Εθνικο Μετσοβιο Πολυτεχνειο
Σχολη Ηλεκτρολογων Μηχανικων Και Μηχανικων Υπολογιστων
Τομεας Τεχνολογιας Πληροφορικης Και Υπολογιστων

Αυτόνομα Δίκτυα Παγκόσμιας Κλίμακας για τη Διαχείριση και Διάθεση Πληροφορίας

ΔΙΔΑΚΤΟΡΙΚΗ ΔΙΑΤΡΙΒΗ

του

ΑΝΤΩΝΙΟΥ Κ. ΧΑΖΑΠΗ

Αθήνα, Δεκέμβριος 2009

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

Αυτόνομα Δίκτυα Παγκόσμιας Κλίμακας για τη Διαχείριση και Διάθεση Πληροφορίας

ΔΙΔΑΚΤΟΡΙΚΗ ΔΙΑΤΡΙΒΗ

του

ΑΝΤΩΝΙΟΥ Κ. ΧΑΖΑΠΗ

Συμβουλευτική Επιτροπή: Νεκτάριος Κοζύρης
Παναγιώτης Τσανάκας
Ανδρέας-Γεώργιος Σταφυλοπάτης

Εγκρίθηκε από την επταμελή εξεταστική επιτροπή την . . .

. . .
Νεκτάριος Κοζύρης
Αν. Καθηγητής ΕΜΠ

. . .
Παναγιώτης Τσανάκας
Καθηγητής ΕΜΠ

. . .
Ανδρέας-Γεώργιος Σταφυλοπάτης
Καθηγητής ΕΜΠ

. . .
Τιμολέων Σελλής
Καθηγητής ΕΜΠ

. . .
Κώστας Κοντογιάννης
Αν. Καθηγητής ΕΜΠ

. . .
Νικόλαος Παπασπύρου
Επ. Καθηγητής ΕΜΠ

. . .
Διομήδης Σπινέλλης
Καθηγητής ΟΠΑ

Αθήνα, Δεκέμβριος 2009

. . . . . .

ΑΝΤΩΝΙΟΣ Κ. ΧΑΖΑΠΗΣ
Πτυχιούχος Τμήματος Επιστήμης Υπολογιστών Πανεπιστημίου Κρήτης (2002)
Διδάκτωρ Σχολής Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών ΕΜΠ (2009)

© 2009, ΑΝΤΩΝΙΟΣ Κ. ΧΑΖΑΠΗΣ
Με επιφύλαξη παντός δικαιώματος. All rights reserved.

Απαγορεύεται η αντιγραφή, αποθήκευση και διανομή της παρούσας εργασίας, εξ ολοκλήρου ή τμήματος αυτής, για εμπορικό σκοπό. Επιτρέπεται η ανατύπωση, αποθήκευση και διανομή για σκοπό μη κερδοσκοπικό, εκπαιδευτικής ή ερευνητικής φύσης, υπό την προϋπόθεση να αναφέρεται η πηγή προέλευσης και να διατηρείται το παρόν μήνυμα. Ερωτήματα που αφορούν τη χρήση της εργασίας για κερδοσκοπικό σκοπό πρέπει να απευθύνονται προς τον συγγραφέα.

Οι απόψεις και τα συμπεράσματα που περιέχονται σε αυτό το έγγραφο εκφράζουν τον συγγραφέα και δεν πρέπει να ερμηνευθεί ότι αντιπροσωπεύουν τις επίσημες θέσεις του Εθνικού Μετσόβιου Πολυτεχνείου.

Αυτόνομα Δίκτυα Παγκόσμιας Κλίμακας
για τη Διαχείριση και Διάθεση Πληροφορίας

Αντώνης Χαζάπης

Abstract

Structured peer-to-peer systems, or else Distributed Hash Tables (DHTs), are widely established as one of the most efficient platforms for global-scale, distributed data management. The resulting overlay networks are based on the concept of equality and homogeneity of capabilities and obligations among participating parties. Their prominent features include scalability to extremely large network sizes and fault-tolerance. DHTs self-regulate and self-manage problems that may arise both at the layer of the physical infrastructure (unadvertised network connectivity failures, or unresponsiveness of computing systems hosting the overlay nodes), or the communication protocol (deliberate malicious behavior).

The research presented in this dissertation, focuses on the fact that proposed DHT algorithms and implementations, succeed on delivering only a subset of data management operations, such as insertion and lookup based on an identification key. They do not allow altering information already stored and tied to a specific identifier, or searching and retrieving a range of keys (range queries). Although these functions represent important research targets in the field of databases (distributed or not) and many implementation options have been suggested in related literature, the applicability of corresponding algorithms in DHTs is limited by the particular, peculiar characteristics of the P2P environment. A DHT is highly dynamic in nature, thus no specific node can be trusted to solitary handle information crucial to the completion of the distributed operation. Moreover, all operations should account for the special techniques employed by DHTs to distribute available data to participating peers. For example, the difficulty in supporting mutable data lies in the fact that each individual stored key-value pair is automatically copied to a dynamic set of nodes. For most widespread protocols, information, once stored in a DHT, can not be changed. Accordingly, most DHT-based range query implementations, are designed so that the search function runs on a separate layer of software, on top of the overlay, disregarding the existence of data replicas and without exploiting the network configuration, which already encodes peer and data relationships in its structure.

We propose and evaluate the implementations of two alternatives for updating key-value pairs in DHTs: A lookup algorithm that helps in maintaining a loose consistency between multiple versions of data replicas, as well as a new DHT protocol – XOROS, based on the Kademlia routing scheme, which inherently supports serializable and consistent data changes. XOROS is also tolerant to "Byzantine" behavior – intentional non-compliance with its protocol by some participants. In addition, we present a protocol that addresses the problem of implementing complex, single or multi-dimensional range queries in DHTs. In all cases, the extra functionality is offered without sacrificing fundamental properties of the system, such as scalability and fault tolerance. Therefore, the resulting infrastructure for storing, retrieving and searching data can form the basis for supporting a multitude of novel, global-scale distributed applications. The proposed algorithms and protocols have been thoroughly evaluated, using a prototype integrated environment for implementing and evaluating P2P networks. Experimental results are used to assess how the various overlay parameters affect its performance and help in proposing directions for future work.

Περιεχόμενα

Πρόλογος **x**

1 Εισαγωγή **1**

2 Δίκτυα ομότιμων κόμβων **5**
2.1 Δομημένες και μη διατάξεις 6
2.2 Το πρωτόκολλο Kademlia 10

3 Ενημέρωση δεδομένων **13**
3.1 Το πρόβλημα της ενημέρωσης δεδομένων 13
3.2 Σχετικές εργασίες 20
3.3 Εντοπίζοντας τις αλλαγές τιμών 23
3.3.1 Υλοποίηση 23
3.3.2 Παρατηρήσεις 24
3.3.3 Αξιολόγηση 25
3.3.4 Συμπεράσματα 29
3.4 XOROS 30
3.4.1 Η συναλλαγή ενημέρωσης 30
3.4.2 Ανοχή ασύμβατης συμπεριφοράς 32
3.4.3 Λεπτομέρειες υλοποίησης 34
3.4.4 Αξιολόγηση 35
3.4.5 Συμπεράσματα 37

4 Αναζήτηση δεδομένων **39**
4.1 Το πρόβλημα της αναζήτησης δεδομένων 39
4.2 Σχετικές εργασίες 43
4.3 Υλοποίηση 47
4.3.1 Αναζήτηση εύρους διευθύνσεων 48
4.3.2 Κλιμάκωση στον πολυδιάστατο χώρο 53
4.4 Αξιολόγηση 57
4.4.1 Μέτρηση των γειτονιών 58
4.4.2 Η επίδραση του αριθμού των διαστάσεων 60
4.4.3 Παράμετροι λειτουργίας 64
4.4.4 Κατανέμοντας τον φόρτο 65
4.5 Συμπεράσματα 67

5 Σενάρια χρήσης **69**
5.1 Το λογισμικό διαχείρισης δεδομένων του Gredia 69
5.1.1 Υλοποίηση της επικάλυψης μεταδεδομένων 70
5.1.2 Υλοποίηση της επικάλυψης αντιστοίχισης ονομάτων 73
5.1.3 Υλοποίηση της επικάλυψης αποθήκευσης και μεταφοράς 75
5.2 Κατανεμημένη μηχανή αναζήτησης 77

6 Επίλογος **81**

Α′ PeerPlatform **83**
Α′.1 Σχεδιασμός και υλοποίηση 84
Α′.1.1 Επισκόπηση 85
Α′.1.2 Τρόποι λειτουργίας 87
Α′.1.3 Τακτικές προσομοίωσης 88
Α′.2 Αξιολόγηση του Kademlia 90
Α′.2.1 Υλοποίηση του πίνακα δρομολόγησης 92
Α′.2.2 Βελτιστοποιήσεις στην εκτέλεση πράξεων δεδομένων 93

Βιβλιογραφία **97**

Δημοσιεύσεις **105**

Πρόλογος

Μου φαίνεται εξαιρετικά δύσκολο να αποτυπώσω σε λίγες μόλις γραμμές τη διαδρομή που καταλήγει σε αυτή την εργασία: Τους λόγους για τους οποίους ασχολήθηκα με το συγκεκριμένο θέμα, τους ανθρώπους που με βοήθησαν, τις ευχάριστες, αλλά και δυσάρεστες στιγμές. Ήταν ένα δύσκολο ταξίδι. Η ιδέα της ολοκλήρωσής του, με γεμίζει αυτοπεποίθηση, αλλά και μελαγχολία, αφού αναπολώ τα περιπετειώδη μονοπάτια του.

Κοιτάζοντας πίσω, πριν επτά περίπου χρόνια, θυμάμαι τη περιέργεια και τον ενθουσιασμό που αισθανόμουν για τις "νέες" – τότε – τεχνολογίες *δικτύων ομότιμων κόμβων*. Η επανάσταση που θα έφερναν στον τρόπο με τον οποίο χρησιμοποιούμε τους υπολογιστές, φαινόταν ότι θα είναι αντίστοιχη με εκείνη που έφερε το Διαδίκτυο. Η αλήθεια είναι ότι κάτι τέτοιο δεν έγινε όπως ακριβώς το φανταζόταν η ακαδημαϊκή, αλλά και επιχειρηματική κοινότητα. Σήμερα, τα περισσότερα συστήματα αυτού του τύπου χρησιμοποιούνται για το "κατέβασμα" αρχείων από το Διαδίκτυο – "πειρατικών" εκδόσεων λογισμικού ή video και μουσικής. Παρόλο που μια μεγάλη πλειοψηφία ανθρώπων έχει εκούσια ή ακούσια συμμετάσχει σε ένα ΔΟΚ, η "επανάσταση" δεν έγινε. Ακόμα. Όλοι περιμένουν ότι κάποια στιγμή το μοντέλο του "κεντρικού σημείου αναφοράς" δεν θα επαρκεί για το μέγεθος του Διαδικτύου και το πλήθος των δεδομένων που θα διακινούνται. Ο αριθμός των διασυνδεδεμένων συσκευών πολλαπλασιάζεται συνεχώς και αντίστοιχα αυξάνονται οι απαιτήσεις του συνόλου, τόσο σε διαθέσιμο αποθηκευτικό χώρο όσο και σε υπολογιστική ισχύ. Έτσι, ο σχεδιασμός συνεργατικών συστημάτων αποτελεί πλέον διεθνή τάση και πρακτική. Πιστεύω ότι κάποια στιγμή στο μέλλον θα γίνει η αναμενόμενη αλλαγή. Ελπίζω οι προτάσεις και τα συμπεράσματα της παρούσας διατριβής να συμβάλλουν σε αυτή τη κατεύθυνση και να αποτελέσουν βάση και έναυσμα για περαιτέρω έρευνα σε αυτόν τον τομέα της επιστήμης.

Προσωπικά, κλείνοντας αυτόν τον κύκλο της ζωής μου, θέλω να ευχαριστήσω με ειλικρίνεια και σεβασμό όσους με τον τρόπο τους συνέβαλαν στην ολοκλήρωση αυτής της εργασίας. Ευχαριστώ την επιτροπή των καθηγητών που είχε το δύσκολο έργο της παρακολούθησης και κρίσης της δουλειάς μου και ιδιαίτερα τον επιβλέποντα καθηγητή. Ευχαριστώ τους αδερφικούς μου φίλους από τα χρόνια μου στο σχολείο και το Πανεπιστήμιο Κρήτης. Ευχαριστώ τους φίλους και συναδέλφους, μέλη του Εργαστηρίου Υπολογιστικών Συστημάτων και ειδικότερα αυτούς με τους οποίους είχα την ευκαιρία να συνεργαστώ στενά σε καθημερινή βάση, είτε στα πλαίσια της έρευνας, είτε για άλλα θέματα. Εύχομαι σε όλους κάθε επιτυχία σε ό,τι επιλέξουν και καλή τύχη σε ό,τι δεν μπορούν να προβλέψουν. Αποτέλεσαν πηγή έμπνευσης σε όλη τη διαδρομή μου στο Ε.Μ.Π. Έμπνευση όμως κυρίως αντλώ από τους γονείς μου, την αδερφή μου, τη μέλλουσα γυναίκα μου

και όλη την οικογένεια μου – τους ανθρώπους που είναι καθημερινά δίπλα μου, αλλά και εκείνους που έφυγαν, αφήνοντας πίσω τους εκτός από ευχάριστες αναμνήσεις, ένα σοφό παράδειγμα για εμάς. Τους αφιερώνω τη δουλειά αυτή, μέσα από τη καρδιά μου.

Αντώνης Χαζάπης

Αθήνα, Οκτώβριος 2009

Κεφάλαιο 1

Εισαγωγή

Η διεθνής ακαδημαϊκή κοινότητα αναγνωρίζει τα "δομημένα Δίκτυα Ομότιμων Κόμβων" (ΔΟΚ), ή αλλιώς τους "Κατανεμημένους Πίνακες Κατακερματισμού" (ΚΠΚ), ως την πλέον σύγχρονη και αποδοτική λύση στο πρόβλημα της κατανεμημένης διαχείρισης μεγάλων όγκων πληροφορίας, που μπορεί να αποθηκεύεται και να διατίθεται παράλληλα από πολλαπλά φυσικά σημεία σε παγκόσμιο επίπεδο. Μάλιστα, τα συγκεκριμένα δίκτυα — που βασίζονται στην έννοια της ισότητας και ομογένειας δυνατοτήτων και υποχρεώσεων μεταξύ των συμμετεχόντων — έχουν την ικανότητα να αυτορυθμίζονται και να αυτοδιαχειρίζονται προβλήματα που μπορεί να προκύπτουν τόσο ακούσια όσο και εκούσια, στο επίπεδο της φυσικής υποδομής (φυσικό δίκτυο διασύνδεσης, φυσικά μηχανήματα φιλοξενίας των κόμβων) ή ακόμα και στο επίπεδο του πρωτοκόλλου επικοινωνίας (κακόβουλη χρήση).

Η έρευνα που διεξήχθη, εστίασε στο γεγονός ότι οι βασικοί αλγόριθμοι και οι προτεινόμενες υλοποιήσεις των παραπάνω δικτύων, υλοποιούν μεν ένα σύνολο πράξεων διαχείρισης δεδομένων (εισαγωγή, εντοπισμός με βάση κάποιο αναγνωριστικό κλειδί, κτλ.), δεν επιτρέπουν όμως τη μεταβολή πληροφοριών που αντιστοιχούν σε κάποιο συγκεκριμένο αναγνωριστικό ή την αναζήτηση ενός εύρους διευθύνσεων (range queries). Οι λειτουργίες αυτές αποτελούν άξονες έρευνας με σημαντική σχετική βιβλιογραφία κυρίως στον τομέα των βάσεων δεδομένων (κατανεμημένων ή μη). Παρόλα αυτά, στα ΔΟΚ δεν εφαρμόζονται άμεσα οι προτεινόμενες λύσεις, λόγω του ιδιαίτερου περιβάλλοντος που δημιουργείται από τη δυναμικότητα του δικτύου και των τεχνικών κατανομής των δεδομένων. Για παράδειγμα, η δυσκολία στην υποστήριξη ανανεώσιμων δεδομένων οφείλεται στο γεγονός ότι τηρούνται αυτόματα πολλαπλά αντίγραφα των τιμών σε ένα δυναμικό σύνολο κόμβων. Έτσι, για τα περισσότερα ευρέως διαδεδομένα πρωτόκολλα, τα αποθηκευμένα δεδομένα δεν μπορούν να αλλάζουν, καθώς δεν υλοποιούνται οι κατάλληλοι αλγόριθμοι στο επίπεδο του ΚΠΚ για τον χειρισμό των ενημερώσεων. Αντίστοιχα, οι περισσότερες μέχρι σήμερα υλοποιήσεις αναζήτησης εύρους διευθύνσεων, έχουν σχεδιαστεί έτσι ώστε οι λειτουργίες αναζήτησης να εκτελούνται σε ένα ξεχωριστό επίπεδο λογισμικού, εκτός του ΚΠΚ, μη λαμβάνοντας υπόψη την ύπαρξη αντιγράφων και χωρίς να εκμεταλλεύονται τα πλεονεκτήματα της δομής που διατηρεί το ίδιο το δίκτυο για να δρομολογεί μηνύματα μεταξύ των κόμβων του.

Στα πλαίσια της διατριβής, προτείνονται και υλοποιούνται δύο εναλλακτικές λύσεις για την ενημέρωση δεδομένων: Ένας αλγόριθμος ανάκτησης για ΚΠΚ που βοηθά στη τήρηση χαλαρής συνέπειας μεταξύ των αλλαγών στα αντίγραφα μιας τιμής και ένα πρωτόκολλο ΚΠΚ (που ονομάζεται XOROS), βασισμένο στο σχήμα δρομολόγησης του Kademlia, το οποίο παρέχει σειριοποιήσιμες και συνεπείς ανανεώσεις εγγενώς στο επίπεδο του ΔΟΚ και είναι ανεκτικό σε "Βυζαντινά" σφάλματα (εκούσια μη συμμόρφωση με τα σχετικά πρωτόκολλα από κάποιους συμμετέχοντες). Επίσης, παρουσιάζεται ένα πρωτόκολλο που αντιμετωπίζει το πρόβλημα της εκτέλεσης σύνθετων, πολυδιάστατων λειτουργιών αναζήτησης εύρους διευθύνσεων σε ΚΠΚ.

Σε κάθε περίπτωση, η ζητούμενη επιπλέον λειτουργικότητα, προσφέρεται χωρίς να θυσιάζονται τα θεμελιώδη χαρακτηριστικά του συστήματος, όπως η δυνατότητα κλιμάκωσης και η ανοχή σε σφάλματα. Έτσι το παραγόμενο υπόστρωμα εγγραφής και ανάγνωσης δεδομένων, αναζήτησης, κτλ., μπορεί να αποτελέσει τη βάση μιας πληθώρας νέων και καινοτόμων εφαρμογών – αναφέρονται ενδεικτικά παραδείγματα και σενάρια χρήσης. Επιπλέον, μέσω πειραματικών αποτελεσμάτων, τα οποία συλλέχθηκαν χρησιμοποιώντας ένα πρότυπο, ολοκληρωμένο σύστημα προσομοίωσης και υλοποίησης ΔΟΚ, έχουν αξιολογηθεί πως οι διάφοροι παράμετροι επηρεάζουν τα χαρακτηριστικά απόδοσης των πρωτοκόλλων.

Η συμβολή της διατριβής εντοπίζεται στα παρακάτω σημεία:

- Προτείνεται ένα μοντέλο εκμετάλλευσης της αυτοδιαχειριζόμενης και αυτοσυντηρούμενης οργάνωσης των κόμβων που συμμετέχουν σε ένα ΔΟΚ, μέσω της ομαδοποίησής τους σε «γειτονιές». Παρόλο που οι ομάδες αυτές αναφέρονται σε σχετικές εργασίες, δεν είχε προταθεί τρόπος που χρησιμοποιεί τις δομές γειτονικών κόμβων για την υλοποίηση νέων λειτουργιών απευθείας στο επίπεδο του πρωτοκόλλου ενός ΚΠΚ. Τα χαρακτηριστικά των γειτονιών περιγράφονται και αξιολογούνται τόσο σε θεωρητικό, όσο και σε πρακτικό επίπεδο.

- Τόσο στη περίπτωση του XOROS, όσο και στου πρωτοκόλλου αναζήτησης εύρους διευθύνσεων, με βάση προηγούμενες εργασίες, παρουσιάζονται για πρώτη φορά ενοποιημένοι αλγόριθμοι ειδικά για ΚΠΚ. Οι αλγόριθμοι αυτοί είναι ειδικά διαμορφωμένοι για το κατανεμημένο περιβάλλον που δημιουργείται σε ένα δομημένο δίκτυο ομότιμων κόμβων και λειτουργούν με δεδομένο την ύπαρξη αντιγράφων της πληροφορίας και την συνεχή ανακατάταξη του δικτύου – δεν θεωρείται ποτέ ότι το σύνολο βρίσκεται σε κατάσταση απόλυτης ισορροπίας. Για το XOROS, μάλιστα, αυτή η παραδοχή γενικεύεται στο ότι μπορεί να συμμετέχουν στον ΚΠΚ και κόμβοι που θέλουν να "βλάψουν" το σύνολο με το να μην εφαρμόζουν κατά γράμμα τα συμφωνημένα πρωτόκολλα επικοινωνίας.

- Παρουσιάζεται μια πρότυπη πλατφόρμα υλοποίησης και αξιολόγησης ΔΟΚ. Η προσομοίωση μεγάλων δικτύων, που μπορεί να αποτελούνται από δεκάδες χιλιάδες κόμβους, αποτελεί σημαντικό ζήτημα σύμφωνα με τη σχετική βιβλιογραφία. Συνήθως χρησιμοποιούνται είτε προσομοιωτές δικτύου, που μετρούν σε πραγματικό χρόνο και σε επίπεδο πακέτου την απόδοση του συστήματος (packet-level), είτε γίνονται γενικές αλγοριθμικές με-

τρήσεις σε βήματα. Η προτεινόμενη πλατφόρμα εκτός από το ότι υποστηρίζει και τις δύο μεθόδους, μπορεί αξιοποιηθεί και για την επιχειρησιακή λειτουργία ενός δικτύου.

Η συνέχεια του κειμένου οργανώνεται ως εξής: Στο επόμενο κεφάλαιο παρουσιάζονται οι γενικές έννοιες που διέπουν τη λειτουργία των Δικτύων Ομότιμων Κόμβων, καθώς και – ειδικότερα – το πρωτόκολλο Kademlia που χρησιμοποιείται ως παράδειγμα διάταξης κόμβων και δρομολόγησης μηνυμάτων στο σύνολο των αλγορίθμων που αναπτύχθηκαν. Η δομή δυαδικού δέντρου του Kademlia έχει το ιδιαίτερο χαρακτηριστικό ότι δημιουργεί μια σχέση συμμετρίας μεταξύ των αντιγράφων των αποθηκευμένων τιμών και των αντίστοιχων κόμβων τήρησής τους – γεγονός που απλουστεύει τις διαδικασίες εύρεσης και διαχείρισης αντιγράφων, σε σχέση με ΚΠΚ που βασίζονται σε κυκλικές δομές. Στα κεφάλαια 3 και 4 αναλύονται οι αλγόριθμοι ενημέρωσης δεδομένων και αναζήτησης εύρους διευθύνσεων αντίστοιχα. Για κάθε ένα από τα δύο διακριτά προβλήματα που αντιμετωπίστηκαν, καταγράφονται λεπτομερώς οι σχετικές εργασίες από τη βιβλιογραφία και στη συνέχεια παρουσιάζεται η λύση που προτείναμε, ακολουθούμενη από πειραματικά αποτελέσματα για την αξιολόγησή της. Πριν τον επίλογο, περιγράφονται σενάρια χρήσης των αποτελεσμάτων της έρευνας, ενώ στο παράρτημα αναφέρεται λεπτομερώς η αρχιτεκτονική και ο τρόπος λειτουργίας της πλατφόρμας υλοποίησης και αξιολόγησης ΔΟΚ.

Κεφάλαιο 2

Δίκτυα ομότιμων κόμβων

Η σχεδίαση και υλοποίηση των δικτύων δεδομένων είχε ως πρωταρχικό ρόλο να καλύψει την ανάγκη για τη διάθεση και ανταλλαγή πόρων μεταξύ υπολογιστικών συστημάτων. Με τη σειρά της, η δικτύωση αποτέλεσε μοχλό ανάπτυξης νέων ερευνητικών κατευθύνσεων στον κλάδο της επιστήμης των υπολογιστών: στόχος των *κατανεμημένων συστημάτων* είναι η προσφορά ενιαίας λειτουργικότητας από ένα σύνολο υπολογιστών που συμμετέχουν σε μια κοινή δικτυακή υποδομή. Σήμερα, η συνεχής ανάπτυξη του Διαδικτύου, τόσο σε μέγεθος όσο και σε χωρητικότητα, έχει προκαλέσει ανάλογη αύξηση και στις προκλήσεις που καλούνται να αντιμετωπίσουν άμεσα, αλλά και στο μέλλον, τα κατανεμημένα συστήματα διαχείρισης πληροφορίας μεγάλης κλίμακας. Τα νέα δεδομένα ορίζουν την ανάγκη εκμετάλλευσης των διεσπαρμένων πόρων αποθήκευσης μέσα από τη δημιουργία αυθαίρετων και μη ιεραρχικών ομάδων μηχανημάτων και χρηστών. Πρέπει να επιτρέπεται στα άκρα του δικτύου να ενοποιούν και να μοιράζονται πληροφορίες μέσω αυτοδιαχειριζόμενων, συνεργατικών πλεγμάτων που δεν εξαρτώνται από κεντρικές υπηρεσίες ή αρχές λειτουργίας.

Αυτά τα περιβάλλοντα, είναι πλέον επιτεύξιμα μέσω της εξάπλωσης των *Δικτύων Ομότιμων Κόμβων* (ΔΟΚ). Τα ΔΟΚ (peer-to-peer networks) αποτελούν μια κατηγορία κατανεμημένων συστημάτων τα οποία στηρίζουν τη λειτουργία τους στην κατασκευή ενός ιδεατού πλέγματος διασυνδέσεων μεταξύ κόμβων που θεωρούνται ομότιμοι ως προς τις δυνατότητες και ιδιότητές τους. Χαρακτηρίζονται από την ικανότητα κλιμάκωσής τους, ενώ η μορφολογία τους και τα σχετικά πρωτόκολλα διαχείρισης εγγυώνται την απρόσκοπτη λειτουργία τους ακόμα και στη περίπτωση σφαλμάτων — είτε υπό τη μορφή δυσλειτουργίας κάποιων κόμβων, είτε σε περιπτώσεις που δεν είναι δυνατή η επικοινωνία σε τμήματα του δικτύου. Σε ένα ΔΟΚ, η ικανότητα επεξεργασίας, η χωρητικότητα αποθήκευσης, αλλά και η ανεκτικότητα σε σφάλματα του συνόλου αυξάνεται όσο περισσότεροι κόμβοι συμμετέχουν στην υποδομή.

Πρακτικά, ένα ΔΟΚ δημιουργείται από ένα σύνολο εφαρμογών που εκτελούνται σε κόμβους οι οποίοι ανήκουν σε μία ενιαία υποδομή δικτύου, όπως το Διαδίκτυο. Οι εφαρμογές αυτές "κωδικοποιούν" τις μεταξύ τους σχέσεις σε κανόνες δρομολόγησης ενός νέου, λογικού δικτύου, το οποίο στη σχετική ορολογία αναφέρεται και ως *επικάλυψη* ή *επίστρωση* (overlay) του πραγματι-

κού. Τα μηνύματα που ανταλλάσσουν οι κόμβοι ενός ΔΟΚ μπορεί να φαίνεται ότι ακολουθούν τις ακμές του ιδεατού πλέγματος, αλλά στη πραγματικότητα ταξιδεύουν πάνω από το υφιστάμενο υπόστρωμα δικτύωσης που τους ενοποιεί σε φυσικό επίπεδο.

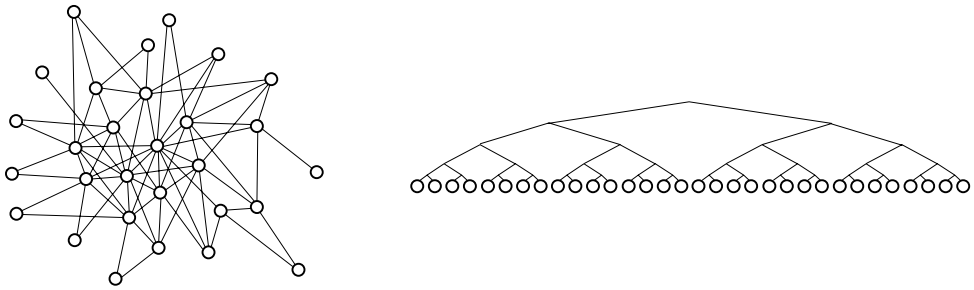
Το αφηρημένο επίπεδο της επικάλυψης δίνει τη δυνατότητα στους συμμετέχοντες κόμβους να ορίζουν τον αριθμό, τις ιδιότητες, καθώς και τη σημασιολογία των συνδέσεων μεταξύ τους. Έτσι, ειδικοί αλγόριθμοι μπορούν να εκμεταλλεύονται τη δομή του πλέγματος για να παρέχουν τελικά κάποιου είδους κατανεμημένη λειτουργικότητα. Χαρακτηριστικό παράδειγμα αποτελεί η χρήση των ΔΟΚ για την αποδοτική κατανεμημένη αναζήτηση ή εντοπισμό δεδομένων που μπορεί να αποθηκεύονται σε οποιονδήποτε κόμβο στο Διαδίκτυο. Γενικά, οι δυνατότητες χρήσης των ΔΟΚ είναι πάρα πολλές και τα σημαντικά πλεονεκτήματα που τα χαρακτηρίζουν έχουν βοηθήσει στην ταχύτατη εξάπλωσή τους. Η υλοποίηση και διάδοση ενός ΔΟΚ γίνεται συνήθως στα πλαίσια της λειτουργίας κάποιου κατανεμημένου λογισμικού το οποίο επιτρέπει στον τελικό χρήστη τον διαμοιρασμό δεδομένων, την κατανεμημένη μεταφόρτωση αρχείων ή την παρακολούθηση ροών πολυμέσων στο Διαδίκτυο. Μάλιστα, μια ολόκληρη γενιά τέτοιων εφαρμογών ευθύνεται σήμερα για ένα μεγάλο μέρος της συνολικής δραστηριότητας του Διαδικτύου [KBFc04].

Ένα βασικό χαρακτηριστικό των ΔΟΚ είναι ότι όλες οι λειτουργίες που επιτελεί κάθε κόμβος ορίζονται σαφώς από κάποιο σχετικό πρωτόκολλο. Οι κόμβοι ενός ΔΟΚ συμφωνούν αρχικά σε ένα κοινό σύνολο κανόνων το οποίο στη συνέχεια ορίζει τη συμπεριφορά του κάθε μέλους στο νοητό πλέγμα. Η ομότιμη φύση των κόμβων συνεπάγεται την υποχρέωση τους να εξυπηρετούν οποιονδήποτε άλλο κόμβο του δικτύου με τον ίδιο τρόπο που προσφέρεται και σε αυτούς η αντίστοιχη υπηρεσία. Σε αντίθεση με το κλασσικό μοντέλο του πελάτη-εξυπηρετητή, ένα λογισμικό που συμμετέχει σε ένα ΔΟΚ επιτελεί και τους δύο ρόλους. Το πρωτόκολλο μπορεί να ορίζει εκτός του τρόπου επικοινωνίας και τον ίδιο τον τρόπο με τον οποίο πρέπει να λειτουργεί ένας κόμβος. Έτσι, σε κάποια συστήματα οι κόμβοι δεν αποφασίζουν καν τι αντικείμενα θα αποθηκεύουν. Αυτά ορίζονται δυναμικά από το πρωτόκολλο, το οποίο παίρνει τις σχετικές αποφάσεις κατανεμημένα, ανάλογα με τη χρήση του δικτύου, τη δημοτικότητα των συγκεκριμένων δεδομένων, κτλ.

2.1 Δομημένες και μη διατάξεις

Στη σχετική βιβλιογραφία, τα δίκτυα ομότιμων κόμβων διακρίνονται ανάλογα με τη διάταξη διασύνδεσης (τη μορφολογία του πλέγματος) και τους αντίστοιχους κανόνες δρομολόγησης που ορίζονται από το πρωτόκολλο λειτουργίας. Έτσι, αναφέρονται δύο βασικές κατηγορίες ΔΟΚ. Τα *μη δομημένα* δίκτυα, όπως το Gnutella, τοποθετούν τους συμμετέχοντες στις κορυφές ενός ιδεατού γράφου και ορίζουν ότι κάθε κόμβος πρέπει να διατηρεί ένα συγκεκριμένο αριθμό επαφών ή αλλιώς *γειτόνων*. Στο νοητό πλέγμα οι ακμές είναι κατευθυντικές (αφού οι γείτονες κάθε κόμβου δεν οφείλουν να διατηρούν την αντίστροφη επαφή) και ενώ ο βαθμός των εξερχόμενων ακμών διατηρείται σταθερός σε όλο το δίκτυο, έχει παρατηρηθεί ότι η κατανομή των εισερχόμενων διασυνδέσεων συγκλίνει μετά από εκτενή χρήση σε εκθετική μορφή: Ένα μικρό υποσύνολο "δημοφιλών" κορυφών συνδέεται με ένα τεράστιο ποσοστό του δικτύου, ενώ ο αριθμός των κόμβων σε σχέση με το πλήθος των ακμών που καταλήγουν σε αυτούς μειώνεται εκθετικά καθώς αυξάνεται

ο αριθμός των ακμών [RF02]. Η τοπολογία αυτή, που θυμίζει τη σχέση των συνδέσμων μεταξύ ιστοσελίδων του Παγκόσμιου Ιστού, έχει το ιδιαίτερο χαρακτηριστικό ότι μπορεί να διαχειριστεί λειτουργίες ελεύθερης αναζήτησης εκπληκτικά γρήγορα [ALPH01].



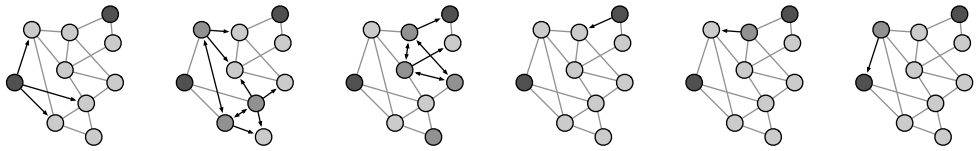
a) Unstructured b) Structured

Σχήμα 2.1: *Διατάξεις μη δομημένων και δομημένων ΔΟΚ*

Σχετικά μηνύματα ξεκινούν από κάποιον κόμβο και προωθούνται από γείτονα σε γείτονα για συγκεκριμένο αριθμό βημάτων. Η διαδρομή που έχει ακολουθήσει το μήνυμα αποθηκεύεται μέσα σε αυτό, ώστε να αποφεύγονται οι κύκλοι δρομολόγησης και για να επιστρέψουν τα αποτελέσματα της αναζήτησης μέσω της ίδιας πορείας στον αρχικό κόμβο που έκανε την αίτηση. Κάθε κόμβος που προωθεί μια απάντηση, φροντίζει να αποθηκεύει τα αποτελέσματα και να τα χρησιμοποιεί για να εξυπηρετεί ο ίδιος επόμενες αναζητήσεις από άλλους κόμβους με ίδια ή παρόμοια θεματολογία. Αυτός ο μηχανισμός ανάδρασης, σε συνδυασμό με την ιδιαίτερη διάταξη των συνδέσεων, εγγυάται ότι με πολύ μεγάλη πιθανότητα οποιοδήποτε μήνυμα αναζήτησης θα περάσει σε ελάχιστα βήματα από κάποιο "κομβικό" μέλος του δικτύου, το οποίο θα έχει και καλύτερη επίγνωση των διαθέσιμων πόρων, έχοντας προωθήσει περισσότερα αντίστοιχα αποτελέσματα στο παρελθόν.

Τα μη δομημένα ΔΟΚ σχεδιάστηκαν για να καλύψουν την άμεση ανάγκη των χρηστών του Διαδικτύου για ανταλλαγή δεδομένων. Διαδόθηκαν ταχύτατα και μέσα σε λίγες μόνο εβδομάδες κατέκλυσαν το δίκτυο. Γρήγορα διαπιστώθηκε ότι πίσω από την καινοτομία και την σχετικά καλή αποδοτικότητα κρύβονταν σημαντικά προβλήματα κλιμάκωσης. Ίσως αφελώς οι σχεδιαστές να μην φαντάστηκαν ποτέ την απήχηση που θα είχε η εφαρμογή τους και τις ενδεχόμενες επιπλοκές που θα δημιουργούσε η διαδικασία αναζήτησης, η οποία "πλημμύριζε" το δίκτυο με μηνύματα για κάθε μεμονωμένη αναζήτηση. Σε μια σχετική εργασία, αναφέρεται ότι το Διαδίκτυο δεν είχε τη χωρητικότητα να αντέξει τον όγκο της κίνησης που δημιουργούνταν από ένα μικρό σχετικά μη δομημένο ΔΟΚ. Το επόμενο βήμα ήταν να προταθούν από την ακαδημαϊκή κοινότητα διάφορες τεχνικές που θα μπορούσαν να λύσουν το πρόβλημα. Χαρακτηριστικά παραδείγματα αποτελούν η προσθήκη βαρών στις ακμές μεταξύ των κόμβων, η προσπάθεια ομαδοποίησης αιτήσεων, η επιλογή γειτόνων για την προώθηση μηνυμάτων ανάλογα με το αντικείμενο της αναζήτησης και προηγούμενες απαντήσεις, κτλ. Επιπρόσθετα, ένα άλλο πρόβλημα που εξετάστηκε σε μη δομημένα ΔΟΚ ήταν η κατανομή του φόρτου επεξεργασίας των αιτήσεων στους κόμβους, η οποία λόγω της φύσης του πλέγματος δεν μπορεί να είναι ομαλή. Η "συγκέντρωση" που δημιουργείται σε ορισμένες κορυφές λόγω του βαθμού εισερχόμενων ακμών συντηρείται και προωθείται από

το πρωτόκολλο, κάτι που έρχεται σε αντίθεση με την ανάγκη ομοιογένειας μεταξύ των κόμβων. Πέραν της απαίτησης κάποια μέλη του πλέγματος να έχουν σημαντικά ισχυρότερα χαρακτηριστικά (σε δυνατότητες επεξεργασίας, αποθήκευσης και χωρητικότητας δικτύου), η πρακτική αυτή θέτει θέματα ασφάλειας και ανοχής του συστήματος σε σφάλματα. Μάλιστα, οι απόψεις της κοινότητας επί του συγκεκριμένου θέματος διίστανται. Κάποιες εργασίες προτείνουν τεχνικές για την ισοκατανομή του συστήματος, ενώ άλλες εκτιμούν ότι η διαφοροποίηση μεταξύ των κόμβων μπορεί να δράσει θετικά και περιγράφουν τεχνικές διαχείρισης και εκμετάλλευσης των "υπερκόμβων" που δημιουργούνται.



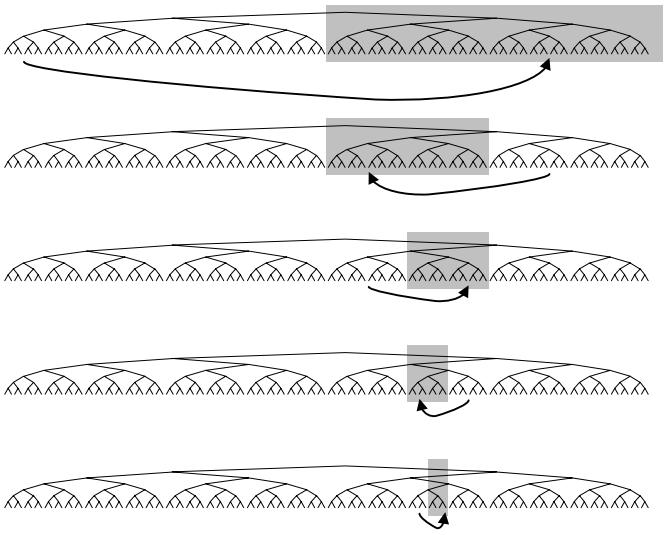
Σχήμα 2.2: *Αναζήτηση ενός αντικειμένου σε ένα μη δομημένο ΔΟΚ*

Στα μη δομημένα δίκτυα η επιτυχία μιας αναζήτησης εξαρτάται από τη πιθανότητα το σχετικό μήνυμα να φτάσει στους κόμβους που αποθηκεύουν το ζητούμενο αντικείμενο, κάτι που καθορίζεται τόσο από το πόσο *δημοφιλές* είναι το θέμα της αναζήτησης, όσο και από τις θέσεις των κόμβων που διασχίζει το ερώτημα καθώς πορεύεται στο πλέγμα. Από την άλλη μεριά όμως, υπάρχουν περιπτώσεις που μπορεί να μην μας ενδιαφέρει η ελεύθερη αναζήτηση όρων, αλλά ο εντοπισμός ενός συγκεκριμένου, *επώνυμου* δεδομένου. Με αυτό το σκεπτικό, σχεδιάστηκαν τα *δομημένα* ΔΟΚ, τα οποία αναφέρονται και ως *κατανεμημένοι πίνακες κατακερματισμού* (Distributed Hash Tables - DHTs). Όπως και στους κλασσικούς πίνακες κατακερματισμού οι ΚΠΚ αναλαμβάνουν την αποθήκευση πληροφοριών που συσχετίζονται, μέσω μιας ειδικής *συνάρτησης κατακερματισμού* (hash function), με ένα μοναδικό *κλειδί*. Σε αυτά τα ΔΟΚ, το πρωτόκολλο χρησιμοποιεί μια ορισμένη διάταξη κόμβων και τους κατάλληλους μηχανισμούς διαχείρισης δεδομένων, έτσι ώστε να προσφέρεται μια βελτιστοποιημένη υποδομή κατανεμημένης *τοποθέτησης* (put/store) και *εντοπισμού* (get/locate): Δεδομένου ενός κλειδιού, ένας ΚΠΚ μπορεί να δρομολογήσει την αντίστοιχη αίτηση στον κόμβο υπεύθυνο για την αποθήκευσή του, σε ελάχιστα και συγκεκριμένα βήματα — μηνύματα από τον ένα κόμβο στον άλλο.

Παρόλο που έχουν προταθεί αρκετοί σχετικοί αλγόριθμοι, η μεθοδολογία που χρησιμοποιείται από όλους τους ΚΠΚ είναι κοινή και βασίζεται στον εκ των προτέρων ορισμό ενός *εικονικού χώρου διευθύνσεων*, με συγκεκριμένη, προαποφασισμένη δομή. Σε αυτόν τον εικονικό χώρο, αντιστοιχίζονται σε κλειδιά (ονοματίζονται) και στη συνέχεια τοποθετούνται τόσο τα δεδομένα όσο και οι ίδιοι οι κόμβοι, χρησιμοποιώντας μια κοινή συνάρτηση κατακερματισμού που εξασφαλίζει την ομαλή κατανομή τους στο σύνολο των διαθέσιμων διευθύνσεων. Στις περισσότερες σχετικές υλοποιήσεις, ως συνάρτηση κατακερματισμού χρησιμοποιείται ο αλγόριθμος SHA-1, ο οποίος κωδικοποιεί το μήνυμα εισόδου σε μία συμβολοσειρά μεγέθους 160 δυαδικών ψηφίων (20 χαρακτήρων). Επειδή ακριβώς τα κλειδιά των δεδομένων παράγονται από τα ίδια τα δεδομένα, οι ΚΠΚ αναφέρεται ότι χρησιμοποιούν *κλειδιά κατακερματισμού δεδομένων* (content hash keys). Σε κάθε ΚΠΚ ορίζεται επίσης μια *συνάρτηση εγγύτητας*, που παίρνει ως όρισμα δύο κλειδιά

και απαντά με την τιμή της απόστασης μεταξύ τους. Στη συνάρτηση αυτή βασίζεται όλη η λειτουργικότητα του ΚΠΚ: Ορίζει τη διάταξη και το σχήμα του εικονικού χώρου και αντίστροφα. Το πρωτόκολλο φροντίζει κάθε κόμβος να αναλαμβάνει την αποθήκευση και τη διαχείριση των αιτήσεων εντοπισμού των "κοντινών" του τιμών.

Όταν ζητηθεί σε κάποιον κόμβο του δικτύου να εντοπίσει ένα κλειδί το οποίο δεν αποθηκεύει ο ίδιος τοπικά, αυτός πρέπει να προωθήσει το αίτημα σε κάποιον άλλο ομότιμό του. Προκειμένου τελικά να μεταφερθεί η αίτηση πιο "κοντά" στη ζητούμενη διεύθυνση, κάθε συμμετέχων διατηρεί έναν πίνακα δρομολόγησης ο οποίος αποθηκεύει την τοποθεσία άλλων κόμβων στο πραγματικό δικτυακό υπόστρωμα. Έτσι, οι διασυνδέσεις μεταξύ των κόμβων σε ένα ΚΠΚ πραγματοποιούνται μέσω των καταχωρήσεων που έχει κάθε κόμβος στον τοπικό του πίνακα δρομολόγησης. Πρακτικά, ο πίνακας δεν μπορεί να έχει απεριόριστο μέγεθος, οπότε και ο κάθε ομότιμος διατηρεί περισσότερες πληροφορίες δρομολόγησης για τις "γειτονιές" του δικτύου που είναι πλησιέστερες σε αυτόν και λιγότερες όσο αυξάνει η απόσταση. Στις περισσότερες υλοποιήσεις ΚΠΚ, οι πίνακες δρομολόγησης αποθηκεύουν ένα συγκεκριμένο αριθμό από καταχωρήσεις για κάθε εκθετικά αυξανόμενο διάστημα απόστασης. Η δρομολόγηση σε ένα κλειδί που αντιστοιχεί σε κάποιο δεδομένο, ολοκληρώνεται σε λογαριθμικό αριθμό βημάτων σε σχέση με το πλήθος των κόμβων του δικτύου. Λόγω της μορφής των πινάκων δρομολόγησης, η απόσταση μεταξύ του κόμβου που καταλήγει η αίτηση και του κλειδιού μειώνεται εκθετικά σε κάθε βήμα προώθησης, όπως φαίνεται και στο σχήμα 2.3. Μια αποτυχημένη πράξη εντοπισμού, ορίζεται ως η περίπτωση που δεν υπάρχουν άλλες επιλογές δρομολόγησης και το κλειδί δεν έχει βρεθεί.



Σχήμα 2.3: *Εντοπίζοντας ένα κλειδί σε ένα δομημένο ΔΟΚ*

Μάλιστα, τα περισσότερα σχετικά συστήματα δεν χρησιμοποιούν αναδρομή, αλλά επανάληψη, με την έννοια ότι ένας κόμβος αναλαμβάνει να κάνει όλες τις σχετικές αιτήσεις και να συλλέξει τις απαντήσεις για ένα ερώτημα. Οι ομότιμοι που δέχονται την αίτηση, απαντούν είτε με το αποτέλεσμα, είτε με πληροφορίες δρομολόγησης. Η διαδικασία της επανάληψης έχει δύο βασικά προτερήματα: Αφενός, δεν χρειάζεται να αποθηκεύεται η πληροφορία της αντίστροφης

δρομολόγησης στο κάθε αίτημα, όπως γίνεται στη περίπτωση των μη δομημένων ΔΟΚ. Έτσι διατηρούνται μικρά τα μηνύματα και αποφεύγονται οι κύκλοι δρομολόγησης. Αφετέρου, μέσω της επαναληπτικής διαδικασίας εντοπισμού, ο κόμβος που κάνει την αίτηση μαθαίνει συνεχώς πληροφορίες για άλλους ομότιμούς του. Έτσι, ανανεώνονται συχνά οι πίνακες δρομολόγησης εντός των κόμβων, και ωφελείται το σύνολο του δικτύου, αφού διαγράφονται οι προβληματικοί συμμετέχοντες. Από την άλλη μεριά, η αναδρομική δρομολόγηση έχει προταθεί ως λύση στο πρόβλημα της διατήρησης της ανωνυμίας των κόμβων [HW02].

Χαρακτηριστικά παραδείγματα υλοποιήσεων ΚΠΚ, αποτελούν τα Chord [SMK+01], Tapestry [HKRZ02] και Kademlia [MM02]. Στο Chord, αμφότεροι οι κόμβοι και τα δεδομένα αντιστοιχίζονται σε θέσεις πάνω σε έναν ιδεατό κύκλο. Η συνάρτηση εγγύτητας ορίζεται ως η απόσταση πάνω στον κύκλο. Κάθε δεδομένο αποθηκεύεται στον επόμενο κόμβο, ενώ ο κάθε συμμετέχων αποθηκεύει στον πίνακα δρομολόγησής του έναν κόμβο σε απόσταση μισό κύκλο από αυτόν, έναν σε απόσταση ενός τετάρτου του κύκλου, έναν σε απόσταση ενός ογδόου, κτλ. Από όποιον κόμβο και αν ζητηθεί οποιοδήποτε στοιχείο, ο εντοπισμός θα ολοκληρωθεί σε ελάχιστα βήματα, αφού κάθε φορά το αίτημα θα προωθείται σε κάποιον ομότιμο που βρίσκεται τουλάχιστον στα μισά της απόστασης προς το ζητούμενο κλειδί.

Από την άλλη μεριά, το Tapestry, βασίζεται στην αντιστοίχιση κόμβων και δεδομένων με μια αριθμητική συμβολοσειρά. Σε κάθε βήμα εντοπισμού, γίνεται "διόρθωση" ενός ψηφίου, ώστε οι αιτήσεις να προωθηθούν προς τον κατάλληλο κόμβο. Για παράδειγμα, αν ο κόμβος με διεύθυνση 45678 κληθεί να απαντήσει για το αντικείμενο 45789, θα προωθήσει το αίτημα σε κάποιον ομότιμό του που να έχει διεύθυνση με ένα ακόμη ψηφίο κοινό με το ζητούμενο κλειδί — π.χ. τον 45712. Για το σκοπό αυτό, κάθε κόμβος αποθηκεύει στον πίνακα δρομολόγησής του, ένα συγκεκριμένο πλήθος κόμβων για κάθε πρόθεμα διευθύνσεων που διαφέρει ένα ψηφίο από τη δική του διεύθυνση. Στο παραπάνω παράδειγμα, ο κόμβος 45678, διατηρεί επαφές για κόμβους με κλειδιά που αρχίζουν από 1, 2, 3, ... 41, 42, 43, 44, 46, ... 451, 452, κτλ.

Οι Balakrishnan, et al. παρέχουν μια συνοπτική περιγραφή των δομών και αλγορίθμων που χρησιμοποιούνται από διάφορες υλοποιήσεις δομημένων δικτύων ομότιμων κόμβων [BKK+03]. Στη συνέχεια της διατριβής, χρησιμοποιείται κυρίως το πρωτόκολλο Kademlia, το οποίο παρουσιάζεται παρακάτω.

2.2 Το πρωτόκολλο Kademlia

Το Kademlia είναι ένα δομημένο ΔΟΚ, που χρησιμοποιεί ως συνάρτηση εγγύτητας το "αποκλειστικό ή" (XOR). Έτσι δημιουργείται ένας ιδεατός χώρος που έχει τη μορφή ενός δυαδικού δέντρου, με τους κόμβους και τα δεδομένα να τοποθετούνται στα φύλλα της δομής. Η συνάρτηση XOR, αφού καταγράφει τις διαφορές δύο αριθμών σε δυαδικό επίπεδο, έχει το πρακτικό αποτέλεσμα ότι καταδεικνύει άμεσα πόσα "υποδέντρα" απέχουν δύο κλειδιά. Έτσι, αν για παράδειγμα το πρώτο δυαδικό ψηφίο του αποτελέσματος είναι "αναμμένο", οι δύο διευθύνσεις εισόδου βρίσκονται η μία στο πρώτο αριστερό υποδέντρο και η άλλη στο δεξί. Οι συμμετέχοντες

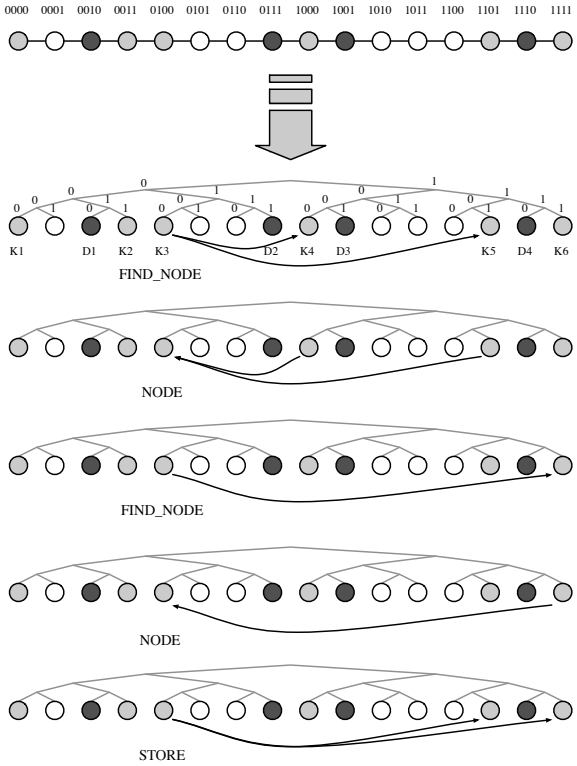
του Kademlia διατηρούν στους πίνακες δρομολόγησής τους έναν προκαθορισμένο αριθμό από κόμβους για εκθετικά αυξανόμενα διαστήματα αποστάσεων από τους ίδιους, δηλαδή τον ίδιο αριθμό για κάθε υποδέντρο "μακρύτερα" από αυτό στο οποίο ανήκουν.

Το πρωτόκολλο χαρακτηρίζεται κυρίως από το ότι χρησιμοποιεί τον ίδιο αλγόριθμο κατά τη διαδικασία εντοπισμού/ανάκτησης δεδομένων και ότι διαχειρίζεται εγγενώς τη διαδικασία τήρησης πολλαπλών αντιγράφων για κάθε αποθηκευμένη τιμή. Επίσης, οι κόμβοι του υποστηρίζουν την πολλαπλή, παράλληλη αποστολή μηνυμάτων σε κάθε βήμα μιας αίτησης (μέχρι α), το οποίο συντελεί στη μείωση του συνολικού απαιτούμενου χρόνου ολοκλήρωσης κάθε ερώτησης και βοηθά στην αναγνώριση και αποφυγή προβληματικών κόμβων που μπορεί να καθυστερήσουν τη διαδικασία. Μια δεύτερη καθολική παράμετρος, η κ, ορίζει τον αριθμό των αντιγράφων κάθε αντικειμένου και ελέγχει το μέγεθος των πινάκων δρομολόγησης.

Συνολικά, μόλις τρεις εντολές (απομακρυσμένες κλήσεις συνάρτησης - ΑΚΣ) χρησιμοποιούνται για κάθε πράξη τοποθέτησης και ανάκτησης δεδομένων: `FIND_NODE`, `FIND_VALUE` και `STORE`. Για να αποθηκεύσει ένα αντικείμενο, ένας κόμβος πρέπει πρώτα να εντοπίσει τους πλησιέστερους ομότιμούς του στο αντίστοιχο κλειδί. Έτσι, ξεκινώντας από τον δικό του πίνακα δρομολόγησης, φτιάχνει μια λίστα με τους κοντινότερους κόμβους, την οποία και χρησιμοποιεί για να στείλει παράλληλα, ασύγχρονα μηνύματα `FIND_NODE` στις πρώτες α καταχωρήσεις. Οι κόμβοι που δέχονται μήνυμα `FIND_NODE` πρέπει να απαντήσουν με μια λίστα από τους κ κόμβους που αυτοί θεωρούν πλησιέστερους στο συγκεκριμένο κλειδί. Ο κόμβος που κάνει την αρχική ερώτηση, συλλέγει τα αποτελέσματα, τα ενσωματώνει σε μια κοινή, ταξινομημένη λίστα (με βάση την απόσταση από τη διεύθυνση του αντικειμένου) και συνεχίζει τη διαδικασία μέχρι οι κ πρώτοι της λίστας να έχουν απαντήσει. Πρακτικά, ο αρχικός κόμβος δεν περιμένει και τις α παράλληλες αιτήσεις να ολοκληρωθούν πριν συνεχίσει, αλλά παράγει ένα καινούριο μήνυμα κάθε φορά που μία από τις α ΑΚΣ επιστρέψει νέους υποψήφιους. Όταν η λίστα παγιωθεί, το ζεύγος κλειδιού-τιμής στέλνεται προς αποθήκευση στους αντίστοιχους ομότιμους μέσω κλήσεων `STORE`. Το Kademlia ορίζει ότι αυτή η διαδικασία πρέπει να επαναλαμβάνεται κάθε ώρα, αφού τα αντικείμενα λήγουν και διαγράφονται μετά από 24 ώρες αδράνειας.

Για την ανάκτηση δεδομένων από το σύστημα, χρησιμοποιείται ένας παρόμοιος βρόχος, με μηνύματα τύπου `FIND_VALUE` όμως, αντί για `FIND_NODE`. Οι απαντήσεις σε `FIND_VALUE` ΑΚΣ πρέπει να επιστρέφουν, είτε δεδομένα, είτε — αν δεν υπάρχει τέτοιο αντικείμενο αποθηκευμένο στον κόμβο — μια λίστα με τους κ κοντινότερους κόμβους στο κλειδί. Η δεύτερη περίπτωση, βοηθά τον αρχικό κόμβο να ψάξει βαθύτερα στο δίκτυο, προχωρώντας σε κάθε του βήμα κοντύτερα σε κάποιον ομότιμό του που να διαθέτει τη ζητούμενη τιμή. Η διαδικασία ανάκτησης σταματά αμέσως μόλις επιστραφούν δεδομένα ή όταν κανένας από τους κ πλησιέστερους του κλειδιού δεν έχει απαντήσει με μια τιμή. Αν το αποτέλεσμα της ανάκτησης είναι θετικό, ο κόμβος που έκανε την αναζήτηση, μπορεί να αποθηκεύσει το αποτέλεσμα και στον κοντινότερο κόμβο που δεν επέστρεψε αποτέλεσμα (μέσω μιας κλήσης `STORE`). Επίσης, όταν ένας κόμβος δεχτεί οποιαδήποτε εντολή, ελέγχει τα δεδομένα που αποθηκεύει τοπικά και στέλνει αυτά που είναι κοντύτερα στον απομακρυσμένο κόμβο από τον ίδιο. Αυτό εγγυάται ότι οι τιμές αντιγράφονται σε όλους τους κοντινούς τους κόμβους και βοηθά στο να λαμβάνουν οι κόμβοι τα δεδομένα που τους

αναλογούν όταν εισέρχονται στο δίκτυο.



Σχήμα 2.4: *Αποθήκευση τιμής σε ένα δίκτυο Kademlia. Οι κόμβοι (ανοιχτό γκρι) και τα δεδομένα (σκούρο γκρι) έχουν τοποθετηθεί στα φύλλα ενός δυαδικού δέντρου, βάση του οποίου εκτελούνται οι διαδικασίες δρομολόγησης.*

Στο απλοποιημένο παράδειγμα του δικτύου που φαίνεται στο σχήμα 2.4, αμφότεροι οι κόμβοι και τα δεδομένα έχουν αντιστοιχηθεί στο χώρο διευθύνσεων τεσσάρων δυαδικών ψηφίων. Θεωρώντας ότι $\kappa = 2$, κάθε αντικείμενο θα αποθηκεύεται τουλάχιστον στους 2 κοντινότερους κόμβους του — το D4 θα πρέπει να τοποθετηθεί στους κόμβους K5 και K6). Έστω ότι και $\alpha = 2$. Για την αποθήκευση του D4, ο K3 στέλνει δύο παράλληλα μηνύματα FIND_NODE στις επαφές που διατηρεί από όλο το δεξί υποδέντρο. Οι ομότιμοί του απαντούν με μία λίστα κόμβων, από την οποία ανακαλύπτει την ύπαρξη του K6. Για να σιγουρευτεί ότι δεν υπάρχουν και άλλοι κόμβοι ακόμη πιο κοντά στο κλειδί, ο K3 θα ζητήσει και από τον K6 νέες πληροφορίες δρομολόγησης. Ο K6 προφανώς δεν απαντά με νέα δεδομένα, οπότε και το αντικείμενο δίνεται για αποθήκευση. Να σημειωθεί ότι στο συγκεκριμένο παράδειγμα, αν ο K3 αναζητήσει το D4, είναι σίγουρο ότι θα το βρει σε ένα μόλις βήμα, αφού για όλο το δεξί υποδέντρο κρατάει δύο επαφές, άρα και οπωσδήποτε έναν τουλάχιστον από τους K5 και K6.

Στα επόμενα κεφάλαια, θα δείξουμε, πώς εμπλουτίζοντας αυτές τις "βασικές" πράξεις, μπορούν να επιτευχθούν νέες, καινοτόμες υπηρεσίες απευθείας στο επίπεδο του ΚΠΚ. Στόχος είναι η δημιουργία ενός συστήματος που μπορεί να διαχειρίζεται πιο πολύπλοκες λειτουργίες από τις απλές *put* και *get*, χωρίς όμως αυτό να επηρεάζει τόσο τα ιδιαίτερα χαρακτηριστικά του συστήματος (δυνατότητες κλιμάκωσης και ανοχής σε σφάλματα), όσο και το απλό και ανεξάρτητο περιβάλλον εκτέλεσης που παρουσιάζεται στον χρήστη.

Κεφάλαιο 3

Ενημέρωση δεδομένων

Η έρευνα που παρουσιάζεται σε αυτό το κεφάλαιο έχει ως άμεσο στόχο τον σχεδιασμό πρωτοκόλλων ΚΠΚ, που θα προσφέρουν, μέσα από τους κατάλληλους αλγορίθμους, εγγενείς πράξεις ενημέρωσης των τιμών που έχουν τοποθετηθεί στο δίκτυο των συμμετεχόντων κόμβων. Στα πλαίσια της διατριβής, αναπτύξαμε τα συγκεκριμένα βήματα που απαιτούνται για την ενσωμάτωση τεχνικών κατανεμημένης ενημέρωσης δεδομένων, υπό το πρίσμα ενός ανασφαλούς και δυναμικού δικτύου, σε μια υλοποίηση ΚΠΚ που εγγενώς και διαφανώς αντιμετωπίζει όλα τα σχετικά ζητήματα [CK07]. Οι κόμβοι του τελικού ΔΟΚ - του XOR Object Store (XOROS) - χρησιμοποιούν το σχήμα δρομολόγησης του Kademlia με διαφορετικά πρωτόκολλα τοποθέτησης και ανάκτησης δεδομένων. Μάλιστα, το πρωτόκολλο του XOROS αποτελεί φυσική συνέχεια μιας πρώτης απόπειρας που έγινε στην ίδια κατεύθυνση, κατά την οποία διερευνήσαμε κατά πόσο ένας νέος αλγόριθμος ανάκτησης δεδομένων μπορεί να οδηγήσει σε ανάλογο αποτέλεσμα [CK05]. Σε αυτή τη περίπτωση θεωρήσαμε ότι η ενημέρωση ενός αντιγράφου μπορεί να γίνει απευθείας, χρησιμοποιώντας τον ίδιο τον αλγόριθμο αποθήκευσης — απλά με την προσθήκη ενός αναγνωριστικού έκδοσης σε κάθε τιμή. Ένας νέος αλγόριθμος ανάκτησης βοηθά στη τήρηση χαλαρής συνέπειας μεταξύ των αντιγράφων μιας τιμής, καθώς οι αντίστοιχοι υπεύθυνοι κόμβοι αποθήκευσης μπορεί να αλλάζουν δυναμικά. Να σημειωθεί ότι μόνο στο XOROS λαμβάνεται υπόψη ενδεχόμενη κακόβουλη συμπεριφορά από την πλευρά των κόμβων. Στις επόμενες παραγράφους αναλύεται εις βάθος το πρόβλημα της έλλειψης πράξεων ενημέρωσης και σχολιάζονται τα αποτελέσματα προηγούμενων σχετικών εργασιών, πριν εξηγηθούν οι λεπτομέρειες σχεδιασμού και υλοποίησης των δύο προτεινόμενων λύσεων.

3.1 Το πρόβλημα της ενημέρωσης δεδομένων

Οι ΚΠΚ αποτέλεσαν τη βάση για την υλοποίηση ενός μεγάλου αριθμού από κατανεμημένες εφαρμογές, που αυτόματα κληρονομούν την ικανότητα κλιμάκωσής τους σε χιλιάδες ή ακόμα και εκατομμύρια κόμβους και την ανεκτικότητά τους σε μη αναμενόμενα σφάλματα του δικτύου ή των ίδιων των κόμβων. Για παράδειγμα, έχουν παρουσιαστεί εφαρμογές όπως κατανεμημένα συστήματα αρχείων και συστήματα διαχείρισης μεταδεδομένων, που εκμεταλλεύονται άμεσα το

υπόστρωμα του σχηματιζόμενου δομημένου ΔΟΚ ως μια κατανεμημένη υποδομή αποθήκευσης. Παρόλα αυτά, καμία υλοποίηση ΚΠΚ δεν προσφέρει τη δυνατότητα ανανέωσης των αποθηκευμένων δεδομένων. Για να διασφαλιστεί η διαθεσιμότητά τους, οι τιμές αντιγράφονται αυτόματα σε ένα σύνολο ομότιμων, το οποίο μπορεί να αποφασίζεται εσωτερικά — από το ίδιο το πρωτόκολλο του ΚΠΚ — ή μέσω εξωτερικών αλγορίθμων τοποθέτησης [GAH05]. Η διαδικασία συνεπούς μεταβολής όλων των σχετικών αντιγράφων μιας τιμής είναι εξαιρετικά περίπλοκη, καθώς οι αντίστοιχοι κόμβοι που πρέπει να διαχειριστούν την αλλαγή μπορεί να αλλάζουν συνέχεια λόγω της δυναμικής φύσης του δικτύου.

Η δυνατότητα ενημέρωσης όμως μπορεί να φανεί εξαιρετικά χρήσιμη — ειδικά για τις εφαρμογές που μπορούν να κωδικοποιούν εσωτερική για αυτές σημασιολογία σε συγκεκριμένες διευθύνσεις. Για παράδειγμα, σε ένα κατανεμημένο σύστημα αρχείων που λειτουργεί πάνω από ένα δομημένο ΔΟΚ, ορισμένα κλειδιά του ΚΠΚ θα μπορούσαν να αντιπροσωπεύουν εγγενή αντικείμενα, όπως δείκτες σε δομές τύπου i-node. Αντίστοιχα, οι ανανεώσιμες τιμές θα επέτρεπαν την υλοποίηση δομών δεδομένων, όπως δέντρα, λίστες ή ουρές πάνω από έναν ΚΠΚ. Τα στοιχεία των δομών αυτών, θα μπορούσαν να αποτελούνται από αντικείμενα του ΚΠΚ που αποθηκεύουν δείκτες σε προαποφασισμένες διευθύνσεις. Κάθε εφαρμογή μπορεί να αξιοποιήσει ένα δομημένο ΔΟΚ που μπορεί να διαχειριστεί ενημερώσεις στα δεδομένα του, αρκεί να βρει νέους τρόπους να αντιστοιχίζει τα κλειδιά με τις σχετικές τους τιμές. Στις περισσότερες υλοποιήσεις μέχρι στιγμής, τα κλειδιά παράγονται αυτόματα από την ίδια την τιμή μέσω μιας συνάρτησης κατακερματισμού. Με αυτό τον τρόπο όμως, αν αλλάξει η τιμή, αλλάζει και το αντίστοιχο κλειδί.

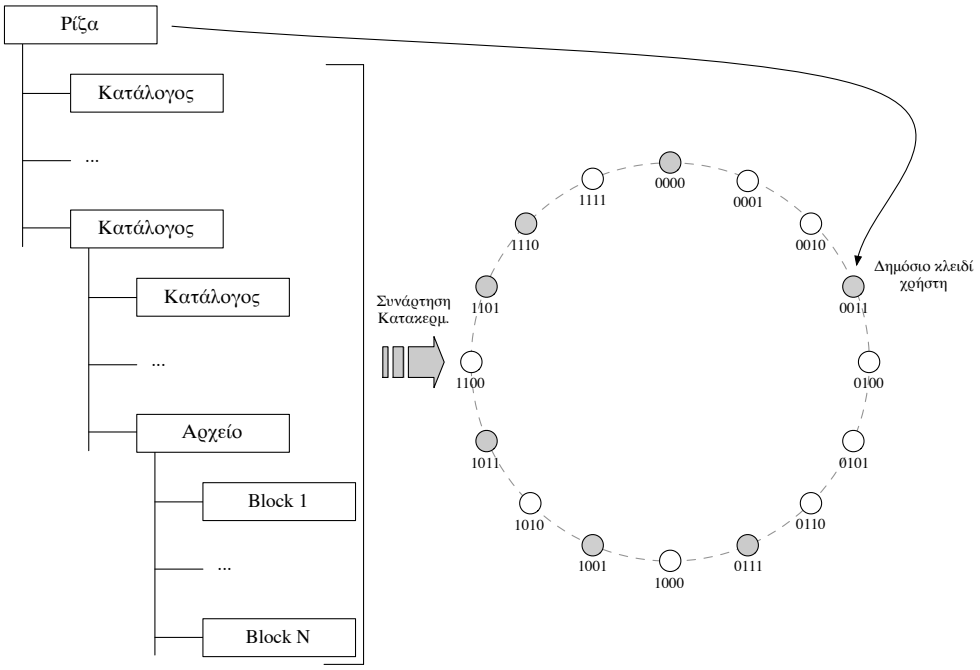
Αναδεικνύουμε το πρόβλημα της έλλειψης εγγενούς υποστήριξης ανανεώσιμων δεδομένων από ΚΠΚ, μέσα από την περιγραφή της λειτουργικότητας κατανεμημένων εφαρμογών σχετικών με τη διαχείριση μεγάλων όγκων πληροφορίας σε παγκόσμιο επίπεδο. Οι εφαρμογές αυτές, στηρίζονται πάνω σε δομημένα ΔΟΚ για την κατανομή είτε των δεδομένων, είτε του φόρτου εργασίας, ή ακόμη και των δύο ταυτόχρονα. Η παροχή μιας κοινής εικόνας της πληροφορίας, μέσα από το κατανεμημένο σύστημα, αναλαμβάνεται από το επίπεδο της εφαρμογής. Σε όλα αυτά τα συστήματα, είτε χρησιμοποιείται κάποιο "τέχνασμα" για την αποφυγή του προβλήματος, είτε απλά θεωρείται ότι το πρόβλημα δεν υπάρχει, κάτι το οποίο μπορεί να προκαλέσει δυσάρεστες εκπλήξεις στους χρήστες που περιμένουν μια συγκεκριμένη λειτουργικότητα. Σημαντικό είναι ότι αν υπήρχε ένας ΚΠΚ με τη δυνατότητα ανανέωσης δεδομένων, οι εφαρμογές αυτές θα λειτουργούσαν χωρίς καμία αλλαγή πάνω από τη νέα υποδομή αποθήκευσης.

Το Cooperative File System — CFS [DKK+01], είναι ένα παράδειγμα κατανεμημένου συστήματος αρχείων που χρησιμοποιεί έναν ΚΠΚ για να αποθηκεύει τη κατάσταση και τα δεδομένα του. Λειτουργεί όπως ένα "κλασσικό" σύστημα αρχείων, όμως αντί να τοποθετεί τις πληροφορίες καταλόγων και τα αρχεία σε κάποιο μαγνητικό μέσο, εκμεταλλεύεται, μέσω του DHash, τον κατανεμημένο "δακτύλιο" που σχηματίζουν οι κόμβοι ενός δικτύου Chord. Το DHash [Cat03] είναι ένα σύνολο αλγορίθμων που αναλαμβάνει τις λεπτομέρειες αποθήκευσης δεδομένων πάνω από έναν ΚΠΚ. Χρησιμοποιώντας *κωδικοποίηση ανοχής σε διαγραφές* (erasure coding) τοποθετεί για κάθε block πληροφορίας, πολλαπλά κωδικοποιημένα νέα block σε διάφορους κόμβους του ΔΟΚ. Όταν ζητηθούν τα περιεχόμενα μιας διεύθυνσης, το DHash λειτουργεί αντίστροφα, ανα-

κατασκευάζοντας το ζητούμενο block από τα διαθέσιμα κωδικοποιημένα δεδομένα στο δίκτυο. Εν τέλει, η διεπαφή διαχείρισης δεδομένων με τους κόμβους του ΚΠΚ, όπως φαίνεται από το επίπεδο της εφαρμογής, είναι αντίστοιχη ενός μαγνητικού δίσκου: Επιτρέπεται η εισαγωγή και ανάγνωση block δεδομένων. Κάθε block έχει μια μοναδική διεύθυνση, που όμως στη περίπτωση του ΚΠΚ ορίζεται από τα ίδια τα δεδομένα εντός του block, μέσω της συνάρτησης κατακερματισμού. Κάθε φορά που αλλάζουν τα δεδομένα ενός block, θα πρέπει να εισέρχονται στον ΚΠΚ με τη νέα τους διεύθυνση, όπως αυτή προκύπτει από την αντίστοιχη συνάρτηση. Αφού, λοιπόν, δεν υποστηρίζεται η λειτουργία ενημέρωσης στα δεδομένα ενός κλειδιού, το CFS αναφέρεται ως σύστημα αρχείων ανάγνωσης-μόνο (read-only). Μπορεί να χρησιμοποιηθεί κυρίως για τη δημιουργία στατικών, μη-ενημερώσιμων δομημένων συλλογών αρχείων, όπου το σύνολο της πληροφορίας τοποθετείται μια φορά στο δίκτυο και μετά δεν αλλάζει.

Παρόλα αυτά, οι σχεδιαστές του CFS συζητούν μια διαδικασία ενημέρωσης δεδομένων, σύμφωνα με την οποία μια αλλαγή στα δεδομένα ενός αρχείου δημιουργεί ένα αναδρομικό κύμα μεταβολών μέχρι τη ρίζα του δέντρου που σχηματίζει το σύστημα αρχείων. Κάθε φορά που αλλάζει κομμάτι ενός αρχείου, θα πρέπει να ενημερώνονται και τα μεταδεδομένα του, αφού θα έχει αλλάξει και η διεύθυνση του κομματιού. Με τη σειρά της, η αλλαγή της περιγραφής του αρχείου θα δημιουργήσει μια νέα διεύθυνση αποθήκευσης για τα αντίστοιχα δεδομένα, κ.ο.κ. μέχρι την ενημέρωση των μεταδεδομένων του κύριου καταλόγου. Ειδικά για τη δεικτοδότηση του block που αποθηκεύονται οι πληροφορίες για τον πρώτο κατάλογο του συστήματος αρχείων, προτείνεται η χρήση ενός συγκεκριμένου, προαποφασισμένου κλειδιού, που αντιστοιχεί στο *δημόσιο κλειδί* (public key) του χρήστη στον οποίο ανήκει ο κατάλογος. Η χρήση ενός δημόσιου κλειδιού στο παράδειγμα του CFS δεν έχει ιδιαίτερη σημασία — βαρύτητα έχει το γεγονός ότι επιχειρείται η ανανέωση δεδομένων, μέσω της απευθείας επαναεισαγωγής στο επίπεδο του ΚΠΚ χρησιμοποιώντας το ίδιο κλειδί. Είναι προφανές ότι σε ένα στατικό ΔΟΚ η πρακτική αυτή δεν μπορεί να προκαλέσει κάποιου είδους πρόβλημα. Με ακριβώς τον ίδιο τρόπο που έγινε η πρώτη εισαγωγή του κλειδιού, είναι δυνατόν να γίνουν οσεσδήποτε μεταγενέστερες όμοιες πράξεις στην ίδια διεύθυνση. Ακόμα και στη περίπτωση που διατηρούνται αντίγραφα της τιμής, δεδομένου ότι οι κοντινότεροι κόμβοι θα παραμένουν οι ίδιοι, θα αλλάξουν σε ένα βήμα όλες οι σχετικές εμφανίσεις του κλειδιού.

Στη πραγματικότητα όμως, ο λόγος που στους ΚΠΚ δημιουργούνται και συντηρούνται πολλαπλά αντίγραφα μιας τιμής, είναι ότι το δίκτυο έχει δυναμικό χαρακτήρα και οι πλησιέστεροι σε μία διεύθυνση κόμβοι (άρα και αυτοί που αποθηκεύουν τα αντίγραφά της) αλλάζουν συνεχώς. Σε ένα δυναμικό δίκτυο, μια *ad hoc* επαναεισαγωγή ενημερωμένων δεδομένων με το ίδιο κλειδί, θα έχει απροσδιόριστα αποτελέσματα. Η αλλαγή μπορεί να φτάσει σε κάποιους από τους κόμβους που αποθηκεύουν αντίγραφα της τιμής και ενδεχομένως και σε κάποιους νέους κόμβους που μόλις μπήκαν στο δίκτυο. Έτσι, μπορεί ανά πάσα στιγμή να υπάρχουν πολλαπλές διαφορετικές "εκδόσεις" του ίδιου κλειδιού, με διαφορετικά περιεχόμενα. Αν ένας κόμβος χαθεί για λίγα δευτερόλεπτα από το δίκτυο και μετά ξαναεμφανιστεί, μπορεί να μην λάβει μια ενημέρωση. Σε μια μεταγενέστερη πράξη εντοπισμού, μπορεί να εμφανιστεί οποιαδήποτε έκδοση. Ποιά έκδοση θα περιέχει τις τελευταίες αλλαγές; Ποιά εκδοχή θα είναι η σωστή;

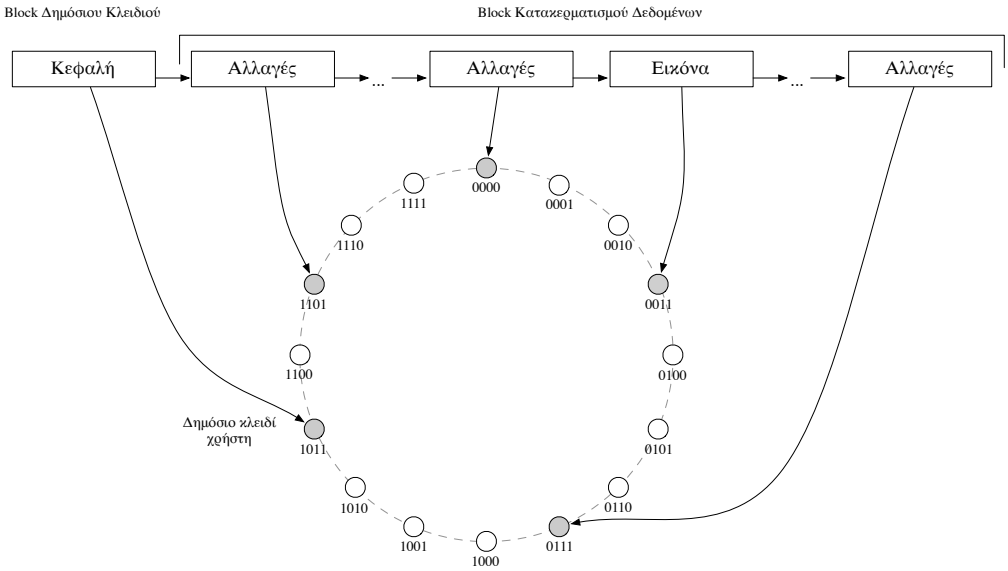


Σχήμα 3.1: *Στο CFS όλα τα δεδομένα και μεταδεδομένα του συστήματος αρχείων τοποθετούνται σε ένα ΔΟΚ μέσω της συνάρτησης κατακερματισμού, εκτός της ρίζας, που έχει ως διεύθυνση το δημόσιο κλειδί του αντίστοιχου χρήστη*

Αντίστοιχη πρακτική με το CFS, ακολουθεί και το κατανεμημένο σύστημα αρχείων Pastis [BPS05], μόνο που όλα τα block μεταδεδομένων έχουν σταθερή διεύθυνση, οπότε δεν απαιτείται η αλλαγή όλων των καταλόγων κάθε φορά που μεταβάλλεται ένα αρχείο. Τις λεπτομέρειες για το πώς ενημερώνονται με μία πράξη όλα τα δυναμικά αντίγραφα μιας τιμής στο δίκτυο, αναλαμβάνει το υπόστρωμα αποθήκευσης, που αποτελείται από το Pastry [RD01] και το PAST [DR01] (σε αντιστοιχία με τα Chord και DHash), αν και στη σχετική βιβλιογραφία δεν αναφέρονται ειδικοί αλγόριθμοι που να επιλύουν τα προβλήματα συνέπειας που μπορεί να προκύψουν. Έτσι, υποθέτουμε ότι και το Pastis επηρεάζεται από τα ίδια ζητήματα που καθιστούν το CFS μια καλή επιλογή για δεδομένα ανάγνωσης-μόνο.

Η απουσία μιας λειτουργίας ενημέρωσης έχει οδηγήσει την έρευνα σε σχεδιαστικές πρωτοβουλίες που αποφεύγουν τον περιορισμό. Το Ivy [MMGC02] είναι ένα *επανεγγράψιμο* σύστημα αρχείων, το οποίο λειτουργεί πάνω από το DHash. Για να λυθεί το ζήτημα των ενημερώσεων στην υποδομή ανάγνωσης-μόνο που προσφέρεται από τον ΚΠΚ, χρησιμοποιείται η ιδέα της "αλυσίδας" ή "ουράς" αλλαγών: Κάθε χρήστης του συστήματος αποθηκεύει σε αναγνώσιμη-μόνο μορφή τις αλλαγές που έχει κάνει στο σύστημα, σε μια αλυσίδα (log). Με το που γίνεται μια ενημέρωση, προστίθεται ένας νέος "κρίκος" στην αλυσίδα, ο οποίος περιγράφει την αλλαγή μαζί με έναν δείκτη στον αμέσως προηγούμενο κρίκο. Έτσι, για να βρεθεί η τελική μορφή ενός αρχείου, το σύστημα επισκέπτεται όλες τις σχετικές καταχωρήσεις, ξεκινώντας από τη τελευταία εισαγωγή (νεότερη) και προχωρώντας προς της προηγούμενες (παλιότερες). Είναι προφανές ότι απαιτείται και μια "κεφαλή" της αλυσίδας που να έχει τη δυνατότητα να αλλάζει κάθε φορά ώστε να δείχνει στη πιο πρόσφατη αλλαγή. Όπως και στο CFS, αυτό υλοποιείται με ένα "ειδικό" block

το οποίο έχει ως αναγνωριστικό το δημόσιο κλειδί του αντίστοιχου χρήστη. Το πρόβλημα της ενημέρωσης της κεφαλής αφήνεται στο υπόστρωμα αποθήκευσης. Η μόνη μέριμνα από πλευράς του Ivy, είναι ότι τα block δημόσιου κλειδιού (public key blocks) δεν αποθηκεύονται σε ενδιάμεσες μνήμες (caches).



Σχήμα 3.2: *Το Ivy αποθηκεύει για κάθε χρήστη μια αλυσίδα αλλαγών στο σύστημα αρχείων*

Στο Ivy, η τελική εικόνα του συστήματος αρχείων δίνεται από ένα σύνολο τέτοιων αλυσίδων — π.χ. από τις αλυσίδες 3 χρηστών, κάτι που ονομάζεται "όψη" (view). Μάλιστα μπορούν να δημιουργούνται οσεσδήποτε όψεις, με όλους τους πιθανούς συνδυασμούς αλυσίδων. Οι χρήστες που συμφωνούν να φτιάξουν μία κοινή όψη του συστήματος, αποθηκεύουν ουσιαστικά ένα νέο αντικείμενο στον ΚΠΚ, το οποίο δεικτοδοτεί τις "κεφαλές" όλων των αλυσίδων που συμμετέχουν. Οι σχεδιαστές του συστήματος αναλύουν πως επιλύονται αντιφάσεις ή συγκρούσεις σε μια όψη του συστήματος. Επειδή ο κάθε χρήστης μπορεί να καταγράφει ανεξάρτητα τις αλλαγές που κάνει στη δική του αλυσίδα, απαιτείται ένας αλγόριθμος, βασιζόμενος σε "διανύσματα εκδόσεων" (version vectors), για να λαμβάνεται αυτόματα η απόφαση του ποιές αλλαγές θεωρούνται έγκυρες όταν εξετάζονται ταυτόχρονα πολλές αλυσίδες. Όταν πολλαπλοί χρήστες προσπελαύνουν την ίδια όψη του συστήματος αρχείων, η απόφαση αυτή πρέπει να είναι κοινή για όλους — ανεξάρτητη και συνεπής. Μάλιστα, για να μην χρειάζεται κάθε φορά να επισκέπτεται ένας χρήστης όλη την αλυσίδα αλλαγών (τη δική του ή μιας όψης), φυλάσσονται ανά τακτά χρονικά διαστήματα "εικόνες" (snapshots) της τελευταίας συνολικής κατάστασης του συστήματος. Η προσπέλαση στην αλυσίδα γίνεται μέχρι να βρεθεί η πιο πρόσφατη εικόνα. Και οι εικόνες αποθηκεύονται στον ΚΠΚ.

Τα CFS, Pastis και Ivy θεωρούν ότι με κάποιον τρόπο (παρόλο που αναφέρεται ότι προς το παρόν αυτό δεν ισχύει), ο ΚΠΚ μπορεί και προσφέρει συνέπεια εγγραφής-ανάγνωσης (read-write consistency). Βασίζονται, δηλαδή, στο ότι αν μια πράξη εγγραφής για ένα κλειδί ολοκληρωθεί επιτυχώς, οποιαδήποτε επόμενη ανάγνωση του κλειδιού θα επιστρέψει τα δεδομένα της εγγραφής. Τέτοιες εγγυήσεις όμως δεν δίνονται στη πράξη από κανέναν "κλασσικό" αλγόριθμο ΚΠΚ.

Περαιτέρω αναλύσεις περί συνέπειας δεδομένων στο επίπεδο του συστήματος αρχείων, είναι μεν χρήσιμες, αλλά δεν έχουν πρακτική υπόσταση. Σε όλα αυτά τα συστήματα αρχείων, οι αλλαγές μπορεί να γίνονται σε πολλά βήματα, αλλά η παγίωση της νέας κατάστασης σηματοδοτείται από την ενημέρωση ενός και μόνο κλειδιού: Στο CFS του block μεταδεδομένων του κύριου καταλόγου, στο Pastis του αντίστοιχου block μεταδεδομένων καταλόγου και στο Ivy της κεφαλής της αλυσίδας αλλαγών. Αυτή η τελευταία φάση, που θυμίζει την ολοκλήρωση μιας "συναλλαγής" (transaction), πρέπει να μπορεί να ολοκληρώνεται επιτυχώς ή όχι. Δεν μπορεί να υπάρχει πιθανότητα το σύστημα να δώσει άλλη τιμή για την κεφαλή μιας αλυσίδας σε έναν χρήστη του Ivy και άλλη σε κάποιον άλλο που προσπελαύνει την ίδια όψη του συστήματος αρχείων. Το πρόβλημα της ενημέρωσης, λοιπόν, είναι ουσιώδες. Μόνο τότε θα μπορέσουν τα παραπάνω συστήματα να προσφέρουν τη λειτουργικότητα που προδιαγράφουν στους χρήστες τους. Το Eliot [STS02], αποτελεί μια αντίστοιχη υλοποίηση συστήματος αρχείων, που αποθηκεύει τα αναγνώσιμα-μόνο blocks των αρχείων σε έναν ΚΠΚ, αλλά χρησιμοποιεί κεντρικούς εξυπηρετητές που διαχειρίζονται τις μεταβολές στα μεταδεδομένα και τους δείκτες στα έγκυρα κλειδιά εντός του ΔΟΚ. Έτσι, λύνεται μεν το πρόβλημα της συνέπειας των μεταδεδομένων, αλλά επανεμφανίζονται τα θέματα κλιμάκωσης που έχει μια λύση που εξαρτάται από κάποιο κεντρικά τοποθετημένο πόρο.

Εκτός από τις περιπτώσεις κατανεμημένων συστημάτων αρχείων, που λειτουργούν πάνω από ΚΠΚ, ιδιαίτερο ενδιαφέρον παρουσιάζουν και κάποιες τεχνικές που χρησιμοποιούνται από το πρωτόκολλο κατανεμημένης μεταφοράς αρχείων BitTorrent [Coh03]. Σημασία δεν έχουν οι λεπτομέρειες του ίδιου του αλγορίθμου μεταφοράς, αλλά το ότι απαιτείται από το πρωτόκολλο μια μέθοδος διαρκούς παρακολούθησης και καταγραφής των κόμβων που έχουν το σύνολο ή μέρη κάθε αρχείου. Για το σκοπό αυτό, οι κόμβοι του BitTorrent χρησιμοποιούν έναν κεντρικό εξυπηρετητή "παρακολούθησης" (tracker) για κάθε ομάδα αρχείων, τον οποίο ενημερώνουν ανά τακτά χρονικά διαστήματα για το ποσοστό της μεταφοράς που έχουν ολοκληρώσει. Κάποιες νέες υλοποιήσεις ενσωματώνουν ένα άλλο πρωτόκολλο παρακολούθησης, που στηρίζεται σε έναν ΚΠΚ [BTD]. Δεδομένου ότι το BitTorrent ορίζει ένα αναγνωριστικό 160 δυαδικών ψηφίων για κάθε αρχείο (που προκύπτει από τα περιεχόμενά του), εντοπίζονται με βάση το αναγνωριστικό αυτό οι αντίστοιχοι κοντινότεροι κόμβοι από τον ΚΠΚ. Αυτοί οι κόμβοι θα πρέπει να επιτελούν και τη παρακολούθηση του συγκεκριμένου αρχείου.

Στη περίπτωση του BitTorrent, ο ΚΠΚ χρησιμοποιείται για την εύρεση κόμβων και όχι την αποθήκευση δεδομένων. Έτσι, κατανέμεται η διαδικασία παρακολούθησης των αρχείων στους συμμετέχοντες, αλλά χωρίς να αξιοποιούνται τα χαρακτηριστικά ανοχής σε σφάλματα που παρέχει το ΔΟΚ. Αν κάποιος κόμβος φύγει από το δίκτυο, μαζί του χάνονται προς στιγμήν όλα τα δεδομένα παρακολούθησης (tracker data). Από την άλλη μεριά, αν υπήρχε η δυνατότητα να αποθηκεύονται στο ΚΠΚ αυτές οι πληροφορίες, δεν θα επηρεαζόταν η διαθεσιμότητά τους από τη διαθεσιμότητα των κόμβων. Μάλιστα, αφού το κάθε αρχείο έχει ένα αναγνωριστικό στον ίδιο χώρο διευθύνσεων με του ΚΠΚ, θα μπορούσαν τα δεδομένα να διευθυνσιοδοτούνται απευθείας με βάση τον αριθμό αυτό. Επιπρόσθετα, αντίστοιχα προβλήματα τήρησης μιας κατανεμημένης βάσης παρακολούθησης ή "δεικτοδότησης", αντιμετωπίζονται και από υποσυστήματα της ομάδας πρωτοκόλλων που υποστηρίζουν τις ανάγκες διαχείρισης δεδομένων ενός Υπερυπολογιστικού

Πλέγματος (Grid).

Ένα *Πλέγμα* είναι ένα κατανεμημένο υπολογιστικό σύστημα μεγάλης κλίμακας, στο οποίο ενοποιούνται απομακρυσμένες, ανεξάρτητες και διαφορετικής τεχνολογίας μονάδες αποθήκευσης και επεξεργασίας δεδομένων, μέσω μιας κοινής αρχιτεκτονικής λογισμικού που αποτελείται από *ηλεκτρονικές υπηρεσίες* [FK99, FKT01]. Ένα σύνολο τέτοιων βασικών υπηρεσιών αναλαμβάνει την κατανεμημένη αποθήκευση και διαχείριση μεγάλων όγκων δεδομένων, προσφέροντας στις υπόλοιπες υπηρεσίες υψηλότερου επιπέδου και στις εφαρμογές γνωστές διεπαφές, που περιλαμβάνουν αρχεία, καταλόγους, αναζήτηση στα μεταδεδομένα, κτλ. Από τα σημαντικότερα συστατικά στοιχεία της αρχιτεκτονικής "Πλέγματος Δεδομένων" (DataGrid architecture), είναι η Υπηρεσία Εντοπισμού Αντιγράφων (ΥΕΑ), η οποία αναλαμβάνει την παρακολούθηση και καταγραφή των τοποθεσιών που έχουν αντίγραφα ενός αρχείου [CFK+00]. Σε ένα Πλέγμα, τα δεδομένα που χρησιμοποιούνται μπορεί να είναι φυσικά αποθηκευμένα σε διάφορους κόμβους, κυρίως διότι το μέγεθός τους δεν επιτρέπει την αντιγραφή τους σε όλες τις τοποθεσίες πρόσβασης. Όταν μία υπηρεσία ή εφαρμογή χρειαστεί κάποια αρχεία, πρέπει να τα μεταφέρει πρώτα σε τοπικό επίπεδο, προκειμένου να τα επεξεργαστεί. Έτσι δημιουργούνται αυτόματα πολλαπλά αντίγραφα του συνόλου της πληροφορίας στα διάφορα υπολογιστικά και αποθηκευτικά συστήματα που συμμετέχουν στο Πλέγμα. Όταν η πρόσβαση σε ένα απομακρυσμένο αρχείο χρειάζεται να γίνει πολλές φορές, ένα τοπικό αντίγραφο είναι προφανές ότι συμβάλλει στο να μειώνεται ο χρόνος προσπέλασης και να περιορίζεται η άσκοπη χρήση του δικτύου. Η ύπαρξη πολλών αντιγράφων, μπορεί να βοηθήσει και στην ταχύτερη μεταφόρτωση αρχείων προς έναν κόμβο του Πλέγματος, με τη προϋπόθεση ότι αξιοποιούνται πολλαπλές οδεύσεις μεταφοράς μέσω κατάλληλων αλγορίθμων. Επίσης, σε περίπτωση ακούσιας διαγραφής ή σφάλματος στο μέσο αποθήκευσης ενός αντιγράφου, τα δεδομένα συνεχίζουν να υπάρχουν και μπορούν να ανακατασκευαστούν στην αρχική τοποθεσία.

Οι υλοποιήσεις της ΥΕΑ, παρόλο που έχουν εξελιχθεί σε σημαντικό βαθμό τα τελευταία χρόνια, παρουσιάζουν ακόμη προβλήματα κλιμάκωσης και ανοχής σε σφάλματα. Η πιο πρόσφατη λύση που ακολουθείται — το Giggle Framework [CDF+02, CPB+04], χρησιμοποιεί καταλόγους εντοπισμού, οργανωμένους σε ιεραρχίες τοπικών και υπερτοπικών εγκαταστάσεων. Έτσι όμως τα ίδια τα δεδομένα παρακολούθησης μπορεί να υπάρχουν σε πολλά αντίγραφα, οπότε απαιτούνται πολύπλοκοι αλγόριθμοι συγχρονισμού μεταξύ των εγκαταστάσεων. Όπως και στη περίπτωση του BitTorrent, μια συνολική λύση θα μπορούσε να αποτελέσει η χρήση ενός ΚΠΚ για την τοποθέτηση όλων των σχετικών δεδομένων — ΚΠΚ με δυνατότητα ενημέρωσης των αποθηκευμένων τιμών. Μπορεί μια υλοποίηση με ΚΠΚ να εισάγει μεγαλύτερη καθυστέρηση στην διαδικασία εύρεσης, λόγω των πολλαπλών μηνυμάτων που πρέπει να ανταλλάξουν μεταξύ τους οι κόμβοι, δεν παύει όμως να αποτελεί έναν τρόπο αντιμετώπισης των προβλημάτων που μπορεί να προκύψουν όταν το μοντέλο του Πλέγματος αναπτυχθεί πέρα από τα στενά όρια της ακαδημαϊκής κοινότητας και των ερευνητικών ιδρυμάτων. Ένα παγκόσμιο Πλέγμα, για να κλιμακωθεί σε εκατομμύρια χρηστών, θα πρέπει να αναθέσει και κάποιες από τις εσωτερικές του διαδικασίες στους συμμετέχοντες κόμβους. Κάποιες κεντρικές υπηρεσίες, όπως η ΥΕΑ, μπορούν να εκμεταλλευτούν τα ιδιαίτερα σημαντικά χαρακτηριστικά των δομημένων ΔΟΚ, αρκεί αυτά με τη σειρά τους να εί-

ναι σε θέση να εκτελέσουν μια πληρέστερη σειρά πράξεων στα αποθηκευμένα δεδομένα (ένα σχετικό σενάριο χρήσης περιγράφεται στο κεφάλαιο 5).

Τα πρωτόκολλα που παρουσιάζονται στη συνέχεια, μας επιτρέπουν την αναθεώρηση του σχεδιασμού των παραπάνω εφαρμογών, έτσι ώστε αυτές να χρησιμοποιούν την κατανεμημένη υποδομή αποθήκευσης και να εκμεταλλεύονται εις έπακρον τις δυνατότητες κλιμάκωσης και ανοχής σε σφάλματα του σχηματιζόμενου ΚΠΚ. Εκτός βέβαια από τα κατανεμημένα συστήματα αρχείων και τις υπηρεσίες διαχείρισης και παρακολούθησης αντιγράφων, μπορούμε να γενικεύσουμε το πλήθος των επωφελούμενων εφαρμογών, σε οποιαδήποτε σύστημα χρειάζεται να αποθηκεύει πληροφορίες που αντιστοιχίζονται σε ένα, μοναδικό κλειδί. Δομημένες συλλογές δεδομένων στις οποίες η πρόσβαση ελέγχεται με δύο ή περισσότερα κλειδιά, θα πρέπει να μετασχηματιστούν με κάποιο τρόπο σε έναν γραμμικό χώρο διευθύνσεων. Από την άλλη μεριά, αυτό πρακτικά συμβαίνει στη πλειονότητα των εφαρμογών, αφού οι σύγχρονη τεχνολογία επιβάλλει έτσι και αλλιώς την αποθήκευση των δεδομένων στις σειριακές διευθύνσεις ενός μαγνητικού ή οπτικού μέσου.

3.2 Σχετικές εργασίες

Χρειάζεται, λοιπόν, ένα καινοτόμο σύνολο αλγορίθμων και πρωτοκόλλων για ΚΠΚ που να επιτρέπουν την εγγενή, αυτόματη και σειριοποιήσιμη διαχείριση μεταβολών στα *πρωτεύοντα αντίγραφα* οποιασδήποτε τιμής που μπορεί να αποθηκεύεται εντός του συστήματος. Ορίζουμε τα πρωτεύοντα αντίγραφα ως τις τιμές που αποθηκεύονται στους πλησιέστερους κόμβους ενός κλειδιού — τους κόμβους στους οποίους θα φτάσει ένα αίτημα εντοπισμού, δεδομένου ότι δεν θα χρησιμοποιηθούν ενδιάμεσες μνήμες. Η μελέτη σχετικών εργασιών οδήγησε στο συμπέρασμα ότι μια συνεπή λειτουργία κατανεμημένης ενημέρωσης απαιτεί την υλοποίηση ενός ασφαλούς και ανεκτικού σε σφάλματα πρωτοκόλλου αμοιβαίου αποκλεισμού, απευθείας στο επίπεδο του ΚΠΚ, συνοδευόμενο από μια διαδικασία διάδοσης των δεδομένων που θα συγχρονίζει όλα τα πρωτεύοντα αντίγραφα σε μια μοναδική και ενιαία πράξη συναλλαγής. Επιπρόσθετα, η ανεκτικότητα του πρωτοκόλλου, θα πρέπει να καλύπτει και περιπτώσεις που κάποιοι κόμβοι επιδεικνύουν — εκούσια — αυθαίρετη συμπεριφορά. Η ηθελημένη προσβολή των διαδικασιών επικοινωνίας, ή αλλιώς, η εν γνώση μεταφορά εσφαλμένων μηνυμάτων, αναφέρεται και ως "Βυζαντινή" συμπεριφορά (Byzantine behavior). Τα ζητήματα αυτά έχουν αποτελέσει ενεργό ερευνητικό στόχο για πολλά χρόνια. Στις επόμενες παραγράφους αναφέρονται κάποιες από τις σημαντικότερες σχετικές εργασίες στον τομέα.

Διαδικασίες συνεχούς παρακολούθησης της θέσης όλων των κατανεμημένων αντιγράφων μιας τιμής και μέθοδοι συγχρονισμού των σχετικών αλλαγών, έχουν προταθεί για κάθε μορφής ΔΟΚ — δομημένου [RB03, CRWZ05] ή μη [LLSR02, DHA03, WDKS04]. Η βασική ιδέα πίσω από όλους τους αλγορίθμους είναι η δημιουργία ενός κατανεμημένου ευρετηρίου αντιγράφων που ενημερώνεται δυναμικά, είτε κάθε φορά που οι εντολές αναζήτησης/εντοπισμού ακολουθούν κάποιο μονοπάτι από κόμβο σε κόμβο, είτε μέσω ενός πρωτοκόλλου "εγγραφής συνδρομητών" (subscription protocol). Όταν γίνεται κάποια αλλαγή στα δεδομένα, η νέα τιμή διαχέεται σε όλα τα μέλη χρησιμοποιώντας έναν μηχανισμό προώθησης (push). Μάλιστα, για να ελαχιστοποι-

ηθεί η πιθανότητα μη ενημέρωσης κάποιων αντιγράφων, μερικά συστήματα απαιτούν και μια περιοδική φάση επανανάγνωσης (pull), όταν δεν έχουν ληφθεί αλλαγές για ορισμένο χρονικό διάστημα.

Οι παραπάνω πρακτικές απαιτούν συνεργασία μεταξύ των κόμβων που προωθούν τις ενημερώσεις, η οποία μπορεί να επιβάλλεται ως προαπαιτούμενο ή να να συντηρείται μέσω της δημιουργίας σχετικών κινήτρων. Ακόμη, θεωρείται ότι υπάρχει πάντα ένας πρωτεύων ή κύριος κόμβος ο οποίος διατηρεί την τελευταία τιμή ενός αντικειμένου και ένα σύνολο αντιγράφων που πρέπει να ενημερωθεί. Αντιθέτως, ένας ΚΠΚ με εγγενή υποστήριξη ανοχής σε σφάλματα, αντιγράφει αυτόματα τις τιμές στους πλησιέστερούς τους κόμβους, επομένως υπάρχουν πολλαπλά πρωτεύοντα αντίγραφα τα οποία πρέπει να αλλάξουν. Επιπρόσθετα, αυτά τα αντίγραφα είναι πάντα διαθέσιμα και η θέση τους είναι εκ των προτέρων γνωστή, οπότε δεν υπάρχει λόγος να παρακολουθείται η θέση τους μέσω ειδικών ευρετηρίων.

Οι αλγόριθμοι που βασίζονται σε ευρετήρια αντιγράφων δεν προσφέρουν εγγυήσεις συνέπειας. Ο συγχρονισμός των αντιγράφων έχει μεν καλή πιθανότητα να ολοκληρωθεί, αλλά δεν προβλέπεται η σειρά με την οποία θα φανούν οι αλλαγές στο δίκτυο ή τι θα συμβεί σε περίπτωση αποτυχίας μιας ενημέρωσης. Παρόλα αυτά, η τήρηση χαλαρής συνέπειας, μπορεί να εφαρμοστεί παράλληλα με πρωτόκολλα αυστηρού συγχρονισμού, προκειμένου να λυθεί το θέμα της ενημέρωσης *δευτερευόντων αντιγράφων*. Εκτός από τις πρωτεύοντες τιμές, μπορεί να αποθηκεύονται δευτερεύοντα αντίγραφα στις ενδιάμεσες μνήμες των κόμβων οπουδήποτε στο πλέγμα. Η τήρηση ενός σχετικού ευρετηρίου έχει ως βασικό στόχο την παρακολούθηση της διαθεσιμότητάς τους, προκειμένου να είναι ξεκάθαρο ποιοί κόμβοι πρέπει να λάβουν ενημερώσεις ή ειδοποιήσεις ακύρωσης για κάποια τιμή. Με αυτό το σκεπτικό, έχουμε προτείνει έναν αντίστοιχο αλγόριθμο, ο οποίος εκμεταλλεύεται τη διαδικασία ανάκτησης δεδομένων σε ένα δομημένο ΔΟΚ, ενημερώνοντας τους πλησιέστερους κόμβους για τυχόν αλλαγές και πολλαπλές εκδόσεις της ίδιας τιμής στο δίκτυο [CK05].

Ένα κατανεμημένο σύστημα το οποίο διαθέτει μηχανισμούς συντονισμού ενημέρωσης των τιμών μεταξύ πρωτευόντων και δευτερευόντων αντιγράφων είναι το Oceanstore[KBC+00]. Το Oceanstore χρησιμοποιεί ένα ανώτερο επίπεδο καλά διασυνδεδεμένων εξυπηρετητών, που αποφασίζουν τη σειρά των αλλαγών και στέλνουν τις νέες τιμές στους υπόλοιπους κόμβους μέσω ενός "επιδημικού" αλγορίθμου. Στοχεύουμε σε αντίστοιχη λειτουργικότητα, η οποία όμως να παρέχεται εντός του ΚΠΚ, όπου όλοι οι συμμετέχοντες κόμβοι έχουν τις ίδιες ικανότητες και υποχρεώσεις. Η ανάγκη υποστήριξης ατομικών λειτουργιών σε όλα τα αντίγραφα των δεδομένων στο επίπεδο ενός ΚΠΚ έχει σημειωθεί από τους Lynch *et. al* [LMR02]. Όμως, οι αλγόριθμοι που προτείνουν προϋποθέτουν ένα ασφαλές μέσο μετάδοσης μηνυμάτων και ένα δίκτυο από έμπιστους κόμβους που συμμορφώνονται πλήρως με τα σχετικά πρωτόκολλα εισόδου και εξόδου από το δίκτυο.

Πολλές σχετικές εργασίες αναφέρονται σε αλγορίθμους κατανεμημένου αμοιβαίου αποκλεισμού [Vel93], αν και μόνο το πρωτόκολλο Sigma παρουσιάζει μια μέθοδο για την υλοποίηση αμοιβαίου αποκλεισμού απευθείας στο επίπεδο των ομότιμων κόμβων ενός ΚΠΚ [LLCZ04, Zha04]. Ο

αλγόριθμος, που εντάσσεται στην κατηγορία αλγορίθμων "αδειοδότησης" (permission-based), ορίζει ότι για να "κλειδωθεί" ένας πόρος, πρέπει να δοθεί η αντίστοιχη άδεια από την πλειοψηφία των λογικών του αντιγράφων. Κάθε τέτοιο αντίγραφο, όπως συνηθίζεται στους ΚΠΚ, αντιπροσωπεύεται από μία μοναδική διεύθυνση, η οποία με τη σειρά της αντιστοιχεί σε έναν κόμβο (τον πλησιέστερο). Κάθε τέτοιος κόμβος είναι υπεύθυνος για την απόδοση μίας άδειας/ψήφου για τον αμοιβαίο αποκλεισμό. Το πρωτόκολλο καλύπτει τις περιπτώσεις που κάποιοι κόμβοι μπορεί να αποτύχουν και συνθήκες υψηλών ρυθμών ζήτησης και συναγωνισμού, αλλά στηρίζεται στο ότι τα μέλη που συμμετέχουν στη διαδικασία θα είναι έμπιστα και θα παραχωρήσουν τις άδειές τους αν είναι αδύνατο να αποδοθεί πλειοψηφία. Το Sigma θα μπορούσε να χρησιμοποιηθεί για την υλοποίηση ενός πρωτοκόλλου ενημέρωσης δεδομένων, αλλά ποιός ο λόγος να υπάρχουν διαφορετικές εντολές για τον αμοιβαίο αποκλεισμό και την ενημέρωση δεδομένων, όταν η σημασιολογία του κλειδώματος είναι εκ των προτέρων γνωστή;

Παρόλο που η μεθοδολογία αμοιβαίου αποκλεισμού που προτείνεται από το Sigma μπορεί να χρησιμοποιηθεί σε ένα ευρύ φάσμα περιπτώσεων, η υλοποίηση δεν είναι κατάλληλη για χρήση ως τμήμα ενός γενικότερου αλγορίθμου υποστήριξης ανανεώσιμων δεδομένων σε έναν ΚΠΚ. Αν χρησιμοποιήσουμε ένα διακριτό βήμα αμοιβαίου αποκλεισμού, θα πρέπει κάθε αλλαγή στα δεδομένα να αποτελείται από τη φάση συγκέντρωσης των ψήφων, την αποστολή της αλλαγής και τέλος την ελευθέρωση του αντίστοιχου πόρου. Η απεμπλοκή του πρωτοκόλλου αμοιβαίου αποκλεισμού με την διαδικασία ενημέρωσης των δεδομένων απαιτεί έναν μεγάλο αριθμό πλεοναζόντων μηνυμάτων και επιφέρει επιπλέον περιπλοκές όσον αφορά την ασφάλεια του συστήματος, σε ένα περιβάλλον το οποίο ήδη θεωρείται μη έμπιστο. Πρέπει να προστατεύεται ο μηχανισμός ενημέρωσης από κόμβους που μπορεί να προσπαθήσουν να τον εκμεταλλευτούν, είτε με το να μην εκτελούν τα απαραίτητα βήματα με την ενδεδειγμένη σειρά, είτε με το να στέλνουν αλλαγές στις τιμές χωρίς καν να έχουν πάρει τις απαραίτητες άδειες. Άρα η διαδικασία της αδειοδότησης θα πρέπει να είναι αλληλένδετη με τον μηχανισμό ανανέωσης των τιμών. Η άδεια πρέπει να δίνεται για κάποιο σκοπό και αν αυτός δεν τηρείται, τότε όχι μόνο να ακυρώνεται η πράξη, αλλά να "τιμωρείται" και ο κόμβος που προσπαθεί να προσβάλλει τους μηχανισμούς. Τέτοιες περιπτώσεις εξετάζονται διεξοδικά σε σχετικές εργασίες που μελετούν την ανταλλαγή μηνυμάτων υπό "Βυζαντινές" συνθήκες [LSP82, MMR99]. Παρόμοιους στόχους έχει και το Rosebud [RL03], ένα σύστημα αποθήκευσης που βασίζεται σε κάποιο δομημένο ΔΟΚ. Επί του σχηματιζόμενου ΚΠΚ χρησιμοποιείται ένα ανεκτικό σε "Βυζαντινές" συνθήκες πρωτόκολλο για την διαχείριση των ενημερώσεων σε αντικείμενα που ορίζονται βάση ενός δημόσιου κλειδιού. Για να γίνει μια ενημέρωση, ένας κόμβος πρέπει πρώτα να έρθει σε επαφή με την "απαρτία" των ομότιμών του που διατηρούν αντίγραφα του αντικειμένου, προκειμένου να πάρει έναν νέο αριθμό έκδοσης της σχετικής τιμής. Ύστερα μπορεί να αποστείλει μια υπογεγραμμένη αίτηση εγγραφής. Το Rosebud παρακολουθεί τους συμμετέχοντες στο δίκτυο μέσω μιας κατανεμημένης "υπηρεσίας μελών" και χρησιμοποιεί εκτενώς πιστοποιητικά κρυπτογραφίας (κάποια έμπιστη οντότητα προτείνει νέα μέλη του δικτύου). Αυτές οι πρακτικές έρχονται σε αντίθεση με τις προδιαγραφές εφαρμογών που μπορεί να έχουν ιδιαίτερα δυναμικούς κόμβους ή να απαιτούν την διατήρηση της ανωνυμίας των μελών.

3.3 Εντοπίζοντας τις αλλαγές τιμών

Η πρώτη μέθοδος που προτείναμε για τη διαχείριση ενημερώσεων στα δεδομένα ενός ΚΠΚ, διατηρεί χαλαρή συνέπεια μεταξύ των αντιγράφων που υπάρχουν στο δίκτυο. Στο νέο πρωτόκολλο παραμένει αυτούσια η διαδικασία αποθήκευσης που χρησιμοποιείται από τον ΚΠΚ, αλλά μεταβάλλεται ο τρόπος με τον οποίο εντοπίζονται τα δεδομένα. Οι μεταβολές τιμών δρομολογούνται στο ΔΟΚ ως μια "κλασσική" πράξη αποθήκευσης στους κοντινότερους κόμβους του αντίστοιχου κλειδιού, αν και επιβάλλεται η χρήση ενός αναγνωριστικού "έκδοσης" για κάθε νέα τιμή. Θεωρώντας ότι υπάρχει εμπιστοσύνη μεταξύ των κόμβων, μπορεί να βρεθεί κάποιος τρόπος παραγωγής αναγνωριστικών για κάθε έκδοση, είτε με τη παράλληλη υλοποίηση κάποιου αντίστοιχου κατανεμημένου αλγορίθμου, είτε με τη χρήση διανυσμάτων εκδόσεων. Με τον χαρακτηριστικό αλγόριθμο τοποθέτησης ενός ΚΠΚ, η αλλαγή φτάνει σε κάποιους τουλάχιστον από τους πρωτεύοντες κόμβους αποθήκευσης. Αν ένας συμμετέχων δεχτεί μια εντολή αποθήκευσης για ένα ζεύγος κλειδιού-τιμής που αποθηκεύει ήδη, θα πρέπει να ελέγξει την έκδοση της τιμής ώστε να αποφανθεί αν όντως το νέο αντικείμενο αποτελεί ενημερωμένη έκδοση του υπάρχοντος ή όχι. Οι νέες εκδόσεις πάντα αντικαθιστούν τις παλαιότερες.

Γνωρίζοντας ότι οι πλησιέστεροι κόμβοι ενός κλειδιού μπορεί να αλλάζουν απρόβλεπτα στον χρόνο, είναι προφανές ότι η διαδικασία *ad hoc* αποθήκευσης δεν μπορεί από μόνη της να εγγυηθεί ότι η ενημέρωση θα παραμείνει στους πρωτεύοντες κόμβους. Όμως, σε επόμενη ανάκτηση ερωτηθούν όλοι οι σχετικοί κόμβοι, υπάρχει πολύ μεγάλη πιθανότητα ότι ένας τουλάχιστον θα προλάβει να επικοινωνήσει την αλλαγή σε κάποιον από τους υπόλοιπους συμμετέχοντες του δικτύου. Με αυτό το σκεπτικό, οι αλλαγές μπορούν να διαδοθούν μέσω της διαδικασίας εντοπισμού του κλειδιού. Αυτό προϋποθέτει ότι ο αντίστοιχος αλγόριθμος δεν θα τερματίζει με το που θα βρεθεί ένα οποιοδήποτε αντίγραφο του αντικειμένου, αλλά θα συνεχίζει μέχρι να βρεθούν όλες οι διαθέσιμες εκδόσεις ενός ζεύγους κλειδιού-τιμής. Τότε, γνωρίζοντας πλέον όλες τις σχετικές εκδόσεις, ο κόμβος που έκανε το αρχικό ερώτημα θα μπορεί να αποφασίσει ποια έκδοση πρέπει να διατηρηθεί και να στείλει τα απαραίτητα μηνύματα αποθήκευσης στους ομότιμούς του που έχουν μείνει με παλιές ή άκυρες τιμές.

3.3.1 Υλοποίηση

Οι αλλαγές αυτές υλοποιήθηκαν σε ένα δίκτυο Kademlia. Ο νέος αλγόριθμος ανάκτησης δεδομένων, λειτουργεί όπως ένας βρόχος από εντολές `FIND_NODE`, κάτι που αρχικά χρησίμευε μόνο για την διαδικασία αποθήκευσης. Έτσι, πρώτα βρίσκονται οι πλησιέστεροι κόμβοι ενός κλειδιού και ύστερα, μέσω `FIND_VALUE` ΑΚΣ ζητείται να επιστραφούν τα δεδομένα που αποθηκεύουν. Ο κόμβος που κάνει την αναζήτηση, θα ελέγξει όλες τις σχετικές τιμές, θα βρει την πιο πρόσφατη (αν χρησιμοποιείται η ημερομηνία αλλαγής ως δείκτης έκδοσης μιας τιμής) και θα ενημερώσει τους συμμετέχοντες που ενδέχεται να έχουν λάθος αντίγραφα. Προφανώς, αν ένας από τους πλησιέστερους κόμβους δεν απαντήσει καθόλου με δεδομένα, θεωρείται ότι πρέπει να του σταλεί μια νέα έκδοση. Όταν και οι κ πλησιέστεροι θα έχουν επιστρέψει ένα μήνυμα (είτε δεδομένα,

είτε λίστα από κόμβους), στέλνονται οι κατάλληλες εντολές STORE. Οι κόμβοι που λαμβάνουν STORE αντικαθιστούν τα δεδομένα που αποθηκεύουν τοπικά με την ενημερωμένη τους έκδοση. Η αποθήκευση ενός νέου κλειδιού στο σύστημα γίνεται με τον ίδιο ακριβώς τρόπο, μόνο που η τελευταία έκδοση της τιμής δίνεται από τον χρήστη, οπότε δεν υπάρχει λόγος να σταλούν FIND_VALUE ΑΚΣ στους πλησιέστερους κόμβους του αντικειμένου. Επιπρόσθετα, αν μία τιμή έχει μηδενικό μήκος, θεωρείται ότι έχει διαγραφεί. Οι τιμές αυτές εξέρχονται τελικά από το δίκτυο όταν λήξει η ισχύς τους.

3.3.2 Παρατηρήσεις

Στο αυθεντικό πρωτόκολλο του Kademlia, η λειτουργία ανάκτησης απαιτεί υπό φυσιολογικές συνθήκες το πολύ $log(N)$ βήματα σε ένα δίκτυο N κόμβων. Η διαδικασία εντοπισμού των πλησιέστερων κόμβων είναι συμπληρωματική της αναζήτησης της ζητούμενης τιμής. Αν κάποια από τις "πρώτες" FIND_VALUE ΑΚΣ επιστρέψει δεδομένα, δεν υπάρχει λόγος να συνεχιστεί η έμμεση εκτέλεση εντολών FIND_NODE. Από την άλλη μεριά, οι αλλαγές που προτείνουμε ενοποιούν τους αλγορίθμους αποθήκευσης και ανάκτησης σε μια κοινή διαδικασία δύο βημάτων: Πρώτα πρέπει να βρεθούν οι κοντινότεροι κόμβοι ενός κλειδιού και μετά να αποθηκευτεί η ενημερωμένη τιμή του. Δεδομένα που μπορεί να βρίσκονται σε ενδιάμεσες μνήμες αγνοούνται και η λειτουργία ανάκτησης συνεχίζει μέχρι να βρεθούν όλοι οι υπεύθυνοι φύλαξης του αντικειμένου. Το μειονέκτημα είναι ότι έτσι πρέπει κάθε φορά να ολοκληρωθούν τουλάχιστον $log(N)$ βήματα μέσα στο πλέγμα των κόμβων, προκειμένου να βρεθούν οι πλησιέστεροι.

Το αποτέλεσμα είναι ότι η ανάκτηση δεδομένων δεν μπορεί να εκτελεστεί ταχύτερα — στοιχίζει ακριβώς όσο και η αποθήκευση. Από την άλλη μεριά, κάθε αλλαγή σε ένα ΔΟΚ έχει κάποιο κόστος, είτε υπό τη μορφή bytes που ανταλλάσσονται, είτε υπό τη μορφή μεγαλύτερης καθυστέρησης στον χρόνο που απαιτείται για ένα αποτέλεσμα (δυο μεγέθη που έχουν προταθεί ως ένας κοινός παρονομαστής αξιολόγησης τέτοιων συστημάτων). Παρόλα αυτά, η "ποινή" που επιβάλλεται σε κάθε ανάκτηση δεδομένων μπορεί να δράσει και εις όφελος του δικτύου. Με το να χρησιμοποιούνται περισσότερα μηνύματα κατά τον εντοπισμό, οι πίνακες δρομολόγησης ενημερώνονται πιο συχνά και το συνολικό δίκτυο γίνεται ακόμη πιο ανεκτικό σε σφάλματα.

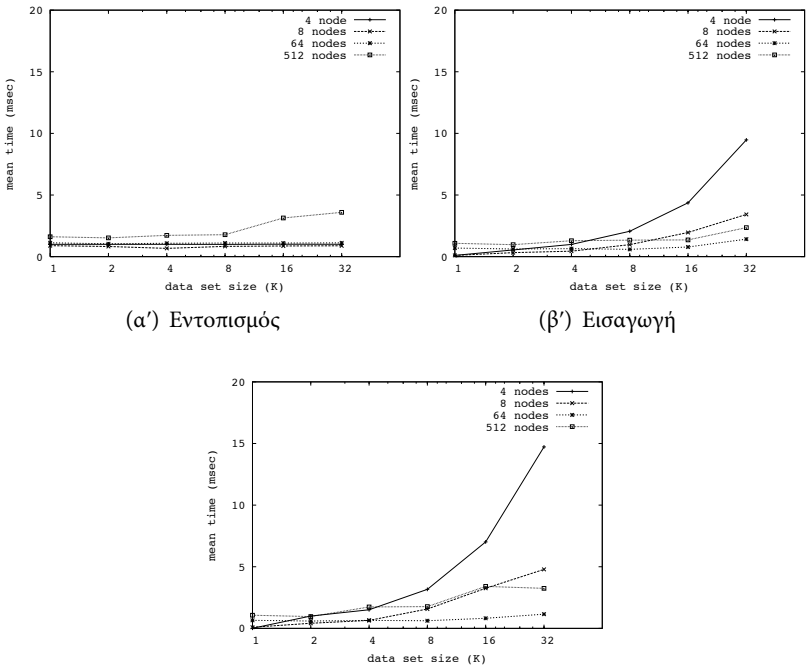
Επιπλέον, η αλλαγή του αλγορίθμου ανάκτησης είναι κάτι που μπορεί να υιοθετηθεί εύκολα και σε άλλα πρωτόκολλα ΚΠΚ. Απαιτείται ένας μικρός αριθμός μετατροπών, οι οποίες επηρεάζουν μόνο τους μηχανισμούς αποθήκευσης και ανάκτησης του πρωτοκόλλου. Στην συγκεκριμένη υλοποίηση, δεν υπήρξε η ανάγκη να μεταβληθεί ο τρόπος με τον οποίον το Kademlia διαχειρίζεται την εισαγωγή νέων κόμβων στο δίκτυο, ούτε η διαδικασία ενημέρωσης των πινάκων δρομολόγησης. Διατηρείται δε ακόμη η απαίτηση ότι όλα τα αντικείμενα λήγουν 24 ώρες μετά την τελευταία τους ενημέρωση. Αυτό βοηθά κυρίως στο να "καθαρίζεται" ο χώρος από δεδομένα που έχουν διαγραφεί.

3.3.3 Αξιολόγηση

Προκειμένου να μελετήσουμε τη συμπεριφορά του συστήματος υλοποιήσαμε το πλήρες πρωτόκολλο του Kademlia, σε συνδυασμό με τις προτεινόμενες αλλαγές, χρησιμοποιώντας τη γλώσσα προγραμματισμού C[1]. Ο κορμός του προγράμματος αποτελείται από έναν ασύγχρονο διαχειριστή μηνυμάτων που προωθεί όλα τα εισερχόμενα πακέτα UDP σε μια μηχανή καταστάσεων. Τα εξερχόμενα πακέτα αποστέλλονται απευθείας στο δίκτυο, καθώς στο πρωτόκολλο UDP ο αποστολέας δεν αναμένει για επιβεβαίωση της λήψης. Το εκτελέσιμο, εκτός από την δίοδο πακέτων του πρωτοκόλλου, διαχειρίζεται και συνδέσεις τύπου TCP για την επικοινωνία με προγράμματα πελάτη, που μπορούν να ζητούν από τη μηχανή καταστάσεων τη διεκπεραίωση εντολών εισαγωγής, εντοπισμού ή ενημέρωσης δεδομένων (put, get και update αντίστοιχα). Μάλιστα, το πρόγραμμα μπορεί να εκτελείται πολλές φορές παράλληλα σε έναν υπολογιστή, καθώς οι μεμονωμένες απαιτήσεις του σε μνήμη είναι ελάχιστες και ο υπολογιστικός φόρτος που δημιουργεί είναι αμελητέος. Έτσι, μας δίνεται η δυνατότητα να προσομοιώσουμε ένα σύνολο κόμβων του ΚΠΚ και να παρακολουθήσουμε σε πραγματικό χρόνο τις ενημερώσεις και το περιεχόμενο των αντίστοιχων μηνυμάτων. Η πραγματική αποθήκευση των δεδομένων σε κάθε ομότιμο γίνεται σε κάποιο τοπικό μαγνητικό μέσο (σκληρός δίσκος). Για να εκτελέσουμε τα πειράματα που παρουσιάζονται στη συνέχεια, χρησιμοποιήθηκε μια συστοιχία οκτώ υπολογιστών (ο καθένας με ενσωματωμένες πολυεπεξεργαστικές μονάδες), όπου κάθε μηχάνημα φιλοξενούσε μερικές δεκάδες κόμβους. Μια άλλη, εξωτερική εφαρμογή παρήγαγε και μεταβίβαζε στους ομότιμους εντολές αποθήκευσης, εντοπισμού και ενημέρωσης, ανάλογα με τα χαρακτηριστικά του σεναρίου προς αξιολόγηση.

Πρώτα μετρήθηκε ο μέσος χρόνος που απαιτείται από το σύστημα για την ολοκλήρωση των βασικών πράξεων σε ένα στατικό δίκτυο, αλλάζοντας κάθε φορά τον αριθμό των κόμβων (μέγεθος δικτύου) και το πλήθος των αντικειμένων που αυτοί διαχειρίζονται. Το ζητούμενο ήταν η καταγραφή των χαρακτηριστικών κλιμάκωσης του ΚΠΚ και ο εντοπισμός ενδεχόμενων προβλημάτων της πρότυπης υλοποίησης. Στα Σχήματα 3.3(α′), 3.3(β′) και 3.3(γ′), όπου απεικονίζονται τα αποτελέσματα, φαίνεται ότι απαιτούνται κατά μέσο όρο μόλις 2.5 msec για την ολοκλήρωση μιας πράξης εισαγωγής σε ένα δίκτυο που αποτελείται από 512 κόμβους και αποθηκεύει 8 χιλιάδες τιμές. Το συνολικό σύστημα παρουσιάζει καλά χαρακτηριστικά κλιμάκωσης μέχρι αυτό το σημείο. Για περισσότερες από 8 χιλιάδες τιμές, αρχίζουν να εμφανίζονται προβλήματα απόδοσης, που οφείλονται κυρίως στο υποσύστημα τοπικής αποθήκευσης των δεδομένων. Η επιπλέον καθυστέρηση εισάγεται αφενός από το σύστημα αρχείων που καλείται να διαχειριστεί δεκάδες χιλιάδες μικρά αρχεία σε έναν κατάλογο, αφετέρου δε από το τελικό μέσο αποθήκευσης, καθώς οι σχετικές πράξεις ανάγνωσης και εγγραφής έχουν τυχαία σειρά και πολύ μικρό μέγεθος. Τα προβλήματα κλιμάκωσης είναι πολύ πιο έντονα όταν ο ΚΠΚ αποτελείται από 4 κόμβους. Σε αυτές τις μετρήσεις, λόγω του μικρού μεγέθους του δικτύου χρησιμοποιήθηκε και μικρή σχετικά παράμετρος κ στο επίπεδο του πρωτοκόλλου ($\kappa = 4$ και $\alpha = 3$). Στη περίπτωση των 4 κόμβων, όλοι οι ομότιμοι αποθηκεύουν όλα τα δεδομένα. Έτσι, όσο αυξάνεται ο αριθμός των αντικειμένων στο δίκτυο, τόσο αυξάνονται και τα αρχεία που αποθηκεύονται στον τοπικό κατάλογο του

[1]Η υλοποίηση αυτή είναι προπομπός της πλατφόρμας που παρουσιάζεται στο Παράρτημα.



(α′) Εντοπισμός (β′) Εισαγωγή

(γ′) Ενημέρωση

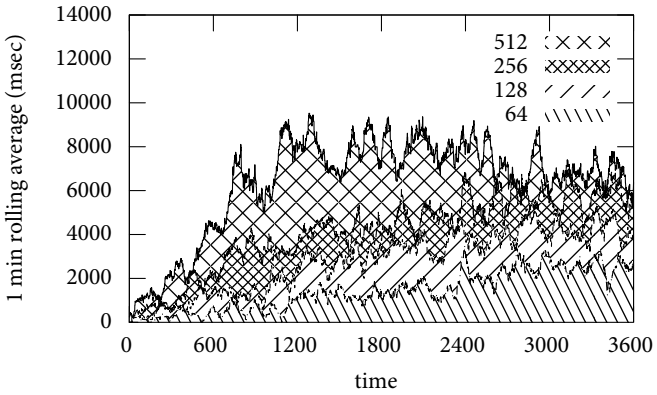
Σχήμα 3.3: *Μέσος χρόνος για την ολοκλήρωση των πράξεων σε ένα στατικό δίκτυο*

κάθε κόμβου. Σε επόμενες υλοποιήσεις, εγκαταλείφθηκε η χρήση του συστήματος αρχείων ως μέσο αποθήκευσης.

Οι ΚΠΚ με περισσότερους από 4 κόμβους παρουσιάζουν πολύ καλύτερη συμπεριφορά, καθώς ο μέσος χρόνος ολοκλήρωσης των εντολών δεν αποκλίνει σημαντικά όσο διπλασιάζεται το πλήθος των τιμών. Επίσης, οι εισαγωγές και οι ενημερώσεις έχουν παρόμοια χαρακτηριστικά, αφού το πρωτόκολλο χειρίζεται και τις δύο πράξεις με όμοιο τρόπο. Οι όποιες διαφορές οφείλονται στο γεγονός ότι στα πειράματα οι εισαγωγές έγιναν με "άδειο" το δίκτυο, ενώ οι ενημερώσεις εκ των υστέρων, ώστε να υπάρχουν ήδη εκδόσεις των τιμών για να επεξεργαστεί ο σχετικός αλγόριθμος.

Το δεύτερο σενάριο που αξιολογήθηκε ήταν ένα δυναμικό δίκτυο, όπου οι συμμετέχοντες εισέρχονται και εξέρχονται του ΚΠΚ με τυχαίο τρόπο. Χρησιμοποιώντας την πρότυπη υλοποίηση, κατασκευάσαμε ένα σύνολο 256 κόμβων οι οποίοι αποθήκευαν 2048 τιμές. Οι διευθύνσεις αμφότερων των κόμβων και των αντικειμένων επιλέχτηκαν τυχαία από έναν χώρο 32 δυαδικών ψηφίων, ενώ οι παράμετροι παραλληλίας της επικοινωνίας και της τήρησης αντιγράφων του Kademlia αρχικοποιήθηκαν στις τιμές $\alpha = 3$ και $\kappa = 4$ αντίστοιχα. Η μικρή τιμή του κ εξασφαλίζει και ότι οι πίνακες δρομολόγησης των ομότιμων θα έχουν εγγραφές, ακόμη και για τις ομάδες καταχωρήσεων που αντιστοιχούν σε μεγάλα διαστήματα διευθύνσεων. Επίσης, οριοθετεί τον αριθμό των αντιγράφων που θα δημιουργούνται αυτόματα για κάθε αντικείμενο, καθώς το δίκτυο είναι σχετικά μικρό. Κάθε πείραμα είχε διάρκεια μία ώρα, κατά την οποία το σύνολο των κόμβων άλλαζε δυναμικά, ενώ παράλληλα εκτελούνταν πράξεις εντοπισμού και ενημέρωσης δεδομένων. Η παραγωγή των αντίστοιχων εντολών *startup*, *shutdown*, *get* και *update* ακολουθούσε την κατανομή Poisson. Ο ρυθμός εκτέλεσης εντολών ορίστηκε αρχικά στις 1024 $\frac{\text{πράξεις}}{\text{ώρα}}$, ενώ ο

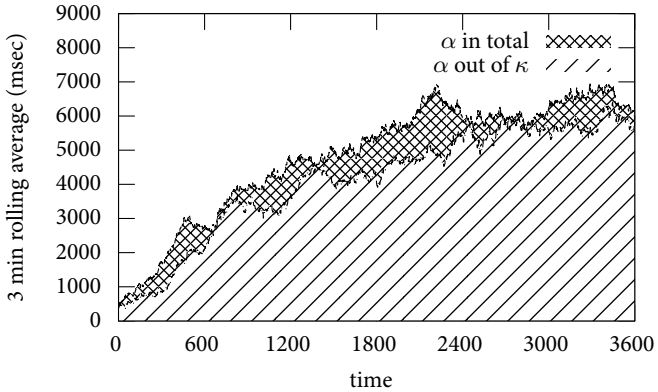
ρυθμός ανανέωσης του δικτύου σε 64 $\frac{\text{κόμβους}}{\text{ώρα}}$. Σε κάθε πείραμα ο ρυθμός ανανέωσης του δικτύου διπλασιαζόταν, φτάνοντας τελικά στους 512 $\frac{\text{κόμβους}}{\text{ώρα}}$. Διατηρώντας τον αριθμό νέων κόμβων ίδιο με τον αριθμό των κόμβων που αποχωρούν στο ίδιο διάστημα, το συνολικό πλήθος του ΔΟΚ παρέμενε κατά μέσο όρο σταθερό σε όλη τη διάρκεια του κάθε πειράματος. Στο Σχήμα 3.4, φαίνεται ο μέσος χρόνος που απαιτείται για την ολοκλήρωση των πράξεων εντοπισμού και ενημέρωσης για κυλιόμενο διάστημα διάρκειας ενός λεπτού, καθώς αλλάζει κάθε φορά ο ρυθμός ανανέωσης των κόμβων του δικτύου. Στο περιβάλλον εκτέλεσης της προσομοίωσης δεν υπάρχει πρακτικά χρόνος καθυστέρησης στην επικοινωνία μεταξύ δύο κόμβων (communication latency), ενώ το διάστημα αναμονής για απάντηση (communication timeout) ορίστηκε στα 4 δευτερόλεπτα.



Σχήμα 3.4: *Χρόνος ολοκλήρωσης των πράξεων με αυξανόμενο ρυθμό ανανέωσης του δικτύου*

Όπως αναμενόταν, η αύξηση του αριθμού των εξερχόμενων κόμβων (αστοχίες κόμβων), συντελεί και στην ανάλογη αύξηση του χρόνου που απαιτείται για την ολοκλήρωση πράξεων επί των δεδομένων που αποθηκεύονται στον ΚΠΚ. Μεγάλοι ρυθμοί ανανέωσης, οδηγούν σε λανθασμένες καταχωρήσεις στους πίνακες δρομολόγησης των ομότιμων, οπότε πολλοί κόμβοι στέλνουν μηνύματα σε ανύπαρκτες διευθύνσεις και αναγκάζονται να περιμένουν μέχρι να κριθεί άκυρη η επικοινωνία για να συνεχίσουν. Σύμφωνα με το σχήμα δρομολόγησης του Kademlia, οι αιτήσεις εντοπισμού ακολουθούν α παράλληλα μονοπάτια για να παρακάμπτονται οι προβληματικοί κόμβοι, ενώ είναι πολύ πιθανό ένα από αυτά τα μονοπάτια να οδηγήσει ταχύτερα σε κάποια τιμή που αποθηκεύεται στη λανθάνουσα μνήμη ενός κοντινού ομότιμου. Αντίθετα, στη περίπτωση του πρωτοκόλλου που παρουσιάζεται, όπου δεν χρησιμοποιούνται λανθάνουσες μνήμες, προσπαθούμε να μειώσουμε τον συνολικό χρόνο ολοκλήρωσης μιας αίτησης με τη δυναμική προσαρμογή των μονοπατιών στις απαντήσεις των κόμβων. Στη πρώτη φάση του αλγορίθμου εντοπισμού, όπου αποστέλλονται ΑΚΣ τύπου `FIND_NODE`, ο αντίστοιχος κόμβος φροντίζει να συνεχίσει την επικοινωνία με τους α πρώτους ομότιμους, από αυτούς που εμφανίζονται δυναμικά στον πίνακα των υποψήφιων πλησιέστερων κόμβων στην ζητούμενη τιμή. Έτσι, κάθε φορά που επιστρέφεται μια απάντηση και αλλάζει η κατάταξη των υποψηφίων, μπορεί να παραχθούν περισσότερα του ενός μηνύματα `FIND_NODE` και τελικώς να έχουν δημιουργηθεί περισσότερα από α παράλληλα μονοπάτια αναζήτησης. Στο Kademlia, δημιουργούνται αρχικά α μηνύματα και ένα για κάθε απάντηση, οπότε και ο αριθμός των μονοπατιών παραμένει σταθερός. Η αλλαγή αυτή, βελτιστοποιεί αισθητά τους χρόνους ολοκλήρωσης των εντολών, σε αντάλλαγμα με μια

μικρή αύξηση στον συνολικό αριθμό των μηνυμάτων που παράγονται για κάθε αίτηση. Η διαφορά μεταξύ των δύο αλγορίθμων για μια εκτέλεση του πειράματος με ρυθμό ανανέωσης 256 $\frac{\text{κόμβων}}{\text{ώρα}}$ παρουσιάζεται στο Σχήμα 3.5.



Σχήμα 3.5: *Δυναμική προσαρμογή των παράλληλων μονοπατιών επικοινωνίας σε αλλαγές της λίστας των πλησιέστερων κόμβων*

Οι υψηλοί ρυθμοί αποχώρησης έχουν επίσης σημαντικό αντίκτυπο και στη πράξη εντοπισμού. Όπως φαίνεται στον Πίνακα 3.1, ο διπλασιασμός του αριθμού των κόμβων που αλλάζουν ανά ώρα, προκαλεί σχεδόν εκθετική αύξηση του αριθμού των σφαλμάτων που παρουσιάζονται στις αντίστοιχες πράξεις εντοπισμού. Τα σφάλματα αυτά αναφέρονται στις περιπτώσεις όπου η αναζήτηση για ένα κλειδί αποτυγχάνει, ενώ αυτό αποθηκεύεται όντως εντός του δικτύου. Καθώς η αποτυχία εντοπισμού συνεπάγεται αδυναμία δρομολόγησης στην αντίστοιχη διεύθυνση, επαναλάβαμε το πείραμα με τα χειρότερα αποτελέσματα (ανανέωση 512 $\frac{\text{κόμβων}}{\text{ώρα}}$), διπλασιάζοντας κάθε φορά τον αριθμό των αναζητήσεων από 1024 σε 16384 $\frac{\text{πράξεις}}{\text{ώρα}}$. Θεωρητικά — αλλά και στη πράξη, όπως δείχνει ο Πίνακας 3.2 — τα επιπλέον μηνύματα που ανταλλάσσουν οι κόμβοι συντελούν και στην συχνότερη ανανέωση των πινάκων δρομολόγησής τους, οπότε και το συνολικό δίκτυο αποκτά μεγαλύτερη συνοχή. Έτσι, ακόμα και στο σενάριο όπου οι κόμβοι είναι εντελώς αναξιόπιστοι, ο υψηλός ρυθμός αιτήσεων μπορεί να οδηγήσει τον αριθμό σφαλμάτων σε ποσοστά κάτω του 1%. Εξάλλου, η πράξη εντοπισμού στον συγκεκριμένο ΚΠΚ, αναλαμβάνει και την αποθήκευση των αντικειμένων στους πλησιέστερους κόμβους τους, οπότε βοηθά και στη τήρηση των αντιγράφων ακόμα και σε δίκτυα με πολύ δυναμικά χαρακτηριστικά.

Πίνακας 3.1: *Αυξάνοντας τον ρυθμό ανανέωσης του δικτύου*

$\frac{\text{κόμβοι}}{\text{ώρα}}$	64	128	256	512
Σφάλματα	0	2	32	154
Ρυθμός	0.00%	0.19%	3.12%	15.03%

Ο αρχικός υψηλός αριθμός σφαλμάτων, οφείλεται και στον τρόπο με τον οποίο το Kademlia διαχειρίζεται την εισαγωγή νέων κόμβων. Όταν ένας κόμβος μάθει για την ύπαρξη ενός νέου ομότιμου, δεν είναι απαραίτητο ότι θα τον προσθέσει στον πίνακα δρομολόγησής του. Μπορεί οι σχετικές θέσεις να είναι ήδη κατειλημμένες και πρέπει να αποτύχουν αρκετές προσπάθειες επι-

Πίνακας 3.2: *Αυξάνοντας τον ρυθμό εκτέλεσης πράξεων*

$\frac{\text{πράξεις}}{\text{ώρα}}$	1024	2048	4096	8192	16384
Σφάλματα	162	106	172	126	84
	131	91	137	80	58
	163	63	145	116	106
	143	61	130	120	87
Ρυθμός	15.82%	5.17%	4.19%	1.53%	0.51%
	12.79%	4.44%	3.34%	0.97%	0.35%
	15.91%	3.07%	3.54%	1.41%	0.64%
	13.96%	2.97%	3.17%	1.46%	0.53%

κοινωνίας προκειμένου να αντικατασταθεί μια καταχώρηση. Ο αλγόριθμος, λοιπόν, είναι σχεδιασμένος για να ευνοεί τους κόμβους που παραμένουν στο δίκτυο. Η τυχαία κατανομή που ακολουθήθηκε στα πειράματα δεν εκμεταλλεύεται αυτό το γεγονός.

3.3.4 Συμπεράσματα

Οι αλλαγές που σχεδιάστηκαν και υλοποιήθηκαν στη πράξη ανάκτησης δεδομένων ενός ΚΠΚ, επιτρέπουν ως ένα σημείο την ενημέρωση των τιμών που αποθηκεύονται στο δίκτυο. Ο νέος αλγόριθμος φροντίζει παράλληλα με την διεκπεραίωση μιας σχετικής αίτησης εντοπισμού να ανακαλύψει ασυνέπειες που μπορεί να δημιουργηθούν μεταξύ των αντιγράφων που τηρούν οι πλησιέστεροι κόμβοι ενός αντικειμένου, λόγω της δυναμικής φύσης του δικτύου. Μάλιστα, όπως φαίνεται από τα αντίστοιχα πειραματικά αποτελέσματα, οι δυνατότητες κλιμάκωσης του συστήματος δεν μεταβάλλονται, καθώς και ο επιπλέον φόρτος που δημιουργείται από το νέο πρωτόκολλο μοιράζεται ομαλά μεταξύ των κόμβων, ενώ τα επιπλέον μηνύματα συντελούν στην ακόμη καλύτερη συμπεριφορά του δικτύου κάτω από υψηλούς ρυθμούς ανανέωσης. Παρόλα αυτά, η αποτελεσματικότητα των ενημερώσεων σε ένα εξαιρετικά δυναμικό περιβάλλον δεν διασφαλίζεται. Εξαρτάται κυρίως από το πόσες αιτήσεις αναζήτησης θα γίνουν κατά τη διάρκεια της συνεχούς ανανέωσης των κόμβων. Δεν διασφαλίζεται ούτε η πράξη της ενημέρωσης, ούτε είναι δυνατόν να προκύψει κατά την αποθήκευση μιας νέας έκδοσης αν η πράξη ήταν τελικά επιτυχής ή όχι. Ακόμα και αν αλλάξει μια τιμή, μπορεί σε επόμενη αναζήτηση να επιστραφεί προηγούμενη έκδοση. Για αυτό το λόγο, δεν προτείνεται η χρήση του συγκεκριμένου πρωτοκόλλου σε κατανεμημένα συστήματα αρχείων και γενικά εφαρμογές όπου απαιτείται αυστηρή σειριοποίηση των εγγραφών και παροχή αντίστοιχων εγγυήσεων συνέπειας των δεδομένων. Όμως, άλλου τύπου συστήματα, μπορεί να χρησιμοποιούν τον ΚΠΚ ως μια υποδομή αποθήκευσης, όπου η έκδοση της επιστρεφόμενης τιμής δεν έχει ιδιαίτερη σημασία, αλλά απαιτείται ένας χαλαρός συντονισμός των σχετικών εγγραφών. Παραδείγματα τέτοιων συστημάτων αποτελούν οι κατανεμημένες βάσεις εντοπισμού αντιγράφων αρχείων προς μεταφορά ή ενημέρωση, όπου οι σχετικές τιμές που επιστρέφονται είναι ενδεικτικές για την αρχικοποίηση του συστήματος. Έτσι, ένα σύστημα μεταφοράς αρχείων μπορεί να αποθηκεύει σε έναν ΚΠΚ τις τοποθεσίες αρχείων για έναν αλ-

γόριθμο κατανεμημένης μεταφόρτωσης. Ακόμα και αν κάποιες καταχωρήσεις είναι άκυρες, το σύστημα θα μπορεί να λειτουργήσει με τις υπόλοιπες. Η χρήση του αλγορίθμου εντοπισμού αλλαγών σε ένα τέτοιο σενάριο, παρουσιάζεται στις εργασίες [CZK05] και [CZKT06].

3.4 XOROS

Ως συνέχεια της προηγούμενης προσέγγισης στο πρόβλημα παρουσιάσαμε ένα πλήρες πρωτόκολλο ενημέρωσης δεδομένων στα πλαίσια ενός νέου ΚΠΚ, που ονομάσαμε XOROS (XOR Object Store). Το XOROS βασίζεται και πάλι στο Kademlia. Διατηρείται η έννοια του χώρου διευθύνσεων με τη μορφή ενός δυαδικού δέντρου, στο οποίο οι αποστάσεις μεταξύ των φύλλων υπολογίζονται με τη πράξη του "αποκλειστικού ή" (XOR), ορίζονται, όμως, εκ νέου οι πράξεις εντοπισμού και αποθήκευσης δεδομένων. Επιπρόσθετα, λαμβάνεται υπόψη σε όλες τις διαδικασίες η δυνατότητα αλλαγής των τιμών και η ενδεχόμενη απροθυμία κάποιων κόμβων να εκτελούν το προσυμφωνημένο πρωτόκολλο. Έτσι, απαιτείται ο ορισμός επιπλέον τύπων ΑΚΣ και μεταβλητών λειτουργίας — όπως η μεταβλητή λ, που καθορίζει το ποσοστό των λαθών που μπορεί να διαχειριστεί το σύστημα. Η πράξη εισαγωγής δεδομένων (put) πρακτικά καταργείται και τη θέση της παίρνει μια ολοκληρωμένη πράξη ενημέρωσης (update), που μπορεί να χειριστεί νέες τιμές χωρίς την ανάγκη εκχώρησης αναγνωριστικών εκδόσεων ή τη χρήση σχετικών διανυσμάτων. Μάλιστα, υποστηρίζεται και η πράξη αντικατάστασης μιας τιμής με βάση τα περιεχόμενά της. Στις επόμενες παραγράφους παρουσιάζεται εις βάθος το πρωτόκολλο και τα σχετικά αποτελέσματα αξιολόγησης.

3.4.1 Η συναλλαγή ενημέρωσης

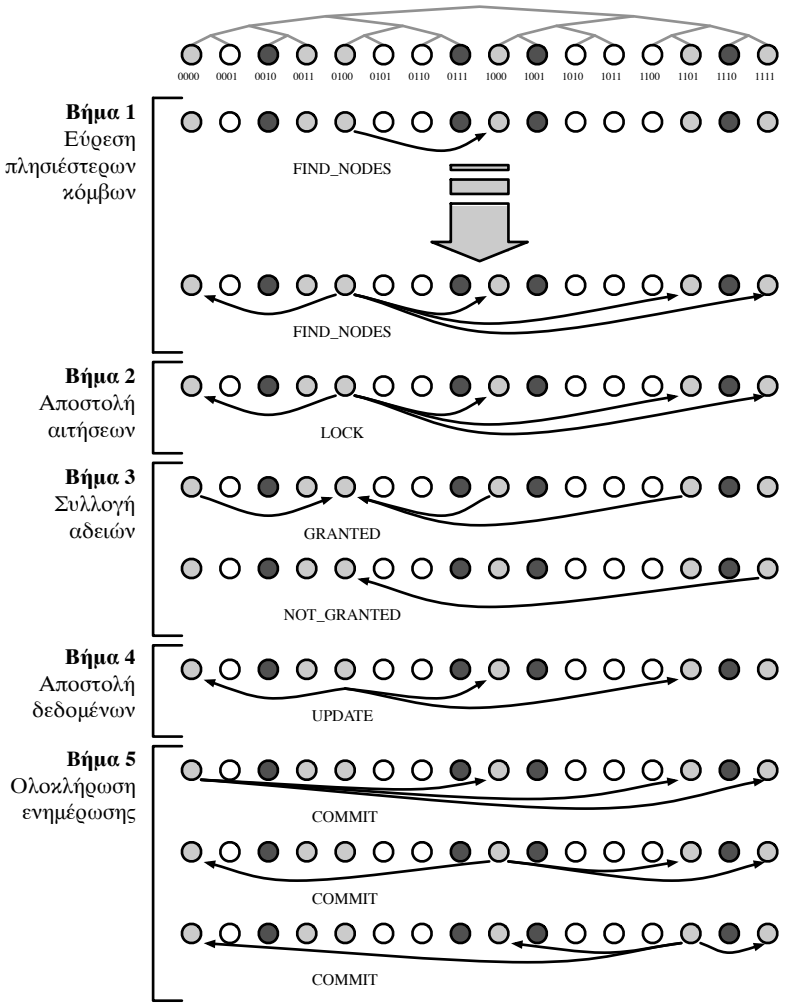
Προκειμένου να υποστηρίζονται σειριοποιήσιμες ενημερώσεις στα δεδομένα, το XOROS απαιτεί πολλαπλές, στενά συνδεδεμένες φάσεις ανταλλαγής μηνυμάτων, που συνολικά αποτελούν μια ενιαία συναλλαγή. Για να ανανεώσει δεδομένα, ένας κόμβος πρώτα εκτελεί έναν βρόχο `FIND_NODE` για να βρει τους κ κόμβους που αποθηκεύουν αντίγραφα του συγκεκριμένου αντικειμένου (τους πλησιέστερους). Θεωρούμε αυτούς τους κόμβους ως τους υπεύθυνους για να δώσουν την άδεια ανανέωσης του αντίστοιχου κλειδιού. Το επόμενο βήμα περιλαμβάνει την αποστολή ΑΚΣ τύπου `LOCK` σε όλα τα μέλη της ομάδας ομότιμων για να ζητηθεί η αποκλειστική πρόσβαση στα δεδομένα. Οι κόμβοι μπορεί να απαντήσουν σε εντολή κλειδώματος είτε θετικά (`GRANTED`), είτε αρνητικά (`NOT_GRANTED`), αναλόγως με το αν έχουν ήδη δώσει την άδεια τους σε κάποιον άλλο κόμβο. Η "άδεια" ή "ψήφος" έχει διπλή σημασία, καθώς αποτελεί στοιχείο του πρωτοκόλλου αμοιβαίου αποκλεισμού που προστατεύει την πράξη ενημέρωσης (αμοιβαίος αποκλεισμός με "αδειοδότηση"), αλλά και υποδηλώνει την ουσιαστική προθυμία του ομότιμου να δεχτεί επόμενες εντολές αποθήκευσης για το αντικείμενο. Οπότε και ένας κόμβος μπορεί να στείλει εντολές τύπου `UPDATE` μόνο στους κόμβους που του το έχουν επιτρέψει. Βέβαια, ΑΚΣ τύπου `UPDATE` θα πρέπει να στέλνονται μόνο αφού ο ομότιμος που κάνει την αλλαγή έχει καταφέρει να συλλέξει μ_{lock} άδειες — ένας αριθμός που συμβολίζει τη πλειοψηφία των πλησιέστερων (της "απαρτίας") και εγγυάται

ότι κανένας άλλος συναγωνιστής δεν θα έχει καταφέρει τον αποκλεισμό του κοινού πόρου. Τα μηνύματα τύπου GRANTED περιέχουν επίσης και την τρέχουσα τιμή ενός κλειδιού, προκειμένου να είναι δυνατή η υλοποίηση ατομικών συναλλαγών ανάγνωσης-ενημέρωσης-εγγραφής.

Από την άλλη μεριά, όταν ένα κόμβος δέχεται UPDATE για κάποια τιμή που αποθηκεύει, δεν μπορεί να θεωρήσει με ασφάλεια ότι όντως η απαρτία των πλησιέστερων κόμβων του αντικειμένου έχει καταλήξει ότι ο συγκεκριμένος αποστολέας επιτρέπεται να κάνει την αλλαγή. Το τελευταίο βήμα του αλγορίθμου, περιλαμβάνει την επικοινωνία των πλησιέστερων κόμβων μεταξύ τους, μέσω ΑΚΣ τύπου COMMIT, προκειμένου να ολοκληρωθεί η συναλλαγή και να διαδοθεί η σωστή τιμή ακόμα και στα μέλη που μπορεί να μην έχουν δώσει τη σχετική άδεια ενημέρωσης. Κάθε κοντινός κόμβος μπορεί να θεωρήσει ότι ολοκληρώθηκε μια ενημέρωση και να αποθηκεύσει τη νέα τιμή, μόνο αν δεχτεί μ_{store} μηνύματα COMMIT (συμπεριλαμβανομένου και του αρχικού UPDATE αν υπάρχει αντίστοιχη ψήφος), τα οποία να αναφέρονται όλα στα ίδια δεδομένα από τον ίδιο αποστολέα. Αντίστοιχα με το μ_{lock}, η παράμετρος μ_{store} προστατεύει τη συνέπεια της συναλλαγής σε περίπτωση συναγωνισμού για την ίδια ενημέρωση. Προφανώς, όταν όλοι οι κόμβοι θεωρούνται έμπιστοι και δεν υπάρχουν σφάλματα $\mu_{lock} = \mu_{store} = \kappa/2 + 1$. Σε κάθε άλλη περίπτωση, όμως, το μ_{store} θα έχει διαφορετική τιμή και το μ_{lock} θα είναι μεγαλύτερο από το μ_{store} κατά τον μέγιστο αριθμό κόμβων που μπορεί να μην συμπεριφέρονται ή να μη λειτουργούν σωστά.

Ένα απλοποιημένο σχέδιο του αλγορίθμου φαίνεται στο Σχήμα 3.6, όπου οι κόμβοι και τα δεδομένα παρουσιάζονται με ανοιχτό και σκούρο γκρι χρώμα αντίστοιχα. Με τη ποσότητα των αντιγράφων να ορίζεται σε 4 ($\kappa = 4$) και μια μινιμαλιστική έκδοση της τοπολογίας του Kademlia που αποτελείται από 16 θέσεις (διευθύνσεις των τεσσάρων δυαδικών ψηφίων), οι κοντινότεροι κόμβοι του αντικειμένου με κλειδί 1001 είναι και οι τρεις κόμβοι που ανήκουν στο δεξί υποδέντρο συν ο κόμβος που βρίσκεται αριστερότερα από όλους. Χωρίς υποστήριξη για "Βυζαντινά" σφάλματα, τα μέλη της ομάδας θα αποθηκεύουν τιμές όταν δέχονται τουλάχιστον τρία συνεπή μηνύματα ενημέρωσης, ενώ οι αρχικοί κόμβοι θα πρέπει να εξασφαλίζουν τουλάχιστον τρεις άδειες για να καταφέρουν να ενημερώσουν ένα αντικείμενο. Στο παράδειγμα, ακριβώς τρεις κόμβοι δίνουν τη συγκατάθεσή τους στον κόμβο 0100, ο οποίος τους απαντά με ανανεωμένα δεδομένα. Η ενημέρωση θα ολοκληρωθεί όταν κάθε κόμβος λάβει συνολικά τρεις εντολές UPDATE ή COMMIT, είτε από τον αρχικό κόμβο, είτε από τα υπόλοιπα μέλη της απαρτίας.

Θεωρούμε ότι κάθε κόμβος έχει καλή γνώση των κοντινών του ομότιμων, και ως αποτέλεσμα θα μπορεί να καταλαβαίνει ποια μέλη αντιστοιχούν στους έχοντες αντίγραφα των αντικειμένων που αποθηκεύει, κοιτάζοντας απλά τον τοπικό του πίνακα δρομολόγησης. Παρόλα αυτά, ανάλογα με τον συγκεκριμένο τρόπο που κάθε υλοποίηση ΚΠΚ διαχειρίζεται τους πίνακες δρομολόγησης, η παραπάνω διαδικασία μπορεί να περιλαμβάνει επιπλέον βήματα ανταλλαγής μηνυμάτων. Στο XOROS, μόλις ληφθεί μια ΑΚΣ τύπου COMMIT, ελέγχονται οι πίνακες δρομολόγησης προκειμένου να πιστοποιηθεί αν ένας κόμβος είναι όντως μέσα στους πλησιέστερους του αντίστοιχου κλειδιού. Αν όμως ο κόμβος δεν βρεθεί, αποστέλλονται τα ανάλογα μηνύματα FIND_NODE. Η χρήση διαφορετικών ΑΚΣ για την ενημέρωση ενός αντικειμένου και την ολοκλήρωση της συναλλαγής, γίνεται ώστε να διαχωρίζεται η αντίστοιχη σημασιολογία όταν ο κόμβος που κάνει την ενημέρωση είναι επίσης και μέλος της ομάδας των πλησιέστερων κόμβων ενός κλειδιού.



Σχήμα 3.6: *Βήμα-προς-βήμα αναπαράσταση της συναλλαγής ενημέρωσης*

3.4.2 Ανοχή ασύμβατης συμπεριφοράς

Ας εξετάσουμε τώρα ένα σενάριο όπου λ από τους κ κόμβους της απαρτίας ενός κλειδιού είναι "κακόβουλοι", το οποίο συνεπάγεται ότι μπορεί να εκτελούν το πρωτόκολλο με αυθαίρετο τρόπο. Έστω ότι με σ συμβολίζεται ο αριθμός των κόμβων που είναι έμπιστοι. Αν ζητηθεί από όλους τους κ κόμβους να αναφέρουν την ίδια τιμή ο ένας στον άλλο, μπορούμε να είμαστε σίγουροι ότι οποιοσδήποτε κόμβος θα λάβει τουλάχιστον σ συνεπείς τιμές. Στη συνέχεια της συζήτησης, θεωρείται ότι υπάρχει τρόπος να πιστοποιείται ο αποστολέας κάθε μηνύματος, το οποίο πρακτικά διασφαλίζεται αν λ.χ. το κλειδί κάθε κόμβου παράγεται από τη συνάρτηση κατακερματισμού για την δικτυακή του διεύθυνση.

$$\kappa = \sigma + \lambda \tag{3.1}$$

Έτσι, σε κάθε γύρο διάθεσης αδειών, σ από τις ψήφους μπορεί να κατανεμηθούν σε όσους κόμβους συναγωνίζονται για τον αμοιβαίο αποκλεισμό, ενώ λ επιπλέον μπορεί να απονεμηθούν

ταυτόχρονα σε όλους. Μάλιστα, στο επόμενο βήμα, θεωρώντας ότι όλοι οι συναγωνιστές στείλουν εντολές τύπου UPDATE, σ κόμβοι θα προωθήσουν τη σωστή τιμή, ενώ οι υπόλοιποι λ μπορεί να να μην συνεχίσουν με το πρωτόκολλο ή να στείλουν οποιαδήποτε εσφαλμένη τιμή. Μπορεί επίσης να προωθούν κάθε τιμή που λαμβάνουν, σαν να είχαν δώσει την άδειά τους στον αντίστοιχο κόμβο, δημιουργώντας έτσι λ επιπλέον μηνύματα COMMIT για κάθε ανταγωνιστή.

Τα έμπιστα μέλη της ομάδας, όταν λαμβάνουν μ συνεπή μηνύματα θα πρέπει να μπορούν να αποθηκεύουν τη τιμή με ασφάλεια, αν ο αριθμός αυτός εξασφαλίζει ότι καμιά άλλη ενημέρωση για το ίδιο κλειδί δεν θα επιτύχει κατά τη διάρκεια της ίδιας συναλλαγής. Από τα μ αυτά μηνύματα, το πολύ λ μπορεί να προέρχονται από κακόβουλους κόμβους, οπότε το ελάχιστο $\mu - \lambda$ είναι αληθή/έμπιστα. Αυτό συνεπάγεται ότι οι υπόλοιποι ανταγωνιστές μπορεί να καταφέρουν να αποκομίσουν $\sigma - (\mu - \lambda)$ προωθήσεις από έμπιστους κόμβους, συν επιπλέον λ μηνύματα το οποία μπορεί να προέρχονται από κακόβουλους. Θέτουμε το μ_{store} στην ελάχιστη δυνατή ακέραια τιμή, ώστε ακόμα και αν λ από αυτά τα μηνύματα προέρχονται από κακόβουλους κόμβους, κανένας άλλος ανταγωνιστής δεν θα καταφέρει τη παραγωγή μ_{store} αντίθετων μηνυμάτων, διαταράσσοντας τη συνέπεια του συστήματος. Διαισθητικά, αυτό σημαίνει ότι αν οποιοδήποτε μέλος λάβει μ_{store} συνεπή μηνύματα σχετικά με μια ενημέρωση, ο αποστολέας της τιμής έχει καταφέρει τον αμοιβαίο αποκλεισμό του πόρου.

$$\mu_{store} > \sigma - (\mu - \lambda) + \lambda \overset{(3.1)}{>} \frac{\kappa + \lambda}{2} \Rightarrow \mu_{store} = \lfloor \frac{\kappa + \lambda}{2} \rfloor + 1 \tag{3.2}$$

Από τη πλευρά του αποστολέα, ο αριθμός μ_{store} αντιστοιχεί στον ελάχιστο αριθμό μελών που πρέπει να ψηφίσουν υπέρ του προκειμένου να προχωρήσει με την ενημέρωση ενός αντικειμένου. Παρόλα αυτά, δεν υπάρχει τρόπος να γνωρίζει πόσοι κακόβουλοι κόμβοι μπορεί να περιέχονται μέσα σε αυτούς τους μ_{store} και κατά συνέπεια πόσοι πραγματικά θα προωθήσουν τις αντίστοιχες ΑΚΣ τύπου COMMIT. *Προτείνουμε* κάθε κόμβος να αποκομίζει το ελάχιστο μ_{lock} άδειες (μέσω μηνυμάτων GRANTED) πριν συνεχίσει με την αποστολή της ανανεωμένης τιμής.

$$\mu_{lock} \geq \mu_{store} + \lambda \overset{(3.2)}{>} \frac{\kappa + 3\lambda}{2} \Rightarrow \mu_{lock} = \lfloor \frac{\kappa + 3\lambda}{2} \rfloor + 1 \tag{3.3}$$

Επιπρόσθετα, αν θέλουμε το σύστημα να μπορεί να *εγγυάται* ότι κάθε αποστολέας ο οποίος εκτελεί σωστά το πρωτόκολλο θα καταφέρει να ενημερώσει μια τιμή, πρέπει να θεωρήσουμε ότι το μ_{lock} δεν μπορεί να υπερβαίνει το κ. Αυτό μας οδηγεί στο άνω όριο του λ ως συνάρτηση του κ.

$$\kappa \geq \mu_{lock} \overset{(3.3)}{>} \frac{\kappa + 3\lambda}{2} \Rightarrow \lambda < \frac{\kappa}{3} \tag{3.4}$$

Για κάθε ομάδα $3\lambda + 1$ κόμβων που διατηρούν αντίγραφα μιας τιμής, το XOROS μπορεί να αντέξει μέχρι λ κακόβουλους κόμβους - το οποίο έχει αποδειχθεί και ως βέλτιστο. Αν και το κ μπορεί θεωρητικά να είναι όσο μεγάλο θέλουμε, στη πράξη, μεγάλες τιμές αυξάνουν αντίστοιχα τις απαιτήσεις σε αριθμό μηνυμάτων για κάθε λειτουργία του ΚΠΚ και αποθηκευτικού χώρο σε κάθε ομότιμο.

Ακόμα και αν δεν υπάρχουν κακόβουλοι στο δίκτυο, η "Βυζαντινή" συμπεριφορά που αναμένεται από έως $\frac{\lambda}{\kappa}$ ομότιμους κατά μέσο όρο, μπορεί να αποδοθεί σε ξαφνικά σφάλματα των κόμβων ή εισαγωγή νέων. Υποθέτοντας ότι ένας αποστολέας ενημέρωσης έχει καταφέρει να συλλέξει μ_{lock} ψήφους, αν μέχρι λ πλησιέστεροι κόμβοι αποσυνδεθούν μετά που θα δώσουν την άδειά τους, πριν ή μετά που θα λάβουν την ενημέρωση, η συναλλαγή θα ολοκληρωθεί κανονικά. Αν μέχρι $\lambda_{timeout} \leq \lambda$ κόμβοι δεν απαντήσουν εντός του προκαθορισμένου χρονικού πλαισίου κατά τη διάρκεια της φάσης που συλλέγονται οι ψήφοι, ο αποστολέας μπορεί να συνεχίσει, όσο $\mu_{lock} - \lambda_{timeout}$ κόμβοι της ομάδας έχουν δώσει κανονικά την άδειά τους. Αναμενόμενη είναι η ακούσια κακή συμπεριφορά και όταν νέοι κόμβοι εισέρχονται στο δίκτυο ή όταν το πλέγμα των ομότιμων έχει ιδιαίτερα δυναμικό χαρακτήρα. Οι νέοι συμμετέχοντες λαμβάνουν τις τιμές που τους αντιστοιχούν σύμφωνα με τη διαδικασία εισόδου στο δίκτυο, αλλά αυτό συμβαίνει αφού τους γνωρίσουν οι γειτονικοί τους κόμβοι και τους τοποθετήσουν στους πίνακες δρομολόγησής τους. Ως αποτέλεσμα, μπορεί κάποιοι νέοι συμμετέχοντες να εμφανίζονται στην ομάδα των πλησιέστερων κόμβων μιας τιμής, χωρίς να έχουν λάβει ακόμα το σχετικό αντικείμενο, με αποτέλεσμα να αρνούνται να επιτρέψουν οποιαδήποτε σχετική πράξη. Η δυναμική του φυσικού δικτύου διασύνδεσης μπορεί επίσης να προκαλέσει την έξοδο και μετέπειτα επανείσοδο ενός κόμβου σε κάποια ομάδα, έχοντας χάσει στο ενδιάμεσο διάστημα κάποιες φάσεις μιας ενημέρωσης.

3.4.3 Λεπτομέρειες υλοποίησης

Το γεγονός ότι κάποιοι από τους πλησιέστερους σε μια τιμή κόμβους μπορεί να αποτύχουν ή να μη συμπεριφέρονται συμβατά με το πρωτόκολλο, λαμβάνεται υπόψη και στον αλγόριθμο που χρησιμοποιείται για την ανάκτηση δεδομένων. Αφού μέχρι λ ομότιμοι μπορεί να επιστρέψουν οποιαδήποτε τιμή, ο αρχικός κόμβος πρέπει να εντοπίσει τουλάχιστον $2\lambda + 1$ πρωτεύοντα αντίγραφα προκειμένου να καταλήξει σε συνεπή δεδομένα. Έτσι, για την ανάκτηση ενός κλειδιού, απαιτείται η πλήρης ολοκλήρωση ενός βρόχου `FIND_NODE`, για τον εντοπισμό όλων των μελών της αντίστοιχης απαρτίας. Ύστερα, απαιτείται η αποστολή ΑΚΣ τύπου `FIND_VALUE` σε ένα υποσύνολο $2\lambda + 1$ κόμβων. Η πλειοψηφία (τουλάχιστον $\lambda + 1$) οφείλει να απαντήσει με την ίδια τιμή, εκτός αν το συγκεκριμένο αντικείμενο βρίσκεται σε φάση ενημέρωσης και κάποια μέλη δεν έχουν λάβει ακόμα τις σχετικές ενημερώσεις. Σε αυτή τη περίπτωση, η διαδικασία ανάκτησης διακόπτεται και επαναλαμβάνεται μετά από κάποιο χρονικό διάστημα για προκαθορισμένο αριθμό επαναλήψεων. Να σημειωθεί ότι με τη συγκεκριμένη υλοποίηση τα δεδομένα μπορούν να αλλοιωθούν και αν οι "Βυζαντινοί" κόμβοι επιστρέφουν συλλογικά μια παλαιά τιμή του αντικειμένου. Αρκεί τότε ένας κόμβος που δεν έχει λάβει τις απαραίτητες ενημερώσεις να απαντήσει με την ίδια τιμή για να βγει τελικά "λανθασμένο" αποτέλεσμα. Αν και δεν θα είναι άκυρα τα δεδομένα, μια διαφορετική πολιτική ανάκτησης θα μπορούσε να απαιτεί παραπάνω συνεπείς απαντήσεις ή ακόμα και τον αμοιβαίο αποκλεισμό του αντικειμένου πριν την αποστολή των ΑΚΣ τύπου `FIND_VALUE`.

Για την προστασία της συναλλαγής ενημέρωσης από άπληστους κόμβους και την αποφυγή αδιεξόδων στη διαδικασία κλειδώματος, ορίζεται ότι όλοι οι ψήφοι λήγουν μετά από ένα προκαθορι-

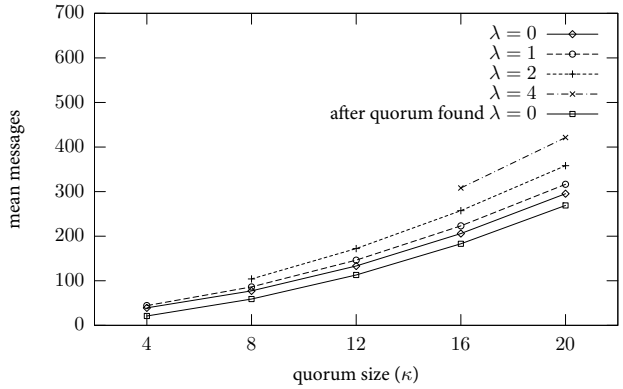
σμένο χρονικό διάστημα. Επίσης, κόμβοι που συναγωνίζονται για την ενημέρωση μια τιμής, θα πρέπει σε περίπτωση αποτυχίας περιμένουν για εκθετικά αυξανόμενο χρόνο πριν επαναλάβουν τη διαδικασία. Και οι δύο συνθήκες προστατεύονται και ελέγχονται από τα μέλη της κάθε αντίστοιχης ομάδας πρωτευόντων αντιγράφων. Αν ένας ομότιμος λάβει μήνυμα `GRANTED`, πρέπει να απαντήσει με `UPDATE` ή `YIELD` πριν λήξει η ισχύς της ψήφου που του έχει αποδοθεί. Σε αντίθετη περίπτωση, ο κόμβος "τιμωρείται" με τον αποκλεισμό του από τη διαδικασία αδειοδότησης για ένα μεγάλο χρονικό διάστημα (το αντίστοιχο μέλος δεν του δίνει τη ψήφο του). Επιπλέον, η λήψη του πρώτου μηνύματος `UPDATE` για ένα αντικείμενο σε κάθε κόμβο, σηματοδοτεί την αρχή μιας ενημέρωσης η οποία και θα πρέπει να ολοκληρωθεί σε συγκεκριμένο χρονικό περιθώριο. Το πρωτόκολλο ορίζει ότι θα πρέπει να απελευθερώνεται ο πόρος όταν ολοκληρωθεί η διαδικασία ενημέρωσης ή λήξει ένα από τα δύο χρονικά περιθώρια (αποστολής ενημέρωσης και ολοκλήρωσης συναλλαγής). Έτσι, ούτε ο κόμβος που κάνει την ενημέρωση, ούτε κάποιο από τα μέλη της απαρτίας, μπορούν εκούσια να καθυστερήσουν οποιαδήποτε φάση της ανανέωσης μιας τιμής. Ανάλογη συμπεριφορά μπορεί, βέβαια, να παρατηρείται και λόγω του δικτύου διασύνδεσης ή σε προβλημάτων στο φυσικό επίπεδο των κόμβων.

Το XOROS χρησιμοποιεί την "τυπική" διεπαφή των ΚΠΚ, που αποτελείται από τις εντολές get/put. Έτσι, μπορεί να χρησιμοποιηθεί απευθείας στη θέση κάποιου άλλου ΚΠΚ για τις ανάγκες μιας εφαρμογής που είναι κατασκευασμένη για να λειτουργεί επί ενός δομημένου ΔΟΚ. Επιπλέον, οι πράξεις αυτές επιστρέφουν και αποτέλεσμα — κάτι που δεν είναι συμβαίνει σε αντίστοιχα συστήματα. Η εισαγωγή/ενημέρωση μιας τιμής (put), μπορεί να επιστρέψει μήνυμα λάθους αν π.χ. ο ομότιμος δεν μπορεί να επιτύχει τον αμοιβαίο αποκλεισμό σε προκαθορισμένο χρονικό διάστημα. Η διεπαφή επιτρέπει επίσης τις εφαρμογές να ορίσουν μια συνάρτηση ενημέρωσης για την περίπτωση που απαιτείται μια ατομική πράξη ανάγνωσης-ενημέρωσης-εγγραφής. Η ανάκτηση (get) μπορεί επίσης να επιστρέψει σφάλμα, αν η τιμή δεν βρεθεί στο δίκτυο ή η λειτουργία δεν ολοκληρωθεί στο προβλεπόμενο χρονικό όριο. Οι εφαρμογές μπορούν να αγνοήσουν τους κωδικούς σφάλματος του XOROS, αλλά τότε δεν θα μπορούν να εκμεταλλευτούν τα επιπλέον χαρακτηριστικά που προσφέρονται από το δίκτυο.

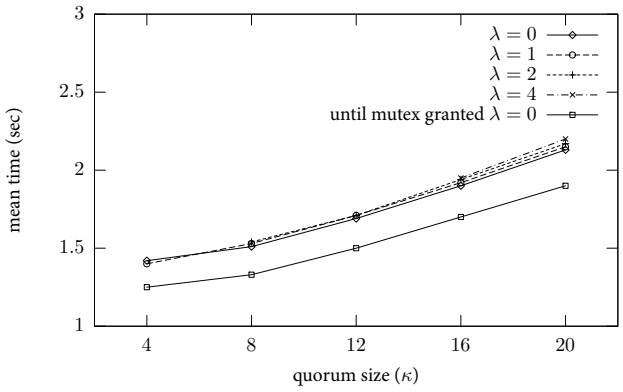
3.4.4 Αξιολόγηση

Το XOROS έχει υλοποιηθεί πλήρως, ως μια επέκταση του Kademlia, στα πλαίσια της πλατφόρμας προσομοίωσης και εκτέλεσης δικτύων ομότιμων κόμβων PeerPlatform. Επιγραμματικά, το ιδιαίτερο χαρακτηριστικό της υλοποίησης είναι η καθολική χρήση "ειδοποιήσεων" (notifications) για την επικοινωνία μεταξύ των κόμβων, αλλά και τη χρήση υπηρεσιών του συστήματος, όπως την αρχικοποίηση νέων κόμβων, τον χρονοπρογραμματισμό μελλοντικών ειδοποιήσεων, τη συλλογή στατιστικών στοιχείων κτλ. Ο πυρήνας της πλατφόρμας είναι ένας ασύγχρονος διαχειριστής ειδοποιήσεων, που αναλαμβάνει την εκτέλεση των αντίστοιχων συναρτήσεων (callbacks), ανάλογα με τα στοιχεία του αποστολέα, του παραλήπτη και το αναγνωριστικό της κάθε ειδοποίησης. Έτσι, οι ΑΚΣ μεταξύ των κόμβων υλοποιούνται ως ειδοποιήσεις, χωρίς να απασχολεί τον αλγόριθμο αν ο παραλήπτης είναι τοπικός (κόμβος που εκτελείται στην ίδια πλατφόρμα) ή απο-

μακρυσμένος (κόμβος με τον οποίο πρέπει γίνει επικοινωνία μέσω δικτύου). Με τη πλατφόρμα εκτέλεσης και προσομοίωσης ΔΟΚ, είναι δυνατόν να εκτελούνται ταυτόχρονα πολλαπλοί κόμβοι στο ίδιο φυσικό μηχάνημα ή σε μια συστοιχία μηχανημάτων. Έτσι, μας δίνεται η δυνατότητα να μιμηθούμε σε εργαστηριακό περιβάλλον — χρησιμοποιώντας πραγματικό ή εικονικό χρόνο — τη λειτουργία μεγάλων ΚΠΚ σε πραγματικές συνθήκες δικτύου (packet-level simulations).



(α′) Μέσος αριθμός μηνυμάτων ανά ενημέρωση



(β′) Μέσος χρόνος ανά ενημέρωση

Σχήμα 3.7: *Αποτελέσματα πολλαπλών εκτελέσεων του ίδιου σεναρίου για διαφορετικές τιμές του κ και του λ*

Προκειμένου να αξιολογήσουμε τα χαρακτηριστικά απόδοσης του XOROS, δημιουργήσαμε ένα σενάριο όπου 1024 κόμβοι εκτελούν τυχαίες ενημερώσεις σε ένα σύνολο 1024 αποθηκευμένων τιμών. Ο ρυθμός εκτέλεσης ορίστηκε σε 1 $\frac{\text{ενημέρωση}}{\text{δευτερόλεπτο}}$. Στο Σχήμα 3.7, παρουσιάζονται τα αποτελέσματα από πολλαπλά πειράματα με βάση το ίδιο δίκτυο, αλλά διαφορετικές τιμές του κ και του λ. Η παράμετρος α του Kademlia είναι πάντα 3. Στον κάθετο άξονα συμβολίζεται ο μέσος αριθμός μηνυμάτων και ο μέσος χρόνος που χρειάζεται για να ολοκληρωθεί επιτυχώς κάθε πράξη. Στο περιβάλλον εκτέλεσης, γίνεται και προσομοίωση του χρόνου απόκρισης στο επίπεδο του δικτύου. Στα συγκεκριμένα πειράματα ο μέσος χρόνος καθυστέρησης στην επιστροφή ενός πακέτου (roundtrip latency) κατανέμεται μεταξύ 120 και 180 χιλιοστά του δευτερολέπτου.

Ο αριθμός μηνυμάτων που παράγεται για κάθε ενημέρωση δεδομένων, αυξάνεται σε συνάρτηση με το πλήθος των κόμβων που πρέπει να ελέγξουν την αντίστοιχη πράξη. Η παράμετρος ανοχής του δικτύου σε σφάλματα δεν επηρεάζει τη μορφή, αλλά προκαλεί μια ελάχιστη μετατόπιση

των γραφημάτων του Σχήματος 3.7(α′) προς τα άνω (περισσότερα μηνύματα). Ανατρέχοντας στον αλγόριθμο που εκτελείται για κάθε συναλλαγή ενημέρωσης, βλέπουμε ότι τα βήματα 2, 3 και 4 του πρωτοκόλλου απαιτούν το πολύ κ μηνύματα το καθένα, ενώ στη φάση ολοκλήρωσης δημιουργούνται $\mu_{lock}(\kappa - 1)$ ΑΚΣ, το οποίο περιέχει έναν όρο κ^2. Από την άλλη μεριά, είναι προφανές από το Σχήμα 3.7(β′), ότι το κόστος επικοινωνίας κατανέμεται στο δίκτυο — τα μηνύματα ανταλλάσσονται παράλληλα και ο συνολικός χρόνος που απαιτείται για κάθε πράξη επηρεάζεται ελάχιστα από το κ. Τα πρώτα τέσσερα γραφήματα του Σχήματος 3.7(β′) αντιστοιχούν στον χρόνο που απαιτείται για κάθε συναλλαγή ενημέρωσης, από τη στιγμή που δίνεται η εντολή έως ότου και ο τελευταίος πρωτεύων κόμβος αποθηκεύσει τα σχετικά δεδομένα. Το πέμπτο γράφημα, παρουσιάζει το χρονικό διάστημα που καταλαμβάνει ο μηχανισμός αμοιβαίου αποκλεισμού (σημειώνεται ως "until mutex granted"). Μετριέται δηλαδή ο χρόνος μέχρι να αποδοθούν οι άδειες στον ομότιμο που εκτελεί την ενημέρωση. Ενδιαφέρον παρουσιάζει το γεγονός ότι η διαφορά μεταξύ αυτής της μέτρησης και του συνολικού χρόνου ολοκλήρωσης της συναλλαγής (για το ίδιο λ) είναι σταθερή. Σε αυτό το διάστημα, αποστέλλεται και προωθείται η νέα τιμή, δηλαδή ένας κόμβος εκτελεί μ_{lock} ΑΚΣ τύπου `UPDATE` στους πρωτεύοντες κόμβους ενός κλειδιού, και εκείνοι με τη σειρά τους παράγουν $\mu_{lock}(\kappa - 1)$ παράλληλα `COMMIT`. Παρατηρείται ότι για τον λόγο αυτό, η σταθερή διαφορά είναι ίση με τον μέγιστο χρόνο καθυστέρησης στην επιστροφή ενός πακέτου. Αυτή η παράμετρος διαμορφώνει σε μεγάλο βαθμό τα γραφήματα του Σχήματος 3.7(β′).

Ο μέσος χρόνος επηρεάζεται από το πλήθος των πρωτευόντων κόμβων μόνο λόγω του πρώτου βήματος του πρωτοκόλλου, όπου εντοπίζονται οι πλησιέστεροι κόμβοι του αντικειμένου. Στο Σχήμα 3.7(α′) φαίνεται πόσα μηνύματα απαιτούνται για τη φάση αναζήτησης της απαρτίας, αφού μετράμε χωριστά πόσα μηνύματα δημιουργούνται στα βήματα 2-5 της συναλλαγής (σημειώνεται ως "after quorum found"). Αν και το πλήθος των αντίστοιχων ΑΚΣ τύπου `FIND_NODE` είναι σχετικά μικρό, οι πράξεις αυτές εκτελούνται σε μόλις α παράλληλα μονοπάτια — μια διαδικασία που απαιτεί περισσότερο χρόνο όσο αυξάνεται το κ.

3.4.5 Συμπεράσματα

Οι Κατανεμημένοι Πίνακες Κατακερματισμού χρησιμοποιούνται ευρέως από μια πληθώρα εφαρμογών με ανάγκες υψηλής κλιμάκωσης και μεγάλης χωρητικότητας. Τα μοναδικά τους χαρακτηριστικά έχουν επιτρέψει σε πολλά συστήματα να λειτουργούν στα μεγέθη του Διαδικτύου. Παρόλα αυτά, πιστεύουμε ότι ο σχεδιασμός τους έχει ακόμα μεγαλύτερη δυναμική ανάπτυξης. Το ιδεατό υπόστρωμα αποθήκευσης που δημιουργείται στερείται σημαντικής λειτουργικότητας, δεδομένου ότι περιορίζει τις εφαρμογές σε μια υποδομή που επιτρέπει την ανάγνωση μόνο των αποθηκευμένων τιμών. Ένας ΚΠΚ που επιτρέπει την ενημέρωση των αντικειμένων μπορεί να καλύψει ένα μεγαλύτερο σύνολο απαιτήσεων από τις εφαρμογές και να φανεί εξαιρετικά χρήσιμος και σημαντικός για την υλοποίηση μιας ολόκληρης γενιάς μελλοντικών κατανεμημένων συστημάτων.

Για το σκοπό αυτό παρουσιάσαμε το XOROS — έναν ΚΠΚ, που αντιμετωπίζει το πρόβλημα συ-

νεπούς ενημέρωσης των πρωτευόντων αντιγράφων των αποθηκευμένων τιμών. Το πρωτόκολλο που χρησιμοποιείται, βασίζεται σε προηγούμενες εργασίες που αναφέρονται στον κατανεμημένο αμοιβαίο αποκλεισμό και την επικοινωνία υπό την παρουσία "Βυζαντινών" κόμβων. Μάλιστα, οι πράξεις κλειδώματος και αλλαγής μια τιμής συγχωνεύονται σε μία συναλλαγή, που με τη σειρά της ελέγχεται και οδηγείται από τους πλησιέστερους κόμβους του αντίστοιχου κλειδιού. Η ομάδα που είναι υπεύθυνη για τον αμοιβαίο αποκλεισμό αποτελείται από τους ίδιους κόμβους που αποθηκεύουν και τα δεδομένα προς ενημέρωση. Έτσι, δεν επηρεάζονται οι δυνατότητες κλιμάκωσης του συστήματος, ενώ η συνολική ανοχή σε σφάλματα είναι υπολογίσιμη.

Τα αποτελέσματα αξιολόγησης του XOROS είναι ενθαρρυντικά. Δείχνουν ότι αν και αυξήσεις στην παράμετρο αντιγραφής και του ποσοστού των κακόβουλων κόμβων έχουν ως αποτέλεσμα περισσότερα μηνύματα, η συνολική ικανότητα εξυπηρέτησης του συστήματος δεν πέφτει αισθητά. Όμως, εκτός από τις παραμέτρους επικοινωνίας, και τα α, κ και λ, ο ρυθμός εξυπηρέτησης του συνολικού ΚΠΚ, μετρούμενος σε επιτυχημένες $\frac{\text{ενημερώσεις}}{\text{δευτερόλεπτο}}$ εξαρτάται και από τον φόρτο και τη δυναμική του δικτύου, τον χρόνο αναμονής για την ολοκλήρωση μιας συναλλαγής, τον χρόνο αναμονής για αποστολή μιας τιμής, κτλ. Άλλα πειράματα, φανερώνουν ότι όταν ο ρυθμός αιτήσεων φτάνει και ξεπερνά ένα όριο, το σύστημα αδυνατεί να αντεπεξέλθει και μειώνεται απότομα ο ρυθμός εξυπηρέτησης (denial-of-service). Η στρατηγική που ακολουθείται σε περίπτωση αποτυχίας του αμοιβαίου αποκλεισμού, με την εθελούσια αυξανόμενη αναμονή για επόμενη αίτηση, βοηθά στην διατήρηση ενός ποσοστού επιτυχίας. Επιπρόσθετα, θα μπορούσε να ενσωματωθεί μια "ουρά αιτήσεων" στους κόμβους, ώστε να αξιοποιείται η τελευταία φάση της συναλλαγής, προκειμένου να δημιουργείται μια συνεπή εικόνα της σειράς με την οποία πρέπει να εξυπηρετηθούν οι ενημερώσεις. Μια ενδεικτική εφαρμογή στην οποία χρησιμοποιήθηκε το XOROS, παρουσιάζεται στο κεφάλαιο 5 και στην εργασία [ZDCK07].

Κεφάλαιο 4

Αναζήτηση δεδομένων

Σε αυτό το κεφάλαιο παρουσιάζουμε και αξιολογούμε ένα πρωτόκολλο που επιτρέπει την γρήγορη και αποτελεσματική εκτέλεση λειτουργιών αναζήτησης σε πολυδιάστατα δεδομένα που αποθηκεύονται σε ΚΠΚ. Έτσι, εκτός από την εύρεση συγκεκριμένων, *σημειακών* δεδομένων με βάση το κλειδί που τους αντιστοιχεί, μπορούν να επιτελούνται και πράξεις που αφορούν ένα σύνολο πληροφοριών. Οι λειτουργίες αναζήτησης που περιγράφονται είναι ιδιαίτερα χρήσιμες όταν στο ΔΟΚ τοποθετούνται ομάδες ή συλλογές δεδομένων, των οποίων τα κλειδιά έχουν παραχθεί με τέτοιο τρόπο ώστε να διατηρούνται οι μεταξύ τους σχέσεις και στο επίπεδο της επικάλυψης. Συνήθως αυτό επιτυγχάνεται μέσω μιας συνάρτησης παραγωγής κλειδιών, που κωδικοποιεί τη σημασιολογική εγγύτητα των δεδομένων στις αποστάσεις που χωρίζουν τα ζεύγη κλειδιού-τιμής – *σχετικές* τιμές παραμένουν *κοντά* και στον εικονικό χώρο διευθύνσεων του ΚΠΚ.

Σε αντίθεση με συναφείς στο αντικείμενο εργασίες, ο προτεινόμενος μηχανισμός αναζήτησης μπορεί να λειτουργήσει και σε ΚΠΚ που αυτοδιαχειρίζονται αντίγραφα των αποθηκευμένων τιμών. Ο αλγόριθμος που χρησιμοποιείται βασίζεται στον ορισμό της *γειτονιάς κόμβων* ή αλλιώς της *γειτονιάς αντιγράφων*, προκειμένου να ελαχιστοποιήσει τον αριθμό των εμπλεκόμενων κόμβων σε κάθε αναζήτηση και άρα και το σύνολο των μηνυμάτων που απαιτούνται για την ολοκλήρωση μιας αίτησης. Επιπρόσθετα, δείχνουμε πως η ενσωμάτωση της λειτουργίας αναζήτησης στο χαμηλότερο επίπεδο της υλοποίησης ενός κόμβου – το επίπεδο της δρομολόγησης – επιτρέπει την αυτόματη προσαρμογή του πρωτοκόλλου στα δυναμικά χαρακτηριστικά του δικτύου. Για τις ανάγκες της αναζήτησης επαναχρησιμοποιούνται οι πινάκες δρομολόγησης που διατηρεί και ανανεώνει συνεχώς κάθε κόμβος, χωρίς να χρειάζονται άλλες, πρόσθετες δομές και μηχανισμοί διαχείρισης ή παρακολούθησης.

4.1 Το πρόβλημα της αναζήτησης δεδομένων

Οι ΚΠΚ έχουν καθιερωθεί ως η πλέον αποδοτική λύση για την κατανομή "κλασικών" κεντρικοποιημένων βάσεων δεδομένων σε δίκτυα παγκόσμιας κλίμακας, ειδικά όταν οι πληροφορίες

που αποθηκεύονται μπορούν να αναπαρασταθούν ως ένα ζεύγος κλειδιού-τιμής. Χαρακτηριστικό παράδειγμα αυτής της μορφής δεδομένων είναι οι καταχωρήσεις ονομάτων διαδικτύου (Domain Name System – DNS records), που αποτελούνται από ένα κλειδί (το όνομα διαδικτύου) και την αντίστοιχη τιμή (τη διεύθυνση δικτύου ή κάποιον δείκτη σε αυτή, κτλ.). Μέχρι και τις μέρες μας χρησιμοποιείται μια δενδρική δομή κεντρικών εξυπηρετητών για τη διαχείριση και την αποθήκευση των καταχωρήσεων ονομάτων, αν και έχουν προταθεί και λύσεις που βασίζονται σε δομημένα ΔΟΚ [CMM02]. Να σημειωθεί ότι όταν απαιτούνται συχνές ενημερώσεις στις τιμές ένας τυπικός ΚΠΚ δεν είναι αρκετός, εκτός αν εφαρμοστούν οι αλγόριθμοι που περιγράφονται στο κεφάλαιο 3. Από την άλλη μεριά, αν τα δεδομένα που αποθηκεύονται δεν αλλάζουν ποτέ ή μπορεί να θεωρηθεί ότι "λήγουν" μετά από ένα χρονικό διάστημα, η χρήση ενός ΚΠΚ όχι μόνο επιτρέπει την άμεση κατανομή του όγκου των δεδομένων, αλλά και διασφαλίζει ότι τα ανώτερα επίπεδα της εφαρμογής μπορούν να χρησιμοποιούν ένα απλό προγραμματιστικό μοντέλο για την διαχείριση της πληροφορίας, χωρίς να απασχολούνται με τις λεπτομέρειες υλοποίησης.

Εξαίρεση στον παραπάνω γενικό κανόνα, αποτελούν οι εφαρμογές που εργάζονται απευθείας σε υποσύνολα δεδομένων. Τότε, παρόλο που κάθε λειτουργία μπορεί να εκφραστεί μέσω πολλαπλών κλήσεων τοποθέτησης/εντοπισμού (put/get), η συνολική απόδοση μπορεί να χειροτερεύει, όσο πολλαπλασιάζεται ο αριθμός των παραγόμενων πρωτογενών πράξεων. Ένα πρόγραμμα που εκτελεί λειτουργίες αναζήτησης με βάση κάποιους όρους ή ζητά από το υπόστρωμα αποθήκευσης να επιστρέψει υποσύνολα ομοειδών δεδομένων, ανήκει σε αυτή τη κατηγορία. Μια πράξη αναζήτησης σε δεδομένα που δεν έχουν τοποθετηθεί με κάποια ταξινόμηση, περιλαμβάνει την εξέταση όλων των διαθέσιμων τιμών – κάτι που μπορεί στη περίπτωση ενός ΚΠΚ είναι αδύνατον να ολοκληρωθεί σε λογικό χρονικό διάστημα, λόγω του υπέρογκου πλήθους των πράξεων εντοπισμού που πρέπει να ολοκληρωθούν. Τα δομημένα ΔΟΚ είναι βελτιστοποιημένα για λειτουργίες που αφορούν συγκεκριμένα *σημεία* της επικάλυψης και όχι υποπεριοχές του χώρου διευθύνσεων. Αν υπήρχε ένα πρωτόκολλο που να επέτρεπε την εργασία σε υποσύνολα του χώρου, είναι σίγουρο ότι η εφαρμοσιμότητα των ΚΠΚ θα εξαπλωνόταν σε ακόμα περισσότερα πεδία εφαρμογών.

Η υλοποίηση υπηρεσιών αναζήτησης ομάδων δεδομένων σε δομημένα ΔΟΚ είναι στόχος πολλών σύγχρονων ερευνητικών εργασιών. Ιδανικά, όπως αναφέρεται, το δίκτυο επικάλυψης θα πρέπει να υποστηρίζει *προηγμένες* ή *σύνθετες* λειτουργίες εντοπισμού, που να είναι σε θέση να εκμεταλλεύονται τις όποιες σημασιολογικές σχέσεις μεταξύ των δεδομένων που αποθηκεύονται. Για παράδειγμα, δεδομένου μιας τιμής, θα πρέπει να προσφέρεται η αναζήτηση παρόμοιων ή σχετικών τιμών. Αυτό προϋποθέτει βέβαια ότι κάθε πληροφορία προσδιορίζεται στη βάση ενός ή περισσότερων χαρακτηριστικών – για να αναφερθούμε μονοσήμαντα σε κάθε δεδομένο μπορούμε να χρησιμοποιήσουμε τις τιμές των χαρακτηριστικών αυτών. Όλες οι εργασίες συγκλίνουν στο να προτείνουν έναν εναλλακτικό τρόπο παραγωγής κλειδιών για την τοποθέτηση τέτοιου είδους δεδομένων σε έναν ΚΠΚ. Η κλασική συνάρτηση κατακερματισμού παράγει έναν αριθμό σύμφωνα με το περιεχόμενο των δεδομένων, οπότε χάνεται η σημασιολογική πληροφορία. Η κεντρική ιδέα είναι ότι η θέση που παίρνει το αντικείμενο μέσα στον εικονικό χώρο διευθύνσεων του ΚΠΚ, πρέπει να σχετίζεται άμεσα με τις τιμές των χαρακτηριστικών του. Επιπρόσθετα, πρέπει να διατηρείται η εγγύτητα μεταξύ των αντικειμένων κατά τη μετατροπή των αναγνωριστικών

τους από τον πολυδιάστατο χώρο που δημιουργεί το σύνολο τιμών τους, στον αριθμό που τους αντιστοιχεί στο δίκτυο επικάλυψης. Τιμές με *κοντινά* χαρακτηριστικά πρέπει να καταλήγουν και σε *κοντινές* θέσεις στον ΚΠΚ.

Για το σκοπό αυτό μπορεί να χρησιμοποιηθεί μια ειδική συνάρτηση, η οποία κατά τη μετατροπή ενός πολυδιάστατου διανύσματος σε έναν μονοσήμαντο αριθμό και αντίστροφα, διατηρεί – όσο είναι δυνατόν – την εγγύτητα σε κάθε διάσταση. Σε αυτό τον τύπο συναρτήσεων ανήκει και η οικογένεια των Καμπυλών Πλήρωσης Χώρου (Space Filling Curve – SFC), που περιλαμβάνει και τις ευρέως διαδεδομένες καμπύλες Z και Hilbert, Οι συναρτήσεις αυτές ονομάζονται έτσι, γιατί η γεωμετρική τους αναπαράσταση είναι μια γραμμή που διατρέχει το σύνολο ενός πολυδιάστατου χώρου, χωρίς να διακόπτεται και χωρίς να διασταυρώνεται με τον εαυτό της. Η χρήση μιας τέτοιας καμπύλης για την παραγωγή των κλειδιών των δεδομένων, λύνει το πρόβλημα της τοποθέτησης και ταξινόμησης – απαραίτητο χαρακτηριστικό για την εκτέλεση πράξεων σε ομάδες τιμών.

Από τη μελέτη αλγορίθμων αναζήτησης για ΚΠΚ που έχουν ήδη προταθεί, την υλοποίηση και αξιολόγηση ορισμένων από αυτούς, διακρίνουμε δύο βασικά μειονεκτήματα: Είτε στην συνολική αρχιτεκτονική του συστήματος, που θέλει τη μηχανή αναζήτησης σε ξεχωριστό στρώμα από το ΔΟΚ, είτε στην επιλογή της "δενδρικής" προώθησης των αιτήσεων. Στη πρώτη περίπτωση, τα πολυδιάστατα διανύσματα αναζήτησης μεταφράζονται από το στρώμα του αλγορίθμου σε κατατμήσεις του χώρου διευθύνσεων (τμήματα της γραμμής που εννοείται από τη καμπύλη που διατρέχει τον πολυδιάστατο χώρο), ώστε να αποσταλούν οι αντίστοιχες εντολές εντοπισμού στο στρώμα του ΚΠΚ. Παράγονται, δηλαδή, και αναζητούνται όλα τα πιθανά κλειδιά που μπορούν να απαντούν στο αρχικό ερώτημα. Αυτό απαιτεί αφενός ένα σημαντικό υπολογιστικό φόρτο, ανάλογα βέβαια και με την καμπύλη που χρησιμοποιείται, αφετέρου δε διαπιστώνει κανείς ότι το μόνο που επιτυγχάνεται έτσι είναι να μετατοπίζεται το αρχικό πρόβλημα της παραγωγής πολλαπλών πρωτογενών πράξεων από το επίπεδο της εφαρμογής, στο επίπεδο του αλγόριθμου αναζήτησης. Επιπλέον, η διαφοροποίηση του επιπέδου αναζήτησης με το επίπεδο δρομολόγησης, δημιουργεί διαχειριστικά προβλήματα, αφού οι διακριτές δομές δεδομένων των δύο επιπέδων χρειάζεται να συγχρονίζονται ώστε να ενημερώνεται το σύστημα για αλλαγές στο σύνολο των αποθηκευμένων τιμών ή των κόμβων που συμμετέχουν στην επικάλυψη.

Κάποια συστήματα ενσωματώνουν τη μηχανή αναζήτησης στον ΚΠΚ και στη συνέχεια κατανέμουν το φόρτο παραγωγής των κλειδιών και τα μηνύματα εντοπισμού στους διαθέσιμους κόμβους. Η πιο σύνηθης τακτική είναι να θεωρείται η διάταξη των κόμβων στον εικονικό χώρο της επικάλυψης ως δενδρική και να προωθούνται αναδρομικά οι αιτήσεις αναζήτησης από τη κορυφή του δέντρου, στα φύλλα – από το γενικό στο ειδικό. Αρχικά το αίτημα δίνεται στον κόμβο που θεωρείται ότι "καλύπτει" το σύνολο του χώρου, ο οποίος αφού υπολογίσει το πρώτο βήμα της αναδρομής παράγει μια σειρά πιο συγκεκριμένων κατατμήσεων, που προωθούνται στους αντίστοιχους κόμβους, κτλ. Έτσι όμως, εκτός του ότι επιβάλλεται συνεχής φόρτος στους κόμβους που είναι πλησιέστερα στη κορυφή του δέντρου, δεν λύνεται το πρόβλημα του να πρέπει να υπολογιστούν όλα τα δυνατά κλειδιά που μπορεί να απαντούν το αίτημα. Ας σημειωθεί εδώ ότι λόγω του μεγέθους του χώρου διευθύνσεων ενός ΚΠΚ, είναι μεγάλη η πιθανότητα τα περισ-

σότερα από αυτά τα κλειδιά να είναι άκυρα – να μην έχουν καταληφθεί δηλαδή από τιμές.

Ένα ακόμη ζήτημα που προέκυψε από τη μελέτη των συστημάτων αναζήτησης, είναι η έλλειψη αλγορίθμων που να είναι συμβατοί με ΚΠΚ που αυτοδιαχειρίζονται εγγενώς αντίγραφα των αποθηκευμένων τιμών. Η αξιόπιστη διαχείριση δεδομένων στα πρώτα δομημένα ΔΟΚ διασφαλιζόταν μέσω καλά ορισμένων διαδικασιών εισαγωγής και εξαγωγής κόμβων από την επικάλυψη. Γρήγορα φάνηκε όμως, πως η τήρηση αυστηρών πρωτοκόλλων τέτοιου τύπου είναι πολλές φορές πρακτικά αδύνατη λόγω της δυναμικότητας του δικτύου και είναι πιο εύκολο και αποδοτικό απλά να τηρούνται πολλαπλά αντίγραφα κάθε τιμής σε διαφορετικούς συμμετέχοντες. Τα προτεινόμενα από τη βιβλιογραφία μοντέλα αναζήτησης σε ΚΠΚ, συνήθως θεωρούν την επικάλυψη ως ένα απλό σχήμα δρομολόγησης και αναθέτουν τη διαχείριση αντιγράφων σε κάποιο ξεχωριστό σύστημα ή επίπεδο. Όταν, όμως, τέτοιου τύπου αντίγραφα τηρούνται ήδη στο επίπεδο του ΔΟΚ, ο αλγόριθμος αναζήτησης θα πρέπει να τα λαμβάνει υπόψη.

Στα πλαίσια της διατριβής παρουσιάζουμε τις αναγκαίες προσθήκες στο πρωτόκολλο ενός ΚΠΚ ώστε να είναι σε θέση το σύστημα να χειριστεί πράξεις αναζήτησης που αναφέρονται σε ομάδες ή συλλογές δεδομένων. Δεν προτείνεται μια νέα αρχιτεκτονική ΔΟΚ, αλλά δείχνεται πώς μπορεί το σύστημα να εκμεταλλευτεί τις ήδη υπάρχουσες εσωτερικές δομές που τηρεί κάθε κόμβος, για να υλοποιηθεί μια νέα στρατηγική δρομολόγησης αιτημάτων αναζήτησης. Σε αντίθεση με άλλα συστήματα, δεν χρησιμοποιούνται αναδρομικοί αλγόριθμοι υπολογισμού των κατατμήσεων του χώρου που αντιστοιχούν στο αίτημα, αλλά εφαρμόζεται μια τεχνική κατά την οποία οι απαντήσεις συλλέγονται στη σειρά από διάφορα σκέλη του δικτύου. Μάλιστα, καθώς τα σκέλη αυτά είναι ανεξάρτητα μεταξύ τους, η συνολική διαδικασία μπορεί να παραλληλοποιηθεί ή και να διαμοιραστεί περαιτέρω σε διαφορετικούς κόμβους του ΚΠΚ. Ανάλογοι αλγόριθμοι χρησιμοποιούνται εκτενώς σε συστήματα διαχείρισης βάσεων δεδομένων, αλλά όχι σε ΔΟΚ.

Η υλοποίηση της μηχανής αναζήτησης γίνεται απευθείας στο επίπεδο δρομολόγησης του ΚΠΚ, γεγονός που προσφέρει σημαντικά πλεονεκτήματα. Πρώτον, τα αιτήματα αναζήτησης αυτοδιαχειρίζονται το μονοπάτι που ακολουθούν μέσα από το δίκτυο επικάλυψης, αποφεύγοντας να επισκεφτούν κόμβους που θα απαντούσαν με αντίγραφα τιμών που έχουν ήδη ληφθεί υπόψη. Ο αλγόριθμος έχει επίγνωση των λειτουργιών αυτόματης αντιγραφής του δικτύου – κάτι που αποτελούσε εξαρχής μια από τις σημαντικότερες προκλήσεις κατά τον σχεδιασμό του συστήματος. Για το σκοπό αυτό, εισάγουμε την έννοια της *γειτονιάς*, ως την ομάδα των κόμβων που αποθηκεύουν αντίγραφα των ίδιων δεδομένων. Οι ομάδες αυτές επιτελούν σημαντικό ρόλο στη διαδικασία αναζήτησης και εξυπηρετούν ως εργαλείο που βοηθά στην εκτίμηση των χαρακτηριστικών απόδοσης του αλγορίθμου. Δεύτερον, εργαζόμενοι στο επίπεδο της δρομολόγησης του ΚΠΚ και μην εισάγοντας νέους πίνακες ή νέες δομές, δεν απαιτούμε και επιπλέον περιοδικές ή μη εργασίες διαχείρισης από τους κόμβους προκειμένου να εξισορροπείται το δίκτυο ή να ανανεώνονται δομές δεδομένων, κτλ. Για τους παραπάνω λόγους υποστηρίζουμε ότι ακόμα και η ίδια η διαδικασία ενσωμάτωσης του αλγορίθμου στο πρωτόκολλο του ΚΠΚ αποτελεί βασική συνεισφορά του συγκεκριμένου αλγορίθμου.

Στη συνέχεια, και αφού εξετάσουμε τις σχετικές δημοσιεύσεις στον τομέα, θα παρουσιάσουμε

αναλυτικά το πρωτόκολλο, ξεκινώντας με την μονοδιάστατη περίπτωση (απλή αναζήτηση ομάδων δεδομένων που προσδιορίζονται από ένα χαρακτηριστικό). Ύστερα, γίνεται η κλιμάκωση του συστήματος στις πολλές διαστάσεις, η ανάλυση της απόδοσης του αλγορίθμου – με επίσημο τρόπο αλλά και εμπειρικά, διασταυρώνεται η θεωρία με αποτελέσματα μετρήσεων και αναφέρεται πως επηρεάζουν οι τιμές των παραμέτρων του ΚΠΚ τη συμπεριφορά του συστήματος στις πράξεις αναζήτησης. Στο κεφάλαιο 5, περιγράφεται το σενάριο που αποτέλεσε το κίνητρο να ασχοληθούμε με το συγκεκριμένο ζήτημα, καθώς και οι λεπτομέρειες υλοποίησης της λύσης που χρησιμοποιήσαμε τελικά. Ο προτεινόμενος αλγόριθμος προσφέρει τις λειτουργίες αναζήτησης μεταδεδομένων στα πλαίσια του λογισμικού που αναπτύχθηκε για το ευρωπαϊκό ερευνητικό έργο Gredia.

4.2 Σχετικές εργασίες

Οι ΚΠΚ [MM02, SMK+01, RFH+01, RD01, HKRZ02] έχουν εδραιωθεί ως μια αποδοτική λύση στο πρόβλημα της κατανεμημένης τοποθέτησης δεδομένων και της δρομολόγησης σχετικών *σημειακών* αναζητήσεων (εντοπισμών). Με τα αντίστοιχα πρωτόκολλα, μια τιμή μπορεί να ανακτηθεί μόνο αν το ακριβές της κλειδί είναι εκ των προτέρων γνωστό, το οποίο με τη σειρά του δημιουργείται τυχαία, μέσω μιας συνάρτησης κατακερματισμού. Έτσι όμως, τα κλειδιά δεν περιέχουν καμία πληροφορία για τις τιμές στις οποίες αναφέρονται. Όμως, καθώς οι εφαρμογές απαιτούν όλο και πιο πολύπλοκες λειτουργίες επί των δεδομένων – ικανές να εκμεταλλεύονται τις μεταξύ τους σχέσεις, για παράδειγμα – η σχετική έρευνα στον τομέα έχει επικεντρωθεί στον σχεδιασμό και ανάπτυξη των κατάλληλων αλγορίθμων και μηχανισμών, που να επιτρέπει την εκτέλεση σύνθετων ερωτημάτων σε ΚΠΚ. Για την υποστήριξη λειτουργιών αναζήτησης εύρους διευθύνσεων, απαιτείται αρχικά μιας μορφής ταξινόμηση των διαθέσιμων τιμών, κάτι που έρχεται σε αντίθεση με την αρχική μεθοδολογία υλοποίησης των δομημένων ΔΟΚ – τη τυχαία αντιστοίχιση δεδομένων και κόμβων. Ανάλογα με τον τρόπο που αντιμετωπίζεται αυτό το πρόβλημα, μπορούμε να κατηγοριοποιήσουμε σε πρώτο επίπεδο τα σχετικά συστήματα που αναφέρονται στη βιβλιογραφία:

- Επικαλύψεις που βασίζονται σε έναν ΚΠΚ και είτε αλλάζουν τη συνάρτηση κατακερματισμού, είτε προσθέτουν νέες δομές ταξινόμησης σε ένα επιπλέον επίπεδο.
- Επικαλύψεις που ορίζουν τον δικό τους τρόπο κατανομής των δεδομένων στους κόμβους – τρόπο που θυμίζει λειτουργικά ένα δομημένο ΔΟΚ, χωρίς όμως να χρησιμοποιούν κάποιο διαδεδομένο πρωτόκολλο ΚΠΚ.

Στις επόμενες παραγράφους, παρουσιάζονται οι σχετικές εργασίες, ξεκινώντας από εκείνες που ασχολούνται με τη μονοδιάστατη αναζήτηση εύρους διευθύνσεων, πριν προχωρήσουμε σε συστήματα που λαμβάνουν υπόψη την ύπαρξη πολλαπλών αναγνωριστικών ανά τιμή και επιτρέπουν τη εκτέλεση ερωτημάτων σε όλες, ταυτόχρονα, τις διαστάσεις του χώρου που ορίζεται από τα δεδομένα.

Μια πρώτη προσέγγιση στο ζήτημα της υποστήριξης αναζητήσεων εύρους διευθύνσεων σε ΚΠΚ, αναφέρεται στην [GAA03]. Τμήματα μιας σχεσιακής βάσης δεδομένων αποθηκεύονται σε έναν ΚΠΚ και επιτρέπεται η αναζήτηση σε ένα αναγνωριστικό τη φορά, μέσω της συσχέτισης των τελευταίων με διευθύνσεις της επικάλυψης, με τη βοήθεια μιας ειδικής συνάρτησης κατακερματισμού. Τα αποτελέσματα της μεθόδου είναι προσεγγιστικά, καθώς η επιτυχία του μετασχηματισμού έχει πιθανοτικά κριτήρια και εξαρτάται από τη πολυπλοκότητα της συνάρτησης. Επιπλέον, προκύπτουν ζητήματα άνισης κατανομής του φόρτου στους συμμετέχοντες. Σε μια άλλη μέθοδο, που παρουσιάζεται στο [AX02], χρησιμοποιείται η καμπύλη Hilbert για την αντιστοίχιση ορίων διευθύνσεων με τμήματα ενός δικτύου CAN. Η ΚΠΧ προσφέρει το πλεονέκτημα ότι κοντινά όρια συσχετίζονται με κοντινά τμήματα της επικάλυψης. Μια αναζήτηση δρομολογείται αρχικά στον κόμβο που είναι υπεύθυνος για το κέντρο του εύρους διευθύνσεων και στη συνέχεια "διαδίδεται" σύμφωνα με μία εκ των τριών προτεινόμενων τεχνικών.

Μια κοινή μεθοδολογία που ακολουθείται από διάφορα συστήματα είναι η τοποθέτηση μιας επιπλέον δενδρικής δομής, ως ευρετήριο, σε ένα επίπεδο πάνω από τον ΚΠΚ. Ο ΚΠΚ σε αυτές τις περιπτώσεις χρησιμοποιείται ως μια ανεξάρτητη υποδομή δρομολόγησης. Μια τέτοια υλοποίηση προτείνεται και από τους σχεδιαστές του Range Search Tree (RST) [GS04], όπου το ευρετήριο τοποθετείται πάνω από μια επικάλυψη Chord. Κάθε επίπεδο του δέντρου αντιστοιχεί και σε διαφορετικό επίπεδο κατάτμησης των δεδομένων. Όλοι οι κόμβοι του ευρετηρίου καταχωρούνται μέσω της συνάρτησης κατακερματισμού στην επικάλυψη. Οι τιμές, ανάλογα με το διάστημα στο οποίο ανήκουν, τοποθετούνται σε όλα ή συγκεκριμένα επίπεδα του δέντρου και άρα τους ανάλογους κόμβους του δικτύου. Παρόμοια είναι και τα Distributed Segment Trees [ZSLS06]. Τα P-Trees [nPLGS04] είναι επίσης κατανεμημένα ευρετήρια για επικαλύψεις Chord, όπου κάθε κόμβος διαχειρίζεται και αποθηκεύει τμήματα ημι-ανεξάρτητων B+ δέντρων. Ο σκοπός των δέντρων είναι να βοηθούν στη δρομολόγηση των αναζητήσεων στους κατάλληλους κόμβους. Επειδή η διαχείριση των συγκεκριμένων δομών είναι πολύπλοκη, ορίζονται ειδικές διαδικασίες συγχρονισμού και ενημέρωσης των ευρετηρίων. Άλλες προσπάθειες, στοχεύουν στην κατανομή *tries*. Τα tries είναι μια γενίκευση των δέντρων, για την αποθήκευση και την επεξεργασία συμβολοσειρών, όπου δημιουργείται ένας κόμβος για κάθε κοινό πρόθεμα. Στα tries, τα δεδομένα αποθηκεύονται στα φύλλα της δομής. Μια πράξη εντοπισμού περιλαμβάνει την διαδοχική επίσκεψη όλων των κόμβων, των οποίων η ετικέτα αποτελεί πρόθεμα της τιμής που ζητάται, μέχρι να βρεθεί το κατάλληλο φύλλο. Το Prefix Hash Tree (PHT) [RRHS04] είναι μια τέτοια δομή, βασισμένη σε trie, που χρησιμοποιείται για την εκτέλεση μονοδιάστατων εντολών αναζήτησης σε ΚΠΚ. Κάθε αναζήτηση ολοκληρώνεται μέσω πολλαπλών πράξεων εντοπισμού στο επίπεδο του ΚΠΚ, οπότε και το κόστος της σχετίζεται με τη τελική μορφή και πλήθος επιπέδων του ευρετηρίου. Τα PHT έχουν χρησιμοποιηθεί και στα πλαίσια μιας εφαρμογής που λειτουργεί σε διδιάστατο χώρο [CRR+05]. Σε αυτή τη περίπτωση, για την πλήρωση του PHT χρησιμοποιείται μια τεχνική σειριοποίησης, βασισμένη στη καμπύλη Z. Το Distributed Lexical Placement Table (DLPT) [CDT07] είναι μια αντίστοιχη δομή, που παράγεται δυναμικά, για τις ανάγκες δεικτοδότησης και εύρεσης υπηρεσιών στο Πλέγμα. Ένα ευρετήριο trie, ως μια σημασιολογική επικάλυψη επί ενός ΚΠΚ, χρησιμοποιείται και από τη πλατφόρμα IMAGINE-P2P [ZSL+05]. Οι αναζητήσεις προωθούνται μέσω

του ευρετηρίου, το οποίο περιέχει τις συσχετίσεις των αντικειμένων που αποθηκεύονται σε ένα δίκτυο Chord.

Σε όλες τις παραπάνω προσεγγίσεις, το υπόστρωμα του ΚΠΚ εξυπηρετεί το ρόλο μιας αποδοτικής και κλιμακούμενης υποδομής που παρέχει πολύ βασικές λειτουργίες. Η διαφοροποίηση που δημιουργείται μεταξύ των δύο στρωμάτων, εύρεσης και δρομολόγησης, χρησιμοποιείται πολλές φορές και σαν επιχείρημα σχετικά με τη γενικότητα των αλγορίθμων, ή αναφέρεται σαν συγκριτικό πλεονέκτημα, αφού με αυτό το τρόπο διευκολύνεται τόσο η υλοποίηση όσο και η μεταφορά από την μία υποδομή στην άλλη. Από την άλλη μεριά όμως, οι επιπλέον δομές αυξάνουν την πολυπλοκότητα και απαιτούν πρόσθετους μηχανισμούς διαχείρισης. Επίσης, πρέπει να υλοποιούνται σύνθετες λειτουργίες συγχρονισμού των δομών των δύο επιπέδων. Η πιο κοντινή λύση σε αυτήν που προτείνεται στα πλαίσια της διατριβής, είναι η τοποθέτηση ενός PHT εντός μιας επικάλυψης P-Grid [DHJ+05], μέσω της κατάλληλης αναδιοργάνωσης των πινάκων δρομολόγησης των κόμβων. Η τοπολογία του P-Grid είναι δενδρική και θυμίζει αυτή που δημιουργείται και στα δίκτυα Kademlia. Ο ακολουθιακός αλγόριθμος που χρησιμοποιείται για την εκτέλεση αναζητήσεων πλησιάζει τον προτεινόμενο για την μονοδιάστατη περίπτωση. Για την κλιμάκωση σε πολλαπλές διαστάσεις, οι συγγραφείς προτείνουν τη χρήση ενός P-Grid για κάθε αναγνωριστικό. Αυτή η τακτική, όμως – παρόλο που αναφέρεται και σε άλλες εργασίες, προϋποθέτει πολλαπλάσιο κόστος για κάθε αναζήτηση, καθώς και ένα επιπλέον βήμα φιλτραρίσματος διπλών αποτελεσμάτων. Η αναζήτηση σε κάθε διάσταση δεν μπορεί να απαιτεί την επανάληψη του ίδιου αλγορίθμου, κάθε φορά για άλλη, ανεξάρτητη επικάλυψη.

Σύμφωνα με κάποιες άλλες προτάσεις, όπου αποφεύγεται η χρήση ενός διαδεδομένου πρωτοκόλλου ΚΠΚ, οι τιμές κατανέμονται σε έναν δομημένο σχηματισμό κόμβων με εναλλακτικό τρόπο, ο οποίος όμως διατηρεί την ταξινόμησή τους. Το Skip Graph [AS07], για παράδειγμα, είναι μια κατανεμημένη δομή που βασίζεται σε ένα skip list [Pug90] – μια ομάδα από όλο και πιο αραιές διπλά-συνδεδεμένες, ταξινομημένες λίστες κλειδιών. Όλοι οι κόμβοι συμμετέχουν στο κατώτερο επίπεδο, ενώ οι καταχωρήσεις των ανώτερων επιπέδων χρησιμοποιούνται σαν δείκτες για την επικοινωνία με μακρινότερους κόμβους. Καθώς τα skip list δεν είναι ανεκτικά σε σφάλματα και δεν αποθηκεύουν πολλαπλά αντίγραφα της πληροφορίας, ένα Skip Graph ενσωματώνει πολλά skip list σε κάθε επίπεδο της δομής. Τα Skip Tree Graphs [GBMS08] είναι μια αντίστοιχη αρχιτεκτονική, που χρησιμοποιεί skip trees και επιτρέπει την εκτέλεση συγκεντρωτικών ερωτημάτων, καθώς και λειτουργίες multicast/broadcast. Το Mercury [BAS04] είναι ένα σύστημα που αποτελείται από πολλαπλές κυκλικές επικαλύψεις. Κάθε μία αντιστοιχεί και σε ένα αναγνωριστικό και κάθε κόμβος είναι υπεύθυνος για ένα συγκεκριμένο εύρος διευθύνσεων, σε κάθε επικάλυψη. Οι κόμβοι χωρίζονται σε ομάδες που ονομάζονται "attribute hubs" (κέντρα αναγνωριστικών), τα οποία συμβάλλουν στη δρομολόγηση των αιτημάτων. Μια αναζήτηση για παραπάνω από ένα αναγνωριστικό δρομολογείται παράλληλα σε όλα τα σχετικά κέντρα. Οι σχεδιαστές του Mercury δίνουν ιδιαίτερη έμφαση στο πρόβλημα της άνισης κατανομής τιμών στους κόμβους. Το Chord# [SSR08] είναι επίσης μια κυκλική επικάλυψη, στην οποία τα κλειδιά αποθηκεύονται με τη λεξικογραφική τους σειρά. Το SONAR [SSR08] αποτελεί προέκταση του Chord# ώστε να υποστηρίζονται αναζητήσεις σε πολλαπλά αναγνωριστικά. Η τοπολογία του είναι συ-

ναφής του CAN. Ο ορισμός μιας νέας μεθοδολογίας τοποθέτησης κόμβων, απαιτεί την παράλληλη ανάπτυξη αλγορίθμων για την επίτευξη λειτουργιών και την θεμελίωση ιδιοτήτων που ήδη υποστηρίζονται ή διαπιστώνονται στο περιβάλλον ενός ΚΠΚ (λογαριθμική δρομολόγηση, αυτόματη διαχείριση αφίξεων και αναχωρήσεων κόμβων, κτλ.). Αυτό το πρόβλημα εμφανίζεται στις παραπάνω εργασίες, οι οποίες, επιπλέον, δεν αναφέρονται καθόλου σε διαδικασίες αυτόματης αντιγραφής τιμών και διαχείρισης των σχετικών αντιγράφων. Οι τεχνικές σειριακής ταξινόμησης και κατανομής των δεδομένων δεν υποστηρίζουν διπλές εγγραφές. Συνήθως, για την αντιμετώπιση προβλημάτων που μπορεί να προκύπτουν στο επίπεδο αποθήκευσης λόγω της δυναμικής φύσης του δικτύου, προτείνεται η προσθήκη ενός ακόμα επιπέδου που θα αναλαμβάνει την τήρηση πολλαπλών εμφανίσεων της ίδιας τιμής.

Για την υποστήριξη πράξεων που αναφέρονται ταυτόχρονα σε πολλαπλά αναγνωριστικά, προτείνεται η κατασκευή αντίστοιχων κλειδιών. Θεωρείται ότι όλα τα αναγνωριστικά μιας τιμής ορίζουν έναν πολυδιάστατο χώρο, ο οποίος μέσω μιας ΚΠΧ μπορεί να μετασχηματιστεί σε μια σειριακή απεικόνιση και αντίστροφα. Οι ΚΠΧ μπορούν να παράγουν τέτοιες απεικονίσεις, που επιπλέον, ως ένα βαθμό διατηρούν την εγγύτητα των αναγνωριστικών από τον ένα χώρο στον άλλο. Για το λόγο αυτό, έχουν χρησιμοποιηθεί εκτενώς σε συστήματα διαχείρισης βάσεων δεδομένων [Bay97, TH81, LK00], επεξεργασίας εικόνας, κτλ. Μια ανάλυση [MJFS01] δείχνει ότι ο μετασχηματισμός που ορίζεται μέσω της καμπύλης Hilbert έχει καλύτερα χαρακτηριστικά ομαδοποίησης τιμών, δηλαδή διατηρεί καλύτερα τις συγκεντρώσεις ομοειδών δεδομένων κατά τη μετατροπή. Σε δομημένα ΔΟΚ, η συνάρτηση κατακερματισμού μπορεί εύκολα να αντικατασταθεί με μια αντιστοίχιση βασισμένη σε κάποια ΚΠΧ. Το Squid [SP08] είναι μια επικάλυψη τύπου Chord, που επιτρέπει με αυτόν τον τρόπο την αναζήτηση εύρους διευθύνσεων. Τα κλειδιά των δεδομένων παράγονται από μια συνάρτηση που μετατρέπει τα χαρακτηριστικά τους σε ένα σημείο επί της καμπύλης Hilbert. Ένα ερώτημα ολοκληρώνεται μέσω πολλαπλών πράξεων εντοπισμού στο επίπεδο του ΚΠΚ, με την αναδρομική αναζήτηση των αντίστοιχων τμημάτων της ΚΠΧ. Σε κάθε βήμα, μεγαλώνει το πρόθεμα του τμήματος της ΚΠΧ, οπότε γίνεται και πιο συγκεκριμένο το εύρος. Αν ερωτηθεί ένας κόμβος που είναι υπεύθυνος για το συγκεκριμένο τμήμα της καμπύλης, απαντά με τις τιμές που αποθηκεύει. Αλλιώς, παράγει νέα ερωτήματα για άλλους κόμβους που αναφέρονται σε μικρότερα διαστήματα. Πρακτικά με αυτόν τον τρόπο, κατανέμεται ο υπολογιστικός φόρτος για τον μετασχηματισμό στους κόμβους. Όμως, ο φόρτος αυτός παραμένει μεγάλος και συγκεντρώνεται στους κόμβους που τυχαίνει να είναι υπεύθυνοι για μικρότερα προθέματα – καθώς όλα τα ερωτήματα αρχικά προωθούνται σε αυτούς. Πολυδιάστατη αναζήτηση μέσω μιας ΚΠΧ Hilbert, πραγματοποιείται και από το CISS [LLK+07]. Υποθέτοντας μια ιεραρχία για κάθε αναγνωριστικό, οι σχετικές πράξεις ξεκινούν από τον κόμβο που είναι υπεύθυνος για το πρώτο υποψήφιο τμήμα της καμπύλης και προωθούνται με τη σειρά στους επόμενους κόμβους. Η κατανομή του φόρτου γίνεται είτε σε τοπικό επίπεδο, ανακατανέμοντας τα δεδομένα μεταξύ κοντινών κόμβων, είτε καθολικά στο δίκτυο, με τον εντοπισμό κόμβων στους οποίους καταλήγουν ελάχιστα ερωτήματα. Το SCRAP [GYGM04], προσφέρει ανάλογη λειτουργικότητα πολυδιάστατης αναζήτησης, χρησιμοποιώντας και πάλι μια ΚΠΧ για τη παραγωγή των μονοδιάστατων κλειδιών των δεδομένων. Για την εκτέλεση ενός ερωτήματος, υπολογίζονται τα τμήματα

της καμπύλης που αντιστοιχούν στην αναζήτηση, τα οποία με τη σειρά τους δρομολογούνται σε ένα δίκτυο Skip Graph. Ωστόσο, ο αριθμός των υπολογιζόμενων τμημάτων μπορεί να είναι πολύ μεγάλος και εξαρτάται από την τάξη της χρησιμοποιούμενης καμπύλης – και άρα τον αριθμό των δυαδικών ψηφίων που μπορούν να χρησιμοποιηθούν για την αναπαράσταση κάθε αναγνωριστικού. Στον αλγόριθμο που παρουσιάζουμε, αποφεύγεται εξ ολοκλήρου η επί τόπου μετάφραση του διαστήματος αναζήτησης σε τμήματα της ΚΠΧ, προκειμένου η λύση που προτείνεται να είναι ρεαλιστική, χωρίς περιορισμούς στη κλιμάκωση του επιπέδου λεπτομέρειας της καμπύλης και του συνολικού συστήματος.

Μερικά συστήματα υποστηρίζουν πολυδιάστατες αναζητήσεις μέσω του δικού τους σχήματος οργάνωσης των δεδομένων, αλλά και των κόμβων. Το MURK [GYGM04] υλοποιεί ένα κατανεμημένο kd-tree προκειμένου να διαχωρίσει τον χώρο δεδομένων σε ορθογώνια, τα οποία και αναθέτει στους συμμετέχοντες. Κάθε κόμβος γνωρίζει τους κοντινούς του, οπότε και προωθεί τα αιτήματα που δέχεται ανάλογα. Για την δεικτοδότηση και αναζήτηση χωρικών δεδομένων έχει προταθεί και η λύση ενός κατανεμημένου quadtree [THS06], όπου το κέντρο κάθε κομματιού περνάει από μια συνάρτηση κατακερματισμού και τοποθετείται σε μια επικάλυψη Chord. Το ZNet [SOTZ05] διαχωρίζει δυναμικά τον χώρο σύμφωνα με μια καμπύλη Z και χρησιμοποιεί Skip Graphs για την οργάνωση των κόμβων. Ο κάθε συμμετέχων μπορεί να είναι υπεύθυνος για διάφορα συνεχόμενα διαστήματα της καμπύλης, ανάλογα με τη θέση του. Μια αναζήτηση ξεκινά από τους κόμβους στους οποίους έχουν ανατεθεί τα πιο μικρά προθέματα, όπου υπολογίζονται πιο συγκεκριμένα τμήματα, κτλ. Το Armanda [LCLC09], χρησιμοποιεί το FissionE [LLW05], μια επικάλυψη δομημένη σύμφωνα με τα χαρακτηριστικά ενός γράφου Kautz. Υποστηρίζει πολυδιάστατες αναζητήσεις, μέσω αλγορίθμων που αντιστοιχούν πολυδιάστατα διανύσματα τιμών σε μια ταξινόμηση. Έτσι, αναθέτονται τα δεδομένα σε κόμβους, με τρόπο που διασφαλίζει ότι κάθε κόμβος είναι υπεύθυνος για ένα συγκεκριμένο διάστημα κοντινών τιμών ενός αναγνωριστικού. Και πάλι, αυτές οι προσεγγίσεις δεν προσφέρουν λειτουργίες ήδη διαθέσιμες από έναν ΚΠΚ ή απαιτούν πολύπλοκες διαδικασίες συγχρονισμού δομών δεδομένων στα διάφορα επίπεδα που ορίζουν.

4.3 Υλοποίηση

Στις επόμενες παραγράφους θα περιγραφεί το προτεινόμενο πρωτόκολλο αναζήτησης ομάδων δεδομένων στα πλαίσια του σχήματος δρομολόγησης που χρησιμοποιεί ο ΚΠΚ Kademlia. Αρχικά, θα δείξουμε την διαδικασία αναζήτησης ομάδων κλειδιών (μονοδιάστατη περίπτωση) και πως αυτή μπορεί να απεικονιστεί σε μια σειρά ειδικών εντολών δρομολόγησης και εντοπισμού στο επίπεδο της επικάλυψης που αποφεύγουν – όσο αυτό είναι δυνατόν – τα πολλαπλά αντίγραφα που αποθηκεύονται στο δίκτυο. Το αποτέλεσμα της αναζήτησης είναι το σύνολο των τιμών που εμπίπτουν στο ζητούμενο εύρος διευθύνσεων. Έπειτα, επεκτείνεται ο αλγόριθμος και ορίζεται ένα ενοποιημένο πρωτόκολλο που μπορεί να διαχειριστεί τόσο μονοδιάστατα όσο και πολυδιάστατα διανύσματα αναζήτησης. Στη πολυδιάστατη περίπτωση, ζητείται ένα εύρος τιμών για κάθε χαρακτηριστικό των δεδομένων.

Αν και δείχνεται πως η έννοια της *γειτονιάς* μπορεί να γενικευθεί και σε ΚΠΚ που χρησιμοποιούν κυκλικούς χώρους διευθύνσεων, επιλέξαμε και πάλι το παράδειγμα του Kademlia, λόγω της συμμετρίας που παρουσιάζει στη σχέση κόμβων και αντιγράφων. Στο Kademlia χρησιμοποιείται μόνον ένας πίνακας δρομολόγησης για όλες τις εργασίες, κάτι που το καθιστά απλό και κατανοητό. Μέσω αυτού του πίνακα, μπορούν άμεσα να οριστούν οι γειτονιές των κόμβων, καθώς και να εξαχθούν οι γείτονες κάθε συμμετέχοντα. Επίσης, η συνάρτηση εγγύτητας που εφαρμόζεται – το αποκλειστικό ή (XOR) – δημιουργεί αμφίδρομες σχέσεις μεταξύ των κόμβων, οπότε και απλουστεύεται η διαχείριση των σκελών στα οποία θεωρούμε ότι χωρίζεται η επικάλυψη κατά την εκτέλεση των πράξεων εντοπισμού. Το Kademlia τηρεί αυτόματα αντίγραφα των δεδομένων που δίνονται προς αποθήκευση.

Αντίστοιχα, για την μετάφραση των διανυσμάτων χαρακτηριστικών σε διευθύνσεις δικτύου χρησιμοποιούμε την καμπύλη Z, ως απλούστερη και πιο διαδεδομένη στη χρήση. Οποιαδήποτε καμπύλη με τα ίδια χαρακτηριστικά μπορεί να εξυπηρετήσει τις ανάγκες του συστήματος.

4.3.1 Αναζήτηση εύρους διευθύνσεων

Για την ανάκτηση μιας τιμής από έναν ΚΠΚ απαιτείται καταρχάς η διεύθυνση ή το κλειδί που της αντιστοιχεί. Από τον ίδιο χώρο διευθύνσεων, όμως, αντλούν τα αναγνωριστικά τους και οι ίδιοι οι κόμβοι. Όλοι οι συμμετέχοντες διαθέτουν έναν πίνακα δρομολόγησης με συγκεκριμένο αριθμό επαφών – περισσότερες στις κοντινές τους περιοχές και λιγότερες όσο μεγαλώνει η απόσταση στον εικονικό χώρο που ορίζεται από το πρωτόκολλο, τους οποίους και χρησιμοποιούν για να προωθούνται τα αιτήματα όσο γίνεται πιο κοντά στο ζητούμενο κλειδί. Ο κόμβος με την πλησιέστερη διεύθυνση πρέπει να αποθηκεύει το ζεύγος κλειδιού-τιμής, αλλιώς θεωρείται ότι το δεδομένο αυτό δεν υπάρχει.

Σε αυτό το πλαίσιο, μια "αφελής" προσέγγιση στην υλοποίηση ενός αλγορίθμου ανάκτησης ενός εύρους διευθύνσεων, θα περιελάμβανε την εκτέλεση απλών εντολών εντοπισμού όλων των πιθανών κλειδιών του διαστήματος $[Q_{start}, Q_{end}]$ και στη συνέχεια την αντιμετώπιση των σοβαρών προβλημάτων απόδοσης που θα προέκυπταν. Εκτός αν το $[Q_{start}, Q_{end}]$ αναφέρεται σε ένα εξαιρετικά μικρό διάστημα, δεδομένης της αραιής κατανομής των δεδομένων σε έναν ΚΠΚ – κυρίως λόγω του τεράστιου μεγέθους του εικονικού χώρου, ένας μεγάλος αριθμός κλήσεων εντοπισμού θα πάει χαμένος. Επιπρόσθετα, επειδή θα γίνονται πράξεις σε κοντινά κλειδιά, πολλές από τις κλήσεις αυτές πιθανότατα θα καταλήγουν συνεχώς στο ίδιο υποσύνολο κόμβων. Ήδη φαίνεται ότι κάποια τακτική ομαδοποίησης των κόμβων και δρομολόγησης των αιτημάτων ανά υποδιαστήματα στις αντίστοιχες ομάδες συμμετεχόντων θα βοηθούσε στην αποφυγή ενός σημαντικού πλήθους περίσσιων, περιττών μηνυμάτων. Το κλειδί στη λύση του προβλήματος βρίσκεται στην χρήση μιας θεμελιώδους διαφορετικής μεθοδολογίας στην εκτέλεση αιτημάτων αναζήτησης, που θέλει τους κόμβους να συνεργάζονται σε τοπικό επίπεδο προκειμένου να εκμεταλλευτούν το σύνολο της πληροφορίας που έχουν σαν σύνολο, λαμβάνοντας, βέβαια, υπόψη και τη συνολική τους δυναμική εικόνα σε κάθε βήμα της διαδικασίας. Ιδανικά, οι αναζητήσεις αυτού του τύπου θα πρέπει να προσφέρονται εγγενώς από τον ίδιο τον ΚΠΚ με έναν τρόπο που αφενός σέβεται και

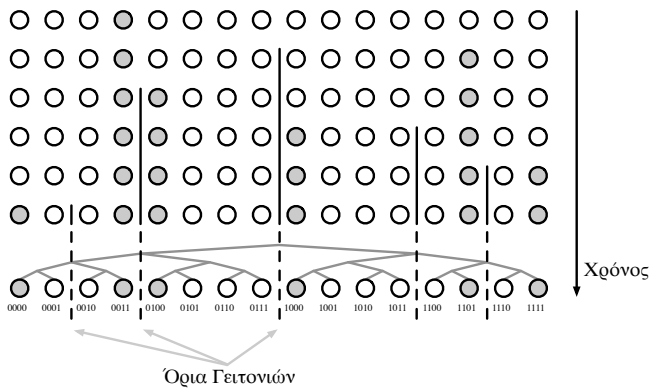
αφετέρου διατηρεί τους μηχανισμούς κλιμάκωσης και ανοχής σε σφάλματα που χαρακτηρίζουν τα δομημένα ΔΟΚ και τα έχουν κάνει τόσο δημοφιλή ως μια γενική πλατφόρμα υποστήριξης κατανεμημένων εφαρμογών.

Σε έναν ΚΠΚ, κάθε κόμβος μπορεί να αποθηκεύει πολλά από τα κλειδιά ενός διαστήματος, οπότε δεν υπάρχει λόγος να εκτελεστούν όλες οι ενδιάμεσες εντολές εντοπισμού. Η λειτουργία της αναζήτησης, στηριζόμενη σε αυτό το γεγονός, μπορεί να προχωρά από τον ένα κόμβο στον επόμενο, λαμβάνοντας υπόψη μόνο τα σχετικά όρια. Επιπλέον, σε έναν ΚΠΚ που τηρούνται αντίγραφα των τιμών σε ένα εύρος κοντινών στην αντίστοιχη διεύθυνση κόμβων, η διαδικασία θα πρέπει αυτόματα να παραλείπει σε κάθε βήμα το κατάλληλο πλήθος συμμετεχόντων, που έτσι και αλλιώς θα απαντούσαν με τα ίδια δεδομένα. Στο Kademlia το εύρος των κόμβων που τηρούν αντίγραφα του ίδιου κλειδιού, ελέγχεται από τη παράμετρο κ – κάθε ζεύγος κλειδιού-τιμής αποθηκεύεται στους κ πλησιέστερους κόμβους. Έτσι, ξεκινώντας από τον συμμετέχοντα υπεύθυνο για το Q_{start}, η αναζήτηση πρέπει να μεταπηδήσει στον επόμενο κόμβο που διατηρεί έστω και μια διαφορετική τιμή, κτλ., έως ότου καταλήξει σε εκείνον που αποθηκεύει το Q_{end}. Το μονοπάτι που περνά από όλους τους ενδιάμεσους, εκτός του ότι απαιτεί περισσότερο χρόνο, οδηγεί σε άσκοπη σπατάλη των πόρων δικτύου του συνολικού συστήματος, καθώς θα μεταφερθούν όλα τα αντίγραφα των σχετικών τιμών.

Ο ακριβής αριθμός των κόμβων που θα πρέπει να παραλείπονται σε κάθε βήμα της λειτουργίας αναζήτησης, μπορεί να οριστεί με βάση τη δομή του εικονικού χώρου που φιλοξενεί τους συμμετέχοντες του ΔΟΚ. Χρησιμοποιούμε τους ακόλουθους ορισμούς για τη περίπτωση του δυαδικού δέντρου που εννοείται από το πρωτόκολλο Kademlia: Δεδομένου ενός κλειδιού, ονομάζουμε *υπο-γειτονιά* (sub-neighborhood) κάθε υποδέντρο που το περιέχει, όσο στο υποδέντρο αυτό το πλήθος των κόμβων είναι κάτω από κ. Η μεγαλύτερη υπο-γειτονιά, ορίζεται ως η μοναδική *γειτονιά* (neighborhood) του κλειδιού. Στην επικάλυψη που δημιουργεί το Kademlia, όλοι οι κόμβοι της ίδιας γειτονιάς μοιράζονται τουλάχιστον όλες τις τιμές των οποίων τα κλειδιά ανήκουν στο διάστημα διευθύνσεων που αντιστοιχεί στα άκρα της. Ή αλλιώς: Όλα τα κλειδιά της ίδιας γειτονιάς υπάρχουν σε αντίγραφα τουλάχιστον σε όλους τους εσωτερικούς της κόμβους. Αν ονομάσουμε $R_N = [N_{start}, N_{end}]$ τα όρια της γειτονιάς, μεταξύ των κλειδιών και των κόμβων στο R_N διατηρείται μια αμφίδρομη σχέση.

Βέβαια, θα πρέπει να σημειωθεί ότι καθώς ο πληθυσμός κάθε γειτονιάς είναι πάντα μικρότερος από την τιμή της παραμέτρου κ, επιπλέον αντίγραφα των *εσωτερικών* τιμών μπορεί να αποθηκεύονται και σε άλλους κόμβους – που ανήκουν σε παρακείμενα σκέλη της επικάλυψης. Η δυναμική φύση του δικτύου δεν επιτρέπει την παρακολούθηση όλων των πιθανών θέσεων που μπορεί να αντιγράφεται κάθε δεδομένο. Παρόλα αυτά κάθε κόμβος γνωρίζει ποιες τιμές που αποθηκεύει ανήκουν στη δική του γειτονιά και ποιες όχι, ώστε να μπορεί να απαντά σωστά. Αν μια αναζήτηση τύχει να πέσει σε κάποιον κόμβο που για οποιονδήποτε λόγο έχει λανθασμένες γνώσεις για τα όρια της περιοχής του, οι διπλές απαντήσεις φιλτράρονται και απορρίπτονται. Κάτι τέτοιο αναμένεται όμως να συμβαίνει σπάνια και μόνο όταν το δίκτυο υπόκειται σε συνεχείς αλλαγές.

Οι γειτονιές στο Kademlia είναι εξ ορισμού διακριτές, μη επικαλυπτόμενες ομάδες κόμβων που

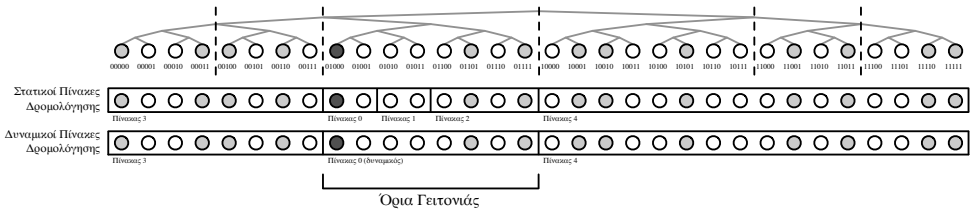


Σχήμα 4.1: *Παράδειγμα δημιουργίας γειτονιών για $\kappa = 2$. Ο χώρος διευθύνσεων αναπαριστάται οριζόντια, ως άδειες θέσεις, που σταδιακά γεμίζουν με νέους κόμβους (με γκρι χρώμα).*

μοιράζονται τις θέσεις των φύλλων του ίδιου υποδέντρου. Σε μια επικάλυψη που έχει λιγότερους από κ συμμετέχοντες, αυτό το υποδέντρο θα συμπίπτει με το σύνολο του χώρου διευθύνσεων, οπότε και όλοι οι κόμβοι θα είναι γείτονες. Όσο αυξάνεται ο πληθυσμός του δικτύου και αμέσως μόλις φτάσει το κ, η μοναδική γειτονιά θα χωριστεί σε δύο: μία που θα αντιστοιχεί στο αριστερό πρώτο υποδέντρο και μία στο δεξί. Η διαδικασία επαναλαμβάνεται κάθε φορά που ο αριθμός των κόμβων σε μια γειτονιά φτάσει το όριο που τίθεται από την παράμετρο που ορίζει το πλήθος των αντιγράφων. Ένα απλό παράδειγμα για $\kappa = 2$ φαίνεται στο σχήμα 4.1. Καθώς σταδιακά αυξάνεται το μέγεθος τους δικτύου, δημιουργείται μια νέα γειτονιά για κάθε καινούριο κόμβο.

Ο ορισμός της *γειτονιάς* ως εργαλείο είναι πολύ σημαντικός όσον αφορά την υλοποίηση λειτουργιών αναζήτησης, αφού έτσι δημιουργείται μια σχέση μεταξύ των καταχωρήσεων των πινάκων δρομολόγησης με τις θέσεις των αντιγράφων των δεδομένων. Από την οπτική γωνία του κάθε κόμβου, εξυπηρετεί ως ένας δείκτης στο πόσα βήματα δρομολόγησης μακριά βρίσκεται ο επόμενος συμμετέχοντας που διατηρεί έστω και μια διαφορετική τιμή. Για τις λειτουργίες αναζήτησης, αρκεί, λοιπόν, να ερωτηθεί *οποιοσδήποτε* κόμβος σε μια γειτονιά. Αυτός θα μπορεί να απαντήσει για όλα τα αντίγραφα που περιέχονται στο αντίστοιχο διάστημα διευθύνσεων. Αυτή είναι και η βασική ιδιότητα που προσφέρεται από την ομαδοποίηση των κόμβων και θα πρέπει να διατηρείται όταν ορίζεται η έννοια της γειτονιάς σε άλλα δομημένα ΔΟΚ, που χρησιμοποιούν άλλου τύπου εικονικό χώρο και άλλους μηχανισμούς αυτόματης τήρησης αντιγράφων.

Σε επικαλύψεις που χρησιμοποιούν κυκλικούς σχηματισμούς, όπου κάθε ζεύγος κλειδιού-τιμής αντιγράφεται αυτόματα στους f *ακόλουθους* κόμβους – ως μια παραλλαγή του Chord που τοποθετεί κάθε τιμή στον κόμβο με την αμέσως επόμενη διεύθυνση, οι γειτονιές αντιγράφων επικαλύπτονται. Παρόλα αυτά, για ένα εύρος διευθύνσεων R_N, που περιέχει $\leq f$ συμμετέχοντες, όλοι οι κόμβοι του διαστήματος $[R_{mid}, R_{end}]$, $R_{start} \leq R_{mid} \leq R_{end}$ μοιράζονται όλες τις τιμές με κλειδιά που ανήκουν στο $[R_{start}, R_{mid}]$. Μετά την επικοινωνία με έναν κόμβο στο $[R_{mid}, R_{end}]$, η αναζήτηση μπορεί να συνεχίσει με έναν από τους f προηγούμενούς του με διεύθυνση $< R_{start}$. Αν τα αντίγραφα τοποθετούνται στους $\frac{f}{2}$ ακόλουθους και $\frac{f}{2}$ προηγούμενους της αντίστοιχης διεύθυνσης [RD01], όλα τα διαστήματα που περιέχουν $\leq f/2$ κόμβους είναι γειτονιές. Υπάρχουν, όμως, ακόμα επικαλύψεις, οπότε για να αποφευχθούν οι διπλές απαντήσεις θα πρέπει ο

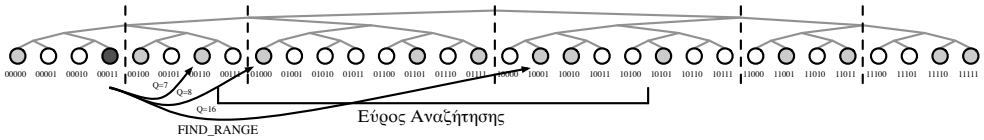


Σχήμα 4.2: *Υλοποιήσεις πινάκων δρομολόγησης για κόμβο (με σκούρο γκρι) που ανήκει σε δίκτυο Kademlia με $\kappa = 4$. Ο πρώτος πίνακας στην δυναμικά διαχειριζόμενη περίπτωση περιέχει όλους του γείτονες του κόμβου και ορίζει τα όρια της γειτονιάς.*

αλγόριθμος να προχωρά κατά f συνεχόμενους κόμβους σε κάθε βήμα.

Στο Kademlia, κάθε κόμβος γνωρίζει καλά τους γείτονές του, καθώς διατηρεί πολύ περισσότερη πληροφορία δρομολόγησης για τις κοντινές του διευθύνσεις. Στο Kademlia, οι επαφές οργανώνονται σε 160 πίνακες σταθερής χωρητικότητας (ίση με κ πάλι), όπου κάθε ένας αντιπροσωπεύει και ένα εκθετικά μεγαλύτερο τμήμα του συνολικού χώρου των 2^{160} διευθύνσεων – μεγαλύτερο όσο αυξάνεται η απόσταση από τον αντίστοιχο συμμετέχοντα. Πρακτικά, κάθε κόμβος διατηρεί κ δείκτες για τη μισή δομή που δεν τον περιλαμβάνει (το αριστερό ή δεξί πρώτο υποδέντρο), κ για τον μισό εναπομείναντα χώρο που και πάλι δεν τον περιλαμβάνει (ένα δέντρο δεύτερου επιπέδου), κτλ. Έτσι, μπορεί να βρει τη γειτονιά του ξεκινώντας από τα περιεχόμενα του πίνακα που αντιστοιχεί στη κοντινότερη σε αυτόν περιοχή και προχωρώντας με τη σειρά, όσο το συνολικό πλήθος επαφών παραμένει εντός ορίων.

Στη σχετική βιβλιογραφία, οι στατικοί πίνακες δρομολόγησης χρησιμοποιούνται από τους σχεδιαστές του πρωτοκόλλου για να αποσαφηνιστεί η δομή και ο τρόπος λειτουργίας της επικάλυψης, ενώ παράλληλα προτείνεται και μια υλοποίηση *δυναμικά διαχειριζόμενων* πινάκων. Όταν οι πληροφορίες δρομολόγησης τηρούνται με αυτόν τον τρόπο δεν δημιουργούνται εξαρχής και οι 160 πίνακες, αλλά κάθε κόμβος ξεκινά με έναν δυναμικό πίνακα που περιέχει τις κοντινότερες επαφές του και προσθέτει νέους πίνακες μόνο όταν χρειαστεί. Ο μηχανισμός παραγωγής νέων πινάκων θυμίζει τη διαδικασία με την οποία δημιουργούνται οι γειτονιές κόμβων στο δίκτυο, όπως αυτή περιγράφηκε παραπάνω. Κάθε φορά που το πλήθος των επαφών στον δυναμικό πίνακα φτάνει το κ, παράγεται ένας νέος, στατικός πίνακας στον οποίο τοποθετούνται οι επαφές που ανήκουν στον μισό χώρο που δεν περιλαμβάνει τον κόμβο. Οι νέες επαφές προστίθενται στον αντίστοιχο πίνακα, στατικό ή δυναμικό, ανάλογα με το σε ποιά όρια ανήκει το αναγνωριστικό τους. Οι στατικοί πίνακες έχουν συγκεκριμένη χωρητικότητα και η διαχείρισή τους όταν γεμίσουν γίνεται με κάποιο σχετικό αλγόριθμο – π.χ. παραμένουν οι παλαιότεροι κόμβοι (LIFO) ή εναλλακτικά παραμένουν οι νεότερες επαφές (FIFO). Ο δυναμικός πίνακας, όταν γεμίσει με επαφές, επαναλαμβάνεται η διαδικασία παραγωγής ενός νέου στατικού πίνακα, έως ότου το μέγεθός του είναι μικρότερο από κ. Στη πράξη, λοιπόν, το όριο μεταξύ του δυναμικού πίνακα και του πρώτου στατικού είναι κάθε φορά το σημείο στο οποίο τελειώνει η γειτονιά του αντίστοιχου κόμβου. Διαχωρίζει το δίκτυο στο υποδέντρο που περιέχει τον ίδιο τον κόμβο και λιγότερους από κ κοντινούς του και όλα τα υπόλοιπα υποδέντρα. Όταν τηρούνται δυναμικά οι πίνακες δρομολόγησης, πληροφορίες σχετικά με τα όρια της γειτονιάς και τους γείτονες ενός κόμβου είναι



Σχήμα 4.3: *Η επιτυχής ολοκλήρωση μιας αναζήτησης προϋποθέτει μια κλήση* `FIND_RANGE` *ανά γειτονιά του συνολικού εύρους διευθύνσεων.*

άμεσα προσβάσιμες και δίνονται από τα όρια και περιεχόμενα του δυναμικού πίνακα αντίστοιχα (σχήμα 4.2).

Στην υλοποίησή μας, έχουμε αντικαταστήσει το ζεύγος μηνυμάτων `FIND_VALUE`/`VALUE` του Kademlia με το `FIND_RANGE`/`RANGE`. Ένα ερώτημα `FIND_RANGE` περιέχει τα όρια των διευθύνσεων που ενδιαφέρουν τον κόμβο που κάνει το αίτημα, ενώ μια απάντηση RANGE περιέχει τόσο τις τιμές που υπάρχουν στο συγκεκριμένο διάστημα, όσο και πληροφορία σχετικά με τη γειτονιά του κόμβου – προσδιορίζει ποιο τμήμα του χώρου διευθύνσεων *καλύπτεται* από τον κόμβο που απαντά. Έτσι, μια αναζήτηση εύρους διευθύνσεων ακολουθεί τα παρακάτω βήματα:

1. Ξεκίνα με το $Q = Q_{start}$.
2. Βρες τους κοντινότερους κόμβους στο Q.
3. Στείλε ένα μήνυμα `FIND_RANGE` στον πρώτο κόμβο της λίστας.
4. Συγκέντρωσε τις τιμές που περιέχει το μήνυμα `RANGE` και πρόσθεσε την επιστρεφόμενη κάλυψη στο συνολικό εύρος διευθύνσεων που έχουν καλυφθεί.
5. Ανανέωσε το Q ώστε να ισούται με την επόμενη διεύθυνση εκτός κάλυψης.
6. Αν το Q είναι εντός του ζητούμενου εύρους, επανέλαβε τη διαδικασία από το δεύτερο βήμα.

Η όλη διαδικασία συντονίζεται από την αρχή έως το τέλος από έναν κόμβο. Ένας – αυτός στον οποίο έγινε το αρχικό αίτημα από την εφαρμογή – κάνει τις απαραίτητες κλήσεις `FIND_NODE` και `FIND_RANGE` και συλλέγει όλα τα αποτελέσματα. Έτσι λειτουργούν συνήθως οι αλγόριθμοι σε ΚΠΚ, και όχι αναδρομικά, καθώς είναι ένας τρόπος να διασφαλιστεί ότι ο κόμβος με το αντίστοιχο κίνητρο, αναλαμβάνει και την ολοκλήρωση της διαδικασίας. Επίσης, έτσι είναι πολύ πιο απλή η αποφυγή σφαλμάτων και η διαχείριση των όποιων προβλημάτων που μπορεί να προκύψουν κατά τη διάρκεια της εκτέλεσης του αλγορίθμου. Αν, για οποιοδήποτε Q, ένας κόμβος δεν απαντήσει, η διαδικασία μπορεί να συνεχιστεί με κάποιον άλλο κόμβο της ίδιας γειτονιάς. Στο παράδειγμα του σχήματος 4.3, όπου το $\kappa = 4$, ο συμμετέχων με τον αριθμό 3 χρειάζεται να ολοκληρώσει επιτυχώς τρεις κλήσεις `FIND_RANGE` για την αναζήτηση στο διάστημα $[7, 20]$. Αρχικά το Q είναι 7, που αντιστοιχεί στην γειτονιά $[4, 7]$. Υποθέτοντας ότι ο κόμβος 6 απαντά με ένα μήνυμα `RANGE`, το Q γίνεται 8, που ανήκει στη γειτονιά $[8, 15]$. Οι πλησιέστεροι κόμβοι στο 8 είναι οι $8, 13, 15$ και 0. Αν ο 8 δεν απαντήσει εμπρόθεσμα, ο 3 θα επιλέξει τον επόμενο

στη λίστα, δηλαδή τον 13, ο οποίος και θα πρέπει να απαντήσει με τα ίδια δεδομένα και την ίδια ακριβώς κάλυψη. Ο αλγόριθμος διατηρεί τα χαρακτηριστικά αυτονομίας και αυτοδιαχείρισης του δικτύου, αποφεύγοντας αυτόματα τον κατάλληλο αριθμό κόμβων σε κάθε βήμα, σύμφωνα με τις παραμέτρους λειτουργίας και τη τρέχουσα κατάσταση της επικάλυψης.

Μάλιστα, προκειμένου να ταιριάζουν απόλυτα τα χαρακτηριστικά της διαδικασίας αναζήτησης εύρους διευθύνσεων με αυτά της αντίστοιχης περίπτωσης όπου εντοπίζονται και ανακτώνται τα περιεχόμενα μιας σημειακής διεύθυνσης, ορίζουμε ένα πρόσθετο, αλλά προαιρετικό βήμα στον αλγόριθμο: Με τη παραλαβή ενός μηνύματος `RANGE`, ο κόμβος που διαχειρίζεται την αναζήτηση μπορεί να στείλει παράλληλες αιτήσεις `FIND_MORE` και σε όλους τους υπόλοιπους συμμετέχοντες της ίδιας γειτονιάς, προκειμένου να ενημερωθεί για τυχόν επιπλέον τιμές που μπορεί να έχουν "ξεφύγει" από άλλους κόμβους. Τα μηνύματα τύπου `FIND_MORE` περιέχουν μια λίστα με τα αναγνωριστικά των τιμών που έχουν ήδη βρεθεί. Σε πολύ δυναμικά δίκτυα, λόγω των συνεχώς μεταβαλλόμενων σχέσεων μεταξύ κόμβων και τιμών, το πρωτόκολλο Kademlia απαιτεί την επικοινωνία με όλους τους κοντινούς κόμβους ενός κλειδιού σε μια πράξη εντοπισμού. Έτσι, για παράδειγμα, καταλήγουμε ότι ένα ζεύγος κλειδιού-τιμής δεν υπάρχει στο δίκτυο όταν και οι κ πλησιέστεροι κόμβοι έχουν απαντήσει με μηνύματα τύπου `NODE`. Σε αυτό αποσκοπούν και τα μηνύματα `FIND_MORE`, αν και είναι προαιρετικά, καθώς η χρησιμότητα της επιπλέον επικοινωνίας που απαιτούν εξαρτάται από το περιβάλλον εκτέλεσης και τις απαιτήσεις της εφαρμογής. Μπορούν να αποφευχθούν αν είναι γνωστό εκ των προτέρων ότι στον ΚΠΚ δεν θα παρουσιάζονται ακραίοι ρυθμοί ανανέωσης κόμβων ή αν η εφαρμογή – ακόμα και σε αυτές τις περιπτώσεις – μπορεί να λειτουργήσει με ένα υποσύνολο των τιμών από ένα εύρος διευθύνσεων.

Επίσης, να σημειωθεί ότι ειδικά στη περίπτωση που τηρούνται δυναμικά οι πληροφορίες δρομολόγησης εντός των κόμβων, ο αλγόριθμος δεν προϋποθέτει επιπλέον δομές ή λειτουργίες διαχείρισης για τον εντοπισμό και τον χειρισμό των ορίων των γειτονιών. Αν η λειτουργία αναζήτησης εύρους διευθύνσεων εφαρμοστεί σε άλλα πρωτόκολλα ΚΠΚ, η προσθήκη ενός κόμβου στην επικάλυψη ή η αποσύνδεσή του, μπορεί να απαιτεί την ενημέρωση ειδικών πινάκων που παρακολουθούν τα μέλη μιας υποπεριοχής του δικτύου. Και σε αυτή τη περίπτωση όμως, αυτοί οι πίνακες θα έχουν μόνο τοπικό χαρακτήρα και θα επηρεάζονται μόνο από αλλαγές που συμβαίνουν σε συγκεκριμένα, κοντινά διαστήματα για κάθε κόμβο.

4.3.2 Κλιμάκωση στον πολυδιάστατο χώρο

Πολυδιάστατες ονομάζουμε τις τιμές που ορίζονται από πολλά χαρακτηριστικά – ένα κλειδί για κάθε διάσταση. Για κάποιες εφαρμογές, η ύπαρξη πολλών χαρακτηριστικών είναι αναγκαία ώστε να διατηρείται η σημασιολογία των δεδομένων. Ένα μεγάλο πλήθος σχετικών εργασιών έχουν αφιερωθεί στην αποδοτική διαχείριση τιμών αυτής της μορφής. Έτσι, υπάρχουν αρκετοί αλγόριθμοι που περιγράφουν τρόπους αποθήκευσης πολυδιάστατων συλλογών και μεθόδους αναζήτησης συγκεκριμένων ή συγγενών τιμών από το σύνολο των δεδομένων. Κύριο χαρακτηριστικό των περισσότερων προτάσεων είναι η χρήση μιας συνάρτησης που επιτρέπει την μετατροπή των πολλαπλών κλειδιών σε ένα μοναδικό νούμερο. Οι συναρτήσεις αυτές καλούνται και Καμπύλες

Πλήρωσης Χώρου, καθώς η λειτουργία τους αναπαρίσταται με μια γραμμή που περνά συνεχόμενα από όλα τα σημεία του πολυδιάστατου χώρου που ορίζεται από το εύρος διευθύνσεων του κάθε κλειδιού. Οι ΚΠΧ διατηρούν τη τοπικότητα των αναφορών σε κάθε διάσταση, που σημαίνει ότι κοντινά σημεία του πολυδιάστατου χώρου παραμένουν κοντινά και στην μονοδιάστατη απεικόνισή τους. Οπότε, σημασιολογικά κοντινές τιμές αντιστοιχίζονται τελικά σε ένα σχετικά μικρό εύρος διευθύνσεων. Το γεγονός αυτό, το εκμεταλλεύονται οι εφαρμογές με δύο τρόπους: Στα συστήματα διαχείρισης βάσεων δεδομένων, για παράδειγμα, οι αναζητήσεις σε ένα εύρος τιμών του πολυδιάστατου χώρου ικανοποιούνται χρησιμοποιώντας όσο το δυνατόν λιγότερες προσβάσεις στη μνήμη ή στα φυσικά μέσα αποθήκευσης του μηχανήματος και το ίδιο το σύστημα μπορεί αυτόματα να συμπεράνει τις σχετικές σε κάποιο σημείο τιμές [RMF+00].

Υπάρχουν πολλές ΚΠΧ. Η πιο απλή μετατροπή δίνεται από την απλή προσθήκη όλων των κλειδιών στη σειρά, το ένα μετά το άλλο. Αν και η τακτική αυτή έχει μερικά πλεονεκτήματα και χρησιμοποιείται από κάποιες εφαρμογές, η εγγύτητα μεταξύ των πολυδιάστατων δεδομένων διατηρείται μόνο στη μία τους διάσταση – αυτή που χρησιμοποιείται τελευταία στη σειρά. Συνήθως η επιλογή της κατάλληλης καμπύλης περιλαμβάνει την εξισορρόπηση της αποτελεσματικότητας κατά την απεικόνιση και της πολυπλοκότητας του αλγορίθμου μετατροπής. Δύο διαδεδομένες ΚΠΧ, που κλιμακώνονται καλά σε αρκετές διαστάσεις, είναι η λεγόμενη Z ή *σειρά Morton* και η *καμπύλη Hilbert* που προτάθηκε από τον ομώνυμο μαθηματικό στα τέλη του 19ου αιώνα. Όλες οι καμπύλες χρησιμοποιούν ένα αυτο-αναφερόμενο, αναδρομικό γεωμετρικό σχέδιο, που κατασκευάζεται σε βήματα ή αλλιώς *τάξεις*. Το χαρακτηριστικό σχήμα που επαναλαμβάνεται στη καμπύλη Z δίνεται από το ίδιο της το όνομα, ενώ στη καμπύλη του Hilbert, το σχέδιο είναι πιο πολύπλοκο. Η πρώτη τάξη, που περνά από 2^1 θέσεις σε κάθε μία από D διαστάσεις, δίνεται από τον αντίστοιχο κώδικα Grey για D δυαδικά ψηφία. Σε μεγαλύτερες τάξεις, διατηρείται η σειρά τοποθέτησης των αναδρομικών σχημάτων, αλλά τμήματα της καμπύλης περιστρέφονται για να αποφευχθούν οι διασταυρώσεις. Η υλοποίηση της καμπύλης του Hilbert δεν είναι τόσο απλή όσο της Z – που επιτυγχάνεται μέσω της απλής διαπλοκής των δυαδικών αναπαραστάσεων του κλειδιού της κάθε διάστασης – αλλά πετυχαίνει καλύτερη τοπικότητα στις αναφορές.

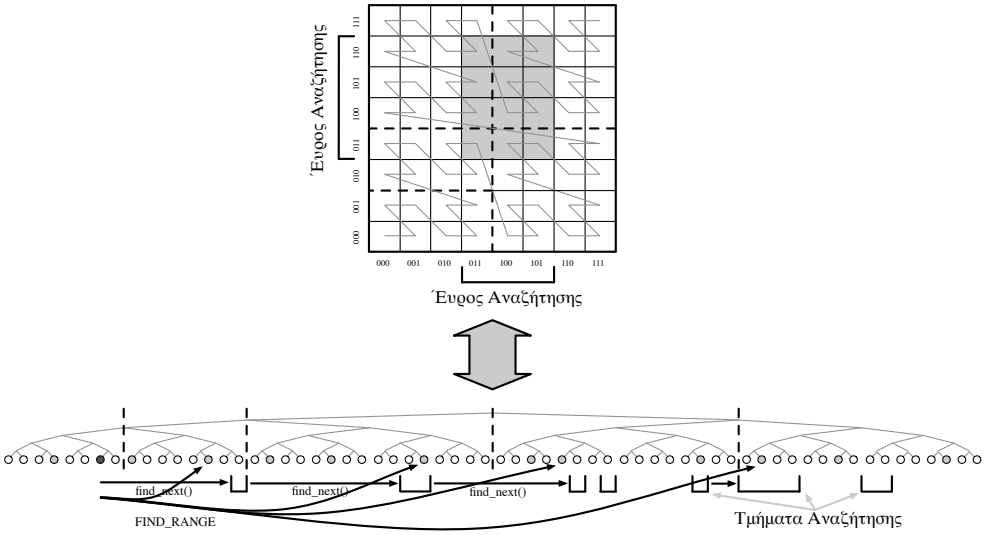
Επεκτείναμε, λοιπόν, μέσω της υλοποίησης των κατάλληλων συναρτήσεων παραγωγής και μεταγλώττισης κλειδιών ΚΠΧ το δομημένο ΔΟΚ που διαχειρίζεται αναζητήσεις εύρους διευθύνσεων. Όταν αποθηκεύεται ή ζητείται ο εντοπισμός μιας πολυδιάστατης τιμής, γίνεται πρώτα η κατάλληλη μετατροπή των πολλαπλών κλειδιών σε μια διεύθυνση, η οποία και χρησιμοποιείται στη επίπεδο της επικάλυψης. Έτσι, υποθέτοντας ότι κάθε κλειδί του αρχικού χώρου κωδικοποιεί και ένα χαρακτηριστικό για το αντίστοιχο δεδομένο, στον ΚΠΚ θα τοποθετούνται αυτόματα οι σημασιολογικά σχετικές τιμές και σε κοντινές διευθύνσεις του ιδεατού χώρου. Σε ένα τέτοιο δίκτυο, η αναζήτηση ενός εύρους διευθύνσεων αποκτά σημαντικό ρόλο και μετατρέπεται σε ουσιώδες χαρακτηριστικό του ΚΠΚ. Η δυνατότητα πραγματοποίησης αναζητήσεων γρήγορα και με ακρίβεια είναι απαραίτητη για την εκμετάλλευση των σχέσεων που έχουν πλέον μεταξύ τους οι γειτονικές τιμές.

Πριν εξετάσουμε τα βήματα του σχετικού αλγορίθμου, να σημειωθεί ότι μια αναζήτηση εύρους διευθύνσεων σε πολλαπλές διαστάσεις αντιστοιχεί κάθε φορά σε ένα σύνολο από ζευγάρια ορίων

διευθύνσεων στον μονοδιάστατο χώρο. Η μετατροπή εξαρτάται από την καμπύλη που χρησιμοποιείται. Όμως, ο υπολογισμός των ορίων στον μονοδιάστατο χώρο είναι μια εξαιρετικά πολύπλοκη υπολογιστικά πράξη και μπορεί να απαιτεί απαγορευτικά πολύ χρόνο. Σε κάθε περίπτωση πρέπει να αποφεύγεται η απευθείας μετατροπή, καθώς ακόμα και μια απλή σχετικά αναζήτηση στον πολυδιάστατο χώρο μπορεί να αναφέρεται σε πολλαπλά, μη συνεχόμενα τμήματα της καμπύλης. Είναι προφανές ότι ο υπολογισμός των τμημάτων αυτών και η μετέπειτα παραγωγή και εκτέλεση όλων των σχετικών πράξεων αναζήτησης εύρους διευθύνσεων, δεν επαρκεί. Και πάλι, η πρόκληση που καλούμαστε να αντιμετωπίσουμε είναι η εύρεση μιας μεθόδου που να αποδίδει καλύτερα από την αφελή προσέγγιση του να επεξεργαστούμε το πρόβλημα σειριακά, επαναλαμβάνοντας κάθε φορά τον ίδιο αλγόριθμο για μια υποπεριοχή του συνόλου, χωρίς να λαμβάνουμε υπόψη ότι οι περιοχές αυτές δεν είναι κατά ανάγκη ανεξάρτητες.

Προτείνουμε, λοιπόν, μια προσέγγιση, που εκμεταλλεύεται τα χαρακτηριστικά των *γειτονιών κόμβων*. Σε αντιστοιχία με το πρωτόκολλο που ακολουθείται στη μονοδιάστατη περίπτωση, ο στόχος είναι να επισκεπτόμαστε κάθε γειτονιά το πολύ μία φορά ανά αναζήτηση. Σε κάθε βήμα, ο συμμετέχων που ερωτάται για τη γειτονιά του, πρέπει να απαντά με όλες τις τιμές που ανήκουν στο εύρος διευθύνσεων που καλύπτει – τα όρια της γειτονιάς στην οποία ανήκει. Για να προχωρήσει ο αλγόριθμος στον επόμενο κόμβο, της επόμενης γειτονιάς, χρησιμοποιούμε μια ειδική συνάρτηση της καμπύλης, που ονομάζεται `find_next(query, id)`, η οποία επιστρέφει το επόμενο κλειδί, μετά τη δοθείσα διεύθυνση, που περιλαμβάνεται στο πολυδιάστατο εύρος αναζήτησης [TH81]. Η πολυπλοκότητα της `find_next()` εξαρτάται από τον τύπο της καμπύλης, αλλά σε κάθε περίπτωση είναι πολύ πιο αποδοτικό να εκτελείται η αναζήτηση από τη σκοπιά των γειτονιών, παρά να χρησιμοποιείται η ΚΠΧ για την εύρεση των μονοδιάστατων ορίων που αντιστοιχούν στα αρχικά κλειδιά. Ενδεικτικά αναφέρεται ότι η κακή απόδοση και κλιμάκωση ενός συστήματος που χρησιμοποιούσε απευθείας τον ΚΠΧ αποτέλεσε εν μέρει το έναυσμα για την ευρύτερη ενασχόληση με το συγκεκριμένο πρόβλημα. Από την άλλη μεριά, με τη χρήση των γειτονιών, ακόμα και αν η αναζήτηση περιλαμβάνει ένα μεγάλο εύρος διευθύνσεων, ο αλγόριθμος φροντίζει αυτόματα να προωθηθεί το αίτημα μόνο στους απόλυτα αναγκαίους κόμβους, χωρίς να σπαταλά χρόνο σε πολύπλοκους υπολογισμούς ορίων.

Στο σχήμα 4.4 δίνεται ένα παράδειγμα της διαδικασίας αναζήτησης χρησιμοποιώντας έναν διδιάστατο χώρο έξι δυαδικών ψηφίων – τρία ανά διάσταση. Μέσω του μετασχηματισμού Z αντιστοιχίζεται κάθε ζευγάρι συντεταγμένων του επιπέδου, σε μια μοναδική διεύθυνση της επικάλυψης. Το εύρος αναζήτησης στις δύο διαστάσεις αναπαριστάται με ένα ορθογώνιο παραλληλόγραμμο, το οποίο περιέχει όλες τις τιμές που θα πρέπει να απαντηθούν – το *κουτί αναζήτησης*. Θεωρούμε ότι στο δίκτυο Kademlia, που φαίνεται στη κάτω μεριά, φιλοξενούνται 13 κόμβοι, από τις θέσεις των οποίων προκύπτουν και οι γειτονιές που διαχωρίζονται με τις έντονες, διακεκομμένες γραμμές. Με τις ίδιες γραμμές διαχωρίζουμε και τον διδιάστατο χώρο, δείχνοντας ποιά τμήματα της καμπύλης αντιστοιχούν σε κάθε γειτονιά. Παρατηρούμε ότι υπάρχει μια αντιστοιχία ένα-προς-ένα μεταξύ των ορίων των γειτονιών και των περιοχών της καμπύλης Z, που προέρχεται από το γεγονός ότι και οι δύο δομές κατασκευάζονται αναδρομικά με παρόμοιο τρόπο. Έτσι, το πρώτο μισό της καμπύλης αντιστοιχεί στο πρώτο αριστερό υποδέντρο της επικάλυψης. Το



Σχήμα 4.4: *Αναπαράσταση μιας διδιάστατης αναζήτησης εύρους διευθύνσεων στο επίπεδο, αλλά και στον ΚΠΚ. Τα όρια των γειτονιών και στις δύο περιπτώσεις, παρουσιάζονται με διακεκομμένες γραμμές. Η αναζήτηση χρησιμοποιεί την συνάρτηση* `find_next()` *για να διασχίσει τις γειτονιές απευθείας.*

παραλληλόγραμμο αναζήτησης διασχίζει 4 γειτονιές, οπότε και η διαδικασία ολοκληρώνεται σε 4 μόλις βήματα. Αν γινόταν μετάφραση του εύρους σε τμήματα της καμπύλης, θα δημιουργούνταν 7 ανεξάρτητα ζευγάρια ορίων διευθύνσεων (4 εκ των οποίων συμπίπτουν με απλά σημεία) και κάθε ένα θα απαιτούσε μια ξεχωριστή πράξη μονοδιάστατης αναζήτησης. Με την αξιοποίηση των γειτονιών τα τμήματα της καμπύλης που εμπεριέχονται στην ίδια γειτονιά εξυπηρετούνται από έναν κόμβο.

Τα βήματα του αλγορίθμου που περιγράφηκαν για τη μονοδιάστατη περίπτωση ισχύουν και στην πολυδιάστατη, με τη μόνη εξαίρεση ότι στο βήμα 5, όπου εκμεταλλευόμαστε τη συνάρτηση `find_next()` για να βρεθεί το επόμενο Q εκτός του εύρους διευθύνσεων που έχει ήδη καλυφθεί. Αυτό το Q μπορεί να είναι η επόμενη διεύθυνση αν η κάλυψη σταματά στα μέσα ενός τμήματος αναζήτησης ή μια μακρινή διεύθυνση αν υπερκαλύπτεται το τοπικό κομμάτι της καμπύλης που μας ενδιαφέρει. Σε κάθε περίπτωση, όπως και στη μονοδιάστατη περίπτωση, με το βήμα 5 γίνεται η μετάβαση σε μια επόμενη γειτονιά. Επίσης, τα μηνύματα `FIND_RANGE` περιέχουν το αρχικό διάνυσμα αναζήτησης. ώστε ο παραλήπτης να μπορεί εύκολα να βρει τις τιμές που πρέπει να απαντήσει από την τοπική του βάση δεδομένων. Τα ζεύγη κλειδιών-τιμής αποθηκεύονται με τις διευθύνσεις που τους αντιστοιχούν σε κάθε διάσταση, ώστε η εξυπηρέτηση των αιτήσεων εύρεσης να γίνεται με μία μόνο πράξη. Επιπρόσθετα, η μονοδιάστατη αναζήτηση μπορεί πλέον να χαρακτηριστεί ως μια υποπερίπτωση ενός ενοποιημένου αλγορίθμου, όπου στην απλή περίπτωση η συνάρτηση `find_next()` υλοποιείται απλά με την ανάθεση στο Q της επόμενης διεύθυνσης εκτός κάλυψης, όταν το Q είναι εντός ορίων αναζήτησης. Επίσης, παρατηρούμε ότι το Q_{start} ισούται πάντα με την κλήση `find_next(range, -1)`.

4.4 Αξιολόγηση

Η εκτίμηση της απόδοσης των πράξεων αναζήτησης εύρους διευθύνσεων σε έναν ΚΠΚ απαιτεί τη μελέτη όλων των σχετικών παραμέτρων: του μεγέθους του δικτύου, του μεγέθους του διαστήματος αναζήτησης, της κατανομής των δεδομένων, της κατανομής των κόμβων, του αριθμού των διαστάσεων και της παραμέτρου κ, που ελέγχει το πλήθος των αντιγράφων που τηρούνται για κάθε αντικείμενο, άρα και το μέγεθος των γειτονιών. Μέσω του κ και του μεγέθους του δικτύου, μπορούμε να βρούμε και τον συνολικό αριθμό των γειτονιών που δημιουργούνται σε ένα δίκτυο. Θα δείξουμε, επίσης, ότι η κατανομή των κόμβων δεν παίζει σημαντικό ρόλο στην εκτέλεση μιας αναζήτησης, καθώς η διαδικασία παραγωγής των γειτονιών, λόγω της αναδρομικής της φύσης, μπορεί να ανεχτεί μεγάλες συγκεντρώσεις συμμετεχόντων σε διάφορες περιοχές της επικάλυψης. Από την άλλη μεριά, οι θέσεις των κόμβων, αλλά και των δεδομένων, μπορούν να επηρεάσουν τα χαρακτηριστικά κατανομής φόρτου του συστήματος. Σημεία του δικτύου που φιλοξενούν μεγάλο πλήθος κόμβων ή δεδομένων, είναι λογικό ότι θα συγκεντρώνουν και μεγάλο μέρος των ερωτημάτων αναζήτησης, οπότε και θα απαιτούνται από τα μέλη των αντίστοιχων γειτονιών αυξημένοι πόροι αποθήκευσης, επεξεργασίας, αλλά και εύρους ζώνης (φυσικής διασύνδεσης δικτύου). Το μέγεθος του διαστήματος αναζήτησης, επιβάλλει των αριθμό των γειτονιών που θα καταλήξουν τα μηνύματα εύρεσης τιμών όταν οι διαστάσεις που χρησιμοποιούνται είναι σχετικά λίγες. Μετά από κάποιο πλήθος διαστάσεων, η ΚΠΧ χάνει το χαρακτηριστικό της γνώρισμα να διατηρεί την τοπικότητα των αναφορών, οπότε και ένα μικρό εύρος αναζήτησης μπορεί κατά τη μετατροπή να εκτείνεται σε τμήματα που καλύπτουν ένα μεγάλο ποσοστό της επικάλυψης, άρα και να απαιτεί την επίσκεψη περισσότερων κόμβων.

Καθώς οι προσθήκες στο πρωτόκολλο του ΚΠΚ δεν αλλάζουν τον τρόπο με τον οποίο οργανώνονται οι επαφές, συνεπάγεται ότι απλές, σημειακές πράξεις συνεχίζουν να κοστίζουν περίπου $\log N$ βήματα, για ένα δίκτυο συμμετεχόντων. Θεωρούμε ότι οι λειτουργίες αναζήτησης εύρους διευθύνσεων απαιτούν ένα μέγιστο $\frac{S}{S_{max}} E \log N$ μηνυμάτων, όπου S είναι το εύρος αναζήτησης, S_{max} το συνολικό πλήθος διευθύνσεων του ιδεατού χώρου της επικάλυψης και E ο αριθμός των γειτονιών του δικτύου. Σε μια πολυδιάστατη αναζήτηση, αν με το S αναπαριστούμε τη συνολική κάλυψη όλων των τμημάτων της καμπύλης που αντιστοιχούν στο ερώτημα, συμπεριλαμβανομένου και των κενών διαστημάτων, ο τύπος μας δίνει τη χειρότερη περίπτωση – εκείνη που θα πρέπει κατά τη διαδικασία να επισκεφτούμε όλες τις ενδιάμεσες γειτονιές. Διαισθητικά, η πολυπλοκότητα μιας αναζήτησης είναι ανάλογη με τον αριθμό των γειτονιών που καλύπτει. Αφού κάθε επίσκεψη σε μια γειτονιά απαιτεί μια εύρεση κοντινών κόμβων κόστους $\log N$, ο όρος $\frac{S}{S_{max}}$ ορίζει το ποσοστό των υποδέντρων που θα επισκεφτεί η αναζήτηση, ανάλογα με το μέγεθος του εύρους που ζητείται. Αν και ο υπολογισμός του κόστους αναζήτησης με βάση το συγκεκριμένο ποσοστό υπονοεί την ομαλή κατανομή των κόμβων, χρησιμεύει σαν ένα αρχικό εργαλείο εκτίμησης πριν συζητηθεί το θέμα της άνισης κατανομής των κόμβων ή των δεδομένων. Έτσι και αλλιώς, η χειρότερη περίπτωση κάθε φορά είναι η αναζήτηση *όλων* των διευθύνσεων, κάτι που περιλαμβάνει το πέρασμα από όλες τις γειτονιές, ανεξαρτήτως κατανομής.

Στις επόμενες παραγράφους παρουσιάζεται ο τρόπος υπολογισμού του E, τόσο μέσω μαθημα-

τικών εκφράσεων όσο και εμπειρικά. Επίσης, δείχνεται μέσω πειραματικών μετρήσεων πώς επηρεάζεται το S από τον αριθμό των διαστάσεων και προτείνονται τακτικές ρύθμισης και βελτιστοποίησης του δικτύου, καθώς και τεχνικές κατανομής του φόρτου.

4.4.1 Μέτρηση των γειτονιών

Για να μετρήσουμε το συνολικό πλήθος των γειτονιών που δημιουργούνται σε ένα δίκτυο Kademlia με N συμμετέχοντες και παράμετρο μεγέθους γειτονιάς κ, θεωρούμε το ακόλουθο γενικό πρόβλημα.

Πρόβλημα. Έστω ένα δυαδικό δέντρο *άπειρου* βάθους. Θεωρούμε ότι το βάθος είναι ένας αριθμός που μας προσδιορίζει την απόσταση από τη ρίζα. Το βάθος της ρίζας ίσο με 0 και αυξάνεται κατά 1 για κάθε επίπεδο του δέντρου. Θεωρούμε ότι κάθε κόμβος είναι ένας *κουβάς* που μπορεί να χωρέσει μέχρι $\kappa - 1$ μπάλες. Όταν στον κόμβο υπάρχουν ήδη κ μπάλες και προστεθεί άλλη μία, ο πάτος του κουβά *σπάει* και το περιεχόμενό του κατανέμεται τυχαία στους δύο κόμβους του επόμενου επιπέδου. Όλες οι μπάλες κατρακυλούν στο παρακάτω επίπεδο – κάθε μία ακολουθεί ένα προκαθορισμένο, αλλά ομοιόμορφα τυχαίο (άπειρο) μονοπάτι.

Αν σε αυτό το δέντρο τοποθετήσουμε N μπάλες στη ρίζα, όπου $N \geq \kappa$, κάποια στιγμή η παραπάνω διαδικασία θα σταματήσει και οι μπάλες θα έχουν κατανεμηθεί τυχαία σε κάποιους κόμβους του δυαδικού δέντρου, που ονομάζουμε και φύλλα της δομής. Το ερώτημα που τίθεται είναι: Ποιός είναι ο μέσος αριθμός των φύλλων; Πόσοι θα είναι κατά μέσο όρο οι κόμβοι που θα έχουν μπάλες;

Λύση. Ξεκινώντας από τη ρίζα του δέντρου, ο αριθμός των φύλλων είναι 1. Κάθε φορά που γεμίζει ένας κόμβος και σπάει ο πάτος του αντίστοιχου κουβά, ο αριθμός αυτός αυξάνεται κατά 1. Άρα ο μέσος αριθμός των φύλλων ισούται με τον μέσο αριθμό των κόμβων που έχουν γεμίσει. Ονομάζουμε τη πιθανότητα ένας κόμβος να γεμίσει ως P_{break}. Οπότε, ο μέσος αριθμός των φύλλων ισούται με το άθροισμα της πιθανότητας αυτής για *όλους* τους κόμβους του άπειρου δέντρου.

Στο επίπεδο d θα υπάρχουν 2^d κόμβοι, και κάθε ένας θα έχει πιθανότητα 2^{-d} να λάβει μια μπάλα που θα έχει φτάσει στο επίπεδό του, καθώς η κατανομή των μπαλών που περνάνε από κάθε επίπεδο είναι ομοιόμορφη. Επίσης, δεδομένης της ομοιόμορφης κατανομής των μπαλών εντός ενός επιπέδου, κάθε κουβάς του ίδιου επιπέδου έχει την ίδια πιθανότητα να σπάσει $P_{break}(N, \kappa, d)$.

Ο μέσος όρος των φύλλων είναι λοιπόν:

$$E(N, \kappa) = 2 + \sum_{d=1}^{\infty} 2^d P_{break}(N, \kappa, d) \tag{4.1}$$

Αφού $N \geq \kappa$, η ρίζα πάντα σπάει και δίνει ένα φύλλο.

Για να βρούμε το P_{break}, θεωρούμε P_x να είναι η πιθανότητα ότι ακριβώς x από τις N μπάλες θα περάσουν μέσα από έναν κόμβο του επιπέδου d. Το γεγονός μπορεί να κατασκευαστεί επιλέ-

γοντας x από N για να τοποθετηθούν στον κόμβο και αφήνοντας τις υπόλοιπες $N - x$ μπάλες να πάνε ελεύθερα σε οποιονδήποτε από τους υπόλοιπους $2^d - 1$ κόμβους:

$$P_x = \binom{N}{x}(2^d - 1)^{(N-x)}(2^{-d})^N$$

Επομένως,

$$P_{break} = \sum_{x=\kappa}^{N} P_x = 1 - \sum_{x=0}^{\kappa-1} P_x$$

Οπότε η (4.1) γίνεται:

$$E(N, \kappa) = 2 + \sum_{d=1}^{\infty} 2^{-d(N-1)} \sum_{x=\kappa}^{N} \binom{N}{x}(2^d - 1)^{(N-x)}$$

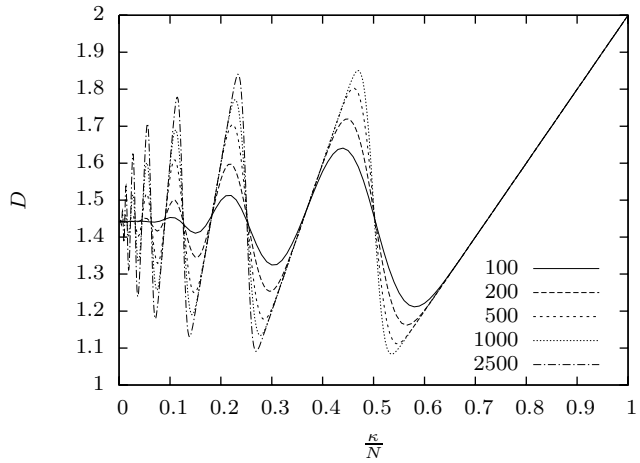
Ή σε μια μορφή πιο κατάλληλη για υπολογισμό, όταν $\kappa \ll N$:

$$E(N, \kappa) = 2 + \sum_{d=1}^{\infty} 2^d \left(1 - 2^{-dN} \sum_{x=0}^{\kappa-1} \binom{N}{x}(2^d - 1)^{(N-x)}\right) \tag{4.2}$$

□

Υποθέτοντας την "ιδανική" τοποθέτηση κόμβων, όπου όλες οι γειτονιές θα έχουν όσο το δυνατόν πιο κοντά στους $\kappa - 1$ συμμετέχοντες, το E θα πλησιάζει το $\frac{N}{\kappa-1}$. Ωστόσο, λόγω της τυχαίας κατανομής τους στον εικονικό χώρο διευθύνσεων, η *πυκνότητα* κάθε γειτονιάς θα είναι σχεδόν πάντα μικρότερη από την μέγιστη. Γνωρίζοντας το $E(N, \kappa)$, κατασκευάζουμε μια συνάρτηση που συνδέει τον μέσο αριθμό γειτονιών με το μικρότερο δυνατό $E_{min} = \frac{N}{\kappa-1}$. Στο σχήμα , 4.5 παρουσιάζεται ο *συντελεστής πυκνότητας* $D = \frac{E(N,\kappa)}{E_{min}}$ ως συνάρτηση του $\frac{\kappa}{N}$. Είναι εμφανές ότι το D ταλαντεύεται με μεγαλύτερο πλάτος όσο το κ πλησιάζει το N. Παρόλα αυτά, η περιοχή του γραφήματος που έχει ιδιαίτερο ενδιαφέρον για ένα δίκτυο Kademlia είναι εκεί όπου $\kappa \ll N$. Από αυτό και αντίστοιχα γραφήματα, εικάζεται ότι η άξονας της ταλάντωσης είναι η γραμμή $D_{axis} = \frac{1}{\ln 2}$, οπότε και γενικώς χρησιμοποιούμε τον τύπο $E = 1.44\frac{N}{\kappa-1}$ για να υπολογίσουμε πρακτικά τον αριθμό των γειτονιών σε μια τυπική επικάλυψη.

Επιπλέον, έχοντας εκτελέσει πολλαπλά πειράματα δημιουργίας γειτονιών, καταλήγουμε στο ότι αυτή η εμπειρική προσέγγιση του E είναι πολύ κοντά στη πραγματικότητα, ακόμα και όταν δεν κατανέμονται οι κόμβοι ομοιόμορφα. Στο σχήμα 4.6, δείχνουμε τον αριθμό των κόμβων ως συνάρτηση του κ, για δύο διαφορετικές κατανομές αναγνωριστικών σε έναν διδιάστατο χώρο, μήκους 20 δυαδικών ψηφίων σε κάθε διάσταση. Ο διδιάστατος χώρος επιλέχθηκε ως πιο απλός στην οπτικοποίηση των συγκεντρώσεων των κόμβων στις αντίστοιχες περιοχές της επικάλυψης. Για τη μετατροπή των αναγνωριστικών σε μορφή κατάλληλη για το Kademlia, χρησιμοποιήθηκε ο μετασχηματισμός Z. Έχει ενδιαφέρον το γεγονός ότι ακόμα και οι μη τυχαίες κατανομές φαί-



Σχήμα 4.5: *Ο συντελεστής πυκνότητας των γειτονιών ως συνάρτηση του $\frac{\kappa}{N}$. Όταν το $\kappa \ll N$ ο αριθμός των γειτονιών είναι περίπου 44% μεγαλύτερος από το ιδανικό $\frac{N}{\kappa-1}$.*

νεται ότι ακολουθούν τον εμπειρικό κανόνα, κάτι που οφείλεται κυρίως στον αναδρομικό τρόπο με τον οποίο δημιουργούνται οι γειτονιές.

$$\lim_{\frac{\kappa}{N}\to 0} D = D_{axis} = \frac{1}{\ln 2}$$

Μάλιστα, η εφαρμοσιμότητα της εμπειρικής πρόβλεψης ενισχύεται από το γεγονός ότι ακόμα και όταν η εξίσωση (4.2) γενικευτεί για δέντρα b φύλλων ανά κόμβο (b-ary trees):

$$E_b(N,\kappa,b) = b + (b-1)\sum_{d=1}^{\infty} b^d P_{nobreak}(N,\kappa,d,b)$$

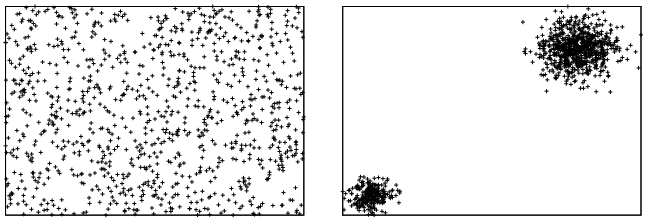
$$P_{nobreak}(N,\kappa,d,b) = 1 - b^{-dN}\sum_{x=0}^{\kappa-1}\binom{N}{x}(b^d-1)^{(N-x)}$$

Τα πειραματικά αποτελέσματα υποδεικνύουν ότι για $\kappa \ll N$ ο συντελεστής πυκνότητας φαίνεται ότι ορίζεται από τον τύπο:

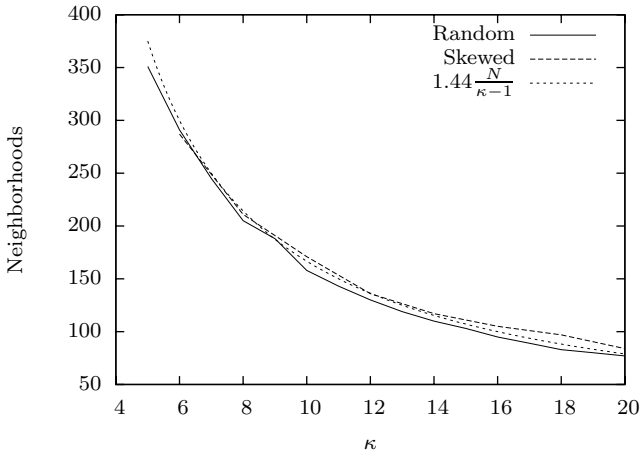
$$\lim_{\frac{\kappa}{N}\to 0} \frac{E_b(N,\kappa,b)}{E_{min}} = \frac{b-1}{\ln b}$$

4.4.2 Η επίδραση του αριθμού των διαστάσεων

Όταν μια αναζήτηση αναφέρεται σε μονοδιάστατα δεδομένα, το ποσοστό των γειτονιών στις οποίες καταλήγουν μηνύματα ισούται με το ζητούμενο εύρος διευθύνσεων δια του συνολικού εύρους που χρησιμοποιείται στο δίκτυο, S_{max}. Όταν, όμως, οι εντολές εκτελούνται στον πολυδιάστατο χώρο, ο αριθμός των επηρεαζόμενων γειτονιών εξαρτάται και από τον αριθμό των διαστάσεων που έχουν οριστεί κατά τη κατασκευή της επικάλυψης. Κάθε πολυδιάστατων περιοχή



(α′) Τυχαία κατανομή (β′) Μη τυχαία κατανομή

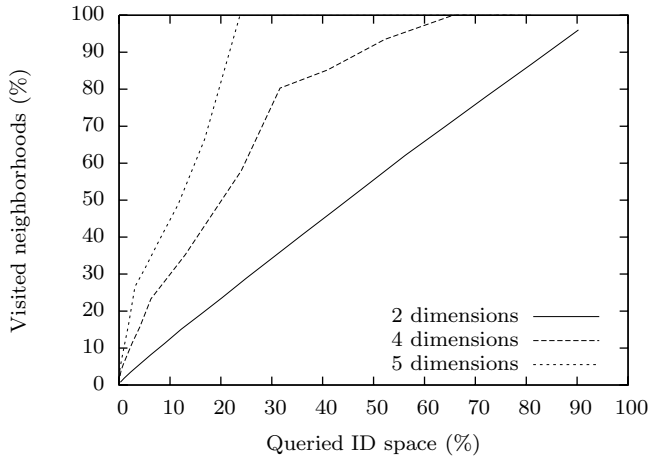


(γ′) Αριθμός γειτονιών

Σχήμα 4.6: *Δύο ενδεικτικές κατανομές κόμβων και ο αντίστοιχος αριθμός γειτονιών που δημιουργούν ως συνάρτηση του κ. Κάθε δίκτυο περιλαμβάνει 1000 κόμβους τοποθετημένους σε έναν διδιάστατο χώρο 20 δυαδικών ψηφίων ανά διάσταση.*

διευθύνσεων αντιστοιχεί σε ένα σύνολο τμημάτων της επικάλυψης. Μάλιστα, όσο λιγότερες διαστάσεις χρησιμοποιούνται τόσο πιο μεγάλα και συνεχόμενα μεταξύ τους είναι αυτά τα τμήματα. Η κλιμάκωση των διαστάσεων σε μεγάλα νούμερα, έχει ως αποτέλεσμα οι πράξεις αναζήτησης να αντιστοιχίζονται σε όλο και μικρότερα σύνολα διευθύνσεων, όλο και πιο απλωμένα στο δίκτυο. Για να προσδιορίσουμε το μέγεθος και τις συνέπειες αυτού του φαινομένου, εκτελέσαμε πολλαπλές αναζητήσεις διαφόρων μεγεθών σε έναν ΚΠΚ 10.000 κόμβων, μετρώντας κάθε φορά τον αριθμό των γειτονιών που συμμετείχαν στην συλλογή των αποτελεσμάτων. Τα αποτελέσματα εξαρτώνται βέβαια και από τη καμπύλη που χρησιμοποιείται. Τα πειράματα που παρουσιάζονται έγιναν με την ΚΠΧ Z. Αν και η καμπύλη Hilbert συμπεριφέρεται γενικώς καλύτερα από την Z για ίδιες διαστάσεις, η ανάλυση είναι αντιπροσωπευτική και οι ίδιες επιπτώσεις αναμένονται στα αντίστοιχα όρια κλιμάκωσης και άλλων ΚΠΧ.

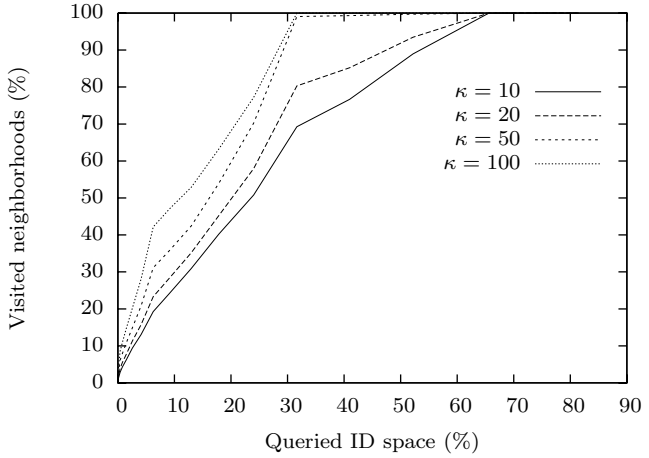
Στο σχήμα 4.7, φαίνεται η σχέση μεταξύ δύο ποσοστών: του μέσου όρου βημάτων του αλγορίθμου (γειτονιές που επισκέπτονται), σε συνάρτηση με το ποσό του χώρου που περιλαμβάνεται στο ερώτημα. Κάθε εκτέλεση, για κάθε διάσταση, περιλαμβάνει τα αποτελέσματα περίπου 10.000 αναζητήσεων που έχουν δημιουργηθεί τυχαία ώστε να μοιράζονται ομοιόμορφα στον οριζόντιο άξονα. Όλα τα κουτιά αναζήτησης ή *υπερκύβοι* για μεγαλύτερες διαστάσεις, έχουν ίσες πλευρές. Το κ είναι 20, οπότε ο συνολικός αριθμός των γειτονιών σε κάθε περίπτωση είναι περίπου 750.



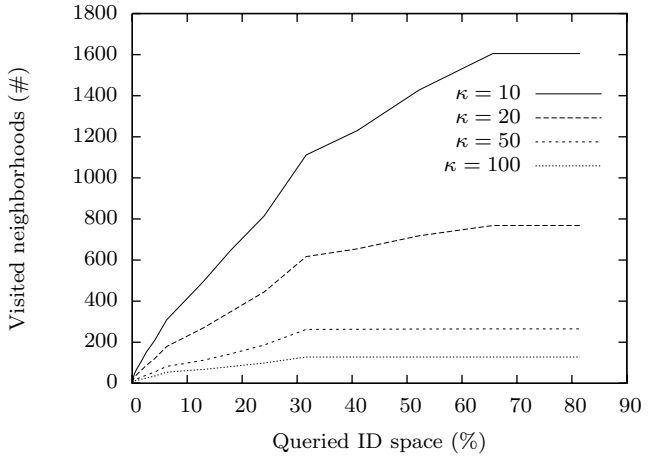
Σχήμα 4.7: *Επισκεπτόμενες γειτονιές ως συνάρτηση του εύρους αναζήτησης, για δεδομένα διαφορετικών διαστάσεων.*

Είναι εμφανές ότι αυξάνοντας τον αριθμό των διαστάσεων, ο κόμβος που επιτελεί την αναζήτηση επισκέπτεται όλες τις διαθέσιμες γειτονιές ταχύτερα. Αν και στις δύο διαστάσεις η συμπεριφορά είναι γραμμική, για τέσσερις όλες οι γειτονιές συμμετέχουν στη παραγωγή της απάντησης όταν το ποσοστό του χώρου που αναζητείται ξεπερνά το 70% του συνόλου. Για πέντε διαστάσεις αυτό συμβαίνει αρκετά νωρίτερα, περίπου στο 23%.

Τα γραφήματα εξαρτώνται επίσης από την παράμετρο κ. Αυξάνοντας το κ, μειώνεται ο αριθμός των γειτονιών και άρα ο αριθμός των μηνυμάτων ανά ερώτημα, αλλά μεγαλώνουν τα υποδέντρα, οπότε και η διαδικασία γίνεται λιγότερο "λεπτομερής'' και το ποσοστό κάλυψης του δικτύου μεγαλύτερο. Όταν οι γειτονιές είναι μικρές, μπορεί να είναι περισσότερες, αλλά με μεγάλη πιθανότητα μια αναζήτηση θα προσπεράσει μερικές καθώς κινείται από το Q_{start} στο Q_{end}. Αυτό δείχνεται στο σχήμα 4.8. Για τον ίδιο αριθμό διαστάσεων, όταν $\kappa = 10$ η κλιμάκωση του ποσοστού των γειτονιών που επηρεάζονται ανάλογα με το μέγεθος του εύρους αναζήτησης είναι πιο ομαλή, παρόλο που ο πραγματικός αριθμός των γειτονιών είναι σχεδόν διπλάσιος από όταν $\kappa = 20$. Ακόμα και έτσι όμως, μια αναζήτηση που αναφέρεται σχεδόν σε όλα τα διαθέσιμα δεδομένα, θα εξυπηρετηθεί από έναν μικρό σχετικά αριθμό κόμβων – ένα μικρό ποσοστό του συνολικού πλήθους – το οποίο εξαρτάται άμεσα από την τιμή του κ. Μεγαλύτερες τιμές μπορεί να μειώνουν τον αριθμό των μηνυμάτων και άρα τον χρόνο που απαιτεί το ερώτημα, αλλά έχουν παρενέργειες που θα συζητηθούν στη συνέχεια.



(α′) Ποσοστό γειτονιών



(β′) Αριθμός γειτονιών

Σχήμα 4.8: *Μεταβάλλοντας το μέγεθος των γειτονιών, ενώ διατηρείται ο αριθμός των διαστάσεων σε* 4*, σε ένα δίκτυο* 10.000 *κόμβων.*

4.4.3 Παράμετροι λειτουργίας

Θεωρώντας ότι η κατανομή των κόμβων στον εικονικό χώρο διευθύνσεων είναι τυχαία, εκτός από το εύρος των ζητούμενων διευθύνσεων, μια πράξη αναζήτησης εξαρτάται από τις παραμέτρους N, κ, καθώς και από τον αριθμό των διαστάσεων που χρησιμοποιούνται από τα δεδομένα. Μάλιστα, ο αριθμός αυτός και το κ επιλέγονται μία φορά και παραμένουν σταθερά σε τη διάρκεια ζωής του δικτύου. Η επιλογή των κατάλληλων τιμών έχει ιδιαίτερη σημασία και σε αυτό μπορεί να βοηθήσει μια γενική γνώση ή εκτίμηση των λειτουργιών που αναμένεται να εξυπηρετεί το ΔΟΚ.

Η παράμετρος πλήθους αντιγράφων και μεγέθους γειτονιάς, κ, ισούται συνήθως με 20. Δεδομένου ενός δικτύου χιλίων κόμβων, $\kappa = 20$ και μιας αναζήτησης που περιλαμβάνει όλο το εύρος διευθύνσεων, θα απαιτούνται περίπου 75 πράξεις – βήματα του αλγορίθμου. Ο αριθμός αυτός μπορεί να θεωρηθεί λογικός, τόσο όσον αφορά τον χρόνο, όσο και τους πόρους δικτύου, που αντιστοιχεί. Στο ίδιο σενάριο με δέκα φορές περισσότερους κόμβους, δηλαδή δέκα χιλιάδες, θα απαιτούνται και σχεδόν δέκα φορές παραπάνω βήματα, εκτός αν αυξηθεί το κ επί δέκα και γίνει 200. Ο αριθμός των γειτονιών E είναι ευθέως ανάλογος του πληθυσμού των κόμβων N, οπότε όταν μεγαλώνει το δίκτυο κατά μία τάξη μεγέθους, αναμένεται και η αντίστοιχη αύξηση στη ποσότητα της επικοινωνίας που απαιτεί κάθε αναζήτηση εύρους διευθύνσεων. Αν και μέρος της επικοινωνίας γίνεται παράλληλα, ο συνολικός χρόνος που θα πάρει μια αναζήτηση θα είναι πολλαπλάσιος όσο αυξάνουν οι κόμβοι που συμμετέχουν στην επικάλυψη. Μεγαλύτερες τιμές του κ μπορούν να βοηθήσουν να διατηρηθεί ο αριθμός των γειτονιών σε χαμηλά επίπεδα, αλλά επιβάλουν αντίστοιχη αύξηση στον αριθμό των αντιγράφων που τηρούνται για κάθε ζεύγος κλειδιού-τιμής, που με τη σειρά του προκαλεί επιπλέον επικοινωνία για τη διαχείριση και κατανομή των δεδομένων.

Προτείνεται, λοιπόν, το κ να παίρνει τιμές από 10 έως 20 – ανάλογα με το αναμενόμενο μέγεθος του δικτύου, προκειμένου να αποφεύγεται και το πρόβλημα της *υπερ-αντιγραφής*. Επιπλέον, όταν η επικάλυψη φιλοξενεί παραπάνω από μερικές χιλιάδες συμμετέχοντες, οι λειτουργίες αναζήτησης θα πρέπει να περιορίζονται ως προς το μέγεθος του εύρους διευθύνσεων στο οποίο αναφέρονται ή πολύ μεγάλες περιοχές να χωρίζονται σε μικρότερες και οι αντίστοιχες πράξεις να δίνονται παράλληλα προς εκτέλεση. Έτσι, το σύστημα θα φαίνεται πιο αποκρίσιμο από τη πλευρά της εφαρμογής. Εξ άλλου, η μελέτη των σχετικών εργασιών δείχνει ότι συνήθως οι εφαρμογές εργάζονται μόνο σε ένα υποσύνολο του χώρου, που δεν ξεπερνά το 10% του συνόλου. Μάλιστα, μπορεί να βελτιστοποιηθεί ο αλγόριθμος αναζήτησης, ώστε να διαιρεί και να παραλληλοποιεί αυτόματα μεμονωμένες αναζητήσεις ή ακόμα και να διατρέχει τον χώρο διευθύνσεων αμφίδρομα – ταυτόχρονα, δηλαδή να επεξεργάζεται το αίτημα και από τις μικρότερες στις μεγαλύτερες διευθύνσεις, αλλά και αντίστροφα.

Σχετικά με τον αριθμό των διαστάσεων, προτείνεται να χρησιμοποιείται ο όσο το δυνατόν μικρότερος, προκειμένου να διατηρείται η αναλογία εύρους αναζήτησης και περιοχής κάλυψης του δικτύου. Γενικά, λόγω των ίδιων των ΚΠΧ, ένας πρακτικός περιορισμός είναι κοντά στις 4-5 διαστάσεις. Μεγαλύτερα νούμερα ενισχύουν το φαινόμενο όπου μια αναζήτηση που αναφέρε-

ται σε πολύ μικρό ποσοστό του εικονικού χώρου διευθύνσεων, αναγκάζεται να επισκέπτεται ένα μεγάλο ποσοστό γειτονιών. Αν τα δεδομένα έχουν ήδη παραπάνω αναγνωριστικά, μπορεί να χρειαστεί να περάσουν από κάποιο φίλτρο επεξεργασίας για να μειωθεί ο αριθμός που θα χρησιμοποιηθούν τελικά. Για το σκοπό αυτό, προτείνονται από τη σχετική βιβλιογραφία δύο μέθοδοι, που ανάλογα με τα χαρακτηριστικά των αναγνωριστικών μπορούν ακόμα και να συνδυαστούν. Σύμφωνα με την πρώτη μέθοδο, κάποιο αναγνωριστικά μετατρέπονται σε παραμέτρους των δεδομένων. Ουσιαστικά, μέσα από τη διαδικασία αυτή, επιλέγεται ένα μέρος των αναγνωριστικών και αυτά ορίζονται ως διαστάσεις. Οι καλύτεροι υποψήφιοι είναι, εκτός των πιο αντιπροσωπευτικών, αυτά των οποίων οι τιμές κατανέμονται καλύτερα στα δεδομένα. Από την άλλη, αναγνωριστικά που παίρνουν, για παράδειγμα, δυαδικές τιμές μπορούν να αποθηκευτούν μαζί με τα δεδομένα και να προστεθεί σε κάθε αναζήτηση ένα τελικό στάδιο επιλογής, όπου να φιλτράρονται τα αποτελέσματα με βάση τα αναγνωριστικά που περιέχουν. Η δεύτερη μέθοδος προτείνει τον συνδυασμό δύο ή περισσότερων αναγνωριστικών σε μία αριθμητική διάσταση και έχει οφέλη μόνο όταν δεν αναμένεται ή δεν επιτρέπεται να γίνονται αναφορές ταυτόχρονα στα αναγνωριστικά αυτά.

4.4.4 Κατανέμοντας τον φόρτο

Η χρήση μιας ΚΠΧ για την τοποθέτηση των δεδομένων στον εικονικό χώρο της επικάλυψης, έχει ως αποτέλεσμα την ομαδοποίηση των παραγόμενων διευθύνσεων ανάλογα με τις σημασιολογικές τους συγγένειες στον πολυδιάστατο χώρο. Έτσι, εκτός αν τα δεδομένα είναι τυχαία κατανεμημένα σε όλες τις διαστάσεις, θα δημιουργούνται ανώμαλες συγκεντρώσεις τιμών στο επίπεδο του δικτύου. Αν και, όπως έχουμε δείξει, το γεγονός αυτό δεν επηρεάζει την ομαλή λειτουργία του δικτύου με κάποιο τρόπο, οι άνισες πυκνότητες δεδομένων μπορεί να προκαλέσουν και αντίστοιχες ανισότητες στον φόρτο στον οποίο καλούνται να αντεπεξέλθουν οι κόμβοι. Για παράδειγμα, κάποιες γειτονιές μπορεί να αποθηκεύουν πολύ περισσότερα δεδομένα από άλλες. Η έννοια του φόρτου μπορεί να έχει δύο βασικές μορφές. Αναφέρεται στις αυξημένες απαιτήσεις που δημιουργούνται, είτε στην χωρητικότητα αποθήκευσης ενός κόμβου, είτε στις δυνατότητες επεξεργασίας του, δηλαδή τον συνδυασμό της διαθέσιμης υπολογιστικής ισχύος του και της χωρητικότητας της γραμμής μεταφοράς δεδομένων που τον συνδέει με τους υπόλοιπους συμμετέχοντες.

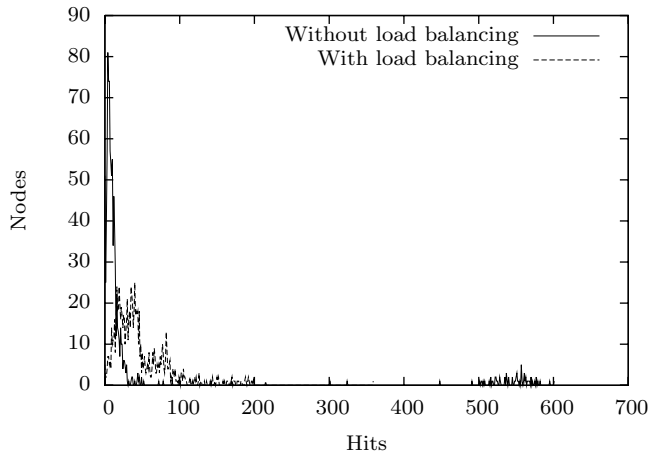
Ζητήματα κατανομής φόρτου μπορούν να προκύψουν αν το πλήθος και μέγεθος των δεδομένων είναι πολύ μεγάλα, αν η αναμενόμενη κατανομή των αναζητήσεων είναι και αυτή μη ομαλή, ή ακόμα και αν οι συσκευές που απαρτίζουν το δίκτυο είναι σχετικά ανίσχυρες ή περιορίζεται η συνδεσιμότητά τους. Οι λύσεις που έχουν προταθεί μέχρι στιγμής για όταν αντιμετωπίζονται τέτοιου τύπου προβλήματα σε ΚΠΚ είναι, είτε να ελέγχεται η είσοδος των κόμβων στο δίκτυο, οπότε να επιβάλλεται η τοποθέτηση τους σε σημεία που έχουν πολύ φόρτο, είτε να διατίθεται κάποια μέθοδος ώστε να μπορούν να μετακινούνται οι υπάρχοντες κόμβοι από τα σημεία της επικάλυψης στα οποία βρίσκονται, σε σημεία που είναι υπερφορτωμένα. Και οι δύο τακτικές ουσιαστικά προσπαθούν να ταιριάξουν την κατανομή των κόμβων με την ανώμαλη κατανομή

των δεδομένων, έτσι ώστε να διατηρείται σταθερό το ποσοστό των κόμβων που περιλαμβάνονται σε κάθε περιοχή του δικτύου, ανάλογα με τον φόρτο του. Για να επιτευχθεί όμως οποιουδήποτε τύπου ανακατανομή, απαιτείται οπωσδήποτε ένας εξωτερικός ή εσωτερικός μηχανισμός που να παρακολουθεί τα χαρακτηριστικά συγκέντρωσης των ζευγών κλειδιού-τιμής – μηχανισμός, ο οποίος σε συγκεκριμένα όρια πυροδοτεί τους κατάλληλους αλγορίθμους ανάδρασης. Επίσης, αν η συλλογή των δεδομένων ή τα χαρακτηριστικά της είναι γνωστά εκ των προτέρων, είναι πολύ απλό να οριστούν οι θέσεις στις οποίες θα τοποθετηθούν οι κόμβοι. Να σημειωθεί ότι ο αριθμός των γειτονιών δεν θα επηρεαστεί (σχήμα 4.6), αλλά αφού το εύρος διευθύνσεων που θα καλύπτει κάθε γειτονιά θα είναι διαφορετικό, ο μέσος αριθμός μηνυμάτων ανά πράξη αναζήτησης δεν είναι εύκολα προβλέψιμος. Σίγουρα θα είναι μεγαλύτερος από $\frac{S}{S_{max}}E$, αν και φραγμένος από το E.

Επιπρόσθετα, η μεταφορά ενός κόμβου πάνω στην επικάλυψη επιτυγχάνεται καλύτερα μέσω *εικονικών κόμβων*, δηλαδή με το να δίνονται στους κόμβους πολλαπλές διευθύνσεις, οπότε και να εμφανίζονται ταυτόχρονα στα αντίστοιχα σημεία μέσα στον εικονικό χώρο του ΚΠΚ – άρα και στις αντίστοιχες γειτονιές. Αλλιώς, η διαδικασία αποσύνδεσης και επανασύνδεσης του ίδιου κόμβου απαιτεί επιπλέον πράξεις διαχείρισης, με στόχο την ανανέωση των δομών δρομολόγησης και την ανακατανομή των δεδομένων, που με τη σειρά τους εισάγουν καθυστερήσεις σε διάφορα σημεία, αναλόγως τη φύση των μηχανισμών αναγνώρισης λαθών και σταθεροποίησης του δικτύου.

Παρόλα αυτά, θεωρούμε ότι σε δυναμικά δίκτυα η εκτίμηση του φόρτου δεν μπορεί να βασίζεται μόνο σε αντικειμενικά κριτήρια. Οι τεχνικές που αναφέρονται παραπάνω στηρίζονται αποκλειστικά στον αριθμό των κόμβων που συμμετέχουν σε μια περιοχή της επικάλυψης, χωρίς να υπολογίζουν τα ιδιαίτερα χαρακτηριστικά του καθενός. Σε ένα δίκτυο παγκόσμιας κλίμακας, οι συμμετέχοντες θα είναι οπωσδήποτε ετερογενείς, οπότε ο αλγόριθμος κατανομής θα πρέπει σε κάθε περίπτωση και ανά πάσα στιγμή να λαμβάνει υπόψη όχι μόνο τον αριθμό των δεδομένων ή των κόμβων, αλλά και την ποιότητα των μηχανημάτων που δημιουργούν το δίκτυο, τη φύση των διασυνδέσεων, τα δυναμικά χαρακτηριστικά των ερωτημάτων που δίνονται από τους χρήστες, κτλ. Η κατανεμημένη μέτρηση του φόρτου υπό αυτές τις συνθήκες, ο παράλληλος τρόπος υπολογισμού των ορίων της υποδομής, καθώς και η καθολική επικοινωνία και κατάληξη σχετικά με τη κατάσταση του δικτύου αποτελούν σε μεγάλο βαθμό άλυτα προβλήματα, η ανάλυση και ενασχόληση με τα οποία θα αποτελούσε ενδεχομένως έναν ενδιαφέρον μελλοντικό προσανατολισμό για την επέκταση του προτεινόμενου αλγορίθμου.

Στην επικάλυψη που εξετάζεται, ένα ανάλογο αποτέλεσμα κατανομής φόρτου μπορεί να επιτευχθεί μέσω της εκμετάλλευση του γεγονότος ότι *οποιοσδήποτε* κόμβος και να ερωτηθεί από μια γειτονιά, το αποτέλεσμα θα είναι το ίδιο. Στο σχήμα 4.9, απεικονίζεται ο αριθμός των κόμβων ανά αριθμό επισκέψεων, για δυο περιπτώσεις του ίδιου σεναρίου εκτέλεσης, που περιλαμβάνει 10.000 αναζητήσεις σε ένα δίκτυο 1.000 κόμβων. Κάθε αναζήτηση αντιστοιχεί σε 5% του συνολικού χώρου διευθύνσεων. Όταν διασχίζεται το δίκτυο προς μία κατεύθυνση, ο φόρτος συγκεντρώνεται στους συμμετέχοντες που είναι πλησιέστερα στα όρια των γειτονιών. Αυτοί δημιουργούν την αντίστοιχη συγκέντρωση στα δεξιά του γραφήματος – ερωτώνται από 500 έως 600 φορές. Αντίθετα, όταν επιλέγεται τυχαία ένας αντιπρόσωπος από κάθε γειτονιά, οι δύο διακρι-



Σχήμα 4.9: *Αριθμός κόμβων ανά αριθμό επισκέψεων, για* 10.000 *αναζητήσεις εύρους διευθύνσεων σε ένα δίκτυο* 1.000 *κόμβων. Η τυχαία κατανομή των επισκέψεων ανά γειτονιά έχει ως αποτέλεσμα την καλύτερη κατανομή του φόρτου.*

τές περιοχές του αρχικού γραφήματος ενσωματώνονται σε μία. Ο ίδιος αριθμός επισκέψεων κατανέμεται καλύτερα στους διαθέσιμους κόμβους. Ανάλογα αποτελέσματα επιτυγχάνονται όταν κλιμακώνεται το δίκτυο ή το εύρος και ο αριθμός των αναζητήσεων.

4.5 Συμπεράσματα

Η υποστήριξη πράξεων αναζήτησης εύρους διευθύνσεων σε δομημένα ΔΟΚ, απαιτεί αφενός τη κατάλληλη μέθοδο τοποθέτησης των δεδομένων, αφετέρου δε έναν ειδικό μηχανισμό εκτέλεσης, ο οποίος θα επιφέρει τον ελάχιστο δυνατό υπολογιστικό και διαχειριστικό φόρτο στην υποδομή. Το προτεινόμενο πρωτόκολλο επιτυγχάνει την εφαρμογή τεχνικών πολυδιάστατης αναζήτησης στα πλαίσια ενός ΚΠΚ, λαμβάνοντας, παράλληλα, υπόψη τις εσωτερικές του διαδικασίες αντιγραφής τιμών. Χρησιμοποιώντας ως παράδειγμα το δυαδικό δέντρο του Kademlia, δείχνουμε πως οι κόμβοι μπορούν να ομαδοποιηθούν σε γειτονιές αντιγράφων, εκμεταλλευόμενοι ήδη διαθέσιμες πληροφορίες από τους πίνακες δρομολόγησής τους. Στη συνέχεια, κάθε αναζήτηση μπορεί να ακολουθήσει ένα μονοπάτι από γειτονιά σε γειτονιά, αποφεύγοντας αυτόματα τις διπλές απαντήσεις και εξασφαλίζοντας την επιτυχία του αποτελέσματος ανεξαρτήτως της δυναμικότητας του δικτύου.

Εξ άλλου, οι διαδικασίες προώθησης της αίτησης, υπολογισμού των δυνατών απαντήσεων και συλλογής τους, εφόσον πραγματοποιούνται στα πλαίσια ενός ΚΠΚ, θα πρέπει να μην περιορίζουν τα ειδικά χαρακτηριστικά κλιμάκωσης και ανοχής σε σφάλματα του συστήματος. Ο στόχος είναι να χρειάζεται η επικοινωνία με ένα μικρό υποσύνολο κόμβων – μόνο όσων είναι τελικά απολύτως απαραίτητοι για την ολοκλήρωση του αιτήματος, σε μια πλήρως κατανεμημένη διαδικασία που να μπορεί να αναγνωρίζει και να αποφεύγει αυτόματα οποιεσδήποτε δυσλειτουργίες τόσο των ίδιων των συμμετεχόντων όσο και του δικτύου διασύνδεσής τους. Ο αλγόριθμος ακολουθεί τις ίδιες αρχές και χρησιμοποιεί τις ίδιες τεχνικές που επιτρέπουν στις απλές πράξεις εντοπισμού να έχουν τα παραπάνω χαρακτηριστικά. Με το να εξαρτάται η εκτέλεση μόνο από

τις πληροφορίες των πινάκων δρομολόγησης, η ίδια η διαδικασία ρυθμίζεται αυτόματα όταν μεταβάλλεται η επικάλυψη ή οι συνθήκες κάτω από τις οποίες λειτουργεί.

Η λεπτομερής ανάλυση της επίδρασης κάθε μεμονωμένης μεταβλητής του συστήματος, οδηγεί στην πλήρη κατανόηση της λειτουργικότητας που προσφέρεται και επιτρέπει την πρόβλεψη των επιδόσεων. Έτσι, είναι δυνατή η εκ των προτέρων ρύθμιση των χαρακτηριστικών της επικάλυψης, ανάλογα με τα διαθέσιμα δεδομένα ή την αναμενόμενη χρήση του δικτύου. Ο αριθμός των γειτονιών E, που παράγονται σε ένα σχηματισμό Kademlia, μεγέθους N και με παράμετρο κ μοντελοποιήθηκε θεωρητικά, αλλά επαληθεύτηκε και μέσω πειραματικών αποτελεσμάτων. Παρουσιάστηκε, επίσης, πώς ο αριθμός των διαστάσεων που χρησιμοποιούνται για τον χαρακτηρισμό των δεδομένων επηρεάζει τη κλίση της γραφικής παράστασης που συνδέει το εύρος μιας αναζήτησης με τον αριθμό των βημάτων που απαιτεί η ολοκλήρωσή της – μέχρι του μέγιστου E. Αν και η κατανομή των δεδομένων δεν μεταβάλει με κανέναν τρόπο τον αλγόριθμο, μη κανονικές κατανομές μπορεί να απαιτούν μεθόδους κατανομής του φόρτου στο σύνολο των διαθέσιμων κόμβων. Για τον σκοπό αυτό, εξετάσαμε έναν μηχανισμό εκμετάλλευσης των γειτονιών και προς αυτή τη κατεύθυνση.

Πολλές είναι οι εφαρμογές που θα μπορούσαν να αξιοποιήσουν μια τέτοια υποδομή αποθήκευσης και ανάκτησης δεδομένων. Ένα ενδεικτικό σενάριο χρήσης παρουσιάζεται στο κεφάλαιο 5.

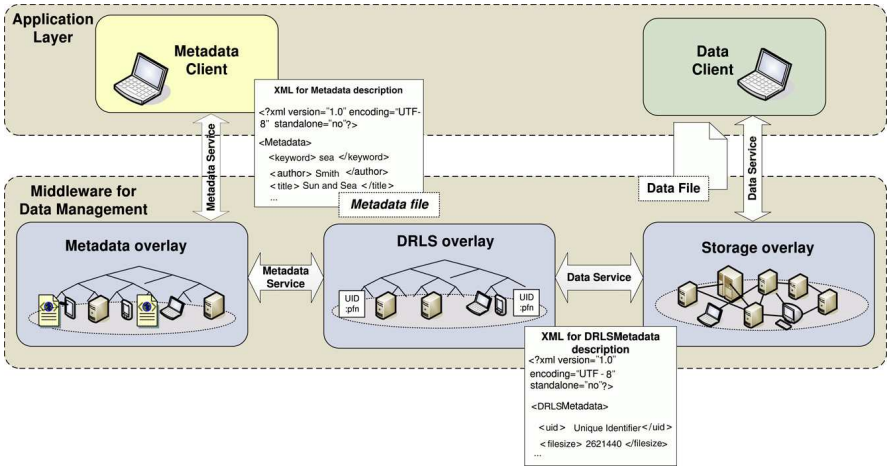
Κεφάλαιο 5

Σενάρια χρήσης

5.1 Το λογισμικό διαχείρισης δεδομένων του Gredia

Για να παρουσιάσουμε ένα πρακτικό παράδειγμα χρήσης των προτεινόμενων ΚΠΚ, θα αναφερθούμε στο ευρωπαϊκό ερευνητικό πρόγραμμα Gredia, μέρος του "IST 6th Framework Programme (FP6-34363)". Στόχος του Gredia είναι η δημιουργία μιας υποδομής Πλέγματος, η οποία, με το κατάλληλο λογισμικό, θα επιτρέπει σε δημιουργούς, επαγγελματίες και απλούς χρήστες να μοιράζονται και να ανταλλάσσουν πολυμέσα. Αν και είναι ερευνητικό πρόγραμμα, τα σενάρια χρήσης που υλοποιούνται βασίζονται σε ρεαλιστικές ανάγκες των εταιρικών συνεταίρων (Deutsche Welle, Popso, κτλ.). Ένα παράδειγμα είναι το εξής: Διάφορα γραφεία τύπου έχουν δημιουργήσει ένα κοινό χώρο δεδομένων στο Πλέγμα, όπου οι εργαζόμενοί τους (δημοσιογράφοι, φωτογράφοι και συντάκτες) ή οι πολίτες μπορούν να αποθηκεύσουν, αναζητήσουν ή μεταφορτώσουν νέα στη μορφή άρθρων, εικόνων και video, ανάλογα με τα δικαιώματα πρόσβασης του καθενός. Έτσι, υπάρχει η υποδομή που μπορεί, με το που θα συμβεί ένα σημαντικό γεγονός (π.χ. ακραία καιρικά φαινόμενα, σεισμοί, διαδηλώσεις, κτλ.), να αποθηκεύσει και να διαθέσει το περιεχόμενο που παράγεται επί τόπου από τους δημοσιογράφους ή ακόμα και τους περαστικούς. Η πληροφορία αυτή είναι άμεσα διαθέσιμη σε όλη την υφήλιο, οπότε άλλοι δημοσιογράφοι – από όλο τον κόσμο – θα μπορούν εύκολα να βρουν και να χρησιμοποιήσουν ό,τι χρειάζονται. Η πλατφόρμα του Gredia προσφέρει τα μέσα διαχείρισης των δεδομένων και οργανώνει τις διαδικασίες σύνταξης, αξιολόγησης και έκδοσης που οδηγούν στην τελική ανακοίνωση ενός νέου.

Στο σχήμα 5.1 παρουσιάζεται η αρχιτεκτονική του λογισμικού διαχείρισης δεδομένων του Gredia. Αποτελείται από 3 επικαλύψεις: Η επικάλυψη μεταδεδομένων αποτελεί το ευρετήριο αναζήτησης των αρχείων πολυμέσων, τα οποία αντιστοιχίζονται μέσω της δεύτερης επικάλυψης σε πραγματικά κατανεμημένα αντίγραφα που αποθηκεύονται στην τρίτη. Στην επικάλυψη μεταδεδομένων τοποθετούνται δείκτες στα διαθέσιμα αρχεία, οργανωμένοι με βάση τα αναγνωριστικά τους, στους οποίους και εκτελούνται οι αναζητήσεις. Στην επικάλυψη αντιστοίχισης αρχείων-αντιγράφων (επικάλυψη αντιστοίχισης ονομάτων) τηρείται ένα δεύτερο επίπεδο δεικτοδότησης, αφού για κάθε αρχείο σε επίπεδο ονοματολογίας μπορεί να υπάρχουν πολλαπλά φυσικά αντί-



Σχήμα 5.1: *Επισκόπηση του λογισμικού διαχείρισης δεδομένων του Gredia.*

γραφα στο δίκτυο – μια αντιστοιχία που αλλάζει συνεχώς. Στην επικάλυψη αποθήκευσης, οργανώνονται οι κόμβοι που έχουν τον πραγματικό αποθηκευτικό χώρο, που φιλοξενεί τον όγκο των δεδομένων. Οι κόμβοι αυτοί, επί των αρχείων που αποθηκεύουν, προσφέρουν και κατανεμημένες υπηρεσίες μεταφοράς δεδομένων. Στη πράξη, ένας κόμβος μπορεί να συμμετέχει σε παραπάνω από μία επικαλύψεις, ανάλογα με τις δυνατότητες και τις προθέσεις του. Ο διαχωρισμός σε επίπεδο ρόλων γίνεται επειδή για κάθε περίπτωση απαιτούνται διαφορετικοί αλγόριθμοι και πρωτόκολλα, οπότε και τα αντίστοιχα ΔΟΚ εξετάζονται και υλοποιούνται ανεξάρτητα.

Στη συνέχεια θα αναφερθούμε στις λεπτομέρειες υλοποίησης και των τριών αυτών επικαλύψεων. Στην επικάλυψη μεταδεδομένων, εφαρμόζονται οι λειτουργίες του ΚΠΚ με δυνατότητα αναζήτησης εύρους τιμών (κεφάλαιο 4). Οι δείκτες που αποθηκεύονται, αποτελούν πολυδιάστατα δεδομένα, τα οποία οι χρήστες του συστήματος μπορούν να ζητούν, είτε μεμονωμένα, είτε κατά ομάδες – επιλέγοντας το εύρος τιμών που απαιτείται σε κάθε διάσταση. Για την επικάλυψη αντιστοίχισης ονομάτων χρησιμοποιείται το πρωτόκολλο XOROS, που επιτρέπει την ανανέωση των τηρούμενων δεικτών (κεφάλαιο 3). Στην επικάλυψη αποθήκευσης, τις υπηρεσίες κατανεμημένης μεταφοράς αναλαμβάνει το GridTorrent, ένα πρωτόκολλο βασισμένο στις αρχές του BitTorrent, αλλά με τις κατάλληλες αλλαγές και επεκτάσεις προκειμένου να υποστηρίζονται λειτουργίες βασισμένες σε υπηρεσίες Πλέγματος.

5.1.1 Υλοποίηση της επικάλυψης μεταδεδομένων

Στο επίπεδο διαχείρισης δεδομένων, θεωρούμε δύο βασικές οντότητες για κάθε μονάδα πληροφορίας: Το *αρχείο*, που περιέχει την πληροφορία, και τα *μεταδεδομένα* – πολυδιάστατοι δείκτες, με τους οποίους αντιστοιχίζονται οι τιμές συγκεκριμένων αναγνωριστικών σε ένα αρχείο. Τα μεταδεδομένα δημιουργούνται, μεταφέρονται και αποθηκεύονται σε μορφή XML. Ένα τμήμα του λογισμικού του Gredia υλοποιεί την διεπαφή μεταξύ εφαρμογής και επικάλυψης, οπότε χειρίζεται όλες τις λειτουργίες τοποθέτησης, αναζήτησης και ανάκτησης από τη πλευρά της εφαρμογής. Εκεί κωδικοποιούνται οι τιμές των αναγνωριστικών σε δυαδικές αναπαραστάσεις, που

με τη σειρά τους μετατρέπονται σε διευθύνσεις από τον ΚΠΚ. Για την αποθήκευση των μεταδεδομένων, λοιπόν, εξάγονται οι τιμές των αναγνωριστικών από το αρχείο XML, γίνεται η κωδικοποίηση και έπειτα τοποθετείται το ίδιο το αρχείο στο δίκτυο με την αντίστοιχη διεύθυνση. Η λειτουργία αναζήτησης, εκτός από την κωδικοποίηση των τιμών που ζητούνται, με τον ίδιο τρόπο, μπορεί να απαιτεί και την αναδιάταξή τους, ώστε να ταιριάζει η σειρά των αναγνωριστικών με αυτή που χρησιμοποιείται στο επίπεδο του ΚΠΚ.

Οι εφαρμογές που έχουν σχεδιαστεί στα πλαίσια του Gredia, χρησιμοποιούν τρεις βασικούς τύπους αναγνωριστικών: αλφαριθμητικά, ημερομηνίες και εκείνα που οι τιμές τους αντιστοιχούν σε κατηγορίες. Κάθε εφαρμογή ορίζει τον δικό της πολυδιάστατο χώρο, με πλήθος διαστάσεων ανάλογο με τον αριθμό των αναγνωριστικών που έχουν επιλεγεί ώστε να είναι σαφείς οι αναφορές στα δεδομένα. Το μέγεθος όμως του πολυδιάστατου χώρου δεν είναι εύκολο να οριστεί. Εφόσον ο ΚΠΚ απαιτεί ένα σταθερό πλήθος δυαδικών ψηφίων αναπαράστασης ανά διάσταση, πρέπει να μετατραπούν οι τιμές του κάθε αναγνωριστικού σε δυαδική μορφή. Σε κάποιες περιπτώσεις αυτό επιτυγχάνεται μέσω μιας απλής αντιστοίχισης – ειδικά όταν το σύνολο όλων των πιθανών τιμών είναι μικρό. Σε άλλες όμως, απαιτείται μια ειδική συνάρτηση μετατροπής που συμπιέζει το πεδίο τιμών του αναγνωριστικού σε έναν μικρότερο χώρο, ή αντίστροφα. Ο περιορισμός στον χώρο που μπορεί να χρησιμοποιηθεί εξαρτάται και από την ΚΠΧ, τη μορφή και τα χαρακτηριστικά της. Οι περισσότερες συναρτήσεις θεωρούν ότι το μήκος κάθε διάστασης είναι το ίδιο, οπότε πρέπει να επιλεγεί ένα μέγεθος ικανό να χωρέσει κάθε αναπαράσταση, ακόμα και αν αυτή είναι συμπιεσμένη.

Η κωδικοποίηση των αναγνωριστικών που αντιστοιχούν σε κατηγορίες είναι εύκολη, καθώς το πλήθος των πιθανών τιμών είναι ήδη γνωστό και η τιμή μπορεί άμεσα να μετατραπεί σε διεύθυνση. Αν και η αναζήτηση ενός εύρους τιμών σε κατηγορίες δεν ήταν απαραίτητη κατά τον σχεδιασμό του συστήματος, εκμεταλλευόμαστε την ταξινόμηση που δημιουργείται στο επίπεδο των διευθύνσεων και επιτρέπουμε την αναζήτηση μεταδεδομένων που ανήκουν σε πολλές κατηγορίες ταυτόχρονα. Σε αυτό συμβάλλει και το λογισμικό, που αναδιατυπώνει κατάλληλα τις εντολές αναζήτησης αυτού του τύπου, και βάζει τις ζητούμενες τιμές στη σειρά που χρησιμοποιείται από το δίκτυο. Έτσι, είναι πιθανό μια πράξη που αναφέρεται σε δεδομένα πολλών διαφορετικών κατηγοριών, να αντιστοιχεί σε μια ενιαία εντολή αναζήτησης εύρους τιμών όταν φτάσει στην επικάλυψη.

Για τους άλλους δύο τύπους, χρησιμοποιείται μια απλή κωδικοποίηση χαρακτήρων που παράγει δυαδική αναπαράσταση για το αλφαριθμητικό εισόδου – ακόμα και αν αυτό είναι ημερομηνία. Ωστόσο, ο αριθμός των χαρακτήρων που κωδικοποιούνται και τελικά αποτελούν τη διεύθυνση στην αντίστοιχη διάσταση, εξαρτάται από τον αριθμό των δυαδικών ψηφίων που είναι διαθέσιμα στο επίπεδο της επικάλυψης. Η συμπίεση σε αυτή τη περίπτωση γίνεται κόβοντας τους επιπλέον χαρακτήρες που δεν χωράνε στον χώρο. Όταν επιλέγεται το μέγεθος της κάθε διάστασης, ο στόχος είναι να μην χρησιμοποιούνται τεράστιες διευθύνσεις, ώστε αφενός να περιορίζονται οι απαιτούμενοι υπολογισμοί στις συναρτήσεις της καμπύλης, και αφετέρου οι αναζητήσεις να αφορούν πιο περιορισμένους χώρους και εν τέλει να καταλήγουν σε λιγότερες γειτονιές και κόμβους. Αν απαιτείται περαιτέρω φιλτράρισμα των αποτελεσμάτων, αυτό μπορεί να γίνει εύκολα

Πίνακας 5.1: *Αναγνωριστικά που χρησιμοποιούνται από την εφαρμογή διαχείρισης πολυμέσων*

Αναγνωριστικό	Τύπος	Παράδειγμα
author	String	Smith John
date created	Date	2008.02.01
date modified	Date	2009.05.01
type	Categorical	audio, text, video
category	Categorical	entertainment, business, politics, weather, travel, health

στον κόμβο που συλλέγει τα αποτελέσματα. Επίσης, με αυτή την κωδικοποίηση τα δεδομένα τοποθετούνται στο δίκτυο με αλφαβητική σειρά. Για τους τύπους ημερομηνίας ταξινομούνται πρώτα με τον χρόνο, μετά τον μήνα, την μέρα, την ώρα και τα λεπτά.

Ως παραδείγματα χρήσης του Gredia, έχουν υλοποιηθεί διάφορες βασικές εφαρμογές. Εδώ θα εξετάσουμε μια εφαρμογή που χειρίζεται αρχεία πολυμέσων. Κάθε τέτοιο αρχείο μπορεί να περιγραφεί μέσω αναγνωριστικών που αφορούν το περιεχόμενό του, τον τύπο του, τη κωδικοποίησή του, κτλ. Από όλα τα αναγνωριστικά επιλέχθηκαν αυτά που παρουσιάζονται στον πίνακα 5.1 ως τα πιο αντιπροσωπευτικά και ως αυτά που αναμένεται να χρησιμοποιούνται ως επί το πλείστον από τους χρήστες στις αντίστοιχες πράξεις αναζήτησης. Το σύνολο των αναγνωριστικών συνεχίζει να αποθηκεύεται στο αντίστοιχο αρχείο XML, ώστε η εφαρμογή να μπορεί να επιλέξει με μεγαλύτερη λεπτομέρεια τους κατάλληλους δείκτες, αν αυτό είναι απαραίτητο. Το ίδιο μπορεί να γίνει και από τους ίδιους τους κόμβους τους ΚΠΚ, αν μαζί με κάθε αναζήτηση εύρους διευθύνσεων αποστέλλεται και η αρχική αναζήτηση, η οποία και εκτελείται απευθείας στην τοπική βάση δεδομένων του κάθε συμμετέχοντα. Πρακτικά, αυτή η τεχνική, που χρησιμοποιεί το πιο αντιπροσωπευτικό υποσύνολο αναγνωριστικών για τη δρομολόγηση της αναζήτησης, και την αρχική αίτηση για τοπικό φιλτράρισμα, απαιτεί την υλοποίηση επιπλέον λογισμικού στο επίπεδο του κάθε κόμβου, αλλά επιτυγχάνει καλή κλιμάκωση στις διαστάσεις του προβλήματος μέσω πολλαπλής κατανομής του φόρτου στην επικάλυψη.

Στο παράδειγμα της εφαρμογής διαχείρισης πολυμέσων, μια σημειακή αναζήτηση ορίζεται όταν δίνονται συγκεκριμένες τιμές για όλα τα αναγνωριστικά του πίνακα 5.1. Τότε στο επίπεδο του ΚΠΚ εκτελείται μια μοναδική πράξη ανάκτησης. Οι αναζητήσεις εύρους διευθύνσεων όπως η "Επέστρεψε όλα τα μεταδεδομένα που αναφέρονται σε αρχεία μορφής video, κινηματογραφημένα από τον John Smith, που δημιουργήθηκαν από τις 2008.09.28 μέχρι τις 2008.10.25 και στη συνέχεια τροποποιήθηκαν από τις 2008.09.29 μέχρι τις 2008.10.30 και αναφέρονται στον καιρό", δρομολογούνται αυτόματα μέσω του πρωτοκόλλου. Επιπρόσθετα, ο χρήστης μπορεί να ζητήσει όλες τις τιμές ενός αναγνωριστικού δίνοντας τον χαρακτήρα '*' στο αντίστοιχο πεδίο. Το λογισμικό που επικοινωνεί με τον ΚΠΚ, μετατρέπει τον συγκεκριμένο χαρακτήρα στο κατάλληλο εύρος διευθύνσεων.

Η υλοποίηση του ΚΠΚ και του σχετικού πρωτοκόλλου αναζήτησης έχει γίνει σε γλώσσα Python και ενσωματώνεται στη πλατφόρμα PeerPlatform. Το υπόλοιπο λογισμικό του Gredia, η πλειοψηφία του οποίου υλοποιείται σε γλώσσα Java, επικοινωνεί με τις επικαλύψεις μέσω ενός ανοιχτού πρωτοκόλλου που βασίζεται στο μοντέλο ανταλλαγής μηνυμάτων ειδοποιήσεων του PeerPlatform.

Για ακόμα πιο εύκολη επικοινωνία με τον ΚΠΚ, έχει υλοποιηθεί και μια απλή διεπαφή που χρησιμοποιεί υπηρεσίες Παγκόσμιου Ιστού (web-services) για την εκτέλεση εντολών get και put στους τοπικούς κόμβους της πλατφόρμας.

5.1.2 Υλοποίηση της επικάλυψης αντιστοίχισης ονομάτων

Όπως αναφέρθηκε στο κεφάλαιο 3, ένα από τα πιο κρίσιμα υποσυστήματα για τη διαχείριση δεδομένων στο Πλέγμα είναι η Υπηρεσία Εντοπισμού Αντιγράφων (Replica Location Service). Το περιβάλλον του Πλέγματος επιβάλλει τα δεδομένα να είναι διασκορπισμένα σε παγκόσμια κλίμακα λόγω των περιορισμών σε αποθηκευτικό χώρο των διαφόρων αποθετηρίων δεδομένων, αλλά και λόγω της απόστασης σε αυτά – στο επίπεδο του δικτύου – από τους χρήστες. Σε πολλές περιπτώσεις, είναι σύνηθες να χρησιμοποιούνται τοπικά αντίγραφα για να ελαττωθεί ο χρόνος απόκρισης του δικτύου που προστίθεται κάθε φορά που υπάρχει μια λειτουργία απομακρυσμένης πρόσβασης. Στην ορολογία του Πλέγματος το τοπικό αντίγραφο με ιδιότητες ανάγνωσης μόνο (read-only local copies) ενός απομακρυσμένα αρχείου που βρίσκεται σε κάποιο αποθετήριο δεδομένων, ονομάζεται *replica* και η ΥΕΑ είναι υπεύθυνη για την διαχείριση αυτών των αντιγράφων. Για την επεξεργασία ενός αρχείου στο Πλέγμα, μια εργασία θα πρέπει πρώτα να επικοινωνήσει με την ΥΕΑ, ώστε να βρει όλες τις τοποθεσίες που υπάρχει το ζητούμενο αρχείο. Σε περίπτωση που το αρχείο υπάρχει τοπικά, τότε η εργασία μπορεί να χρησιμοποιήσει τις συνηθισμένες λειτουργίες αρχείων προκειμένου να το επεξεργαστεί. Σε περίπτωση που δεν υπάρχει τοπικά, πρέπει να ενεργοποιηθεί ο μηχανισμός μεταφοράς δεδομένων του Πλέγματος ώστε να δημιουργηθεί ένα τοπικό αντίγραφο και να ενημερωθεί για την νέα τοποθεσία η ΥΕΑ, πριν μπορεί να προχωρήσει η εργασία με την πρόσβαση στο αρχείο.

Σύγχρονες διανομές λογισμικού Πλέγματος όπως το Globus Toolkit περιλαμβάνουν υπηρεσίες διαχείρισης αντιγράφων (Replica Location and Management Services). Μάλιστα, οι μηχανισμοί που έχουν χρησιμοποιηθεί από την επιστημονική κοινότητα τα τελευταία χρόνια για τις υπηρεσίες αυτές έχουν εξελιχθεί σημαντικά. Το αρχικό σχέδιο μιας κεντρικής βάσης δεδομένων που είχε αποθηκευμένες πληροφορίες για τις τοποθεσίες όλων των αντιγράφων, παραμερίστηκε προς όφελος μιας κατανεμημένης προσέγγισης. Η πιο διαδεδομένη λύση που χρησιμοποιείται ευρέως στο Πλέγμα, γνωστή και ως Giggle Framework (GIGa-scale Global Location Engine) [CDF+02, CPB+04], συντάσσει ένα ενιαίο χώρο ονοματολογίας αρχείων με μοναδικό αναγνωριστικό ανά Εικονικό Οργανισμό (Virtual Organization) το λογικό όνομα αρχείου (Logical Filename – LFN), το οποίο συνδέει με την φυσική τοποθεσία όλων των αντιγράφων του αρχείου (Physical Filenames – PFN). Τα LFN χρησιμοποιούνται από τις εφαρμογές για την εύρεση δεδομένων, χωρίς να χρειάζεται ιδιαίτερη γνώση για την φυσική τοποθεσία των δεδομένων. Τα PFN χρησιμοποιούνται από τις υπηρεσίες του Πλέγματος, όπως η ΥΕΑ και είναι δομημένα με τρόπο παρόμοιο με τις διευθύνσεις ιστοσελίδων (URL). Στο PFN αναφέρεται το πρωτόκολλο πρόσβασης στα δεδομένα που πρέπει να χρησιμοποιηθεί, η διεύθυνση του αποθετηρίου που είναι αποθηκευμένα καθώς και η ακριβής τοποθεσία (path) μέσα στο κόμβο. Για την κατανομή της πληροφορίας της ΥΕΑ, το Giggle Framework ορίζει δύο επίπεδα:

- Το πρώτο επίπεδο υπάρχει ο τοπικός κατάλογος αντιγράφων (Local Replica Catalog – LRC). Το LRC διατηρεί πληροφορία σχετική με λογικά ονόματα αρχείων, όπως λίστες εξουσιοδοτημένων ατόμων ή ομάδων και το είδος πρόσβασής τους στα δεδομένα, χρόνος/ώρα δημιουργίας του αρχείου κτλ. Επίσης, διατηρεί μια λίστα για κάθε λογικό όνομα αρχείου που περιέχει τις τοποθεσίες όλων των αντιγράφων που υπάρχουν για αυτό και πιο συγκεκριμένα το URL τους. Επιγραμματικά, δοθέντος ενός LFN, το LRC επιστρέφει τη λίστα με τα PFN που αντιστοιχούν σε αυτό.

- Στο δεύτερο επίπεδο και πιο υψηλό στην ιεραρχία της ΥΕΑ, υπάρχει το ευρετήριο τοποθεσίας των αντιγράφων (Replica Location Index – RLI). Στο ευρετήριο αυτό φυλάσσονται πληροφορίες για τους διαθέσιμους τοπικούς καταλόγους και τα αντίστοιχα λογικά ονόματα αρχείου. Επιγραμματικά, δοθέντος ενός LFN, το RLI γνωρίζει ποιος τοπικός κατάλογος διαθέτει πληροφορία σχετικά με τα αντίγραφα του συγκεκριμένου LFN.

Σε ένα συνηθισμένο σενάριο υλοποίησης, κάθε συμμετέχων σε έναν ΕΟ (VO) διαχειρίζεται ένα LRC, ενώ για την συνολική διαχείριση της υπηρεσίας υπάρχει ένα κεντρικό RLI για κάθε ΕΟ. Όταν οι προδιαγραφές το απαιτήσουν, μπορούν να συνυπάρξουν πολλαπλά RLI σε παράλληλη συνδεσμολογία, προσδίδοντας μηχανισμούς εξισορρόπησης φορτίου και ανάνηψης από βλάβη στην συνολική υπηρεσία. Το Giggle Framework επιτρέπει, επίσης, ένα κατάλογο να συνδεθεί με πολλαπλά ευρετήρια και αντίστροφα. Τέλος, τα πολλαπλά ευρετήρια μπορούν να συνδυαστούν σε μια δενδρική ιεραρχία. Περισσότερες πληροφορίες για τον σχεδιασμό του συστήματος και ένα σύντομο ιστορικό των διάφορων υλοποιήσεων ΥΕΑ αναφέρονται στην εργασία [CZKT06].

Παρόλο που η επεκτασιμότητα σε δίκτυα παγκόσμιας κλίμακας ήταν από τις κύριες προδιαγραφές του Giggle κατά τον σχεδιασμό του, παραμένει ανοιχτό ζήτημα το κατά πόσον η συγκεκριμένη αρχιτεκτονική θα συνεχίσει να λειτουργεί το ίδιο αποδοτικά όταν το πλήθος των αποθηκευμένων δεδομένων, των καταλόγων και των ευρετηρίων αυξηθούν κατά μερικές τάξεις μεγέθους. Το Giggle έχει σχεδιαστεί και δοκιμαστεί σε Πλέγματα υψηλής απόδοσης και δεν έχει δοκιμαστεί σε συνθήκες υποδομών παγκόσμιας κλίμακας άνευ υψηλής διαθεσιμότητας πόρων. Επίσης, οι πολύπλοκοι μηχανισμοί διάδοσης των αλλαγών που υλοποιεί το Giggle, ενδέχεται να αποτελέσουν πλήγμα στην αποδοτικότητα της ΥΕΑ σε συνθήκες δικτύων μεγάλης κλίμακας, όπου υπάρχει αυξημένη καθυστέρηση και χαμηλότεροι ρυθμοί μετάδοσης δεδομένων. Ενώ δεν υπάρχουν αποτελέσματα απόδοσης για το Giggle σε συνθήκες παραγωγικής υπηρεσίας παγκόσμιας κλίμακας, με εκατομμύρια και δισεκατομμύρια αποθηκευμένα δεδομένα λογικών/φυσικών αρχείων και κάτι τέτοιο είναι πρακτικά αδύνατο να δοκιμαστεί, οι σχεδιαστικές επιλογές που ακολουθήθηκαν θέτουν όρια στην επεκτασιμότητα του.

Για την υλοποίηση της επικάλυψης αντιστοίχισης ονομάτων στο λογισμικό διαχείρισης δεδομένων του Gredia, χρησιμοποιήθηκαν τεχνικές από το πεδίο των ΔΟΚ για την αποδοτικότερη μεταφορά δεδομένων στο Πλέγμα, με κύριο στόχο την απαλοιφή κάθε κεντρικού σημείου βλάβης. Καταργείται πρακτικά, λοιπόν, η ιεραρχική δομή ευρετηρίων και καταλόγων και τη θέση της παίρνει μια επικάλυψη XOROS, ενώ το συνολικό σύστημα ονομάζεται *Κατανεμημένη ΥΕΑ* (DRLS – Distributed RLS). Τα ζεύγη κλειδιού-τιμής που αποθηκεύονται, αντιστοιχούν το λογικό όνομα

αρχείου (LFN) με ένα σύνολο μεταδεδομένων. Τα μεταδεδομένα αυτά μπορεί να χωριστούν σε δυο κατηγορίες:

- στατική πληροφορία που αφορά μη μεταβλητά στοιχεία του αρχείου όπως το όνομα και το μέγεθός του
- δυναμική πληροφορία που αφορά σε συνεχή μεταβλητά στοιχεία για ένα αρχείο, όπως οι τοποθεσίες αποθήκευσής του (PFN)

Όσον αφορά την δυναμική πληροφορία σχετικά με τις τοποθεσίες αποθήκευσης ενός αρχείου, αυτή ελέγχεται περιοδικά για αλλαγές κατά την μεταφορά ενός αρχείου. Ουσιαστικά, πρόκειται για μια λίστα με φυσικά ονόματα αρχείων, που έχουν την ακόλουθη δομή: `protocol://fqdn:port/path/to/file`, όπου το `protocol` είναι το πρωτόκολλο που χρησιμοποιείται για την μεταφορά δεδομένων. Τη συγκεκριμένη στιγμή υποστηρίζονται τα πρωτόκολλα `gsiftp` (GridFTP) και `gtp` (GridTorrent) (αναφέρεται στη συνέχεια του κεφαλαίου). Το fully qualified domain name `fqdn` είναι το όνομα που ειναι καταχωρημένο στην υπηρεσία ονοματοδοσίας του Internet (DNS) για τον κάθε κόμβο και ακολουθεί η πλήρης διαδρομή στο σημείο του τοπικού δίσκου που έχει αποθηκευτεί το αρχείο.

Ένα πλεονέκτημα του DRLS είναι η υλοποίησή του με τέτοιο τρόπο, ώστε να διατηρείται η συμβατότητα με τις υπάρχουσες τεχνολογίες και την αρχιτεκτονική του Πλέγματος. Για παράδειγμα, υποστηρίζονται τα ίδια πρωτόκολλα επικοινωνίας. Τα επιπλέον δεδομένα που αποθηκεύονται στην Κατανεμημένη ΥΕΑ δεν λαμβάνονται υπόψη από τις υπάρχουσες εφαρμογές. Εάν μια εφαρμογή δεν αναγνωρίσει το πρόθεμα του GridTorrent, απλά θα το αγνοήσει. Επομένως, οι προτεινόμενες αλλαγές στην αρχιτεκτονική του Πλέγματος, όχι μόνο το βελτιώνουν, αλλά ενσωματώνονται στο υφιστάμενο περιβάλλον χωρίς επιπτώσεις στις ήδη υλοποιημένες υπηρεσίες και εφαρμογές. Παράλληλα, η Κατανεμημένη ΥΕΑ αντιμετωπίζει μερικές από τις αδυναμίες του υφιστάμενου μοντέλου, όπως της ανεπάρκειάς του να αντεπεξέλθει σε φορτία όπου υπάρχει απότομη και δυσανάλογη αύξηση της ζήτησης για δημοφιλή δεδομένα και για περιορισμένο χρόνο (flash crowds).

5.1.3 Υλοποίηση της επικάλυψης αποθήκευσης και μεταφοράς

Ένα βασικό κομμάτι στην συνολική αρχιτεκτονική του Πλέγματος είναι η μεταφορά δεδομένων. Σαν βασική υπηρεσία μεταφοράς αρχείων στο Πλέγμα, έχει επικρατήσει το δημοφιλές πρωτόκολλο μεταφοράς GridFTP [ABB+02] [ABK+05], το οποίο έχει οριστεί από το Open Grid Forum και περιλαμβάνεται στις περισσότερες σύγχρονες διανομές λογισμικού Πλέγματος, όπως το Globus Toolkit. Πρόκειται για μια επέκταση του πρωτοκόλλου File Transfer Protocol (FTP) με βελτιώσεις όπως οι δυνατότητες επικοινωνίας με την Υποδομή Ασφάλειας Πλέγματος (Grid Security Infrastructure – GSI) και διαχείρισης μεταφοράς δεδομένων από τρίτο συμμετέχοντα. Το τελευταίο επιτρέπει σε έναν χρήστη ή μια εφαρμογή να ξεκινήσει, παρακολουθήσει και να ελέγξει μια μεταφορά αρχείων μεταξύ δύο άλλων αποθετηρίων, που έχουν το ρόλο της πηγής

και του προορισμού αντίστοιχα. Αυτό επιτυγχάνεται με το διαχωρισμό του καναλιού ελέγχου από το κανάλι δεδομένων. Μια επιπρόσθετη επέκταση είναι η υποστήριξη για χειροκίνητη και αυτόματη ρύθμιση των μεγεθών του TCP buffer, ώστε να βελτιστοποιηθεί η μεταφορά μεγάλων αρχείων ή μεγάλων ομάδων από μικρότερα αρχεία. Σημαντικές βελτιώσεις, επίσης, είναι και η υποστήριξη μεταφορών μερικών κομματιών από ένα αρχείο, μεταφορών από N σε M κόμβους και παράλληλα TCP streams μεταξύ δύο κόμβων ή και περισσότερων.

Παρόλα αυτά οι τεχνικές που χρησιμοποιούνται στο GridFTP είναι βασισμένες στο μοντέλο του πελάτη-εξυπηρετητή επιφέροντας και τις παρενέργειες και τα ανεπιθύμητα χαρακτηριστικά του, όπως υπερφόρτωση του κεντρικού εξυπηρετητή, της ύπαρξης μοναδικού σημείου βλάβης και της αδυναμίας να αντεπεξέλθει σε flash crowds. Για την βελτιστοποίηση των μεταφορών δεδομένων στο Πλέγμα μπορούν να χρησιμοποιηθούν ειδικοί αλγόριθμοι, από τον χώρο των ΔΟΚ, ώστε η μεταφορά δεδομένων να εκμεταλλεύεται τα πολλαπλά αντίγραφα του κάθε αρχείου και να βελτιστοποιεί τη ολική χρήση του δικτύου. Οι τοποθεσίες των αντιγράφων κάθε αρχείου καταγράφονται, εξάλλου στην ΥΕΑ.

Στα πλαίσια του Gredia, δημιουργήσαμε το GridTorrent [ZDCK07], το οποίο εκμεταλλεύεται τα αντίγραφα ενός αρχείου και είτε χρησιμοποιεί το πρωτόκολλο GridFTP απευθείας, είτε – σε περίπτωση που υπάρχουν και άλλοι GridTorrent κόμβοι ενεργοί την ίδια στιγμή – συνεργάζεται μαζί τους για την επίσπευση της μεταφοράς. Το GridTorrent λαμβάνει τις θέσεις των αντιγράφων από την απλή ή την Κατανεμημένη ΥΕΑ, που περιγράφηκε παραπάνω. Επιπλέον, επικοινωνεί με την Υποδομή Ασφαλείας Πλέγματος (GSI), ενώ διατηρεί ένα ξεχωριστό κανάλι επικοινωνίας, στο οποίο μπορεί να λάβει εντολές. Στη συνέχεια δίνονται συνοπτικές λεπτομέρειες αυτών των χαρακτηριστικών του πρωτοκόλλου.

Ο αλγόριθμος μεταφοράς είναι βασισμένος στο BitTorrent, και επιτρέπει από κάθε κόμβο να κατεβάζει τοπικά ένα αρχείο από πολλές πηγές, ενώ ταυτόχρονα ανεβάζει το ίδιο αρχειο σε άλλους κόμβους. Χρησιμοποιώντας την ορολογία του BitTorrent, το GridTorrent δημιουργεί μια ομάδα κόμβων (swarm) όπου υπάρχουν οι κόμβοι ή οι χρήστες που ζητάνε το αρχείο ή το έχουν ημιτελές (leechers) και οι κόμβοι ή χρήστες (seeds) που έχουν ολόκληρο το αρχείο και το διαμοιράζουν στους πρώτους. Η συνεργατική φύση του αλγορίθμου, διασφαλίζει την μέγιστη χρησιμοποίηση του διαθέσιμου εύρους της δικτυακής σύνδεσης, ενώ ταυτόχρονα ο μηχανισμός tit-for-tat προσφέρει επεκτασιμότητα σε συνθήκες υψηλού φορτίου. Ειδικότερα, σε μια υπάρχουσα υποδομή Πλέγματος, οι υφιστάμενοι αποθηκευτικοί χώροι και εξυπηρετητές GridFTP μπορούν να χρησιμοποιηθούν σαν seeds και οι κόμβοι GridTorrent να μεταφέρουν από αυτούς αρχεία με την επιλογή του GridFTP για μερική μεταφορά αρχείου. Το αρχείο *.torrent* που χρησιμοποιείται στο BitTorrent αντικαθίσταται από την ΥΕΑ. Για να ξεκινήσει το κατέβασμα ενός αρχείου, πρέπει να υπάρχει το μοναδικό αναγνωριστικό του αρχείου (unique identifier - *UID*), το οποίο είναι συνήθως το αποτέλεσμα μιας κρυπτογραφικής συνάρτησης με είσοδο τα περιεχόμενα του αρχείου. Οι υπόλοιπες. απαραίτητες πληροφορίες μπορούν να αποκτηθούν από την ΥΕΑ χρησιμοποιώντας το UID. Οι κόμβοι που συμμετέχουν σε ένα GridTorrent swarm καταγράφονται στην ΥΕΑ, ώστε να μπορούν να βρεθούν μεταξύ τους και να συνεργαστούν.

Σε ένα Πλέγμα, μόνο ορισμένοι χρήστες είναι έμπιστοι για την μεταφορά κομματιών ή ενός ολόκληρου αρχείου. Επίσης, η κρυπτογράφηση προσφέρεται για την μεταφορά ευαίσθητων δεδομένων. Για την εγγύηση των προδιαγραφών ασφαλείας, το GridTorrent, χρησιμοποιεί την υλοποίηση της Υποδομής Ασφάλειας Πλέγματος (GSI) από το Globus Toolkit. Πιο συγκεκριμένα, το GridTorrent αναπτύσσει όλους τους πρότυπους μηχανισμούς της υποδομής, δηλαδή ταυτοποίηση, ακεραιότητα και κρυπτογραφία. Κάθε φορά που χρησιμοποιείται ένα TCP socket, περιτυλίσσεται μαζί με τα πιστοποιητικά του χρήστη ή της υπηρεσίας σε ένα Grid Socket, σύμφωνα με τις μεθόδους που προσφέρει το GSS API. Έτσι, κάθε φορά που ένα κανάλι επικοινωνίας δημιουργείται, τηρούνται οι παράμετροι της κρυπτογραφίας, ακεραιότητας και ταυτοποίησης που έχουν τεθεί από τον χρήστη του κόμβου ή τον διαχειριστή.

Στο GridTorrent, οι κόμβοι επικοινωνούν μεταξύ τους και ανταλλάσσουν πληροφορίες σχετικά με την εξέλιξη της μεταφοράς δεδομένων. Ένα νέο χαρακτηριστικό του GridTorrent, είναι η δυνατότητα ένας κόμβος να στέλνει απομακρυσμένες εντολές σε έναν δεύτερο, μέσω ενός ξεχωριστού καναλιού επικοινωνίας, το κανάλι ελέγχου. Η δυνατότητα αυτή είναι παρόμοια με τη σχεδιαστική επιλογή του GridFTP, για χωριστά κανάλια μεταφοράς εντολών και δεδομένων και επιτρέπει εκτός των άλλων και οργάνωση μεταφοράς δεδομένων μεταξύ δύο κόμβων από ένα τρίτο (third party transfer). Με την δυνατότητα αυτή το BitTorrent ξεπερνά το μειονέκτημα που υφίσταται σε αρχιτεκτονικές τύπου torrent, όπου για να αρχίσει η μεταφορά δεδομένων για ένα αρχείο, πρέπει κάποιος να βρει αυτό το αρχείο και στην συνέχεια να ενδιαφερθεί, το οποίο είναι κοινή πρακτική σε δίκτυα ομότιμων κόμβων όπως το BitTorrent, αλλά διαφέρει αρκετά από το μοντέλο χρήσης που υπάρχει στο Πλέγμα.

Μέσω του GridTorrent, λοιπόν, επιτυγχάνεται και στο επίπεδο αποθήκευσης και μεταφοράς δεδομένων του Gredia, η πλήρης κατανομή των αντίστοιχων λειτουργιών στους διαθέσιμους κόμβους, με τεχνικές ΔΟΚ που προσφέρουν ιδιαίτερα χαρακτηριστικά κλιμάκωσης και ανοχής σε σφάλματα.

5.2 Κατανεμημένη μηχανή αναζήτησης

Τα πρωτόκολλα που προτείνονται, μπορούν να αποτελέσουν και τη βάση μιας κατανεμημένης μηχανής αναζήτησης. Δηλαδή, ενός κατανεμημένου αποθετηρίου δεικτών σε αρχεία, τα οποία χαρακτηρίζονται από συγκεκριμένες "λέξεις κλειδιά" (keywords). Σε ένα τέτοιο σύστημα χρησιμοποιούνται συνήθως ανάστροφα ευρετήρια – δομές που αντιστοιχούν τις "λέξεις κλειδιά" στα αρχεία που χαρακτηρίζονται από αυτές. Στο παράδειγμα του Πίνακα 5.2, τρία αρχεία `a.jpg`, `b.txt` και `c.jpg` με αντίστοιχα κλειδιά, παράγουν ένα ανάστροφο ευρετήριο κλειδιών. Για να βρεθούν τα αρχεία που περιέχουν ένα σύνολο κλειδιών, απαιτείται η προσπέλαση σε όλες τις σχετικές εγγραφές του ευρετηρίου.

Το ευρετήριο αυτό, εύκολα αποθηκεύεται σε έναν ΚΠΚ, αρκεί να οριστεί μια συνάρτηση κατακερματισμού των λέξεων, η οποία να παράγει ένα μοναδικό αριθμητικό κλειδί για κάθε διαφορετική λέξη. Ύστερα, μπορούμε να αποθηκεύουμε και να ανακτούμε τις εγγραφές του ευρετηρίου

(α′) Αρχεία

Αρχείο	**"Λέξεις Κλειδιά"**
a.jpg	Euro 2004 Greece
b.txt	Olympics 2004 Greece
c.jpg	Euro 2008 Spain

(β′) Ευρετήριο

"Λέξεις Κλειδιά"	**Δείκτες**
2004	a.jpg, b.txt
2008	c.jpg
Euro	a.jpg, c.jpg
Greece	a.jpg, b.txt
Olympics	b.txt
Spain	c.jpg

Πίνακας 5.2: *Ανάστροφο ευρετήριο αναζήτησης για τρία αρχεία με "λέξεις κλειδιά"*

μέσω των χαρακτηριστικών συναρτήσεων *put* και *get*. Αν χρησιμοποιηθεί όμως ένας "κλασσικός" ΚΠΚ, παρουσιάζονται δύο σημαντικά προβλήματα υλοποίησης:

- Δεδομένου ότι το σύνολο των αρχείων δεν είναι στατικό, αλλά μπορούν να προστίθενται και να αφαιρούνται αρχεία από τη συλλογή αναφοράς, απαιτούνται μεταβολές στις εγγραφές του ευρετηρίου.
- Η αναζήτηση παραπάνω μίας "λέξης κλειδί" μπορεί να γίνει μόνο αν είναι σαφώς ορισμένο το κλειδί. Δεν επιτρέπονται δηλαδή αναζητήσεις σε κλειδιά της μορφής "αρχίζει από Gr" ή "σχετικά με Greece".

Τα δύο αυτά προβλήματα μπορούν να λυθούν με τους αλγορίθμους που παρουσιάζονται στα κεφάλαια 3 και 4 αντίστοιχα. Χρησιμοποιώντας ως ΚΠΚ το XOROS, αντιμετωπίζεται το πρόβλημα των ενημερώσεων στα δεδομένα του ευρετηρίου. Κατά τη προσθήκη ενός αρχείου, μπορούν να προστίθενται και τα αντίστοιχα κλειδιά απευθείας στην σχετική εγγραφή του ευρετηρίου. Επιπλέον, σύμφωνα με τις προτεινόμενες τεχνικές αναζήτησης εύρους διευθύνσεων, μπορεί να αλλάξει η συνάρτηση κατακερματισμού και ο αλγόριθμος ανάκτησης (ή να προστεθεί μια επιπλέον εντολή ανάκτησης πέραν της *get* του XOROS), ώστε να επιτρέπονται συγκεντρωτικές ή σχετικές αναζητήσεις στα κλειδιά του ευρετηρίου. Η νέα συνάρτηση κατακερματισμού θα πρέπει να κωδικοποιεί τη σημασιολογία σχετικότητας των λέξεων που χρησιμοποιούνται ως κλειδιά στην εγγύτητα των παραγόμενων αριθμητικών κλειδιών, αν απαιτείται η αναζήτηση σχετικών όρων. Για την υλοποίηση μιας πιο απλής μορφής αναζήτησης, της μορφής "αρχίζει από …" μπορεί να εφαρμοστεί η λεξικογραφική κωδικοποίηση που περιγράφεται σε προηγούμενη παράγραφο του κεφαλαίου.

Μια απλούστερη μορφή αυτού του συστήματος χρησιμοποιήθηκε για κάποιες λειτουργίες της πλατφόρμας του Gredia. Χρησιμοποιήθηκε το XOROS για υποδομή αποθήκευσης, αλλά δεν επεκτάθηκε ο ΚΠΚ με τις προτεινόμενες λειτουργίες αναζήτησης. Στη συγκεκριμένη εκδοχή της μηχανής αναζήτησης "λέξεων κλειδιών", θεωρήθηκε ότι τα κλειδιά θα ορίζονται σαφώς από τον χρήστη. Για την ελαχιστοποίηση των απαιτήσεων σε επίπεδο δικτύου από την πολλαπλή, παράλληλη εκτέλεση των ερωτημάτων *get*, εφαρμόστηκε μια τεχνική "αλυσιδωτού φιλτραρίσματος"

[LRL04], όπου τα ερωτήματα εκτελούνται σειριακά και τα συνολικά αποτελέσματα μεταφέρονται από το ένα βήμα στο επόμενο, ώστε να διατηρηθούν στο τέλος μόνο οι κοινές εγγραφές.

Κεφάλαιο 6

Επίλογος

Η κατανεμημένη διαχείριση μεγάλων όγκων πληροφορίας σε υπολογιστικά δίκτυα παγκόσμιας κλίμακας, αποτελεί έναν τομέα συνεχούς ανάπτυξης. Ήδη οι σχεδιαστές αντίστοιχων συστημάτων έχουν καταλήξει στο συμπέρασμα ότι απαιτούνται οπωσδήποτε κατάλληλοι αλγόριθμοι που θα εκμεταλλεύονται όλο το φάσμα των διαθέσιμων πόρων, προκειμένου να είναι εφικτή η κλιμάκωση των υπηρεσιών στα μεγέθη που επιβάλλονται από τις σύγχρονες δικτυακές τεχνολογίες. Έτσι, τα συστήματα διαχείρισης και ανταλλαγής δεδομένων που χρησιμοποιούνται σήμερα, στηρίζονται όλο και περισσότερο στην έννοια της *ομότιμης συνεισφοράς*. Το μοντέλο του κεντρικού εξυπηρετητή που συντονίζει τις λειτουργίες μεταξύ ενός μεγάλου όγκου "πελατών" ή "χρηστών", δίνει τη θέση του στην έννοια της άναρχης και αυτόνομης υπηρεσίας, η προσφορά της οποίας συνεπάγεται αυτόματα και την αποδοχή ενός κοινού πρωτοκόλλου συνεργασίας για την ανάπτυξη και εξέλιξη του ίδιου του δικτύου. Στα *Δίκτυα Ομότιμων Κόμβων* η πληροφορία ταξιδεύει από τον ένα συμμετέχοντα στον άλλο. Οι λειτουργίες επιτελούνται από "πελάτη" σε "πελάτη" απευθείας — η ζήτηση δεδομένων συνεπάγεται και την εθελούσια προσφορά. Έτσι, ο συνολικά διαθέσιμος αποθηκευτικός χώρος ή η συνολική υπολογιστική ισχύς του συστήματος πολλαπλασιάζεται με τη προσθήκη νέων συσκευών ή μηχανημάτων στο *δίκτυο επικάλυψης* που δημιουργείται.

Η ιδέα της κλιμάκωσης των χαρακτηριστικών ενός υπολογιστικού συστήματος με τη συνεχή προσθήκη νέων κόμβων επεξεργασίας ή αποθήκευσης δεδομένων σε ένα κοινό δίκτυο διασύνδεσης, προφανώς δεν αποτελεί καινοτομία. Οι βασικές σχετικές μεθοδολογίες και αλγόριθμοι έχουν παρουσιαστεί και αναλυθεί πολλές δεκαετίες πριν, στα πλαίσια των κλάδων της επιστήμης που μελετούν τα *παράλληλα* και *κατανεμημένα συστήματα*. Ο σχεδιασμός των *Δικτύων Ομότιμων Κόμβων*, αρχικά στηρίχθηκε σε αυτές τις βάσεις και εξελίχθηκε παράλληλα, ώστε να δώσει λύσεις σε προβλήματα που παρουσιάστηκαν στις μέρες μας ως αποτέλεσμα των διαρκών τεχνολογικών εξελίξεων: Το δίκτυο διασύνδεσης είναι πλέον παγκόσμιο και οι κόμβοι υπεράριθμοι, μεγάλων και μικρών διαστάσεων – πολλαπλών ή περιορισμένων δυνατοτήτων. Σε αντίθεση με τις παραδοχές που γίνονται ως επί το πλείστον στα παράλληλα και κατανεμημένα συστήματα, σε ένα τέτοιο περιβάλλον δεν μπορεί κανείς να παρέχει εγγυήσεις όσον αφορά τον χρόνο σύνδεσης, τα ποιοτικά χαρακτηριστικά της γραμμής μεταφοράς δεδομένων, αλλά και του ίδιου του

κόμβου, τον τρόπο λειτουργίας του, κτλ. Έτσι, υπάρχει η ανάγκη ορισμού άλλων τεχνικών και αλγορίθμων, που να θεωρούν ως δεδομένα σενάρια συμπεριφοράς, που σε παραδοσιακά συστήματα χαρακτηρίζονται ως "ακραία".

Τα ΔΟΚ ορίζουν καταρχήν μια μεθοδολογία διάταξης κόμβων και άρα δρομολόγησης αιτημάτων και απαντήσεων, επί της οποίας βασίζονται αλγόριθμοι αποθήκευσης και ανάκτησης δεδομένων. Σκοπός της παρούσας διατριβής ήταν η μελέτη των ΔΟΚ που χρησιμοποιούνται σήμερα, καθώς και η πρόταση νέων, καινοτόμων τεχνικών για την καλύτερη και αποδοτικότερη αξιοποίηση των διαθέσιμων πόρων. Ερευνώντας τη σχετική βιβλιογραφία, διαπιστώθηκε ότι οι εφαρμογές που στηρίζονται στις συγκεκριμένες δομές περιορίζονται από τα σχετικά πρωτόκολλα αυτόματης αντιγραφής των δεδομένων στους συμμετέχοντες κόμβους. Η τήρηση αντιγράφων, που χρησιμοποιείται κυρίως σε *Κατανεμημένους Πίνακες Κατακερματισμού* (δομημένα ΔΟΚ), βοηθά στην ανεκτικότητα του συστήματος σε ένα περιβάλλον δυναμικών — και πολλές φορές αναξιόπιστων — κόμβων, τη διατήρηση των δεδομένων και την αποδοτικότερη διάθεσή τους. Από την άλλη μεριά όμως, η ύπαρξη αντιγράφων σημαίνει ότι απαιτούνται ειδικές διαδικασίες για την ενημέρωση τους, όταν αλλάζει η πρωτεύουσα πληροφορία. Επίσης, κάθε διαδικασία αναζήτησης ή απαρίθμησης των δεδομένων θα πρέπει να γνωρίζει πως να αποφεύγει τα αντίγραφα τους, είτε κατά τη προσπέλαση, είτε κατά τη παρουσίαση των τελικών αποτελεσμάτων.

Αναπτύξαμε και υλοποιήσαμε αλγορίθμους που δίνουν λύση σε αυτά τα προβλήματα. Για το πρόβλημα της ενημέρωσης δεδομένων, προτείνουμε δύο λύσεις, ανάλογα με τα χαρακτηριστικά της εφαρμογής. Για το πρόβλημα της απαρίθμησης, παρουσιάσαμε μια τεχνική με την οποία μπορεί να ανακτώνται όλα τα δεδομένα που ανήκουν σε ένα εύρος διευθύνσεων – τεχνική η οποία επεκτείνεται σε μεθοδολογία αναζήτησης πληροφοριών, όταν σε κάθε μία από αυτές αντιστοιχούν ένα ή περισσότερα αναγνωριστικά. Όλοι οι παραπάνω αλγόριθμοι υλοποιούνται συμπληρωματικά στον πυρήνα εκτέλεσης του εκάστοτε ΚΠΚ. Αυτό σημαίνει ότι δεν αλλάζουν τα βασικά χαρακτηριστικά του συστήματος, αλλά προστίθενται κάθε φορά οι νέες λειτουργίες. Μάλιστα, οι αλγόριθμοι αυτοί μπορούν να χρησιμοποιηθούν και συνδυαστικά. Δείξαμε κάποια αντιπροσωπευτικά σενάρια χρήσης στα πλαίσια μιας ολοκληρωμένης πλατφόρμας διαχείρισης δεδομένων και μεταδεδομένων για το Πλέγμα και ευελπιστούμε ότι ΚΠΚ με δυνατότητες ενημέρωσης και αναζήτησης θα αξιοποιηθούν μελλοντικά από ένα ευρύ φάσμα σχετικών εφαρμογών.

Σε αυτή τη κατεύθυνση προτείνεται μελλοντικά να χρησιμοποιηθεί η πλατφόρμα που έχει αναπτυχθεί για την υλοποίηση των σχετικών αλγορίθμων. Η πλατφόρμα αυτή μπορεί να βοηθήσει και στην ενδελεχή μελέτη άλλων πρωτοκόλλων ΚΠΚ και των χαρακτηριστικών τους. Η προσομοίωση και εκτέλεση μεγάλων δυναμικών ΔΟΚ – της τάξης των εκατοντάδων χιλιάδων συμμετεχόντων – όπως συνέβαλε στην ανάδειξη των προβλημάτων που παρουσιάσαμε και στον σχεδιασμό των αντίστοιχων λύσεων, περιμένουμε ότι θα επιδείξει νέες, σημαντικές μελλοντικές κατευθύνσεις έρευνας στον τομέα των αντίστοιχων πρωτοκόλλων. Μια τέτοια κατεύθυνση μπορεί για παράδειγμα να είναι η προσαρμογή των αλγορίθμων ώστε να διατηρείται η ανωνυμία των χρηστών του συστήματος.

Παράρτημα Α′

PeerPlatform

Η μελέτη της συμπεριφοράς ενός ΚΠΚ απαιτεί είτε την εγκατάσταση και λειτουργία ενός μεγάλου αριθμού κόμβων και την συλλογή στατιστικών στοιχείων σε πραγματικό χρόνο, είτε τη δημιουργία ενός περιβάλλοντος προσομοίωσης, μέσα στο οποίο όλο το δίκτυο λειτουργεί σε ένα ελεγχόμενο περιβάλλον και καταγράφονται οι αλλαγές των χαρακτηριστικών της επικάλυψης στον χρόνο. Επιπλέον, είναι πλέον απαραίτητη μια πλατφόρμα λογισμικού που να επιτρέπει την γρήγορη και εύκολη ανάπτυξη νέων πρωτοκόλλων, ενώ παράλληλα να προσφέρει τη δυνατότητα για αξιολόγηση του κάθε βήματος της διαδικασίας προγραμματισμού. Στις επόμενες παραγράφους παρουσιάζεται το PeerPlatform – ένα πλαίσιο που παρέχει τα κατάλληλα για την ανάπτυξη, αξιολόγηση και λειτουργία ΚΠΚ. Το PeerPlatform βοηθά στην ανάπτυξη του πρωτοκόλλου, διαθέτοντας μια ολοκληρωμένη βιβλιοθήκη λογισμικού με συναρτήσεις επικοινωνίας, διαχείρισης και παρακολούθησης. Υποστηρίζεται ένα ενιαίο μοντέλο ανάπτυξης, όπου ο ίδιος κώδικας μπορεί να χρησιμοποιηθεί διαφανώς, τόσο για την εκτέλεση του πρωτοκόλλου σε πραγματικά μηχανήματα, όσο και για την οργάνωση προσομοιώσεων μεγάλης κλίμακας σε επίπεδο πακέτων δικτύου. Μάλιστα, καθώς κάθε πλατφόρμα μπορεί να φιλοξενήσει πολλαπλούς κόμβους ΚΠΚ στο ίδιο μηχάνημα, υποστηρίζονται δύο μέθοδοι για την προσομοίωση των αντίστοιχων επικαλύψεων. Είτε μπορεί να εκτελείται όλη η επικάλυψη σε εικονικό χρόνο στο ίδιο μηχάνημα, είτε να παρακολουθείται ένα δίκτυο κόμβων σε πραγματικό χρόνο, που σε φυσικό επίπεδο κατανέμεται προς εκτέλεση σε μια συστοιχία μηχανημάτων – τοπικού χαρακτήρα ή με μέλη διάσπαρτα στο Διαδίκτυο.

Μια κοινή υποδομή τέτοιου τύπου επιτρέπει, επίσης, και την σύγκριση δύο ή περισσότερων πρωτοκόλλων, καθώς και την δημιουργία συγκεκριμένων μετρικών αξιολόγησης που θα αποτελούν κοινή βάση για την αποτίμηση των χαρακτηριστικών μιας επικάλυψης. Πολλές υλοποιήσεις ΚΠΚ συμπεριλαμβάνουν τον δικό τους κώδικα προσομοίωσης, με πιο γνωστό παράδειγμα αυτό του FreePastry [Fre]. Ωστόσο, η προσαρμογή διαφορετικών προσομοιωτών σε ένα κοινό περιβάλλον εκτέλεσης, δεν είναι μόνο εξαιρετικά δύσκολη, αλλά μπορεί τελικά να μην είναι εφικτή λόγω των εσωτερικών χαρακτηριστικών και των μεθόδων που χρησιμοποιεί η κάθε υλοποίηση. Ένα αντίστοιχο σύστημα έχει προταθεί από τους συγγραφείς του PlanetSim, αν και υπάρχουν σημαντικές διαφορές τόσο στην αρχιτεκτονική του λογισμικού, όσο και στη μέθοδο που ακολουθείται

για την λειτουργία των κόμβων. Το PlanetSim αποτελείται από επίπεδα λογισμικού και χρησιμοποιεί την Κοινή Διεπαφή για ΚΠΚ, που περιγράφεται στην [DZD+03]. Έτσι, κώδικας που έχει γραφτεί για τον προσομοιωτή, μπορεί τελικά να εκτελεστεί σε πραγματικές συνθήκες αν μεταγλωττιστεί εκ νέου με τις βιβλιοθήκες επικοινωνίας του FreePastry. Ένας ακόμα προσομοιωτής που αναφέρεται συχνά στη βιβλιογραφία είναι το p2psim [p2p].

Ένα ακόμη σύνολο σχετικών εργασιών αναφέρονται σε μεθόδους καταγραφής των πακέτων που ανταλλάσσονται στο δίκτυο σε πραγματικό χρόνο, μέσω εικονικών κόμβων, ή αλλιώς *πρακτόρων* (agents), που τοποθετούνται στην επικάλυψη και παρακολουθούν τους υπόλοιπους συμμετέχοντες εν αγνοία τους. Τέτοιου τύπου λύσεις μπορούν να βοηθήσουν στην εύρεση και αξιολόγηση προβλημάτων που παρουσιάζονται σε ήδη εγκατεστημένα δίκτυα. Από την άλλη μεριά, η συλλογή όλων των διαφορετικών δεδομένων και πληροφοριών από ένα "ζωντανό" δίκτυο είναι σχεδόν αδύνατη, ειδικά όταν δεν υποστηρίζεται εν μέρει από τους ίδιους τους κόμβους. Για παράδειγμα, μπορεί με αυτόν τον τρόπο να φαίνονται κάποιες από τις επαφές που διατηρεί ένας συμμετέχοντας στον πίνακα δρομολόγησής του, αλλά δεν υπάρχει τρόπος να γνωρίζουμε πόσες έχει συνολικά, πόσες φορές έχει αποτύχει η επικοινωνία και με ποιές από αυτές, κτλ. Υπάρχουν, επίσης, μελέτες που προσπαθούν να φτιάξουν μαθηματικά μοντέλα των ΚΠΚ, χρησιμοποιώντας πιθανοτικά εργαλεία, όπως οι αλυσίδες Markov [SP06]. Οι αντίστοιχες εργασίες δεν αντιτίθενται σε ένα πρακτικό μοντέλο αξιολόγησης, αλλά μπορούν να δράσουν συμπληρωματικά, αφού ο ταυτόχρονος πειραματισμός με ένα πρακτικό δίκτυο μπορεί να αναδείξει προβλήματα στο θεωρητικό μοντέλο ή, από την άλλη μεριά, να επικυρώσει τα σχετικά αποτελέσματα.

Στη συνέχεια, παρουσιάζεται συνοπτικά η αρχιτεκτονική της πλατφόρμας και αναλύονται οι διαδικασίες προσομοίωσης που μπορούν να χρησιμοποιηθούν. Ο ρόλος του PeerPlatform στην ανάπτυξη και τον έλεγχο του πρωτοκόλλου Kademlia ήταν καταλυτικός, αφού επέτρεψε την πρακτική μελέτη διαφόρων παραλλαγών που έχουν προταθεί στη σχετική βιβλιογραφία. Όπως φαίνεται στο κεφάλαιο Α΄.2, μια διαφοροποιημένη υλοποίηση των πινάκων δρομολόγησης έχει σαν αποτέλεσμα την δραματική μείωση των μηνυμάτων διαχείρισης, καθώς και την γενικότερα καλύτερη συμπεριφορά του συστήματος στις λειτουργίες ανάκτησης δεδομένων.

Α΄.1 Σχεδιασμός και υλοποίηση

Το PeerPlatform είναι γραμμένο σε γλώσσα προγραμματισμού Python και δομημένο ως ένα άρθρωμα (module) το οποίο μπορεί να εκτελεστεί είτε ανεξάρτητα, είτε ενσωματωμένο σε μια εφαρμογή. Η Python λειτουργεί με εικονική μηχανή και μεταφραστή, οπότε κάθε κώδικας γραμμένος σε Python μπορεί να εκτελεστεί σε οποιοδήποτε μηχάνημα διαθέτει τον αντίστοιχο μεταφραστή. Μάλιστα, η Python διαθέτει και τα κατάλληλα εργαλεία ώστε να είναι εύκολη η δημιουργία διεπαφών με προγράμματα που είναι γραμμένα σε άλλες γλώσσες, όπως η C ή η Java. Παρακάτω παρουσιάζονται οι εσωτερικοί μηχανισμοί της πλατφόρμας, ώστε να αποσαφηνιστεί ο τρόπος με τον οποίο υλοποιούνται τα πρωτόκολλα ΚΠΚ και εκτελούνται τα σχετικά πειράματα.

Α΄.1.1 Επισκόπηση

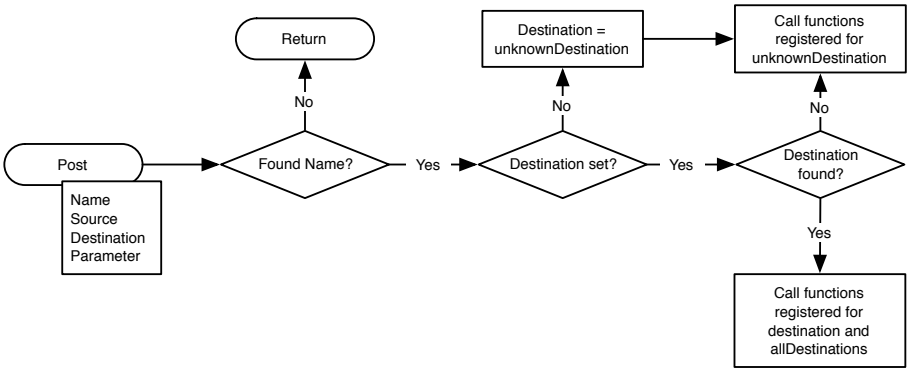
Το PeerPlatform, ακολουθεί εσωτερικά μια αρθρωτή αρχιτεκτονική. Αποτελείται από πέντε μόλις αρθρώματα, το πιο σημαντικό εκ των οποίων είναι το Κέντρο Ειδοποιήσεων (`NotificationCenter`). Η πλατφόρμα είναι κατασκευασμένη σύμφωνα με το προγραμματιστικό μοντέλο ανταλλαγής μηνυμάτων ειδοποιήσεων (event-driven message passing – notification exchanging). Όλα τα αρθρώματα επικοινωνούν μεταξύ τους μέσω μηνυμάτων που έχουν προκαθορισμένη μορφή και αντιπροσωπεύουν ειδοποιήσεις για κάποιο συμβάν. Κάθε ειδοποίηση αποτελείται από ένα όνομα, μια πηγή, έναν προορισμό και ένα σύνολο παραμέτρων. Το Κέντρο Ειδοποιήσεων αναλαμβάνει τον ρόλο του τοπικού δρομολογητή μηνυμάτων σε κάθε εκτελούμενη πλατφόρμα, τηρώντας μια βάση δεδομένων από δείκτες σε συναρτήσεις – που ονομάζονται *παρατηρητές* (observers), τις οποίες και καλεί όποτε δημιουργούνται οι αντίστοιχες ειδοποιήσεις. Ο σχεδιασμός αυτός είναι εν μέρει επηρεασμένος από το Foundation framework της γλώσσας προγραμματισμού Objective-C. Η παραγωγή ειδοποιήσεων, λοιπόν, συνεπάγεται τη σύγχρονη κλήση όλων των συναρτήσεων-παρατηρητών που έχουν καταχωρηθεί στη βάση δεδομένων με το ίδιο όνομα συμβάντος και την κατάλληλη διεύθυνση προορισμού. Σε αυτό το μοντέλο, όλες οι συναρτήσεις-παρατηρητές έχουν τα ίδια ορίσματα – δέχονται σαν είσοδο μια ειδοποίηση.

Επιπρόσθετα, το Κέντρο Ειδοποιήσεων ορίζει και δύο ειδικούς προορισμούς που μπορούν να χρησιμοποιηθούν κατά τη καταχώρηση παρατηρητών για κάποιο συγκεκριμένο όνομα συμβάντος (σχήμα Α΄.1):

- Ο προορισμός `allDestinations` σημαίνει ότι η συνάρτηση πρέπει να κληθεί για κάθε ειδοποίηση με το αντίστοιχο όνομα. Οι παρατηρητές που χρησιμοποιούν τον προορισμό αυτό λαμβάνουν παράλληλα όλα τα σχετικά μηνύματα, οπότε και μπορούν να υλοποιούν μετρητές ή καταγραφείς χωρίς να χρειάζεται να αλλάξει ο κώδικας των συναρτήσεων που υλοποιούν κάποιον αλγόριθμο.

- Ο προορισμός `unknownDestination` σημαίνει ότι η συνάρτηση πρέπει να κληθεί όταν δεν βρεθεί άλλος παρατηρητής στον πίνακα καταχωρήσεων για το αντίστοιχο όνομα. Έτσι υλοποιούνται συναρτήσεις διαχείρισης λαθών που προβάλουν ή καταγράφουν το πρόβλημα για περαιτέρω επεξεργασία.

Το μοντέλο των ειδοποιήσεων χρησιμοποιείται για όλες σχεδόν τις κλήσεις συναρτήσεων εντός της πλατφόρμας. Με αυτόν τον τρόπο, η ίδια η διαδικασία μεταφοράς της εκτέλεσης από ένα σημείο του προγράμματος σε ένα άλλο γίνεται σε ένα αφηρημένο επίπεδο και δεν έχει πια σημασία αν ο καλούμενος κώδικας βρίσκεται τοπικά ή σε κάποιο απομακρυσμένο μηχάνημα. Η κλήση ενός μη τοπικού παρατηρητή φαίνεται στο σχήμα Α΄.2. Όταν υλοποιούνται πρωτόκολλα ΚΠΚ εντός της πλατφόρμας, κάθε κόμβος ορίζει τα μηνύματα που δέχεται ζητώντας από το τοπικό Κέντρο Ειδοποιήσεων να καταχωρήσει τους κατάλληλους παρατηρητές για ειδοποιήσεις με προορισμό τον εαυτό του. Όταν χρειάζεται να επικοινωνήσει με έναν άλλο κόμβο, παράγει μια ειδοποίηση, που περνά σε πρώτο στάδιο από το τοπικό Κέντρο Ειδοποιήσεων. Αν ο κόμβος που

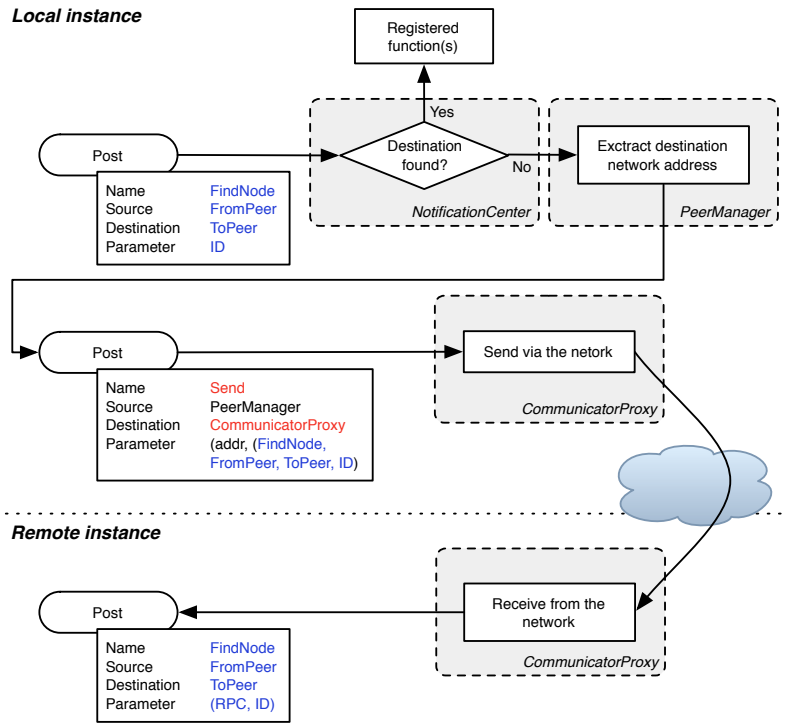


Σχήμα Α΄.1: *Συνοπτικό διάγραμμα εκτέλεσης του Κέντρου Ειδοποιήσεων. Αν το όνομα του συμβάντος αντιστοιχεί σε κάποιον ειδικό προορισμό, δρομολογείται κατάλληλα η κλήση των αντίστοιχων συναρτήσεων.*

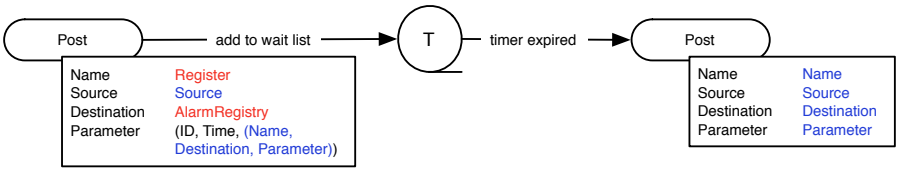
πρέπει να λάβει το μήνυμα – ο προορισμός – είναι στη βάση δεδομένων, καλείται άμεσα η αντίστοιχη συνάρτησή του. Αλλιώς, η ειδοποίηση (μέσω του προορισμού `unknownDestination`), πυροδοτεί τη διαμεσολάβηση των αρθρωμάτων `PeerManager` και `CommunicatorProxy`. Το πρώτο, γνωρίζοντας τις λεπτομέρειες διευθυνσιοδότησης των κόμβων, μετατρέπει την ειδοποίηση σε κατάλληλη μορφή για να την λάβει το δεύτερο και να την αποστείλει μέσω δικτύου στο ίδιο άρθρωμα που εκτελείται σε κάποια άλλη πλατφόρμα. Κατά τη παραλαβή ειδοποιήσεων μέσω δικτύου, το `CommunicatorProxy` τις ξαναδημιουργεί τοπικά. Έτσι, υλοποιείται η διαφανής μεταφορά ειδοποιήσεων από Κέντρο σε Κέντρο, χωρίς να χρειάζεται καμία ειδική διαδικασία από πλευράς κόμβων. Το `PeerManager`, μάλιστα, εκτός από το να βοηθά στην διατήρηση του αφαιρετικού επιπέδου επικοινωνίας, λαμβάνει και ειδοποιήσεις που μπορεί να σημάνουν την δημιουργία νέων κόμβων.

Όλες οι λειτουργίες που απαιτούν κάποια χρονική καθυστέρηση ή εξαρτώνται κατά κάποιο τρόπο από τον χρόνο οδηγούνται στο άρθρωμα `AlarmRegistry`. Αυτό αναλαμβάνει την δημιουργία ειδοποιήσεων μετά από ένα χρονικό διάστημα ή σε συγκεκριμένο σημείο στο μέλλον (σχήμα Α΄.3). Το `AlarmRegistry` επιτρέπει με τη σειρά του την υλοποίηση ενός πλαισίου χρονοπρογραμματιζόμενων διεργασιών, που είναι βασικό για την εκτέλεση πειραμάτων μεγάλης κλίμακας.

Το τελευταίο βασικό άρθρωμα του PeerPlatform είναι το `Peer`, που αποτελεί την αρχική κλάση για την δημιουργία ενός κόμβου ΚΠΚ. Το `Peer` είναι σχεδιασμένο σαν μια αφηρημένη μηχανή καταστάσεων, που μεταβαίνει από κατάσταση σε κατάσταση μέσω ειδοποιήσεων – είτε αυτές προέρχονται από μηνύματα, είτε από χρονοπρογραμματισμένα σήματα. Νέα γραφήματα καταστάσεων δημιουργούνται όταν ζητείται από έναν κόμβο να εκτελέσει μια λειτουργία ανάκτησης ή αποθήκευσης δεδομένων, ή όταν ο ίδιος ο κόμβος ξεκινά μια εσωτερική διαχειριστική λειτουργία. Όλες οι αλληλεπιδράσεις με τη μηχανή καταστάσεων γίνονται μέσω ειδοποιήσεων. Για παράδειγμα, τα μηνύματα μεταξύ των κόμβων έχουν τη μορφή ειδοποιήσεων, με όνομα τον τύπο του μηνύματος, πηγή και προορισμό τις αντίστοιχες διευθύνσεις των κόμβων. Η κλάση `Peer` προσφέρει ακόμη μια γενική υλοποίηση βάσης δεδομένων για την τοπική αποθήκευση και ανάκτηση πληροφοριών της μορφής κλειδιού-τιμής, καθώς και τις απαραίτητες υποδομές (τύποι δεδομένων και συναρτήσεις) για την επικοινωνία με το άρθρωμα `PeerManager`.



Σχήμα Α′.2: *Μεταφέροντας μηνύματα μεταξύ διαφορετικών εκτελέσεων της πλατφόρμας, μέσω δικτύου.*



Σχήμα Α′.3: *Χρονοκαθυστέρηση μηνυμάτων μέσω του αρθρώματος* `AlarmRegistry`.

Α′.1.2 Τρόποι λειτουργίας

Η αρχιτεκτονική αρθρωμάτων που περιγράφηκε μπορεί να χρησιμοποιηθεί στη πράξη με δύο διαφορετικούς τρόπους: είτε σε *κανονική διάταξη*, είτε σε *διάταξη προσομοίωσης*. Αφού οι κόμβοι είναι μοντελοποιημένοι ως μηχανές καταστάσεων και όλες οι λειτουργίες που εξαρτώνται από τον χρόνο είναι συγκεντρωμένες σε ένα ανεξάρτητο σημείο, μπορεί να χρησιμοποιηθεί ο ίδιος κώδικας, αφενός για την λειτουργία μιας πραγματικής επικάλυψης, αφετέρου δε για την οργάνωση προσομοιώσεων με εικονικό ρολόι. Στη *κανονική διάταξη* κάθε διαφορετικός κόμβος πραγματώνεται μέσω ενός ξεχωριστού νήματος εκτέλεσης (thread). Αυτό θέτει όμως και ένα πρακτικό όριο στον αριθμό των κόμβων που μπορούν να λειτουργούν ταυτόχρονα, καθώς κάθε φυσικό μηχάνημα έχει έναν μέγιστο αριθμό νημάτων που μπορεί να διαχειριστεί. Στη *διάταξη προσομοίωσης*, το άρθρωμα `PeerManager` δημιουργεί ένα μόνο νήμα, που αναλαμβάνει τον χρονοπρογραμματισμό των κόμβων. Κάθε εισερχόμενη ειδοποίηση που καταλήγει σε κόμβο, απλά ενημερώνει τον χρονοπρογραμματιστή ότι πρέπει να εκτελεστεί η αντίστοιχη συνάρτηση. Οι κόμβοι, σε αυτή τη περίπτωση, έχουν τη μορφή απλών συναρτήσεων, που καλούνται μόνο όταν

χρειάζεται. Ο χρόνος που περνά ελέγχεται από τον ίδιο τον χρονοπρογραμματιστή, οπότε το ρολόι δεν προχωρά όσο εκτελείται ο κώδικας των κόμβων, αλλά μπορεί να προσπερνώνται τεράστια διαστήματα όταν δεν συμβαίνει τίποτα – δεν παράγονται ειδοποιήσεις.

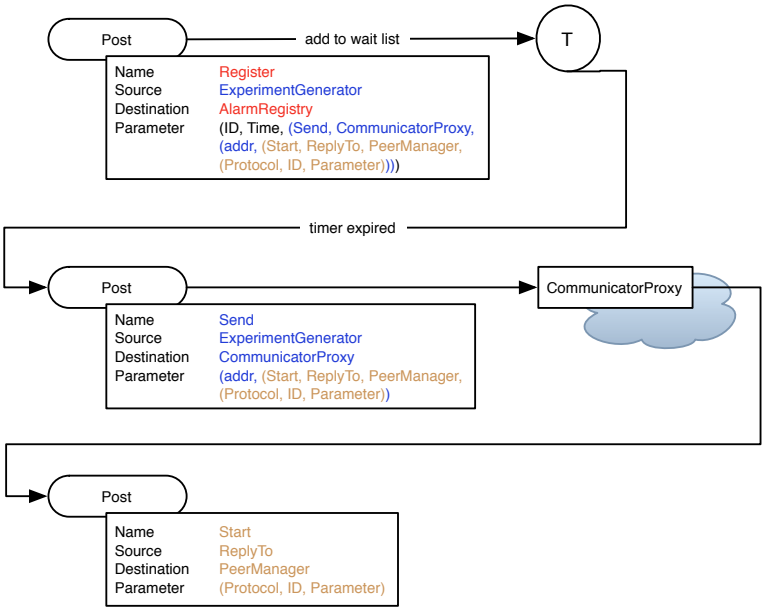
Στη διάταξη προσομοίωσης, θεωρείται ότι οι πράξεις εντός των κόμβων ολοκληρώνονται ακαριαία – κάτι που δεν απέχει και πολύ από τη πραγματικότητα, αφού συνήθως οι σχετικές συναρτήσεις είναι πολύ απλές και δεν απαιτούν πολύπλοκους υπολογισμούς. Από την άλλη μεριά, ο χρόνος επικοινωνίας δεν μπορεί να αγνοηθεί, άρα και πρέπει να υπολογιστεί. Έτσι, όλα τα μηνύματα που λαμβάνονται, ξαναστέλνονται μετά από μια τυχαία καθυστέρηση που ακολουθεί κανονική κατανομή. Στα πρωτόκολλα ΔΟΚ που δεν λαμβάνεται υπόψη η δικτυακή τοπικότητα μεταξύ των κόμβων, μπορεί απλά να θεωρηθεί, ότι κάθε βήμα επικοινωνίας έχει ένα μέσο κόστος. Βέβαια, η συνάρτηση που παράγει την τυχαία καθυστέρηση μπορεί εύκολα να αντικατασταθεί με μια άλλη που προσμετρά τον αντίκτυπο μιας πραγματικής διάταξης κόμβων στον χρόνο που απαιτείται για κάθε μήνυμα. Έτσι είναι δυνατόν να μελετηθούν και άλλου τύπου πρωτόκολλα που η τοπολογία τους ή οι διαδικασίες δρομολόγησης εξαρτώνται άμεσα από τις δικτυακές θέσεις των κόμβων. Ένας πολύ διαδεδομένος προσομοιωτής δικτύων, με τον οποίο θα μπορούσε να επικοινωνεί το PeerPlatform, είναι ο ns [ns].

Στη κανονική διάταξη, οι προγραμματιστές εφαρμογών που θέλουν να φτιάξουν υπηρεσίες σε ένα επίπεδο πάνω από τη πλατφόρμα, μπορούν να χρησιμοποιούν το ανοιχτό πρωτόκολλο επικοινωνίας με το οποίο επικοινωνούν και οι ίδιες οι εγκαταστάσεις μεταξύ τους και να δίνουν απευθείας εντολές στο αντίστοιχο Κέντρο Ειδοποιήσεων. Τέτοιου τύπου ειδοποιήσεις μπορεί να απευθύνονται στους ίδιους τους κόμβους ή στο άρθρωμα `PeerManager`, προκειμένου π.χ. να επιστρέψει ή να αποθηκεύσει στατιστικά στοιχεία. Έχει υλοποιηθεί, επίσης, ένα "τείχος προστασίας" (firewall) ειδοποιήσεων, ώστε οι διαχειριστές μιας εκτελούμενης υποδομής να μπορούν να ελέγχουν ποιοί τύποι μηνυμάτων, ανάλογα και με τη πηγή και τον προορισμό τους, μπορούν να καταλήγουν στο Κέντρο. Επιπρόσθετα, υποστηρίζεται μια απλή διεπαφή, της μορφής υπηρεσίας Παγκόσμιου Ιστού (web service) για την αποστολή εντολών ανάκτησης και αποθήκευσης στους τοπικούς κόμβους της πλατφόρμας.

Α΄.1.3 Τακτικές προσομοίωσης

Για να γίνει μια προσομοίωση, πρώτα κατασκευάζεται ένα *σενάριο*, ως μια λίστα από εντολές που δίνονται στο άρθρωμα `AlarmRegistry`, ώστε να εκτελεστούν σε συγκεκριμένες χρονικές στιγμές. Οι εντολές αυτές, ή καλύτερα οι ειδοποιήσεις, μπορεί να ζητούν την δημιουργία ενός νέου κόμβου, την αποχώρηση ενός συμμετέχοντα από την επικάλυψη, κάποια πράξη δεδομένων, κτλ.. Στη συνέχεια, σύμφωνα με τη διάταξη λειτουργίας, μπορεί να αρχικοποιηθεί η πλατφόρμα φορτώνοντας όλο το σενάριο ή να αποστέλλονται μία-μία οι εντολές στο Κέντρο Ειδοποιήσεων μέσω δικτύου, σε πραγματικό χρόνο. Η δεύτερη περίπτωση, συνιστάται όταν εκτελούνται παράλληλα πολλαπλές πλατφόρμες σε αντίστοιχα μηχανήματα και χρειάζεται ένας μηχανισμός χειρισμού τους. Με αυτό το τρόπο μπορούν να οργανωθούν προσομοιώσεις μεγάλης κλίμακας στο Πλέγμα ή σε μια συστοιχία υπολογιστών. Στο κεντρικό σημείο που προωθεί τις ειδοποιήσεις

εκτελείται και πάλι το PeerPlatform, αλλά χωρίς να δημιουργούνται κόμβοι – απλά χρησιμοποιούνται τα αρθρώματα `CommunicatorProxy` και `AlarmRegistry` για να δίνονται οι εντολές και να συλλέγονται τα αποτελέσματα, όπως φαίνεται στο σχήμα Α΄.4.



Σχήμα Α΄.4: *Σε κανονική διάταξη, η πλατφόρμα μπορεί να χρησιμοποιηθεί για την προγραμματισμένη αποστολή μηνυμάτων από ένα κεντρικό σημείο σε πολλαπλά μηχανήματα.*

Στη διάταξη προσομοίωσης, το σενάριο εκτελείται σε εικονικό χρόνο, εντός μίας πλατφόρμας, σε ένα μηχάνημα. Ο διαμοιρασμός του σχετικού φόρτου σε πολλαπλά μηχανήματα, θα απαιτούσε την κατανομή της λειτουργίας του εικονικού ρολογιού. Αντίθετα, προτιμήθηκε η κεντρικοποιημένη διάταξη προσομοίωσης, καθώς επιτρέπει την εύκολη συλλογή δεδομένων και στατιστικών στοιχείων από όλους τους κόμβους ταυτόχρονα και την εκτέλεση πράξεων κατά τις οποίες απαιτείται συνολική άποψη των χαρακτηριστικών όλης της επικάλυψης. Έτσι, σε αυτή τη διάταξη, το PeerPlatform καταγράφει στατιστικά για όλα τα μηνύματα που ανταλλάσσονται, καθώς και πληροφορίες από την εποπτεία όλων των πινάκων δρομολόγησης των κόμβων. Από τα δεδομένα αυτά εξάγονται:

- Ο συνολικός αριθμός των κόμβων ως συνάρτηση του χρόνου, όπου φαίνονται και οι αφίξεις και αναχωρήσεις στη διάρκεια της προσομοίωσης.
- Ο αριθμός των μηνυμάτων ως συνάρτηση του χρόνου, είτε ως σύνολο, είτε ανά τύπο, είτε οργανωμένα σε αυτά που παραδόθηκαν και αυτά που απέτυχαν.
- Ο αριθμός των μηνυμάτων και ο χρόνος που χρειάστηκε κατά μέσο όρο για κάθε πράξη ανάκτησης ή αποθήκευσης δεδομένων. Τα μηνύματα αυτά αναλύονται, στη συνέχεια, ανά τύπο και σημειώνεται πόσα από αυτά παραδόθηκαν και πόσα απέτυχαν.
- Η κατανομή εισερχόμενων επαφών ανά αριθμό κόμβων, δηλαδή, η συνάρτηση του αριθμού των κόμβων που "γνωρίζουν" κάποιον άλλο, προς τον αριθμό των κόμβων που έχουν τόσες επαφές.

- Η κατανομή δεδομένων, δηλαδή, η συνάρτηση του αριθμού των αντιγράφων που υπάρχουν για ένα ζεύγος κλειδιού-τιμής, προς τον αριθμό των δεδομένων για τα οποία υπάρχουν τόσα αντίγραφα.

Αυτές οι πληροφορίες βοηθούν στην κατανόηση των χαρακτηριστικών μιας επικάλυψης, καθώς και στην αποτίμηση της αποδοτικότητας, της ικανότητας κλιμάκωσης και της αξιοπιστίας της, σύμφωνα με το σενάριο και τις παραμέτρους λειτουργίας της.

Α΄.2 Αξιολόγηση του Kademlia

Για την επίδειξη της πλατφόρμας και την αξιολόγηση της υλοποίησης του πρωτοκόλλου Kademlia, εκτελέστηκαν διάφορα σενάρια προσομοίωσης, σε επικαλύψεις μεγέθους από μερικές εκατοντάδες μέχρι μερικές χιλιάδες κόμβων. Σε όλες τις περιπτώσεις, το PeerPlatform χρησιμοποιήθηκε σε *διάταξη προσομοίωσης*. Στις επόμενες παραγράφους, παρουσιάζονται τα σχετικά αποτελέσματα από σενάρια στα οποία το δίκτυο έχει δυναμικά χαρακτηριστικά – κόμβοι συνδέονται και αποσυνδέονται συνεχώς. Τα σενάρια αυτά δημιουργήθηκαν με βάση τις παρακάτω παραμέτρους:

- Αρχικό μέγεθος δικτύου: Ο αριθμός των κόμβων που συμμετέχουν στην επικάλυψη με τη κατασκευή της. Αυτοί οι κόμβοι χρησιμοποιούνται από όλους τους υπόλοιπους σαν σημεία αναφοράς για όταν εισέρχονται στο δίκτυο.
- Αρχικό πλήθος δεδομένων: Ο αριθμός των ζευγών κλειδιού-τιμής που είναι αποθηκευμένα στους αρχικούς κόμβους.
- Ρυθμός εισερχόμενων και εξερχόμενων κόμβων ανά ώρα: Ο αριθμός των κόμβων που δημιουργούνται και καταστρέφονται ανά ώρα προσομοίωσης. Οι αντίστοιχες πράξεις παράγονται από μια τυχαία συνάρτηση που ακολουθεί κανονική κατανομή.
- Μέγιστο μέγεθος δικτύου: Πόσοι κόμβοι πρέπει να είναι στο δίκτυο πριν αρχίσουν οι αποχωρήσεις. Αφού οι ρυθμοί εισερχόμενων και εξερχόμενων κόμβων ανά ώρα είναι ίδιοι, όταν το δίκτυο φτάσει σε αυτό το μέγεθος, η ποσότητα των κόμβων διατηρείται κατά μέσο όρο σταθερή και ίση με τη παράμετρο.
- Ρυθμός αιτήσεων ανάκτησης και αποθήκευσης ανά ώρα: Ο αριθμός των αιτημάτων *put* και *get* που θα παραχθούν και θα αποσταλούν στους συμμετέχοντες σε μία ώρα. Όπως και με τον ρυθμό ανανέωσης κόμβων, οι εντολές παράγονται από μια τυχαία συνάρτηση που ακολουθεί κανονική κατανομή.
- Συνολική διάρκεια προσομοίωσης: Πόσα δευτερόλεπτα εικονικού χρόνου θα διαρκέσει η εκτέλεση του σεναρίου από τη στιγμή της αρχικοποίησης – της δημιουργίας των αρχικών κόμβων και την αποθήκευση των αρχικών δεδομένων.
- Καθυστέρηση κατά την εισαγωγή κόμβου, ζεύγους κλειδιού-τιμής και πράξης: Πόσα δευτερόλεπτα πρέπει να περάσουν προτού να θεωρηθεί ένας νέος κόμβος ότι αποτελεί πλέον

μέρος της επικάλυψης, προτού γίνουν πράξεις ανάκτησης για μια τιμή που μόλις αποθηκεύτηκε και μεταξύ πράξεων στον ίδιο κόμβο. Αυτές οι παράμετροι αφορούν κυρίως την αναμενόμενη συμπεριφορά της εφαρμογής που χρησιμοποιεί τον ΚΠΚ.

Η επιλογή των παραπάνω παραμέτρων υπονοεί ότι έχουν γίνει κάποιες παραδοχές σχετικά με το δίκτυο που μελετάται. Από τη μια μεριά, όμως, είναι θεμιτό τα σενάρια που παράγονται να έχουν κοινά χαρακτηριστικά συμπεριφοράς, ώστε να είναι δυνατή η απευθείας σύγκριση διαφορετικών περιπτώσεων, που ενώ στηρίζονται στην ίδια βάση, διαφέρουν στις τάξεις μεγέθους μιας απλής ποσότητας – όπως τον ρυθμό αιτήσεων ή το μέγεθος της επικάλυψης. Επιπλέον, οι παράμετροι του σεναρίου έχουν σαν στόχο η προσομοίωση να πλησιάζει όσο το δυνατόν περισσότερο τις καταστάσεις που επικρατούν και στη πραγματικότητα. Για παράδειγμα, κάποιοι αρχικοί κόμβοι χρησιμοποιούνται και σε πραγματικά δίκτυα, σαν *επαφές εισόδου* (kickstart peers) στην επικάλυψη. Η δικτυακή τους διεύθυνση συνήθως δημοσιοποιείται στο Διαδίκτυο, ώστε να μπορεί οποιοσδήποτε νέος κόμβος να ξεκινήσει από εκεί τη διαδικασία αρχικοποίησης των πινάκων δρομολόγησής του. Η πρώτη φάση προσθήκης κόμβων και δεδομένων, αντιστοιχεί στη δημιουργία μιας σταθερής επικάλυψης. Μόλις το μέγεθος του δικτύου φτάσει στο μέγιστο, αρχίζουν διάφοροι συμμετέχοντες και αναχωρούν τυχαία, προκαλώντας ανακατατάξεις στους πίνακες δρομολόγησης και προβλήματα επικοινωνίας, αφού τα μηνύματα που απευθύνονται στους ανύπαρκτους – πια – κόμβους δεν μπορούν να παραδοθούν. Έτσι, καθυστερούν και οι αντίστοιχες πράξεις στα δεδομένα, αφού οι αλγόριθμοι συνήθως περιμένουν κάποιο χρονικό διάστημα ή επαναλαμβάνουν το μήνυμα, πριν καταλήξουν ότι δεν είναι δυνατή η επικοινωνία.

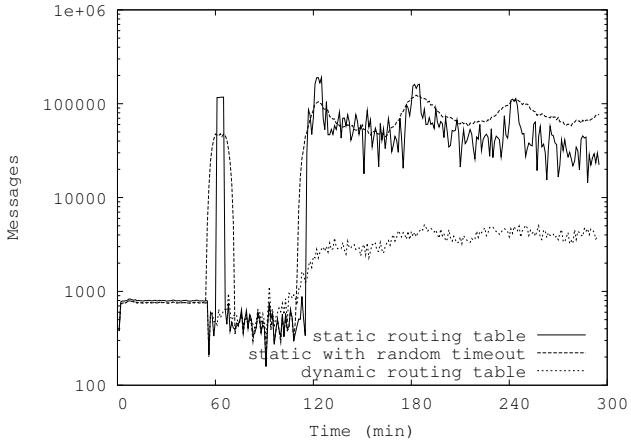
Για τα αποτελέσματα που αναλύονται στη συνέχεια, εκτελέστηκαν τρία βασικά σενάρια, με τις παραμέτρους που φαίνονται στον πίνακα Α΄.1. Οι καθυστερήσεις κατά την εισαγωγή κόμβου, ζεύγους κλειδιού-τιμής και πράξης, ορίστηκαν σε 3, 3 και 10 αντίστοιχα. Ειδικά το τελευταίο νούμερο, δεν είναι απαραίτητο να είναι τόσο υψηλό – εξάλλου οι κόμβοι μπορούν να επεξεργάζονται ταυτόχρονα πολλά ερωτήματα – αλλά παρατηρήθηκε ότι έτσι κατανέμονται καλύτερα οι εντολές στους διαθέσιμους συμμετέχοντες.

Πίνακας Α΄.1: *Παράμετροι ανά σενάριο εκτέλεσης*

	A	B	C
Initial network size	10	100	100
Initial data items count	1000	10000	10000
Nodes in and out per hour	600	6000	1200
Max network size	1000	10000	10000
Item get rate	600	6000	9600
Total simulation time	4 h	8 h	16 h
Total nodes in	2452	47994	19141
Total nodes out	1415	37867	9110
Total get commands	2430	47826	153959

Α΄.2.1 Υλοποίηση του πίνακα δρομολόγησης

Ένα από τα σημαντικότερα ίσως σημεία στην υλοποίηση ενός ΚΠΚ, είναι η αποδοτικότητα του πίνακα δρομολόγησης. Στον πίνακα αυτό, που στην ουσία είναι μια βάση δεδομένων για κάθε κόμβο, αποθηκεύονται οι διευθύνσεις των ομότιμών του στο δίκτυο και εκτελούνται συγκεκριμένες πράξεις δεδομένων, που πρέπει να γίνονται με ταχύτητα και αξιοπιστία. Στο Kademlia οι επαφές οργανώνονται σε 160 υπο-πίνακες ή ομάδες, η κάθε μία εκ των οποίων αντιπροσωπεύει και εκθετικά μεγαλύτερο μέρος του εικονικού χώρου διευθύνσεων που έχει συμφωνηθεί μεταξύ των συμμετεχόντων. Στη σχετική βιβλιογραφία, οι σχεδιαστές του πρωτοκόλλου χρησιμοποιούν μια στατική δομή πινάκων για να εξηγήσουν τις διαδικασίες, αλλά προτείνουν και μια δυναμική υλοποίηση, όπου οι διάφοροι υπο-πίνακες δημιουργούνται μόνο όταν χρειάζεται. Δεδομένου ότι η στατική δομή είναι πολύ πιο εύκολη στην υλοποίηση και μπορεί ήδη να χρησιμοποιείται από διάφορα δίκτυα εν λειτουργία, συγκρίναμε τους δύο μηχανισμούς, αξιολογώντας τις επιπτώσεις που έχει η διάταξη του πίνακα δρομολόγησης στις λειτουργίες διαχείρισης του δικτύου. Και στις δύο περιπτώσεις, για την ανανέωση των επαφών, χρησιμοποιείται λανθάνουσα μνήμη διευθύνσεων ανά ομάδα και όχι μηνύματα `PING` – μιά βελτίωση που αναφέρεται ως *optimized contact accounting* (βελτιστοποιημένη διαχείριση επαφών).



Σχήμα Α΄.5: *Διαχειριστικά μηνύματα για στατική και δυναμική υλοποίηση των πινάκων δρομολόγησης (σενάριο Α)*

Στο σχήμα Α΄.5, φαίνονται οι μετρήσεις που προέκυψαν από το δίκτυο Α. Παρουσιάζονται μόνο τα σύνολα των μηνυμάτων `FIND_NODE` και `NODE`, αφού για το εξεταζόμενο σενάριο λειτουργίας, τα συγκεκριμένα μηνύματα, μετά την αρχική φάση εισαγωγής δεδομένων, εξυπηρετούν μόνο την διαχείριση των πινάκων δρομολόγησης. Επειδή η πράξη *put* συμπεριλαμβάνει μια φάση εύρεσης των πλησιέστερων στο κλειδί κόμβων, το πρωτόκολλο ορίζει ότι ένας υπο-πίνακας μπορεί να θεωρηθεί "φρέσκος" όταν έχει γίνει κάποια πράξη εισαγωγής στην αντίστοιχη περιοχή του εικονικού χώρου διευθύνσεων, μέσα σε ένα προκαθορισμένο χρονικό διάστημα. Όταν περάσει μια ώρα από την τελευταία εισαγωγή ζεύγους κλειδιού τιμής, ο κόμβος θα επιλέξει τυχαία μια διεύθυνση μέσα στο εύρος που αντιστοιχεί στον υπο-πίνακα και θα κάνει από μόνος του μια αναζήτηση κοντινών ομότιμων. Με αυτό τον τρόπο το Kademlia εξισορροπεί την επικάλυψη (stabilization),

δηλαδή διορθώνει αυτόματα τις ενδεχομένως λανθασμένες πληροφορίες δρομολόγησης στους πίνακες των κόμβων, κάνοντας το σύνολο του δικτύου πιο συνεκτικό και άρα πιο ανθεκτικό σε προβλήματα που μπορεί να προκύπτουν στο επίπεδο του δικτύου ή των ίδιων των ομότιμων.

Όταν, όμως, χρησιμοποιούνται στατικοί πίνακες δρομολόγησης, το πλήθος των μηνυμάτων που απαιτούνται για την εξισορρόπηση του δικτύου είναι τεράστιο. Στο γράφημα παρουσιάζονται απότομες κορυφές που αντιπροσωπεύουν την έντονη δραστηριότητα διαχείρισης που συμβαίνει κάθε μία ώρα από την δημιουργία των αρχικών κόμβων. Τα υπόλοιπα σημεία αντιστοιχούν στους κόμβους που εισέρχονται μετέπειτα στο δίκτυο. Εισάγοντας μια τυχαία καθυστέρηση της τάξης του $\pm 10\%$ στο διάστημα της μίας ώρας βοηθά στην εξομάλυνση των κορυφών – και κατά συνέπεια των αντίστοιχων απότομων ρυθμών επικοινωνίας στο επίπεδο του δικτύου – αλλά όχι στη μείωση του αριθμού των μηνυμάτων. Ακόμα και η αύξηση του ρυθμού εισαγωγής δεδομένων δεν επιφέρει σημαντικές διαφορές, αντίθετα στις προβλέψεις ότι έτσι θα γίνονται τελικά περισσότερες αναζητήσεις πλησιέστερων κόμβων έμμεσα, οπότε και δεν θα χρειάζεται να τις κάνει μετά από μόνο του το δίκτυο. Το πρόβλημα εντοπίζεται στο γεγονός ότι κάθε κόμβος συνεχώς ανανεώνει υπο-πίνακες που είναι στην ουσία άδειοι – και το σημαντικότερο, παραμένουν κενοί ακόμα και μετά την σχετική αναζήτηση. Δεδομένου του μεγέθους του εικονικού χώρου διευθύνσεων και της μεθοδολογίας διαχωρισμού των ομάδων επαφών, αναμένεται ότι ακόμα και σε υπερβολικά μεγάλα δίκτυα, η πλειοψηφία των 160 υπο-πινάκων είναι περιττοί. Κάθε κόμβος δεν χρειάζεται παραπάνω από $\log_2(N/\kappa)$ ομάδες επαφών. Ακόμα και για μια επικάλυψη που αποτελείται από 1 τρισεκατομμύριο κόμβους, αυτό σημαίνει ότι τελικά κάθε συμμετέχων θα αξιοποιεί περίπου 36 υπο-περιοχές.

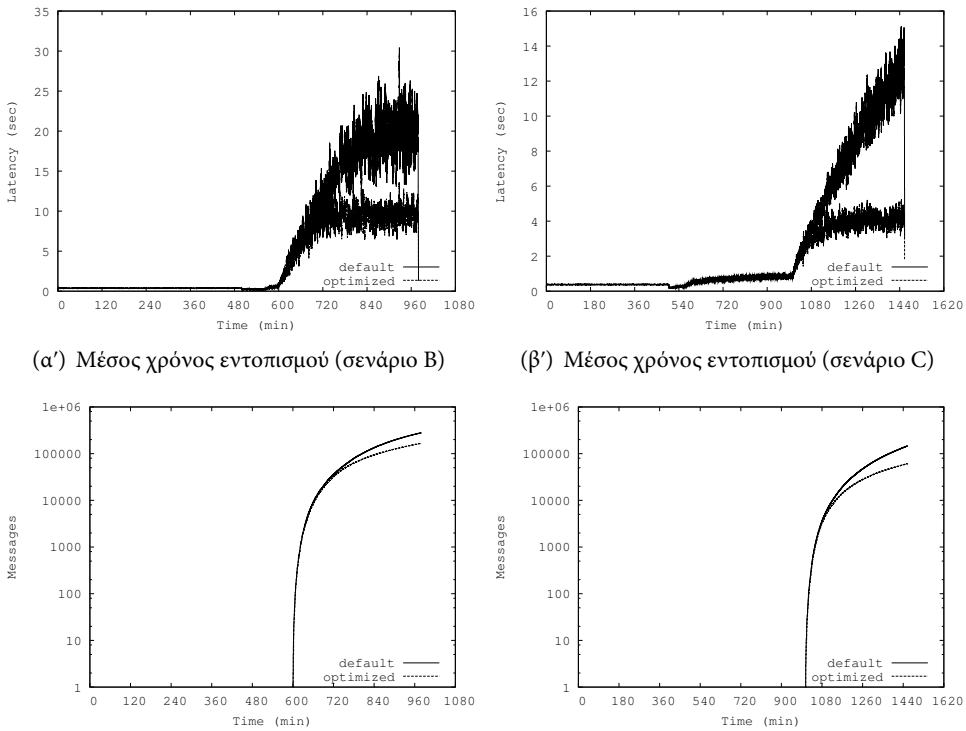
Έτσι, όπως προβλεπόταν, το κόστος διαχείρισης ενός δικτύου όπου χρησιμοποιούνται δυναμικοί πίνακες δρομολόγησης είναι σημαντικά μικρότερο. Στο σενάριο που εξετάζεται, με δυναμικούς πίνακες, το άθροισμα των μηνυμάτων `FIND_NODE` και `NODE`, πέφτει από πάνω από 14 εκατομμύρια σε λιγότερο από 800.000 και με πολύ καλύτερη κατανομή της επικοινωνίας. Άρα, η υλοποίηση των συγκεκριμένων πινάκων είναι απαραίτητη για μια αποδοτική επικάλυψη Kademlia. Μπορεί με το συγκεκριμένο σενάριο προσομοίωσης το πρόβλημα των αυξημένων μηνυμάτων διαχείρισης να φαίνεται στην υπερβολή του, καθώς δεν γίνονται καθόλου εισαγωγές δεδομένων στο δίκτυο, παρόλα αυτά ανάλογα μεγέθη αναμένονται και σε πραγματικές επικαλύψεις όπου οι αναγνώσεις είναι σημαντικά περισσότερες από τις εγγραφές.

Α′.2.2 Βελτιστοποιήσεις στην εκτέλεση πράξεων δεδομένων

Ο μέσος χρόνος που απαιτείται για κάθε πράξη *get* κλιμακώνεται όσο περισσότεροι κόμβοι αναχωρούν από το δίκτυο. Αυτό συμβαίνει επειδή κάθε αναχώρηση οδηγεί στο να απευθύνονται μηνύματα σε ανύπαρκτους τελικά προορισμούς. Οι αντίστοιχοι συμμετέχοντες που στέλνουν τα ερωτήματα, πρέπει να περιμένουν κάποιο προκαθορισμένο διάστημα πριν συνεχίσουν με τον αλγόριθμο (timeout). Οι μετρήσεις που έγιναν, έδειξαν ότι οι πίνακες δρομολόγησης ανανεώνονται υπερβολικά αργά, ειδικά για τα σενάρια του πίνακα Α′.1. Αυτό οφείλεται αφενός στο ότι πρέπει να γίνουν αρκετές απόπειρες επικοινωνίας πριν κριθεί ότι ένας κόμβος έχει φύγει από το δίκτυο,

και αφετέρου στο ότι οι συμμετέχοντες στο Kademlia δεν ενημερώνουν κανέναν όταν αποσυνδέονται. Υπάρχουν όμως και ισχυρές δικαιολογίες στο να συνεχίσουν να ακολουθούνται οι δύο φαινομενικά προβληματικές τακτικές. Η πρώτη συμβάλει στην συνοχή του δικτύου, αφού δεν διαγράφεται απευθείας μια επαφή με μια προβληματική επικοινωνία – κάτι που αναμένεται, τόσο επειδή χρησιμοποιείται το πρωτόκολλο UDP για την ανταλλαγή μηνυμάτων, όσο και επειδή το πρωτόκολλο έχει οριστεί ότι πρέπει να είναι ανεκτικό σε απότομες διακοπές λειτουργίας του δικτύου. Η δεύτερη αιτία, μπορεί να θεωρηθεί και πλεονέκτημα του Kademlia, αφού με το να μην απαιτούνται αυστηρές διαδικασίες για την έξοδο ενός κόμβου από την επικάλυψη, οι πίνακες δρομολόγησης είναι απλούστεροι. Εννοείται, από τη σχεδίαση του πρωτοκόλλου, ότι η ανοχή σε τέτοιου είδους σφάλματα υλοποιείται στο επίπεδο των πράξεων δεδομένων και όχι μέσω ισχυρών παραδοχών για την πληρότητα και αξιοπιστία των ίδιων των πινάκων δρομολόγησης – κάτι που ισχύει αντίστροφα για άλλες επικαλύψεις, όπως το Chord.

Χωρίς, λοιπόν, να αλλάξουμε τις βασικές λειτουργικές προδιαγραφές του πρωτοκόλλου, επιχειρήσαμε την βελτιστοποίηση των πράξεων ανάκτησης και εισαγωγής δεδομένων, ειδικά για τις περιπτώσεις που στο δίκτυο συνεχώς αναχωρούν κόμβοι και εισέρχονται νέοι, αλλάζοντας συνεχώς την ισορροπία του. Μια πρώτη παρατήρηση είναι ότι η αύξηση του αριθμού των πράξεων κατά τη διάρκεια της ανανέωσης κόμβων, οδηγεί σε καλύτερους χρόνους εκτέλεσης, αφού για κάθε *put* ή *get* ανταλλάσσονται μηνύματα που βοηθούν στην αναγνώριση των προβληματικών επαφών και την απομάκρυνσή τους από τους πίνακες δρομολόγησης. Η δομή των πινάκων (δυναμική ή στατική) δεν επηρεάζει καθόλου το αποτέλεσμα.



(α΄) Μέσος χρόνος εντοπισμού (σενάριο B)

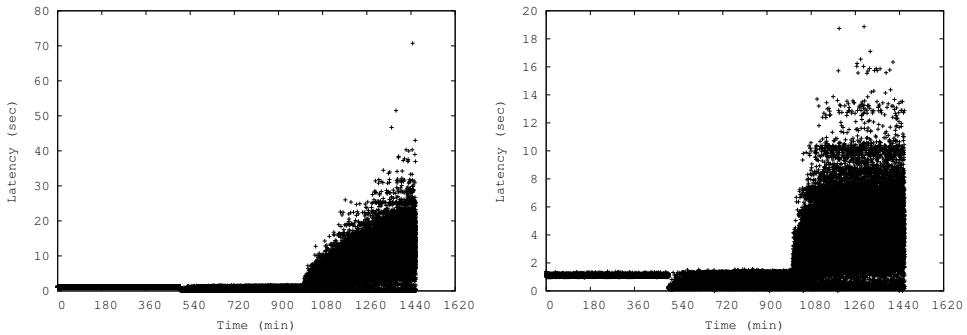
(β΄) Μέσος χρόνος εντοπισμού (σενάριο C)

(γ΄) Αποτυχημένα μηνύματα (σενάριο B)

(δ΄) Αποτυχημένα μηνύματα (σενάριο C)

Σχήμα Α΄.6: *Βελτίωση στην απόκριση εντολών, μέσω της εναλλακτικής επιλογής επαφών.*

Οι προτεινόμενες βελτιστοποιήσεις έχουν ως σκοπό, τόσο την αποφυγή των άκυρων επαφών στη διαδρομή που ακολουθείται μέσα στο δίκτυο, όσο και την πιο αποδοτική χρήση των πληροφοριών δρομολόγησης που διατηρεί κάθε κόμβος. Το πρώτο επιτυγχάνεται μέσω της δυναμικής προσαρμογής του μονοπατιού εκτέλεσης, όπου οι κόμβοι δεν περιορίζουν τον αριθμό των παράλληλων μηνυμάτων σε α κατά τη διάρκεια της πράξης, αλλά προσπαθούν να διατηρούν α από κ εκκρεμή μηνύματα σε κάθε βήμα του αλγορίθμου. Η μέθοδος αυτή περιγράφεται αναλυτικά στο κεφάλαιο 3. Τα αποτελέσματα δεν είναι θεαματικά, αλλά υπάρχει βελτίωση. Η δεύτερη μέθοδος, βασίζεται στο γεγονός ότι οι νέες επαφές που μαθαίνει κάποιος κόμβος καταχωρούνται μεν αμέσως στη λανθάνουσα μνήμη του αντίστοιχου υπο-πίνακα, αλλά δεν χρησιμοποιούνται παρά μόνο όταν κάποιος ομότιμος της κύριας δομής αποτύχει στο να απαντήσει έναν συγκεκριμένο αριθμό φορών (συνήθως 3). Αντί να μειώσουμε τον αριθμό αυτό – κάτι που μπορεί να επηρεάσει τη σταθερότητα της επικάλυψης συνολικά, αλλάξαμε τον τρόπο που επιλέγονται οι επαφές από τον πίνακα, ώστε να συμπεριλαμβάνονται όποτε χρειάζεται και μερικές φρέσκες διευθύνσεις επικοινωνίας. Έτσι, όταν συμπληρώνεται η λίστα των κοντινότερων κόμβων από τον πίνακα δρομολόγησης, αντικαθιστώνται επαφές που έχουν αποτύχει έστω και μία φορά με ισάριθμες καταχωρήσεις από την κορυφή της λανθάνουσας μνήμης (η δομή που χρησιμοποιείται είναι ουρά LIFO). Για να καθαρίσουν τελικά οι πίνακες δρομολόγησης, στέλνονται περιοδικά μηνύματα PING, με εκθετικά αυξανόμενη καθυστέρηση, σε όσους συμμετέχοντες δεν απαντήσουν εντός χρονικών ορίων. Όταν περάσει ο προκαθορισμένος αριθμός αποτυχιών για μια επαφή, διαγράφεται, και τη θέση της παίρνει η νεότερη από τη λανθάνουσα μνήμη. Πρακτικά, με αυτό το τρόπο, η ανανέωση των πινάκων γίνεται προληπτικά και όχι αφού οι προβληματικές επαφές επηρεάσουν την εκτέλεση των πράξεων δεδομένων στον μέγιστο επιτρεπτό βαθμό.



(α΄) Χωρίς βελτιστοποίηση (β΄) Με βελτιστοποίηση

Σχήμα Α΄.7: *Σύγκριση των απόλυτων χρόνων που απαιτούνται για κάθε πράξη εντοπισμού (σενάριο C)*

Στο σενάριο προσομοίωσης A, τα πλεονεκτήματα της παραπάνω μεθόδου είναι ευδιάκριτα, αλλά όχι τόσο προφανή όσο όταν εφαρμόζονται στα σύνολα εντολών που προκύπτουν από τα σενάρια B και C, που παράγουν συνεχώς μεταβαλλόμενες επικαλύψεις 10.000 κόμβων. Στο σχήμα Α΄.6, παρουσιάζονται και για τα δύο σενάρια, ο κυλιόμενος μέσος όρος ενός λεπτού του χρόνου που απαιτείται για κάθε πράξη ανάκτησης, καθώς και ο συνολικός αριθμός των μηνυμάτων που δεν κατάφεραν να παραδοθούν. Η καθυστέρηση που αντιστοιχεί σε κάθε πράξη, όχι μόνο μειώνεται,

αλλά φαίνεται να περιορίζεται από έναν μέγιστο αριθμό αποτυχιών – κάθε μία αντιστοιχεί σε 3 δευτερόλεπτα καθυστέρησης. Όταν δεν επιλέγονται επαφές από τη λανθάνουσα μνήμη, οι πράξεις χρειάζονται συνεχώς περισσότερο χρόνο για να ολοκληρωθούν, όσο τα μέλη του δικτύου συνεχίζουν να αποχωρούν με σταθερό ρυθμό. Χρησιμοποιήσαμε κυλιόμενο μέσο όριο για να είναι πιο ξεκάθαρη η απεικόνιση της διαφοράς των δύο μεθόδων. Τα απόλυτα νούμερα, που φαίνονται στο σχήμα Α΄.7, αποκαλύπτουν ότι για το σενάριο C, χωρίς τις βελτιστοποιήσεις, μερικές εντολές μπορεί να χρειαστούν μέχρι και 30 δευτερόλεπτα για να εκτελεστούν προς το τέλος της προσομοίωσης. Από την άλλη μεριά, με την εναλλακτική επιλογή επαφών, οι περισσότερες ανακτήσεις δεδομένων ολοκληρώνονται με 4 μόλις αποτυχίες. Για το σενάριο C, το κέρδος εμπλουτισμού των πινάκων πλησιέστερων κόμβων με φρέσκους ομότιμους, είναι πάνω από 50% και αυξάνεται ακόμα περισσότερο αν συνεχιστεί η προσομοίωση με τους ίδιους ρυθμούς ανανέωσης συμμετεχόντων.

Βιβλιογραφία

[ABB+02] Bill Allcock, Joe Bester, John Bresnahan, Ann L. Chervenak, Ian Foster, Carl Kesselman, Sam Meder, Veronika Nefedova, Darcy Quesnel, and Steven Tuecke, *Data Management and Transfer in High-Performance Computational Grid Environments*, Parallel Computing **28** (2002), no. 5, 749–771.

[ABK+05] William Allcock, John Bresnahan, Rajkumar Kettimithu, Michael Link, Catalin Dumitresku, Ioan Raicu, and Ian Foster, *The Globus Striped GridFTP Framework and Server*, Proceedings of the 2005 ACM/IEEE Conference on Supercomputing (Seattle, WA, USA), November 2005.

[ALPH01] L. A. Adamic, R. M. Lukose, A. R. Puniyani, and B. A. Huberman, *Search in Power Law Networks*, Physical Review E **64** (2001), 046135–1 – 046135–8.

[AS07] James Aspnes and Gauri Shah, *Skip graphs*, ACM Transactions on Algorithms **3** (2007), no. 4, 384 – 393.

[AX02] Artur Andrzejak and Zhichen Xu, *Scalable, Efficient Range Queries for Grid Information Services*, Proceedings of the 2nd International Conference on Peer-to-Peer Computing (P2P2002) (Linköping, Sweden), September 2002.

[BAS04] Ashwin R. Bharambe, Mukesh Agrawal, and Srinivasan Seshan, *Mercury: Supporting scalable multi-attribute range queries*, Proceedings of the ACM SIGCOMM '04 Conference (Portland, OR, USA), August 2004.

[Bay97] Rudolf Bayer, *The Universal B-tree for Multidimensional Indexing: General Concepts*, Proceedings of the International Conference on Worldwide Computing and Its Applications '97 (WWCA97) (Tsukuba, Japan), March 1997.

[BKK+03] Hari Balakrishnan, M. Frans Kaashoek, David Karger, Robert Morris, and Ion Stoica, *Looking Up Data in P2P Systems*, Communications of the ACM **46** (2003), no. 2, 43–48.

[BPS05] Jean-Michel Busca, Fabio Picconi, and Pierre Sens, *Pastis: A Highly-Scalable Multi-User Peer-to-Peer File System*, Proceedings of the 11th International Euro-Par Conference (Lisbon, Portugal), August 2005.

[BTD] *BitTorrent.org — DHT Protocol*, Available online: http://www.bittorrent.org/.

[Cat03] Josh Cates, *Robust and Efficient Data Management for a Distributed Hash Table*, Master's thesis, MIT, May 2003.

[CDF+02] Ann Chervenak, Ewa Deelman, Ian Foster, Leanne Guy, Wolfgang Hoschek, Adriana Iamnitchi, Carl Kesselman, Peter Kunszt, Matei Ripeanu, Bob Schwartzkopf, Heinz Stockinger, Kurt Stockinger, and Brian Tierney, *Giggle: A Framework for Constructing Scalable Replica Location Services*, Proceedings of the 2002 ACM/IEEE Conference on Supercomputing (Baltimore, MD, USA), November 2002.

[CDT07] Eddy Caron, Frederic Desprez, and Cédric Tedeschi, *Enhancing Computational Grids with Peer-to-Peer Technology for Large Scale Service Discovery*, Journal of Grid Computing **5** (2007), no. 3, 337 – 360.

[CFK+00] A. Chervenak, I. Foster, C. Kesselman, C. Salisbury, and S. Tuecke, *The Data Grid: Towards an Architecture for the Distributed Management and Analysis of Large Scientific Datasets*, Journal of Network and Computer Applications **23** (2000), no. 3, 187–200.

[CK05] Antony Chazapis and Nectarios Koziris, *Storing and Locating Mutable Data in Structured Peer-to-Peer Overlay Networks*, Proceedings of the 10th Panhellenic Conference on Informatics (PCI'2005) (Volos, Greece), November 2005.

[CK07] _____, *XOROS: A Mutable Distributed Hash Table*, Proceedings of the 5th International Workshop on Databases, Information Systems and Peer-to-Peer Computing (DBISP2P 2007) (Vienna, Austria), September 2007.

[CMM02] Russ Cox, Athicha Muthitacharoen, and Robert T. Morris, *Serving DNS using a Peer-to-Peer Lookup Service*, Proceedings of the 1st International Workshop on Peer-to-Peer Systems (IPTPS'02) (Cambridge, MA, USA), March 2002.

[Coh03] Bram Cohen, *Incentives build robustness in bittorrent*, Available online: http://www.bittorrent.org/, May 2003.

[CPB+04] A. Chervenak, N. Palavalli, S. Bharathi, C. Kesselman, and R. Schwartzkopf, *Performance and Scalability of a Replica Location Service*, Proceedings of the 13th IEEE International Symposium on High Performance Distributed Computing (HPDC-13) (Honolulu, HI, USA), June 2004.

[CRR+05] Yatin Chawathe, Sriram Ramabhadran, Sylvia Ratnasamy, Anthony Lamarca, Scott Shenker, and Joseph M. Hellerstein, *A case study in building layered dht applications*, Proceedings of the ACM SIGCOMM '05 Conference (Philadelphia, PA, USA), August 2005.

[CRWZ05] Xin Chen, Shansi Ren, Haining Wang, and Xiaodong Zhang, *SCOPE: Scalable Consistency Maintenance in Structured P2P Systems*, Proceedings of the 24th IEEE INFOCOM Conference (Miami, FL, USA), March 2005.

[CZK05] Antony Chazapis, Antonis Zissimos, and Nectarios Koziris, *A Peer-to-Peer Replica Management Service for High-Throughput Grids*, Proceedings of the 2005 International Conference on Parallel Processing (ICPP-05) (Oslo, Norway), June 2005.

[CZKT06] Antony Chazapis, Antonis Zissimos, Nectarios Koziris, and Panayiotis Tsanakas, *Replica Management Services on the Grid: Evolving from a Centralized Design to a Fully Distributed, Scalable and Fault-tolerant Peer-to-Peer Infrastructure*, Grid Technologies: Emerging from Distributed Architectures to Virtual Organizations (M. P. Bekakos, G. A. Gravvanis, and H. R. Arabnia, eds.), WIT Press, 2006, pp. 107–138.

[DHA03] Anwitaman Datta, Manfred Hauswirth, and Karl Aberer, *Updates in Highly Unreliable, Replicated Peer-to-Peer Systems*, In Proceedings of the 23rd IEEE International Conference on Distributed Computing Systems (ICDCS'03) (Providence, RI, USA), May 2003.

[DHJ+05] Anwitaman Datta, Manfred Hauswirth, Renault John, Roman Schmidt, and Karl Aberer, *Range Queries in Trie-Structured Overlays*, Proceedings of the 5th International Conference on Peer-to-Peer Computing (P2P2005) (Konstanz, Germany), August 2005.

[DKK+01] Frank Dabek, M. Frans Kaashoek, David Karger, Robert Morris, and Ion Stoica, *Wide-area Cooperative Storage with CFS*, Proceedings of the 18th ACM Symposium on Operating Systems Principles (SOSP-18) (Alberta, Canada), October 2001.

[DR01] Peter Druschel and Antony Rowstron, *PAST: A Large-Scale, Persistent Peer-to-Peer Storage Utility*, Proceedings of the 8th Workshop on Hot Topics in Operating Systems (HotOS-VIII) (Elmau/Oberbayern, Germany), May 2001.

[DZD+03] Frank Dabek, Ben Zhao, Peter Druschel, John Kubiatowicz, and Ion Stoica, *Towards a Common API for Structured Peer-to-Peer Overlays*, Proceedings of the 2nd International Workshop on Peer-to-Peer Systems (IPTPS'03) (Berkeley, CA, USA), February 2003.

[FK99] Ian Foster and Carl Kesselman (eds.), *The Grid: Blueprint for a New Computing Infrastructure*, Morgan-Kaufmann, 1999.

[FKT01] Ian Foster, Carl Kesselman, and Steven Tuecke, *The Anatomy of the Grid: Enabling Scalable Virtual Organizations*, International Journal of Supercomputer Applications **15** (2001), no. 3.

[Fre] *Pastry: A Substrate for Peer-to-Peer Applications*, Available online: http://freepastry.rice.edu/.

[GAA03] Abhishek Gupta, Divyakant Agrawal, and Amr El Abbadi, *Approximate Range Selection Queries in Peer-to-Peer Systems*, Proceedings of the 1st Biennial Conference on Innovative Data Systems Research (CIDR 2003) (Asilomar, CA, USA), January 2003.

[GAH05] Ali Ghodsi, Luc Onana Alima, and Seif Haridi, *Symmetric Replication for Structured Peer-to-Peer Systems*, Proceedings of the 3rd International Workshop on Databases, Information Systems and Peer-to-Peer Computing (DBISP2P 2005) (Trondheim, Norway), August 2005.

[GBMS08] Alejandra González-Beltrán, Peter Milligan, and Paul Sage, *Range queries over skip tree graphs*, Computer Communications **31** (2008), no. 2, 358 – 374.

[GS04] Jun Gao and Peter Steenkiste, *An Adaptive Protocol for Efficient Support of Range Queries in DHT-Based Systems*, Proceedings of the 12th IEEE International Conference on Network Protocols (ICNP 2004) (Berlin. Germany), October 2004.

[GYGM04] Prasanna Ganesan, Beverly Yang, and Hector Garcia-Molina, *One Torus to Rule Them All: Multi-dimensional Queries in P2P Systems*, Proceedings of the 7th International Workshop on the Web and Databases (WebDB 2004) (Paris, France), June 2004.

[HKRZ02] Kirsten Hildrum, John Kubiatowicz, Satish Rao, and Ben Y. Zhao, *Distributed Object Location in a Dynamic Network*, Proceedings of the Fourteenth ACM Symposium on Parallel Algorithms and Architectures (SPAA) (Manitoba, Canada), August 2002.

[HW02] Steven Hazel and Brandon Wiley, *Achord: A Variant of the Chord Lookup Service for Use in Censorship Resistant Peer-to-Peer Publishing Systems*, Proceedings of the 1st International Workshop on Peer-to-Peer Systems (IPTPS'02) (Cambridge, MA, USA), March 2002.

[KBC+00] John Kubiatowicz, David Bindel, Yan Chen, Steven Czerwinski, Patrick Eaton, Dennis Geels, Ramakrishna Gummadi, Sean Rhea, Hakim Weatherspoon, Westley Weimer, Chris Wells, and Ben Zhao, *OceanStore: An Architecture for Global-Scale Persistent Storage*, Proceedings of the 9th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS 2000) (Cambridge, MA, USA), November 2000.

[KBFc04] Thomas Karagiannis, Andre Broido, Michalis Faloutsos, and Kc claffy, *Transport Layer Identification of P2P Traffic*, Proceedings of the 2004 Internet Measurement Conference (IMC 2004) (Sicily, Italy), October 2004.

[LCLC09] Dongsheng Li, Jiannong Cao, Xicheng Lu, and Keith C.C. Chen, *Efficient Range Query Processing in Peer-to-Peer Systems*, IEEE Transactions on Knowledge and Data Engineering **21** (2009), no. 1, 78 – 91.

[LK00] J. K. Lawder and P. J. H. King, *Using Space-Filling Curves for Multi-dimensional Indexing*, Proceedings of the 17th British National Conferenc on Databases (BNCOD 17) (Exeter, UK), July 2000.

[LLCZ04] Shi-Ding Lin, Qiao Lian, Ming Chen, and Zheng Zhang, *A Practical Distributed Mutual Exclusion Protocol in Dynamic Peer-to-Peer Systems*, Proceedings of the 3rd International Workshop on Peer-to-Peer Systems (IPTPS'04) (San Diego, CA, USA), February 2004.

[LLK+07] Jinwon Lee, Hyonik Lee, Seungwoo Kang, Su Myeon Kim, and Junehwa Song, *CISS: An Efficient Object Clustering Framework for DHT-based Peer-to-Peer Applications*, Computer Networks **51** (2007), no. 4, 1072 – 1094.

[LLSR02] Jiang Lan, Xiaotao Liu, Prashant Shenoy, and Krithi Ramamritham, *Consistency Maintenance in Peer-to-Peer File Sharing Networks*, In Proceedings of 3rd IEEE Workshop on Internet Applications (WIAPP '03) (San Jose, CA, USA), June 2002.

[LLW05] Dongsheng Li, Xicheng Lu, and Jie Wu, *FissionE: A Scalable Constant Degree and Low Congestion DHT Scheme Based on Kautz Graphs*, Proceedings of the 24th IEEE INFOCOM Conference (Miami, FL, USA), March 2005.

[LMR02] Nancy Lynch, Dahlia Malkhi, and David Ratajczak, *Atomic Data access in Content Addressable Networks, A Position Paper*, Proceedings of the 1st International Workshop on Peer-to-Peer Systems (IPTPS'02) (Cambridge, MA, USA), March 2002.

[LRL04] L. Liu, K.D. Ryu, and K.W. Lee, *Supporting Efficient Keyword-Based File Search in Peer-to-Peer File Sharing Systems*, Tech. Report RC23145 (W0403-068), IBM Research, 2004.

[LSP82] Leslie Lamport, Robert Shostak, and Marshall Pease, *The Byzantine Generals Problem*, ACM Transactions on Programming Languages and Systems (TOPLAS) **4** (1982), no. 3, 382–401.

[MJFS01] Bongki Moon, H. V. Jagadish, Christos Faloutsos, and Joel H. Saltz, *Analysis of the Clustering Properties of the Hilbert Space-Filling Curve*, IEEE Transactions on Knowledge and Data Engineering **13** (2001), no. 1, 124 – 141.

[MM02] Petar Maymounkov and David Mazières, *Kademlia: A Peer-to-Peer Information System Based on the XOR Metric*, Proceedings of the 1st International Workshop on Peer-to-Peer Systems (IPTPS'02) (Cambridge, MA, USA), March 2002.

[MMGC02] Athicha Muthitacharoen, Robert Morris, Thomer M. Gil, and Benjie Chen, *Ivy: A Read/Write Peer-to-Peer File System*, Proceedings of 5th Symposium on Operating Systems Design and Implementation (OSDI'02) (Boston, MA, USA), December 2002.

[MMR99] Dahlia Malkhi, Yishay Mansour, and Michael K. Reiter, *On Diffusing Updates in a Byzantine Environment*, Proceedings of the 18th IEEE Symposium on Reliable Distributed Systems (SRDS'99) (Lausanne, Switzerland), October 1999.

[nPLGS04] Adina Crainiceanu nad Prakash Linga, Johannes Gehrke, and Jayavel Shanmugasundaram, *Querying Peer-to-Peer Networks Using P-trees*, Proceedings of the 7th International Workshop on the Web and Databases (WebDB 2004) (Paris, France), June 2004.

[ns] *The Network Simulator - ns-2*, http://www.isi.edu/nsnam/ns/.

[p2p] *A Simulator for Peer-to-Peer Protocols*, Available online: http://pdos.csail.mit.edu/p2psim/.

[Pug90] William Pugh, *Skip lists: A probabilistic alternative to balanced trees*, Communications of the ACM **33** (1990), no. 6, 668 – 676.

[RB03] Mema Roussopoulos and Mary Baker, *CUP: Controlled Update Propagation in Peer-to-Peer Networks*, In Proceedings of the 2003 USENIX Annual Technical Conference (San Antonio, TX, USA), June 2003.

[RD01] Antony Rowstron and Peter Druschel, *Pastry: Scalable, Decentralized Object Location and Routing for Large-Scale Peer-to-Peer Systems*, Proceedings of the IFIP/ACM International Conference on Distributed Systems Platforms (Middleware 2001) (Heidelberg, Germany), November 2001.

[RF02] Matei Ripeanu and Ian Foster, *Mapping the Gnutella Network: Macroscopic Properties of Large-Scale Peer-to-Peer Systems*, Proceedings of the 1st International Workshop on Peer-to-Peer Systems (IPTPS'02) (Cambridge, MA, USA), March 2002.

[RFH+01] Sylvia Ratnasamy, Paul Francis, Mark Handley, Richard Karp, and Scott Shenker, *A Scalable Content-Addressable Network*, Proceedings of the ACM SIGCOMM '01 Conference (San Diego, CA, USA), August 2001.

[RL03] Rodrigo Rodrigues and Barbara Liskov, *Rosebud: A Scalable Byzantine-Fault-Tolerant Storage Architecture*, Tech. Report LCS TR/932, MIT, 2003.

[RMF+00] Frank Ramsak, Volker Markl, Robert Fenk, Martin Zirkel, Klaus Elhardt, and Rudolf Bayer, *Integrating the UB-Tree into a Database System Kernel*, Proceedings of the 26th International Conference on Very Large Data Bases (VLDB 2000) (Cairo, Egypt), September 2000.

[RRHS04] Sriram Ramabhadran, Sylvia Ratnasamy, Joseph M. Hellerstein, and Scott Shenker, *Brief Announcement: Prefix Hash Tree*, Proceedings of the 23rd Annual ACM Symposium on Principles of Distributed Computing (PODC 2004) (St. John's, Newfoundland, Canada), July 2004.

[SMK+01] Ion Stoica, Robert Morris, David Karger, M. Frans Kaashoek, and Hari Balakrishnan, *Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications*, Proceedings of the ACM SIGCOMM '01 Conference (San Diego, CA, USA), August 2001.

[SOTZ05] Yanfeng Shu, Beng Chin Ooi, Kian-Lee Tan, and Aoying Zhou, *Supporting multi-dimensional range queries in peer-to-peer systems*, Proceedings of the 5th International Conference on Peer-to-Peer Computing (P2P2005) (Konstanz, Germany), August 2005.

[SP06] Angelo Spognardi and Roberto Di Pietro, *A Formal Framework for the Performance Analysis of P2P Networks Protocols*, Proceedings of the Third International Workshop on Hot Topics in Peer-to-Peer Systems (HotP2P 2006) (Rhodes Island, Greece), April 2006.

[SP08] Cristina Schmidt and Manish Parashar, *Squid: Enabling Search in DHT-based Systems*, Journal of Parallel and Distributed Computing **68** (2008), no. 7, 962 – 975.

[SSR08] Thorsten Schütt, Florian Schintke, and Alexander Reinefeld, *Range queries on structured overlay networks*, Computer Communications **31** (2008), no. 2, 280 – 291.

[STS02] Christopher Stein, Michael Tucker, and Margo Seltzer, *Building a Reliable Mutable File System on Peer-to-Peer Storage*, Proceedings of the International Workshop on Reliable Peer-to-peer Distributed Systems (RPPDS) (Osaka, Japan), October 2002.

[TH81] H. Tropf and H. Herzog, *Multidimensional Range Search in Dynamically Balanced Trees*, Applied Informatics **2/1981** (1981), 71 – 77.

[THS06] Egemen Tanin, Aaron Harwood, and Hanan Samet, *Using a distributed quadtree index in peer-to-peer networks*, The VLDB Journal **16** (2006), no. 2, 165 – 178.

[Vel93] Martin G. Velasquez, *A Survey of Distributed Mutual Exclusion Algorithms*, Tech. Report CS-93-116, Colorado State University, 1993.

[WDKS04] Zhijun Wang, Sajal K. Das, Mohan Kumar, and Huaping Shen, *Update Propagation Through Replica Chain in Decentralized and Unstructured P2P Systems*, Proceedings of the 4th International Conference on Peer-to-Peer Computing (P2P2004) (Zurich, Switzerland), August 2004.

[ZDCK07] Antonis Zissimos, Katerina Doka, Antony Chazapis, and Nectarios Koziris, *GridTorrent: Optimizing Data Transfers in the Grid with Collaborative Sharing*,

Proceedings of the 11th Panhellenic Conference on Informatics (PCI'2007) (Patras, Greece), May 2007.

[Zha04] Zheng Zhang, *The Power of DHT as a Logical Space*, In Proceedings of the 10th IEEE International Workshop on Future Trends of Distributed Computing Systems (FTDCS'04) (Suzhou, China), May 2004.

[ZSL+05] Hai Zhuge, Xiaoping Sun, Jie Liu, Erlin Yao, and Xue Chen, *A Scalable P2P Platform for the Knowledge Grid*, IEEE Transactions on Knowledge and Data Engineering **17** (2005), no. 12, 1721–1736.

[ZSLS06] Changxi Zheng, Guobin Shen, Shipeng Li, and Scott Shenker, *Distributed Segment Tree: Support of Range Query and Cover Query over DHT*, Proceedings of the 5th International Workshop on Peer-to-Peer Systems (IPTPS'06) (Santa Barbara, CA, USA), February 2006.

Δημοσιεύσεις

ΚΕΦΑΛΑΙΑ ΒΙΒΛΙΩΝ

- **Replica management services on the Grid: Evolving from a centralized design to a fully distributed, scalable and fault-tolerant peer-to-peer infrastructure**, από τους *Αντώνιο Χαζάπη*, Αντώνιο Ζήσιμο, Νεκτάριο Κοζύρη και Παναγιώτη Τσανάκα, στο βιβλίο *Grid Technologies: Emerging from Distributed Architectures to Virtual Organizations* (ISBN: 1-84564-055-1), σελ. 107-138. του εκδοτικού οίκου WIT Press, το 2006.

ΔΙΕΘΝΗ ΠΕΡΙΟΔΙΚΑ

- **Replica-aware, multi-dimensional range queries in Distributed Hash Tables**, από τους *Αντώνιο Χαζάπη*, Αθανασία Ασίκη, Γεώργιο Τσουκαλά, Δημήτριο Τσουμάκο και Νεκτάριο Κοζύρη, στο περιοδικό *Computer Communications*, από τον εκδοτικό οίκο Elsevier. **ΥΠΟ ΚΡΙΣΗ**

ΔΙΕΘΝΗ ΣΥΝΕΔΡΙΑ

- **Optimizing Data Management in Grid Environments**, από τους Αντώνιο Ζήσιμο, Αικατερίνη Δόκα, *Αντώνιο Χαζάπη*, Δημήτριο Τσουμάκο και Νεκτάριο Κοζύρη, στα πρακτικά του συνεδρίου *11th International Symposium on Distributed Objects, Middleware, and Applications (DOA'09)*, που πραγματοποιήθηκε στη Βιλαμούρα της Πορτογαλίας, τον Νοέμβριο του 2009.
- **XOROS: A mutable Distributed Hash Table**, από τους *Αντώνιο Χαζάπη* και Νεκτάριο Κοζύρη, στα πρακτικά του συνεδρίου *5th International Workshop on Databases, Information Systems and Peer-to-Peer Computing (DBISP2P 2007)*, που πραγματοποιήθηκε στη Βιέννη της Αυστρίας, τον Σεπτέμβριο του 2007.
- **Global-scale peer-to-peer file services with DFS**, από τους *Αντώνιο Χαζάπη*, Γεώργιο Τσουκαλά, Γεώργιο Βεριγάκη, Κορνήλιο Κούρτη, Αριστείδη Σωτηρόπουλο και Νεκτάριο Κοζύρη, στα πρακτικά του συνεδρίου *8th IEEE/ACM International Conference on Grid Computing (Grid 2007)*, που πραγματοποιήθηκε στο Austin, Texas των Η.Π.Α., τον Σεπτέμβριο του 2007.

- **GridTorrent: Optimizing data transfers in the Grid with collaborative sharing** από τους Αντώνιο Ζήσιμο, Αικατερίνη Δόκα, *Αντώνιο Χαζάπη* και Νεκτάριο Κοζύρη, στα πρακτικά του συνεδρίου *11th Panhellenic Conference in Informatics (PCI2007)*, που πραγματοποιήθηκε στην Πάτρα, τον Μάϊο του 2007.

- **A peer-to-peer replica management service for high-throughput Grids**, από τους *Αντώνιο Χαζάπη*, Αντώνιο Ζήσιμο και Νεκτάριο Κοζύρη, στα πρακτικά του συνεδρίου *2005 International Conference on Parallel Processing (ICPP05)*, που πραγματοποιήθηκε στο Όσλο της Νορβηγίας, τον Ιούνιο του 2005.

- **Storing and locating mutable data in structured P2P overlay networks**, από τους *Αντώνιο Χαζάπη* και Νεκτάριο Κοζύρη, στα πρακτικά του συνεδρίου *10th Panhellenic Conference on Informatics (PCI2005)*, που πραγματοποιήθηκε στον Βόλο, τον Νοέμβριο του 2005.