Gnutella και βασιζόμενοι στην αρχιτεκτονική του, δημιούργησαν παρεμφερή αποκεντρωτικά P2P δίκτυα και agents, με αποτέλεσμα σήμερα, με το όνομα Gnutella να αναφερόμαστε στο σύνολο αυτών των δικτύων και όχι στον αρχικό agent. Στο Gnutella, αλλά και στα περισσότερα αποκεντρωτικά P2P δίκτυα που ακολούθησαν, η αναζήτηση των αρχείων γίνεται με τον εξής τρόπο: αρχικά ο χρήστης που επιθυμεί να κατεβάσει κάποιο αρχείο στέλνει ένα αίτημα σε κοντινούς σε αυτόν κόμβους. Σε περίπτωση που κάποιος από αυτούς τους κόμβους έχει το αρχείο που ζητείται, δημιουργεί μια σύνδεση με τον υπολογιστή του χρήστη και γίνεται η μεταφορά του αρχείου. Σε αντίθετη περίπτωση, οι κόμβοι αυτοί στέλνουν ειδοποίηση σε άλλους κόμβους, μέχρι τελικά να βρεθεί κάποιος κόμβος που να έχει το ζητούμενο αρχείο. Σε μερικά δίκτυα που βασίζονται στην αρχιτεκτονική του Gnutella υπάρχουν κάποιοι κόμβοι, γνωστοί ως "υπερκόμβοι" , οι οποίοι έχουν περισσότερα δικαιώματα από τους υπόλοιπους κόμβους των δικτύων και αναλαμβάνουν την δρομολόγηση των αναζητήσεων και την σύνδεση των άλλων κόμβων μεταξύ τους. Με τον τρόπο αυτό αυξάνεται η αποδοτικότητα του δικτύου.

Το πρωτόκολλο BitTorrent

Σήμερα ένα πολύ μεγάλο μερίδιο των P2P δικτύων χρησιμοποιούν το πρωτόκολλο BitTorrent. Το πρωτόκολλο αυτό έκανε την εμφάνιση του το 2002 και χρησιμοποιείται κυρίως για την μεταφορά αρχείων μεγάλου μεγέθους. Ο τρόπος λειτουργίας των P2P δικτύων που χρησιμοποιούν αυτό το πρωτόκολλο παρουσιάζει πολλές διαφορές σε σχέση με τα υπόλοιπα P2P δίκτυα. Κάθε αρχείο δεδομένων, που διατίθεται για διαμοιρασμό σε ένα τέτοιο δίκτυο, "σπάει" σε πολλά μικρά τμήματα. Για να πραγματοποιηθεί το "κατέβασμα" αυτού του αρχείου, θα πρέπει πρώτα ο χρήστης να κατεβάσει ένα πολύ μικρό αρχείο (το οποίο συνήθως είναι μόνο μερικά kb) με κατάληξη .torrent, το οποίο στην συνέχεια πρέπει να ανοιχθεί με την βοήθεια κάποιου torrent client. Το αρχείο αυτό περιέχει πληροφορίες σχετικά με το μέγεθος του αρχείου που θέλει να κατεβάσει ο χρήστης, το πλήθος τον τμημάτων που αποτελούν αυτό το αρχείο και τον tracker στον οποίο πρέπει να συνδεθεί ο χρήστης. Με τον όρο tracker εννοούμε έναν κεντρικό server, η αποκλειστική λειτουργία του οποίου είναι να ενημερώνει τον agent που χρησιμοποιεί ο χρήστης, σχετικά με το ποιοι κόμβοι έχουν εκείνη την στιγμή ολόκληρο ή κάποια τμήματα του επιθυμητού αρχείου και μπορούν να το διαμοιράσουν .

Σε αντίθεση με τα άλλα δίκτυα P2P, που έχουν μια δενδροειδή δομή και όπου η ανταλλαγή των αρχείων γίνεται κυρίως ανάμεσα σε δύο υπολογιστές, σε ένα δίκτυο που χρησιμοποιεί το BitTorrent πρωτόκολλο, οι χρήστες που διαμοιράζονται ένα αρχείο σχηματίζουν τα λεγόμενα "κοπάδια" (swarm). Κάθε χρήστης που έχει ολόκληρο το αρχείο ονομάζεται seeder, ενώ οι χρήστες που δεν έχουν ακόμη ολόκληρο το αρχείο ονομάζονται peers ή, αλλιώς, leechers. Όταν κάποιος leecher αποκτήσει ολόκληρο το αρχείο, μετατρέπεται σε seeder και μπορεί να συνεχίσει να μεταβιβάζει τμήματα του αρχείου στα υπόλοιπα μέλη του swarm. Με τον τρόπο

αυτό μεγιστοποιείται η πιθανότητα των πιθανών leecher να αποκτήσουν τελικά το αρχείο. Ένα άλλο πλεονέκτημα αυτού του πρωτοκόλλου είναι πως ακόμα και οι χρήστες οι οποίοι δεν έχουν ακόμα ολόκληρο το αρχείο, μπορούν να δίνουν σε άλλους χρήστες τα τμήματα του αρχείου τα οποία έχουν ήδη κατεβάσει, αυξάνοντας με τον τρόπο αυτό την ταχύτητα με την οποία διαμοιράζεται τελικά ένα αρχείο. Όταν το αρχείο αποκτηθεί από μερικούς peers, ο αρχικός seeder, δηλαδή αυτός ο οποίος δημιούργησε το .torrent αρχείο και που ήταν ο πρώτος που άρχισε το "ανέβασμα" του αρχείου μπορεί να σταματήσει το ανέβασμα του συγκεκριμένου αρχείου και να αρχίσει να ανεβάζει κάποιο άλλο. Σε αυτό το πρωτόκολλο, δηλαδή, δεν υπάρχει μεγάλη εξάρτηση από τον αρχικό seeder, κάτι που διασφαλίζει, εκτός των άλλων, και την μεγάλη βιωσιμότητα των αρχείων.

Με την εξέλιξη αυτού του πρωτοκόλλου δημιουργήθηκαν και περισσότερα αποκεντρωτικά δίκτυα, τα οποία δεν βασίζονται σε κάποιον tracker. Σε αυτά τα δίκτυα η εύρεση των κόμβων που μοιράζονται κάποιο αρχείο και η σύνδεση με αυτά γίνεται μέσω της τεχνολογίας DHT (Distributed Hash Tables).

Τα δίκτυα που βασίζονται στο πρωτόκολλο BitTorrent είναι ιδιαίτερα δημοφιλή και διαδεδομένα. Χαρακτηριστικό είναι πως σύμφωνα με μετρήσεις, το 2002 τα δίκτυα αυτά ευθύνονταν για το 35% περίπου του συνολικού όγκου δεδομένων που μεταφέρονταν μέσω ίντερνετ, ποσοστό που μέχρι σήμερα έχει αυξηθεί δραματικά.

3.2.2 Γενιές P2P δικτύων

Τα P2P δίκτυα, από την στιγμή της εμφάνισης τους μέχρι και σήμερα, δεν έπαψαν να εξελίσσονται, έτσι ώστε να προσαρμόζονται στις ολοένα και αυξανόμενες ανάγκες των χρηστών. Μπορούμε να κατηγοριοποιήσουμε τα P2P δίκτυα που έχουν εμφανιστεί μέχρι σήμερα, σε τέσσερις γενιές.

Πρώτη γενιά

Με τον όρο P2P δίκτυα πρώτης γενιάς αναφερόμαστε σε δίκτυα P2P που χαρακτηρίζονται συγκεντρωτικά, ακριβώς επειδή χρησιμοποιούν κάποιον κεντρικό εξυπηρετητή. Τέτοια δίκτυα είναι το Napster, όπως επίσης και κάποια μεταγενέστερα προγράμματα (clients) όπως είναι το DC++ και το Soulseek.

Δεύτερη γενιά

Τα νομικά προβλήματα που αντιμετώπισε το Napster, τα οποία οφείλονταν, ως ένα βαθμό, στην ύπαρξη του κεντρικού server, σε συνδυασμό με τους περιορισμούς των συγκεντρωτικών δικτύων, οδήγησαν στην δημιουργία των αποκεντρωτικών P2P δικτύων. Τέτοια δίκτυα, όπως το Gnutella, αλλά και το παρεμφερές KaZaa, ονομάζονται δίκτυα P2P δεύτερης γενιάς. Οι περισσότεροι χρήστες P2P δικτύων σήμερα χρησιμοποιούν δίκτυα δεύτερης γενιάς.

Χαρακτηριστικό είναι πως στα τέλη του 2007, το δίκτυο Gnutella ήταν το πλέον δημοφιλές δίκτυο διαμοιρασμού αρχείων, αφού ευθυνόταν για παραπάνω από το 40% των ανταλλαγών αρχείων στο διαδίκτυο.

Τρίτη γενιά

Όπως θα αναλύσουμε στην συνέχεια, από την πρώτη στιγμή της ανάπτυξης των P2P δικτύων, ανέκυψαν σοβαρά νομικά ζητήματα, όπως το κατά πόσο μπορούν οι χρήστες να ανταλλάσσουν αρχεία τα οποία προστατεύονται από τα πνευματικά δικαιώματα των δημιουργών τους, αλλά και το κατά πόσο, και με ποιον τρόπο, μπορεί να ελεγχθεί η κίνηση αυτών των αρχείων στα P2P δίκτυα από τις αρμόδιες αρχές.

Για να καταστεί δύσκολος ο εντοπισμός και ο έλεγχος των χρηστών και των αρχείων που αυτοί ανταλλάσσουν, δημιουργήθηκαν τα P2P δίκτυα τρίτης γενιάς. Τα δίκτυα αυτά είναι αποκεντρωτικά και εστιάζουν κυρίως στην ανωνυμία των χρηστών. Στα δίκτυα τρίτης γενιάς, οι κόμβοι που συμμετέχουν στο δίκτυο δεν περιέχουν χαρακτηριστικά που μπορούν να αποκαλύψουν την ταυτότητα τους, (όπως για παράδειγμα την ηλεκτρονική διεύθυνση) ή τα χαρακτηριστικά αυτά είναι κρυπτογραφημένα.

Ταυτόχρονα, γίνεται επαναδρομολόγηση της ροής των δεδομένων, ώστε να καταστεί αδύνατος ο εντοπισμός των χρηστών που συμμετέχουν σε μια ανταλλαγή αρχείων. Αναλυτικότερα, ένα αρχείο αντί να μεταφερθεί απ΄ ευθείας από τον κόμβο αποστολέα στον κόμβο παραλήπτη, μπορεί να μεταφερθεί μέσω άλλων κόμβων, καθιστώντας, με αυτόν τον τρόπο, τον εντοπισμό πολύ δύσκολο. Κάθε κόμβος μπορεί να λειτουργήσει και σαν επαναδρομολογητής, δηλαδή να δρομολογεί κάποιο αρχείο για λογαριασμό ενός άλλου κόμβου. Έτσι δεν είναι εύκολο κάποιος να ελέγξει αν ένας κόμβος μετέφερε παράνομα ένα αρχείο ή αν απλά λειτούργησε σαν επαναδρομολογητής για λογαριασμό κάποιου άλλου κόμβου, πράξη για την οποία είναι δύσκολο να ασκηθούν ποινικές διώξεις. Τα πιο γνωστά P2P δίκτυα τρίτης γενιάς είναι τα Freenet , I2P και MUTE.

Τέταρτη γενιά

Τα τελευταία χρόνια έχουν κάνει την εμφάνιση τους και τα λεγόμενα P2P δίκτυα τέταρτης γενιάς, τα οποία διαφοροποιούνται αρκετά σε σχέση με τα δίκτυα των προηγούμενων γενεών. Στα δίκτυα αυτά παρέχεται η δυνατότητα στους χρήστες να δέχονται συνεχή ροή δεδομένων (streaming) και όχι μεμονωμένα αρχεία. Η τεχνολογία αυτή χρησιμοποιείται κυρίως για την μετάδοση προγραμμάτων τηλεόρασης (P2PTV) και ραδιοφώνου, μέσω ίντερνετ. Με τον τρόπο αυτό οι χρήστες του δικτύου μπορούν να παρακολουθήσουν ταινίες, αθλητικά γεγονότα κ.ά., πολλές φορές σε ζωντανό σχεδόν χρόνο, χωρίς να χρειάζεται πρώτα να κατεβάσουν κάποιο ολόκληρο αρχείο.

Στο σημείο αυτό, αξίζει να αναφέρουμε, πως υπάρχουν πολλά δίκτυα τα οποία χρησιμοποιούν χαρακτηριστικά και από τα δίκτυα συγκέντρωσης αλλά και από αυτά της αποσυγκέντρωσης. Συγκεκριμένα, στα δίκτυα αυτά κάποιοι κόμβοι, οι οποίοι διαθέτουν περισσότερους πόρους από άλλους, λειτουργούν προσωρινά σαν “υπερκόμβοι” και είναι αυτοί οι οποίοι αναλαμβάνουν την ευθύνη για την προώθηση των αιτημάτων αναζήτησης σε ένα σύνολο κόμβων του δικτύου. Με τον τρόπο αυτό, αποφεύγονται χαρακτηριστικά συμφόρησης που είχαν παρατηρηθεί στα πρώτα αποκεντρωτικά δίκτυα, όπως π.χ στο Gnutella, ενώ ταυτόχρονα. λόγω του ότι κάποιος κόμβος παίρνει προσωρινά τον ρόλο του υπερκόμβου, είναι δυσχερής ο εντοπισμός

του (σε αντίθεση με τους κεντρικούς σερερ των συγκεντρωτικών δικτύων).

3.2.3 Άλλες χρήσεις

Πέρα από την ανταλλαγή αρχείων στο διαδίκτυο, τα P2P δίκτυα έχουν και πολλές άλλες εφαρμογές.

Κάποιες από τις δυνατότητες που παρέχουν τα δίκτυα αυτά, είναι:

- Η δυνατότητα μεταφοράς τηλεοπτικού και ραδιοφωνικού σήματος μέσω διαδικτύου με χρήση P2P δικτύου (P2PTV).

- Η δυνατότητα να επικοινωνούν άνθρωποι από όλο τον κόσμο, με χρήση εικόνας και ήχου, μέσω εφαρμογών που χρησιμοποιούν δίκτυο P2P. Ένα από τα ποιο γνωστά προγράμματα που παρέχει αυτή την δυνατότητα, είναι το Skype, το οποίο έχει δημιουργηθεί από τους προγραμματιστές του Gnutella.

- Δυνατότητα χρήσης για στρατιωτικούς σκοπούς.

- Δυνατότητα χρήσης για ιατρικούς σκοπούς.

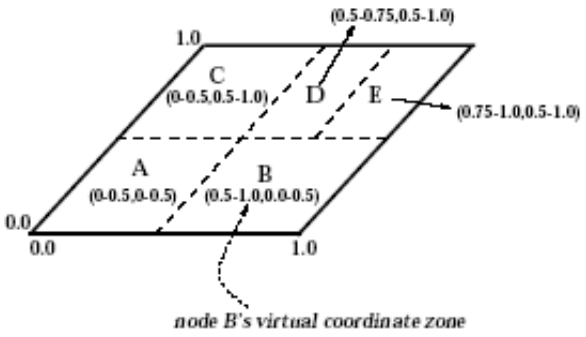
Χαρακτηριστικό παράδειγμα είναι η χρήση P2P δικτύου από την εφαρμογή Folding@Home. Πρόκειται για μια εφαρμογή, που αναπτύχθηκε από το Stanford University και ξεκίνησε την λειτουργία της τον Οκτώβριο του 2000. Η εφαρμογή αυτή σχεδιάστηκε με σκοπό να δημιουργεί προσομοιώσεις πρωτεϊνικών αναδιπλώσεων, ώστε να δώσει απαντήσεις στους ερευνητές, σχετικά με τους λόγους για τους οποίους η αναδίπλωση αυτή πολλές φορές δεν γίνεται σωστά, γεγονός στο οποίο οφείλονται πολλές ανίατες ασθένειες όπως Parkinson, Alzheimer, αλλά και κάποιες μορφές καρκίνου. Ακριβώς επειδή οι προσομοιώσεις αυτές απαιτούν τεράστια επεξεργαστική ισχύ, η εφαρμογή αυτή χρησιμοποιεί ένα P2P δίκτυο, έτσι ώστε οι υπολογιστές χρηστών από όλο τον κόσμο να προσφέρουν υπολογιστική ισχύ, δημιουργώντας, κατ΄ αυτόν τον τρόπο, έναν εικονικό υπερυπολογιστή. Σύμφωνα με το βιβλίο Guinness, αυτή τη στιγμή το Folding@Home αποτελεί την μεγαλύτερη συστοιχία υπολογιστών στον κόσμο.

3.3 Δομημένα δίκτυα

Το χαρακτηριστικό των δομημένων συστημάτων ομότιμων κόμβων είναι ότι στα δεδομένα αναθέτονται μοναδικά αναγνωριστικά τα οποία λέγονται κλειδιά. Με βάση τα κλειδιά γίνεται η αντιστοίχιση μεταξύ κόμβων και δεδομένων, η οποία δείχνει για ποια δεδομένα είναι υπεύθυνος ο κάθε κόμβος. Η ανάθεση κλειδιών σε κόμβους γίνεται με τη χρήση συναρτήσεων κατακερματισμού έτσι ώστε τα δεδομένα να διαμοιράζονται όσο το δυνατόν περισσότερο ομοιόμορφα. Με αυτόν τον τρόπο δομείται ένας γράφος στον οποίο οι κόμβοι είναι συνδεδεμένοι με καθορισμένο τρόπο. Τα πιο χαρακτηριστικά δομημένα συστήματα είναι το Content Addressable Network (CAN) και το Chord για τα οποία θα μιλήσουμε πιο κάτω.

3.3.1 CAN

Το Content Addressable Network (CAN), είναι δομημένο σαν ένα εικονικό σύστημα συντεταγμένων d διαστάσεων (d-torus). Στο Σχήμα 3.3, φαίνεται η δομή ενός CAN. Ο χώρος διαμερίζεται σε ζώνες, δυναμικά ανάμεσα στους κόμβους έτσι ώστε κάθε κόμβος να αναλαμβάνει μία ή περισσότερες ζώνες. Κάθε κόμβος κρατάει τις συντεταγμένες της ζώνης του και τις συντεταγμένες των γειτονικών ζωνών. Επίσης, κάθε κόμβος κρατάει ένα ζεύγος κλειδιού-δεδομένου (K,V). Η αντιστοίχηση του ζεύγους (K,V) σε κάποιο κόμβο γίνεται με τη χρησιμοποίηση μιας συνάρτησης κατακερματισμού η οποία αντιστοιχίζει το κλειδί K σε ένα σημείο P στον εικονικό χώρο συντεταγμένων. Το (K,V) αποθηκεύεται σε εκείνο τον κόμβο που είναι υπεύθυνος για τη ζώνη στην οποία ανήκει το P. Η ανάκτηση ενός δεδομένου με κλειδί K, γίνεται χρησιμοποιώντας την ίδια συνάρτηση κατακερματισμού πάνω στο K, οπότε αυτό αντιστοιχίζεται σε ένα σημείο P. Στη συνέχεια, από τον κόμβο στον οποίο ανήκει το P, ανακτούμε το δεδομένο V. Αν το σημείο P δεν ανήκει στον κόμβο που έκανε την αίτηση για το (K,V) και δεν ανήκει ούτε στους κόμβους γειτονικών ζωνών, τότε η αίτηση δρομολογείται στην κοντινότερη γειτονική ζώνη του P.



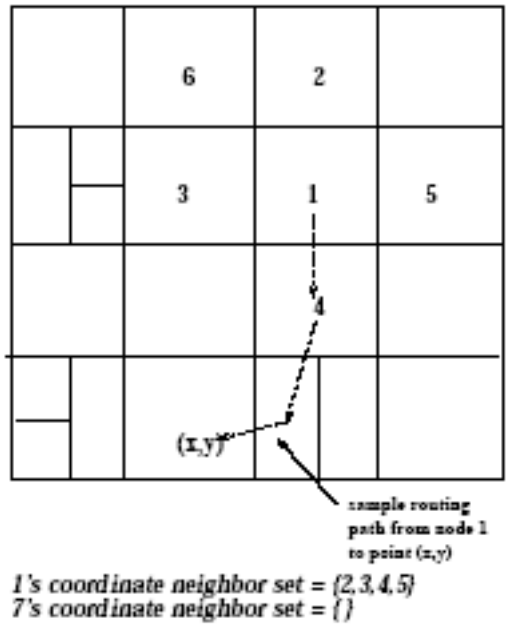
Σχήμα 3.3: Δομή του CAN

Αναζήτηση - Δρομολόγηση στο CAN

Για να δρομολογήσουμε ένα μήνυμα αναζήτησης από μια πηγή σε ένα συγκεκριμένο προορισμό, ακολουθούμε το μονοπάτι από τη πηγή στο προορισμό με βάση τις Καρτεσιανές συντεταγμένες του d-διάστατου χώρου στον οποίο έχει δομηθεί το CAN.

Κάθε κόμβος κρατάει τις συντεταγμένες της ζώνης την οποία έχει αναλάβει, καθώς και τις συντεταγμένες των γειτονικών ζωνών. Ένας greedy αλγόριθμος αναλαμβάνει να προωθήσει το μήνυμα στην κοντινότερη στον προορισμό γειτονική ζώνη, με βάση αυτές τις συντεταγμένες. Στο Σχήμα 3.4 φαίνεται ένα παράδειγμα δρομολόγησης. Έστω ότι ο κόμβος που έχει αναλάβει τη ζώνη 1, ψάχνει για το δεδομένο με συντεταγμένες (x, y). Ο κόμβος 1 θα προωθήσει το μήνυμα, σε μια από τις γειτονικές του ζώνες η οποία απέχει τη μικρότερη απόσταση από το (x, y). Η ζώνη αυτή είναι η 4, άρα το μήνυμα αναζήτησης προωθείται προς τον κόμβο που

έχει αναλάβει τη συγκεκριμένη ζώνη. Από τη ζώνη 4, το μήνυμα αναζήτησης προωθείται στον γειτονική ζώνη του (x, y), όπως φαίνεται στο σχήμα, αφού αυτή πλέον απέχει λιγότερο από το (x, y). Τέλος, το μήνυμα προωθείται στον κόμβο που ανήκει το δεδομένο (x, y) οπότε και τελειώνει η αναζήτηση.



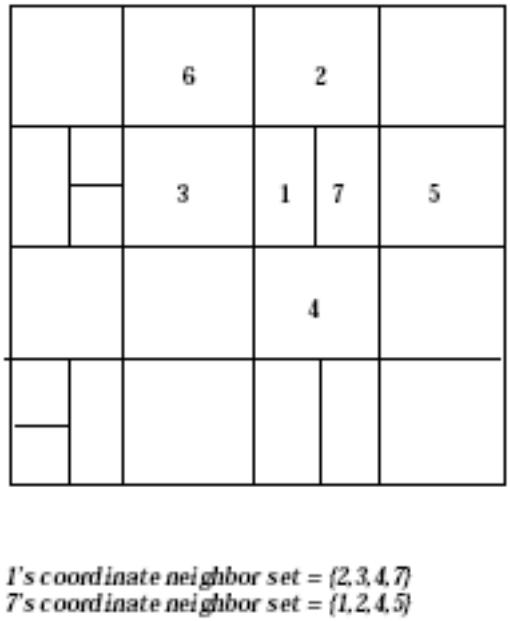
Σχήμα 3.4: Παράδειγμα αναζήτησης. Ο 1 αναζητάει το (x, y).

Έτσι, για ένα χώρο διάστασης d, χωρισμένο σε n ίσες ζώνες (κάθε ζώνη αντιστοιχεί σε ένα και μόνο ένα κόμβο, άρα συνολικά έχουμε n κόμβους), κάθε κόμβος έχει $2d$ γείτονες και το μέσο μήκος ενός μονοπατιού είναι $(d/4)(n_1/d)$. Επίσης, για ένα χώρο d διαστάσεων, αυξάνοντας τον αριθμό των κόμβων (άρα και τον αριθμό των ζωνών) το μέσο μήκος μονοπατιού αυξάνεται με $O(n_1/d)$, ενώ ο όγκος της πληροφορίας που χρειάζεται να κρατάει ο κάθε κόμβος μένει ίδιος ($2d$). Άρα, η απόδοση της αναζήτησης είναι $O(d * n_1/d)$, αφού ο αριθμός των μηνυμάτων που θα χρειαστούμε, είναι $O((d/4)(n_1/d)) = O(d(n_1/d))$.

Κατασκευή του CAN - Εισαγωγή Κόμβου

Το CAN (πιο συγκεκριμένα ο εικονικός χώρος συντεταγμένων) κατασκευάζεται με την άφιξη κάθε καινούριου κόμβου. Έστω λοιπόν, ότι ένας κόμβος θέλει να ενταχθεί στο CAN. Πρώτα επιλέγει ένα τυχαίο σημείο P του χώρου συντεταγμένων και στέλνει ένα μήνυμα συνένωσης στον κόμβο ο οποίος αντιστοιχεί στη ζώνη που περιβάλλει το P. Όταν το μήνυμα φτάσει στον κόμβο που κατέχει το P, αυτός ο κόμβος διαιρεί τη ζώνη που του αντιστοιχεί σε δύο ίσες ζώνες. Αν για παράδειγμα, έχουμε ένα χώρο 2 διαστάσεων, τότε ο κόμβος θα χωρίσει τη ζώνη του πρώτα κατά την πρώτη διάσταση (π.χ. κατά την Q) και μετά κατά την δεύτερη διάσταση (π.χ. κατά την U). Αν ο ίδιος κόμβος χρειαστεί να ξαναχωρίσει τη ζώνη του, τότε θα τη χωρίσει πάλι κατά την διάσταση Q και η όλη διαδικασία επαναλαμβάνεται χωρίζοντας τη ζώνη κατά Q, U εναλλάξ. Στη συνέχεια, τα ζεύγη (K, V) που ανήκουν στη

ζώνη που δημιουργήθηκε και που θα καταλάβει ο καινούριος κόμβος, μεταφέρονται σε αυτόν τον κόμβο και οι γείτονες του καινούριου και του παλιού κόμβου ενημερώνονται για την άφιξη του καινούριου κόμβου και την δημιουργία καινούριας ζώνης. Τέλος, ενημερώνεται και ο καινούριος κόμβος για τις γειτονικές ζώνες του. Στο Σχήμα 3.5, φαίνεται η εισαγωγή του κόμβου 7. Πριν την εισαγωγή του 7 το CAN ήταν όπως στο Σχήμα 3.4. Ο κόμβος 7 βρίσκει έναν κόμβο, τυχαία, ο οποίος υπάρχει ήδη στο σύστημα. Στην προκειμένη περίπτωση, βρίσκει τον κόμβο 1. Ο κόμβος 1 χωρίζει την ζώνη του στη μέση, κατά την διάσταση U. Την μισή ζώνη αναλαμβάνει ο 1 και την υπόλοιπη ο 7. Επίσης, ο 7 παίρνει και τα ζεύγη (K, V) που του αναλογούν και ενημερώνεται για τους γείτονές του. Τέλος, ενημερώνονται και όλοι οι γείτονες (οι 2, 3, 4, 5 και 6) για την εισαγωγή του 7.



Σχήμα 3.5: Παράδειγμα εισαγωγής κόμβου. Ο 7 εισάγεται στο σύστημα.

Αποχώρηση Κόμβου, Αποτυχία Κόμβου και Συντήρηση στο CAN

Όταν ένας κόμβος φεύγει εθελοντικά, τότε παραδίδει τη ζώνη του και τα ζεύγη (K,V) σε κάποιον γείτονά του (έτσι ώστε να διατηρούνται έγκυρες ζώνες, όπως περιγράφηκαν πιο πάνω, κατά την κατασκευή τους). Το πρόβλημα υπάρχει όταν ένας κόμβος αποτυγχάνει.

Σε αυτή την περίπτωση, οι μη-προσβάσιμοι κόμβοι θέτουν σε λειτουργία ένα αλγόριθμο άμεσης ανακατάληψης ο οποίος αντιστοιχίζει τη ζώνη του κόμβου που απέτυχε, σε κάποιον γείτονα. Μια αποτυχία ανιχνεύεται με περιοδικά μηνύματα που στέλνουν οι κόμβοι μεταξύ τους.

3.3.2 Chord

Το Chord αποτελεί ένα πρωτόκολλο ομότιμου δικτύου που στοχεύει στην γρήγορη και αποτελεσματική αναζήτηση μεταξύ των κόμβων. Το Chord οργανώνει τους κόμβους του δικτύου γύρω από ένα νοητό δακτύλιο, βάση ενός μοναδικού ID που αντιστοιχεί σε κάθε κόμβο. Η γρήγορη αναζήτηση επιτυγχάνεται με τη χρήση αναφορών σε πολλαπλούς κόμβους σταθερής θέσης (Finger Tables), που αποτελούν σημεία με αποστάσεις τις δυνάμεις του δύο, έτσι ώστε να λογαριθμείται ο χρόνος που χρειάζεται να ταξιδέψει ένα αίτημα γύρω από τον κύκλο. Στο Κεφάλαιο 4 θα εξετάσουμε αναλυτικά το πρωτόκολλο Chord.

3.3.3 Συμπεράσματα πάνω στα Δομημένα Συστήματα

Στον Πίνακα 3.1 φαίνονται, συγκεντρωτικά, οι διαφορές μεταξύ CAN και Chord:

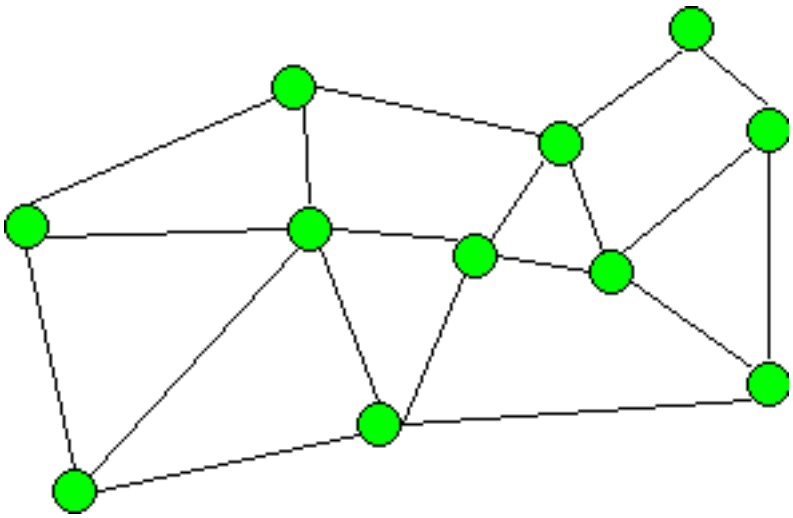
Σύστημα	CAN	Chord
Αρχιτεκτονική	Πολυδιάστατος χώρος Καρτεσιανών συντεταγμένων	Δακτύλιος αναγνωριστικών Κόμβων
Πρωτόκολλο Αναζήτησης	Συναρτήσεις κατακερματισμού για αντιστοίχιση ζευγών (K,V) σε σημείο P του χώρου συντεταγμένων	Συναρτήσεις κατακερματισμού για αντιστοίχιση κλειδιού και id κόμβου
Μήκος μονοπατιού	$O(dN_{1/d})$	$O(\log N)$
Καταστάσεις γειτόνων (εισαγωγή/διαγραφή)	2δ	λογ2N
Καταστάσεις γειτόνων (αναζήτηση)	$2d$	$\log N$

Πίνακας 3.1: Διαφορές CAN και Chord

3.4 Αδόμητα δίκτυα

Τα δομημένα συστήματα που εξετάσαμε πιο πάνω, έχουν το πλεονέκτημα ότι οι κόμβοι τους ενώνονται με συγκεκριμένο τρόπο οπότε δημιουργείται ένας γράφος. Ο τρόπος με τον οποίο συνδέονται έχει επιλεχθεί έτσι, ώστε η αναζήτηση, η εισαγωγή και η αποχώρηση ενός κόμβου, να γίνονται αποδοτικότερα. Ωστόσο, είδαμε ότι κατά την εισαγωγή ή την αποτυχία ενός κόμβου, χρειάζεται να ενημερωθεί ένας αρκετά μεγάλος αριθμός από κόμβους προκειμένου το σύστημα να λειτουργήσει σωστά. Το πρόβλημα αυτό, λύνουν τα αδόμητα συστήματα, στα οποία η εισαγωγή και αποχώρηση η αποτυχία γίνεται σε ένα βήμα, χωρίς να επηρεάζονται

οι υπόλοιποι κόμβοι. Στο Σχήμα 3.6 φαίνεται ένα παράδειγμα αδόμητου συστήματος. Παρατηρούμε ότι οι κόμβοι είναι συνδεδεμένοι τυχαία, χωρίς να υπακούν σε κάποια συγκεκριμένη δομή. Στα αδόμητα συστήματα οι κόμβοι συνδέονται τυχαία μεταξύ τους χωρίς να είναι απαραίτητο να διατηρηθεί κάποια δομή. Έτσι, λοιπόν, δεν μπορούμε να εκμεταλλευτούμε τις ιδιότητες της δομής ή τον τρόπο σύνδεσης των κόμβων. Αντίθετα, η προσπάθεια για αποδοτική αναζήτηση στηρίζεται στην εύρεση κατάλληλων αλγορίθμων. Έτσι στα αδόμητα συστήματα επικεντρωνόμαστε στους μηχανισμούς αναζήτησης. Μερικούς από αυτούς τους μηχανισμούς εξετάζουμε πιο κάτω.



Σχήμα 3.6: Παράδειγμα αδόμητου συστήματος. Οι κύκλοι συμβολίζουν τους κόμβους.

3.4.1 Τυφλές μέθοδοι αναζήτησης

Στις τυφλές μεθόδους αναζήτησης [30], τα μηνύματα αναζήτησης προωθούνται στους κόμβους με "τυφλό", μη ευριστικό τρόπο, αν και μόνο αν οι απαντήσεις που έχουν δοθεί δεν ικανοποιούν τη συνθήκη τερματισμού. Η συνθήκη τερματισμού μπορεί να είναι ένας συγκεκριμένος αριθμός από απαντήσεις που επιθυμούμε. Έτσι, όταν αυτός ο αριθμός ικανοποιηθεί, η αναζήτηση σταματάει. Επίσης, συνθήκη τερματισμού μπορεί να είναι και ο Time-To-Live (TTL). Ο TTL είναι ένας μετρητής ο οποίος εμπεριέχεται στο μήνυμα αναζήτησης και μετράει ανάποδα κάθε φορά που το μήνυμα αναζήτησης περνάει από κάποιον κόμβο. Όταν ο μετρητής φτάσει στο 0, τότε η αναζήτηση σταματάει. Οι αλγόριθμοι που υλοποιούν τυφλές μεθόδους αναζήτησης είναι:

- Gnutella: Το σύστημα της Gnutella χρησιμοποιεί πλημμύρα (ή αλλιώς αναζήτηση κατά πλάτος (BFS) για την προώθηση των μηνυμάτων αναζήτησης. Η πλημμύρα συνεχίζεται είτε μέχρι να δοθεί ικανός αριθμός από απαντήσεις είτε μέχρι να λήξει η τιμή του TTL. Ο αλγόριθμος αυτό είναι απλός αλλά παρουσιάζει μεγάλο χρόνο απόκρισης για μεγάλο αριθμό κόμβων.

- Τροποποιημένου-BFS: Ο αλγόριθμος αυτός είναι ίδιος με τον αλγόριθμο της πλημμύρας με τη διαφορά ότι εδώ ο κόμβος που προωθεί το μήνυμα αναζήτησης, το προωθεί τυχαία σε μερικούς και όχι σε όλους τους κόμβους. Ο αλγόριθμος αυτός βελτιώνει το μειονέκτημα του μεγάλου χρόνου απόκρισης, που παρουσιάζει ο αλγόριθμος της πλημμύρας, το οποίο όμως για πολύ μεγάλο αριθμό κόμβων, παραμένει.

- Επαναληπτικής εκβάθυνσης: Ο αλγόριθμος αυτός είναι ένας συνδυασμός των αλγορίθμων DFS και BFS. Πρώτα, εφαρμόζεται ο αλγόριθμος BFS σε βάθος ένα. Αν ο αριθμός των απαντήσεων δεν είναι ικανοποιητικός, εφαρμόζεται BFS σε βάθος 2 κ.ο.κ μέχρι να ικανοποιηθεί η συνθήκη τερματισμού. Ο αλγόριθμος αυτός δουλεύει καλά αν η συνθήκη τερματισμού έχει δοθεί από τον χρήστη οπότε είναι πιθανό το ερώτημα να ικανοποιηθεί σε μικρό βάθος. Αυτό συμβαίνει γιατί οι χρήστες, συνήθως, αναζητούν κάτι συγκεκριμένο, οπότε ο αριθμός των απαντήσεων που επιθυμούν είναι μικρός τις περισσότερες φορές. Σε διαφορετική περίπτωση, ο αλγόριθμος προκαλεί μεγαλύτερο φόρτο από ότι η πλημμύρα.

- Random Walks: Στον αλγόριθμο αυτό, ο κόμβος εκδίδει λ αντίγραφα του ίδιου μηνύματος αναζήτησης και τα προωθεί σε λ τυχαίους γείτονες. Κάθε ένα από αυτά τα αντίγραφα ακολουθεί το δικό του μονοπάτι αφού οι ενδιάμεσοι κόμβοι προωθούν το κάθε αντίγραφο σε ένα μόνο τυχαίο γείτονά τους. Κάθε αντίγραφο αυτού του μηνύματος αναζήτησης τερματίζει αν οι απαντήσεις είναι αρκετές ή αν το λήξει TTL. Το πλεονέκτημα αυτού του αλγορίθμου είναι ότι παράγει ένα μικρό αριθμό από μηνύματα ($\lambda \times TTL$ το πολύ) εν αντιθέσει με την πλημμύρα. Το μειονέκτημά του είναι ότι η απόδοση του δεν είναι σταθερή μιας και εξαρτάται πολύ από την τοπολογία του δικτύου και την τυχαία επιλογή των γειτόνων.

- GUESS: Στον αλγόριθμο αυτό υπάρχουν υπερκόμβοι οι οποίοι συνδέονται με άλλους υπερκόμβους και καθένας από αυτούς με ένα σύνολο από κόμβους-φύλλα στους οποίους λειτουργεί ως server. Η αναζήτηση γίνεται ρωτώντας επαναληπτικά τους υπόλοιπους υπερκόμβους (όχι απαραίτητα γειτονικούς) οι οποίοι ρωτάνε όλους τους κόμβους-φύλλα.

- Gnutella2: Ο αλγόριθμος αυτός είναι παρόμοιος με τον GUESS με τη διαφορά ότι εδώ οι υπερκόμβοι προωθούν το μήνυμα αναζήτησης πρώτα στους δικούς τους κόμβους-φύλλα και στη συνέχεια σε γειτονικούς υπερκόμβους.

3.4.2 Μέθοδοι αναζήτησης με πληροφορία

Στις μεθόδους αυτές χρησιμοποιούνται αλγόριθμοι οι οποίοι προωθούν τα μηνύματα στους κόμβους βασισμένοι σε κάποια πληροφορία. Οι μέθοδοι αυτές είναι:

- Έξυπνος-BFS: Παραλλαγή του τροποποιημένου BFS. Οι κόμβοι αποθηκεύουν δυάδες από μηνύματα αναζήτησησ-γείτονες (για αιτήσεις που έχουν απαντηθεί πρόσφατα από τους γείτονές τους) έτσι ώστε να τους κατατάξουν. Πρώτα, ο κόμβος αναγνωρίζει όλα τα μηνύματα που είναι παρόμοια με το μήνυμα αναζήτησης που θέλουμε να προωθήσουμε,

σύμφωνα με κάποια μετρική. Στη συνέχεια, επιλέγει να το προωθήσει στους γείτονες οι οποίοι έχουν επιστρέψει τα περισσότερα αποτελέσματα για αυτά τα μηνύματα. Σε σύγκριση με τον τροποποιημένο-BFS ο έξυπνοσ-BFS παράγει περισσότερα μηνύματα αναζήτησης, αλλά πετυχαίνει μεγάλη ακρίβεια στις απαντήσεις για ένα συγκεκριμένο μήνυμα αναζήτησης.

- APS: Στο APS κάθε κόμβος κρατάει ένα τοπικό πίνακα με γραμμές τα αντικείμενα, για τα οποία έχει γίνει αίτηση από τον κόμβο, και στήλες τους γείτονες. Οι τιμές που αποθηκεύονται στον πίνακα αντιστοιχούν στη πιθανότητα που έχει ένας γείτονας να επιλεχθεί (ως επόμενος κόμβος προώθησης του μηνύματος αναζήτησης) για το συγκεκριμένο αντικείμενο. Η αναζήτηση γίνεται όπως και στα Random Walks με τη διαφορά ότι η προώθηση εδώ γίνεται με βάση την πιθανότητα των γειτόνων. Οι πιθανότητες σε ένα τοπικό πίνακα μεταβάλλονται ανάλογα με τις απαντήσεις των μηνυμάτων αναζήτησης. Όταν ένα μήνυμα απαντάται, τότε η πιθανότητα του γείτονα (του κόμβου που έκανε την αίτηση) που είχε αρχικά επιλεγεί αυξάνεται, στην αντίθετη περίπτωση μειώνεται.

- Distributed Resource Location Protocol (DRLP): Εδώ, οι κόμβοι προωθούν τα μηνύματα αναζήτησης σε ένα από τους γείτονες τους με βάση κάποια πιθανότητα. Όταν ένα αντικείμενο βρεθεί, τότε ο κόμβος που έκανε την αίτηση αποθηκεύει την τοποθεσία του αντικειμένου. Σε επόμενες αιτήσεις για το συγκεκριμένο αντικείμενο, ο κόμβος αυτός θα επικοινωνήσει άμεσα με τον κόμβο που έχει το αντικείμενο, αν η τοποθεσία του αντικειμένου έχει αποθηκευτεί.

- Τοπικά indices: Κάθε κόμβος αποθηκεύει indices για αρχεία που βρίσκονται σε μακρινούς κόμβους, γύρω από μια συγκεκριμένη ακτίνα, και έτσι μπορεί να απαντάει σε μηνύματα αναζήτησης που προορίζονται για αυτούς. Η αναζήτηση γίνεται όπως στο BFS αλλά προσβάσιμοι είναι μόνο οι κόμβοι που ανήκουν μέσα στην ακτίνα που προαναφέρθηκε. Το μειονέκτημα αυτής της μεθόδου είναι ο μεγάλος αριθμός μηνυμάτων που χρειάζεται για την διατήρηση των indices (συγκρίσιμος με αυτόν της πλημμύρας). Από την άλλη, η μέθοδος των τοπικών ινδιςες προσφέρει μεγάλη ακρίβεια. Στη συνέχεια, περιγράφεται εκτενέστερα μια μορφή indices τα οποία λέγονται routing indices.

Routing indices

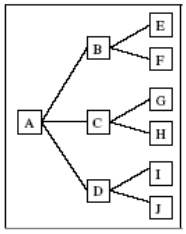
Τα Routing Indices (RIs) είναι ένας μηχανισμός ο οποίος διατηρεί μία δομή δεδομένων σε κάθε κόμβο. Η δομή αυτή θα πρέπει να είναι μικρή σε μέγεθος, άρα αντί για δομές που δείχνουν την ακριβή τοποθεσία του αντικειμένου (κάτι που θα έπιανε αρκετό χώρο σε κάθε κόμβο), χρησιμοποιούνται δομές που δείχνουν προς την κατεύθυνση (στο μονοπάτι) του αντικειμένου. Όταν η δομή αυτή δέχεται ένα μήνυμα αναζήτησης επιστρέφει μια λίστα των γειτόνων ταξινομημένη σε σχέση με την goodness για αυτό το μήνυμα αναζήτησης. Η goodness είναι μια μετρική η οποία απεικονίζει τον αριθμό των αντικειμένων των κοντινών κόμβων που απαντούν στο συγκεκριμένο μήνυμα. Με βάση τη μετρική αυτή, κι αν οι απαντήσεις που έχουν δοθεί στο μήνυμα από τον κόμβο στον οποίο βρισκόμαστε, δεν είναι αρκετές (με βάση

κάποια συνθήκη τερματισμού που μεταδίδεται μαζί με το μήνυμα αναζήτησης), το μήνυμα προχωράει στον κόμβο με το μεγαλύτερο goodness για περισσότερες απαντήσεις. Στη συνέχεια θα περιγράψουμε τρεις τύπους RIs, το Compound RI (CRI), το Exponential RI (ERI) και το Hop-Count RI (HRI).

CRI

Το Compound RI (CRI) είναι ένας πίνακας ο οποίος, σε κάθε κόμβο, κρατάει (α) τον αριθμό των αντικειμένων σε κάθε μονοπάτι (δηλαδή μονοπάτια που προκύπτουν από τον κόμβο που βρισκόμαστε) και (β) τον αριθμό των αντικειμένων για κάθε θέμα ενδιαφέροντος. Τα θέματα ενδιαφέροντος είναι διαφορετικές θεματικές περιοχές στις οποίες μπορούν να καταταχθούν τα αντικείμενα. Για παράδειγμα, ένα θέμα ενδιαφέροντος θα μπορούσε να είναι φιλολογικά κείμενα ενώ ένα άλλο θα μπορούσε να είναι τεχνολογικά κείμενα. Επίσης, κρατάει και ένα τοπικό index με τον συνολικό αριθμό των αντικειμένων που έχει ο κόμβος καθώς και τον αριθμό των αντικειμένων του κόμβου ανά θέμα ενδιαφέροντος. Οι γραμμές του πίνακα αναπαριστούν τους γείτονες του κόμβου και οι στήλες τα θέματα ενδιαφέροντος. Άρα, η κάθε θέση του πίνακα περιέχει τον αριθμό των αντικειμένων ανά θέμα ενδιαφέροντος που μπορούν να προσεγγιστούν μέσω του γείτονα της συγκεκριμένης γραμμής. Τα μηνύματα αναζήτησης για ένα αντικείμενο, προωθούνται στους κόμβους (αν βέβαια η συνθήκη τερματισμού δεν έχει εκπληρωθεί) με βάση μια μετρική η οποία λέγεται goodness.

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 3.7: Παράδειγμα CRI

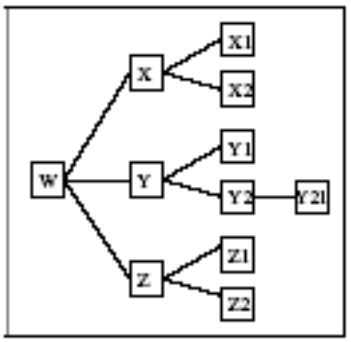
Αν έχουμε το μήνυμα αναζήτησης si, τότε στο CRI η goodness υπολογίζεται με βάση τον τύπο $NumberofDocuments \times \Pi_i \; CRI(si)/NumberofDocuments$ αν υποθέσουμε ότι τα αντικείμενα μπορεί να ανήκουν σε περισσότερα από ένα θέματα ενδιαφέροντος και ότι τα θέματα ενδιαφέροντος είναι ανεξάρτητα μεταξύ τους. $CRI(si)$ είναι τα θέματα ενδιαφέροντος για το μήνυμα si και $NumberofDocuments$ είναι ο αριθμός των αντικειμένων σε κάθε θέμα. Στο Σχήμα 3.7 φαίνεται ένα παράδειγμα CRI πίνακα για τον κόμβο A. Παρατηρούμε ότι σε κάθε γραμμή του πίνακα φαίνεται ο αριθμός των αντικειμένων ανά γείτονα. Στην πρώτη στήλη φαίνεται ο συνολικός αριθμός των αντικειμένων, ενώ στις υπόλοιπες φαίνεται ο αριθμός των αντικειμένων ανά θεματική περιοχή.

HRI

Ο κύριος περιορισμός του CRI είναι ότι δεν λαμβάνει υπόψη τον αριθμό των κόμβων (ή αλλιώς hops) που απαιτούνται για να βρεθεί ένα αντικείμενο. Την παράμετρο αυτή τη

λαμβάνει υπόψη το Hop-Count RI (HRI) το οποίο αποθηκεύει ότι και το CRI, αλλά για κάθε hop μέχρι ένα μέγιστο αριθμό από hops, που καλείται ορίζοντας του HRI. Στο Σχήμα 3.8, φαίνεται ένα παράδειγμα HRI. Αν υποθέσουμε ότι το δίκτυο είναι κανονικό δέντρο με βαθμό F τότε η goodness για κάποιον γείτονα $Neighbor_i$ και για κάποιο μήνυμα αναζήτησης Q δίνεται από τον τύπο: $goodness\left(Neighbor_i Q\right) = \sum_{j=0...h}\left(goodness\left(N_i\left[j\right], Q\right)/F_j\right)$ όπου h ο ορίζοντας του HRI, $goodness()$ η goodness του CRI και $N_i[j]$ είναι η γραμμή του HRI για j ηοπς διαμέσου του γείτονα $Neighbor_i$. Λαμβάνοντας υπόψη την καινούρια goodness, το HRI κάθε κόμβου ανανεώνεται και διατηρείται όπως και στο CRI, με τη διαφορά ότι ένας κόμβος που θέλει να ειδοποιήσει τους γείτονές του για μια αλλαγή στη βάση δεδομένων του, αφού υπολογίσει το νέο HRI του, μετατοπίζει τις στήλες του προς τα δεξιά έτσι ώστε οι στήλες για το hop 1 να γίνουν στήλες του hop 2 κ.ο.κ.

	1 Hop					2 Hops				
Node	#	DB	N	T	L	#	DB	N	T	L
X	60	13	2	5	10	20	10	10	4	17
Y	30	0	3	15	12	50	31	0	15	20
Z	5	2	0	3	3	70	10	40	20	50



Σχήμα 3.8: Παράδειγμα HRI για 2 hops

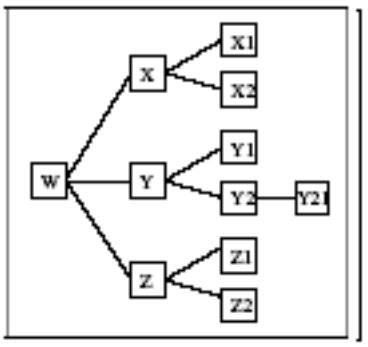
ERI

Το HRI λαμβάνει υπόψη και τον αριθμό των hops, όπως είδαμε. Αυτό έχει αποτέλεσμα το κόστος μετάδοσης και ο χώρος αποθήκευσης να είναι μεγαλύτερα από ότι στο CRI. Αυτά τα προβλήματα αντιμετωπίζονται από το ERI, με κόστος, όμως, την απώλεια σε ακρίβεια. Αν υποθέσουμε ότι το δίκτυο είναι κανονικό δέντρο με ύψος th και βαθμό F, τότε το ERI ενός κόμβου N αποθηκεύει σε κάθε γραμμή μια τιμή η οποία υπολογίζεται με βάση τον τύπο $goodness\left(Neighbor_i T\right) = \sum_{j=0...h}\left(goodness\left(N_i\left[j\right], T\right)/F_{j-1}\right)$, όπου $goodness()$ η goodness του CRI και $N[j]$ το άθροισμα του τοπικού index του γείτονα j του κόμβου N. Στο ERI, η προώθηση ενός μηνύματος αναζήτησης δεν γίνεται με κάποια goodness, αλλά με βάση την ίδια την τιμή που αποθηκεύεται σε κάθε γραμμή. Στο Σχήμα 3.9, φαίνεται ένα παράδειγμα ERI. Παρατηρούμε ότι σε κάθε γραμμή του πίνακα και για κάθε θέμα ενδιαφέροντος αποθηκεύουμε την τιμή που βρίσκουμε με τον πιο πάνω τρόπο. Με βάση αυτή την τιμή, γίνεται η προώθηση ενός μηνύματος αναζήτησης. Για να ανανεώσουμε τα ERI, ο κόμβος που χρειάζεται να στείλει μια ανανέωση σε ένα γείτονα, προσθέτει όλες τις γραμμές (εκτός από τη γραμμή του γείτονα στην οποία θα σταλεί η ανανέωση), πολλαπλασιάζει το αποτέλεσμα με $1/F$ και προσθέτει τη goodness του τοπικού του index. Στη συνέχεια η διαδικασία συνεχίζεται όπως στο CRI. Πρέπει να τονισθεί, ότι τα ανανεωμένα ERIs μεταδίδονται μόνο αν η καινούρια

και η παλιά τιμή έχουν αρκετή διαφορά μεταξύ τους (συγκεκριμένο όριο).

Στον Πίνακα 3.2, φαίνονται οι διαφορές των τριών RIs που εξετάσαμε. Παρατηρούμε ότι η γραμμή της γοοδνεσς για το ERI είναι κενή γιατί για την προώθηση ενός μηνύματος αναζήτησης στο ERI, δεν υπολογίζουμε κάποια goodness, αλλά χρησιμοποιούμε την τιμή που αποθηκεύουμε σε κάθε γραμμή του ERI.

Path	#	DB	N	T	L
X	66.67	16.33	5.33	6.33	15.67
Y	46.67	10.33	3.00	20.00	18.67
Z	28.33	5.33	13.33	9.67	19.67



Σχήμα 3.9: Παράδειγμα ERI

RI	CRI	HRI	ERI
Τιμή που αποθηκεύεται	Αριθμός αντικειμένων στο μονοπάτι	Αριθμός αντικειμένων ανά hop για ορίζοντα h	$goodness\left(Neighbor_i T\right) = \sum_{j=0\ldots h}\left(goodness\left(N_i\left[j\right], T\right)/F_{j-1}\right)$
goodness	$goodness\left(Neighbor_i Q\right) = \sum_{j=0\ldots h}\left(goodness\left(N_i\left[j\right], Q\right)/F_j\right)$	-	

Πίνακας 3.2: Διαφορές CRI, HRI, ERI

Κύκλοι στα RIs

Υπάρχει περίπτωση σε ένα P2P δίκτυο να σχηματίζονται κύκλοι. Κάτι τέτοιο θα μπορούσε να δημιουργήσει πρόβλημα στην όλη διαδικασία των ανανεώσεων των RIs. Το πρόβλημα μπορεί να λυθεί αν κάθε κόμβος που στέλνει ένα ανανεωμένο RI, προσθέσει στο μήνυμα κι ένα μοναδικό αναγνωριστικό. Κάθε φορά που ένα μήνυμα επιστρέφει στον αρχικό κόμβο, ο κόμβος θα καταλάβει ότι δημιουργήθηκε κύκλος και το σύστημα μπορεί να επανέλθει.

3.4.3 Ταξινόμηση αλγορίθμων αναζήτησης Αδόμητων Συστημάτων

Στον Πίνακα 3.3 φαίνονται, συγκεντρωτικά, οι αλγόριθμοι αναζήτησης αδόμητων συστημάτων που μελετήσαμε και η κατηγορία στην οποία ανήκουν (τυφλοί ή με πληροφορία):

	Τυφλοί	Με πληροφορία
Gnutella ή πλημμύρα ή BFS	×	
Τροποποιημένος BFS	×	
Έξυπνος BFS		×
Επαναληπτικής εκβάθυνσης	×	
Gnutella2	×	
GUESS	×	
APS		×
DRLP		×
Random Walks	×	
Τοπικά indices		×

Πίνακας 3.3: Ταξινόμηση αλγορίθμων αναζήτησης αδόμητων συστημάτων

3.5 Συστήματα Βασισμένα στο Περιεχόμενο

Στα συστήματα αυτά, οι κόμβοι συνδέονται μεταξύ τους με βάση το περιεχόμενο τους. Σε τέτοια συστήματα είναι πιο εύκολο να γίνουν πιο πολύπλοκες αναζητήσεις (ασαφής ή ευρέως φάσματος). Ένα παράδειγμα μιας πιο πολύπλοκης αναζήτησης είναι η εύρεση ενός rock τραγουδιού. Η αναζήτηση δε θα γίνει πάνω σε ένα συγκεκριμένο τίτλο τραγουδιού αλλά σε μια γενικότερη κατηγορία. Στα συστήματα που περιγράψαμε προηγουμένως, τέτοιες αναζητήσεις είναι δύσκολο να απαντηθούν. Παρακάτω, θα περιγράψουμε τέσσερα από τα πιο σημαντικά συστήματα που έχουν προταθεί.

3.5.1 Σημασιολογικά Επικαλυπτόμενα Δίκτυα (Semantic Overlay Networks for P2P Systems - SONS)

Στα SONS [13] οι κόμβοι ομαδοποιούνται με βάση το περιεχόμενό τους. Οι ομάδες επικαλύπτονται, δηλαδή μπορεί ένας κόμβος να ανήκει σε πολλές ομάδες. Ένα μήνυμα αναζήτησης κατανέμεται μόνο στις σχετικές με το μήνυμα ομάδες και δρομολογείται μόνο μέσα σε αυτές. Με αυτό τον τρόπο, οι άσχετες με το μήνυμα, ομάδες (άσχετοι κόμβοι δηλαδή) δεν καταναλώνουν πόρους για αυτό το μήνυμα.

Κάθε σύνδεσμος μεταξύ δύο κόμβων στα SONS αναγνωρίζεται από μια τριάδα n_i, n_j, L όπου n_i, n_j οι κόμβοι που συνδέει ο σύνδεσμος και L το όνομα της ομάδας στην οποία ανήκουν. Ο στόχος είναι κάθε κόμβος και μήνυμα αναζήτησης να σχετίζονται με μια έννοια-όνομα L που θα είναι και το όνομα της ομάδας στην οποία ανήκουν. Ένα μήνυμα αναζήτησης θα κατατάσσεται σε κάποιες ομάδες ανάλογα με το L και η αναζήτηση θα γίνεται μόνο στους

κόμβους που ανήκουν στην ομάδα με αυτό το L. Ένας κόμβος κατατάσσεται σε μια ή περισσότερες ομάδες ανάλογα με τα αρχεία τα οποία διαθέτει. Επίσης, τα SONS δομούνται ιεραρχικά. Δηλαδή μια ομάδα (SON) μπορεί να είναι υποσύνολο ή υπερσύνολο κάποιας άλλης κ.ο.κ.

Η κατηγοριοποίηση γίνεται έτσι ώστε τα αρχεία κάθε κατηγορίας να ανήκουν σε ένα μικρό αριθμό κόμβων (πολλά επίπεδα ιεραρχίας, ίση δημοτικότητα), οι κόμβοι να έχουν αρχεία σε μικρό αριθμό κατηγοριών και ο αλγόριθμος κατηγοριοποίησης να είναι γρήγορος και όσο γίνεται αλάνθαστος. Γι' αυτό το λόγο, οι κόμβοι κατατάσσονται με βάση δύο στρατηγικές:

1. Συντηρητική στρατηγική. Τοποθετεί έναν κόμβο σε μια ομάδα, αν έχει έστω και ένα αρχείο κατηγοριοποιημένο στην ομάδα αυτή. Το μειονέκτημα της στρατηγικής αυτής είναι ότι παράγει πολλές συνδέσεις.

2. Επιθετική στρατηγική: Τοποθετεί έναν κόμβο σε μια ομάδα, αν έχει έναν ικανό αριθμό εγγράφων κατηγοριοποιημένα σε αυτή την ομάδα. Το μειονέκτημα αυτής της στρατηγικής είναι ότι υπάρχει περίπτωση να μην βρεθούν όλα τα αρχεία που μπορεί να απαντούν ένα ερώτημα.

Επίσης, τα αρχεία κατατάσσονται με βάση δύο τεχνικές:

1. Ακριβής ανάθεση. Ένα αρχείο κατηγοριοποιείται μόνο στην έννοια-ομάδα στην οποία ανήκει.

2. Ολική ανάθεση. Ένα αρχείο κατηγοριοποιείται στην έννοια-ομάδα στην οποία ανήκει αλλά και σε όλη την ανώτερη ιεραρχία.

Η αναζήτηση γίνεται ανάλογα με τις στρατηγικές κατάταξης κόμβων και με τις τεχνικές κατάταξης δεδομένων. Δηλαδή, ανάλογα με τις στρατηγικές και τις τεχνικές αυτές εξαρτάται αν η αναζήτηση θα γίνει σε μια ομάδα, σε μέρος της ιεραρχίας ή και σε όλη την ιεραρχία.

3.5.2 Κατάταξη Περιεχομένου Χρησιμοποιώντας Τοπικότητα Βάσει Ενδιαφέροντος

Το σύστημα αυτό οργανώνεται πάνω από το σύστημα της Gnutella για να λύσει το πρόβλημα της κλιμάκωσης που παρουσιάζεται. Εδώ, οι κόμβοι οργανώνονται με βάση το περιεχόμενό τους. Στηριζόμαστε στην ιδέα ότι αν ένας κόμβος έχει αντικείμενα με περιεχόμενο που ενδιαφέρει κάποιον άλλο κόμβο, τότε είναι πολύ πιθανό ότι θα έχει κι άλλα αντικείμενα με περιεχόμενο που θα ενδιαφέρει τον ίδιο κόμβο. Για να διασυνδέσουμε κόμβους οι οποίοι έχουν αρχεία με κοινό περιεχόμενο χρησιμοποιούμε τον αλγόριθμό των Συντομεύσεων. Το πρωτόκολλο των Συντομεύσεων ορίζει ότι κόμβοι με κοινό περιεχόμενο δημιουργούν συντομεύσεις μεταξύ τους. Οι κόμβοι χρησιμοποιούν αυτές τις συντομεύσεις για να εντοπίσουν το περιεχόμενο που ψάχνουν. Αν κάποια συντόμευση αποτύχει, τότε καταφεύγουμε στο μηχανισμό αναζήτησης της Gnutella (πλημμύρα). Αρχικά, όταν ένας κόμβος δεν έχει δημιουργήσει συντομεύσεις χρησιμοποιεί πλημμύρα για να ψάξει για κάποιο συγκεκριμένο περιεχόμενο. Από τη στιγμή, όμως, που το βρει, δημιουργεί συντόμευση προς τον κόμβο που το έχει, έτσι σε επόμενες αναζητήσεις χρησιμοποιείται αυτή η συντόμευση (για το περιεχόμενο αυτό). Εκτός

από αυτόν τον τρόπο, συντομεύσεις μπορούν να βρεθούν αν οι κόμβοι ανταλλάσσουν λίστες από συντομεύσεις. Ένας κόμβος μπορεί να αποθηκεύσει και πολλαπλές συντομεύσεις αλλά δε μπορεί να κρατάει απεριόριστο αριθμό συντομεύσεων κι αυτό γιατί αρχικά δεσμεύει ένα σταθερό ποσό αποθηκευτικού χώρου ειδικά για τις συντομεύσεις. Έτσι, θα πρέπει να ξέρουμε ποιες από τις συντομεύσεις να κρατήσουμε αν αυτός ο χώρος γεμίσει. Αυτό που γίνεται είναι να σβήνουμε από τον αποθηκευτικό αυτό χώρο τη συντόμευση που έχει χρησιμοποιηθεί λιγότερο. Ένας κόμβος μπορεί να κρατάει περισσότερες από μια συντομεύσεις προς ένα κόμβο. Το ποια θα χρησιμοποιηθεί εξαρτάται από ένα σύνολο από μετρικές.

3.5.3 Συσχετιστική Αναζήτηση σε P2P Δίκτυα

Εδώ, οι κόμβοι κατατάσσονται με βάση κάποιους κανόνες-οδηγούς. Οι κόμβοι που ανήκουν σε ένα κανόνα-οδηγό πρέπει να έχουν αντικείμενα με σημασιολογικά ίδιο περιεχόμενο. Επίσης, οι κόμβοι κρατάνε για κάθε αντικείμενο που έχουν, έναν κανόνα κατοχής που υποδεικνύει ποιοι κόμβοι έχουν το συγκεκριμένο αντικείμενο. Οι κανόνες-οδηγοί περιέχουν κόμβους με ίδιο σημασιολογικά περιεχόμενο, ενώ οι κανόνες κατοχής σχετίζονται με δεδομένα τα οποία πρέπει να ακολουθούν έναν προαπαιτούμενο για να ανήκουν στον ίδιο κόμβο. Για να γίνει η αναζήτηση, ο κόμβος που κάνει την αίτηση αποφασίζει σε ποιο κανόνα-οδηγό να στείλει το μήνυμα αναζήτησης. Αυτό γίνεται με τον αλγόριθμο τυχαίου κανόνα κατοχής (RAPIER). Ο αλγόριθμος διαλέγει ένα τυχαίο αντικείμενο από αυτά που έχει, ψάχνει τους κόμβους που έχουν αυτό το αντικείμενο (χρησιμοποιώντας τον κανόνα κατοχής για αυτό το αντικείμενο) και ελέγχει τους κόμβους που βρίσκει αν έχουν το αντικείμενο για το οποίο έχει αρχικά κάνει αίτηση. Η διαδικασία συνεχίζεται μέχρι να βρεθεί το αντικείμενο. Στη συνέχεια, η αναζήτηση μέσα στον κανόνα-οδηγό που επιλέχθηκε, γίνεται με τη μέθοδο της πλημμύρας. Ο αλγόριθμος RAPIER στηρίζεται στις λανθάνουσες σημασιολογικές έννοιες των αντικειμένων. Δηλαδή, αν ένας κόμβος ψάχνει ένα αντικείμενο και το βρει σε έναν άλλο συγκεκριμένο κόμβο, τότε υπάρχει πιθανότητα, αντικείμενα που σχετίζονται με το προηγούμενο αντικείμενο σημασιολογικά, να βρίσκονται στον ίδιο κόμβο. Για να μετρηθεί η απόδοση του RAPIER, συγκρίνεται με τους αλγορίθμους Ομοιόμορφης τυχαίας αναζήτησης (URAND) και Αναλογικής τυχαίας αναζήτησης (PRAND). Όπως αποδεικνύεται και στο ο αλγόριθμος RAPIER είναι τουλάχιστον όσο καλός είναι και ο PRAND. Ένας βελτιωμένος αλγόριθμος RAPIER είναι ο αλγόριθμος GAS. Ο GAS δεν επιλέγει τυχαία τον κανόνα-οδηγό στον οποίο θα προωθήσει το μήνυμα αναζήτησης, αλλά λαμβάνει υπόψη του τα προηγούμενα μηνύματα αναζήτησης και επιλέγει τον κανόνα-οδηγό με κάποια πιθανότητα, που θα απέδιδε καλύτερα για όλα αυτά τα μηνύματα.

3.5.4 Ανάκτηση Πληροφορίας Χρησιμοποιώντας SONS

Το σύστημα αυτό είναι βασισμένο σε κατανεμημένους πίνακες κατακερματισμού (DHT) όπως τα CAN και Chord. Η διαφορά, είναι ότι εδώ οι κόμβοι κατατάσσονται στο DHT με βάση το περιεχόμενό τους και όχι με βάση κάποιο αναγνωριστικό. Επίσης, και η τοποθέτηση των αντικειμένων στους κόμβους γίνεται με βάση το περιεχόμενο και όχι με βάση κάποιο id. Η τοποθέτηση αυτή, γίνεται δημιουργώντας ένα vector για κάθε αντικείμενο (VSM - Vector

Space Model). Τα αντικείμενα ή τα μηνύματα αναζήτησης σε ένα τέτοιο vector αναπαρίστανται σαν όροι του vector. Κάθε στοιχείο του vector αντιστοιχεί στην σπουδαιότητα του όρου μέσα στο αντικείμενο (ή στο μήνυμα αναζήτησης). Τα στοιχεία του εστορ υπολογίζονται στατιστικά και η κατάταξη των αντικειμένων, που παίρνουμε σαν απάντηση από ένα ερώτημα, γίνεται με βάση την ομοιότητα μεταξύ του vector του αντικειμένου και του εστορ του ερωτήματος.

Ο VSM υποφέρει από συνώνυμα και από θόρυβο στα αντικείμενα που αναπαριστά. Αυτό το πρόβλημα λύνεται από τη Λανθάνουσα Σημασιολογική Δεικτοδότηση (LSI). Στο LSI χρησιμοποιείται Μονή Αποσύνθεση Τιμής (SVD) για να μετατρέψουμε τους πολυδιάστατους όρους vector σε μικρότερης διάστασης σημασιολογικό vector. Εδώ, τα στοιχεία του vector αντιστοιχούν στην σπουδαιότητα μιας αφηρημένης έννοιας μέσα σε ένα αντικείμενο/ερώτημα. Η εισαγωγή ενός κόμβου (αντικειμένου) στο LSI γίνεται δημιουργώντας ένα σημασιολογικό vector για αυτό το αντικείμενο και στη συνέχεια χρησιμοποιώντας τη διαδικασία της εισαγωγής στο CAN (με κλειδί το vector και τιμή την url διεύθυνση του κόμβου που έχει το αντικείμενο). Η αναζήτηση γίνεται δημιουργώντας ένα εστορ για το ερώτημα και στη συνέχεια χρησιμοποιώντας τη διαδικασία της αναζήτησης στο CAN. Όταν το ερώτημα φτάσει στον προορισμό του, μεταδίδεται σε όλους τους κόμβους, σε κάποια ακτίνα, η οποία καθορίζεται από κάποιο όριο ομοιότητας ή από των αριθμό των απαντήσεων που έχει θέσει ο χρήστης. Στη συνέχεια, όλοι οι κόμβοι που λαμβάνουν το ερώτημα κάνουν μια τοπική αναζήτηση χρησιμοποιώντας LSI και επιστρέφονται στο χρήστη αναφορές των αντικειμένων.

3.6 Μέθοδοι Replication στα P2P Συστήματα

Παρακάτω θα αναφερθούμε σε μεθόδους συντήρησης και αντιγραφής δεδομένων (replication) σε δομημένα και αδόμητα συστήματα. Επίσης, θα αναφερθούμε και σε μερικούς αλγορίθμους ανανέωσης δεδομένων, οι οποίοι βοηθούν να κρατάμε τα δεδομένα ενημερωμένα.

3.6.1 Μέθοδοι Replication στα Δομημένα Συστήματα

Οι μέθοδοι replication αποσκοπούν στο να κάνουν τα δεδομένα περισσότερο διαθέσιμα και άρα να έχουμε πιο γρήγορη αναζήτηση.

Μια μέθοδος replication που χρησιμοποιεί το CAN, είναι η ανάθεση πολλαπλών πραγματικοτήτων σε κάθε κόμβο, δηλαδή πολλαπλών, ανεξάρτητων χώρων συντεταγμένων. Σε κάθε πραγματικότητα, θα ανατίθενται στον κόμβο διαφορετικά σύνολα από γείτονες, ενώ τα δεδομένα θα αντιγράφονται σε κάθε πραγματικότητα.

Μια άλλη μέθοδος replication που χρησιμοποιεί το CAN, είναι η χρησιμοποίηση πολλών, διαφορετικών συναρτήσεων κατακερματισμού για ένα δεδομένο, οπότε αυτό θα αντιστοιχίζεται σε διαφορετικά σημεία στο χώρο συντεταγμένων, άρα και σε διαφορετικούς κόμβους. Τα δεδομένα αντιστοιχίζονται σε περισσότερους του ενός κόμβους, άρα, ενώ αρχικά θα τα βρίσκαμε σε ένα συγκεκριμένο κόμβο ο οποίος απέχει k βήματα από τον κόμβο που τα αναζητεί, τώρα ενδέχεται να τα βρούμε σε κάποιον κόμβο (όχι στον αρχικό) που απέχει απόσταση k' βήματα από τον κόμβο που τα αναζητεί, όπου $k' < k$.

Μια ακόμη μέθοδος replication του CAN είναι και οι υπερφορτωμένες ζώνες. Με αυτή τη μέθοδο, περισσότεροι από ένας κόμβοι αναλαμβάνουν μία ζώνη. Η ζώνη αυτή λέγεται υπερφορτωμένοι ζώνη.

Εκτός από αυτές μεθόδους, το CAN προσφέρει και caching έτσι ώστε τα δεδομένα να μπορούν να αναπαράγονται σε πολλούς κόμβους.

Επίσης, μια μέθοδος replication που χρησιμοποιεί το Chord, είναι ο μηχανισμός των successor-lists. Η successor-list είναι μια λίστα από τους r κοντινότερους συςςεσσορς ενός κόμβου. Κάθε κόμβος κρατάει μια successor-list. Αν ένας κόμβος παρατηρήσει ότι ο successor του απέτυχε, τότε τον αντικαθιστά με τον πρώτο ενεργό κόμβο αυτής της λίστας.

3.6.2 Μέθοδοι Replication στα Αδόμητα Συστήματα

Το πρόβλημα που προσπαθούν να λύσουν οι replication μέθοδοι είναι πόσα αντίγραφα ενός αντικειμένου θα έπρεπε να δημιουργηθούν έτσι ώστε να ελαχιστοποιηθεί το overhead των αναζητήσεων για το αντικείμενο, υποθέτοντας ότι ο συνολικός χώρος αποθήκευσης των αντικειμένων σε ένα δίκτυο είναι σταθερός. Έτσι αναπτύχθηκαν 3 είδη replication: 1) ομοιόμορφο, 2) αναλογικό και 3) τετραγωνικής ρίζας. Αν υποθέσουμε ότι ένα αντικείμενο i αντιγράφεται σε r_i κόμβους και ο συνολικός αριθμός των αντικειμένων που αποθηκεύονται είναι R έτσι ώστε $\Sigma_{i=1,m} r_i = R$ τότε:

1. Στο ομοιόμορφο replication όλα τα αντικείμενα αντιγράφονται στον ίδιο αριθμό κόμβων: $r_i = R/m$

2. Στο αναλογικό, η αντιγραφή ενός αντικειμένου είναι αναλογική της πιθανότητας του ερωτήματος για αυτό το αντικείμενο: $r_i - q_i$

3. Στο replication τετραγωνικής ρίζας η αντιγραφή ενός αντικειμένου είναι αναλογική της τετραγωνικής ρίζας της πιθανότητας του ερωτήματος για αυτό το αντικείμενο: $r_i - q_i^{1/2}$.

Μέχρι τώρα έχουμε αναφερθεί στον αριθμό των replications που πρέπει να δημιουργηθούν αλλά δεν έχουμε αναφερθεί στην τοποθεσία που ενδείκνυται να αποθηκευτούν αυτά τα replications. Οι στρατηγικές που ακολουθούνται (και είναι υλοποιήσιμες) είναι:

- Replication κατόχου. Όταν μια αναζήτηση είναι επιτυχής, το αντικείμενο αποθηκεύεται μόνο στον κόμβο που το έχει ζητήσει (χρησιμοποιείται στην Gnutella).

- Replication μονοπατιού. Όταν μια αναζήτηση είναι επιτυχής, το αντικείμενο αποθηκεύεται σε όλους τους κόμβους κατά μήκος του μονοπατιού από τον κόμβο-πηγή στον κόμβο-προορισμό. Αποδεικνύεται, ότι αν ένα P2P σύστημα χρησιμοποιεί k-walkers, και το πλήθος των κόμβων μεταξύ του κόμβου-πηγής και του κόμβου-προορισμού είναι το $1/k$ των συνολικών κόμβων, τότε το replication μονοπατιού καταλήγει σε replication τετραγωνικής ρίζας.

- Τυχαίο replication. Όταν μια αναζήτηση είναι επιτυχής, μετράμε τον αριθμό των κόμβων μεταξύ κόμβου-πηγής και κόμβου-προορισμού, έστω ρ. Τότε, επιλέγουμε τυχαία ρ κόμβους από αυτούς που συνάντησαν οι k-walkers, για να κάνουμε το replication.

3.6.3 Ανανέωση Δεδομένων σε Replicated P2P Συστήματα

Το πρόβλημα της ανανέωσης των δεδομένων όταν αυτά έχουν γίνει replicated (έτσι ώστε να διατηρηθούν έγκυρα), αντιμετωπίζεται από πρωτόκολλα όπως το CUP [21] και από αλγορίθμους όπως οι επιδημικοί αλγόριθμοι [10]. Πιο συγκεκριμένα, το πρόβλημα που τίθεται είναι το πώς θα ανανεώσουμε όλα τα αντίγραφα των δεδομένων που έχουν γίνει replicated, με συνέπεια και με μικρό αριθμό μηνυμάτων. Εδώ, θα αναφερθούμε σε μια στρατηγική ανανεώσεων δεδομένων που προτάθηκε από τους Datta et al.

Η στρατηγική αυτή, βασίζεται στον αλγόριθμο δύο φάσεων προώθησης (push) και έλξης (pull). Η φάση της προώθησης ξεκινά από τον κόμβο που ανανεώνει το δεδομένο. Ο κόμβος αυτός, προωθεί την ανανέωση σε κόμβους που γνωρίζει και οι οποίοι έχουν αντίγραφο του δεδομένου που ανανεώθηκε. Αυτοί, προωθούν την ανανέωση σε κόμβους που γνωρίζουν και που έχουν αντίγραφο του δεδομένου, κ.ο.κ. (όπως η πλημμύρα με TTL). Η φάση της έλξης ξεκινά από τον κόμβο που χρειάζεται να ανανεώσει το αντίγραφό του, είτε γιατί είχε αποσυνδεθεί από το δίκτυο, είτε γιατί έχει δεχθεί ένα αίτημα έλξης και δεν γνωρίζει αν έχει την πιο πρόσφατη τιμή του δεδομένου. Οι φάσεις προώθησης και έλξης είναι συνεχόμενες, αλλά μπορεί και να επικαλύπτονται χρονικά.

3.7 Παραδείγματα σύγχρονων δικτύων ομότιμων κόμβων

Στην ενότητα αυτή παρουσιάζονται δύο πρωτόκολλα δικτύων ομότιμων κόμβων που βρίσκουν ευρεία χρήση σε σημερινές εφαρμογές: Το BitTorrent και το Gnutella.

3.7.1 BitTorrent

Το BitTorrent είναι ένα από τα πιο διαδεδομένα πρωτόκολλα ανταλλαγής αρχείων για δίκτυα P2P που χρησιμοποιείται για κοινή χρήση μεγάλου όγκου δεδομένων. Ο προγραμματιστής Bram Cohen σχεδίασε αρχικά το πρωτόκολλο τον Απρίλιο του 2001 και σήμερα το διαχειρίζεται η εταιρία του ιδίου, BitTorrent, Inc. Στο διαδίκτυο διατίθενται πληθώρα από εφαρμογές πελάτη BitTorrent διαθέσιμες για μια μεγάλη ποικιλία από πλατφόρμες υπολογιστών. Με τον όρο πελάτης BitTorrent, αναφερόμαστε σε οποιοδήποτε πρόγραμμα υλοποιεί το πρωτόκολλο BitTorrent. Κάθε πελάτης είναι σε θέση να προετοιμάζει, να αναζητά και να μεταδίδει οποιοδήποτε τύπο αρχείου πάνω από το δίκτυο, κάνοντας χρήση του πρωτοκόλλου. Ως κόμβο θεωρούμε οποιονδήποτε Η/Υ τρέχει μια εφαρμογή πελάτη.

Το πρωτόκολλο BitTorrent μπορεί να διαβιβάσει ένα μεγάλου όγκου αρχείο χωρίς να επιβαρύνει ιδιαίτερα ούτε τον υπολογιστή που μοιράζεται το αρχείο αλλά ούτε και το ίδιο το δίκτυο. Αντίθετα με τη διαδικασία του απλού κατεβάσματος (download) ενός αρχείου από

έναν εξυπηρετητή, το BitTorrent επιτρέπει στους χρήστες να συμμετέχουν σε ένα πλήθος διακομιστών έτσι ώστε να μπορούν να κατεβάζουν και να ανεβάζουν αρχεία ταυτόχρονα από όλους και σε όλους αντίστοιχα. Η διαδικασία αυτή αποτελεί μια εναλλακτική μέθοδο για την κοινή χρήση αρχείων πάνω από δίκτυα περιορισμένου εύρους μεταγωγής δεδομένων (bandwidth) και λειτουργεί αποδοτικά τόσο σε υπολογιστές μικρής ισχύος όσο και σε συσκευές κινητής τηλεφωνίας.

Ένας χρήστης που επιθυμεί να μοιραστεί ένα αρχείο, αρχικά δημιουργεί ένα αρχείο περιγραφής torrent το οποίο διανέμει με τα γνωστά συμβατικά μέσα (διαδίκτυο, ηλεκτρονικό ταχυδρομείο, κ.τ.λ.). Το παραπάνω αρχείο περιέχει πληροφορίες σχετικά με το αρχείο που πρόκειται να μοιραστεί και σχετικά με τον κόμβο που συντονίζει τη μεταγωγή του αρχείου (tracker). Στη συνέχεια κάνει διαθέσιμο το εν λόγω αρχείο μέσω ενός κόμβου BitTorrent, λειτουργώντας σαν αρχικός τροφοδότης (seed). Όσοι έχουν στην κατοχή τους το αρχείο περιγραφής μπορούν να το χρησιμοποιήσουν από τους δικούς τους BitTorrent κόμβους, οι οποίοι λειτουργώντας σαν απορροφητές (leechers), κατεβάζουν το αρχείο συνδεόμενοι με τον αρχικό τροφοδότη ή/και άλλους απορροφητές. Το κατέβασμα με χρήση του παραπάνω πρωτοκόλλου διαφέρει από το κλασικό κατέβασμα αρχείων στα παρακάτω στοιχεία:

- Το BitTorrent πραγματοποιεί πολλά και μικρά ερωτήματα πάνω από συνδέσεις TCP σε διάφορους κόμβους, ενώ η κλασική πρακτική υποδηλώνει μια και μοναδική TCP σύνδεση με ένα μόνο επιπλέον κόμβο.

- Το BitTorrent κατά το κατέβασμα αρχείων δεν υπαγορεύει την ακριβή σειρά με την οποία θα καταφτάσουν τα κομμάτια του αρχείου, ενώ με την κλασική πρακτική το κατέβασμα γίνεται σειριακά για τα κομμάτια ενός αρχείου.

Τα αρχεία που μοιράζονται οι χρήστες, αρχικά χωρίζονται σε μικρά κομμάτια, έτσι ώστε όποτε κάποιος κόμβος απορροφητής λαμβάνει ένα από τα κομμάτια αυτά, αμέσως ξεκινά να λειτουργεί ως τροφοδότης για το συγκεκριμένο κομμάτι, ανακουφίζοντας έτσι τον αρχικό κόμβο από το φόρτο της αποστολής ενός αντιγράφου του συγκεκριμένου κομματιού σε κάθε απορροφητή. Με το BitTorrent, ο φόρτος της διαδικασίας του μοιράσματος ενός αρχείου καταμερίζεται μεταξύ όσων ενδιαφέρονται να το αποκτήσουν. Καθένα από τα παραπάνω κομμάτια προστατεύεται από μια κρυπτογραφική συνάρτηση hash που περιέχεται στο αρχείο περιγραφής. Αυτό αποτρέπει τους κόμβους να τροποποιήσουν κακόβουλα τα κομμάτια του αρχείου που τροφοδοτούν σε άλλους χρήστες. Αν κάποιος κόμβος ξεκινήσει έχοντας ένα αυθεντικό αντίγραφο του αρχείου περιγραφής, μπορεί να εξακριβώσει την αυθεντικότητα του αρχείου που θα έχει κατεβάσει στο τέλος. Όταν ένας κόμβος απορροφητής ολοκληρώσει το κατέβασμα ενός αρχείου, αυτομάτως μετατρέπεται σε κόμβο τροφοδότη. Η πρακτική αυτή συμβάλει στο γενικότερο βαθμό υγείας του αρχείου, κάτι που προσδιορίζεται από τον αριθμό των αυτούσιων αντιγράφων του που διατίθενται στο δίκτυο.

Η κατανεμημένη φύση του BitTorrent οδηγεί τελικά σε εξάπλωση των κοινόχρηστων αρχείων μεταξύ των χρηστών με τρόπο που να θυμίζει πλημμύρα. Όσο περισσότεροι κόμβοι συμμετέχουν στο πλήθος των διακομιστών, τόσο μεγαλύτερη η πιθανότητα για ένα επιτυχημένο κατέβασμα ενός αρχείου. Παρόλα αυτά, το παραπάνω εμπεριέχει κι ένα κόστος, ότι

δύναται να καθυστερήσει αρκετά η εφαρμογή να φτάσει στη μέγιστη ταχύτητα μεταγωγής γιατί είναι πιθανό να καθυστερεί η δημιουργία των συνδέσεων. Σε αντίθεση με την απλούστερη και καθιερωμένη πρακτική της κοινής χρήσης αρχείων μέσω απλών εξυπηρετητών του διαδικτύου, αυτή η μέθοδος μειώνει σημαντικά τόσο το φόρτο εργασίας του Hardware όσο και των δικτυακών πόρων του κάθε κόμβου τροφοδότη. Επιπλέον, παρέχει πλεονεκτήματα στους χρήστες έναντι τεχνικών προβλημάτων, ελαττώνει την εξάρτηση από τον αρχικό τροφοδότη και γενικότερα παρέχει μια προσωρινή πηγή για κάθε αρχείο η οποία είναι δυσκολότερο να εντοπιστεί, καθότι προσωρινή, αντιθέτως δηλαδή με ότι συμβαίνει με τη συμβατική πρακτική της παροχής αρχείων με χρήση ενός σταθερού διακομιστή.

3.7.2 Gnutella

Το Gnutella είναι ένα πρώιμο P2P δίκτυο που προσπάθησε να διαφοροποιήσει την ανταλλαγή αρχείων μουσικής (που συνήθως καταπατά τα πνευματικά δικαιώματα ορισμένων) από την ανταλλαγή αρχείων γενικότερου περιεχομένου. Το ενδιαφέρον σχετικά με το Gnutella είναι ότι υπήρξε ένα από τα πρώτα συστήματα αυτού του τύπου που δεν ήταν εξαρτημένο από ένα συγκεντρωτικό μητρώο καταγραφής αρχείων-objects. Αντιθέτως, οι συμμετέχοντες στο Gnutella τοποθετούνταν οι ίδιοι σε ένα επικαλυπτόμενο δίκτυο. Συγκεκριμένα, κάθε κόμβος στον οποίο τρέχει το λογισμικό του Gnutella έχει πληροφορίες για ορισμένα από τα υπόλοιπα μηχανήματα όπου επίσης τρέχει το ίδιο λογισμικό.

Οποτεδήποτε κάποιος χρήστης σε ένα συγκεκριμένο κόμβο του δικτύου θελήσει να βρει ένα αρχείο-object, το Gnutella στέλνει ένα ερώτημα για το συγκεκριμένο αρχείο - λ.χ., καθορίζοντας το όνομα του αρχείου - στους γειτονικούς κόμβους του σύμφωνα με το γράφο. Αν σε κάποιον από αυτούς βρίσκεται το υπό αναζήτηση αρχείο, απαντάει στον κόμβο που έκανε το ερώτημα στέλνοντάς του ένα απαντητικό μήνυμα διευκρινίζοντας από που μπορεί να κατέβει το συγκεκριμένο αρχείο (π.χ., δίνοντάς του τη διεύθυνση IR και τον αριθμό του TCP πορτ). Ο κόμβος του χρήστη στη συνέχεια μπορεί να χρησιμοποιήσει μηνύματα του τύπου GET ή PUT για να έχει πρόσβαση στο αρχείο. Αν ο κόμβος δεν μπορεί να επιληφθεί του αρχικού ερωτήματος, προωθεί το ερώτημα σε καθέναν από τους γειτονικούς κόμβους του (εκτός από τον κόμβο ο οποίος έστειλε το ερώτημα αρχικά), και επαναλαμβάνεται η παραπάνω διαδικασία. Με άλλα λόγια, το Gnutella διαχέει τα ερωτήματα σε όλη την έκταση του δικτύου μέχρις ότου εντοπιστεί το συγκεκριμένο αρχείο. Το Gnutella ορίζει ένα συγκεκριμένο TTL για κάθε ερώτημα, ούτως ώστε αυτή η διάχυση να μη συνεχιστεί αόριστα.

Επιπλέον του TTL και του αλφαριθμητικού που περιέχει κάθε ερώτημα, υπάρχει και ένα μοναδικό στοιχείο ταυτοποίησής του (Query Identifier - QID) στο οποίο δεν εμφανίζεται η ταυτότητα του αρχικού αποστολέα του ερωτήματος. Αντί αυτού, κάθε κόμβος διατηρεί ένα αρχείο καταγραφής όλων των ερωτημάτων που έχει επεξεργαστεί προσφάτως, δηλαδή αποθηκεύει τόσο το στοιχείο ταυτοποίησης του ερωτήματος όσο και την ταυτότητα του γειτονικού κόμβου που έστειλε το ερώτημα. Το παραπάνω ιστορικό χρησιμοποιείται από τους κόμβους με δυο τρόπους. Πρώτον, αν ο κόμβος λάβει κάποιο ερώτημα με στοιχείο ταυτοποίησης ίδιο με κάποιο που έχει ήδη στη λίστα, το νέο αυτό ερώτημα δεν το προωθεί. Αυτή η πρακτική

εξυπηρετεί στην παρεμπόδιση κυκλικά επαναλαμβανόμενων προωθήσεων που λαμβάνουν χώρα γρηγορότερα από ότι ορίζει το TTL. Δεύτερον, οποτεδήποτε ο κόμβος λαμβάνει ένα απαντητικό μήνυμα σε κάποιο ερώτημα από κάποιο κόμβο μεταγενέστερο στην κατεύθυνση ροής του ερωτήματος, αμέσως γνωρίζει πως να προωθήσει την απάντηση αυτή στον πρωθύστερο κόμβο που του είχε στείλει αρχικά το ερώτημα. Με τον τρόπο αυτό, η απάντηση βρίσκει το δρόμο προς τον αρχικό αποστολέα του ερωτήματος, χωρίς κανείς από τους ενδιάμεσους κόμβους να γνωρίζει ποιος κόμβος έκανε στην πραγματικότητα το αρχικό ερώτημα προς το δίκτυο.

Επιστρέφοντας στο ερώτημα του πως εξελίσσεται ένας γράφος, ένας κόμβος συγκεκριμένα πρέπει να γνωρίζει τουλάχιστον έναν ακόμη κόμβο όταν αρχικά συνδέεται στο δίκτυο του Gnutella. Ο νέος αυτός κόμβος συνδέεται στο δίκτυο μέσω αυτής τουλάχιστον της σύνδεσης. Στη συνέχεια, ο κόμβος αυτός λαμβάνει πληροφορίες για τους γειτονικούς του κόμβους από τις απαντήσεις που του επιστρέφονται στα ερωτήματά του ή σε ερωτήματα άλλων για τα οποία τυχαίνει να μεσολαβήσει. Κάθε κόμβος αποφασίζει ελεύθερα αν τους κόμβους για τους οποίους πληροφορείται ότι είναι γειτονικοί του με τον παραπάνω τρόπο, θα τους διατηρήσει σαν γείτονές του ή όχι. Το πρωτόκολλο Gnutella παρέχει μηνύματα τύπου PING και PONG μέσω των οποίων ο κάθε κόμβος εξετάζει αν ένας γειτονικός κόμβος συνεχίζει να υφίσταται και μέσω ποιού κόμβου ο συγκεκριμένος απαντά, αντίστοιχα.

Πρέπει να γίνει ξεκάθαρο ότι το Gnutella δεν αποτελεί ένα ιδιαίτερα έξυπνο πρωτόκολλο, γεγονός για το οποίο αρκετά μεταγενέστερα συστήματα προσπάθησαν να το βελτιώσουν. Μια διάσταση κατά την οποία θα μπορούσαν να υπάρξουν σημαντικές αλλαγές περιλαμβάνει τον τρόπο με τον οποίο διαχέονται τα ερωτήματα στο δίκτυο. Η συνεχής διάχυση, με την έννοια της πλημμύρας, είναι βολική διότι εξασφαλίζει την εύρεση του υπό αναζήτηση αρχείου στη μικρότερη δυνατή απόσταση μετρημένη σε hops, αλλά δεν ανέρχεται με ικανοποιητικό τρόπο. Δύναται επίσης η αποστολή ερωτημάτων σε τυχαίους παραλήπτες, ή σύμφωνα με την πιθανότητα ευστοχίας βασιζόμενοι σε στατιστικά από παλιότερες αναζητήσεις. Μια άλλη διάσταση αφορά την εκ των προτέρων αντιγραφή των αρχείων σε κάθε κόμβο, εφόσον η λογική λέει ότι όσο περισσότερα αντίγραφα ενός αρχείου υπάρχουν, τόσο πιο εύκολος θα είναι ο εντοπισμός τους.

Κεφάλαιο 4

Το δίκτυο Chord

Στο συγκεκριμένο κεφάλαιο γίνεται μια σύντομη περιγραφή του πρωτοκόλλου Chord. Το συγκεκριμένο πρωτόκολλο καθορίζει τον τρόπο με τον οποίο γίνεται η εύρεση της θέσης των κλειδιών, την διαδικασία με την οποία οι νέοι κόμβοι συνδέονται με το δίκτυο καθώς και πως πραγματοποιείται η ανάκαμψη μετά από τυχόν αστοχία κάποιου κόμβου ή ακόμα και μετά από μια προγραμματισμένη αποχώρηση του. Όπως έχει αναφερθεί, το μοντέλο της επικοινωνίας σε δίκτυα που εφαρμόζεται το εν λόγω πρωτόκολλο χαρακτηρίζεται από την συμμετρική και την μεταβατική ιδιότητα μεταξύ των διαφόρων κόμβων που είναι μέλη στο δίκτυο.

4.1 Επισκόπιση

Το πρωτόκολλο Chord προσφέρει τη δυνατότητα για κατανεμημένο υπολογισμό συναρτήσεων hash με μεγάλη ταχύτητα, αντιστοιχίζοντας κλειδιά σε αντίστοιχους κόμβους του δικτύου.

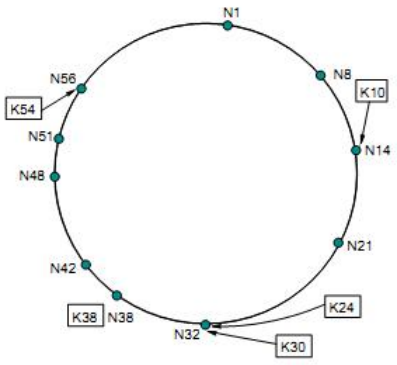
Το πρωτόκολλο αναθέτει τα κλειδιά στους αντίστοιχους κόμβους με την διαδικασία του "consistent hashing" η οποία χαρακτηρίζεται από αρκετές επιθυμητές ιδιότητες. Η συγκεκριμένη διαδικασία έχει την ιδιότητα του ισότιμου διαμοιρασμού του φόρτου εργασίας μεταξύ των κόμβων και μάλιστα με υψηλό μέτρο πιθανότητας. Επιπλέον, κατά την αποχώρηση ενός υπάρχοντος κόμβου (του N^{ou}) ή την σύνδεση ενός νέου, είναι απαραίτητη η μετακίνηση μόνο ενός μέρους O(1/N) των κλειδιών σε διαφορετικούς κόμβους

Ένα επιπλέον χαρακτηριστικό του πρωτοκόλλου είναι η βελτίωση της επεκτασιμότητας της διαδικασίας "consistent hashing", αφού δεν απαιτείται πλέον από κάθε κόμβο να γνωρίζει πληροφορίες για όλους τους υπόλοιπους κόμβους του δικτύου. Για την λειτουργία ενός κόμβου τύπου Chord είναι μόνο απαραίτητο ένα ελάχιστο σύνολο πληροφοριών δρομολόγησης μεταξύ των άλλων κόμβων. Εφόσον αυτή η πληροφορία είναι κατανεμημένη στο δίκτυο, κάθε κόμβος υπολογίζει την συνάρτηση hash και μέσω της επικοινωνίας με τους υπόλοιπους κόμβους. Σε ένα δίκτυο που αποτελείται από N κόμβους, είναι απαραίτητο για κάθε κόμβο να διατηρεί πληροφορίες μόνο για O(logN) άλλους κόμβους. Τέλος μία αναζήτηση σε ένα τέτοιο δίκτυο απαιτεί την αποστολή O(logN) μηνυμάτων.

4.2 Consistent hashing

Η συνάρτηση "consistent hash" αναθέτει σε κάθε κόμβο και σε κάθε κλειδί ένα αναγνωριστικό (στο εξής ID) μήκους m-bit χρησιμοποιώντας την συνάρτηση SHA-1 ως βασική συνάρτηση hash. Πιο συγκεκριμένα, το ID του κάθε κόμβου υπολογίζεται εφαρμόζοντας την συνάρτηση hash στη διεύθυνση IP του κάθε κόμβου και αντίστοιχα το ID του καθε κλειδιού υπολογίζεται εφαρμόζοντας την συνάρτηση hash στο ίδιο το κλειδί. Στη συνέχεια του κεφαλαίου η έννοια του κλειδιού πριν και μετά την εφαρμογή της συνάρτησης hash δεν θα διαχωρίζεται καθώς θα έχει άμεση σχέση με τα συμφραζόμενα. Ομοίως, ο όρος κόμβος θα αναφέρεται εκ περιτροπής και στον καθ' εαυτού κόμβο όπως και στο αντίστοιχο ID του κόμβου. Το μήκος m του ID θα πρέπει να είναι κατάλληλο ώστε η πιθανότητα είτε δυο κόμβων είτε δυο κλειδιών να έχουν το ίδιο ID να είναι αμελητέα.

Στη συνέχεια περιγράφεται η διαδικασία με την οποία ανατίθενται τα κλειδιά στους αντίστοιχους κόμβους μέσω της διαδικασίας "consistent hashing". Τα ID τοποθετούνται σε μια κυκλική διάταξη πλήθους 2m (mod 2m). Κάθε κλειδί k ανατίθεται στον πρώτο κόμβο του οποίου το ID είναι ίσο ή ακολουθεί το ID του k σε αυτή την κυκλική διάταξη των ID. Ο συγκεκριμένος κόμβος πλέον ονομάζεται successor του κλειδιού k (successor(k)). Δεδομένου ότι τα ID βρίσκονται κυκλικά διατεταγμένα και αριθμημένα από 0 έως 2m-1, ο successor(k) είναι ο πρώτος κόμβος που έπεται, δεξιόστροφα του k. Στη συνέχεια αυτή η κυκλική διάταξη θα αναφέρεται πιο απλά ως δακτύλιος (Chord Ring).



Σχήμα 4.1: Δακτύλιος Chord

Στο σχήμα 4.1 απεικονίζεται ένας δακτύλιος με $m = 6$. Ο δακτύλιος αποτελείται από 10 κόμβους και αποθηκεύονται σε αυτόν 5 κλειδιά. Ο successor(10) είναι ο κόμβος 14. Συνεπώς το κλειδί 10 θα βρίσκεται στον κόμβο 14. Με αντίστοιχο τρόπο τα κλειδιά 24 και 30 θα βρίσκονται στον κόμβο 32, το κλειδί 38 στον κόμβο 38 και το κλειδί 54 στον κόμβο 56.

Η διαδικασία του "consistent hashing" είναι σχεδιασμένη έτσι ώστε να δίνει την δυνατότητα σε νέους κόμβους να συνδέονται στο δίκτυο και σε υπάρχοντες να αποσυνδέονται από αυτό δημιουργώντας την ελάχιστη διαταραχή στο δίκτυο. Για την διατήρηση της σωστής αντιστοίχησης κατά την σύνδεση ενός νέου κόμβου n στο δίκτυο, συγκεκριμένα κλειδιά τα οποία ήταν προηγουμένως τοποθετημένα στον κόμβο successor(n), τώρα πλέον τοποθετούν-

ται στον κόμβο n. Αντίστοιχα, στην περίπτωση που κάποιος κόμβος αποσυνδεθεί, τότε τα κλειδιά που διατηρούσε τοποθετούνται στον κόμβο successor(n). Με αυτόν τον τρόπο δεν χρειάζεται καμία άλλη αλλαγή στην ανάθεση των κλειδιών. Συγκεκριμένα, στο παράδειγμα που αναφέρθηκε, εάν ένας νέος κόμβος με ID 26 συνδεόταν στο δίκτυο τότε σε αυτόν τον κόμβο θα τοποθετούνταν το κλειδί με ID 24 που πριν βρισκόταν στον κόμβο με ID 32.

Τα ακόλουθα αποτελέσματα έχουν αποδειχθεί στις αντίστοιχες εργασίες οι οποίες πρότειναν την διαδικασία του "consistent hashing".

Θεώρημα 4.1: Για κάθε σύνολο N κόμβων και K κλειδιών, αποδεικνύεται ότι με μεγάλη πιθανότητα ισχύει:

1. Κάθε κόμβος μπορεί να διατηρεί το μέγιστο (1+ε)K/N κλειδιά.

2. Όταν ο (N+1)ος κόμβος συνδέεται στο δίκτυο ή αποσυνδέεται από αυτό, τότε ο αριθμός των κλειδιών που αλλάζει θέση είναι O(K/N) και μόνο από και προς τον εν λόγω κόμβο.

Με την ανωτέρω υλοποίηση της διαδικασίας "consistent hashing", το συγκεκριμένο θεώρημα παρουσιάζει φράγμα για ε = O(logN). Στη συγκεκριμένη εργασία περιγράφεται ότι το ε μπορεί να μειωθεί σε μια αυθαίρετα ελάχιστη σταθερά στην περίπτωση που κάθε κόμβος διατηρεί Ω(logN) στον αριθμό εικονικούς κόμβους ο καθένας από τους οποίους έχει το δικό του ID. Στη συνέχεια του κεφαλαίου τα φράγματα θα υπολογίζονται ως φράγματα για κάθε εικονικό κόμβο. Έτσι, αν κάθε κόμβος διατηρεί ν εικονικούς κόμβους, κάθε φράγμα θα πρέπει να πολλαπλασιάζεται κατά ν.

4.3 Απλή αναζήτηση κλειδιού

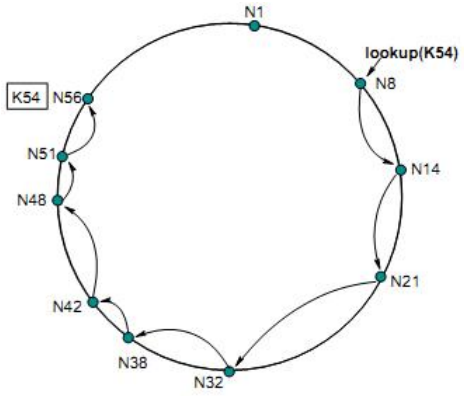
Σε αυτήν την παράγραφο περιγράφεται ο απλός αλλά και ταυτόχρονα σχετικά αργός αλγόριθμος αναζήτησης Chord. Στη συνέχεια περιγράφονται τεχνικές για την επέκταση του βασικού αλγόριθμου από άποψης απόδοσης καθώς επίσης και για την διατήρηση της εγκυρότητας των πληροφοριών δρομολόγησης του πρωτοκόλλου Chord.

Οι αναζητήσεις μπορούν να γίνουν σε έναν δακτύλιο με τους κόμβους να έχουν ελάχιστη γνώση για την κατάσταση του δικτύου. Η απαραίτητη γνώση που θα πρέπει να έχει κάθε κόμβος είναι το πως να επικοινωνήσει με τον successor του σε έναν δακτύλιο. Οι αναζητήσεις για κάθε ID μπορούν να κυκλοφορούν μέσα στον δακτύλιο μέσω αυτών των δεικτών σε successor έως ότου βρεθούν δυο διαδοχικοί κόμβοι που περιβάλλουν το εν λόγω ID. Σε αυτήν την περίπτωση ο κόμβος που αναζητείται είναι ο δεύτερος σε σειρά.

```
// ask node n to find the successor of id
n.find_successor(id)
  if (id ∈ (n, successor])
    return successor;
  else
    // forward the query around the circle
    return successor.find_successor(id);
```

Σχήμα 4.2: Απλή αναζήτηση κλειδιού

Στο σχήμα 4.2 παρουσιάζεται σε μορφή ψευδοκώδικα μια υλοποίηση της απλής αναζήτησης κλειδιού. Ειδικότερα, οι αναφορές σε απομακρυσμένες κλήσεις και απομακρυσμένες μεταβλητές έπονται του ID του απομακρυσμένου κόμβου, ενώ οι αναφορές σε τοπικές μεταβλητές και κλήσεις διεργασιών παραβλέπουν την ύπαρξη του τοπικού κόμβου. Με αυτήν την υπόθεση η n.foo() είναι μια κλήση απομακρυσμένης διεργασίας foo στον κόμβο n, ενώ η n.bar, είναι μια κλήση απομακρυσμένης διεργασίας επιστροφής της μεταβλητής bar από τον κόμβο n. Ο συμβολισμός (a,b] υποδηλώνει τμήμα του δακτυλίου ξεκινώντας από τον κόμβο a (μη συμπεριλαμβανομένου) κινούμενοι δεξιόστροφα έως και τον κόμβο b.



Σχήμα 4.3: Δακτύλιος Chord με αναζήτηση κλειδιού

Στο σχήμα 4.3 απεικονίζεται ένα παράδειγμα στο οποίο ο κόμβος 8 πραγματοποιεί μια αναζήτηση για το κλειδί 54. Ο κόμβος 8 κάνει κλήση της find_successor για το κλειδί 54 η οποία επιστρέφει τον successor για το κλειδί 54 δηλαδή τον κόμβο 56. Το ερώτημα περνά από κάθε κόμβο του δακτυλίου μεταξύ των κόμβων 8 και 56. Το αποτέλεσμα επιστέφει μέσω της ίδιας διαδρομής που είχε ακολουθήσει το αρχικό ερώτημα.

4.4 Κλιμακούμενη αναζήτηση κλειδιού

Το μοντέλο αναζήτησης που παρουσιάστηκε στην προηγούμενη παράγραφο χρησιμοποιεί αριθμό μηνυμάτων γραμμικά ανάλογο με τον αριθμό των κόμβων. Το πρωτόκολλο Chord

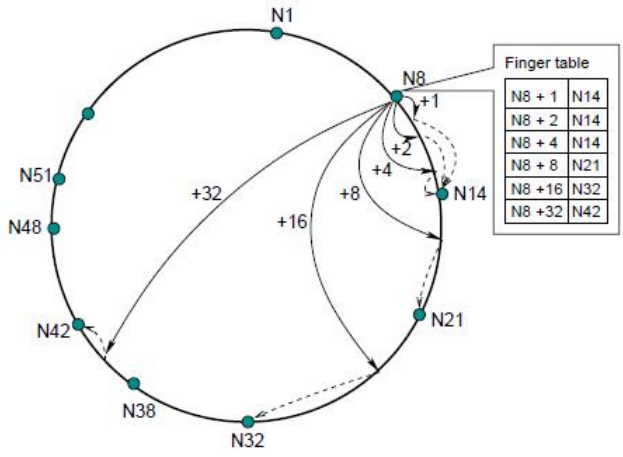
χρησιμοποιεί επιπλέον πληροφορίες δρομολόγησης έτσι ώστε να επιταχύνει την διαδικασία της αναζήτησης. Οι συγκεκριμένες πληροφορίες δεν είναι άμεσα απαραίτητες για την ορθότητα της αναζήτησης η οποία είναι δυνατόν να επιτευχθεί αρκεί κάθε κόμβος να γνωρίζει τον συγγεσσορ του.

Όπως και πριν, θεωρούμε ότι το μήκος του κάθε ID είναι ίσο με m. Επιπλέον κάθε κόμβος n διατηρεί έναν πίνακα δρομολόγησης με μέγιστο μήκος m θέσεων τον οποίο και στο εξής θα αναφέρουμε ως πίνακα finger. Η i^{η} καταχώρηση στον πίνακα finger του κόμβου n περιέχει την ταυτότητα του πρώτου κόμβου s που έπεται του κόμβου ν κατά τουλάχιστον 2i-1 στον συγκεκριμένο δακτύλιο. Πιο συγκεκριμένα, για τον κόμβο s ισχύει: s=successor(n+2i-1) για $1 \leq i \leq m$. Κατά αυτόν τον τρόπο ονομάζουμε τον κόμβο s, το i ^{o} finger του κόμβου n και το συμβολίζουμε ως n.finger[i] (βλ.Σχήμα 4.4).

Notation	Definition
finger[k]	first node on circle that succeeds $(n + 2^{k-1}) \bmod 2^m$, $1 \leq k \leq m$
successor	the next node on the identifier circle; *finger*[1].*node*
predecessor	the previous node on the identifier circle

Σχήμα 4.4: Ορισμός των μεταβλητών του κόμβου ν

Κάθε καταχώρηση στον πίνακα finger περιέχει το ID του δακτυλίου καθώς και την διεύθυνση IP και τον αριθμό θύρας για τον συγκεκριμένο κόμβο. Ας σημειωθεί ότι το πρώτο finger του κόμβου n είναι ο successor(1) του κόμβου n στον δακτύλιο. Για λόγους απλότητας συνήθως το πρώτο finger ενός κόμβου συνήθως θα αναφέρεται ως successor του κόμβου n.



Σχήμα 4.5: Οι καταχωρήσεις του πίνακα finger για τον κόμβο 8

Στο παράδειγμα του σχήματος 4.5 παρουσιάζεται ο πίνακας finger του κόμβου 8. Το πρώτο finger του κόμβου 8 δείχνει στον κόμβο 14 αφού άλλωστε ο κόμβος 14 είναι ο πρώτος κόμβος που έπεται του $(8+20) mod 26 = 9$. Κατά όμοιο τρόπο το τελευταίο finger του κόμβου 8 δείχνει στον κόμβο 42, εφόσον ο κόμβος 42 είναι ο πρώτος κόμβος που έπεται του $(8 + 25) mod 26 =$

40. Το συγκεκριμένο μοντέλο αναζήτησης έχει δυο σημαντικά χαρακτηριστικά. Πρώτον, κάθε κόμβος αποθηκεύει πληροφορία μόνο για έναν μικρό αριθμό άλλων κόμβων και επιπλέον γνωρίζει περισσότερες πληροφορίες για κόμβους οι οποίοι είναι σχετικά κοντά στον δακτύλιο σε σχέση με κόμβους οι οποίοι είναι σε μεγαλύτερη απόσταση. Επιπλέον, ο πίνακας finger ενός κόμβου δεν περιέχει συνήθως αρκετές πληροφορίες έτσι ώστε να είναι δυνατή η εύρεση του successor ενός οποιουδήποτε κλειδιού k. Για παράδειγμα, όπως φαίνεται και στο σχήμα 4.5, ο κόμβος 8 δεν μπορεί να βρει τον successor του κλειδιού 34 από μόνος του, εφόσον ο συγκεκριμένος successor (κόμβος 38) δεν εμφανίζεται στον πίνακα finger του κόμβου 8.

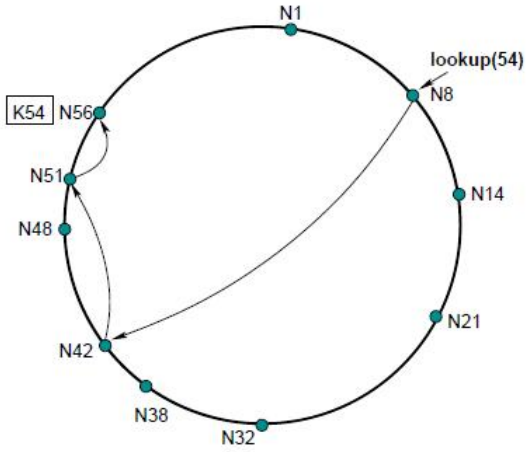
```
// ask node n to find the successor of id
n.find_successor(id)
  if (id ∈ (n, successor])
    return successor;
  else
    n' = closest_preceding_node(id);
    return n'.find_successor(id);

// search the local table for the highest predecessor of id
n.closest_preceding_node(id)
  for i = m downto 1
    if (finger[i] ∈ (n, id))
      return finger[i];
  return n;
```

Σχήμα 4.6: Κλιμακούμενη αναζήτηση χρησιμοποιώντας πίνακα finger

Στο σχήμα 4.6 εμφανίζεται σε μορφή ψευδοκώδικα η διαδικασία find_successor, αναβαθμισμένη με τέτοιο τρόπο ώστε να χρησιμοποιεί πίνακες finger. Στην περίπτωση που το id βρεθεί μεταξύ του n και του successor(n) τότε η find_successor τερματίζει και ο κόμβος n επιστρέφει τον successor του. Σε διαφορετική περίπτωση, ο κόμβος n αναζητά στον πίνακα finger του για τον κόμβο n' του οποίου το ID προηγείται ακριβώς πριν το id. Στη συνέχεια γίνεται κλήση της διαδικασίας find_successor στον κόμβο n'. Ο σκοπός πίσω από αυτήν την επιλογή του n' είναι ότι το όσο περισσότερο κοντά βρίσκεται ο κόμβος n' στο id τόσο περισσότερες πληροφορίες θα κατέχει για την περιοχή του δακτυλίου κοντά στο id.

Για παράδειγμα, ας θεωρήσουμε τον δακτύλιο όπως αυτός φαίνεται στο σχήμα 4.7 και επιπλέον τον κόμβο 8 να επιθυμεί να αναζητήσει τον successor του κλειδιού 54. Εφόσον το μεγαλύτερο finger του κόμβου 8 που να προηγείται του κλειδιού 54 είναι ο κόμβος 42, ο κόμβος 8 θα ζητήσει από τον κόμβο 42 να εκτελέσει το ερώτημα. Στη συνέχεια, ο κόμβος 42 θα υπολογίσει το μεγαλύτερο finger που υπάρχει μέσα στον πίνακα finger που διατηρεί και προηγείται του κλειδιού 54, δηλαδή τον κόμβο 51.Εν τέλει, ο κόμβος 51 θα βρεί ότι ο successor του (κόμβος 56) προηγείται του κλειδιού 54 και άρα θα επιστρέψει σαν απάντηση στον κόμβο 8 τον κόμβο 56.

Δεδομένου ότι κάθε κόμβος έχει καταχωρήσεις για finger σε σημεία που απέχουν από το ID κατά δυνάμεις του δυο μέσα στον δακτύλιο, κάθε κόμβος μπορεί να προωθεί ένα ερώτημα τουλάχιστον κατά την μισή απόσταση μεταξύ του αυτού κόμβου και του ζητούμενου ID. Από αυτό το συμπέρασμα ακολουθεί το θεώρημα:



Σχήμα 4.7: Η διαδρομή του ερωτήματος για το κλειδί 54

Θεώρημα 4.2: Με μεγάλη πιθανότητα, ο αριθμός των κόμβων που θα πρέπει να συμβάλλουν για την αναζήτηση ενός successor σε ένα δίκτυο N κόμβων είναι O(logN).

Απόδειξη: Υποθέτουμε ότι κάποιος κόμβος n επιθυμεί να αναλύσει ένα ερώτημα για τον successor του k. P είναι ο κόμβος που προηγείται άμεσα του k. Στη συνέχεια αναλύεται ο αριθμός των ερωτημάτων που χρειάζονται να προωθηθούν μέχρι τον κόμβο p.

Όπως αναφέρθηκε, στην περίπτωση που $n \neq p$, τότε ο κόμβος n προωθεί το ερώτημά του στον κοντινότερο predecessor του k που υπάρχει στον πίνακα finger του. Επιπλέον, θεωρούμε τέτοιο i έτσι ώστε ο κόμβος p να είναι εντός του διαστήματος $[n+2i-1, n+2i)$. Εφόσον το συγκεκριμένο διάστημα δεν είναι κενό (περιέχει τον κόμβο p), ο κόμβος ν θα επικοινωνήσει με το i^o finger του, τον πρώτο κόμβο f σε αυτό το διάστημα. Η απόσταση (ο αριθμός των ID) μεταξύ του n και του f είναι τουλάχιστον 2i-1. Αυτό σημαίνει ότι ο κόμβος f είναι πιο κοντά στον p από τον κόμβο n ή αντίστοιχα η απόσταση του f από τον p είναι έως και η μισή απόσταση του p από τον n.

Αν η απόσταση μεταξύ του κόμβου που διαχειρίζεται το ερώτημα και του predecessor που υποδιπλασιάζεται σε κάθε βήμα, και έχει αρχικό μέγεθος εως 2m, τότε σε διάστημα m βημάτων η απόσταση αυτή θα είναι ίση με ένα εννοώντας ότι το ερώτημα θα έχει φτάσει στον κόμβο p.

Στην πραγματικότητα, όπως έχει αναφερθεί, υποθέτουμε ότι τα ID κάθε κόμβου και κάθε κλειδιού είναι τυχαία. Σε αυτήν την περίπτωση, ο αριθμός των προωθήσεων που είναι απαραίτητος είναι O(logN). Ύστερα από 2logN προωθήσεις, η απόσταση μεταξύ του κόμβου που έχει το τρέχον ερώτημα και του κλειδιού k θα έχει μειωθεί σε το πολύ $2^m/N^2$. Η πιθανότητα κάθε άλλος κόμβος να βρίσκεται στο συγκεκριμένο διάστημα είναι το πολύ $\frac{1}{N}$, τιμή η οποία θεωρείται αμελητέα. Ως αποτέλεσμα, το επόμενο βήμα προώθησης θα συναντήσει τον επιθυμητό κόμβο. Ο μέσος χρόνος αναζήτησης φτάνει στο $\frac{1}{2logN}$.

Παρά το ότι ο πίνακας finger περιέχει αρκετό χώρο για m εγγραφές στον αριθμό, στην πραγματικότητα μόνο O(logN) finger θα αποθηκευτούν. Όπως αποδείξαμε στο ανωτέρω

θεώρημα, κανένας κόμβος δεν μπορεί να βρίσκεται σε απόσταση $\frac{2m}{N^2}$ από άλλον κόμβο. Κατά αυτόν τον τρόπο, το i^o finger οποιουδήποτε κόμβου, για $i \leq m - 2logN$, θα είναι ίσο με τον άμεσο successor του κόμβου με αρκετά μεγάλη πιθανότητα και έτσι δεν είναι απαραίτητο να αποθηκευτεί σε διαφορετική τοποθεσία.

4.5 Δυναμική λειτουργία και αστοχίες

Στην πραγματικότητα το πρωτόκολλο Chord θα πρέπει να είναι ικανό να ανταπεξέλθει όταν νέοι κόμβοι συνδέονται στο δίκτυο ή όταν οι υπάρχοντες κόμβοι αποσυνδέονται από αυτό (είτε ηθελημένα είτε όχι). Στην επόμενη παράγραφο περιγράφεται ο τρόπος με τον οποίον το πρωτόκολλο χειρίζεται τέτοιες καταστάσεις.

4.5.1 Συνδέσεις νέων κόμβων και σταθεροποίηση

Για να είναι δυνατή η σωστή εκτέλεση των ερωτημάτων σε περιπτώσεις όπου το σύνολο των κόμβων του δικτύου αλλάζει, το πρωτόκολλο θα πρέπει να μεριμνά για την σωστή ανανέωση των ID των successor των κόμβων του δικτύου. Αυτό επιτυγχάνεται χρησιμοποιώντας ένα πρωτόκολλο σταθεροποίησης το οποίο εκτελείται περιοδικά στο παρασκήνιο σε κάθε κόμβο και με αυτόν τον τρόπο ανανεώνονται οι πίνακες finger και οι δείκτες σε successor.

Στο σχήμα 4.8 φαίνονται σε μορφή ψευδοκώδικα η διαδικασίες που καλούνται κατά την σύνδεση ενός νέου κόμβου και κατά την φάση της σταθεροποίησης. Όταν ο κόμβος n ξεκινάει, καλεί την n.join(n'), με n' οποιονδήποτε υπάρχοντα κόμβο του δικτύου. Στην περίπτωση που δημιουργείται ένα νέο δίκτυο Chord καλείται η n.create(). Η συνάρτηση join() ζητά από τον κόμβο n' να βρεί τον άμεσο successor του κόμβου n.

Κάθε κόμβος καλεί την stabilize() κατά περιόδους για να ενημερωθεί για νέους κόμβους που έχουν συνδεθεί στο δίκτυο. Κάθε φορά που ο κόμβος n καλεί την stabilize(), ζητάει από τον successor να τον πληροφορήσει για τον predecessor p του successor και αποφασίζει αν ο p θα πρέπει να είναι ο successor του κόμβου n. Αυτό θα ισχύει στην περίπτωση όπου ο κόμβος p έχει συνδεθεί πρόσφατα με το δίκτυο. Επιπλέον η stabilize() ενημερώνει τον κόμβο successor του κόμβου n για την ύπαρξη του n, δίνοντας την δυνατότητα στον successor να αλλάξει τον predecessor του σε n. Αυτό γίνεται στην περίπτωση όπου ο successor δεν γνωρίζει κάποιον κοντινότερο predecessor από τον n.

Κάθε κόμβος περιοδικά καλεί την fix_fingers έτσι ώστε να διασφαλίσει ότι οι καταχωρήσεις στον πίνακα finger είναι σωστές. Αυτός είναι και ο τρόπος με τον οποίο οι νέοι κόμβοι αρχικοποιούν τους πίνακες finger που διατηρούν και επιπλέον ο τρόπος με τον οποίον οι υπάρχοντες κόμβοι ενσωματώνουν τους νεοεισερχόμενους κόμβους στους πίνακες τους. Επιπλέον, ο κάθε κόμβος καλεί περιοδικά και την check_predecessor για να αρχικοποιήσει τον δείκτη σε predecessor του κόμβου στην περίπτωση που αποτύχει η n.predecessor. Κατά αυτόν τον τρόπο δίνεται η δυνατότητα να δεχτεί έναν νέο predecessor στην notify.

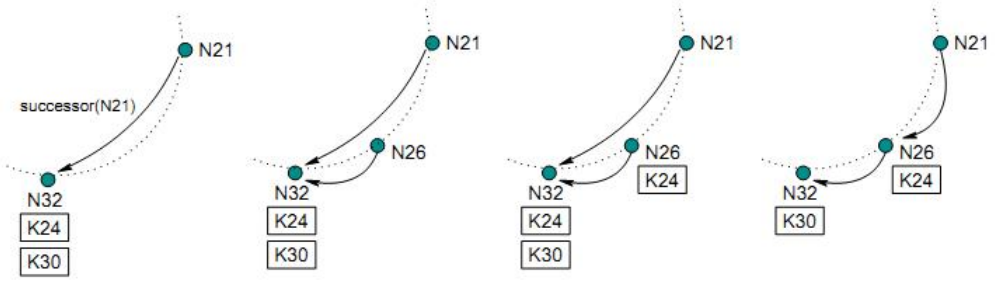
Σαν ένα απλό παράδειγμα, ας υποθέσουμε ότι ο κόμβος n συνδέεται στο δίκτυο και το ID του υπάρχει μεταξύ των κόμβων n^p και n^s. Κατά την κλήση της join(), ο n δηλώνει

```
// create a new Chord ring.
n.create()
    predecessor = nil;
    successor = n;

// join a Chord ring containing node n'.
n.join(n')
    predecessor = nil;
    successor = n'.find_successor(n);

// called periodically. verifies n's immediate
// successor, and tells the successor about n.
n.stabilize()
    x = successor.predecessor;
    if (x ∈ (n, successor))
        successor = x;
    successor.notify(n);

// n' thinks it might be our predecessor.
n.notify(n')
    if (predecessor is nil or n' ∈ (predecessor, n))
        predecessor = n';

// called periodically. refreshes finger table entries.
// next stores the index of the next finger to fix.
n.fix_fingers()
    next = next + 1;
    if (next > m)
        next = 1;
    finger[next] = find_successor(n + 2^(next-1));

// called periodically. checks whether predecessor has failed.
n.check_predecessor()
    if (predecessor has failed)
        predecessor = nil;
```

Σχήμα 4.8: Ψευδοκώδικας της διαδικασίας σταθεροποίησης

τον n^s ως τον successor του. Ο κόμβος n^s, όταν ενημερωθεί για τον κόμβο n δηλώνει τον κόμβο n σαν predecessor του. Στην επόμενη κλήση της stabilize() από τον n^p, ζητά από τον n^s για τον predecessor του (που τώρα είναι ο n). Στη συνέχεια ο n^p ορίζει τον n σαν successor του. Τελικά ο κόμβος n^p ενημερώνει τον κόμβο n^p και ο κόμβος n ορίζει τον n^p σαν predecessor του. Σε αυτό το σημείο όλοι οι δείκτες σε predecessor και successor είναι σωστά ορισμένοι. Σε αυτό το βήμα της διαδικασίας, ο κόμβος n^s είναι προσβάσιμος από τον n^p χρησιμοποιώντας δείκτες σε successor, γεγογός που σημαίνει οτι οι αναζητήσεις κατά την διάρκεια μιας σύνδεσης ενός νέου κόμβου δεν διαταράσσονται. Στο σχήμα 4.9 φαίνεται η διαδικασία μιας νέας σύνδεσης στο δίκτυο όταν το ID του n είναι 26 και τα ID των n^s και n^s είναι 21 και 32 αντίστοιχα.

Όταν έχει τελειώσει η διαδικασία ανανέωσης των δεικτών σε successor, οι κλήσεις στην find_successor θα αντικατοπτρίζουν τον νέο κόμβο. Κόμβοι οι οποίοι συνδέθηκαν στο δίκτυο πρόσφατα και δεν έχουν ακόμα αποτυπωθεί στους πίνακες finger των άλλων κόμβων πιθανόν να δημιουργήσουν μια λανθασμένη λειτουργία της find_successor(). Παρ' όλ' αυτά, ο βρόχος στον αλγόριθμο αναζήτησης θα ακολουθήσει τον δείκτη σε successor (finger[1]) διαμέσου των κόμβων που συνδέθηκαν στο δίκτυο πρόσφατα εως ότου βρεθεί ο σωστός predecessor. Τελικά η fix_fingers() θα ενημερώσει τις καταχωρήσεις στους πίνακες φινγερ αποφεύγοντας



Σχήμα 4.9: Ένα παράδειγμα μιας διαδικασίας σύνδεσης

τυχόν σειριακές αναζητήσεις.

Το επόμενο αποτέλεσμα, δείχνει ότι η ασταθής κατάσταση που οφείλεται στις ταυτόχρονες συνδέσεις νέων κόμβων χαρακτηρίζεται ως μεταβατική.

Θεώρημα 4.3: Στην περίπτωση νέων συνδέσεων που συμβαίνουν σε συνδυασμό με την διαδικασία σταθεροποίησης αποδεικνύεται ότι άμεσα μετά την τελευταία σύνδεση ενός κόμβου στο δίκτυο, οι δείκτες σε successor θα δημιουργήσουν έναν δακτύλιο με όλους τους κόμβους του δικτύου.

Πιο συγκεκριμένα, από το ανωτέρω θεώρημα ισχύει ότι μετά από λίγο χρόνο οποιοσδήποτε κόμβος θα έχει την δυνατότητα να προσπελάσει οποιονδήποτε άλλον κόμβο, ακολουθώντας δείκτες σε successor.

Η συγκεκριμένη μέθοδος σταθεροποίησης εγγυάται την προσθήκη νέων κόμβων στον δακτύλιο Chord με τέτοιο τρόπο ώστε να διατηρεί την προσβασιμότητα μεταξύ των υπαρχόντων κόμβων ακόμα και στην περίπτωση ταυτόχρονων νέων συνδέσεων ή στην περίπτωση χαμένων μηνυμάτων ή παραλαβή μηνυμάτων με λανθασμένη σειρά. Το πρωτόκολλο σταθεροποίησης δεν μπορεί από μόνο του να διορθώσει την εγκυρότητα ενός συστήματος Chord το οποίο είτε έχει χωριστεί σε πολλούς δακτύλιους ασύνδετους μεταξύ τους είτε ένας δακτύλιος εμφανίζεται να έχει πολλαπλές καταχωρίσεις στο σύνολο των ID. Οι συγκεκριμένες βέβαια περιπτώσεις χαρακτηρίζονται ως απίθανες αφού δεν μπορούν να παραχθούν από μια συνηθισμένη σειρά συνεχόμενων συνδέσεων. Στην περίπτωση που εμφανιστούν τέτοιες περιπτώσεις μπορούν να αναγνωριστούν και να διορθωθούν μέσω της περιοδικής δειγματοληψίας της τοπολογίας του δακτυλίου.

4.5.2 Επίδραση των νέων συνδέσεων στις αναζητήσεις

Σε αυτήν την παράγραφο, περιγράφονται οι επιδράσεις των νέων συνδέσεων στο δίκτυο απέναντι στις αναζητήσεις. Αρχικά θα αναφερθούμε στην ορθότητα των αναζητήσεων. Στην περίπτωση που οι κόμβοι που συνδέονται επηρεάζουν μια περιοχή του δακτυλίου Chord τότε μια αναζήτηση που θα συμβεί πριν από την διαδικασία της σταθεροποίησης μπορεί να έχει τρεις συγκεκριμένες συμπεριφορές. Η πιο συνηθισμένη κατάσταση είναι ότι οι καταχωρήσεις στους πίνακες finger είναι κατά βάση ενημερωμένες οπότε η αναζήτηση επιστρέφει τον σωστό successor σε O(logN) βήματα. Η δεύτερη περίπτωση είναι οι δείκτες σε successor να δείχνουν

στην σωστή θέση ενώ αντίστοιχα τα fingers να είναι ανακριβή. Σε αυτήν την περίπτωση οι αναζητήσεις επιστρέφουν σωστά αποτελέσματα αλλά υπολείπονται σε ταχύτητα. Στην τελευταία περίπτωση, είτε οι κόμβοι που βρίσκονται στην περιοχή του δακτυλίου που επηρεάζεται θα έχουν λανθασμένους δείκτες σε successor είτε τα κλειδιά δεν θα έχουν διαμοιραστεί σωστά στους κόμβους που συνδέθηκαν πρόσφατα στο δίκτυο με αποτέλεσμα μια λανθασμένη αναζήτηση. Το λογισμικό ανώτερου επιπέδου που χρησιμοποιεί το πρωτόκολλο Chord θα διαπιστώσει ότι τα ζητούμενα δεδομένα δεν βρέθηκαν και θα επιχειρήσει μια νέα αναζήτηση ύστερα από μια σύντομη παύση. Η συγκεκριμένη παύση είναι μικρής διάρκειας δεδομένου ότι η διαδικασία της σταθεροποίησης δεν διαρκεί πολύ.

Στη συνέχεια γίνεται μια ανάλυση από άποψης απόδοσης. Όταν πλέον η διαδικασία της σταθεροποίησης έχει τελειώσει, οι νέοι κόμβοι δεν θα έχουν κανένα άλλο αντίκτυπο πέραν της αύξησης του N στον χρόνο αναζήτησης O(logN). Στην περίπτωση που η διαδικασία της σταθεροποίησης δεν έχει ακόμα ολοκληρωθεί, οι πίνακες finger των υπαρχόντων κόμβων μπορεί να μην αντικατοπτρίζουν την πραγματικότητα σε σχέση με τους νέους κόμβους. Η δυνατότητα των καταχωρήσεων στους πίνακες finger να μπορούν να μεταφέρουν τα ερωτήματα ανάμεσα στους κόμβους του δακτυλίου δεν εξαρτάται άμεσα από τον κόμβο που έχει σαν στόχο το ερώτημα. Όπως αναφέρθηκε εξαρτάται μόνο από την απόσταση στον χώρο των ID. Για αυτόν τον λόγο, το γεγονός ότι οι καταχωρήσεις στους πίνακες finger μπορεί να μην αντικατοπτρίζουν τους νέους κόμβους δεν επηρεάζει απαραίτητα την ταχύτητα αναζήτησης. Ο βασικός τρόπος μέσω του οποίου μπορεί οι νέοι κόμβοι να επηρεάσουν την ταχύτητα αναζήτησης είναι αν αυτοί βρίσκονται μεταξύ του predecessor του αναζητούμενου κόμβου και του κόμβου που αναζητείτε. Σε αυτήν την περίπτωση θα πρέπει η αναζήτηση να προωθηθεί διαμέσου των ενδιάμεσων κόμβων κατά έναν κόμβο τη φορά. Παρόλ' αυτά, εκτός από την περίπτωση που ένας τεράστιος αριθμός κόμβων συνδεθεί με το δίκτυο ταυτόχρονα, ο αριθμός των κόμβων που θα βρεθεί ανάμεσα σε δυο υπάρχοντες κόμβους θα είναι πιθανότατα αρκετά μικρός, οπότε και θα έχει ελάχιστη επίδραση στην αναζήτηση. Γενικά, θα καλούμε έναν δακτύλιο Chord ευσταθή στην περίπτωση που οι δείκτες σε successor και finger δείχνουν στις σωστές θέσεις.

Θεώρημα 4.4: Στην περίπτωση που υπάρχει ένα ευσταθές δίκτυο αποτελούμενο από N κόμβους με δείκτες σε finger που δείχνουν στις σωστές θέσεις και ένα άλλο σύνολο από το πολύ N κόμβους συνδεθεί με το δίκτυο και όλοι οι δείκτες σε successor είναι ακριβείς (όχι απαραίτητα και οι δείκτες σε finger) τότε οι αναζητήσεις θα διαρκούν για O(logN) με μεγάλη πιθανότητα.

Απόδειξη: Το αρχικό σύνολο από finger θα οδηγήσει το ερώτημα στον παλιό predecessor του σωστού κόμβου σε O(logN) βήματα. Επιπλέον, με μεγάλη πιθανότητα, το πολύ O(logN) στον αριθμό κόμβοι θα τοποθετηθούν ανάμεσα σε δυο υπάρχοντες κόμβους. Κατά αυτόν τον τρόπο μόνο O(logN) στον αριθμό κόμβοι θα πρέπει να αναζητηθούν στους δείκτες σε successor έτσι ώστε να μετακινηθούμε μεταξύ του παλιού και του νέου predecessor.

Γενικότερα, εφόσον κάθε φορά ο χρόνος που είναι απαραίτητος για να ενημερωθούν τα finger είναι μικρότερος από τον χρόνο που χρειάζεται για να διπλασιαστεί σε πλήθος τον δίκτυο, η αναζητήσεις θα συνεχίσουν να διαρκούν έως O(logN) βήματα. Αυτή η προϋπόθεση, μπορεί να ισχύει όταν γίνονται συνεχώς αναζητήσεις έτσι ώστε να ανανεώνονται οι πίνακες finger.

Ως αποτέλεσμα οι αναζητήσεις θα είναι επιτυχείς εφόσον συμβαίνουν τουλάχιστον Ω(log2N) γύροι σταθεροποίησης μεταξύ του χρόνου που θα γίνουν N στον αριθμό νέες συνδέσεις.

4.5.3 Αστοχία και αναπαρωγή της πληροφορίας

Η ορθότητα του πρωτοκόλλου Chord βασίζεται στο γεγονός ότι κάθε κόμβος γνωρίζει ποιος κόμβος είναι ο successor του. Υπάρχει όμως και η περίπτωση αυτή η προϋπόθεση να μην ισχύει στην περίπτωση αστοχίας κόμβων του συστήματος. Για παράδειγμα, όπως φαίνεται στο σχήμα 4.5, στην περίπτωση όπου οι κόμβοι 41,21,32 αστοχήσουν ταυτόχρονα, τότε ο κόμβος 8 δεν θα γνωρίζει οτι πλέον ο κόμβος 38 θα είναι ο successor του εφόσον δεν υπάρχει κάποιο finger που να δείχνει στον κόμβο 38. Στην περίπτωση ενός λανθασμένου successor οδηγούμαστε σε μια λανθασμένη αναζήτηση. Ας θεωρήσουμε ένα ερώτημα για το κλειδί 30 που κατατέθηκε από τον κόμβο 8. Ο κόμβος 8 θα επιστρέψει ως απάντηση τον κόμβο 42 ο οποίος είναι ο πρώτος κόμβος που γνωρίζει βάσει του πίνακα finger που διατηρεί. Η απάντηση είναι λανθασμένη δεδομένου οτι ο σωστός successor του είναι ο κόμβος 38.

Το πρωτόκολλο Chord για να αυξήσει την ανοχή διατηρεί μια λίστα σε successor μεγέθους r. Η λίστα αυτή περιέχει του πρώτους r successor. Στην περίπτωση που ο πρώτος successor ενός κόμβου δεν απαντά, ο ίδιος ο κόμβος μπορεί να κάνει την εναλλαγή με την δεύτερη καταχώρηση στην λίστα που διατηρεί. Θα πρέπει όλοι οι r στον αριθμό successor να αστοχήσουν ταυτόχρονα για να παρακωλλυθεί η λειτουργία του δακτυλίου Chord. Η πιθανότητα για ένα τέτοιο ενδεχόμενο είναι αμελητέα ακόμα και για λογικές τιμές του r. Αν υποθέσουμε ότι κάθε κόμβος αστοχεί ανεξάρτητα με πιθανότητα p τότε η πιθανότητα να αστοχήσουν όλοι οι r successor ταυτόχρονα είναι p^r. Η αύξηση του r αυξάνει την ανοχή του συστήματος σε σφάλματα.

Η διατήρηση μιας λίστας σε successor απαιτεί ελάχιστες αλλαγές στη λειτουργία που απεικονίζεται σε μορφή ψευδοκώδικα στα σχήματα 4.6 και 4.8. Στο σχήμα 4.8 φαίνεται μια τροποποιημένη εκδοχή της διαδικασίας σταβιλιζε η οποία και διατηρεί μια λίστα σε successor. Οι λίστες σε successor μπορούν να σταθεροποιούνται ως εξής: ο κόμβος n αντιπαραθέτει τη λίστα του με αυτήν του s, την αντιγράφει, αφαιρεί την τελευταία καταχώρηση και βάζει στη θέση της τον κόμβο s. Στην περίπτωση που ο κόμβος ν παρατηρήσει οτι ο successor του έχει αστοχήσει, τον αντικαθιστά με την πρώτη ενεργή καταχώρηση που βρίσκεται στην λίστα σε successor που διατηρεί και διασταυρώνει την λίστα του με αυτή του νέου successor του. Σε αυτό το σημείο ο κόμβος n μπορεί να εξυπηρετεί αναζητήσεις για τα κλειδιά για τα οποία ο κόμβος που έχει αστοχήσει ήταν ο successor του νέου successor. Με την πάροδο του χρόνου οι διαδικασίες fix_fingers και stabilize θα διορθώνουν τις εγγραφές των πινάκων finger και των λιστών σε successor που δείχνουν σε κόμβο ο οποίος έχει αστοχήσει.

Μια τροποποιημένη έκδοση της closest_preceding_node όπως φαίνεται στο σχημα 4.6 αναζητά όχι μόνο στον πίνακα finger αλλά και στην λίστα σε successor για τον άμεσο predecessor του id. Επιπλέον ο συγκεκριμένος ψευδοκώδικας θα πρέπει να βελτιωθεί έτσι ώστε να διαειρίζεται τυχόν αστοχίες κόμβων. Στην περίπτωση που ένας κόμβος αστοχήσει κατά την διάρκεια της εκτέλεσης της find_successor τότε η αναζήτηση συνεχίζεται μετά από ένα timeout δοκιμά-

ζοντας στον αμέσως κοντινότερο predecessor μεταξύ των κόμβων που υπάρχουν στον πίνακα finger και στη λίστα σε successor.

Τα ακόλουθα αποτελέσματα προσδιορίζουν την ανοχή του πρωτοκόλλου Chord δείχνοντας οτι ούτε η επιτυχία ούτε η απόδοση των αναζητήσεων μπορεί να επηρεαστεί από μαζικές και ταυτόχρονες αστοχίες. Και τα δυο επόμενα θεωρήματα υποθέτουν ότι η λίστα σε successor πρέπει να έχει μήκος ίσο με r=W(logN).

Θεώρημα 4.5: Στην περίπτωση που χρησιμοποιηθεί μια λίστα σε successor μήκους r= W(logN) σε είναι δίκτυο που είναι αρχικά ευσταθές και στη συνέχεια κάθε κόμβος αστοχεί με πιθανότητα ίση με 0.5 τότε με αρκετά μεγάλη πιθανότητα η find_successor επιστρέφει τον κοντινότερο ενεργό successor του κλείδιού του ερωτήματος.

Απόδειξη: Πρίν από την αστοχία κάποιου κόμβου, κάθε κόμβος γνώριζε για τους r άμεσους successor του. Η πιθανότητα όλοι αυτόι οι successor να αστοχήσουν είναι ίση με 0.5r και έτσι με αρκετά μεγάλη πιθανότητα κάθε κόμβος γνωρίζει τον κοντινότερο ενεργό successor του. Όπως έχει αναφερθεί και σε προηγούμενη παράγραφο, στην περίπτωση όπου ισχύει η υπόθεση ότι κάθε κόμβος γνωρίζει τον κοντινότερο του successor τότε όλα τα ερωτήματα δρομολογούνται σωστά. Αυτό βέβαια συμβαίνει αφού κάθε κόμβος εκτός του κοντινότερου successor του ερωτήματος έχει τουλάχιστον έναν καλύτερο κόμβο στον οποίο μπορεί να προωθήσει το ερώτημα.

Θεώρημα 4.6: Σε ένα δίκτυο το οποίο είναι αρχικά σταθερό, αν κάθε κόμβος αστοχεί με πιθανότητα 0.5 τότε ο αναμενόμενος χρόνος για την εκτέλεση της find_successor είναι O(logN). *Απόδειξη:* Η απόδειξη περιλαμβάνεται στην τεχνική αναφορά.

Κάτω από συγκεκριμένες συνθήκες τα ανωτέρω θεωρήματα μπορούν να εφαρμοστούν είτε σε ηθελημένες αστοχίες κόμβων είτε σε απρόβλεπτες αστοχίες. Υπάρχει περίπτωση κάποιος εχθρός του συστήματος να οδηγήσει μερικούς από τους κόμβους σε αστοχία αλλά σε καμία περίπτωση δεν μπορεί να έχει έλεγχο στο σύνολο των κόμβων. Για παράδειγμα ένας εχθρός του συστήματος μπορεί να είναι ικανός να ελέγξει τους κόμβους από μια συγκεκριμένη γεωγραφική περιοχή ή κόμβους που επικοινωνούν μέσω μιας συγκεκριμένης σύνδεσης δικτύου ή ακόμαι και κόμβους που έχουν ένα συγκεκριμένο πρόθεμα στην διεύθυνση IP τους. Όπως έχουμε αναφέρει ξανά, δεδομένου ότι τα ID παράγονται από τη χρήση συναρτήσεων hash πάνω στις διευθύνσεις IP των κόμβων, αυτά θα είναι ουσιαστικά τυχαία όπως ακριβώς και στις περιπτώσεις αστοχίας που έχουν αναφερθεί πιο πριν.

Ο μηχανισμός διατήρησης των λιστών σε successor επιπλέον βοηθά και λογισμικό ανώτερων επιπέδων να αναπαράγει τα δεδομένα. Σε μια τυπική εφαρμογή που χρησιμοποιεί το πρωτόκολλο Chord μπορεί να αποθηκεύει αντίγραφα των δεδομένων που σχετίζονται με το κλειδί στους k κόμβους που έπονται του κλειδιού. Το γεγονός οτι το πρωτόκολλο Chord διατηρεί μια λίστα με τους r successor του σημαίνει ότι έχει την δυνατότητα να ενημερώνει τα ανώτερα στρώματα λογισμικού στην περίπτωση όπου συνδέονται και αποσυνδέονται άλλοι κόμβοι και κατ επέκταση αυτά να μπορούν να προωθούν τα δεδομένα στα νέα αντίγραφα.

Προγραμματισμένες αποσυνδέσεις κόμβων

Δεδομένου ότι το πρωτόκολλο Chord είναι ανθεκτικό σε αστοχίες, στην περίπτωση που κάποιος κόμβος έχει μια προγραμματισμένη αποσύνδεση από το δίκτυο, τότε αυτή η αποσύνδεση μπορεί κάλλιστα να χειρισθεί σαν μια αστοχία. Παρόλ΄ αυτά δυο ακόμα βελτιώσεις μπορούν να βελτιώσουν την απόδοση στην περίπτωση που υπάρχουν προγραμματισμένες αποσυνδέσεις κόμβων. Αρχικά ο κόμβος n ο οποίος είναι προγραμματισμένος να αποσυνδεθεί, μπορεί να μεταφέρει τα κλειδιά του στον successor του πριν να αποσυνδεθεί από το δίκτυο. Επιπλέον, ο κόμβος n μπορεί να ενημερώσει τον predecessor p και τον successor s πριν από την αποσύνδεσή του. Με τη σειρά του ο κόμβος p θα αφαιρέσει τον κόμβο n από τη λίστα με τους successor του και θα προσθέσει την τελευταία καταχώρηση της λίστας από successor του κόμβου n στη λίστα του. Κατά παρόμοιο τρόπο, ο κόμβος s θα αντικαταστήσει τον predecessor του με τον predecessor του κόμβου n. Αυτό συμβαίνει με την προϋπόθεση ότι ο κόμβος n αποστέλλει τον predecessor του στον κόμβο s και τον τελευταίο κόμβο στην λίστα με τους successor που διατηρεί στον κόμβο p.

4.5.4 Μια πιο ρεαλιστική ανάλυση

Η ανάλυση που γίνεται πιο πάνω, μας δίνει μια καλύτερη πληροφόρηση για τη λειτουργία του συστήματος Chord. Παρόλ΄ αυτά χαρακτηρίζεται ως ανεπαρκής στην πράξη. Τα θεωρήματα τα οποία έχουν αποδειχθεί πιο πριν υποθέτουν ότι ο δακτύλιος Chord ξεκινά από μια σταθερή κατάσταση και στη συνέχεια δέχεται νέες συνδέσεις ή και αστοχίες. Στην πραγματικότητα όμως, ένας δακτύλιος Chord δεν θα βρίσκεται ποτέ σε μια σταθερή κατάσταση. Αντιθέτως νέες συνδέσεις και νέες αποσυνδέσεις θα συμβαίνουν συνεχώς και ταυτόχρονα με την εκτέλεση του αλγόριθμου σταθεροποίησης. Έτσι, ο δακτύλιος δεν θα έχει τον απαραίτητο χρόνο να σταθεροποιηθεί πριν συμβούν περαιτέρω αλλαγές. Οι αλγόριθμοι του πρωτοκόλλου Chord μπορούν να αναλυθούν περισσότερο έχοντας θέσει και αυτές τις παραμέτρους. Επιπλέον, διάφορες εργασίες δείχνουν ότι στην περίπτωση που το πρωτόκολλο σταθεροποίησης εκτελείται με έναν συγκεκριμένο ρυθμό (εξαρτώμενο από το ρυθμό με τον οποίο συνδέονται και αποσυνδέονται οι κόμβοι) τότε ο δακτύλιος Chord μπορεί να συνεχίσει να είναι σε μια κατάσταση “εν μέρει” σταθερότητας στην οποία οι αναζητήσεις είναι ταχύτατες και έγκυρες.

Κεφάλαιο 5

Το δίκτυο Quantum

Στο κεφάλαιο αυτό περιγράφεται το Quantum, ένα πρωτόκολλο δικτύου ομότιμων κόμβων που στοχεύει στην εκτέλεση αποδοτικών κατανεμημένων αριθμητικών υπολογισμών μεταξύ των κόμβων του δικτύου.

Θα εξετάσουμε αναλυτικά τη δομή και τα χαρακτηριστικά του και θα αναφερθούμε στους αλγορίθμους που βρίσκονται πίσω από το Quantum. Τέλος, στο κεφάλαιο αυτό, υπάρχει ενότητα που αφορά στις χρήσεις του πρωτοκόλλου.

5.1 Εισαγωγή

Το Quantum είναι ένα δίκτυο ομότιμων κόμβων που βασίζεται στο Chord για την εκτέλεση κατανεμημένων υπολογισμών μεταξύ αυτόνομων agents (πρακτόρων). Βασικό χαρακτηριστικό του Quantum είναι η προστασία της ιδιωτικότητας των κόμβων που συμμετέχουν σε κάθε κατανεμημένο υπολογισμό. Επιπλέον το Quantum υποστηρίζει τοπολογίες που σχηματίζουν μεγάλο πλήθος agents που επικοινωνούν μέσω του διαδικτύου και παρουσιάζει ανοχή σε προσθήκες/αφαιρέσεις κόμβων.

Πιο συγκεκριμένα, η δικτυακή δομή του Quantum είναι παρόμοια με αυτή του Chord όσων αφορά το δακτύλιο και τους fingers ενώ υπάρχουν ορισμένες διαφορές σε κάποιους από τους αλγορίθμους για τη σταθεροποίηση του δικτύου. Ωστόσο, στο Quantum δεν υπάρχει το κομμάτι της αποθήκευσης δεδομένων που υποστηρίζεται από το Chord. Το Quantum υποστηρίζει επίσης την εκτέλεση υπολογισμών στους οποίους μεσολαβούν όλοι οι κόμβοι του δικτύου σε χρόνο που είναι λογαριθμητικής πολυπλοκότητας σε σχέση με το πλήθος των κόμβων στο δίκτυο χωρίς να καταπονούνται δικτυακά και επεξεργαστικά οι κόμβοι.

5.2 Δικτυακά χαρακτηριστικά

Η δικτυακή αρχιτεκτονική του Quantum περιλαμβάνει βασικά χαρακτηριστικά του Chord, όπως την ένταξη των κόμβων σε δακτύλιο, τη διατήρηση fingers, τη σταθεροποίηση των αναφορών καθώς επίσης και την είσοδο - έξοδο κόμβων από το δίκτυο. Ενδιαφέρον παρουσιάζει ο αλγόριθμος αναζήτησης ενός κλειδιού. Ένας κόμβος που θα λάβει/εκκινήσει ένα αίτημα

αναζήτησης θα προωθήσει το αίτημα στην μεγαλύτερη αναφορά του που είναι μικρότερη από το ζητούμενο κλειδί. Σε αντίθεση με πρωτόκολλα P2P δικτύων, το Quantum δεν ασχολείται με αποθήκευση δεδομένων και τα κλειδιά χρησιμοποιούνται μόνο για τη δρομολόγηση των πακέτων.

Το Quantum, όπως και το Chord, δομείται στο χαμηλότερο στάδιο με ένα νοητό δακτύλιο πάνω στον οποίο τοποθετούνται όλοι οι κόμβοι/αγεντς του δικτύου. Η σειρά με την οποία θα τοποθετηθούν καθορίζεται από το κλειδί (ID) των κόμβων. Το ID ενός κόμβου είναι ένα μοναδικό χαρακτηριστικό του και στην περίπτωση του Quantum υπολογίζεται βάση του SHA-1 hash του συνδυασμού IP:Port (κάθε agent έχει ένα port ανοιχτό στο οποίο "ακούει" διάφορα αιτήματα από άλλους agents). Έτσι, με διαδικασίες που περιγράφονται πιο κάτω, οι κόμβοι τείνουν να τοποθετηθούν γύρω από το νοητό αυτό δακτύλιο με αύξουσα σειρά mod(n), με n το μέγιστο πλήθος κόμβων που μπορεί να υποστηρίξει το δίκτυο. Σύμφωνα με αυτή την αρχιτεκτονική, κάθε agent διατηρεί αναφορές στον αμέσως επόμενο του στον δακτύλιο (successor, αναφέρεται succ()), καθώς επίσης και για τον αμέσως προηγούμενό του (predecessor, αναφέρεται pred()). Η δικτυακή αυτή οργάνωση αποτελεί μια βασική δομή που μπορεί να διατηρήσει ένα δίκτυο ομότιμων κόμβων.

Συχνά ένα ομότιμο δίκτυο αποτελείται από ένα πολύ μεγάλο αριθμό κόμβων. Σε μια τέτοια περίπτωση η δικτυακή αρχιτεκτονική που αναφέρθηκε πιο πάνω, παρότι αξιόπιστη, δεν είναι επαρκής για την επεκτασιμότητα του δικτύου κάτω από τέτοιες συνθήκες και αυτό διότι ο χρόνος ταξιδιού ενός πακέτου γύρω από το δακτύλιο είναι ανάλογος με τον αριθμό των κόμβων. Τη λύση σε αυτό το πρόβλημα προσφέρει η δικτυακή οργάνωση των fingers του Chord, μέσω της οποίας επιτυγχάνουμε την επιθυμητή επεκτασιμότητα. Σύμφωνα με αυτή, κάθε agent διατηρεί έναν πίνακα ο οποίος περιέχει τις διευθύνσεις των κόμβων με εκθετικά αυξανόμενο ID από τον αρχικό. Πιο συγκεκριμένα, ο κόμβος με κλειδί S θα διατηρήσει αναφορές για τους κόμβους με κλειδιά $S + 2k$, με $k >= 1$ και με όριο το μέγιστο αριθμό bits του κλειδιού (πχ στην περίπτωση του SHA-1 είναι 160). Αξίζει να σημειωθεί ότι για $k = 0$ η αναφορά ισοδυναμεί με τον successor. Μια λεπτομερής ανάλυση των fingers έχει προηγηθεί στο κεφάλαιο του Chord.

5.3 Αλγόριθμοι δικτύου

Στην ενότητα αυτή παρουσιάζονται οι αλγόριθμοι που χρησιμοποιούνται στη λειτουργία του Quantum: εισαγωγή κόμβου στο δίκτυο και έξοδος, σταθεροποίηση του δικτύου και προσαρμογή στις αλλαγές, αναζήτηση κλειδιού καθώς επίσης και μέθοδοι broadcast και κατανεμημένων υπολογισμών.

5.3.1 Είσοδος και έξοδος κόμβου

Στην παράγραφο αυτή παρουσιάζεται η διαδικασία που ακολουθείται ώστε να εισέλθει ένας αγεντ στο ομότιμο δίκτυο του Quantum. Επίσης προτείνεται μια διαδικασία για την ανάκαμψη του δικτύου έπειτα από αποχώρηση ενός κόμβου. Για τεχνικούς λόγους, οι παραπάνω διαδικασίες διαφέρουν ελαφρά από τις αντίστοιχες του Chord ενώ σε αντίθεση με το τελευταίο δεν

προβλέπεται οικειοθελής αποχώρηση κόμβου. Μετά την έξοδο ενός agent από το Quantum, η κατάσταση αντιμετωπίζεται όπως θα γινόταν και στην περίπτωση της ξαφνικής αποτυχίας ενός κόμβου.

Είσοδος

Απαραίτητη προϋπόθεση είναι ο προς εισαγωγή agent S να γνωρίζει τα στοιχεία επικοινωνίας (διεύθυνση IP και Port) ενός οποιουδήποτε άλλου κόμβου M που ανήκει στο δίκτυο. Ο S αποστέλλει στον M ένα πακέτο αίτησης εισχώρησης στο δίκτυο μαζί με κάποια χαρακτηριστικά του agent, όπως η θύρα (port) στην οποία λαμβάνει μηνύματα. Εφόσον ο M λάβει το πακέτο, θα στείλει πίσω ένα πακέτο "δοκιμής" στη θύρα που ακούει ο S, στο οποίο ο S είναι υποχρεωμένος να απαντήσει, επιβεβαιώνοντας έτσι τη δυνατότητά του να λάβει μηνύματα. Ο S θέτει τον M ως successor του, δεδομένου ότι είναι ο μοναδικός agent που γνωρίζει και τον ενημερώνει σχετικά. Εάν ο M δεν έχει predecessor ή εάν το ID του pred(M) είναι μικρότερο από το ID του S, τότε τον δέχεται ως predecessor και η διαδικασία εισαγωγής του S έχει ολοκληρωθεί. Αξίζει να σημειωθεί ότι σε αυτό το σημείο δεν έχει επέλθει πλήρης σταθεροποίηση στο δίκτυο, μιας και κανένας άλλος κόμβος πέραν του M δε γνωρίζει την ύπαρξη του S. Στην επόμενη ενότητα αναλύεται ο τρόπος με τον οποίο το δίκτυο σταθεροποιείται.

Έξοδος

Όπως προαναφέρθηκε, δεν υπάρχει κάποια συγκεκριμένη διαδικασία αποχώρησης κόμβου από το Quantum σε αντίθεση με το Chord. Ο λόγος είναι ότι στο Chord είναι απαραίτητο να υπάρχει αξιοπιστία όσων αφορά την αποθήκευση των δεδομένων, γεγονός που υπαγορεύει τη σωστή και έγκαιρη ενημέρωση των υπολοίπων κόμβων. Στο Quantum δε γίνεται χρήση της δυνατότητας αποθήκευσης δεδομένων στους κόμβους ενώ μετά την αποχώρηση ενός κόμβου το δίκτυο σταθεροποιείται σταδιακά με μεθόδους που περιγράφονται πιο κάτω.

5.3.2 Σταθεροποίηση και είδη

Ανά τακτά χρονικά διαστήματα, των οποίων η διάρκεια ορίζεται από το χειριστή του agent (έστω S), εκτελούνται 3 διαφορετικά ήδη σταθεροποίησης (stabilization), τα οποία περιγράφονται πιο κάτω. Θα υποθέσουμε ότι το μέγιστο μέγεθος των IDs του δικτύου είναι imax.

Σταθεροποίηση Successor

Η σταθεροποίηση του successor γίνεται με δύο τρόπους.

Σταθεροποίηση successor με αίτημα: Ο S εκκινεί ένα αίτημα εύρεσης του κλειδιού ID(S)+1 στο δίκτυο (η διαδικασία περιγράφεται σε επόμενη παράγραφο). Εάν η απάντηση M που θα δεχθεί από το δίκτυο αποτελεί έναν κόμβο πιο κοντά στον succ(S), ο successor θα αντικατασταθεί με τον M. Η μέθοδος αυτή είναι αντικειμενικά πιο γρήγορη από τη μέθοδο με έλεγχο predecessor, αλλά εκτελείται με πιο αργούς ρυθμούς δεδομένης της ενδεχόμενης καταπόνησης του δικτύου.

Σταθεροποίηση με έλεγχο predecessor: Ο S στέλνει ένα πακέτο στον succ(S), ζητώντας να του επιστρέψει τον Predecessor του (pred(succ(S) = K). Εάν ο K βρίσκεται μεταξύ του S και του συςς(Σ), ο S θα θέσει ως νέο του Successor τον K και θα ενημερώσει τον K σχετικά. Ο K έχει τη δυνατότητα να δεχθεί τον S ως Predecessor του αναλόγως με τη σχετική του θέση ως προς τον τρέχων του Predecessor. Ο τρόπος αυτός δεν αναφέρεται στο Chord και στο Quantum χρησιμοποιείται πολύ πιο συχνά από τη σταθεροποίηση με αίτημα μιας και δεν απαιτεί την κινητοποίηση περισσότερων των δύο κόμβων. Επιπρόσθετα, ένας τέτοιος τρόπος επιτρέπει στο δίκτυο Quantum σε γρήγορη σταθεροποίηση όταν αυτό έχει μικρό αριθμό κόμβων για τις ανάγκες των δοκιμών.

Σταθεροποίηση Διαδοχικών Κόμβων

Μετά την ένταξή του στο δίκτυο, ο S διαθέτει έναν πίνακα διαδοχικών κόμβων, που στο ξεκίνημα της λειτουργίας του περιέχει μόνο μια τιμή, τον successor του. Κατά τη διαδικασία της εν λόγω σταθεροποίησης, ο S επιλέγει τυχαία έναν κόμβο M από αυτό τον πίνακα και του αποστέλλει ένα ερώτημα, με το οποίο ζητάει να τον πληροφορήσει για τον succ(M). Στη συνέχεια, ο S θα προσθέσει τον succ(M) στον πίνακα των διαδοχικών κόμβων και θα αφαιρέσει (εάν το μέγεθός του υπερβαίνει αυτό που έχει ορίσει ο χειριστής του agent) τον κόμβο με τη μεγαλύτερη απόσταση από τον εαυτό του.

Σταθεροποίηση Fingers

Η λειτουργία αυτή επιτυγχάνεται αποκλειστικά με ερωτήματα στο δίκτυο. Ο N διατηρεί έναν πίνακα (όπως και στην περίπτωση των διαδοχικών) με τα fingers του. Ο πίνακας αυτός έχει σταθερό μέγεθος ίσο με τον αριθμό των bits των IDs του δικτύου. Κατά τη σταθεροποίηση αυτή, ο N θα επιλέξει τυχαία έναν ακέραιο k τέτοιος ώστε $0 <= k < imax$ και ξεκινάει ένα ερώτημα εύρεσης του $S + 2^{k+1}$. Ο N προσθέτει τον κόμβο - απάντηση (έστω M) στον πίνακα των fingers εάν στο διάστημα $(S + 2^k, S + 2^{(k+1)})$ δεν εμπεριέχεται άλλος κόμβος ή αν αυτός που εμπεριέχεται έχει μεγαλύτερο ID από τον M. Στην τελευταία περίπτωση ο ήδη υπάρχον αφαιρείται από τον πίνακα.

5.3.3 Αλγόριθμος Αναζήτησης

Η αναζήτηση στο δίκτυο αφορά την εύρεση ενός κόμβου που χαρακτηρίζεται από ένα συγκεκριμένο κλειδί. Εάν δεν υπάρχει κόμβος με το εν λόγω κλειδί, το δίκτυο επιστρέφει τον κόμβο με το αμέσως επόμενο διαθέσιμο κλειδί. Η παραπάνω διαδικασία είναι απαραίτητη κυρίως για την ύπαρξη των fingers στο δίκτυο και κατ' επέκταση για την επεκτασιμότητά του όσων αφορά των συνολικό αριθμό κόμβων. Είναι επίσης απαραίτητη για τη σταθεροποίηση του successor.

Το αίτημα αποτελείται από τα εξής πεδία: τον κόμβο S που ξεκίνησε το αίτημα, το κλειδί K, το οποίο ζητάμε καθώς επίσης και έναν ακέραιο ttl (time-to-live) ο οποίος περιορίζει τη διάρκεια ζωής του αιτήματος στο δίκτυο. Για την εύρεση του κλειδιού, το αίτημα διασχίζει

ένα μέρος του δίκτυο με τον παρακάτω τρόπο. Κάθε κόμβος M που λαμβάνει το αίτημα πραγματοποιεί τα παρακάτω βήματα:

Βήμα 1ο: Εάν το ttl είναι μηδέν, αγνοεί πλήρως το αίτημα και τερματίζει, αλλιώς το μειώνει κατά ένα. Ο λόγος ύπαρξης του ttl είναι για να αποφευχθούν καταστάσεις στις οποίες ένα πακέτο θα συνεχίζει να διασχίζει επ΄ αόριστον το δίκτυο (φαινόμενα λογικών deadlocks).

Βήμα 2ο: Ο κόμβος M πράττει ανάλογα με τις εξής περιπτώσεις:

1. Εάν το K είναι το κλειδί του κόμβου M, τότε αποστέλλει στον S ένα πακέτο επιβεβαίωσης εύρεσης του κλειδιού. Η αναζήτηση έχει ολοκληρωθεί.

2. Εάν το K βρίσκεται μεταξύ του pred(M) και M, τότε αποστέλλει και πάλι πακέτο επιβεβαίωσης στον S. Η αναζήτηση έχει ολοκληρωθεί.

3. Εάν το K βρίσκεται μεταξύ του M και succ(M), προωθεί το αίτημα στον succ(M).

4. Δημιουργεί μια ταξινομημένη κατά ID λίστα που περιέχει όλους τους κόμβους που βρίσκονται στο Finger Table. Βρίσκει τον κόμβο P της παραπάνω λίστας που έχει το αμέσως μεγαλύτερο ID από το K και προωθεί το αίτημα στον κόμβο P.

Σύμφωνα με τα παραπάνω, το μέγιστο πλήθος αναπηδήσεων που μπορεί να κάνει ένα αίτημα μέσα από το δίκτυο είναι log2n, όπου n ο μέγιστος αριθμός μοναδικών κλειδιών που μπορεί να υποστηρίξει το δίκτυο.

5.3.4 Broadcast

Στο Quantum, οποιοσδήποτε κόμβος έχει τη δυνατότητα αποστολής ενός πακέτου σε όλους τους υπόλοιπους κόμβους που συμμετέχουν στο δίκτυο (broadcast). Η διαδικασία αυτή χρησιμοποιείται σε συνδυασμό για την επίτευξη του κατανεμημένου υπολογισμού που θα αναλυθεί στη συνέχεια ενώ από μόνη της δε χρησιμεύει σε κάποιο μέρος της λειτουργικότητας του δικτύου. Παρόλ΄ αυτά υπάρχουν κάποιες χρήσιμες λειτουργίες που μπορούν να επιτευχθούν με ένα broadcast και παρουσιάζονται ενδεικτικά κάποιες πιο κάτω.

Broadcast για κλείσιμο agents

Ένα broadcast πακέτο με συγκεκριμένο αναγνωριστικό θα μπορούσε να χρησιμοποιηθεί για τον τερματισμό όλων των agents του δικτύου, εφόσον αυτό θα παραληφθεί από όλους. Κάτι τέτοιο δεν είναι πολύ χρήσιμο σε πραγματικές συνθήκες χρήσης του δικτύου αλλά αποτέλεσε απαραίτητη λειτουργικότητα κατά τις δοκιμές που έγιναν για την εργασία μιας και σε λειτουργικά που δεν υποστηρίζουν κλείσιμο ομάδας παραθύρων ήταν πολύ δύσκολος ο τερματισμός ενός μεγάλου αριθμού agents.

Broadcast εκτύπωσης στο παράθυρο του agent

Πολλές φορές κατά τις δοκιμές παρατηρήθηκαν προβλήματα επικοινωνίας μεταξύ των agents, πράγμα που αναπόφευκτα οδηγούσε στη διάσπαση του δικτύου. Για τον περιοδικό

έλεγχο της ενότητας του δικτύου χρησιμοποιήθηκε broadcast εκτύπωσης. Κάθε κόμβος που λαμβάνει το πακέτο αυτό εμφανίζει μια σύντομη φράση στο παράθυρο του προγράμματος κάνοντας ευκολότερη την εκσφαλμάτωση.

Αλγόριθμος broadcast

Ο κόμβος που εκκινεί το broadcast αποστέλλει το πακέτο σε όλες τις αναφορές του ταυτόχρονα, δηλαδή στους fingers του καθώς επίσης και στον successor. Κάθε κόμβος που λαμβάνει το πακέτο το προωθεί κι αυτός σε όλες του τις αναφορές μέχρις ώτου τελικά το πακέτο παραληφθεί από όλους τους κόμβους. Για να αποφευχθεί η κατάσταση στην οποία το broadcast διατρέχει το δίκτυο για απεριόριστο χρονικό διάστημα, ένας κόμβος διατηρεί μια λίστα με πρόσφατα broadcast που έχει δεχθεί και αν το ξαναδεχθεί μέσα σε ένα εύλογο χρονικό διάστημα το αγνοεί.

5.3.5 Κατανεμημένοι υπολογισμοί

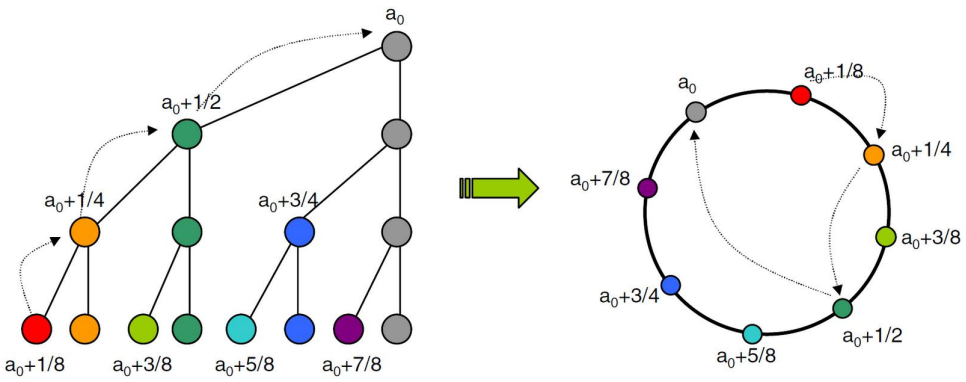
Η διαφοροποίηση του Quantum, από τις τεχνολογίες που αναφέρθηκαν, έγκειται στη δυνατότητα εκτέλεσης αποδοτικών κατανεμημένων υπολογισμών μεταξύ μεγάλου πλήθους κόμβωνχρηστών ενώ παράλληλα διευκολύνεται η προστασία της ιδιωτικότητας αυτών. Η συνεισφορά του Quantum στην προστασία της ιδιωτικότητας έγκειται στην παροχή στοιχείων που βοηθούν τους κατανεμημένους αλγόριθμους, όπως για παράδειγμα ζεύγη δημοσίου/ιδιωτικού κλειδιού για κάθε κόμβο. Ένα τέτοιο παράδειγμα κατανεμημένου υπολογισμού είναι η εύρεση του εκατομμυριούχου (Millionaire's Problem), όχι μόνο ανάμεσα σε δύο κόμβους, αλλά του συνόλου των κόμβων που λαμβάνουν μέρος στο δίκτυο.

Αξίζει να σημειωθεί ότι κατά τη διάρκεια ενός κατανεμημένου υπολογισμού έχουμε θεωρήσει αξιόπιστους κόμβους οι οποίοι δεν αποτυγχάνουν.

Το ξεκίνημα μιας τέτοιας διαδικασίας μπορεί να κάνει οποιοσδήποτε κόμβος ανήκει στο δίκτυο, αποστέλλοντας ένα broadcast μήνυμα, δηλαδή ένα πακέτο προς όλες του τις αναφορές όπως εξηγήθηκε σε προηγούμενη ενότητα. Κάθε ένας από τους κόμβους που θα λάβει το broadcast αναλαμβάνει να το προωθήσει με τη σειρά του στις δικές του αναφορές. Η πρώτη φάση ολοκληρώνεται όταν όλοι οι κόμβοι έχουν λάβει μια φορά το συγκεκριμένο πακέτο. Όπως εξηγήθηκε προηγουμένως, κάθε κόμβος μπορεί να λάβει ένα μοναδικό broadcast μόνο μια φορά.

Ο κατανεμημένος υπολογισμός διαφοροποιείται από ένα απλό broadcast στη δεύτερη φάση του υπολογισμού (τη συγκέντρωση των αποτελεσμάτων) με την έννοια ότι έπειτα από την παραλαβή από κάποιον κόμβο, ο τελευταίος απαντάει στον κόμβο ο οποίος του το απέστειλε ενημερώνοντάς τον σχετικά με το προσωπικό του δεδομένο που πραγματεύεται το πρωτόκολλο. Δημιουργείται έτσι ένα δέντρο κατά τη διάρκεια του υπολογισμού που φαίνεται στο σχήμα 5.1, του οποίου τα φύλλα είναι κόμβοι που δεν έλαβαν απάντηση από την προώθηση του broadcast. Οι κόμβοι που δε βρίσκονται στα φύλλα του δέντρου αναλαμβάνουν να εκτελέσουν την πράξη που ορίζεται από το πρωτόκολλο (στο παράδειγμα του προβλήματος των εκατομυριούχων είναι μια απλή σύγκριση) για τα προσωπικά δεδομένα των απογόνων του. Μετά την εκτέλεση της

πράξης προωθούν το αποτέλεσμα στον πρόγονό τους και η διαδικασία συνεχίζεται μέχρι να συγκεντρωθούν τα αποτελέσματα στον αρχικό κόμβο, ο οποίος εκτελεί τον υπολογισμό για μία τελευταία φορά.



Σχήμα 5.1: Δέντρο Υπολογισμού

5.4 Χρήσεις του Quantum

Στην ενότητα αυτή παρουσιάζεται ένα πιθανό σενάριο υλοποίησης του πρωτοκόλλου του Quantum.

Το πρόβλημα εύρεσης του κοντινότερου γιατρού (NDP) [11] είναι ένα παράδειγμα εφαρμογής διασφάλισης της ιδιωτικότητας και βασίζεται στα δυναμικά προσωπικά δεδομένα της θέσης του κάθε γιατρού. Χρησιμοποιώντας τεχνολογίες P2P δικτύων και συγκεκριμένα του Quantum μπορεί να επιτευχθεί ένας αποδοτικός κατανεμημένος υπολογισμός για ένα μεγάλο πλήθος από γιατρούς που θα διασφαλίζει την ιδιωτικότητά τους. Το άτομο που ανώνυμα προσδιορίζεται ως ο κοντινότερος γιατρός μπορεί να αποκαλύψει (εάν το επιθυμεί) την ταυτότητά του και να προσφέρει τις υπηρεσίες του στο συγκεκριμένο επείγον περιστατικό.

Μια βασική προϋπόθεση είναι ότι όλοι οι γιατροί έχουν στην διάθεση τους έναν προσωπικό agent για την διαχείριση των προσωπικών τους δεδομένων, όπως η τρέχουσα θέση τους. Επίσης, κάθε αγεντ βρίσκεται κάτω από τον έλεγχο του ιδιοκτήτη του και όλοι οι προσωπικοί agents βρίσκονται μόνιμα συνδεδεμένοι με το διαδίκτυο. Σε περίπτωση έκτακτης ανάγκης, οι agents όλων των γιατρών εκτελούν έναν κατανεμημένο υπολογισμό που μπορεί να προσδιορίσει με έναν κρυπτογραφικά ασφαλή τρόπο ποιος είναι ο κοντινότερος γιατρός στο περιστατικό.

Κεφάλαιο 6

Συμπεράσματα

Η ενασχόληση μου με το αντικείμενο της διπλωματικής εργασίας, μου έδωσε χρήσιμες γνώσεις πάνω στις έννοιες τις ιδιωτικότητας, των προσωπικών δεδομένων και της κρυπτογραφίας. Επιπρόσθετα, μου έδωσε την ευκαιρία να ασχοληθώ με έννοιες των δικτύων ομότιμων κόμβων και αποδοτικών κατανεμημένων υπολογισμών. Επίσης, σημαντικές γνώσεις αποκόμισα από την ενασχόληση μου με το Polis καθώς είχα την ευκαιρία να δουλέψω σε ένα προθεςτ που αναπτύσσεται από την Ερευνητική Μονάδα Αλγορίθμων και Ιδιωτικότητας του τμήματος μας. Τέλος, με την ανάπτυξη του Χυαντυμ είχα την ευκαιρία να μάθω περισσότερα πράγματα για το περιβάλλον προγραμματισμού .NET για την γλώσσα C#. Επίσης, αποκόμισα περισσότερες γνώσεις πάνω σε προχωρημένα θέματα προγραμματισμού, όπως τη λειτουργία των νημάτων, sockets, συγχρονισμό μεταξύ threads κ.ά.

Όπως είναι λογικό, κατά την διάρκεια της δουλειάς μου αντιμετώπισα δυσκολίες. Αυτές αφορούσαν κυρίως προβλήματα στην ανάπτυξη του κώδικα, ωστόσο λύθηκαν με την σημαντική βοήθεια του κ. Παύλου Εφραιμίδη. Χαρακτηριστικά, ένα πολύ σημαντικό πρόβλημα ήταν η διαδικασία της εκσφαλμάτωσης (debugging). Το γεγονός της αποκέντρωσης και γενικότερα της κατανεμημένης αρχιτεκτονικής εμπόδισε κατά ένα πολύ μεγάλο παράγοντα την αποδοτική εκσφαλμάτωση, η οποία πραγματοποιήθηκε κατά κύριο λόγο με απλά μηνύματα στα παράθυρα των agents χωρίς τη χρήση των εργαλείων που παρέχουν τα σύγχρονα περιβάλλοντα προγραμματισμού.

Το σημαντικότερο συμπέρασμα που προέκυψε από την εκπόνηση της εργασίας αυτής είναι ότι οι αποδοτικοί αριθμητικοί υπολογισμοί σε κατανεμημένο περιβάλλον είναι εφικτοί. Ωστόσο, υπάρχουν ακόμη ανοιχτά σημαντικά ζητήματα που αφορούν κυρίως την επέκταση του πιλοτικού πρωτοκόλλου Quantum, σε τεχνικό επίπεδο και μη.

Δυναμική αποστολή πρωτοκόλλου

Ένας κατανεμημένος υπολογισμός στο Quantum μπορεί να επιτευχθεί εάν όλοι οι κόμβοι του δικτύου έχουν γνώση του αλγορίθμου του υπολογισμού εκ των προτέρων. Αυτό θεωρείται περιορισμός του Quantum και μπορεί να λυθεί με σύγχρονες τεχνολογίες προγραμματισμού, όπως η ανταλλαγή του ίδιου του αλγόριθμου (π.χ. script ή binary κώδικα) μεταξύ των κόμβων κατά τη διάρκεια εκτέλεσης του υπολογισμού, ο οποίος θα υλοποιεί τον εν λόγω υπολογισμό.

Η παραπάνω λύση, όμως, δημιουργεί ένα σημαντικό μειονέκτημα. Η αποστολή του αλγορίθμου του υπολογισμού από κάποιον κόμβο στο υπόλοιπο δίκτυο θέτει σε κίνδυνο την ιδιωτικότητα των υπολοίπων agents, μιας και αυτοί ενδέχεται να εκτελέσουν αλγόριθμο που δεν τηρεί τα πρότυπα της ιδιωτικότητας.

Υπηρεσίες πιστοποίησης πρωτοκόλλων

Λύση στο παραπάνω πρόβλημα μπορούν να προσφέρουν αξιόπιστες υπηρεσίες, με σκοπό την πιστοποίηση των κατανεμημένων αλγορίθμων που εκτελούνται στο δίκτυο. Ένας κόμβος θα εκτελεί κάποιον αλγόριθμο μόνο εάν αυτός έχει προηγουμένως πιστοποιηθεί από την αξιόπιστη αυτή υπηρεσία.

Άρση παραδοχών

Εξετάζεται η δυνατότητα να προστεθούν μέθοδοι χειρισμού της συμπεριφοράς των κόμβων, έτσι ώστε να άρουμε τις διάφορες παραδοχές που έχουμε θεωρήσει. Έτσι, το Quantum αναμένεται να υποστηρίξει κόμβους οι οποίοι δεν λειτουργούν πάντα προς όφελος του δικτύου καθώς επίσης και μη αξιόπιστους κόμβους κατά τη διάρκεια του κατανεμημένου υπολογισμού. Πιο συγκεκριμένα, το Quantum αναμένεται να είναι σε θέση να εντοπίσει κακόβουλους agents που σαν σκοπό έχουν την παραπλάνηση άλλων και τη δυσλειτουργία της δικτυακής οργάνωσης. Επιπρόσθετα, θα πρέπει να βρεθεί λύση στο πρόβλημα της θεώρησης αξιόπιστων κόμβων κατά τη διάρκεια ενός υπολογισμού, δεδομένου ότι μία αποτυχία ενός κόμβου μπορεί να προκαλέσει εσφαλμένο αποτέλεσμα στον υπολογισμό, ακόμα και την αδυναμία τερματισμού του.

Έλεγχος σταθερότητας κώδικα

Η εφαρμογή που υλοποιεί το πρωτόκολλο Quantum, όπως είναι φυσικό, εμπεριέχει προβλήματα που με τη σειρά τους επηρεάζουν τη σταθερότητα της εφαρμογής. Είναι σημαντικό να δοκιμαστεί επαρκώς ώστε να λυθούν τεχνικά προβλήματα που υπάρχουν.

Πειράματα

Είναι επίσης σημαντικό η εφαρμογή του Quantum να δοκιμαστεί σε πραγματικό περιβάλλον για να παρθούν σωστές μετρήσεις. Οι δοκιμές που έγιναν στα πλαίσια της διπλωματικής αυτής εργασίας πραγματοποιήθηκαν σε τοπικό περιβάλλον ενός υπολογιστή και κατ' επέκταση ο ρόλος του δελαψ σε ένα πραγματικό δίκτυο έπαιξε ελάχιστο ρόλο.

Βιβλιογραφία

[1] Β.Α. Κάτος, και Γ.Χ. Στεφανίδης. *Τεχνικές Κρυπτογραφίας & Κρυπτανάλυσης*. Εκδόσεις Ζυγός, 2003.

[2] Pavlos S. Efraimidis and Georgios Drosatos and Fotis Nalbadis and Aimilia Tasidou. Towards privacy in personal data management. *Journal on Information Management & Computer Security*, 17(4), 2009.

[3] Ashwin R. Bharambe, Mukesh Agrawal και Srinivasan Seshan. Mercury: supporting scalable multi-attribute range queries. *SIGCOMM Comput. Commun. Rev.*, 34:353–366, 2004.

[4] Danny Bickson, Danny Dolev, Genia Bezman και Benny Pinkas. Peer-to-peer secure multi-party numerical computation. Στο *Proceedings of the 2008 Eighth International Conference on Peer-to-Peer Computing*, σελίδες 257–266, Washington, DC, USA, 2008. IEEE Computer Society.

[5] Peter Bogetoft, Dan Lund Christensen, Ivan Damgard, Martin Geisler, Thomas Jakobsen, Mikkel Krøigaard, Janus Dam Nielsen, Jesper Buus Nielsen, Kurt Nielsen, Jakob Pagter, Michael Schwartzbach και Tomas Toft. Multiparty computation goes live. Cryptology ePrint Archive, Report 2008/068, 2008.

[6] John Buford, Heather Yu και Eng Keong Lua. *P2P Networking and Applications*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2008.

[7] Edith Cohen, Amos Fiat και Haim Kaplan. Associative search in peer to peer networks: Harnessing latent semantics. *Comput. Netw.*, 51:1861–1881, 2007.

[8] Arturo Crespo και Hector Garcia-Molina. Routing indices for peer-to-peer systems. Στο *Proceedings of the 22 nd International Conference on Distributed Computing Systems (ICDCS'02)*, ICDCS '02, σελίδες 23–, Washington, DC, USA, 2002. IEEE Computer Society.

[9] Anwitaman Datta, Manfred Hauswirth και Karl Aberer. Updates in highly unreliable, replicated peer-to-peer systems. Στο *Proceedings of the 23rd International Conference on Distributed Computing Systems*, ICDCS '03, σελίδες 76–, Washington, DC, USA, 2003. IEEE Computer Society.

[10] Alan Demers, Dan Greene, Carl Hauser, Wes Irish, John Larson, Scott Shenker, Howard Sturgis, Dan Swinehart και Doug Terry. Epidemic algorithms for replicated database maintenance. Στο *Proceedings of the sixth annual ACM Symposium on Principles of distributed computing*, PODC '87, σελίδες 1–12, New York, NY, USA, 1987. ACM.

[11] Georgios Drosatos και Pavlos S. Efraimidis. A privacy-preserving protocol for finding the nearest doctor in an emergency. Στο *Proceedings of the 3rd International Conference on PErvasive Technologies Related to Assistive Environments*, PETRA '10, σελίδες 18:1–18:8, New York, NY, USA, 2010. ACM.

[12] Prasanna Ganesan, Beverly Yang και Hector Garcia-Molina. One torus to rule them all: multi-dimensional queries in p2p systems. Στο *Proceedings of the 7th International Workshop on the Web and Databases: colocated with ACM SIGMOD/PODS 2004*, WebDB '04, σελίδες 19–24, New York, NY, USA, 2004. ACM.

[13] Hector Garcia-Molina και Arturo Crespo. Semantic overlay networks for p2p systems. Τεχνική Αναφορά υπ. αρίθμ. 2003-75, Stanford InfoLab, 2003.

[14] Craig Gentry. Fully homomorphic encryption using ideal lattices. Στο *STOC '09: Proceedings of the 41st annual ACM symposium on Theory of computing*, σελίδες 169–178, New York, NY, USA, 2009. ACM.

[15] Ken Y. K. Hui, John C. S. Lui και David K. Y. Yau. Small-world overlay p2p networks: construction, management and handling of dynamic flash crowds. *Comput. Netw.*, 50:2727–2746, 2006.

[16] Eng Keong Lua, Jon Crowcroft, Marcelo Pias, Ravi Sharma και Steven Lim. A survey and comparison of peer-to-peer overlay network schemes. *IEEE Communications Surveys and Tutorials*, 7:72–93, 2005.

[17] Qin Lv, Pei Cao, Edith Cohen, Kai Li και Scott Shenker. Search and replication in unstructured peer-to-peer networks. Στο *Proceedings of the 16th international conference on Supercomputing*, ICS '02, σελίδες 84–95, New York, NY, USA, 2002. ACM.

[18] Vassilis Papadimos, David Maier και Kristin Tufte. Distributed query processing and catalogs for peer-to-peer systems. Στο *In CIDR*, σελίδες 5–8, 2003.

[19] Larry L. Peterson και Bruce S. Davie. *Computer Networks: A Systems Approach, 3rd Edition*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2003.

[20] Sylvia Ratnasamy, Paul Francis, Mark Handley, Richard Karp και Scott Shenker. A scalable content-addressable network. *SIGCOMM Comput. Commun. Rev.*, 31:161–172, 2001.

[21] Mema Roussopoulos και Mary Baker. Cup: Controlled update propagation in peer-to-peer networks, 2002.

[22] O. D. Sahin, A. Gupta, D. Agrawal και A. El Abbadi. A peer-to-peer framework for caching range queries. Στο *Proceedings of the 20th International Conference on Data Engineering*, ICDE '04, σελίδες 165–, Washington, DC, USA, 2004. IEEE Computer Society.

[23] Bruce Schneier. *Applied Cryptography*. John Wiley & Sons, Inc., 2νδη έκδοση, 1996.

[24] R. Schollmeier. [16] a definition of peer-to-peer networking for the classification of peer-to-peer architectures and applications. Στο *Proceedings of the First International Conference on Peer-to-Peer Computing*, P2P '01, σελίδες 101–, Washington, DC, USA, 2001. IEEE Computer Society.

[25] R. Schollmeier. A definition of peer-to-peer networking for the classification of peer-to-peer architectures and applications. *Peer-to-Peer Computing*, σελίδες 101–102, 2001.

[26] Kunwadee Sripanidkulchai, Bruce Maggs και Hui Zhang. Efficient content location using interest-based locality in peer-to-peer systems, 2003.

[27] Ion Stoica, Robert Morris, David Liben-Nowell, David R. Karger, M. Frans Kaashoek, Frank Dabek και Hari Balakrishnan. Chord: a scalable peer-to-peer lookup protocol for internet applications. *IEEE/ACM Trans. Netw.*, 11:17–32, 2003.

[28] Chunqiang Tang, Zhichen Xu και Sandhya Dwarkadas. Peer-to-peer information retrieval using self-organizing semantic overlay networks. Στο *Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications*, SIGCOMM '03, σελίδες 175–186, New York, NY, USA, 2003. ACM.

[29] Αρχή Προστασίας Δεδομένων Προσωπικού Χαρακτήρα. Νομοθεσία σχετικά με την προστασία προσωπικών δεδομένων, Φεβρουάριος 2010.

[30] Dimitrios Tsoumakos και Nick Roussopoulos. Analysis and comparison of p2p search methods. Στο *Proceedings of the 1st international conference on Scalable information systems*, InfoScale '06, New York, NY, USA, 2006. ACM.

[31] Samuel D. Warren και Louis D. Brandeis. The right to privacy. *Harvard Law Review, Vol. IV, No. 5*, December 15, 1890.

[32] Alan Westin. Privacy and freedom. *New York, U.S.A.: Atheneum*, 1967.

[33] Wikipedia. Elgamal encryption, February 2010.

[34] Wikipedia. Homomorphic encryption, February 2010.

[35] Wikipedia. Public key certificate, February 2010.

[36] Wikipedia. Rsa, February 2010.

[37] Wikipedia. Secure multi-party computation, February 2010.

[38] Wikipedia. Sha hash functions, February 2010.

[39] Wikipedia. Transport layer security, February 2010.

[40] Wikipedia. Informational self-determination, March 2010.

[41] Andrew C. Yao. Protocols for secure computations (extended abstract). *Proceedings of the 21st Annual IEEE Symposium on the Foundations of Computer Science*, σελίδες 160–164, 1982.