Πανεπιστήμιο Πατρών, Πολυτεχνική Σχολή
Τμήμα Μηχανικών Η/Υ & Πληροφορικής

**Ανάπτυξη και Αξιοποίηση Καινοτόμων Συστημάτων
Παρακολούθησης, Ελέγχου και Εξεύρεσης Καθολικών
Χαρακτηριστικών και Ιδιοτήτων των Οντοτήτων σε Δίκτυα
Ομοτίμων Εταίρων**

Νικόλαος Ντάρμος

Δίπλωμα (Τμήμα Ηλεκτρονικών Μηχ/κών & Μηχ/κών Η/Υ, Πολυτεχνείο Κρήτης) 2002
Μ.Δ.Ε. (Τμήμα Μηχ/κών Η/Υ & Πληροφορικής, Πανεπιστήμιο Πατρών) 2004

Διδακτορική Διατριβή

Προς το Τμήμα Μηχ/κών Η/Υ & Πληροφορικής
της Πολυτεχνικής Σχολής του Πανεπιστημίου Πατρών
στα πλαίσια της εκπλήρωσης των υποχρεώσεων για την απονομή
Διδακτορικού Διπλώματος

Πανεπιστήμιο Πατρών

Ιανουάριος 2008

Copyright

Νικόλαος Ντάρμος

2008

Ευχαριστίες

Θα ήθελα να ευχαριστήσω από καρδιάς τον επιβλέποντά μου, καθ. Παναγιώτη (Peter) Τριανταφύλλου. Πρωτογνωριστήκαμε όταν ήμουν φοιτητής του στο Πολυτεχνείο Κρήτης στα Χανιά. Όντας εντελώς ξένος με την επιστήμη των υπολογιστών, ήταν η διδασκαλία του και το πάθος του για τα ερευνητικά του ενδιαφέροντα, αλλά και ο χαρακτήρας του ως άνθρωπος, που με τράβηξαν και με κέρδισαν. Η συνεργασία μας από τότε υπήρξε πολύπλευρη και πολυεπίπεδη και πάντα θετική. Σήμερα τον θεωρώ κάτι παραπάνω από απλά δάσκαλο, συνεργάτη, φίλο.

Θα ήθελα επίσης να ευχαριστήσω τα μέλη της τριμελούς συμβουλευτικής επιτροπής, αναπ. καθ. Ιωάννη Γαροφαλάκη και καθ. Παύλο Σπυράκη, καθώς και τα υπόλοιπα μέλη της επταμελούς εξεταστικής επιτροπής, καθ. Εμμανουήλ Βαρβαρίγο, αναπ. καθ. Χρήστο Ζαρολιάγκη, καθ. Ιωάννη Ιωαννίδη και καθ. Τιμολέων Σελλή, για την τιμή που μου έκαναν. Ακόμα, θα ήθελα να ευχαριστήσω τον καθ. Gerhard Weikum του Ινστιτούτου Πληροφορικής Max Planck και τους μεταπτυχιακούς και διδακτορικούς του φοιτητές Sebastian Michel, Matthias Bender και Christian Zimmer για την συνεργασία τους.

Ένα μεγάλο ευχαριστώ πάει προς τους συναδέλφους και φίλους Γιάννη Αικατερινίδη, Γρηγόρη Πράσινο και Θεώνη Πιτουρά για την συνεργασία τους και κυρίως για την παρέα τους. Επίσης, ένα αντίστοιχα μεγάλο ευχαριστώ προς τους φίλους μου Γιώργο Βερνάρδο, Θανάση Βλέτση, Βασίλη Ηλιάδη, Κανέλα Καλίγωση, Μάνο Καπρίτσο, Διονύση Κουμουρά, Νίκο Μπάλα και Θανάση Παραθυρά που με βοήθησαν και μου στάθηκαν κατά τη συγγραφή της διατριβής αυτής.

Ιδιαίτερα θα ήθελα να ευχαριστήσω την οικογένειά μου και την Σοφία Σταθοπούλου για την αγάπη τους και τη συντροφιά τους, καθώς και για τη συμπαράσταση, την υπομονή και την κατανόησή τους. Πάντα με στήριζαν και με στηρίζουν και αποτελούν για μένα «καταφύγιο» και «λιμάνι». Τα όσα τους οφείλω δεν μπορούν να περιγραφούν μέσα στις λίγες αυτές λέξεις.

Η έρευνα που οδήγησε στη διατριβή αυτή χρηματοδοτήθηκε από το έργο «03ΕΔ719: Ανάπτυξη και αξιοποίηση καινοτόμων συστημάτων παρακολούθησης, ελέγχου και εξεύρεσης καθολικών χαρακτηριστικών και ιδιοτήτων των οντοτήτων σε Δίκτυα Ομοτίμων Εταίρων (P2P Networks)» στα πλαίσια του Προγράμματος Ενίσχυσης Ερευνητικού Δυναμικού (ΠΕΝΕΔ '03) που συγχρηματοδοτείται από εθνικούς, κοινοτικούς και ιδιωτικούς πόρους (ως προς τη δημόσια δαπάνη, κατά 75% από το Ευρωπαϊκό Κοινοτικό Ταμείο και κατά 25% από το Υπουργείο Ανάπτυξης – Γενική Γραμματεία Έρευνας και Τεχνολογίας) στο πλαίσιο του Μέτρου 8.3 του Ε.Π. Ανταγωνιστικότητα – Γ Κοινοτικό Πλαίσιο Στήριξης.

ΝΙΚΟΛΑΟΣ ΝΤΑΡΜΟΣ

Ανάπτυξη και Αξιοποίηση Καινοτόμων Συστημάτων Παρακολούθησης, Ελέγχου και Εξεύρεσης Καθολικών Χαρακτηριστικών και Ιδιοτήτων των Οντοτήτων σε Δίκτυα Ομοτίμων Εταίρων

Νικόλαος Ντάρμος, Υποψήφιος Διδάκτωρ

Πανεπιστήμιο Πατρών, 2008

Επιβλέπων: Παναγιώτης Τριανταφύλλου, Καθηγητής

Περίληψη

Στο πλαίσιο της διδακτορικής διατριβής αυτής, προσπαθήσαμε να δώσουμε λύση στο κεντρικό πρόβλημα του σχεδιασμού και της υλοποίησης καταναμημένων αρχιτεκτονικών συστήματος, πρωτοκόλλων και αλγορίθμων που παρέχουν την αναγκαία υποδομή για τον υπολογισμό ορισμένων βασικών καθολικών ιδιοτήτων και μεταβλητών της κατάστασης ενός Δικτύου Ομοτίμων Εταίρων (ΔΟΕ). Μπορούμε να διακρίνουμε δύο κύριες διαστάσεις καθολικών ιδιοτήτων: (α) ιδιότητες που αναφέρονται στους κόμβους του δικτύου και τα ιδιαίτερα χαρακτηριστικά τους (υπολογιστική ισχύ, συμπεριφορά, κτλ.) και (β) ιδιότητες και μεταβλητές των αντικειμένων/δεδομένων που χειρίζεται το ΔΟΕ. Για το σκοπό αυτό, κινηθήκαμε προς δύο αλληλοσυμπληρούμενες κατευθύνσεις: (α) ποσοτικοποίηση και εκμετάλλευση της ετερογένειας των κόμβων ενός ΔΟΕ, και (β) υπολογισμός εκτιμήσεων καθολικών μεταβλητών του συστήματος, με απώτερο στόχο την υποστήριξη επεξεργασίας πολύπλοκων ερωτημάτων σε συστήματα διαχείρισης δεδομένων Διαδικτυακής κλίμακας.

Στη πρώτη κατεύθυνση αρχικά ασχοληθήκαμε με το πρόβλημα της ετερογένειας στις υπολογιστικές δυνατότητες και στις συμπεριφορές των κόμβων ενός ΔΟΕ, καθ' οδόν προς ένα πιο αποδοτικό και ανθεκτικό περιβάλλον δρομολόγησης μηνυμάτων και επεξεργασίας ερωτημάτων, σε σχέση με τα κλασσικά υπάρχοντα δομημένα δικτυακά υποστρώματα ΔΟΕ Καταναμημένων Πινάκων Κατακερματισμού (DHT). Τα υποστρώματα DHT έχουν εν τέλει προσφέρει στην ερευνητική κοινότητα των ΔΟΕ μία συνδυασμένη και αποδοτική υποδομή δρομολόγησης και αποθήκευσης/ανάκτησης αντικειμένων. Ο στόχος αυτός επιτυγχάνεται με την δόμηση του συστήματος με συγκεκριμένη γεω-

μετρία, όπως αυτή προκύπτει από την διασύνδεση των κόμβων του ΔΟΕ σε κατανεμημένες δομές δρομολόγησης. Τα υποστρώματα DHT υπόσχονται λογαριθμική διάμετρο δικτύου διατηρώντας δομές δεικτοδότησης λογαριθμικού μεγέθους ως προς το πλήθος των κόμβων του ΔΟΕ, όταν αυτό είναι σε σταθερή κατάσταση. Ωστόσο, κοιτάζοντας την σχετική έρευνα στα υποστρώματα DHT, παρατηρούμε ότι υπό πολύ δυναμικές συνθήκες, η απόδοση της δρομολόγησης μηνυμάτων χειροτερεύει σε γραμμική, όσον αφορά και πάλι το πλήθος των κόμβων του ΔΟΕ – και όλα αυτά, αν και εφόσον το δίκτυο παραμένει συνδεδεμένο. Κατά βάση αυτό οφείλεται στην δυσκολία διατήρησης και ανανέωσης των δεικτών εκείνων προς άλλους κόμβους του ΔΟΕ οι οποίοι εγγυούνται τις λογαριθμικές ιδιότητες του αλγορίθμου δρομολόγησης/εντοπισμού στην σταθερή κατάσταση. Έτσι, πρώτα παρουσιάζουμε ένα νέο παράδειγμα αρχιτεκτονικής δόμησης των ΔΟΕ, το οποίο ονομάζουμε AESOP. Το AESOP αποτελείται από 3 υποστρώματα: (α) μία αρχιτεκτονική, την PLANES, η οποία εγγυάται σημαντικές βελτιώσεις στην απόδοση του συστήματος, θεωρώντας ότι μπορούμε να γνωρίζουμε και να εντοπίζουμε δυνατούς και «αλτρουιστικούς» κόμβους· (β) ένα υπόστρωμα καταγραφής και ελέγχου της συμπεριφοράς των κόμβων, ονομαζόμενο AltSeAl, το οποίο αναλαμβάνει ακριβώς αυτό τον εντοπισμό και την ταυτοποίηση των αλτρουιστικών κόμβων· και (γ) το υπόστρωμα SeAledPLANES, το οποίο διευκολύνει την συνεργασία και τον συντονισμό ανάμεσα στα πρώτα δύο υποστρώματα. Παρουσιάζουμε τα υποστρώματα αυτά καθώς και πειραματικά και αναλυτικά δεδομένα τα οποία δείχνουν σημαντικά κέρδη στην απόδοση καθώς και τα συναφή κόστη. Υπό το φως των πολλά υποσχόμενων αποτελεσμάτων αυτών, προτείνουμε αυτό το αρχιτεκτονικό παράδειγμα ως τον τρόπο δόμησης των ΔΟΕ του μέλλοντος.

Ακόμα, ασχολούμαστε με το πρόβλημα της αποδοτικής επεξεργασίας ερωτημάτων εύρους σε ΔΟΕ βασισμένα σε DHT. Η καινοτομία της προτεινόμενης προσέγγισης βρίσκεται σε αρχιτεκτονικές, αλγορίθμους και πρωτόκολλα ταυτοποίησης και κατάλληλης εκμετάλλευσης δυνατών κόμβων των δικτύων αυτών. Η ύπαρξη τέτοιων κόμβων έχει αναλυθεί ενδελεχώς στην σχετική βιβλιογραφία και παίζει καθοριστικό ρόλο στην δόμηση των περισσότερων επιτυχημένων εφαρμογών των ΔΟΕ. Ωστόσο, ως τώρα η ετερογένεια αυτή δεν λαμβανόταν υπόψιν στην σχεδίαση αρχιτεκτονικών ΔΟΕ για επεξεργασία πολύπλοκων ερωτημάτων και ειδικά για υποστρώματα DHT. Με την εργασία μας αυτή προσπαθούμε να γεμίσουμε το κενό αυτό, βελτιστοποιώντας ερωτήματα εύρους. Επιτυγχάνουμε σημαντικές βελτιώσεις στην απόδοση του συστήματος, εξαιτίας του ότι (α) καταφέρνουμε να μειώσουμε σημαντικά τον αριθμό των μηνυμάτων που απαιτούνται για την επεξεργασία των ερωτημάτων αυτών και (β) αποφεύγουμε τους κινδύνους και τις ανεπάρκειες των προσεγγίσεων που βασίζονται για την επεξεργασία ερωτημάτων εύρους σε αδύνατους κόμβους του ΔΟΕ, με μικρή επεξεργαστική ισχύ, αποθηκευτικό χώρο και εύρος ζώνης δικτύου, καθώς και με ασταθή συνδεσιμότητα με το υπόλοιπο δίκτυο. Παρουσιάζουμε επίσης πειραματικά αποτελέσματα τα οποία επαληθεύουν τα παραπάνω.

Κατόπιν, ασχολούμαστε με την κατανεμημένη εκτίμηση καθολικών μεταβλητών συστημάτων ΔΟΕ, όπως ο πληθάρειος κατανεμημένων πολυσυνόλων, η επεξεργασία καθολικών συναθροιστικών ερωτημάτων και η διατήρηση ιστογραμμάτων επί δεδομένων κατανεμημένων σε όλους τους κόμβους του ΔΟΕ, ώστε να επιτρέψουμε την μεταφορά τεχνικών βελτιστοποίησης ερωτημάτων από τα κεντρικοποιημένα περιβάλλοντα στον ευρέως κατανεμημένο χώρο των συστημάτων διαχείρισης δεδομένων Διαδικτυακής κλίμακας.

Η αποδοτική επεξεργασία ερωτημάτων στα παραδοσιακά συστήματα βάσεων δεδομένων βασίζεται στην διατήρηση στατιστικών επί των δεδομένων αυτών. Για τις κεντρικοποιημένες βάσεις υπάρχει μία πλούσια παρακαταθήκη ερευνητικών αποτελεσμάτων όσον αφορά τα στατιστικά αυτά, από απλά συναθροιστικά ερωτήματα ως πιο πολύπλοκες δομές περιλήψεων των δεδομένων όπως οι συνόψεις κατακερματισμού και τα ιστογράμματα. Από την άλλη, για κατανεμημένα συστήματα Διαδικτυακής κλίμακας η διατήρηση τέτοιων στατιστικών παρουσιάζει ακόμα σημαντικές προκλήσεις. Με το τμήμα αυτό της ερευνητικής μας δραστηριότητας προσπαθούμε να εξοπλίσουμε τα συστήματα διαχείρισης δεδομένων ΔΟΕ με τέτοιου είδους στατιστικά εργαλεία, με έμφαση στην

διατήρηση της κλιμακωσιμότητάς τους.

Αρχικά, παρουσιάζουμε τις Κατανεμημένες Συνόψεις Κατακερματισμού (DHS): έναν καινοτόμο, πλήρως κατανεμημένο μηχανισμό, ικανό για την παροχή εκτιμήσεων για τον πληθάρθμο πολυσυνόλων αντικειμένων σε ένα δομημένο δίκτυο επικάλυψης ομοτίμων εταίρων. Το DHS είναι, εξ όσων γνωρίζουμε, η πρώτη πραγματικά κατανεμημένη μεταφορά των Συνόψεων Κατακερματισμού – ενός πιθανοτικού μηχανισμού καταμέτρησης – μαζί με τα συναφή πρωτόκολλα και αλγορίθμους, στο περιβάλλον των DHT. Ο προτεινόμενος σχεδιασμός: (α) μπορεί να εφαρμοστεί σε οποιοδήποτε δίκτυο επικάλυψης DHT, (β) κατανέμει ομοιόμορφα τον φόρτο στους κόμβους του δικτύου, (γ) παρέχει πιθανοτικές εγγυήσεις για την ορθότητα και την ακρίβεια των υπολογιζόμενων εκτιμήσεων, (δ) προσφέρει επιλογές ανάμεσα στην ακρίβεια και το κόστος διαχείρισης, (ε) επιφέρει μικρή επιβάρυνση όσον αφορά τις απαιτήσεις σε εύρος ζώνης δικτύου, αποθηκευτικό χώρο και επεξεργαστικό χρόνο, ενώ (στ) παρέχει την δυνατότητα για καταμέτρηση με και χωρίς αποκλεισμό διπλοτύπων, (ζ) σέβεται την αυτονομία των κόμβων του δικτύου και κλιμακώνεται σε μεγάλο αριθμό τους και (η) δεν απαιτεί την ύπαρξη επιπλέον δομών δικτυακής δεικτοδότησης και δρομολόγησης. Παρουσιάζουμε επίσης διάφορες σχεδιαστικές επιλογές, οι οποίες επιτρέπουν την ανταλλαγή ανάμεσα στην ακρίβεια των παρεχόμενων εκτιμήσεων, την απόδοση ως προς τον απαιτούμενο αριθμό μηνυμάτων και την εξισορρόπηση στην κατανομή του φόρτου στους κόμβους του ΔΟΕ. Κατόπιν, βασιζόμαστε στα αποτελέσματα αυτά και συνεισφέρουμε αποδοτικούς, ακριβείς και πλήρως κατανεμημένους αλγορίθμους οι οποίοι μπορούν να υπολογίσουν βασικά συναθροιστικά ερωτήματα, όπως τα Count, CountDistinct, Sum και Average. Δείχνουμε ακόμα πως μπορούν να κατασκευαστούν κατανεμημένα ιστογράμματα διαφόρων ειδών, από τα απλά ιστογράμματα τύπου Equi-Width και Average-Shifted Equi-Width, ως και πιο πολύπλοκα ιστογράμματα όπως τα Equi-Depth, Max-Diff(V,F) και Compressed(V,F). Τέλος, παρουσιάζουμε μία πλήρη υλοποίηση ανοιχτού κώδικα των παραπάνω εργαλείων, καθώς και μία εμπειριστατωμένη πειραματική αξιολόγησή τους, όσον αφορά τους τομείς της απόδοσης, της ακρίβειας και της κλιμακωσιμότητας.

Περιεχόμενα

Ευχαριστίες	iii
Περίληψη	v
Κατάλογος Σχημάτων	xii
Κατάλογος Αλγορίθμων	xv
Κεφάλαιο 1 Εισαγωγή	1
1.1 Ορισμός του Προβλήματος	2
1.2 Συνεισφορά	4
1.3 Δομή	7
Κεφάλαιο 2 Σχετικές Εργασίες	9
2.1 Κατανεμημένοι Πίνακες Κατακερματισμού	9
2.2 Δρομολόγηση σε ΔΟΕ	10
2.3 Επεξεργασία Ερωτημάτων σε ΔΟΕ	11
2.3.1 Ερωτήματα Ισότητας	11
2.3.2 Ερωτήματα Εύρους	12
2.3.3 Συναθροιστικά Ερωτήματα	12
2.3.4 Ερωτήματα Equi-Join	14
2.3.5 Πολυπλοκότερα Ερωτήματα	14
2.3.6 Βελτιστοποίηση Ερωτημάτων για ΔΟΕ	15
2.4 Σύνοψη Παραδοσιακών Μεθόδων Βελτιστοποίησης Ερωτημάτων	15
2.4.1 Ιστογράμματα	16
2.4.2 Συνόψεις Κατακερματισμού	17
Κεφάλαιο 3 Δίκτυα Επικάλυψης ΔΟΕ Βασιζόμενα σε Αλτρουιστές	21
3.1 Σχεδιάζοντας Διαστρωματωμένα, Αλτρουιστικά ΔΟΕ	23
3.1.1 Δημιουργία Συστάδων Χρησιμοποιώντας Συνεπή Κατακερματισμό	24
3.1.2 Δρομολόγηση	25
3.1.3 Επιλογές & Επεκτάσεις Δόμησης Δικτύου	26
3.1.4 Αποτελεσματική Προσαρμογή σε Τοπολογικές Αλλαγές	30
3.1.5 Δυναμική Προσαρμογή Ρυθμίσεων: Χρειάζεται Ακόμα ;	30
3.1.6 Σημεία Συσσωρευμένης Δρομολόγησης	30
3.1.7 Ευρωστία	31
3.2 AltSeAl: Εγγύηση Αλτρουισμού	31
3.2.1 Το Επίπεδο Επίβλεψης/Καταγραφής	31
3.2.2 Χάρες και Παράπονα	32

3.3	SeAled PLANES: AltSeAl και AltNets	33
3.4	Επεξεργασία Ερωτημάτων Εύρους Βασιζόμενη σε Αλτρουιστές	34
3.4.1	Μειτρητές Απόδοσης Κόμβων και η Σχέση Απόδοσης Κόμβων	35
3.4.2	Είσοδος στο RangeGuard	35
3.4.3	Έξοδος από το RangeGuard	37
3.4.4	Επεξεργασία Ερωτημάτων Εύρους Χρησιμοποιώντας RGs	37
3.4.5	Μετατροπές στο Δίκτυο DHT-ΔΤ	38
3.4.6	Κατανομή Φόρτου	39
Κεφάλαιο 4	Κατανομημένες Συνόψεις Κατακερματισμού	41
4.1	Προσεγγίσεις Κατανομημένων Συνόψεων	43
4.1.1	Η Προσέγγιση Ραντεβού	44
4.1.2	Η Προσέγγιση DHS	45
4.2	Δρομολόγηση Μηνυμάτων DHS	47
4.2.1	Επαναληπτική και Αναδρομική Δρομολόγηση Μηνυμάτων	47
4.2.2	Σύντομες Διαδρομές Δρομολόγησης	50
4.3	Εισαγωγές στο DHS	50
4.3.1	Μεμονωμένες Εισαγωγές	50
4.3.2	Μαζικές Εισαγωγές	52
4.3.3	Ανανεώσεις/Διαγραφές στο DHS	53
4.3.4	Δημιουργία Αντιγράφων σε Γειτονικούς Κόμβους	54
4.3.5	Απόκρυψη Μηνυμάτων DHS στην Κίνηση Διαχείρισης του DHT	54
4.4	Αύξηση της Ακρίβειας του DHS	54
4.5	Καταμέτρηση με το DHS	55
4.5.1	Βασικός Έλεγχος Θέσεων	56
4.5.2	Σειριακός και Δυναδικός Έλεγχος Θέσεων	56
4.5.3	Λανθάνουσα Αποθήκευση Bits	60
4.5.4	Λάθη και Επαναλήψεις	60
4.6	Κάτω Φράγματα	62
4.7	Επιλογή Μήκους Πινάκων και Συνάρτησης Διασποράς	62
4.8	Σύνοψη Σχεδιαστικών Επιλογών	65
4.9	Κατανομημένη Επεξεργασία Συναθροιστικών Ερωτημάτων	65
4.9.1	Count-Distinct	66
4.9.2	Count	67
4.9.3	Node Cardinality	67
4.9.4	Sum και Average	68
4.9.5	Minimum και Maximum	68
4.10	Ιστογράμματα σε Συστήματα Διαχείρισης Δεδομένων ΔΟΕ	68
4.10.1	Ιστογράμματα Equi-Width	68
4.10.2	Ιστογράμματα Average Shifted Equi-Width	69
4.10.3	Υπολογισμός Πολυπλοκότερων Ιστογραμμάτων	69
4.10.4	Συζήτηση	71
4.11	Εφαρμογές στην Βελτιστοποίηση Ερωτημάτων ΔΟΕ	72
Κεφάλαιο 5	Πειραματική Αξιολόγηση	75
5.1	Δίκτυα Επικάλυψης Βασιζόμενα σε Αλτρουιστές	75
5.1.1	Απόδοση του AltSeAl	75
5.1.2	Απόδοση του RangeGuard	76
5.2	Κατανομημένες Συνόψεις Κατακερματισμού	80

5.2.1	Μεθοδολογία	80
5.2.2	Αποτελέσματα DHS	83
5.2.3	Συναθροιστικά Ερωτήματα	98
Κεφάλαιο 6	Συμπεράσματα & Μελλοντικές Επεκτάσεις	117
	Βιβλιογραφία	121

Κατάλογος Σχημάτων

2.1	Εισαγωγή Αντικειμένων στη Σύνοψη Κατακερματισμού: Περίπτωση Μονού Πίνακα	18
2.2	Εισαγωγή Αντικειμένων στη Σύνοψη Κατακερματισμού: Περίπτωση Πολλαπλών Πινάκων	19
2.3	Καταμέτρηση Αντικειμένων με Χρήση Σύνοψης Κατακερματισμού	20
3.1	Παράδειγμα Δικτύου με $N=30$, $C=6$, $S=5$	24
3.2	Δημιουργία Συστάδων σε ένα ΔΟΕ Βασισμένο σε Αλτρουιστές – Ένα Δαχτυλίδι Chord από Δαχτυλίδια Chord	25
3.3	Πολυεπίπεδο ΔΟΕ Βασισμένο σε Αλτρουιστές	27
3.4	Η Αρχιτεκτονική RangeGuard. Οι κόμβοι RG δημιουργούν ένα δεύτερο δικό τους δαχτυλίδι DHT-ΔΤ, παίρνοντας υπ' ευθύνη τους (χρησιμοποιώντας συνεπή κατακερματισμό) διαστήματα κόμβων του κατώτερου δαχτυλιδιού	34
3.5	Επεξεργασία Ερωτημάτων Εύρους με και χωρίς τους Κόμβους RG	38
4.1	Δομικά Στοιχεία του Συστήματος Στόχου. Όλοι οι αλγόριθμοι που παρουσιάζονται εδώ, χτίζονται πάνω από μία κατανεμημένη σύνοψη, η οποία λειτουργεί πάνω από ένα σύστημα DHT	43
4.2	Παράδειγμα Προσέγγισης Ραντεβού. Ο κόμβος 48 ορίζεται ως ο κόμβος-ραντεβού και όλοι οι υπόλοιποι κόμβοι του στέλνουν τα σχετικά δεδομένα τους	45
4.3	Αντιστοίχιση Θέσεων DHS σε Κόμβους του ΔΟΕ	46
4.4	Παράδειγμα Δρομολόγησης στο DHS για Πράξεις Πολλαπλών Θέσεων. Ο κόμβος 48 εκτελεί μία πράξη η οποία προσπελαύνει τις θέσεις 0, 1 και 2	48
4.5	Παράδειγμα Προσέγγισης Βασισμένης στο DHS. Οι κόμβοι εισάγουν τις πλειάδες τους στο DHS και ο φόρτος μοιράζεται στους κόμβους του δικτύου επικάλυψης	51
5.1	Απόδοση του Μηχανισμού Καταγραφής	76
5.2	Δικτυακό και Αποθηκευτικό Κόστος του AltSeAl	77
5.3	Αριθμός Βημάτων ανά Ερώτημα Εύρους (Λογαριθμική Αποτύπωση)	78
5.4	Λόγος Υψηλότερου-Χαμηλότερου Φόρτου	79
5.5	Μέσος Αριθμός Βημάτων Εισαγωγής ανά Αντικείμενο Εισόδου	85
5.6	Μέσος Φόρτος Εισαγωγής ανά Αντικείμενο Εισόδου	87
5.7	Κατανομή Φόρτου Εισαγωγής: Συντελεστής Gini	88
5.8	Μέσος Αριθμός Βημάτων ανά Ερώτημα – Περιπτώσεις χωρίς Λανθάνουσα Αποθήκευση Bits Αποτελεσμάτων	90
5.9	Μέσος Αριθμός Βημάτων ανά Ερώτημα – Περιπτώσεις με Λανθάνουσα Αποθήκευση Αποτελεσμάτων	91
5.10	Μέσος Φόρτος Κόμβων ανά Ερώτημα – Περιπτώσεις χωρίς Λανθάνουσα Αποθήκευση Αποτελεσμάτων	93

5.11	Μέσος Φόρτος Κόμβων ανά Ερώτημα – Περιπτώσεις με Λανθάνουσα Αποθήκευση Αποτελεσμάτων	94
5.12	Κατανομή Φόρτου Ερωτημάτων (Επαναληπτική Δρομολόγηση) – Συντελεστής Gini .	95
5.13	Κατανομή Φόρτου Ερωτημάτων (Αναδρομική Δρομολόγηση) – Συντελεστής Gini . .	96
5.14	Κατανομή Φόρτου Ερωτημάτων (Διατήρηση Αντιγράφων) – Συντελεστής Gini	97
5.15	Κατανομή Φόρτου Ερωτημάτων (Λανθάνουσα Αποθήκευση Αποτελεσμάτων) – Συντελεστής Gini	99
5.16	Μέσος Όρος % Λάθους Εκτίμησης για Ερωτήματα Count και Sum/Average σε Σχέση με Κεντριοποιημένες Προσεγγίσεις	100
5.17	Αριθμός Μηνυμάτων/Αλμάτων για την Εισαγωγή Αντικειμένων στην Κατανεμημένη Σύνοψη Χρησιμοποιώντας Προσέγγιση Ραντεβού Χωρίς Διατήρηση Αντιγράφων και Προσέγγιση DHS	101
5.18	Αριθμός Μηνυμάτων/Αλμάτων για τον Υπολογισμό των Ερωτημάτων Count, Sum και Average Χρησιμοποιώντας Προσέγγιση Ραντεβού Χωρίς Διατήρηση Αντιγράφων και Προσέγγιση DHS	101
5.19	Συντελεστής Gini (όσο μικρότερο τόσο καλύτερο) και Συντελεστής Fairness Index (όσο μεγαλύτερο τόσο καλύτερο – λογαριθμική κλίματα στον κατακόρυφο άξονα) του Φόρτου των Κόμβων για την Εισαγωγή Πλειάδων στην Κατανεμημένη Σύνοψη στην Προσέγγιση Ραντεβού	102
5.20	Συντελεστής Gini (όσο μικρότερο τόσο καλύτερο) και Συντελεστής Fairness Index (όσο μεγαλύτερο τόσο καλύτερο – λογαριθμική κλίματα στον κατακόρυφο άξονα) του Φόρτου των Κόμβων για την Εισαγωγή Πλειάδων στην Κατανεμημένη Σύνοψη για Ερωτήματα Count στην Προσέγγιση DHS	102
5.21	Συντελεστής Gini (όσο μικρότερο τόσο καλύτερο) και Συντελεστής Fairness Index (όσο μεγαλύτερο τόσο καλύτερο) του Φόρτου των Κόμβων για την Εισαγωγή Πλειάδων στην Κατανεμημένη Σύνοψη για Ερωτήματα Sum και Average στην Προσέγγιση DHS (λογαριθμική κλίματα στον κατακόρυφο άξονα)	103
5.22	Χρονική Εξέλιξη των Συντελεστών Gini και Fairness Index για τον Υπολογισμό των Τριών Ειδών Κατανεμημένων Ερωτημάτων	103
5.23	Συντελεστής Gini και Συντελεστής Fairness Index του Φόρτου των Κόμβων για την Εισαγωγή Αντικειμένων στην Κατανεμημένη Σύνοψη για το Ερώτημα Count για Συνεχώς Αυξανόμενο Αριθμό Κόμβων Ραντεβού που Διαμοιράζονται τον Κατανεμημένο Υπολογισμό (είτε μέσω αντιγράφων ή λόγων πολλαπλών παράλληλων ερωτημάτων) .	104
5.24	Κόστος Μηνυμάτων για το Ερώτημα Count Χρησιμοποιώντας Προσέγγιση Ραντεβού με 50 Κόμβους Ραντεβού	104
5.25	Συνολικός Φόρτος Υπολογισμού Ερωτημάτων Όταν Πολλαπλά Ερωτήματα Επεξεργάζονται Ταυτόχρονα από το Σύστημα	106
5.26	Μέσος Φόρτος Εισαγωγής ανά Κόμβο για Ερωτήματα Count Χρησιμοποιώντας το DHS, για Διάφορες Τιμές του Πλήθους των Κόμβων στο Δίκτυο· ένα δεκαπλασιασμός του αριθμού των αντικειμένων οδηγεί σε μία μόλις κατά $\approx 60\%$ αύξηση στο πλήθος των επισκέψεων σε κάθε κόμβο κατά την φάση εισαγωγής	107
5.27	Μέγιστος Φόρτος Εισαγωγής για Ερωτήματα Count Χρησιμοποιώντας το DHS, για Διάφορες Τιμές του Πλήθους των Κόμβων στο Δίκτυο· ένα δεκαπλασιασμός του αριθμού των αντικειμένων οδηγεί σε μία κατά $\approx 6\times$ αύξηση στο μέγιστο πλήθος επισκέψεων σε οποιονδήποτε κόμβο κατά την φάση εισαγωγής, ενώ παραμένει ακόμα μία τάξη μεγέθους μικρότερο από το αντίστοιχο πλήθος επισκέψεων της προσέγγισης ραντεβού	108

5.28	Μέσος Αριθμός Μηνυμάτων/Αλμάτων ανά Κόμβο για Εισαγωγή Πλειάδων για Ερωτήματα Count Χρησιμοποιώντας το DHS, για Διάφορες Τιμές του Πλήθους των Κόμβων στο Δίκτυο· ένα δεκαπλάσιασμός του αριθμού των αντικειμένων οδηγεί σε μία μόλις κατά $\approx 35\%$ αύξηση στο πλήθος των μηνυμάτων κάθε κόμβου για την εισαγωγή όλων των πλειάδων του στο DHS	109
5.29	Μέσος Αριθμός Κόμβων που Επισκέφθηκε ο Αλγόριθμος Υπολογισμού των Ερωτημάτων Count Χρησιμοποιώντας το DHS, για Διάφορες Τιμές του Πλήθους των Κόμβων στο Δίκτυο	110
5.30	Μέσο Κόστος Αλμάτων ανά Ερώτημα για Ερωτήματα Count Χρησιμοποιώντας το DHS, για Διάφορες Τιμές του Πλήθους των Κόμβων στο Δίκτυο	111
5.31	Αποτελέσματα Υπολογισμού Ιστογραμμάτων: 1,000 κόμβοι, 10,000,000 αντικείμενα, τιμές στο διάστημα $[0, 1,000,000)$, 100 κελιά στο αρχικό ιστόγραμμα DHS, 10 κελιά ανά υπολογιζόμενο ιστόγραμμα	112
5.32	Αποτελέσματα Υπολογισμού Ιστογραμμάτων: 1,000 κόμβοι, 10,000,000 πλειάδες, τιμές στο διάστημα $[0, 1,000)$, 1,000 κελιά στο αρχικό ιστόγραμμα DHS, 10 κελιά ανά υπολογιζόμενο ιστόγραμμα	114

Κατάλογος Αλγορίθμων

4.1	Αλγόριθμος Αντιστοίχισης: Τυχαία επιλογή αναγνωριστικού στο ζητούμενο διαμέρισμα και πρόσβαση στον αντίστοιχο υπεύθυνο κόμβο	47
4.2	Αλγόριθμος <i>IBR</i> : Επαναληπτική εκτέλεση μίας λειτουργίας <i>op</i> του DHS για πολλαπλές θέσεις bits	49
4.3	Αλγόριθμος <i>RBR</i> : Αναδρομική εκτέλεση μίας λειτουργίας <i>op</i> στο DHS για πολλαπλές θέσεις bits. Το <i>result</i> αποθηκεύει το μερικό αποτέλεσμα κάθε βήματος της αναδρομής	49
4.4	Αλγόριθμος Επεξεργασίας Μηνυμάτων «single-check-bit(s)»	56
4.5	Αλγόριθμος <i>estimateSBP</i> : Εκτίμηση του αριθμού διακριτών στοιχείων σε ένα πολυσύνολο με χρήση DHS με σειριακό έλεγχο θέσεων. Το $R[\beta]$ αποθηκεύει τις θέσεις των λιγότερο σημαντικών 0-bits για τον PCSA ή των πιο σημαντικών 1-bits για τον superLogLog εκτιμητή αντίστοιχα	57
4.6	Αλγόριθμος <i>estimateBBP</i> : Εκτίμηση του αριθμού διακριτών στοιχείων σε ένα πολυσύνολο με DHS με έλεγχο θέσεων δυαδικής αναζήτησης	59
4.7	Εκτίμηση του Αριθμού Διακριτών Στοιχείων σε ένα Πολυσύνολο με Χρήση DHS. Τα β , λ και τ περιγράφονται στον πίνακα 4.1	63

Κεφάλαιο 1

Εισαγωγή

Τα Δίκτυα Ομοτίμων Εταίρων (Peer-to-Peer (P2P) networks ή ΔΟΕ) πρωτοεμφανίστηκαν ως ένα μέσο ανταλλαγής αρχείων και υπολογιστικών κύκλων ανάμεσα σε χρήστες του Διαδικτύου. Με τον καιρό, εξελίχθηκαν από την άναρχη λειτουργία των πρώιμων αρχιτεκτονικών μικρού κόσμου [45, 131] στα προηγμένης τεχνολογίας δομημένα δίκτυα δεδομένων του σήμερα. Ο κύριος παράγοντας που συνετέλεσε σε αυτή την αλλαγή ήταν η έλευση των δικτύων επικάλυψης (overlay networks) Κατανεμημένων Πινάκων Κατακερματισμού [32, 84, 109, 121, 132] (Distributed Hash Tables ή DHT). Τα συστήματα αυτά κλιμακώνονται σε μεγάλους αριθμούς κόμβων και μπορούν να διαχειρίζονται έναν ακόμα μεγαλύτερο αριθμό δεδομένων, ενώ παρέχουν πιθανοτικές εγγυήσεις για το βαθμό αξιοπιστίας, ανεκτικότητας σε λάθη, διαθεσιμότητας της πληροφορίας και απόδοσης, υπό την παρουσία λαθών ή δυναμικής συμπεριφοράς κόμβων και ανομοιόμορφης κατανομής των δεδομένων και του φόρτου. Πιο συγκεκριμένα, τα υποσπρώματα DHT παρέχουν δρομολόγηση σε σταθερή κατάσταση σε $O(\log N)$ βήματα, σε ένα δίκτυο N κόμβων, διατηρώντας δομές δεικτοδότησης μεγέθους $O(\log N)$ δεικτών ανά κόμβο. Τα χαρακτηριστικά αυτά αποτέλεσαν ένα σημαντικό βήμα σε σύγκριση με τις πρώιμες προσεγγίσεις των «αδόμητων» υποσπρωμάτων ΔΟΕ [21, 45]. Στα αδόμητα δίκτυα, τα σχετικά κόσθη είναι σαφώς υψηλότερα και δεν υπάρχουν εγγυήσεις ως προς την αποδοτικότητα ή ακόμα και αυτό τον εντοπισμό των ζητούμενων δεδομένων.

Ως φυσική εξέλιξη της ευρείας αποδοχής των τεχνολογιών ΔΟΕ από τους τελικούς χρήστες και την βιομηχανία και των επιθυμητών ιδιοτήτων των δομημένων δικτύων επικάλυψης, η ακαδημαϊκή κοινότητα έχει αρχίσει να ερευνά την δυνατότητα της χρήσης τέτοιων δικτύων ως το υπόβαθρο για συστήματα ευρέως κατανεμημένων βάσεων δεδομένων [7, 8, 13, 26, 40, 41, 48, 52, 55, 56, 59, 61, 65, 73, 93, 101, 113, 125]. Έτσι, τα ΔΟΕ έχουν πλέον ξεφύγει από τον απλό διαμοιρασμό αρχείων και εξελίσσονται ταχέως σε μία ισχυρή υποδομή, ικανή να υποστηρίξει συστήματα διαχείρισης δεδομένων τεράστιας κλίμακας. Σε αυτό τον χώρο δεν υπάρχει κάποια κεντρική εξουσία όσον αφορά τη διαχείριση των δεδομένων και των προσβάσεων σε αυτά, σε αντίθεση με τα παραδοσιακά (κεντροποιημένα ή κατανεμημένα) συστήματα βάσεων δεδομένων, όπου οι διαχειριστές των βάσεων είναι επιφορτισμένοι με αυτό το έργο. Η ίδια έλλειψη κεντροποιημένου ελέγχου ισχύει και για τους υπολογιστικούς και δικτυακούς πόρους. Σε περιβάλλον ΔΟΕ, οι χρήστες/κόμβοι διαχειρίζονται οι ίδιοι τα δεδομένα τους και τους υπολογιστικούς/αποθηκευτικούς/δικτυακούς πόρους τους κατά βούληση, οι οποίοι αποτελούν ελάχιστο τμήμα των συνολικών δεδομένων και πόρων που είναι διαθέσιμα στο συνολικό σύστημα. Για να λειτουργεί όλο το ΔΟΕ αποδοτικά, οι κόμβοι πρέπει να είναι συνδεδεμένοι και να συνεργάζονται, συνεισφέροντας προς τον κοινό στόχο επίτευξης υψηλής απόδοσης και διαθεσιμότητας. Ο δυνητικά τεράστιος αριθμός κόμβων, η αντίστοιχα πολύ μεγάλης κλίμακας κατανομή των δεδομένων και των πόρων και η αυτονομία της συμπεριφοράς των κόμβων, καθιστούν το έργο της διαχείρισης πόρων, δεδομένων και προσβάσεων μία πολύ ενδιαφέρουσα

πρόκληση.

Ο πρωταρχικός μας σκοπός είναι να προσφέρουμε υπηρεσίες διαχείρισης δεδομένων και επεξεργασίας πολύπλοκων ερωτημάτων σε εφαρμογές χτισμένες πάνω από μία δικτυακή υποδομή μεγάλης κλίμακας. Το ιδιαίτερο χαρακτηριστικό και η κύρια πρόκληση είναι το γεγονός ότι οι όποιες λύσεις θα πρέπει να βασίζονται στην εκμετάλλευση όλων των κατανεμημένων πόρων που είναι διαθέσιμοι στο ΔΟΕ ενώ παράλληλα να παρέχονται «παραδοσιακές» λειτουργίες διαχείρισης δεδομένων. Σχετικές εργασίες έχουν προσφέρει λύσεις για ένα μεγάλο αριθμό προβλημάτων, από αρχιτεκτονικές και αλγορίθμους για αναζήτηση δεδομένων, ως επεξεργασία ερωτημάτων εύρους και ενοποίησης πληροφορίας και έχουν αρχίσει να εξετάζουν τρόπους υποστήριξης ερωτημάτων συνάθροισης ή ένωσης. Το γεγονός αυτό είναι άλλη μία ένδειξη της σημασίας που δίνει η κοινότητά μας στην ικανότητα υποστήριξης εφαρμογών εντατικής διαχείρισης δεδομένων πάνω από υποδομές μεγάλης κλίμακας. Οι μελλοντικές εφαρμογές θα πρέπει να χρησιμοποιούν όλες τους διαθέσιμους πόρους. Έτσι, η ικανότητα εντοπισμού και εκμετάλλευσης των ελεύθερων πόρων του δικτύου είναι βασική για την επιτυχία τέτοιου είδους εφαρμογών.

1.1 Ορισμός του Προβλήματος

Βασικό συστατικό στην επιτυχία αυτών των στόχων είναι η δυνατότητα υπολογισμού και διατήρησης καθολικών ιδιοτήτων και μεταβλητών των οντοτήτων που συνυπάρχουν σε ένα ΔΟΕ. Η ερευνητική κοινότητα του χώρου των δικτύων ομοτίμων εταιρών, όμως, απέρριπτε εξ αρχής κάτι τέτοιο, καθώς η καθολική γνώση θεωρείται αδύνατη σε ένα σύστημα Διαδικτυακής κλίμακας και υψηλής δυναμικότητας, όπως τα ΔΟΕ. Στα πλαίσια της διδακτορικής διατριβής αυτής, δώσαμε ένα σύνολο λύσεων στο κεντρικό πρόβλημα του σχεδιασμού και της υλοποίησης κατανεμημένων αρχιτεκτονικών συστήματος, πρωτοκόλλων και αλγορίθμων που να παρέχουν την αναγκαία υποδομή για τον υπολογισμό τέτοιων βασικών καθολικών ιδιοτήτων και μεταβλητών της κατάστασης ενός δικτύου ομοτίμων εταιρών (ΔΟΕ). Μπορούμε να διακρίνουμε δύο κύριες διαστάσεις καθολικών ιδιοτήτων: (α) ιδιότητες που αναφέρονται στους κόμβους του δικτύου και τα ιδιαίτερα χαρακτηριστικά τους (υπολογιστική ισχύ, συμπεριφορά, κτλ.) και (β) ιδιότητες και μεταβλητές των αντικειμένων/δεδομένων που χειρίζεται το ΔΟΕ. Για το σκοπό αυτό, κινηθήκαμε προς δύο αλληλοσυμπληρούμενες κατευθύνσεις: (α) ποσοτικοποίηση και εκμετάλλευση της ετερογένειας των κόμβων ενός ΔΟΕ, και (β) υπολογισμός εκτιμήσεων καθολικών μεταβλητών του συστήματος, με απώτερο στόχο την υποστήριξη επεξεργασίας πολύπλοκων ερωτημάτων σε συστήματα διαχείρισης δεδομένων Διαδικτυακής κλίμακας.

Η έρευνα στα ΔΟΕ και στην επεξεργασία ερωτημάτων στο περιβάλλον αυτό βασιζόταν ανέκαθεν στην θεώρηση ότι όλοι οι κόμβοι του δικτύου είναι ίσοι. Ωστόσο, μελέτες πραγματικών συστημάτων ΔΟΕ μεγάλης κλίμακας [5, 114, 117, 128] (κυρίως συστημάτων διαμοιρασμού αρχείων) έχουν αποδείξει ότι ο κόσμος των ΔΟΕ απέχει πολύ από το τέλειο. Εκτός από την ετερογένεια στις υπολογιστικές δυνατότητες των κόμβων, υπάρχει και το βαθύτερο πρόβλημα της ετερογένειας της επιδεικνυόμενης συμπεριφοράς· η μεγάλη πλειοψηφία των κόμβων (περισσότεροι από το 70% [5, 117]) αποδείχθηκε πως δεν προσέφεραν τίποτα στο σύστημα. Αυτό είναι πραγματικά πολύ κακό νέο για τα υποσυστήματα DHT, καθώς σημαίνει πως η μεγάλη πλειοψηφία των κόμβων συνδέονται στο δίκτυο και το εγκαταλείπουν ύστερα από μικρό χρονικό διάστημα [114, 128]. Τα καλά νέα είναι πως υπάρχει ένα μη αμελητέο ποσοστό κόμβων οι οποίοι αποδείχθηκε πως ήταν «αλτρουιστές» (συνεισέφεραν μεγάλο τμήμα των πόρων τους, έμενα συνδεδεμένοι για μεγάλα χρονικά διαστήματα, κτλ.)

Αρχικά, λοιπόν, αντιμετωπίζουμε το πρόβλημα της ετερογένειας στις υπολογιστικές δυνατότητες και στις συμπεριφορές των κόμβων ενός ΔΟΕ. Στόχος μας είναι ένα πιο αποδοτικό και ανθεκτικό περιβάλλον δρομολόγησης μηνυμάτων και επεξεργασίας ερωτημάτων, σε σχέση με τα

κλασσικά υπάρχοντα δομημένα δικτυακά υποστρώματα DHT, βασισμένο στην «καθολική γνώση» της συμπεριφοράς και των υπολογιστικών δυνατοτήτων των κόμβων του δικτύου. Κοιτάζοντας την σχετική έρευνα στα υποστρώματα DHT, παρατηρούμε ότι υπό πολύ δυναμικές συνθήκες, η απόδοση της δρομολόγησης μηνυμάτων χειροτερεύει σε γραμμική, όσον αφορά και πάλι το πλήθος των κόμβων του ΔΟΕ – και όλα αυτά, αν και εφόσον το δίκτυο παραμένει συνδεδεμένο. Κατά βάση αυτό οφείλεται στην δυσκολία διατήρησης και ανανέωσης των δεικτών εκείνων προς άλλους κόμβους του ΔΟΕ οι οποίοι εγγυούνται τις λογαριθμικές ιδιότητες του αλγορίθμου δρομολόγησης/εντοπισμού στην σταθερή κατάσταση.

Πιο συγκεκριμένα, θεωρούμε ότι:

1. Η συνύφανση των διαφορετικών δυνατοτήτων και συμπεριφορών των κόμβων στη δομή των δομημένων αρχιτεκτονικών ΔΟΕ θα οδηγήσει στην επίτευξη σημαντικών και σίγουρα παρατηρήσιμων βελτιώσεων στην δρομολόγηση μηνυμάτων και την επεξεργασία ερωτημάτων.
2. Οι βελτιώσεις στην δρομολόγηση πρέπει να αναφέρονται τόσο στον αριθμό των βημάτων, όσο και στο μέγεθος των δομών δεικτοδότησης και στις απαιτήσεις συντήρησής τους, καθώς και στην ανθεκτικότητά τους. Επίσης, θα πρέπει να επιτυγχάνονται χωρίς να μεταφέρεται κόστος σε άλλες φάσεις της λειτουργίας του συστήματος (π.χ. στην φάση της «σταθεροποίησης»).
3. Οι βελτιώσεις αυτές θα πρέπει να επιτυγχάνονται τόσο όταν το σύστημα είναι σε σταθερή κατάσταση όσο και όταν βρίσκεται υπό δυναμικές αλλαγές.
4. Οι αλτρουιστές και δυνατοί κόμβοι μπορούν να χρησιμοποιηθούν για να προσφέρουν τα σημαντικά αυτά οφέλη, ενώ υπάρχει απαίτηση μόνο μικρού ποσοστού αλτρουιστών στο σύνολο των κόμβων του δικτύου.
5. Ένα υπόστρωμα το οποίο θα είναι ικανό να εντοπίσει, να ταυτοποιήσει και να διαχειριστεί τους αλτρουιστές και ισχυρούς αυτούς κόμβους θα βοηθήσει σίγουρα σημαντικά στο έργο αυτό.
6. Ως αποτέλεσμα, απαιτείται μία δομή η οποία να διευκολύνει την συνεργασία μίας αρχιτεκτονικής βασισμένης σε αλτρουιστές και ενός υποστρώματος καταγραφής και ελέγχου που εντοπίζει τους αλτρουιστικούς κόμβους, ούτως ώστε να κάνουμε το επόμενο βήμα στις δομημένες αρχιτεκτονικές ΔΟΕ.

Θεωρούμε ότι τα δεδομένα που αποθηκεύονται και διαχειρίζονται από το ΔΟΕ είναι δομημένα σε σχέσεις/πίνακες και, όπως συνηθίζεται στην σχετική βιβλιογραφία [59, 125], διασκορπισμένα (και ίσως αντιγραφμένα) σε όλους τους κόμβους του δικτύου. Κάθε σχέση R αποτελείται από $(k + l)$ χαρακτηριστικά (attributes) ή στήλες, $R(a_1, \dots, a_k, b_1, \dots, b_l)$. οι στήλες a_i κάθε πλειάδας t χρησιμοποιούνται ως μονοδιάστατοι δείκτες της t ενώ δεν διατηρείται καθόλου πληροφορία δεικτοδότησης για τις στήλες b_i . Κάθε στήλη a_i χαρακτηρίζεται από το πεδίο τιμών της $t.a_i.D : \{t.a_i.v_{min}, t.a_i.v_{max}\}$, το οποίο αποτελείται από τη ελάχιστη και μέγιστη τιμή της στήλης. Κάθε πλειάδα t στην R ταυτοποιείται μοναδικά από το πρωτεύον κλειδί της $t.key$. Αυτό το κλειδί μπορεί να είναι οποιοδήποτε από τα χαρακτηριστικά της πλειάδας ή μπορεί να υπολογίζεται με κάποιον άλλο τρόπο (π.χ. με βάση τις τιμές κάποιου συνδυασμού των χαρακτηριστικών της). Σε αυτό το περιβάλλον, μία πλειάδα t τοποθετείται στους κόμβους του ΔΟΕ που υποδεικνύονται από το πρωτεύον κλειδί της $t.key$ καθώς και από καθένα από τα χαρακτηριστικά a_i της t . Εναλλακτικά, αντί για να αποθηκεύονται αντίγραφα της πλειάδας σε όλους αυτούς του κόμβους, μπορεί να αποθηκεύεται ολόκληρη η πλειάδα μόνο σε έναν κόμβο (π.χ. σε αυτόν που αντιστοιχεί στον πρωτεύον κλειδί της) και μόνο δείκτες προς αυτό τον κόμβο στις υπόλοιπες τοποθεσίες.

Η αποδοτική επεξεργασία ερωτημάτων στα παραδοσιακά συστήματα βάσεων δεδομένων βασίζεται στην διατήρηση στατιστικών επί των δεδομένων αυτών. Για τις κεντροποιημένες βάσεις

υπάρχει μία πλούσια παρακαταθήκη ερευνητικών αποτελεσμάτων όσον αφορά τα στατιστικά αυτά, από απλά συναθροιστικά ερωτήματα ως πιο πολύπλοκες δομές περιλήψεων των δεδομένων όπως οι συνόψεις κατακερματισμού και τα ιστογράμματα. Από την άλλη, για κατανεμημένα συστήματα Διαδικτυακής κλίμακας η διατήρηση τέτοιων στατιστικών παρουσιάζει ακόμα σημαντικές προκλήσεις. Οι κυριότεροι περιορισμοί στους οποίους υπόκειται η λύση του προβλήματος αυτού είναι:

- Π1 Αποδοτικότητα: ο αριθμός των κόμβων που χρειάζεται να προσπελαστούν κατά την καταμέτρηση πρέπει να είναι μικρός, έτσι ώστε το σύστημα να έχει μικρή καθυστέρηση (latency) και μικρές απαιτήσεις εύρους ζώνης δικτύου.
- Π2 Κλιμακωσιμότητα, η οποία φαινομενικά αντιτίθεται στην αποδοτικότητα: αυθαίρετα πολλοί κόμβοι ενδεχομένως να προσθέτουν στοιχεία στο πολυσύνολο, κάτι το οποίο προϋποθέτει την ύπαρξη μίας ευρέως κατανεμημένης λύσης η οποία θα αποφεύγει τα προβλήματα κλιμακωσιμότητας, διαθεσιμότητας και παρουσίας συμφορήσεων των κλασσικών αρχιτεκτονικών πελάτη-εξυπηρετητή.
- Π3 Εξισορροπημένος φόρτος πρόσβασης και αποθήκευσης: η καταμέτρηση καθώς και όλες οι σχετικές με αυτήν διαδικασίες θα πρέπει να κατανέμουν τον φόρτο δίκαια στους κόμβους του δικτύου.
- Π4 Ακρίβεια: η καταμέτρηση θα πρέπει να παρέχει εκτίμηση του ζητούμενου πληθάριου η οποία να είναι ρυθμιζόμενα ακριβής και ανθεκτική (παρουσία δυναμικών συμπεριφορών και αποτυχιών των κόμβων) και
- Π5 Απλότητα και ευκολία ολοκλήρωσης με υπάρχοντα συστήματα: δεν θα πρέπει να απαιτούνται ειδικές δομές δεικτοδότησης του δικτύου ειδικά για την εφαρμογή καταμέτρησης.
- Π6 Υποστήριξη μη καταμέτρησης διπλοτύπων: η καταμέτρηση θα πρέπει να μπορεί να εφαρμοστεί τόσο σε κλασσικά σύνολα όσο και σε πολυσύνολα και να υποστηρίζει καταμέτρηση ή μη πολλαπλών αντιγράφων κάθε αντικειμένου.

1.2 Συνεισφορά

Πρώτα παρουσιάζουμε ένα νέο παράδειγμα αρχιτεκτονικής δόμησης των ΔΟΕ, το οποίο ονομάζουμε AESOP και το οποίο βασίζεται στην ιδέα του εντοπισμού και της χρησιμοποίησης των αλτρουιστικών κόμβων. Έτσι επιτυγχάνουμε ξεκάθαρες εγγυήσεις απόδοσης για την σταθερή κατάσταση, αλλά πιο σημαντικά και για καταστάσεις δυναμικών αλλαγών. Πάνω και πέρα από τις όποιες βελτιώσεις στον τομέα του αριθμού των μηνυμάτων/βημάτων, εγγυόμαστε μία πιο σταθερή υποδομή, ειδικά εν μέσω μαζικών αλλαγών στην δομή και τη σύνθεση του δικτύου [112]. Το AESOP αποτελείται από 3 υποσώματα: (α) μία αρχιτεκτονική, την PLANES, η οποία εγγυάται σημαντικές βελτιώσεις στην απόδοση του συστήματος, θεωρώντας ότι μπορούμε να γνωρίζουμε και να εντοπίζουμε δυνατούς και «αλτρουιστικούς» κόμβους· (β) ένα υπόσρωμα καταγραφής και ελέγχου της συμπεριφοράς των κόμβων, ονομαζόμενο AltSeAl, το οποίο αναλαμβάνει ακριβώς αυτό τον εντοπισμό και την ταυτοποίηση των αλτρουιστικών κόμβων· και (γ) το υπόσρωμα SeAledPLANES, το οποίο διευκολύνει την συνεργασία και τον συντονισμό ανάμεσα στα πρώτα δύο υποσώματα. Παρουσιάζουμε τα υποσώματα αυτά καθώς και πειραματικά και αναλυτικά δεδομένα τα οποία δείχνουν σημαντικά κέρδη στην απόδοση καθώς και τα συναφή κόστη. Υπό το φως των πολλά υποσχόμενων αποτελεσμάτων αυτών, προτείνουμε αυτό το αρχιτεκτονικό παράδειγμα ως τον τρόπο δόμησης των ΔΟΕ του μέλλοντος.

Ακόμα, εφαρμόζουμε τις ιδέες μας αυτές και στο πρόβλημα της αποδοτικής επεξεργασίας ερωτημάτων εύρους σε ΔΟΕ βασισμένα σε DHT. Η καινοτομία της προτεινόμενης προσέγγισης

βρίσκεται σε αρχιτεκτονικές, αλγορίθμους και πρωτόκολλα ταυτοποίησης και κατάλληλης εκμετάλλευσης δυνατών κόμβων των δικτύων αυτών. Η ύπαρξη τέτοιων κόμβων έχει αναλυθεί ενδελεχώς στην σχετική βιβλιογραφία και παίζει καθοριστικό ρόλο στην δόμηση των περισσότερων επιτυχημένων εφαρμογών των ΔΟΕ. Ωστόσο, ως τώρα η ετερογένεια αυτή δεν λαμβανόταν υπόψιν στην σχεδίαση αρχιτεκτονικών ΔΟΕ για επεξεργασία πολύπλοκων ερωτημάτων και ειδικά για υποστρώματα DHT. Με την εργασία μας αυτή προσπαθούμε να γεμίσουμε το κενό αυτό, βελτιστοποιώντας ερωτήματα εύρους. Επιτυγχάνουμε σημαντικές βελτιώσεις στην απόδοση του συστήματος, εξαιτίας του ότι (α) καταφέρνουμε να μειώσουμε σημαντικά τον αριθμό των μηνυμάτων που απαιτούνται για την επεξεργασία των ερωτημάτων αυτών και (β) αποφεύγουμε τους κινδύνους και τις ανεπάρκειες των προσεγγίσεων που βασίζονται για την επεξεργασία ερωτημάτων εύρους σε αδύνατους κόμβους του ΔΟΕ, με μικρή επεξεργαστική ισχύ, αποθηκευτικό χώρο και εύρος ζώνης δικτύου, καθώς και με ασταθή συνδεσιμότητα με το υπόλοιπο δίκτυο. Παρουσιάζουμε επίσης πειραματικά αποτελέσματα τα οποία επαληθεύουν τα παραπάνω.

Με το δεύτερο τμήμα της ερευνητικής μας δραστηριότητας προσπαθούμε να εξοπλίσουμε τα συστήματα διαχείρισης δεδομένων ΔΟΕ με βασικά στατιστικά εργαλεία, με έμφαση στην διατήρηση της κλιμακωσιμότητάς τους. Αρχικά, παρουσιάζουμε τις Κατανεμημένες Συνόψεις Κατακερματισμού (DHS): έναν καινοτόμο, πλήρως κατανεμημένο, υψηλά κλιμακώσιμο, αποδοτικό και ακριβή μηχανισμό, ικανό για την παροχή εκτιμήσεων για τον πληθάρη πολυσυνόλων αντικειμένων σε ένα δομημένο δίκτυο επικάλυψης ομοτίμων εταίρων. Το DHS είναι, εξ όσων γνωρίζουμε, η πρώτη πραγματικά κατανεμημένη έκδοση των συνόψεων κατακερματισμού – ενός πιθανοτικού μηχανισμού καταμέτρησης, ο οποίος πρωτοπροτάθηκε από τους Flajolet και Martin [35] και πιο πρόσφατα από τους Durand και Flajolet [33] – μαζί με τους αντίστοιχους αλγορίθμους και πρωτόκολλα για περιβάλλοντα Κατανεμημένων Πινάκων Κατακερματισμού. Επιπλέον, απ' όσο ξέρουμε, το DHS είναι ο πρώτος κατανεμημένος μηχανισμός καταμέτρησης που ικανοποιεί και τους πέντε περιορισμούς που παρουσιάσαμε νωρίτερα. Ο προτεινόμενος σχεδιασμός: (α) μπορεί να εφαρμοστεί σε οποιοδήποτε δίκτυο επικάλυψης DHT, (β) κατανέμει ομοιόμορφα τον φόρτο στους κόμβους του δικτύου, (γ) παρέχει πιθανοτικές εγγυήσεις για την ορθότητα και την ακρίβεια των υπολογιζόμενων εκτιμήσεων, (δ) προσφέρει επιλογές ανάμεσα στην ακρίβεια και το κόστος διαχείρισης και (ε) επιφέρει μικρή επιβάρυνση όσον αφορά τις απαιτήσεις σε εύρος ζώνης δικτύου, αποθηκευτικό χώρο και επεξεργαστικό χρόνο, όταν χρησιμοποιείται για καταμέτρηση πληθάρη ευρέως κατανεμημένων πολυσυνόλων.

Παρουσιάζουμε διάφορες σχεδιαστικές επιλογές, οι οποίες επιτρέπουν την ανταλλαγή ανάμεσα στην ακρίβεια των παρεχόμενων εκτιμήσεων, την απόδοση ως προς τον απαιτούμενο αριθμό μηνυμάτων και την εξισορρόπηση στην κατανομή του φόρτου στους κόμβους του ΔΟΕ. Επιπροσθέτως της βασικής λειτουργικότητας των DHS, συνεισφέρουμε:

1. Βελτιώσεις στο πρωτόκολλο δρομολόγησης μηνυμάτων, με την μορφή:

- (α') Ενός αναδρομικού σχήματος, το οποίο επιτρέπει την μείωση του συνολικού αριθμού βημάτων και την πραγματοποίησή τους πάνω από συνδέσμους μικρότερης καθυστέρησης.
- (β') Ενός μηχανισμού «σύντομης διαδρομής», ο οποίος εκμεταλλεύεται τις εγγραφές του πίνακα δρομολόγησης του κάθε κόμβου, ουσιαστικά μετατρέποντας τα περισσότερα μηνύματα των DHS σε πράξεις ενός βήματος.

2. Βελτιώσεις στον αλγόριθμο ελέγχου θέσεων των πινάκων των DHS: πιο συγκεκριμένα:

- (α') Έναν αλγόριθμο ελέγχου δυαδικής αναζήτησης, ως εναλλακτική στον υπάρχοντα σειριακό αλγόριθμο ελέγχου.
- (β') Έναν μηχανισμό κρυφής αποθήκευσης bits αποτελεσμάτων, ο οποίος – σε συνδυασμό με το αναδρομικό σχήμα δρομολόγησης – επιτρέπει την περαιτέρω μείωση του κόστους

αριθμού βημάτων και την καλύτερη διαμέριση του φόρτου στους κόμβους του δικτύου επικάλυψης.

3. Βελτιώσεις στον αλγόριθμο εισαγωγής αντικειμένων, αποτελούμενες από:

- (α') Έναν αλγόριθμο μαζικής εισαγωγής, ο οποίος ομαδοποιεί πολλά αντικείμενα και επιτρέπει την εισαγωγή τους με μία μόνο πράξη εισαγωγής.
- (β') Ένα απλό σχήμα δημιουργίας αντιγράφων soft-state με χαμηλό κόστος βημάτων, το οποίο προσδίδει μεγαλύτερη ανεκτικότητα σε λάθη και μειώνει κι άλλο το κόστος αριθμού βημάτων.
- (γ') Έναν αλγόριθμο εισαγωγής/ανανέωσης ο οποίος βασίζεται στην απόκρυψη μηνυμάτων των DHS στην διαχειριστική κίνηση του πίνακα δρομολόγησης του DHT.

Τέλος, συνεισφέρουμε μία πλήρη υλοποίηση των προτεινόμενων αλγορίθμων και πρωτοκόλλων πάνω από το σύστημα FreePastry [38], μία ελεύθερα διαθέσιμη υλοποίηση του δικτύου DHT Pastry [32] από τους συγγραφείς του, καθώς και μια εις βάθος ανάλυση της απόδοσης των αλγορίθμων και πρωτοκόλλων μας με αυτή την υποδομή. Ο πηγαίος κώδικας της υλοποίησής μας (η οποία ονομάζεται *FreeDHS*) είναι ελεύθερα διαθέσιμος στον παγκόσμιο ιστό [37].

Κατόπιν, χρησιμοποιούμε τα αποτελέσματα αυτά και συνεισφέρουμε αποδοτικούς, ακριβείς και πλήρως κατανεμημένους αλγορίθμους οι οποίοι μπορούν να υπολογίσουν βασικά συναθροιστικά ερωτήματα και στατιστικές συνόψεις. Ο κοινός παρονομαστής όλων των επιτυχημένων συστημάτων δεδομένων από τη δεκαετία του 70, είναι η διατήρηση και εκμετάλλευση στατιστικών και σχετικών συνόψεων δεδομένων επί των αποθηκευμένων δεδομένων, με σκοπό την επίτευξη σημαντικού κέρδους κατά την επεξεργασία των ερωτημάτων. Στα περισσότερα πραγματικά συστήματα, από το πρωτότυπο System-R [16], ως τα σύγχρονα εμπορικά συστήματα διαχείρισης σχεσιακών βάσεων δεδομένων όπως τα Oracle, DB2 και Informix, αυτό το είδος πληροφορίας συνήθως υλοποιείται με τη μορφή συναθροιστικών ερωτημάτων, συνόψεων κατακερματισμού και ιστογραμμάτων. Με την εργασία μας αυτή, προσφέρουμε μία εμπεριστατωμένη μελέτη της ανάπτυξης και της διατήρησης τέτοιων στατιστικών σε ένα περιβάλλον ΔΟΕ. Εξ όσων γνωρίζουμε, αυτή είναι η πρώτη δουλειά που προσπαθεί κάτι τέτοιο και η πρώτη προσπάθεια να μεταφερθούν παραδοσιακά κεντρικοποιημένα εργαλεία βελτιστοποίησης ερωτημάτων στο περιβάλλον των δικτύων διαχείρισης δεδομένων Διαδικτυακής κλίμακας.

Πιο συγκεκριμένα, οι συνεισφορές μας επίσης συμπεριλαμβάνουν:

1. Πλήρως αποκεντριοποιημένους αλγορίθμους υπολογισμού βασικών συναθροιστικών ερωτημάτων, όπως τα Count-Distinct, Count, Sum, Average, καθώς και το αντίστοιχο της καταμέτρησης σελίδων δεδομένων σε συνθήκες ΔΟΕ.
2. Πλήρως αποκεντριοποιημένες, κλιμακώσιμες και αποδοτικές μεθόδους κατασκευής πολλών ειδών ιστογραμμάτων, συμπεριλαμβανομένων των Equi-Width, Average-Shifted Equi-Width, Equi-Depth, MaxDiff(V,F) και Compressed(V,F) ιστογραμμάτων.
3. Μία πλήρη υλοποίηση των παραπάνω αλγορίθμων για κατανεμημένη επεξεργασία συναθροιστικών ερωτημάτων και για κατασκευή, διατήρηση και χρήση ιστογραμμάτων. Συγκεκριμένα, η υλοποίησή μας – ονομαζόμενη FreeSHADE από το «Statistics, Histograms, and Aggregates in a DHT-based Environment» – βασίζεται επίσης στο FreePastry και είναι διαθέσιμη ελεύθερα στον παγκόσμιο ιστό [39]. Επιπλέον, συνεισφέρουμε μία εμπεριστατωμένη μελέτη της απόδοσης των αλγορίθμων μας όσον αφορά τα λάθη εκτίμησης, το κόστος αριθμού βημάτων και την κατανομή φόρτου στους κόμβους του δικτύου. Έχουμε τέλος υλοποιήσει και την αντίπαλη λύση της προσέγγισης «ραντεβού» και συγκρίνουμε τις συνεισφορές μας με αυτή.

1.3 Δομή

Το Κεφάλαιο 2 κάνει μία επισκόπηση σχετικών εργασιών στους τομείς των αρχιτεκτονικών των δικτύων επικάλυψης ΔΟΕ και της δρομολόγησης στα δίκτυα αυτά, καθώς και στους τομείς των τεχνικών επεξεργασίας και βελτιστοποίησης ερωτημάτων σε συστήματα δεδομένων ΔΟΕ. Ακόμα παρέχει μία εισαγωγή στην βελτιστοποίηση ερωτημάτων και μία σύνοψη των ιστογραμμάτων και των συνόψεων κατακερματισμού. Στο Κεφάλαιο 3 παρουσιάζουμε το AESOP, ένα πολυεπίπεδο σχεδιασμό για την δόμηση υποστρωμάτων ΔΟΕ ούτως ώστε να βελτιωθεί η απόδοση αριθμού βημάτων, η ανοχή λαθών και η ανθεκτικότητα του αλγορίθμου δρομολόγησης. Στη συνέχεια παρουσιάζουμε την αρχιτεκτονική RangeGuard· μία λύση για την αποδοτικότερη επεξεργασία ερωτημάτων εύρους σε συστήματα διαχείρισης δεδομένων που βασίζονται σε DHT. Το Κεφάλαιο 4 παρουσιάζει την αρχιτεκτονική, τους αλγορίθμους και τα πρωτόκολλα του DHS, περιγράφοντας επίσης και την προσέγγιση ραντεβού για τον υπολογισμό συνόψεων κατακερματισμού με κατανεμημένο τρόπο. Στο ίδιο κεφάλαιο συζητούνται τρόποι υπολογισμού βασικών συναθροιστικών ερωτημάτων σε ΔΟΕ και μέθοδοι υπολογισμού πιο πολύπλοκων στατιστικών δομών, όπως απλά και διάφορα είδη από γνωστά πολύπλοκα ιστογράμματα. Τέλος, το Κεφάλαιο 5 παρουσιάζει την υλοποίηση και την πειραματική μελέτη του κόστους των λύσεών μας και των σχετικών βελτιώσεων και το Κεφάλαιο 6 αναφέρει τα συμπεράσματά μας και τις πιθανές μελλοντικές επεκτάσεις της εργασίας μας.

Κεφάλαιο 2

Σχετικές Εργασίες

Στα πλαίσια αυτής της διατριβής, προσπαθούμε να γεφυρώσουμε το χάσμα μεταξύ της παραδοσιακής επεξεργασίας και βελτιστοποίησης ερωτημάτων και των ΔΟΕ συστημάτων διαχείρισης βάσεων δεδομένων. Σε αυτήν την ενότητα θα παρουσιάσουμε τα εργαλεία που χρησιμοποιήσαμε για να επιτύχουμε τους στόχους μας, μια μικρή περίληψη της παραδοσιακής επεξεργασίας και βελτιστοποίησης ερωτημάτων σε ΔΟΕ, καθώς και των σχετικών στατιστικών δομών και τεχνικών.

2.1 Κατανεμημένοι Πίνακες Κατακερματισμού

Οι Κατανεμημένοι Πίνακες Κατακερματισμού (DHT) είναι μία οικογένεια δομημένων δι-κτύων ομοτίμων εταίρων τα οποία προσφέρουν μία διεπαφή και λειτουργικότητα που προσομοιάζει αυτή των κλασσικών πινάκων κατακερματισμού. Το κύριο πλεονέκτημα των DHT επί άλλων δομημένων ΔΟΕ βρίσκεται στις πιθανοτικές (υπό την παρουσία αποτυχιών των κόμβων και δυναμικής του δικτύου) εγγυήσεις που προσφέρονται από τα πρώτα. Γνωστότερα παραδείγματα τέτοιων ΔΟΕ είναι τα συστήματα Pastry [32], Chord [121], CAN [109], Tapestry [132], Kademlia [84], κτλ.

Τα DHT προσφέρουν δύο βασικούς τελεστές: *insert(key, value)* και *lookup(key)*. Οι κόμβοι λαμβάνουν αναγνωριστικά από έναν κυκλικό χώρο αναγνωριστικών και τακτοποιούνται ανάλογα με μια προκαθορισμένη γεωμετρία και μία συνάρτηση απόστασης, βασικά στοιχεία των δομημένων ΔΟΕ [50]. Τα αναγνωριστικά των κόμβων υπολογίζονται είτε ως το αποτέλεσμα μίας ασφαλούς συνάρτησης διασποράς με είσοδο κάποια πληροφορία σχετική με τον κόμβο (π.χ. της IP διεύθυνσης του κόμβου και του αριθμού της δικτυακής θύρας στην οποία δέχεται αιτήσεις η εφαρμογή ΔΟΕ), είτε ως το αποτέλεσμα μίας γεννήτριας ψευδο-τυχαίων αριθμών [84]¹. Αυτό έχει ως αποτέλεσμα μία διαμοίραση του χώρου των αναγνωριστικών των κόμβων ανάμεσα στους κόμβους του δικτύου, έτσι ώστε κάθε κόμβος είναι υπεύθυνος για μία σαφώς καθορισμένη περιοχή αναγνωριστικών. Κάθε αντικείμενο επίσης λαμβάνει ένα αναγνωριστικό από τον ίδιο αυτό χώρο – συνήθως χρησιμοποιώντας το περιεχόμενο ή κάποια άλλη πληροφορία του αντικειμένου ως είσοδο στην ίδια συνάρτηση διασποράς – και αποθηκεύεται στον κόμβο του οποίου το αναγνωριστικό είναι πιο κοντά σε αυτό του αντικειμένου, σύμφωνα με την συνάρτηση απόστασης του DHT.

Κάθε κόμβος σε ένα ΔΟΕ N κόμβων διατηρεί απευθείας συνδέσμους/δείκτες προς $O(\log N)$ άλλους κόμβους σε κατάλληλες αποστάσεις στο χώρο αναγνωριστικών, σύμφωνα με την γεωμετρία του DHT, ούτως ώστε η δρομολόγηση ανάμεσα σε οποιοδήποτε ζεύγος κόμβων να απαιτεί $O(\log N)$ βήματα στην χειρότερη περίπτωση σε σταθερή κατάσταση. Για κάθε κόμβο οι σύνδεσμοι αυτοί

¹ Μία τέτοια γεννήτρια μπορεί να χρησιμοποιηθεί ως συνάρτηση διασποράς χρησιμοποιώντας την είσοδο της συνάρτησης ως «σπόρο» της γεννήτριας και μέρος της εξόδου της ως το αποτέλεσμα της υποτιθέμενης συνάρτησης διασποράς.

συνήθως κατηγοριοποιούνται σε δύο κύριες ομάδες: (α) τους «δείκτες»· συνδέσμους προς κόμβους με αναγνωριστικά μακριά από αυτό του αρχικού κόμβου και (β) τους «προηγούμενους/επόμενους»· συνδέσμους προς κόμβους που βρίσκονται αμέσως πριν/μετά τον αρχικό κόμβο στον χώρο αναγνωριστικών. Η πρώτη ομάδα συνδέσμων χρησιμοποιούνται για να επιτευχθεί η δρομολόγηση σε $O(\log N)$ βήματα ενώ η δεύτερη εγγυάται ότι το ΔΟΕ παραμένει συνδεδεμένο. Η πληροφορία αυτή συντηρείται μέσω μηνυμάτων PING/PONG ή/και περιοδικών ελέγχων και ανανεώνεται είτε όταν κάποιος κόμβος αποχωρήσει – είτε ακολουθώντας το αντίστοιχο πρωτόκολλο είτε απότομα (π.χ. επειδή ο κόμβος αυτός βγήκε εκτός δικτύου ή εκτός λειτουργίας ή εξαιτίας διαμέρισης του δικτύου ή εξαιτίας κάποιας άλλης αποτυχίας χαμηλού επιπέδου) – ή ως τμήμα μίας περιοδικής διαδικασίας η οποία ονομάζεται «σταθεροποίηση». Κατά τη διάρκεια της περιόδου σταθεροποίησης οι κόμβοι του DHT αναζητούν, μέσω του τελεστή lookup, τους κόμβους που είναι υπεύθυνοι για τα αναγνωριστικά που αντιστοιχούν σε καθεμία από τις θέσεις στον πίνακα δρομολόγησης/δευκτοδότησης του κόμβου. Η διαδικασία αυτή έχει κόστος $O(\log^2 N)$ βημάτων/μηνυμάτων ανά κόμβο (δηλ. $O(\log N)$ βήματα για καθεμία από τις $O(\log N)$ εγγραφές του πίνακα). Επιπλέον, για λόγους ανοχής σε λάθη και αυξημένης διαθεσιμότητας, οι κόμβοι διατηρούν μία λίστα υποψηφίων κόμβων για κάθε τέτοια θέση του πίνακα δρομολόγησης, ούτως ώστε αν ένας σύνδεσμος αποτύχει να μπορεί να αντικατασταθεί άμεσα από έναν άλλο από την λίστα υποψηφίων. Τέλος, σε πολλά συστήματα DHT οι κόμβοι διατηρούν ακόμα μία λίστα με πληροφορίες για τους κόμβους με τους οποίους έχουν επικοινωνήσει κατά τη διάρκεια εκτέλεσης διαφόρων ενεργειών στο ΔΟΕ (π.χ. εισαγωγή ή αναζήτηση δεδομένων, συντήρηση πινάκων δρομολόγησης, κτλ.) αλλά οι οποίοι δεν ανήκουν ούτε στον πίνακα δρομολόγησης ούτε στην λίστα των υποψηφίων αντικαταστατών. Οι λίστες με τους λανθάνοντες αυτούς συνδέσμους συνήθως αποκρύπτονται πίσω από την σημειολογία του big-O καθώς είναι είτε σταθερού μεγέθους (ή πολλαπλάσιο του μεγέθους του πραγματικού πίνακα δρομολόγησης με κάποια σταθερά), όπως συμβαίνει με τις λίστες υποψηφίων αντικαταστατών, είτε απλά αγνοούνται κατά τη σχετική ανάλυση επειδή δεν συνεισφέρουν στο κόστος συντήρησης αλλά ανανεώνονται με τυχαίο τρόπο.

2.2 Δρομολόγηση σε ΔΟΕ

Κοιτάζοντας την σχετική έρευνα στα υποσυστήματα DHT, παρατηρούμε ότι υπό πολύ δυναμικές συνθήκες, η απόδοση της δρομολόγησης μηνυμάτων χειροτερεύει σε $O(N)$, αν και εφόσον το δίκτυο παραμένει συνδεδεμένο. Κατά βάση αυτό οφείλεται στην δυσκολία διατήρησης και ανανέωσης των δεικτών εκείνων προς άλλους κόμβους του ΔΟΕ οι οποίοι εγγυούνται τις λογαριθμικές ιδιότητες του αλγορίθμου δρομολόγησης/εντοπισμού στην σταθερή κατάσταση. Οι συγγραφείς του [78] μελέτησαν μεθόδους που εγγυούνται μονοπάτια μήκους $O(\log N)$ υπό δυναμικές συνθήκες. Για να το καταφέρουν αυτό, απαιτούνται $O(\log^2 N)$ επαναλήψεις («γύροι») «σταθεροποίησης» ανά κόμβο ανά ημιζωή του συστήματος για να ανανεώνονται οι δομές δρομολόγησης του (επόμενοι και προηγούμενοι κόμβοι και απομακρυσμένοι «δείκτες»). Ωστόσο, αυτή η προσέγγιση (α) μεταφέρει το κόστος από την φάση της δρομολόγησης στην φάση της σταθεροποίησης, (β) είναι ακριβή, με συνολικό κόστος $O(N \log^4 N)$ μηνυμάτων ανά ημιζωή συστήματος (π.χ. κάθε κόμβος σε ένα ΔΟΕ ενός εκατομμυρίου κόμβων πρέπει να εκτελεί περίπου 400 γύρους σταθεροποίησης ανά περίπου μισή ώρα) και (γ) η ανίχνευση της παρουσίας/απουσίας κόμβων με συνδέσεις δικτύου χαμηλής ταχύτητας (οι οποίοι αποτελούν και την συντριπτική πλειοψηφία) κατά την διάρκεια της φάσης σταθεροποίησης είναι πολύ χρονοβόρα και ιδιαίτερα επιρρεπής σε λάθη. Επομένως, δεδομένης της τεράστιας κλίμακας και της ιδιαίτερα δυναμικής φύσης του μεγαλύτερου ποσοστού των κόμβων, οι τρέχουσες αρχιτεκτονικές αποτυγχάνουν να εγγυηθούν την δρομολόγηση σε $O(\log N)$ βήματα στην περίπτωση υψηλής δυναμικότητας.

Επιπλέον, ακόμα και $O(\log N)$ βήματα, σε σταθερή κατάσταση και υπό την υπόθεση της

«καλής συμπεριφοράς» των κόμβων, ίσως να μην είναι αρκετά· εν τέλει, αυτά τα βήματα αναφέρονται σε μεταβάσεις ανάμεσα σε γειτονικούς κόμβους του δικτύου επικάλυψης του ΔΟΕ, με καθεμία από αυτές να μεταφράζεται σε πολλαπλά βήματα ανάμεσα σε συνδέσμους στο φυσικό δίκτυο. Επιπλέον, ακόμα κι αν τα βήματα στο φυσικό δίκτυο ήταν $O(\log N)$, όταν αυτά είναι πάνω από κόμβους με χαμηλής ταχύτητας συνδέσεις σίγουρα θα δημιουργούσαν προβλήματα απόδοσης. Τέλος, στον χώρο των DHT υπάρχει παντελής έλλειψη προσοχής στην εκμετάλλευση των ισχυρών κόμβων με σκοπό την βελτίωση της απόδοσης.

Ακόμα κι αν εξετάσουμε τις ερευνητικές προσπάθειες στα αδόμητα ΔΟΕ, παρατηρούμε μία έλλειψη σημαντικής προσοχής στην εκμετάλλευση της ετερογένειας ανάμεσα στους κόμβους [114]. Ως εξαίρεση, η εργασία [110] ασχολείται και με την εκμετάλλευση των ισχυρών κόμβων, οι οποίοι προσομοιώνονται με ένα πλήθος από μικρότερους, φανταστικούς κόμβους. Αυτή η θεώρηση μετατρέπει πολλά βήματα ανάμεσα σε αδύναμους κόμβους σε «εσωτερικά» φανταστικά βήματα μέσα σε έναν ισχυρό κόμβο. Επίσης, η εργασία [81] παρουσιάζει καταναεμμένους αλγορίθμους οι οποίοι κατευθύνουν αυξημένες ποσότητες φόρτου προς πιο ικανούς κόμβους.

Ωστόσο, η ετερογένεια σημαίνει περισσότερα από μία απλή διάκριση ανάμεσα σε ισχυρούς και αδύναμους κόμβους· υπάρχει επίσης και ετερογένεια όσον αφορά την συμπεριφορά των κόμβων και το κατά πόσο αυτοί είναι αλτρουιστές ή εγωιστές. Για παράδειγμα, μπορεί να υπάρχουν ισχυροί κόμβοι οι οποίοι δεν διαθέτουν τους πόρους τους στην κοινότητα κι έτσι δεν δρουν αλτρουιστικά. Είναι λογικό να θεωρήσει κανείς ότι οι αλτρουιστές κόμβοι θα έχουν συνήθως περισσότερες δυνατότητες (υπολογιστικές, αποθήκευσης, δικτύου, κτλ), καθώς και την βούληση να τις μοιραστούν (όταν δεν χρησιμοποιούνται) με άλλους κόμβους του δικτύου. Αυτή η θεώρηση έχει επαληθευθεί και στο [128].

Παρ' όλα αυτά, οι παραπάνω σχετικές εργασίες έχουν κάνει κάποια πρόοδο, δείχνοντας τον δρόμο προς την εκμετάλλευση των ισχυρών κόμβων. Όπως κι εμείς, κριτικάρουν τα συστήματα DHT και τα δομημένα δίκτυα εν γένει όσον αφορά την αποτυχία τους στο να ανταπεξέλθουν αποδοτικά στην υψηλή δυναμικότητα του δικτύου, όπως για παράδειγμα στις περιπτώσεις που περιγράφονται στο [20]. Ωστόσο, αυτό οδήγησε πολλούς από αυτούς τους ερευνητές στη ακραία στάση της αποστροφής προς τα δομημένα δίκτυα, το οποίο με τη σειρά του τους στέρησε τη δυνατότητα να παρέχουν λύσεις με συγκεκριμένες εγγυήσεις απόδοσης, όσον αφορά το κόστος βημάτων και σταθερότητας. Απεναντίας, εμείς ακολουθούμε μία διαφορετική οδό: προτείνουμε την προσθήκη επιπλέον δομής στα συστήματα DHT, χρησιμοποιώντας αλτρουιστικούς κόμβους. Με αυτό τον τρόπο, μπορούμε να επιτύχουμε ξεκάθαρες εγγυήσεις απόδοσης για την σταθερή κατάσταση και – πολύ σημαντικότερο – για την δυναμική κατάσταση. Πέρα και πάνω από τις όποιες βελτιώσεις στο κόστος αριθμού βημάτων, κατορθώνουμε να έχουμε μία σταθερότερη υποδομή, ειδικά σε συνθήκες βίαιων και διαρκών αλλαγών στη δομή και την σύνθεση του ΔΟΕ [112].

2.3 Επεξεργασία Ερωτημάτων σε ΔΟΕ

Η ερευνητική κοινότητα των δικτύων ομοτίμων εταιρών έχει ήδη αρχίσει να εξετάζει μεθόδους παροχής κλασσικής λειτουργικότητας συστημάτων διαχείρισης βάσεων δεδομένων πάνω από δίκτυα δεδομένων ΔΟΕ [48]. Τα πιο γνωστά από τα συστήματα αυτά, στα οποία απευθύνονται τα αποτελέσματα της διατριβής αυτής, είναι χτισμένα πάνω σε υποσπρώματα DHT.

2.3.1 Ερωτήματα Ισότητας

Τα υποσπρώματα DHT υποστηρίζουν εγγενώς ερωτήματα ισότητας (σημείου), μέσω του τελεστή *lookup()*· δεδομένου ενός περιορισμού ισότητας σε μία συγκεκριμένη στήλη μίας σχέσης, το μόνο που έχουμε να κάνουμε είναι να επισκεφθούμε τον κόμβο που είναι υπεύθυνος για την τιμή που εμφανίζεται στον περιορισμό και να συγκεντρώσουμε όλες τις πλειάδες (ή δείκτες προς αυτές)

που τον ικανοποιούν. Επομένως, η επεξεργασία τέτοιων ερωτημάτων έχει κόστος αριθμού βημάτων $O(\log N)$.

2.3.2 Ερωτήματα Εύρους

Η κατάσταση αρχίζει να περιπλέκεται αισθητά όταν έχουμε να κάνουμε με ερωτήματα εκτός από ισότητα. Ακόμα και στην απλούστερη περίπτωση των ερωτημάτων εύρους (διαστήματος), τα παραδοσιακά συστήματα DHT αποτυγχάνουν να προσφέρουν ικανοποιητικές λύσεις. Δεδομένου ενός περιορισμού εύρους σε μία συγκεκριμένη στήλη μία σχέσης, η βασική (αν όχι η μοναδική) μέθοδος επεξεργασίας ενός τέτοιου ερωτήματος είναι να εκτελεστεί ένα χωριστό ερώτημα ισότητας για καθμία από τις τιμές που εμπεριέχονται στο ζητούμενο διάστημα. Αυτό είναι άμεση απόρροια της ψευδο-τυχαιότητας της αντιστοίχισης τιμών εισόδου στους κόμβους του δικτύου και της έλλειψης καθολικής γνώσης. Επομένως, για ένα διάστημα v τιμών $c_1 \leq a_i \leq c_2$, με c_1, c_2 σταθερές, το αντίστοιχο ερώτημα εύρους θα απαιτούσε $O(v \cdot \log N)$ μηνύματα/βήματα, κάτι το οποίο είναι απαγορευτικό στις περισσότερες περιπτώσεις.

Για να υποστηριχθούν ερωτήματα εύρους με τρόπο αποδοτικό, πρέπει να χρησιμοποιηθούν συστήματα DHT Διατήρησης Τοπικότητας (ή Διατήρησης Σειράς). Στην περίπτωση αυτή, συνεχόμενες τιμές στο πεδίο τιμών αντιστοιχίζονται σε γειτονικούς κόμβους στο ΔΟΕ. Δεδομένου λοιπόν του ίδιου περιορισμού για το διάστημα v τιμών, στην περίπτωση του DHT-ΔΤ πρέπει απλά να εντοπίσουμε τον κόμβο που είναι υπεύθυνος για τον κάτω άκρο του διαστήματος (c_1) και κατόπιν να ακολουθήσουμε δείκτες ενός βήματος κάθε φορά προς τον αμέσως επόμενο κόμβο στον χώρο αναγνωριστικών έως ότου φτάσουμε στον κόμβο που είναι υπεύθυνος για το άνω άκρο του διαστήματος (c_2), συγκεντρώνοντας τα σχετικά αποτελέσματα στην πορεία. Αυτό μεταφράζεται σε ένα κόστος $O(\log(N) + v)$ μηνυμάτων/βημάτων, το οποίο είναι κατά πολύ καλύτερο του αντίστοιχου κόστους των παραδοσιακών DHT.

Όλες οι προηγούμενες εργασίες που ασχολήθηκαν με την υποστήριξη πολύπλοκων ερωτημάτων σε συστήματα DHT [7, 8, 13, 26, 40, 41, 48, 52, 56, 59, 113, 125] απέτυχαν να αναγνωρίσουν και να εκμεταλλευθούν το βασικό γεγονός ότι η ύπαρξη και η χρήση των δυνατών κόμβων του δικτύου μπορούν να οδηγήσουν σε σημαντικές βελτιώσεις στο κόστος επεξεργασίας τέτοιων ερωτημάτων. Βλέποντας το θέμα από μία συμπληρωματική γωνία, οι προηγούμενες εργασίες απέτυχαν να αναγνωρίσουν το ότι σε δίκτυα διαμοιρασμού δεδομένων μεγάλης κλίμακας υπάρχουν κόμβοι οι οποίοι είναι αδύναμοι, όσον αφορά τις υπολογιστικές, αποθηκευτικές και δικτυακές δυνατότητές τους και το ότι υπάρχουν επίσης κόμβοι με τάξεις μεγέθους περισσότερη ισχύ [114]. Η λύση που προτείνουμε αποφεύγει την παγίδα της επιφόρτισης αδύναμων κόμβων με εργασίες επεξεργασίας ερωτημάτων. Επιπλέον, ακολουθούμε μία προσέγγιση βασισμένη σε καθαρή διαχείριση δεδομένων για να εντοπίσουμε και να εκμεταλλευτούμε τους ισχυρούς αυτούς κόμβους: διατηρώντας τα μεταδεδομένα για τους συμμετέχοντες κόμβους ως μία σχέση πάνω από ένα δαχτυλίδι DHT-ΔΤ, μπορούμε να εντοπίζουμε τέτοιους κόμβους γρήγορα και με τρόπο κλιμακώσιμο και να τους χρησιμοποιούμε κατόπιν ως το κέντρο της επεξεργασίας ερωτημάτων και της δρομολόγησης μηνυμάτων.

2.3.3 Συναθροιστικά Ερωτήματα

Οι κατανεμημένες λύσεις για μέτρηση/συνάθροιση που έχουν προταθεί ως τώρα από την ερευνητική κοινότητα των ΔΟΕ, μπορούν να κατηγοριοποιηθούν ως εξής:

- Πρωτόκολλα βασισμένα σε ραντεβού.
- Πρωτόκολλα gossip.
- Πρωτόκολλα broadcast/convergecast.

- Πρωτόκολλα δειγματοληψίας.

Ο πρώτος τύπος λύσεων είναι ουσιαστικά ο πρώτος που έρχεται στο μυαλό όταν χρησιμοποιούμε ένα δομημένο δίκτυο: επιλέγουμε έναν κόμβο από το δίκτυο (π.χ. χρησιμοποιώντας τις συναρτήσεις κατακερματισμού του δικτύου) και τον χρησιμοποιούμε για να διατηρήσουμε την μετρούμενη τιμή (βλ. τον κατανεμημένο μηχανισμό μέτρησης που παρουσιάζεται στο [35]). Οι μετρητές κατακερματισμένων διαμερίσεων – όπου ο μετρούμενος χώρος είναι διαμερισμένος σε ξένα μεταξύ τους διαστήματα, και το κάθε διάστημα αντιστοιχίζεται σε ένα (σύνολο από) κόμβο(υς) στο δίκτυο – ή λύσεις βασισμένες σε κάποιο «διοργανωτή» (πολύ συχνές σε δίκτυα αισθητήρων και στην κατανεμημένη επεξεργασία ροών δεδομένων) [25, 54]) – όπου περιλήψεις των δεδομένων συγκεντρώνονται σε ένα κεντρικό σημείο συνάθροισης προς επεξεργασία – εμπίπτουν επίσης σε αυτήν την κατηγορία. Λύσεις αυτού του είδους έχουν αρκετά μειονεκτήματα, πιο σημαντικό από τα οποία είναι η πολύ κακή κλιμακωσιμότητά τους: έχοντας έναν κόμβο ανά μετρητή σημαίνει ότι αυτός ο κόμβος θα λάβει μια ειδοποίηση για κάθε ενημέρωση και ερώτημα για την τιμή του μετρητή, θυμίζοντας πιο πολύ ένα κεντροκοποιημένο σύστημα, παραβιάζοντας έτσι τον περιορισμό (Π2) της ενότητας 1.1. Επιπλέον, κάθε ένας από αυτούς τους κόμβους θα υποστεί ένα υψηλό φορτίο πρόσβασης και αποθήκευσης, παραβιάζοντας τον περιορισμό (Π3), ενώ θα μπορούσαμε να επιχειρηματολογήσουμε πως τόσο βαριά φορτωμένοι κόμβοι θα έχουν και υψηλούς χρόνους απόκρισης, παραβιάζοντας τον περιορισμό (Π1). Η χρήση ενός (σταθερού) αριθμού κόμβων για κάθε μετρητή δε λύνει το πρόβλημα, αλλά απλά μεταθέτει τα ζητήματα κλιμακωσιμότητας στο κόστος εισαγωγής αντικειμένων και ερώτησης της τιμής ενός τέτοιου μετρητή, καθώς αυτά τα μεγέθη αυξάνονται γραμμικά με τον αριθμό των κόμβων ραντεβού που μετέχουν στον υπολογισμό, ενώ παράλληλα παραβιάζει και τον περιορισμό (Π1). Παρόμοια επιχειρήματα ισχύουν και για την περίπτωση που πρέπει να επικοινωνήσουμε με πολλαπλούς κόμβους ραντεβού, σαν αποτέλεσμα της ταυτόχρονης διατήρησης πολλών μετρητών (δηλαδή της ταυτόχρονης μέτρησης πολλών ποσοτήτων).

Ο δεύτερος τύπος λύσεων [10, 67, 72, 90] συνήθως προσφέρει αδύναμες πιθανοτικές εγγυήσεις «τελικής συνεκτικότητας» (eventual consistency) για το αποτέλεσμά τους. Οι αλγόριθμοι gossip βασίζονται σε μια επαναλαμβανόμενη διαδικασία, σύμφωνα με την οποία κάθε κόμβος ανταλλάσσει πληροφορία με ένα υποσύνολο των γειτονικών του κόμβων σε κάθε επανάληψη. «Τελική συνεκτικότητα» σημαίνει ότι, υπό την παρουσία αποτυχιών και της δυναμικότητας των ΔΟΕ, ο αλγόριθμος τελικά θα συγκλίνει σε μια σταθερή κατάσταση αφού το δίκτυο το ίδιο έχει σταθεροποιηθεί. Παρόλο που οι απαιτήσεις σε bandwidth αυτών των αλγορίθμων είναι χαμηλές όταν μοιράζονται σε όλους τους κόμβους, η συνολική κατανάλωση bandwidth και ο αριθμός βημάτων κυμαίνονται συνήθως σε πολύ υψηλά επίπεδα. Ακόμα, το γεγονός ότι όλοι οι κόμβοι πρέπει να συμμετέχουν ενεργά σε ένα υπολογισμό βασισμένο σε gossip, ακόμα και αν δεν τους ενδιαφέρει, σε συνδυασμό με την ιδιότητα πολλαπλών γύρων αυτών των λύσεων, παραβιάζει τον περιορισμό (Π1), ενώ τα semantics τους παραβιάζουν τον περιορισμό (Π4).

Ο τρίτος τύπος λύσεων [11, 12, 22, 126, 130] βασίζεται σε μια διαδικασία δύο φάσεων: (α) μια φάση *broadcast*, κατά την οποία ένας κόμβος εκπέμπει το ερώτημα στο δίκτυο, δημιουργώντας ένα (εικονικό) δέντρο κόμβων καθώς το ερώτημα μεταδίδεται (β) μια φάση *convergecast*, κατά την οποία κάθε κόμβος στέλνει το δικό του μέρος της απάντησης, μαζί με τις όποιες απαντήσεις από άλλους κόμβους πιο χαμηλά στο δέντρο, προς τον «γονέα» του. Λύσεις που βασίζονται σε προκατασκευασμένες δεντρικές δομές ανήκουν επίσης σε αυτήν την κατηγορία. Από αυτές τις δουλειές, το Astrolabe [126] ήταν από τις πρώτες που ανέφεραν τη συνάθροιση στον χώρο των ΔΟΕ. Οι συγγραφείς προτείνουν τη δημιουργία και συντήρηση ενός ιεραρχικού δικτύου με μορφή δέντρου, που χρησιμοποιείται για να μεταδίδει πολύπλοκα ερωτήματα και τα αποτελέσματά τους σε όλο το μήκος του δικτύου. Μια παρόμοια ιδέα παρουσιάζεται στο [130]. Το [11] προτείνει τη δημιουργία ενός (συνόλου από) δέντρο(-α) πολλαπλής αποστολής για τη μετάδοση ερωτημάτων και αποτελεσμάτων, ενώ χρησιμοποιεί μεθόδους πλημύρας για να στείλει μηνύματα στο δίκτυο. Παρόλο που αυτές οι δομές έχουν καλές ιδιότητες και μπορούν να υπολογίσουν συναθροιστικά ερωτήματα ευ-

ρείας κλίμακας, δεν είναι κατάλληλες για τη δημιουργία και συντήρηση πολλαπλών ταυτόχρονων μετρητών/συναθροιστών. Ουσιαστικά, όπως και με τις λύσεις ραντεβού, το κόστος διατήρησης πολλαπλών μετρητών ταυτόχρονα αυξάνει γραμμικά με τον αριθμό των μετρητών αυτών. Επίσης, θα μπορούσαμε να χαρακτηρίσουμε τις μεθόδους αυτές ως «κατευθυνόμενο gossip», μιας και η βασική λειτουργικότητα είναι αρκετά παρόμοια με τους gossip αλγόριθμους, με τη διαφορά ότι στη φάση (β) οι κόμβοι ανταλλάσσουν πληροφορία με τους γονείς και τα παιδιά τους στο (εικονικό) δέντρο. Ως εκ τούτου, αυτές οι λύσεις παραβιάζουν τον περιορισμό (Π1), ενώ οι περισσότερες από αυτές (με εξαίρεση το [12]) παραβιάζουν και τους περιορισμούς (Π3) και (Π4).

Η βασική ιδέα του τελευταίου τύπου λύσεων [13, 83] είναι να εκτιμήσουν την τιμή του εν λόγω μετρητή, επικοινωνώντας με μερικούς ειδικά επιλεγμένους κόμβους στο δίκτυο. Το [13] προσπαθεί να υπολογίσει προσεγγιστικά ιστογράμματα στατιστικών του συστήματος χρησιμοποιώντας τυχαία δειγματοληψία των κόμβων στο δίκτυο. Το [83] εκτιμά τον αριθμό των κόμβων στο δίκτυο χρησιμοποιώντας και αυτό έναν αλγόριθμο τυχαίας δειγματοληψίας. Δυστυχώς, δεν υπάρχει κάποιος προφανής τρόπος να γενικεύσουμε τις τεχνικές αυτές ώστε να μετράνε αυθαίρετες ποσότητες (εκτός από αυτές που έχουν σχεδιαστεί να μετράνε). Επίσης, οι τεχνικές δειγματοληψίας είναι γνωστές για την έλλειψη ακρίβειας [17], επομένως παραβιάζουν τον περιορισμό (Π4). Από την άλλη, αν το δείγμα είναι αρκετά μεγάλο [17] ώστε να εγγυάται ένα επίπεδο εμπιστοσύνης τότε αυτές οι λύσεις παραβιάζουν τον περιορισμό (Π1). Τέλος, οι τεχνικές δειγματοληψίας είναι συνήθως ευαίσθητες σε διπλότυπες τιμές, επομένως παραβιάζουν τον περιορισμό (Π6). Τα hash sketches [33,35], εν συντομία, παρέχουν μία κατανομημένη, μη ευαίσθητη σε διπλότυπα μέθοδο για την εκτίμηση του μεγέθους των πολυσυνόλων. Όλες οι γνωστές δουλειές που κατορθώνουν να παρέχουν μη καταμέτρηση διπλοτύπων [11, 12, 22] χρησιμοποιούν hash sketches. Ωστόσο, όλες ανήκουν στην κατηγορία των broadcast/convergecast αλγορίθμων και έχουν επομένως τα μειονεκτήματα που αναφέραμε προηγουμένως.

2.3.4 Ερωτήματα Equi-Join

Η δουλειά σε αυτό το πεδίο είχε σαν πρωτοπόρο το σύστημα PIER [59, 61], το οποίο επικεντρώνεται κυρίως σε equi-join ερωτήματα και αφήνει την υποστήριξη για ερωτήματα εύρους ως μελλοντική δουλειά. Οι συγγραφείς προτείνουν ένα σχήμα συμμετρικού hash join [129] και μια παραλλαγή του παραδοσιακού κατανομημένου join αλγορίθμου όπου ένας από τους joined πίνακες είναι ήδη κατακερματισμένος σε κάποιο join attribute. Και στις δύο περιπτώσεις, ο κόμβος που κάνει το ερώτημα είναι υπεύθυνος για τη διατήρηση ενός προσωρινού DHT namespace, για να κρατά τα ενδιάμεσα αποτελέσματα του ερωτήματος. Μετά από αυτό, στην πρώτη περίπτωση ο κόμβος που κάνει το ερώτημα ξεκινά έναν επανακατακερματισμό των σχετικών δεδομένων στο νέο namespace. Στη δεύτερη περίπτωση κάθε κόμβος που αποθηκεύει δεδομένα της σχέσης που έχει κατακερματιστεί στο join attribute κάνει ένα DHT lookup() για κάθε πλειάδα που ταιριάζει με τον περιορισμό του ερωτήματος (query predicate), συνενώνει αυτές τις πλειάδες με αυτές που έχει αποθηκευμένες και δημοσιεύει τα αποτελέσματα στο προσωρινό DHT namespace. Οι συγγραφείς εξετάζουν περαιτέρω συμμετρικά semi-joins και Bloom joins [59]. Μέχρι τώρα το PIER δεν έχει ένα κατάλογο συστήματος ή κάποιο είδος διαχείρισης στατιστικών – παρόλο που αυτά είναι σχεδιασμένα ως μελλοντική δουλειά – και η βελτιστοποίηση ερωτημάτων γίνεται χρησιμοποιώντας τα FREddies [60] – μια κατανομημένη έκδοση των Eddies [9].

2.3.5 Πολυπλοκότερα Ερωτήματα

Για ερωτήματα ενός attribute, δεν υπάρχει σχεδόν καθόλου περιθώριο βελτίωσης χρησιμοποιώντας έναν βελτιστοποιητή ερωτημάτων. Παρ' όλα αυτά, όταν η έκφραση του ερωτήματος αποτελείται από περιορισμούς σε δύο ή περισσότερες διαφορετικές ιδιότητες ή περιέχει ένα η πε-

ρισσότερα εμφωλευμένα ερωτήματα ή ακόμα αν είναι κάποιος τύπος πιο πολύπλοκου ερωτήματος, όπως τα skyline [127] ή τα similarity [69] ερωτήματα, το σχέδιο εκτέλεσης του ερωτήματος (query plan) γίνεται αρκετά σημαντικό. Είτε είναι ένα ερώτημα πολλών περιορισμών, είτε ένα ερώτημα επιλογής-προβολής, είτε ένα πολλαπλό join, η σειρά με την οποία τα ενδιάμεσα αποτελέσματα υπολογίζονται και δίνονται ως είσοδο στα υπόλοιπα στάδια του αλγορίθμου επεξεργασίας ερωτημάτων μπορεί να έχει σημαντική επίπτωση στο συνολικό χρόνο εκτέλεσης και στο φόρτο που ανατίθεται σε ένα σύστημα διαχείρισης βάσεων δεδομένων σε ΔΟΕ. Και πάλι, λόγω έλλειψης καλύτερης εναλλακτικής (αφού δεν υπάρχει συνολική γνώση), η μηχανή επεξεργασίας ερωτημάτων πρέπει να διαλέξει έναν από τους περιορισμούς στην έκφραση του ερωτήματος και να στείλει το ερώτημα στους κόμβους που αποθηκεύουν τη σχετική πληροφορία όπου θα αξιολογηθεί και τα αποτελέσματα θα σταλούν πίσω στον κόμβο που έκανε το ερώτημα. Η διαδικασία αυτή μπορεί να χρειαστεί να επαναληφθεί για εμφωλευμένα ερωτήματα ή για διάφορους τελεστές (π.χ. συναθροίσεις, joins, κ.λ.π.). Επομένως, ακόμα και ένας στοιχειώδης βελτιστοποιητής ερωτημάτων μπορεί να παίξει ζωτικό ρόλο στη βελτίωση του κόστους και της καθυστέρησης για επεξεργασία ερωτημάτων σε ένα σύστημα διαχείρισης βάσεων δεδομένων ΔΟΕ.

2.3.6 Βελτιστοποίηση Ερωτημάτων για ΔΟΕ

Οι βασικές αρχές του μοντέλου ΔΟΕ – πλήρως κατανεμημένη λειτουργία, πλήρης αυτονομία κόμβων, ανοχή σε λάθη και κλιμακωσιμότητα σε χιλιάδες ή εκατομμύρια κόμβους – υπονοούν ότι κάθε απόφαση που παίρνεται από τους κόμβους σε ένα σύστημα ΔΟΕ θα πρέπει να βασίζεται είτε σε τοπικά διαθέσιμη πληροφορία είτε σε πληροφορία που μπορεί να εξαχθεί γρήγορα και κλιμακωσίμα από το δίκτυο επικάλυψης. Επομένως, κάθε πληροφορία που μπορεί να χαρακτηριστεί ως «συνολική γνώση» είναι συνήθως «απεχθής» για τους υποστηρικτές του μοντέλου. Συναθροίσεις και στατιστικά όπως ιστογράμματα ή περιλήψεις των σχέσεων που αποθηκεύονται σε ένα διαδικτυακής κλίμακας σύστημα αποθήκευσης δεδομένων μπορεί πολύ εύκολα να τοποθετηθούν σε αυτή την κατηγορία. Αυτό έχει οδηγήσει τους ερευνητές είτε στο να καταφεύγουν σε μάλλον ad-hoc [60] λύσεις για τη βελτιστοποίηση ερωτημάτων ή στο να εξετάζουν τεχνικές απλής δειγματοληψίας για την κατασκευή ιστογραμμάτων [13] και εξαγωγής των στατιστικών ιδιοτήτων των ΔΟΕ [72, 89], με τα γνωστά μειονεκτήματα που έχει η δειγματοληψία όταν πρέπει να απαλειφθούν οι διπλές τιμές [17] ενώ παράλληλα επηρεάζουν άμεσα την ακρίβεια των εκτιμήσεων που προκύπτουν.

Από αυτές τις δουλειές, το [13] υποστηρίζει κυρίως ερωτήματα εύρους, με βάρος στην κατανομή φόρτου και δομεί το δίκτυό του με ένα δεδομένο-κεντρικό τρόπο, έτσι ώστε η τοπικότητα των αντικειμένων να αντικατοπτρίζεται και στη διάρθρωση του δικτύου. Προτείνουν ένα pull-based σχήμα τυχαίας δειγματοληψίας, χρησιμοποιώντας τυχαίους περιπάτους στο ΔΟΕ. Βέβαια, μιας και το δίκτυο είναι τύπου «expander», αυτή η τεχνική ισοδυναμεί με ομοιόμορφη τυχαία δειγματοληψία. Χρησιμοποιούν αυτήν την πληροφορία για να διατηρούν προσεγγιστικά ιστογράμματα για την πυκνότητα του πληθυσμού των κόμβων, την οποία μετά την διαθέτουν για εκτίμηση της επιλεκτικότητας και για διάφορες ανάγκες συντήρησης του δικτύου. Η δική μας προσέγγιση των ιστογραμμάτων είναι πολύ πιο γενική, επιτρέποντας τόσο ιστογράμματα «μέτρησης κόμβων», όσο και ιστογράμματα επί των δεδομένων που διαχειρίζονται οι κόμβοι του δικτύου

2.4 Σύνοψη Παραδοσιακών Μεθόδων Βελτιστοποίησης Ερωτημάτων

Όπως αναφέραμε, τα επιτυχημένα παραδοσιακά συστήματα βασίζονται στη χρήση και εκμετάλλευση συγκεκριμένων στατιστικών για να επιτύχουν αποτελεσματική επεξεργασία ερωτημάτων, συνήθως χρησιμοποιώντας τεχνικές συνάθροισης δεδομένων, περιλήψεις και ιστογράμματα.

Τέτοια στατιστικά χρησιμοποιούνται με πολλούς τρόπους κατά τη ζωή ενός ερωτήματος, είτε σαν μέρος της λογικής βελτιστοποίησης – για παράδειγμα, για να βελτιστοποιηθούν τα μονοπάτια προσπέλασης για ερωτήματα μίας σχέσης [116] και για να υπολογιστεί η επιλεκτικότητα κάποιων εκφράσεων [29, 64, 106, 107] ή η βέλτιστη σειρά εφαρμογής εκφράσεων σε ερωτήματα με πολλές εκφράσεις – είτε σαν υψηλής ποιότητας δείγματα των βασικών δεδομένων σε συστήματα προσεγγιστικής απάντησης ερωτημάτων [104]. Ακόμα και αν αναζητούμε ακριβείς απαντήσεις, τέτοιες γρήγορες προσεγγιστικές απαντήσεις μπορεί να είναι επιθυμητές σαν μια μορφή ανάδρασης προς το χρήστη πριν την εκτέλεση ενός ερωτήματος που μπορεί να καταναλώσει πολύ χρόνο και πολλούς πόρους.

Το System-R [116] ήταν από τα πρώτα συστήματα που εφάρμοσαν βελτιστοποίηση με βάση το κόστος. Διατηρούσε στους καταλόγους του συστήματος στατιστικά όπως ο αριθμός πλειάδων και οι σελίδες δεδομένων ανά σχέση, ο αριθμός σελίδων δεδομένων και διακριτών τιμών ανά ευρετήριο και το λόγο των σελίδων δεδομένων ανά τμήμα που διατηρούν πληροφορία για οποιαδήποτε δεδομένη σχέση. Το κόστος των υποψηφίων μονοπατιών προσπέλασης υπολογιζόταν χρησιμοποιώντας ένα σύνολο από τύπους με στατικούς παράγοντες και τα ανωτέρω στατιστικά ως είσοδο. Αργότερα, η βιομηχανία στράφηκε στα ιστογράμματα ως ένα πιο ακριβή και συμπαγή τρόπο περίληψης των αποθηκευμένων δεδομένων. Οι εργασίες [62] και [107] δίνουν μια ταξινόμια και μια σύντομη ιστορία της ανάπτυξης των ιστογραμμάτων, ενώ η [74] παρουσιάζει μία έρευνα για κατανεμημένη επεξεργασία ερωτημάτων στην προ-ΔΟΕ εποχή.

Εν συντομία, το [16] αναγνωρίζει την ακόλουθη πληροφορία ως απαραίτητη ώστε ο βελτιστοποιητής να μπορέσει να πάρει μια σωστή απόφαση: (1) αριθμός πλειάδων στη σχέση, (2) αριθμός φυσικών σελίδων που χρησιμοποιούνται από έναν πίνακα, (3) στατιστική πληροφορία για τις στήλες του πίνακα, με τη μορφή ιστογραμμάτων, ελάχιστης και μέγιστης ή δεύτερης-ελάχιστης και δεύτερης-μέγιστης τιμής και αριθμού διακριτών τιμών στη στήλη και (4) πληροφορία για τη συσχέτιση μεταξύ των τιμών των ιδιοτήτων, με τη μορφή είτε πολυδιάστατων ιστογραμμάτων (που έχουν το μειονέκτημα ότι γίνονται πολύ μεγάλα καθώς αυξάνονται οι διαστάσεις) είτε ενός μονοδιάστατου ιστογράμματος στην κύρια στήλη, συν τον αριθμό των διακριτών συνδυασμών των τιμών της στήλης που είναι παρούσες στα δεδομένα, για ευρετήρια πολλών στηλών. Στις επόμενες ενότητες θα συζητήσουμε τεχνικές για να υπολογίζουμε τέτοια στατιστικά σε ένα περιβάλλον ΔΟΕ.

Περίληψεις, ιστογράμματα, συναθροιστικά ερωτήματα και συνόψεις δεδομένων γενικότερα, έχουν βρει χρήση σε πολλά άλλα πεδία της επιστήμης των υπολογιστών. Η έρευνα σε ροές δεδομένων [23, 24, 49, 91] έχει στραφεί σε τέτοιες συνόψεις δεδομένων για να απαλλαγεί από μεταφορές μεγάλων όγκων δεδομένων και να λιγοστεύει τις ανάγκες για αποθήκευση και επεξεργασία στους επεξεργαστές ροών. Ιστογράμματα και περιλήψεις έχουν επίσης προταθεί ως ένας τρόπος για να μειωθεί ο όγκος δεδομένων που μεταδίδεται – και κατά συνέπεια η καταναλισκόμενη ισχύς – σε κόμβους δικτύων αισθητήρων [22, 123]. Με την ίδια λογική, κατανεμημένα συστήματα, όπως συστήματα δημοσίευσης/συνδρομής [15, 124], κατανεμημένες web proxy caches [34] και μηχανές αναζήτησης ΔΟΕ [77, 85, 86] έχουν όλα χρησιμοποιήσει στατιστικές συνόψεις ώστε να ανταλλάζουν γρήγορα πληροφορίες για δεδομένα αποθηκευμένα σε διάφορους κόμβους του συστήματος.

2.4.1 Ιστογράμματα

Τα ιστογράμματα είναι μακράν η πιο συχνή τεχνική που απαντάται στην πράξη σε εμπορικές βάσεις δεδομένων σαν μια στατιστική περίληψη και μια προσέγγιση της κατανομής των τιμών στις σχέσεις της βάσης. Για μια συγκεκριμένη ιδιότητα/ στήλη, ένα ιστόγραμμα είναι μια ομαδοποίηση των τιμών των ιδιοτήτων σε «κελιά», των οποίων η συνολική συχνότητα προσεγγίζεται από τα στατιστικά που διατηρούνται σε κάθε κελί.

Όλα τα ιστογράμματα κάνουν κάποιες βασικές υποθέσεις όσον αφορά την κατανομή των αντικειμένων σε κάθε κελί. Μία από τις πιο συνήθεις τέτοιες υποθέσεις είναι η υπόθεση ομοιό-

μορφής διασποράς (uniform spread assumption) [107]. Σύμφωνα με αυτή, οι τιμές σε ένα κελί υποτίθεται ότι υπάρχουν μόνο σε σημεία ίσης κατανομής (ίσα με το μέσο όρο του κελιού) και έχουν συχνότητα ίση με το λόγο της συχνότητας του κελιού προς τον αριθμό των διακριτών τιμών σε αυτό.

Η πιο βασική κατηγορία ιστογραμμάτων – τα ιστογράμματα Ίσου Πλάτους (Equi-Width) – διαμερίζει το πεδίο τιμών της ιδιότητας σε κελιά ίσου εύρους και αναθέτει σε καθένα τον αριθμό των πλειάδων στις οποίες η ιδιότητα έχει τιμή μέσα στα όρια του κελιού. Η σχετική έρευνα έχει επικεντρωθεί στη βελτίωση των στατιστικών ιδιοτήτων των ιστογραμμάτων, με πολλά ενδιαφέροντα αποτελέσματα. Το [107] παρουσιάζει μια ταξινόμηση ιστογραμμάτων μαζί με μερικούς καινοτόμους τύπους ιστογραμμάτων και εξετάζει εκτενώς την ακρίβεια και την αποδοτικότητά τους σε σχέση με το χρόνο κατασκευής και τις απαιτήσεις σε αποθηκευτικό χώρο. Οι συγγραφείς καταλήγουν προτείνοντας τη χρήση ιστογραμμάτων $\text{MaxDiff}(V, A)$ και $\text{MaxDiff}(V, F)$.

Στις υπόλοιπες ενότητες, θα συζητήσουμε εργαλεία, πρωτόκολλα και τεχνικές για να κατασκευάσουμε και να υπολογίζουμε διάφορους τύπους ιστογραμμάτων. Εκτός από το απλό ιστόγραμμα Ίσου Πλάτους, θα ασχοληθούμε με τα ιστογράμματα Μετακινημένου Μέσου Ίσου Πλάτους (Average-Shifted Equi-Width), Ίσου Βάθους (Equi-Depth), $\text{MaxDiff}(V, F)$ και $\text{Compressed}(V, F)$, τα οποία και περιγράφουμε εν συντομία.

Τα ιστογράμματα Μετακινημένου Μέσου (ASH) [115] είναι μια ενδιαφέρουσα εναλλακτική για τα απλά ιστογράμματα Ίσου Πλάτους. Ένα ASH αποτελείται από ένα σύνολο από ιστογράμματα Ίσου Πλάτους, με τον ίδιο αριθμό κελιών και το ίδιο μέγεθος κελιού, αλλά διαφορετικό σημείο εκκίνησης. Η συχνότητα της κάθε τιμής σε ένα κελί υπολογίζεται ως ο μέσος όρος των εκτιμήσεων από κάθε ένα τέτοιο ιστόγραμμα. Το ASH – στην ουσία ένας εκτιμητής πυρήνα (kernel estimator) – μπορεί να δώσει μια πιο ομαλή εκτίμηση της πραγματικής κατανομής των τιμών των πλειάδων σε σχέση με τα απλά ιστογράμματα Ίσου Πλάτους.

Τα ιστογράμματα Ίσου Βάθους [92, 118] έχουν χρησιμοποιηθεί ευρέως σε εμπορικά συστήματα διαχείρισης βάσεων δεδομένων. Αποτελούνται από ένα διαμερισμό του πεδίου τιμών της ιδιότητας σε ξένα μεταξύ τους διαστήματα, έτσι ώστε ο αριθμός των πλειάδων των οποίων η τιμή της ιδιότητας ανήκει σε κάθε ένα τέτοιο διάστημα, να είναι (περίπου) ίσος για όλα τα διαστήματα. Με πιο απλά λόγια, στα ιστογράμματα Ίσου Βάθους τα κελιά έχουν την ίδια συχνότητα, αλλά όχι (απαραίτητα) και το ίδιο εύρος.

Τα ιστογράμματα $\text{MaxDiff}(V, F)$ πρώτα ταξινομούν τα αντικείμενα με βάση τις τιμές τους και μετά τοποθετούν τα όρια των κελιών σε σημεία στο πεδίο τιμών όπου τα αντικείμενα έχουν τις μεγαλύτερες διαφορές συχνότητας. Ο Microsoft SQL Server χρησιμοποιεί μια παραλλαγή αυτών των ιστογραμμάτων [62], στην οποία αποθηκεύει επίσης τη συχνότητα των τιμών στα όρια των κελιών μαζί με τη γενική συχνότητα σε κάθε κελί.

Σε ένα $\text{Compressed}(V, F)$ ιστόγραμμα c -κελιών, τα αντικείμενα με τη $c - k$ μεγαλύτερη συχνότητα τοποθετούνται σε κελιά μοναδικής τιμής, ενώ οι υπόλοιπες τιμές τοποθετούνται σε k κελιά Ίσου Βάθους. Αυτός ο τύπος ιστογράμματος έχει αποδειχθεί ότι επιτυγχάνει τη βέλτιστη ισορροπία ανάμεσα στο βέλτιστο όσον αφορά τη μετρική της διασποράς [64] και στο πρακτικό λόγω του μεγέθους και του χρόνου κατασκευής και χρησιμοποιείται από το βελτιστοποιητή της DB2 [62].

2.4.2 Συνόψεις Κατακερματισμού

Τα Hash Sketches προτάθηκαν για πρώτη φορά από τους Flajolet και Martin στο [35] υπό το όνομα Πιθανοτικό Μέτρημα με Στοχαστικό Μέσο Όρο (ή αλλιώς *PCSA* ή *FM85*), σαν ένα μέσο υπολογισμού του αριθμού των διακριτών τιμών σε ένα πολυσύνολο $\mathcal{D}$ δεδομένων σε μια βάση. Η εκτίμηση που παίρνουμε είναι (ουσιαστικά) μη πολωμένη (unbiased), ενώ οι συγγραφείς δίνουν και άνω φράγματα για τη διασπορά της. Η μόνη υπόθεση που κάνουν τα hash sketches είναι η ύπαρξη μιας ψευδο-ομοιόμορφης συνάρτησης κατακερματισμού $h() : \mathcal{D} \rightarrow [0, 1, \dots, 2^\lambda)$ – μία υπόθεση που γίνεται από την περισσότερη (αν όχι όλη) την έρευνα σχετικά με ΔΟΕ. Οι Durand και Flajolet

Σχήμα 2.1: Εισαγωγή Αντικειμένων στη Σύνοψη Κατακερματισμού: Περίπτωση Μονού Πίνακα

παρουσίασαν έναν παρόμοιο αλγόριθμο [33], με το όνομα *superLogLog* ή *DF03* μέτρημα, ο οποίος μείωσε την πολυπλοκότητα χώρου και χαλάρωσε τις υποθέσεις για τη συνάρτηση κατακερματισμού του [35]. Τα hash sketches έχουν χρησιμοποιηθεί σε πολλά πεδία εφαρμογών όπου είναι σημαντική η μέτρηση διακριτών στοιχείων, όπως προσεγγιστική απάντηση ερωτημάτων σε πολύ μεγάλες βάσεις δεδομένων [76], εξόρυξη δεδομένων από το Διαδίκτυο [85, 100] και επεξεργασία ροών [30, 42].

Ένα hash sketch αποτελείται από ένα διάνυσμα bits $B[\cdot]$ μήκους λ - όπου όλα τα bits είναι αρχικά 0 - και μια συνάρτηση κατακερματισμού $h(\cdot)$ όπως παραπάνω.

Καταμέτρηση super-LogLog

Έστω $\rho(y) : [0, 2^\lambda) \rightarrow [0, \lambda)$ η θέση του λιγότερου σημαντικού (αριστερότερου) 1-bit στη δυαδική αναπαράσταση του y , δηλαδή $\rho(y) = \{ \min(i \geq 0) : \text{bit}(y, i) \neq 0 \}$, $y > 0$ και $\rho(0) = \lambda$, όπου $\text{bit}(y, i)$ δηλώνει το i -οστό bit στην αναπαράσταση του y (η θέση 0 αντιστοιχεί στο λιγότερο σημαντικό bit). Για να εκτιμήσουμε τον αριθμό I των διακριτών στοιχείων σε ένα πολυσύνολο $\mathcal{D}$ εφαρμόζουμε την $\rho(h(d))$ σε όλα τα στοιχεία $d \in \mathcal{D}$ και καταγράφουμε τα αποτελέσματα στο διάνυσμα $B[\cdot]$ (βλπ. Σχήμα 2.1). Εφόσον η $h(\cdot)$ κατανέμει τις τιμές ομοιόμορφα στο $[0, 2^\lambda)$, συνεπάγεται ότι:

$$P(\rho(h(d)) = i) = 2^{-i-1}. \quad (2.1)$$

Επομένως, όταν μετράμε τα στοιχεία σε ένα πολυσύνολο I στοιχείων, το $B[0]$ θα τεθεί στην τιμή 1 περίπου $\frac{I}{2}$ φορές, το $B[1]$ περίπου $\frac{I}{4}$ φορές, κοκ. Το γεγονός αυτό είναι μάλλον διαισθητικό. Ας φανταστούμε όλους τους I πιθανούς λ -bit αριθμούς. Το λιγότερο σημαντικό ψηφίο (ψηφίο 0) θα είναι 1 για τους μισούς από αυτούς (μονοί αριθμοί). Από τους υπόλοιπους $\frac{I}{2}$ αριθμούς οι μισοί θα έχουν το δεύτερο ψηφίο 1 ή αλλιώς $\frac{I}{4}$ συνολικά, κοκ.

Τότε, η ισότητα $R(\mathcal{D}) = \max_{d \in \mathcal{D}} \rho(h(d))$ μας δίνει μία εκτίμηση της τιμής του $\log I$, με μια πρόσθετη πόλωση μεγέθους 1,33 και διασπορά 1,87. Επομένως το 2^R είναι μια «λογαριθμική» εκτίμηση του I εντός 1,87 δυαδικών τάξεων μεγέθους. Παρ' όλα αυτά, η αναμενόμενη τιμή του 2^R είναι άπειρη και ως εκ τούτου δεν μπορεί να χρησιμοποιηθεί για την εκτίμηση του I . Σε αυτά τα πλαίσια, το [33] προτείνει την ακόλουθη τεχνική (παρόμοια με την τεχνική στοχαστικού μέσου στο [35]): (1) χρησιμοποιούμε ένα σύνολο από $m = 2^c$ διαφορετικά $B^{(i)}[\cdot]$ διανύσματα (όπου c μη αρνητικός ακέραιος), καθένα από τα οποία δίνει μια διαφορετική εκτίμηση για το $R^{(i)}$, (2) για κάθε στοιχείο d , επιλέγουμε ένα από αυτά χρησιμοποιώντας τα πρώτα c ψηφία του $h(d)$, και (3)

Σχήμα 2.2: Εισαγωγή Αντικειμένων στη Σύνοψη Κατακερματισμού: Περίπτωση Πολλαπλών Πινάκων

ενημερώνουμε το επιλεγμένο διάνυσμα και υπολογίζουμε το $R^{(i)}$ χρησιμοποιώντας τα υπόλοιπα ψηφία του $h(d)$ (βλ. Σχήμα 2.2).

Αν $M^{(i)}$ είναι η (τυχαία) τιμή της παραμέτρου R για το διάνυσμα i , τότε ο αριθμητικός μέσος $\frac{1}{\beta} \sum_{i=1}^{\beta} M^{(i)}$ έχει αναμενόμενη τιμή περίπου $\log\left(\frac{I}{\beta}\right)$ συν μια πρόσθετη πόλωση. Η εκτίμηση για το I υπολογίζεται λοιπόν από τον τύπο: $E(I) = \alpha_{\beta} \cdot \beta \cdot 2^{\frac{1}{\beta} \cdot \sum_{i=1}^{\beta} M^{(i)}}$, όπου $\alpha_{\beta} = (-\beta \cdot \frac{2^{-\frac{1}{\beta}} - 1}{\log(2)} \cdot \int_0^{\infty} e^{-t} \cdot t^{-\frac{1}{\beta}} dt)^{-\beta}$ ([33]). Οι συγγραφείς προτείνουν επίσης και έναν κανόνα σύντμησης, λαμβάνοντας υπ' όψη μόνο τις $\beta_0 = \lfloor \theta_0 \cdot \beta \rfloor$ μικρότερες τιμές. Το θ_0 είναι ένας πραγματικός αριθμός ανάμεσα στο 0 και το 1, ενώ η τιμή $\theta_0 = 0.7$ δίνει αποτελέσματα κοντά στο βέλτιστο. Με αυτή την τροποποίηση, η εξίσωση εκτίμησης γίνεται:

$$E(I) = \tilde{\alpha}_{\beta} \cdot \beta_0 \cdot 2^{\frac{1}{\beta_0} \cdot \sum^* M^{(i)}} \quad (2.2)$$

όπου το $\sum^*$ υποδεικνύει το συντετμημένο άθροισμα και η τροποποιημένη σταθερά $\tilde{\alpha}_{\beta}$ διασφαλίζει ότι η εκτίμηση παραμένει μη πολωμένη (βλ. Σχήμα 2.3). Η εκτίμηση που προκύπτει έχει διασπορά $\frac{1.05}{\sqrt{\beta}}$, ενώ η συνάρτηση κατακερματισμού πρέπει να έχει μήκος τουλάχιστον

$$H_0 = \log(\beta) + \lceil \log\left(\frac{I_{max}}{\beta}\right) + 3 \rceil, \quad (2.3)$$

όπου I_{max} είναι το μέγιστο μέγεθος συνόλου.

Καταμέτρηση PCSA

Ο αλγόριθμος στο [35] βασίζεται στο ίδιο σχήμα κατακερματισμού (δηλαδή χρησιμοποιεί το $\rho(\cdot)$) και τις ίδιες παρατηρήσεις όπως και το [33]. Ο αλγόριθμος PCSA διαφέρει από τον αλγόριθμο superLogLog στα ακόλουθα: (α) το [35] βασίζεται στην ύπαρξη μιας οικογένειας συναρτήσεων κατακερματισμού που έχουν ιδανικές ιδιότητες τυχαιότητας, ενώ το [33] έχει χαλαρώσει αυτή την υπόθεση, (β) το [35] θέτει το R ως το αριστερότερο 0-bit στον διάνυσμα $B[\cdot]$, σε αντίθεση με τη θέση του πιο δεξιού 1-bit στο διάνυσμα στο [33], (γ) το [33] χρησιμοποιεί αριθμό bits της τάξης του $\log \log(\text{μέγιστος πηληθάριθμος συνόλου})$ ανά διάνυσμα ενώ το [35] χρειάζεται αριθμό bits της τάξης

Σχήμα 2.3: Καταμέτρηση Αντικειμένων με Χρήση Σύνοψης Κατακερματισμού

του $\log(\text{μέγιστος πηληθάριθμος συνόλου})$, (δ) η εκτίμηση στο [35] υπολογίζεται ως:

$$E(I) = \frac{1}{0.77351} \cdot \beta \cdot 2^{\frac{1}{\beta}} \sum_{i=0}^{\beta-1} M^{(i)}, \quad (2.4)$$

και (ε) η πόλωση και το μέσο λάθος στο [35] μπορούν να εκτιμηθούν αρκετά καλά από τα $1 + 0.31/\beta$ και $0.78/\sqrt{\beta}$ αντίστοιχα. Να σημειωθεί ότι ο αλγόριθμος εισαγωγής δεδομένων είναι ο ίδιος και για το [33] και για το [35] (με τη μόνη διαφορά των υποθέσεων για τη συνάρτηση κατακερματισμού).

Τα hash sketches μπορούν να κατανεμηθούν με φυσικότητα: το hash sketch της ένωσης οποιουδήποτε αριθμού συνόλων μπορεί να υπολογιστεί από τα hash sketches των συνόλων αυτών με μια απλή πράξη λογικού H (bitwise OR) στα ψηφία των αντίστοιχων διανυσμάτων, δεδομένου ότι όλα τα hash sketches έχουν τον ίδιο αριθμό διανυσμάτων ψηφίων και μήκος και έχουν κατασκευαστεί χρησιμοποιώντας το ίδιο σύνολο συναρτήσεων κατακερματισμού. Επομένως, αν ένα αρχικό σύνολο A είναι κατανεμημένο σε διάφορους υπολογιστές (π.χ. κατά μήκος ενός ΔΟΕ), είναι δυνατό να υπολογιστεί ένα συνολικό hash sketch για το A από τα τοπικά υπολογισμένα hash sketches που αντιστοιχούν στο υποσύνολο του A για το οποίο είναι υπεύθυνος κάθε κόμβος του δικτύου. Έχουμε υλοποιήσει τόσο τον αλγόριθμο superLogLog όσο και τον PCSA στα πλαίσια της υποδομής μας [37]. Εδώ αναφέρουμε τις ιδιότητες μόνο του πρώτου προς χάριν ευκρίνειας της παρουσίασης και λόγω των καλύτερων θεωρητικών ιδιοτήτων και απαιτήσεων του κατά την εκτέλεση.

Κεφάλαιο 3

Δίκτυα Επικάλυψης ΔΟΕ Βασιζόμενα σε Αλτρουιστές

Τα ΔΟΕ λαμβάνουν όλο και μεγαλύτερη αναγνώριση και προσοχή. Δεδομένων των χαρακτηριστικών τους και των εγγενών υποσχέσεων του υπολογιστικού παραδείγματος των ΔΟΕ, η προσοχή αυτή κατά πάσα πιθανότητα θα αυξηθεί στα ερχόμενα χρόνια. Στα πλαίσια της σχετικής έρευνας, ένα θεμελιώδες πρόβλημα έχει τύχει ιδιαίτερης μεταχείρισης και προσοχής: το πρόβλημα της δρομολόγησης στα ΔΟΕ. Δεδομένης μίας αίτησης για ένα συγκεκριμένο αντικείμενο (με ένα συγκεκριμένο αναγνωριστικό) η οποία φτάνει σε έναν κόμβο του ΔΟΕ, το πρόβλημα της δρομολόγησης έγκειται στην εξεύρεση του κόμβου ο οποίος αποθηκεύει και συνεισφέρει το αντικείμενο αυτό. Παράλληλα, ένα από τα κύρια χαρακτηριστικά ευρέως χρησιμοποιούμενων ΔΟΕ (π.χ. Gnutella, Kazaa, κτλ.) είναι το ότι οι κόμβοι που συμμετέχουν σε αυτά είναι πολύ ετερογενείς, όσον αφορά την υπολογιστική ισχύ τους, τον διαθέσιμο αποθηκευτικό χώρο, το εύρος ζώνης δικτύου και την διάρκεια της σύνδεσής τους με το διαδίκτυο. Σχετικές μελέτες σε ΔΟΕ [5, 114] έχουν δείξει ότι η ετερογένεια αυτή αντικατοπτρίζεται και στην κατανομή του φόρτου της επεξεργασίας ερωτημάτων στους κόμβους του δικτύου: στο δίκτυο Gnutella, για παράδειγμα, το 70% των κόμβων δεν συνεισφέρουν καθόλου αρχεία, ενώ το 5% των κόμβων εξυπηρετούσε σχεδόν το 95% όλων των ερωτημάτων του συστήματος.

Η αναγνώριση της ετερογένειας αυτής ανάμεσα στους κόμβους και η εκμετάλλευσή της για την βελτίωση της δρομολόγησης και της επεξεργασίας πολύπλοκων ερωτημάτων δεν έχει γίνει ξανά στο παρελθόν στο περιβάλλον των DHT, αν και οι περισσότερες πραγματικές εφαρμογές των ΔΟΕ την χρησιμοποιούν, χτίζοντας πολυεπίπεδα υβριδικά δίκτυα. Η φιλοσοφία μας βασίζεται ακριβώς σε αυτή την παρατήρηση. Αυτό που θέλουμε είναι να μεταφέρουμε αυτόν τον «υβριδικό» σχεδιασμό στον κόσμο των DHT και να τον χρησιμοποιήσουμε για την επεξεργασία πολύπλοκων ερωτημάτων και την δρομολόγηση πακέτων. Το ερώτημα λοιπόν είναι πώς μπορεί να γίνει αυτό με τρόπο αποδοτικό. Πιστεύουμε, εν τέλει, ότι η εκμετάλλευση της ισχύος των δυνατών και «αλτρουιστικών» κόμβων είναι το κλειδί στην προσπάθεια αυτή. Στην εργασία μας αυτή εξετάζουμε πρωτίστως την περίπτωση των ερωτημάτων εύρους, αν και η προσέγγισή μας μπορεί να εφαρμοστεί και για άλλα είδη ερωτημάτων. Σημειώνουμε ότι δεν προτείνουμε ακόμα ένα υπόστρωμα ΔΟΕ υποστήριξης ερωτημάτων εύρους. Απεναντίας, βασιζόμαστε στην λειτουργικότητα, την κλιμακωσιμότητα και την απόδοση τέτοιων δικτύων επικάλυψης και παρουσιάζουμε μία διεπίπεδη αρχιτεκτονική όπου ισχυροί κόμβοι ταυτοποιούνται και τους ανατίθεται εξειδικευμένο έργο. Αυτό το κάνουμε με τρόπο αποδοτικό και με μικρό επιπλέον κόστος, χρησιμοποιώντας την λειτουργικότητα που ήδη μας παρέχεται από το υποκείμενο δομημένο ΔΟΕ, επιτυγχάνοντας ωστόσο σημαντικά οφέλη στο κόστος επεξεργασίας ερωτημάτων εύρους. Εξ όσων γνωρίζουμε, αυτή είναι η πρώτη εργασία που ασχολείται

με τα ζητήματα αυτά.

Η αρχική μας τοποθέτηση βασίζεται στο ότι στον κόσμο μας υπάρχουν αλτρουιστές, εγωιστές και άλλοι με συμπεριφορά ανάμεσα στις δύο πρώτες κατηγορίες, με ένα μη αμελητέο ποσοστό των τελευταίων να κλίνουν προς τους πρώτους δεδομένων των κατάλληλων κινήτρων. Στον κόσμο των ΔΟΕ το γεγονός αυτό έχει ξεκάθαρα εκδηλωθεί και καταγραφεί: για παράδειγμα, γνωρίζουμε το φαινόμενο των «λαθρεπιβατών» [5], το οποίο πρώτα παρατηρήθηκε στο ΔΟΕ Gnutella και κατόπιν σε όλα τα διαδεδομένα συστήματα ΔΟΕ διαμοιρασμού αρχείων.

Τα άσχημα νέα είναι ότι η μεγαλύτερη πλειοψηφία των κόμβων του Gnutella αποδείχθηκε πως ήταν «λαθρεπιβάτες» (περισσότεροι από το 70%), γεγονός που μπορεί να αποδειχθεί ολέθριο για τα δίκτυα επικάλυψης τύπου DHT, καθώς πολλοί κόμβοι θα συνδέονται με το δίκτυο και θα το εγκαταλείπουν ύστερα από πολύ λίγο – οι [114] έδειξαν ότι η μέση χρονική διάρκεια συνεδρίας στα δίκτυα Gnutella και Napster ήταν περίπου 60 λεπτά! Υπάρχουν ακόμα άλλες ανεξάρτητες αναφορές οι οποίες προσφέρουν ενδείξεις ότι ο μισός πληθυσμός των κόμβων του δικτύου αλλάζει σχεδόν κάθε μισή ώρα [128]. Το γεγονός αυτό αποτελεί τόσο σοβαρό πρόβλημα κυρίως λόγω του κόστους (σε μηνύματα και χρόνο) αναδιοργάνωσης των DHT όταν το δίκτυο βρίσκεται σε διαρκή αλλαγή. Τα καλά νέα είναι ότι ένα μη αμελητέο ποσοστό των κόμβων αποδείχθηκε πως ήταν αλτρουιστές¹. Στο σύστημα Mojonation [87], περισσότεροι από 1-2% όλων των χρηστών έμεναν συνδεδεμένοι διαρκώς. Στο σύστημα Gnutella, το 1% (10%) των κόμβων εξυπηρετούσαν περίπου το 40% (90%) όλων των αιτήσεων [5], ενώ οι [114] έδειξαν ότι όσο περισσότερο μένει συνδεδεμένος ένα κόμβος, τόσο πιθανότερο είναι να παραμείνει συνδεδεμένος για ακόμα μεγαλύτερο χρονικό διάστημα.

Γενικά, στην εργασία μας αυτή, ορίζουμε ως αλτρουιστές τους κόμβους οι οποίοι:

- Παραμένουν συνδεδεμένοι στο ΔΟΕ για σημαντικά μεγαλύτερες χρονικές περιόδους από το μέσο όρο των υπόλοιπων κόμβων και
- Κατέχουν και διαθέτουν την αναγκαία ισχύ για να δεχτούν και να επεξεργαστούν μεγαλύτερα ποσά φόρτου.

Με αυτά τα χαρακτηριστικά κατά νου, επανεξετάζουμε τα «παραδοσιακά» επιχειρήματα για την αποφυγή σημείων συσσωρευμένης δρομολόγησης (routing hotspots) και για το κόστος της αντιμετώπισης διαρκών αλλαγών στη δομή και τη σύνθεση του δικτυακού υποστρώματος των ΔΟΕ. Πιο συγκεκριμένα:

- Προτείνουμε ως καλή ιδέα την συγκέντρωση των περισσότερων αποφάσεων δρομολόγησης και επεξεργασίας ερωτημάτων σε αλτρουιστές κόμβους· οι κόμβοι αυτοί κατέχουν και διαθέτουν τους πόρους που απαιτούνται για τα παραπάνω. Αυτό έχει ως αποτέλεσμα την αποδοτικότερη δρομολόγηση και επεξεργασία ερωτημάτων από τους κλασσικούς αλγορίθμους οι οποίοι θα επιφορτίζουν και αδύνατους κόμβους με τέτοιο έργο.
- Η παραπάνω απόφαση προφανώς θα δημιουργήσει μεγαλύτερες δομές δεικτοδότησης στους αλτρουιστές κόμβους. Παραδοσιακά, αυτό προκαλεί μεγαλύτερο κόστος αναδιοργάνωσης, το οποίο οι κόμβοι πρέπει να πληρώσουν όταν άλλοι κόμβοι εισέρχονται και αποχωρούν από το ΔΟΕ. Ωστόσο, οι επιπλέον εγγραφές στους πίνακες δρομολόγησης των αλτρουιστών θα αφορούν άλλους αλτρουιστικούς κόμβους. Καθώς αυτοί παραμένουν συνδεδεμένοι για μεγαλύτερες χρονικές περιόδους, η διατήρηση της εγκυρότητας της πρόσθετης αυτής πληροφορίας μπορεί να γίνει χωρίς μεγάλο κόστος σε εύρος ζώνης δικτύου.

¹Υπάρχει μία σχετική διαφωνία για το αν οι συμπεριφορές των κόμβων αυτών είναι όντως αλτρουιστικές κατά βάση ή αν πρόκειται απλά για θετική παρενέργεια (π.χ. όφελος προς την κοινότητα των ΔΟΕ το οποίο προκύπτει από κόμβους οι οποίοι όμως δρουν προς δικό τους συμφέρον και μόνο). Ωστόσο και στις δύο περιπτώσεις το κέρδος που αποκομίζει το ΔΟΕ από τους κόμβους αυτούς αναγνωρίζεται από όλους. Για τον λόγο αυτό καθώς και για συντομία, στο υπόλοιπο της διατριβής αυτής θα αναφερόμαστε σε «αλτρουιστικούς» κόμβους εννοώντας και τις δύο αυτές κατηγορίες κόμβων.

Αρχικά παρουσιάζουμε το AESOP, μία υποδομή που μπορεί να χρησιμοποιηθεί για την ανάπτυξη Δικτύων Ομότιμων Εταίρων (ΔΟΕ), τα οποία λαμβάνουν υπόψη τους και την ύπαρξη ισχυρών και αλτρουιστικών κόμβων. Η προσέγγιση μας αποτελείται από τρεις συνιστώσες : (α) μία πολλαπλών επιπέδων αρχιτεκτονική που ονομάζεται PLANES και αναλαμβάνει να ομαδοποιήσει τους κόμβους του ΔΟΕ σύμφωνα με τις ικανότητες τους και την αλτρουιστική τους διάθεση, (β) έναν ελαφρύ ελεγκτικό μηχανισμό (AltSeAl) που αναλαμβάνει να συντηρεί και να συλλέγει στατιστικά που αφορούν τους κόμβους και τα οποία χρησιμοποιούνται από το PLANES, με σκοπό να μπορεί να εντοπίσει τους ισχυρούς και αλτρουιστικούς κόμβους, και (γ) το SeAledPLANES, ένας μηχανισμός που αναλαμβάνει να συντονίζει τα δύο προηγούμενα μέρη. Στη συνέχεια εφαρμόζουμε την ιδέα αυτή στο πεδίο της επεξεργασίας ερωτημάτων εύρους πάνω από δικτυακά υποστρώματα DHT-ΔΤ.

3.1 Σχεδιάζοντας Διαστρωματωμένα, Αλτρουιστικά ΔΟΕ

Θεωρούμε ότι τα αναγνωριστικά (IDs) τόσο των κόμβων όσο και των αντικειμένων αποτελούνται από m bits, επιτρέποντας την υποστήριξη $N \leq 2^m$ κόμβων και αντικειμένων στο σύστημα μας, με ένα μικρό αριθμό των κόμβων ($A \leq N$) να αποτελείται από κόμβους αλτρουιστές.

Η βασική ιδέα πίσω από το PLANES είναι ο τεμαχισμός των N κόμβων του δικτύου σε clusters όπου το καθένα είναι δομημένο με τη βοήθεια ενός Κατανεμημένου Πίνακα Κατακερματισμού (DHT). Για κάθε cluster δημιουργείται ένα δίκτυο βασισμένο σε DHT². Δεδομένου ότι το μέγεθος του cluster είναι S , και ο αριθμός των clusters C , τότε συνολικά θα υπάρχουν $C = \frac{N}{S}$ clusters. Θεωρούμε ότι κάθε κόμβος προκαλεί μία ελάχιστη επιβάρυνση στο δίκτυο για την συσχέτιση ενός αναγνωριστικού κόμβου με ένα αναγνωριστικό cluster. Αυτό μπορεί να γίνει για παράδειγμα με τη χρήση μίας συνάρτησης κατακερματισμού για την αντιστοίχιση των αναγνωριστικών των κόμβων με τα αναγνωριστικά των cluster. Επιπλέον, θεωρούμε ότι η πλειονότητα των κόμβων που ανήκουν σε ένα cluster είναι εγωιστές και ότι μέσα σε κάθε cluster υπάρχει τουλάχιστον ένα κόμβος αλτρουιστής.

Στην προσέγγιση μας, οι αλτρουιστές διατηρούν επιπλέον πίνακες δρομολόγησης, με τις εγγραφές τους να δείχνουν στους υπόλοιπους αλτρουιστές, δημιουργώντας έτσι ένα πλήρως συνδεδεμένο δίκτυο επικάλυψης αποτελούμενο από αλτρουιστές. Έτσι η επικοινωνία μεταξύ αλτρουιστών (και άρα μεταξύ των clusters) απαιτεί ένα μόνο βήμα³ (hop). Αργότερα θα αποδεσμευτούμε από την απαίτηση της υψηλής (ή πλήρους) συνδεσιμότητας. Μέσα σε κάθε DHT cluster, όλοι οι κόμβοι διατηρούν πίνακες δρομολόγησης για τους γείτονες τους, όπως άλλωστε απαιτείται από την δομή DHT του cluster. Έτσι, κάθε κόμβος έχει $O(\log S)$ γείτονες. Επίσης, όλοι οι κόμβοι διατηρούν δείκτες προς τους αλτρουιστές του cluster στο οποίο ανήκουν. Στο σχήμα 3.1 μπορεί κανείς να δει ένα παράδειγμα ενός διαστρωματωμένου δικτύου ομότιμων εταίρων.

Ένας κόμβος (είτε είναι αλτρουιστής είτε όχι) n_i συνδέεται στο δίκτυο ως συνήθως, με το να εντοπίσει την διεύθυνση οποιουδήποτε κόμβου που αποτελεί μέλος για παράδειγμα του cluster c_s . Επικοινωνώντας μαζί του, μπορεί να εντοπίσει την διεύθυνση του κόμβου αλτρουιστή που ανήκει στο cluster c_s . Στη συνέχεια μπορεί να προσδιοριστεί το αναγνωριστικό του cluster c_d στο οποίο ανήκει. Τέλος, ο κόμβος n_i συνδέεται στο DHT δίκτυο του συγκεκριμένου cluster, χρησιμοποιώντας το πρωτόκολλο DHT join ενώ παράλληλα ενημερώνεται και η εγγραφή στον πίνακα δρομολόγησής του που δείχνει στον αλτρουιστή κόμβο του συγκεκριμένου cluster. Η προσθήκη ενός νέου αλτρουιστή απαιτεί την επικοινωνία με το δίκτυο των αλτρουιστών ώστε να ενημερωθεί ο πίνακας δρομολόγησης του, αλλά και οι αντίστοιχες εγγραφές των υπολοίπων αλτρουιστών. Επιπλέον όλοι οι κόμβοι του cluster πρέπει να ενημερώσουν τον πίνακα δρομολόγησης τους, ώστε

²Στην πραγματικότητα, οποιαδήποτε άλλη δομή θα μπορούσε να χρησιμοποιηθεί αντί για ένα DHT, όπως για παράδειγμα τα [2, 82], αποφέροντας παρόμοια κέρδη απόδοσης.

³Στην πράξη, αναφερόμαστε σε ένα υψηλού βαθμού συνδεδεμένο δίκτυο αλτρουιστών και $O(1)$ βήματα κατά την επικοινωνία τους, μιας και πλήρης συνδεσιμότητα είναι δύσκολο να επιτευχθεί.

Σχήμα 3.1: Παράδειγμα Δικτύου με $N=30$, $C=6$, $S=5$

να δείχνει στον νέο αλτρουιστή. Η αποχώρηση ενός κόμβου επηρεάζει μόνο την DHT δομή του cluster στο οποίο ανήκει και επιτυγχάνεται καλώντας το πρωτόκολλο DHT leave. Η αποχώρηση ενός αλτρουιστή απαιτεί την ενημέρωση των κόμβων του cluster και μπορεί επίσης να κριθεί σκόπιμη η εύρεση ενός νέου αλτρουιστή τον οποίο και θα αντικαταστήσει (για παράδειγμα όταν είναι ο τελευταίος αλτρουιστής του cluster). Αυτό μπορεί να γίνει εξετάζοντας τον πίνακα δρομολόγησης του και επιλέγοντας ένα άλλο αλτρουιστή (συνήθως κάποιον που συσχετίζεται με μικρό αριθμό από clusters).

Έχοντας αυτή την υποδομή, η δρομολόγηση γίνεται σε δύο επίπεδα :

1. Μεταξύ των clusters, από ένα κόμβο σε έναν άλλο σε διαφορετικό cluster: δεδομένου του πλήρως συνδεδεμένου δικτύου επικάλυψης αλτρουιστών, η δρομολόγηση μίας αίτησης (μηνύματος) σε διαφορετικό cluster απαιτεί δύο βήματα (hops): ένα βήμα από τον κόμβο προς τον αλτρουιστή που ανήκει στο ίδιο cluster και ένα ακόμα βήμα από τον αλτρουιστή προς τον αλτρουιστή του διαφορετικού cluster.
2. Μεταξύ κόμβων που ανήκουν στο ίδιο cluster (συμπεριλαμβανομένων και των αλτρουιστών του cluster): η δρομολόγηση επιτυγχάνεται με την αξιοποίηση της DHT δομής που διαμορφώνει το δίκτυο μέσα στο cluster.

3.1.1 Δημιουργία Συστάδων Χρησιμοποιώντας Συνεπή Κατακερματισμό

Μία ενδιαφέρουσα εναλλακτική που αποσκοπεί στην διαμόρφωση/δημιουργία των clusters μπορεί να επιτευχθεί με την εφαρμογή αλγορίθμων Consistent Hashing [70], που χρησιμοποιούνται στα DHT ΔΟΕ με σκοπό την συσχέτιση αντικειμένων με κόμβους. Έχοντας αυτό κατά νου, για κάθε cluster c_i ορίζεται το αναγνωριστικό του (ID) από την σχέση: $ID(c_i) = \frac{2^m}{C} \times (i - 1)$, $i = 1, \dots, C$. Χρησιμοποιώντας consistent hashing, κάθε αναγνωριστικό κόμβου συσχετίζεται με ένα από τα C αναγνωριστικά των clusters: ο κόμβος n_j συσχετίζεται με το cluster c_i εάν $ID(c_i) < ID(n_j) < ID(c_{i+1})$. Το consistent hashing εξασφαλίζει ότι κατά μέσο όρο, όλα τα clusters θα αποτελούνται από ίσο αριθμό κόμβων. Έτσι, η συγκεκριμένη προσέγγιση τμηματοποιεί τον χώρο των αναγνωριστικών σε C συνεχόμενους υποχώρους, από τους οποίους επιλέγονται τα αναγνωριστικά των κόμβων (Σχήμα 3.2). Έτσι κάθε κόμβος χρησιμοποιεί τις λειτουργίες που προσφέρονται από την DHT δομή για να αποτελέσει μέλος του cluster που επιθυμεί. Στο Σχήμα 3.2 παρουσιάζεται η διαμόρφωση 6 clusters στο δίκτυο.

Σχήμα 3.2: Δημιουργία Συστάδων σε ένα ΔΟΕ Βασισμένο σε Αλτρουιστές – Ένα Δαχτυλίδι Chord από Δαχτυλίδια Chord

3.1.2 Δρομολόγηση

Ας υποθέσουμε ότι ο κόμβος n_s επιθυμεί την δρομολόγηση ενός μηνύματος στον κόμβο που είναι υπεύθυνος για το αναγνωριστικό $targetID$. Ο αλγόριθμος δρομολόγησης αποτελείται από τα παρακάτω βήματα:

- Ο κόμβος n_s στέλνει ένα μήνυμα στον αλτρουιστή a_s που ανήκει στο ίδιο cluster.
- Ο a_s συσχετίζει το μήνυμα με το τελικό cluster (π.χ. χρησιμοποιώντας κάποια συνάρτηση κατακερματισμού με όρισμα το $targetID$).
- Ο a_s χρησιμοποιεί τον πίνακα δρομολόγησης του για να εντοπίσει τον αλτρουιστή a_t του τελικού cluster και του προωθεί το μήνυμα.
- Ο a_t χρησιμοποιεί τον αλγόριθμο δρομολόγησης που προσφέρεται από το DHT για να στείλει το μήνυμα στον κατάλληλο κόμβο μέσα στο τελικό cluster.

Συνεπώς, με σκοπό να εντοπιστεί ένα αντικείμενο στο PLANES, πρώτα χρησιμοποιούμε μία συνάρτηση κατακερματισμού με το αναγνωριστικό του αντικειμένου σαν όρισμα ώστε να εντοπίσουμε το κατάλληλο cluster, και στη συνέχεια χρησιμοποιούμε τον αλγόριθμο δρομολόγησης του DHT για να εντοπίσουμε τον κατάλληλο κόμβο μέσα στο cluster όπου είναι αποθηκευμένο το αντικείμενο.

Για να παρουσιάσουμε την βελτίωση στην απόδοση δρομολόγησης που επιτυγχάνεται, ας παρουσιάσουμε πρώτα ένα κερδοφόρο σύνολο ρυθμίσεων για το PLANES. Ορίζουμε $C = \frac{N}{\log N}$ το οποίο έχει ως αποτέλεσμα $S = \log N$. Επιπλέον, κάθε cluster είναι δομημένο με την βοήθεια ενός DHT, όπως περιγράφηκε προηγουμένως. Αυτό το σύνολο ρυθμίσεων θα εξασφαλίσει δρομολόγηση μέσα στο cluster αποτελούμενη από $O(\log S) = O(\log(\log N))$ βήματα, όταν το σύστημα θα έχει σταθεροποιηθεί. Αυτή η διατύπωση είναι άμεση απόρροια του γεγονότος ότι το cluster είναι δομημένο με DHT και ότι η δρομολόγηση μεταξύ αλτρουιστών απαιτεί 1 βήμα.

«Δύσκολη» κατάσταση μπορεί να προκύψει όταν οι κόμβοι συνδέονται ή αποχωρούν από το δίκτυο με τόσο γρήγορους ρυθμούς, ώστε το DHT να μην μπορεί να διατηρήσει σχετικά «φρέσκια» πληροφορία δρομολόγησης (σχετικά με τους γειτονικούς κόμβους κάθε κόμβου). Ανάλογα με την DHT οργάνωση που έχουμε υιοθετήσει, η πιθανότητα να αντιμετωπίσουμε μια τέτοια δύσκολη κατάσταση είναι διαφορετική. Όπως κι αν έχει όμως, σε γρήγορα μεταβαλλόμενα δίκτυα, η απόδοση δρομολόγησης θα μειωθεί σε $O(N)$ βήματα. Δεδομένου ότι το DHT εγγυάται βήματα δρομολόγησης της τάξης του μεγέθους του δικτύου σε αυτή την περίπτωση, προκύπτει ότι με την αρχιτεκτονική μας, σε ένα γρήγορα μεταβαλλόμενο δίκτυο, η δρομολόγηση θα γίνεται σε $O(S) = O(\log N)$ βήματα,

δεδομένου των $O(1)$ βημάτων μεταξύ των αλτρουιστών. Η ύπαρξη των $\frac{N}{\log N}$ clusters υποδηλώνει την ύπαρξη τουλάχιστον $\frac{N}{\log N}$ αλτρουιστών. Ας σημειωθεί, όμως, ότι καθώς το N αυξάνει, απαιτείται ένα ολοένα μικρότερο ποσοστό του N : π.χ. για 1000 κόμβους, απαιτείται το 10% αυτών να είναι αλτρουιστές, ενώ για ένα εκατομμύριο μόνο το 5% από αυτούς πρέπει να είναι αλτρουιστές.

Ο ισχυρισμός μας λοιπόν είναι ότι για τη μεγαλύτερη πλειονότητα των κόμβων, απαιτείται η διατήρηση πολύ μικρότερης ποσότητας πληροφορίας σχετικής με τη δρομολόγηση. Αυτό είναι στην ουσία αποτέλεσμα της ίδιας της δομής του δικτύου: τα DHT των clusters είναι πολύ μικρότερα, δηλ. μεγέθους $O(\log N)$. Δεδομένου των χαρακτηριστικών των DHTs, υπάρχουν μόνο $\log(\log N)$ γείτονες ανά μη-αλτρουιστικό κόμβο, σε αντίθεση με τους $\log N$ γείτονες που απαιτούνται από ένα απλό DHT. Ένα πολύ μικρό ποσοστό των κόμβων του δικτύου (δηλ. οι αλτρουιστές) είναι δυνατόν να αποθηκεύουν και να διατηρούν σημαντικά μεγαλύτερη πληροφορία δρομολόγησης, κάτι που εξαρτάται από την αρχιτεκτονική που έχει επιλεγεί. Όμως, όπως ισχυριστήκαμε προηγουμένως, αυτοί είναι ιδιαίτεροι κόμβοι που είναι πρόθυμοι να αντεπεξέλθουν στο παραπάνω αυτό κόστος. Σε γενικές γραμμές η ποσότητα πληροφορίας σχετικά με τη δρομολόγηση που διατηρείται στην αρχιτεκτονική μας, είναι σημαντικά μικρότερη από μία απλή DHT αρχιτεκτονική.

Σε γρήγορα μεταβαλλόμενα δίκτυα, το κύριο μέλημα μας είναι η διατήρηση όσο το δυνατόν πιο «φρέσκιας» πληροφορίας σχετικά με τους πίνακες δρομολόγησης των αλτρουιστών, κάτι που μπορεί να είναι μερικές φορές πολύ δύσκολο ή ακόμα και αδύνατο, πράγμα που θα προκαλέσει ένα μη πλήρως συνδεδεμένο δίκτυο αλτρουιστών. Αρχικά εργαζόμαστε προς δύο συνιστώσες: (α) την μείωση του απαιτούμενου αριθμού αλτρουιστών και (β) την μείωση της ποσότητας πληροφορίας δρομολόγησης. Στη συνέχεια, θα παρουσιάσουμε μια αρχιτεκτονική που δεν απαιτεί πλήρως συνδεδεμένο δίκτυο αλτρουιστών.

3.1.3 Επιλογές & Επεκτάσεις Δόμησης Δικτύου

3.1.3.1 Το PLANES με Λιγότερους Αλτρουιστές

Σε αυτή την ενότητα παρουσιάζουμε μία επέκταση της αρχιτεκτονικής μας, όπου μπορεί να θυσιάσει κατά λίγο την ταχύτητα δρομολόγησης και να προσφέρει δραματική μείωση της ποσότητας πληροφορίας δρομολόγησης που απαιτείται για την διατήρηση του δικτύου αλτρουιστών. Ορίζουμε το δίκτυο ώστε να αποτελείται από $C = \frac{N}{z \times \log N}$ clusters, το οποίο έχει ως αποτέλεσμα κάθε cluster να έχει μέγεθος $S = z \times \log N$ για μία δεδομένη τιμή του z . Αυτό που αξίζει να προσέξει κανείς είναι ότι: (α) ο απαιτούμενος αριθμός αλτρουιστών (που είναι ίσος με C) μειώνεται κατά z και (β) ο αριθμός βημάτων κατά τη δρομολόγηση μίας αίτησης μέσα σε ένα cluster, αυξάνεται κατά $\log z$ εφόσον $\log S = \log(z \times \log N) = \log z + \log(\log N)$.

3.1.3.2 Το PLANES με Λιγότερους Αλτρουιστές ή/και Λιγότερη Ισχύ

Εναλλακτικά (ή συμπληρωματικά) της προηγούμενης αρχιτεκτονικής μπορεί κανείς να υιοθετήσει την ίδια αρχή στρωματοποίησης αναδρομικά στο επίπεδο όμως του δικτύου των αλτρουιστών. Το κίνητρο μας γι' αυτή την επιλογή είναι διπλό: (α) μείωση του κόστους διατήρησης της πληροφορίας δρομολόγησης που χρησιμοποιείται για την επίτευξη πλήρως συνδεδεμένου δικτύου αλτρουιστικών κόμβων και (β) εκμετάλλευση πιθανών ετερογενειών που μπορεί να υπάρχουν μεταξύ των αλτρουιστών. Στη σχήμα 3.3 παρουσιάζεται η προσέγγιση μας για ένα ΔΟΕ τριών επιπέδων. Το συγκεκριμένο δίκτυο αποτελείται από ένα ομαδοποιημένο δίκτυο αλτρουιστών με $A = 50,000$ αλτρουιστικούς κόμβους, το οποίο τμηματοποιείται σε $C = 390$ clusters (AltNets), με $S = 128$ αλτρουιστές το καθένα. Αυτό το σύνολο ρυθμίσεων προκύπτει εάν ορίσουμε $C = \frac{A}{z \times \log A}$ για το δίκτυο των A αλτρουιστών με $z = 8$.

Οι αλτρουιστές ανήκουν τώρα στο AltNet ή στο Alt2Net. Στο Σχήμα 3.3 το δίκτυο αποτελείται από $N = 1,000,000$ κόμβους, με 5% (50,000) από αυτούς να είναι αλτρουιστές και να

Σχήμα 3.3: Πολυεπίπεδο ΔΟΕ Βασισμένο σε Αλτρουιστές

οργανώνονται σε δύο επίπεδα (με 390 clusters στο πρώτο επίπεδο όπου το καθένα περιέχει 128 αλτρουιστές – AltNets – και 390 κόμβους στο δεύτερο επίπεδο – Alt2Net – έναν για κάθε cluster του AltNet). Κάθε DHT cluster (DHTNet) αποτελείται από 20 κόμβους. Κάθε DHTNet εξυπηρετείται από ένα AltNet, με κάθε AltNet να εξυπηρετεί 128 DHTNets. Όλα τα AltNets και το Alt2Net όπως και πριν είναι σχεδόν πλήρως συνδεδεμένα. Οι απαιτήσεις για την διατήρηση αυτής της συνεκτικότητας είναι πλέον ακόμα λιγότερες: οι αλτρουιστές στο AltNet (Alt2Net) πρέπει να έχουν 128 (390) γείτονες. Αυτές οι τιμές δείχνουν μία μείωση κατά 400 και 128 αντίστοιχα.

Την ίδια στιγμή η σημαντική βελτίωση στην ταχύτητα δρομολόγησης διατηρείται. Για ένα δίκτυο που αποτελείται από ένα εκατομμύριο κόμβους, αντί να απαιτούνται 2 βήματα για να φτάσουμε σε ένα συγκεκριμένο cluster, τώρα απαιτούνται 2 ή 4 βήματα για αυτό το σκοπό (εξαρτάται από το αν οι δύο κόμβοι εξυπηρετούνται από το ίδιο ή διαφορετικό AltNet). Έτσι συνολικά απαιτούνται περίπου 6-8 βήματα: 2-4 βήματα για να φτάσουμε στο cluster που επιθυμούμε και περίπου 4 βήματα για να φτάσουμε στον κατάλληλο κόμβο μέσα στο cluster. Αυτό μας δίνει μία βελτίωση στην ταχύτητα της τάξεως του 2.5 με 3.0, συγκρινόμενο με τα 20 βήματα που απαιτούνται σε μία απλή DHT αρχιτεκτονική. Θα πρέπει να σημειωθεί ότι η βελτίωση στην ταχύτητα σε ένα γρήγορα μεταβαλλόμενο περιβάλλον είναι πολλές τάξεις μεγέθους μεγαλύτερη. Επίσης, όσο αυξάνουμε τον αριθμό των επιπέδων τόσο μειώνεται η ποσότητα πληροφορίας που πρέπει να διατηρούν οι ισχυροί κόμβοι. Αυτό όμως έρχεται σε αντίθεση με τον αριθμό των βημάτων που απαιτούνται κατά την δρομολόγηση αιτήσεων, ο οποίος αυξάνεται με την αύξηση των επιπέδων. Ο ακριβής αριθμός επιπέδων εξαρτάται από τα χαρακτηριστικά του δικτύου και προφανώς από την ίδια την εφαρμογή και τις απαιτήσεις της.

3.1.3.3 Το PLANES Χωρίς Πλήρως Συνδεδεμένο Δίκτυο Αλτρουιστών

Το δίκτυο που σχηματίζουν οι αλτρουιστές μπορεί να δομηθεί χρησιμοποιώντας μία DHT δομή χωρίς έτσι να εξασφαλίζεται πλήρης συνδεσιμότητα. Ο αλγόριθμος δρομολόγησης στο επίπεδο των αλτρουιστών θα πρέπει πλέον να μεταβληθεί ώστε να συνδυαστούν τα βήματα 3 και 4 χρησιμοποιώντας τον αλγόριθμο δρομολόγησης που προσφέρει το DHT με σκοπό τον «εντοπισμό» του cluster που πρέπει να καταλήξει η αίτηση μας. Σε αυτή την προσέγγιση, οι αλτρουιστές σχηματίζουν ένα «δακτυλίδι» στα πρότυπα του Chord, με βάση το αναγνωριστικό τους. Επιπλέον, τα αναγνωριστικά των clusters συσχετίζονται στο ίδιο δακτυλίδι (όπως τα αναγνωριστικά των αντικειμένων συσχετίζονται στο δακτυλίδι του Chord). Με αυτό τον τρόπο οι αλτρουιστές είναι υπεύθυνοι για τα clusters, με τον ίδιο τρόπο που οι κόμβοι είναι υπεύθυνοι για αντικείμενα στο δίκτυο Chord. Η δρομολόγηση λοιπόν των αιτήσεων στο επίπεδο των αλτρουιστών (δηλ. από έναν αλτρουιστή σε ένα cluster σε έναν αλτρουιστή σε διαφορετικό cluster) γίνεται με δρομολόγηση στο Chord δίκτυο

των αλτρουιστών χρησιμοποιώντας το αναγνωριστικό του cluster-στόχου. Τα υπόλοιπα βήματα παραμένουν ως έχουν.

Ας υποθέσουμε, λοιπόν, ότι το δίκτυο μας αποτελείται από $N = 1,000,000$ κόμβους, με $S = C = \sqrt{N} = 1,000$. Με την εφαρμογή ενός DHT δικτύου για τους $C = 1,000$ αλτρουιστές δεν βελτιώνεται ο αριθμός βημάτων κατά την δρομολόγηση στην κατάσταση ηρεμίας του δικτύου: περίπου $\log C (=10)$ βήματα απαιτούνται κατά τη δρομολόγηση μέσω του DHT δικτύου των αλτρουιστών και $\log S (=10)$ βήματα κατά τη δρομολόγηση μέσα στο cluster. Παρόλα αυτά όμως, τα οφέλη που μπορούμε να αποκομίσουμε στην περίπτωση όπου το δίκτυο συνεχώς μεταβάλλεται είναι πολλά. Εφόσον μπορούμε να δεχτούμε ότι οι πίνακες δεικτών προς του γείτονες των αλτρουιστών μπορεί να διατηρηθούν ακόμα και σε γρήγορα μεταβαλλόμενο δίκτυο (χαρακτηριστικό που εγγυάται το DHT λόγω της σχεδίασης του), η δρομολόγηση στο δίκτυο των αλτρουιστών θα απαιτεί $O(\log C) = O(\log \sqrt{N})$ βήματα. Ακόμα όμως και στην απίθανη περίπτωση όπου οι πίνακες δεικτών προς τους γείτονες δεν θα μπορούν να διατηρηθούν ενημερωμένοι, η δρομολόγηση στο δίκτυο των αλτρουιστών θα απαιτεί $O(C) = O(\sqrt{N})$ βήματα. Η δρομολόγηση μέσα στο cluster από τη άλλη πλευρά, θα απαιτεί $O(S) = O(\sqrt{N})$ βήματα. Έτσι, συνολικά $O(\sqrt{N})$ βήματα απαιτούνται, κάτι που αποτελεί δραματική βελτίωσή συγκρινόμενο με την εφαρμογή ενός απλού DHT δικτύου, κάνοντας την προσέγγιση μας ιδιαίτερα ελκυστική. Τέλος, στην αρχιτεκτονική που παρουσιάστηκε στην ενότητα 3.1.3.2, το DHT δίκτυο των αλτρουιστών μπορεί να εφαρμοστεί μόνο στο ένα επίπεδο, προσφέροντας μικρότερο αριθμό βημάτων κατά την δρομολόγηση (είτε στην κατάσταση ηρεμίας είτε κάτω από συνεχώς μεταβαλλόμενο δίκτυο), ενώ παράλληλα απαιτείται ένα μικρό μόνο ποσοστό από πλήρως συνδεδεμένους κόμβους.

3.1.3.4 Παραδείγματα Αρχιτεκτονικών

Στο παράδειγμα που ακολουθεί θα ποσοτικοποιήσουμε τη βελτίωση που προσφέρει η αρχιτεκτονική μας. Για χάρη απλότητας στη παρουσίαση, δεν θα χρησιμοποιήσουμε τον συμβολισμό O , επειδή η σταθερά θα είναι η ίδια για όλες τις προσεγγίσεις μας και για τις αρχιτεκτονικές που βασίζονται σε DHTs.

Απόδοση Δρομολόγησης Μεγάλης Κλίμακας

Ας υποθέσουμε ότι το δίκτυο μας αποτελείται από ($N =$) ένα εκατομμύριο κόμβους. Σε ένα παραδοσιακό DHT δίκτυο η δρομολόγηση θα απαιτούσε ($\log N =$) 20 βήματα στη κατάσταση ηρεμίας. Σε υψηλά μεταβαλλόμενο περιβάλλον, η δρομολόγηση στο DHT μπορεί να επιτευχθεί με $O(N)$ βήματα – μια μη αποδεκτή κατάσταση. Στην προσέγγισή μας και εάν υποθέσουμε ότι το 5% (50,000) των κόμβων είναι αλτρουιστές, έχοντας έναν αλτρουιστή ανά cluster, τότε κάθε cluster θα αποτελείται από $\log N \approx 20$ κόμβους. Η δρομολόγηση μέσα σε κάθε τέτοιο cluster στη κατάσταση ηρεμίας, θα απαιτεί $\log(\log N) \approx 4$ βήματα, ενώ συνολικά αυτά θα είναι 6 εάν προσθέσουμε τα 2 βήματα που απαιτούνται για τον εντοπισμό των αλτρουιστών σε κάθε ένα από τα δύο clusters (το αρχικό και το τελικό).

Έτσι λοιπόν σε ίδιου μεγέθους δίκτυα, η απόδοση δρομολόγησης (δηλ. αριθμός βημάτων) βελτιώνεται κατά περίπου 3 φορές. Σε γρήγορα μεταβαλλόμενα δίκτυα (όπου οι περισσότεροι κόμβοι συνδέονται και αποχωρούν από το δίκτυο με υψηλούς ρυθμούς, με αποτέλεσμα οι πίνακες δεικτών προς τους γειτονικούς κόμβους να μην είναι πλήρως ενημερωμένοι), η δρομολόγηση στην προσέγγισή μας απαιτεί $O(S) = O(\log N) \approx 20$ βήματα στο DHT δίκτυο, ενώ απαιτούνται επιπλέον 2 βήματα για τον εντοπισμό του τελικού cluster. Η συγκεκριμένη αρχιτεκτονική έχει ως αποτέλεσμα την βελτίωση στην απόδοση κατά αρκετές τάξεις μεγέθους.

Απόδοση Δρομολόγησης Μικρής Κλίμακας

Σε μικρότερου μεγέθους δίκτυα, με περίπου χίλιους κόμβους, οι παραδοσιακές λύσεις προσφέρουν απόδοση δρομολόγησης περίπου της τάξεως των ($\log N \Rightarrow$) 10 βημάτων στην κατάσταση ηρεμίας και των 1,000 βημάτων σε καταστάσεις όπου το δίκτυο μεταβάλλεται συνεχώς. Στην προσέγγισή μας, απαιτούνται περίπου 3 βήματα (δηλ. $\log(\log 1000)$) στην κατάσταση ηρεμίας και περίπου $S = \log 1000 = 10$ βήματα σε καταστάσεις συνεχούς μεταβαλλόμενου δικτύου, κάτι το οποίο έχει ως αποτέλεσμα βελτίωση στην απόδοση κατά 3 φορές (εάν λάβουμε υπόψη μας και τα βήματα στο δίκτυο των αλτρουιστών) και κατά 2 τάξεις μεγέθους αντίστοιχα.

Με την ύπαρξη ενός εκατομμυρίου κόμβων και $z = 6$, προκύπτουν $C \approx 8335$ clusters, πράγμα που έχει ως αποτέλεσμα την μείωση του κόστους ενημέρωσης και διατήρησης της πληροφορίας των πινάκων δρομολόγησης κατά 6 περίπου φορές (μειώνοντας τον αριθμό των αλτρουιστών από 50,000 σε 8335). Από την άλλη όμως, η απόδοση δρομολόγησης αυξάνεται από τα < 5 βήματα που είναι στο DHT cluster στα < 7 βήματα ($\log 120 \approx 7$). Πηγαίνοντας ένα βήμα παραπέρα και χρησιμοποιώντας $z = 50$, το αποτέλεσμα είναι η ύπαρξη 1,000 clusters, όπου καθένα απ' αυτά αποτελείται από 1,000 κόμβους. Η δρομολόγηση σε κάθε cluster απαιτεί τώρα 10 βήματα, ενώ ο αριθμός των απαιτούμενων αλτρουιστικών κόμβων μειώνεται κατά 50 φορές.

Έτσι, μπορούμε ακόμα να επιτύχουμε σημαντική βελτίωση κατά την δρομολόγηση (κατά 2 φορές περίπου στη κατάσταση ηρεμίας και κατά μερικές τάξεις μεγέθους σε συχνά μεταβαλλόμενα περιβάλλοντα). Την ίδια στιγμή, το κόστος διατήρησης υψηλής συνδεσιμότητας στο δίκτυο των αλτρουιστών μειώνεται δραματικά, μιας και ο αριθμός των αλτρουιστών μειώνεται εξίσου δραματικά. Τέλος, αξίζει να σημειωθεί, ότι η ύπαρξη πολύ μικρού ποσοστού κόμβων αλτρουιστών (0.1%) είναι μία ρεαλιστική επιλογή όπως μπορεί κανείς να δει στη σχετική βιβλιογραφία [5, 114, 128].

Μέγεθος Δομών Δεικτοδότησης/Δρομολόγησης

Ας συγκρίνουμε τις απαιτήσεις σε ποσότητα πληροφορίας σχετική με τη δρομολόγηση, μεταξύ ενός απλού DHT δικτύου και της προσέγγισης που παρουσιάστηκε στην ενότητα 3.1.3.2, για ένα δίκτυο αποτελούμενο από ένα εκατομμύριο κόμβους. Σε ένα δίκτυο DHT κάθε κόμβος πρέπει να διατηρεί $\log N$ δείκτες προς τους γείτονές του, κάτι που έχει ως αποτέλεσμα την ύπαρξη $(1,000,000 \times 20 \Rightarrow) 20,000,000$ εγγραφές στους πίνακες δρομολόγησης σε ολόκληρο το δίκτυο. Στην προσέγγιση της ενότητας 3.1.3.2 υποθέτουμε ότι ένα 5% των κόμβων είναι αλτρουιστές. Οι υπόλοιποι κόμβοι (950,000) οργανώνονται σε DHT clusters, καθένα με μέγεθος ίσο με $\log N$ και έτσι κάθε κόμβος σε αυτά τα clusters διατηρεί $\log(\log N) = 4$ εγγραφές καθώς επίσης και μία επιπλέον εγγραφή με δείκτη στον αλτρουιστή του δικτύου. Συνολικά λοιπόν, στην προσέγγισή μας, απαιτούνται $(950,000 \times 5 \Rightarrow) 4,750,000$ εγγραφές στο επίπεδο των clusters. Καθένας από τους 128 αλτρουιστές του δικτύου AltNet, πρέπει να διατηρεί 127 εγγραφές για να είναι το δίκτυο πλήρως συνδεδεμένο και 4 εγγραφές για τους γείτονες του στο DHT cluster. Εφόσον λοιπόν υπάρχουν 390 AltNet δίκτυα, συνολικά θα πρέπει να διατηρούνται $(390 \times 128 \times 131 \Rightarrow) 6,539,520$ εγγραφές στο δεύτερο επίπεδο. Τέλος στο υψηλότερο επίπεδο (Alt2Net) υπάρχουν 390 κόμβοι. Ο καθένας από αυτούς θα πρέπει να διατηρεί 389 εγγραφές για να παραμείνει το δίκτυο πλήρως συνδεδεμένο και 4 εγγραφές για το DHT του, οπότε συνολικά προκύπτουν $(390 \times 393 \Rightarrow) 153,270$ εγγραφές. Ο συνολικός λοιπόν αριθμός εγγραφών για όλα τα επίπεδα είναι 11,442,790.

Στην αρχιτεκτονική που παρουσιάστηκε στην ενότητα 3.1.3.3 με ένα δομημένο δίκτυο αλτρουιστών που στηρίζεται σε DHT, απαιτείται ακόμα μικρότερη ποσότητα πληροφορίας δρομολόγησης. Ο καθένας από τους 999,000 μη-αλτρουιστές κόμβους, πρέπει να διατηρεί 10 εγγραφές για το cluster στο οποίο ανήκει και επιπλέον μία εγγραφή με δείκτη στον αλτρουιστή του cluster, πράγμα που έχει ως αποτέλεσμα συνολικά την ύπαρξη $(999,000 \times 11 \Rightarrow) 10,989,000$ εγγραφές. Ο καθένας από τους 1,000 αλτρουιστικούς κόμβους, πρέπει να διατηρεί 20 εγγραφές (10 για το DHT δίκτυο των αλτρουιστών και 10 για το cluster στο οποίο ανήκει). Έτσι, συνολικά απαιτούνται

($1,000 \times 20 \Rightarrow$) 20,000 εγγραφές, ανεβάζοντας τον συνολικό αριθμό εγγραφών στα 11,009,000. Έτσι, η συνολική μείωση του αριθμού των εγγραφών που επιτυγχάνεται με αυτή τη προσέγγιση είναι περίπου κατά 2 φορές. Επιπλέον, η πληροφορία που διατηρείται και έχει να κάνει με τη δρομολόγηση δεν κατανέμεται ομοιόμορφα στους κόμβους, αλλά οι αλτρουιστές πρέπει και μπορούν να διατηρούν πολύ περισσότερη πληροφορία: μία σχεδιαστική επιλογή με ως επί το πλείστον θετικά χαρακτηριστικά.

3.1.4 Αποτελεσματική Προσαρμογή σε Τοπολογικές Αλλαγές

Το μέγεθος των πινάκων δρομολόγησης αποτελεί καθοριστικό παράγοντα στην απόδοση του συστήματος, καθώς μεγαλύτεροι πίνακες απαιτούν και μεγαλύτερη προσπάθεια ώστε να διατηρούνται ενημερωμένοι. Οι αρχιτεκτονικές που μόλις παρουσιάσαμε, επιτρέπουν στους περισσότερους κόμβους να διατηρούν σχετικά μικρούς πίνακες δρομολόγησης σε σχέση με τους πίνακες δρομολόγησης ενός απλού DHT δικτύου. Αυτό έχει ως αποτέλεσμα μικρότερο κόστος για την ενημέρωση αυτών των πινάκων. Επιπλέον, οι αλτρουιστές έχουν από τη μια αυξημένη υπολογιστική ισχύ και υψηλό εύρος ζώνης προκειμένου να αντεπεξέλθουν στον αυξημένο φόρτο εργασίας, ενώ από την άλλη ο αυξημένος φόρτος εργασίας δεν είναι ανάλογος του αυξημένου αριθμού εγγραφών στον πίνακα δρομολόγησης τους, αλλά πολύ μικρότερος. Αυτό προκύπτει από το γεγονός ότι η επιπλέον πληροφορία δρομολόγησης αφορά κυρίως τους υπόλοιπους αλτρουιστές οι οποίοι παραμένουν συνεχώς συνδεδεμένοι μεταξύ τους.

3.1.5 Δυναμική Προσαρμογή Ρυθμίσεων: Χρειάζεται Ακόμα;

Μία πιθανή ερώτηση που θα μπορούσε να διατυπωθεί είναι εάν οι τιμές των C και S θα πρέπει να προσαρμόζονται στις μεταβολές του αριθμού των κόμβων του δικτύου, N . Η προσαρμογή αυτών των τιμών θα προκαλούσε ιδιαίτερα χρονοβόρες διαδικασίες οργάνωσης του δικτύου. Η απάντηση, όπως και να 'χει, είναι αρνητική. Μπορούμε να ρυθμίσουμε το δίκτυο στα πρότυπα ενός μέσου μεγέθους δικτύου. Αν υποθέσουμε ότι έχουμε τις ρυθμίσεις που παρουσιάστηκαν νωρίτερα, σε ένα δίκτυο με $N = 1,000,000$ κόμβους, θα περιμέναμε το 1% από αυτούς να είναι αλτρουιστές, το $z = 6$ με αποτέλεσμα $C = 8,335$ και $S = 120$. Οι ίδιες ακριβώς ρυθμίσεις προσφέρουν τις ίδιες βελτιώσεις για δίκτυα αποτελούμενα από 100,000 μέχρι 10,000,000 κόμβους. Έχοντας $N = 100,000$ κόμβους, προκύπτουν 12 κόμβοι ανά cluster, πράγμα που έχει ως αποτέλεσμα την προβλεπόμενη βελτίωση (3 βήματα δρομολόγησης σε κάθε cluster). Από την άλλη, θα είχαμε μόνο 1,000 αλτρουιστές όπου ο καθένας θα κάλυπτε περισσότερα clusters (8) έτσι ώστε το κάθε cluster να έχει τουλάχιστον ένα αλτρουιστή. Με ένα δίκτυο αποτελούμενο από $N = 10,000,000$ κόμβους, θα πρέπει να υπάρχουν 1200 κόμβοι σε κάθε cluster, πράγμα που προκαλεί την επιθυμητή βελτίωση (10 βήματα δρομολόγησης σε κάθε cluster). Σε αυτή την περίπτωση οι αλτρουιστές θα είναι 100,000.

3.1.6 Σημεία Συσσωρευμένης Δρομολόγησης

Η μεγαλύτερη πλειονότητα των κόμβων (δηλ. οι κόμβοι που δεν ανήκουν στην κατηγορία των αλτρουιστών), καταπιάνονται με διαδικασίες δρομολόγησης αιτήσεων που κατευθύνονται σε κόμβους του ίδιου με αυτούς cluster, με αποτέλεσμα να θωρακίζονται από την εξωτερική κίνηση. Έτσι η πιθανότητα να αποτελέσουν σημείο συσσώρευσης δρομολόγησης (routing hotspots) είναι πάρα πολύ μικρή. Οι αλτρουιστές από την άλλη, αντιμετωπίζουν μεγαλύτερο φόρτο εργασίας, όμως έχουν τους απαιτούμενους πόρους ώστε να αντιμετωπίσουν τέτοιες καταστάσεις. Ας σημειωθεί, ότι σε μη δομημένα δίκτυα οι αλτρουιστές αναλαμβάνουν το 90% του φόρτου εργασίας του δικτύου. Στην προσέγγισή μας, οι επιπλέον λειτουργίες που καλούνται να εκτελέσουν αφορούν μόνο την

δρομολόγηση αιτήσεων κάτι που συγκριτικά είναι πολύ μικρότερο. Έτσι πιστεύουμε, ότι τα σημεία συσσώρευσης αιτήσεων δρομολόγησης δεν προκαλούν ιδιαίτερη ανησυχία.

3.1.7 Ευρωστία

Σε ένα δίκτυο με αλτρουιστές και DHT clusters η ευρωστία του δικτύου κατά τη διαδικασία δρομολόγησης υπό την παρουσία βλαβών είναι αποτέλεσμα της σχεδίασης και των μηχανισμών που προσφέρονται από το DHT. Για το δίκτυο των αλτρουιστών, η υψηλή συνδεσιμότητα (αν υπάρχει) προσφέρει ευρωστία. Στο επίπεδο των DHT clusters, η ευρωστία είναι αποτέλεσμα της δομής του cluster (δηλ. των μηχανισμών που προσφέρει κάθε DHT). Έτσι, οι προσεγγίσεις μας εγγυώνται τόσο ευρωστία, όσο είναι δυνατό να προσφέρει ένα DHT. Παρ' όλα αυτά, εάν ένας αλτρουιστής πάψει να λειτουργεί, τότε είναι δυνατό ένα ολόκληρο cluster να αποκοπεί από τα υπόλοιπα. Για να αντιμετωπίσουμε αυτή την κατάσταση συσχετίζουμε περισσότερους τους ενός αλτρουιστές ανά cluster, κάτι το οποίο συνοδεύεται από ένα μικρό αριθμό επιπλέον εγγραφών στους πίνακες δρομολόγησης τους (αυτές τις εγγραφές με δείκτες προς τους επιπλέον γείτονες τους, $\log S = \log(\log N)$ για κάθε επιπλέον cluster στο οποίο ανήκουν). Με αυτή την επιλογή είναι δυνατόν να αποφευχθούν σημεία συσσώρευσης αιτήσεων δρομολόγησης σε ορισμένους αλτρουιστές. Η συσχέτιση ενός αλτρουιστή με περισσότερα από ένα clusters μπορεί να γίνει με την εφαρμογή πολλαπλών συναρτήσεων κατακερματισμού κατά την διαδικασία σύνδεσης τους στο δίκτυο. Αυτές επίσης χρησιμοποιούνται κατά την δρομολόγηση, προσπαθώντας να επιλέξουν εναλλακτικούς αλτρουιστές. Γενικά και αφού η ποσότητα πληροφορίας σχετικά με τη δρομολόγηση στις προσεγγίσεις μας είναι σχετικά μικρή, μπορούμε να επιβαρυνθούμε τις επιπλέον εγγραφές που αφορούν τους γείτονες, επιτυγχάνοντας έτσι περισσότερη ευρωστία, με μία μικρή μείωση των βελτιώσεων που μέχρι τώρα απολαμβάνουμε στον τομέα του μεγέθους των πινάκων δρομολόγησης.

Επιπλέον, η βελτίωση της ευρωστίας στο PLANES είναι απόρροια του διαχωρισμού του ρόλου που έχει ο κάθε κόμβος στο δίκτυο. Συγκεκριμένα, το αποτέλεσμα της εγωιστικής συμπεριφοράς που ορισμένοι κόμβοι ενδεχομένως να έχουν, επηρεάζει ένα πολύ μικρότερο αριθμό κόμβων (από $O(\log N)$ σε $O(\log(\log N))$). Για παράδειγμα, σε ένα DHT δίκτυο ενός εκατομμυρίου κόμβων, η εγωιστική συμπεριφορά ενός κόμβου θα επηρεάσει 20 κόμβους. Στο PLANES, θα επηρεάσει 4 κόμβους. Εάν το δούμε από μία άλλη οπτική γωνία, μία διαδρομή 20 βημάτων σε ένα δίκτυο ενός εκατομμυρίου κόμβων, θα αποτελείται από 19 εγωιστικούς κόμβους (εάν θεωρήσουμε ότι το 95% των κόμβων είναι εγωιστές). Στο PLANES, με το ίδιο ποσοστό εγωιστικών κόμβων, κάθε διαδρομή θα αποτελείται από 4 εγωιστικούς κόμβους. Αυτό το γεγονός χαρακτηρίζει το PLANES σαν ένα εύρωστο και ανθεκτικό στις βλάβες και στις αλλαγές της τοπολογίας, δίκτυο.

3.2 AltSeAl: Εγγύηση Αλτρουισμού

Το Seal [96] είναι ένα «στρώμα» λογισμικού που έχουμε αναπτύξει και σκοπό έχει να εντοπίζει τους κόμβους με εγωιστική συμπεριφορά, ώστε να τους δίνει κίνητρα να συμπεριφέρονται δίκαια με σκοπό να βελτιώσουν την εξισορρόπηση του φόρτου εργασίας στο δίκτυο και γενικά την συνολική απόδοση διαχείρισης της αποθηκευμένης στο δίκτυο πληροφορίας. Το Seal, μπορεί να ενσωματωθεί με διαφανή τρόπο σε δομημένα και μη δομημένα δίκτυα. Σε αυτή την ενότητα, παρουσιάζουμε το AltSeAl, μία τροποποιημένη εκδοχή του Seal, που ικανοποιεί τις ανάγκες του AESOP.

3.2.1 Το Επίπεδο Επίθληψης/Καταγραφής

Η βασική ιδέα είναι ότι όλες οι συναλλαγές μεταξύ των κόμβων μεταφράζονται σε δημιουργία δεδομένων (τα οποία ονομάζονται «Αποδείξεις Συναλλαγής» ή «Transaction Receipts» – TRs), τα

οποία τα μεταχειριζόμαστε όπως περίπου και τις «χάρες» στη πραγματική ζωή: οι κόμβοι που προσφέρουν χάρες (δηλ. προσφέρουν πόρους) κερδίζουν το δικαίωμα να ζητήσουν από τους κόμβους που δέχονται τις προσφορές τους είτε να τους το ανταποδώσουν στο μέλλον είτε να «τιμωρηθούν» διαφορετικά. Όλες αυτές οι λειτουργίες εκτελούνται με τρόπο διαφανή προς τον χρήστη. Οι κόμβοι παρακολουθούν και καταγράφουν τις χάρες που πρόσφεραν ή δέχθηκαν (δηλ. αποθηκεύουν τα αντίστοιχα TRs) σε δύο λίστες: στην F_d («Favors-Done») καταγράφουν τις χάρες που έχουν προσφέρει ενώ στην F_o («Favors-Owed») τις χάρες που χρωστάνε. Επιπλέον, κάθε κόμβος στο AltSeAl χαρακτηρίζεται από τον «βαθμό αλτρουισμού» (συμβολίζεται με $n_i.A$). Το $n_i.A$ είναι στην ουσία μία συνάρτηση των $|F_d|$ και $|F_o|$, όπου με $|X|$ συμβολίζεται το μέγεθος του συνόλου X . Για παράδειγμα θα μπορούσαμε να θεωρήσουμε τα $|F_d| - |F_o|$ ή $\frac{|F_d|}{|F_o|}$ σαν πιθανές συναρτήσεις για το $n_i.A$.

Εάν ο n_1 προσφέρει τον πόρο r_1 και ο κόμβος n_2 χρησιμοποιεί αυτό τον πόρο, τότε ο μηχανισμός «λιστών-χαρών» θα ενεργοποιηθεί και θα επιτρέψει στον n_1 να προωθήσει πιθανές μελλοντικές αιτήσεις για τον πόρο r_1 στον n_2 . Στο AltSeAl, οι κόμβοι αυτόνομα και ανεξάρτητα θέτουν ένα άνω ($n_i.A_{max}$) και ένα κάτω ($n_i.A_{min}$) όριο στο βαθμό αλτρουισμού τους. Εάν ο βαθμός τους είναι πάνω από $n_i.A_{max}$ τότε πάντα προωθούν τις αιτήσεις που δέχονται (αν αυτό είναι δυνατό) ενώ εάν ο βαθμός τους είναι κάτω από $n_i.A_{min}$ τότε δεν προωθούν καμία αίτηση. Σε όλες τις άλλες περιπτώσεις, ο κόμβος με μία πιθανότητα που μπορεί να ρυθμιστεί, μπορεί να επιλέξει είτε να προωθήσει είτε να εξυπηρετήσει μία αίτηση.

3.2.2 Χάρες και Παράπονα

Στο προηγούμενο σενάριο, εάν ο κόμβος n_2 εξυπηρετήσει μία προωθημένη αίτηση, τότε θεωρείται ότι έχει ανταποδώσει την αντίστοιχη χάρη. Εναλλακτικά, ο n_1 μπορεί να επιλέξει να χρησιμοποιήσει το αντίστοιχο TR (δηλ. $TR_{n_1}^{n_2}(r_1)$) σαν μέσο για να κατηγορήσει τον n_2 για εγωισμό. Αυτό επιτυγχάνεται με τον εξής τρόπο: Το AltSeAl χρησιμοποιεί το ίδιο το DHT για να αποθηκεύει «κατηγορίες». Ο κόμβος n_1 στέλνει το $TR_{n_1}^{n_2}(r_1)$ σε ένα κατάλληλο κόμβο – ας πούμε τον n_3 – χρησιμοποιώντας τους μηχανισμούς που προσφέρει το DHT (π.χ. περνώντας το TR σαν όρισμα στην συνάρτηση κατακερματισμού του DHT). Στη συνέχεια ο n_3 λειτουργεί σαν διαιτητής για τους n_1 και n_2 : μπορεί να ζητήσει και από τους δύο κόμβους να επαληθεύσουν το $TR_{n_1}^{n_2}(r_1)$ και να ζητήσει από τον n_2 να ανταποδώσει την χάρη. Εάν η επαλήθευση είναι επιτυχημένη αλλά ο n_2 συνεχίσει να αρνείται να ανταποδώσει τη χάρη, τότε ο n_3 αποθηκεύει το $TR_{n_1}^{n_2}(r_1)$ έτσι ώστε να μπορούν να γνωρίζουν γι' αυτό και οι υπόλοιποι κόμβοι του δικτύου. Εάν η επαλήθευση αποτύχει, τότε ο n_3 μπορεί με τη σειρά του να «κατηγορήσει» τον κόμβο που ψεύδεται.

Ένα ενδιαφέρον χαρακτηριστικό αυτού του μηχανισμού, είναι ότι εάν κάποια στιγμή ο n_2 αποφασίσει να συμπεριφερθεί αλτρουιστικά, μπορεί να συλλέξει όλες τις κατηγορίες που τον αφορούν και στη συνέχεια να ανταποδώσει επιλεκτικά τις χάρες που «οφείλει», ώστε σταδιακά να βελτιώσει την συμπεριφορά του. Πρέπει να σημειωθεί, ότι ενώ οι «κατηγορίες» αποθηκεύονται στο DHT, τα TRs που αφορούν χάρες που δόθηκαν ή ανταποδόθηκαν, αποθηκεύονται τοπικά στο κόμβο αλτρουιστή. Επιπλέον, για να διατηρήσουμε τις απαιτήσεις σε χώρο αποθήκευσης καθώς ο χρόνος περνάει και ο συγκεκριμένος μηχανισμός λειτουργεί, χρησιμοποιούμε ένα σύστημα παλαιώσης για τα αποθηκευμένα TRs.

Τα TRs δημιουργούνται χρησιμοποιώντας μηχανισμούς ισχυρής κρυπτογράφησης, ενώ οι κόμβοι στο AltSeAl είναι εξοπλισμένοι με δημόσια/ιδιωτικά ζεύγη κλειδιών και ταυτοποιούνται χρησιμοποιώντας μία σύνοψη (digest) του δημοσίου κλειδιού τους, που επίσης χρησιμοποιείται για την επαλήθευση των TRs. Έτσι, οι κόμβοι δεν μπορούν να δώσουν απομιμήσεις TRs αλλά ούτε και να αρνηθούν την εγκυρότητα ενός TP, εκτός και αν αλλάξουν το ζεύγος κλειδιών τους. Επιπλέον, στο AltSeAl εφαρμόζεται ένας μηχανισμός ανάδρασης και όχι ένας μηχανισμός τιμωρίας: οι αιτήσεις μπαίνουν σε μία ουρά αναμονής και εξυπηρετούνται με σειρά προτεραιότητας, ενώ οι πόροι που έχουν δεσμευθεί για να εξυπηρετήσουν αυτές τις αιτήσεις (π.χ. εύρος ζώνης, αποθηκευτικός χώρος)

μεταβάλλονται με βάση το συνολικό «βαθμό» αλτρουισμού που χαρακτηρίζει τον κάθε κόμβο. Τέλος, κάθε κόμβος ξεκινάει να δραστηριοποιείται στο δίκτυο με τον χειρότερο δυνατό «βαθμό», ώστε να μην έχει το κίνητρο να αλλάξει την ταυτότητα του (ζεύγος κλειδιών) ή να ξεκινήσει επίθεση τύπου Sybil [31].

3.3 SeAled PLANES: AltSeAl και AltNets

Στο AESOP πρέπει να προσφέρεται η δυνατότητα να μπορεί κανείς να εντοπίσει τα αναγνωριστικά (IDs) ενός αριθμού από αλτρουιστικούς κόμβους, ασύγχρονα αναφορικά με το πότε χαρακτηρίστηκαν ως αλτρουιστές. Στο περιβάλλον του AltSeAl, ο αλτρουισμός μπορεί να εκφραστεί χρησιμοποιώντας τον «βαθμό αλτρουισμού». Για παράδειγμα, μόνο οι κόμβοι των οποίων ο βαθμός είναι $\frac{|F_d|}{|F_o|} \geq 2$ μπορούν να θεωρηθούν ότι ανήκουν στην κατηγορία των αλτρουιστών. Επιπλέον, αποδείξεις αλτρουισμού είναι απαραίτητο να υπάρχουν. Αυτό μπορεί να επιτευχθεί με τα TRs των χαρών που έχουν δοθεί ή ανταποδοθεί⁴.

Τέλος, στο AESOP υπάρχει η ανάγκη ύπαρξης μίας δομής για την διαχείριση των αλτρουιστών. Γι' αυτό το σκοπό χρησιμοποιούμε μια δεύτερη δομή βασισμένη σε DHT (π.χ. Chord [121]) αποκλειστικά για τους αλτρουιστές - AltDHT. Μόλις κάποιος κόμβος n αποδειχθεί ότι είναι αλτρουιστής, καλείται η ρουτίνα $AddToAltDHT(v)$. Αυτή η ρουτίνα:

1. ενημερώνει τον κόμβο n' που είναι υπεύθυνος για τις «κατηγορίες» που αφορούν τον κόμβο n .
2. ο n' αναλαμβάνει να επικοινωνήσει με τον n και να ανακτήσει τις «λευκές εγγραφές» και να τις ελέγξει σε συνδυασμό με τις «μαύρες εγγραφές» που είναι τοπικά αποθηκευμένες.
3. εάν αυτός ο έλεγχος είναι επιτυχημένος (δηλ. ο βαθμός αλτρουισμού του n είναι μεγαλύτερος από το όριο που έχει τεθεί), ο n' υπολογίζει ένα νέο αναγνωριστικό (π.χ. με το να περάσει σαν όρισμα στη συνάρτηση κατακερματισμού το αλφαριθμητικό «Altruist» ενωμένο με το αναγνωριστικό του n) και χρησιμοποιώντας τον μηχανισμό προσθήκης ενός νέου κόμβου στο DHT, να εισάγει τον n στο AltDHT. Εάν ο έλεγχος αποτύχει, τότε ο n' επιστρέφει ένα μήνυμα λάθους στον n .
4. όταν ένας κόμβος προαχθεί στο AltDHT θεωρείται ότι πλέον έχει ιδιαίτερες ευθύνες (π.χ. δρομολόγησης)
5. όταν ένας κόμβος χάσει τον χαρακτηρισμό αλτρουιστής (π.χ. όταν ο βαθμός αλτρουισμού του πέσει κάτω από το όριο που έχει τεθεί στο σύστημα), τότε αφαιρείται από το AltDHT χρησιμοποιώντας τον αντίστοιχο μηχανισμό που παρέχει το DHT.

Ας σημειωθεί ότι οι κόμβοι που βρίσκονται στο AltDHT, έχουν ένα φυσικό κίνητρο ώστε να μην «εξαπατούν» το σύστημα παραμένοντας στο AltDHT ενώ επιθυμούν να μην έχουν αλτρουιστική συμπεριφορά, με το να δέχονται παραπάνω από το κανονικό φόρτο εργασίας. Επιπλέον, κάθε κόμβος στο AltDHT μπορεί να εκτελέσει τυχαίους ελέγχους: περιοδικά επιλέγει ένα τυχαίο αναγνωριστικό n) (από αυτά που χρησιμοποιούνται στο AltDHT) και υπολογίζει τον βαθμό αλτρουισμού. Οι κόμβοι που ανακαλύπτεται ότι έχουν «εξαπατήσει», αποβάλλονται από το AltDHT! Μία πιο εμπειρισταωμένη (αλλά και παράλληλα ιδιαίτερα απαιτητική σε πόρους) προσέγγιση μπορεί να βρεθεί στην σχετική βιβλιογραφία [96]. Έχοντας λοιπόν αυτή την υποδομή, το να ανακαλύψει κανείς αλτρουιστές είναι σχετικά εύκολο. Για παράδειγμα, ο εντοπισμός k αλτρουιστών απαιτεί τον υπολογισμό ενός τυχαίου αναγνωριστικού και την αξιοποίηση του AltDHT με σκοπό να εντοπίσουμε τον

⁴Θα αναφερόμαστε στα «θετικά» TRs ως *λευκές εγγραφές* και στα «αρνητικά» TRs ως *μαύρες εγγραφές*.

Σχήμα 3.4: Η Αρχιτεκτονική RangeGuard. Οι κόμβοι RG δημιουργούν ένα δεύτερο δικό τους δαχτυλίδι DHT-ΔΤ, παίρνοντας υπ' ευθύνη τους (χρησιμοποιώντας συνεπή κατακερματισμό) διαστήματα κόμβων του κατώτερου δαχτυλιδιού

κόμβο που είναι υπεύθυνος για το συγκεκριμένο αναγνωριστικό και στη συνέχεια να ακολουθήσουμε τους k δείκτες προς τους επόμενους (successors) από αυτόν κόμβους (εάν για παράδειγμα το AltDHT έχει υλοποιηθεί με το Chord).

3.4 Επεξεργασία Ερωτημάτων Εύρους Βασιζόμενη σε Αλτρουιστές

Το επόμενο βήμα σε αυτή την κατεύθυνση της έρευνάς μας είναι να εκμεταλλευτούμε την υποδομή και τις ιδέες που παρουσιάσαμε μέχρι τώρα για την επεξεργασία ερωτημάτων εύρους. Το αποτέλεσμα της προσπάθειάς μας είναι η αρχιτεκτονική RangeGuard [94]. Η πρόθεσή μας είναι να σχηματίσουμε ένα δεύτερο DHT-ΔΤ δακτύλιο, το δακτύλιο RangeGuard (Σχήμα 3.4), που αποτελείται από ισχυρούς κόμβους - τους RangeGuards (ή απλά RG s) - επιφορτισμένο με επιπλέον λειτουργικότητα. Κάθε κόμβος αυτού του είδους είναι υπεύθυνος για την αποθήκευση των πλειάδων δεικτοδοτησης (index tuples) που είναι τοποθετημένες σε κόμβους ανάμεσα στον προηγούμενο το RG και του ίδιου. Συνεπώς αν υπάρχουν M RG κόμβοι στο σύστημα, χωρίζουν τον κανονικό DHT-ΔΤ δακτύλιο σε M συνεχόμενες και μη επικαλυπτόμενες περιοχές. Ο κάθε RG διατηρεί πληροφορίες δρομολόγησης τόσο για τον δακτύλιο χαμηλότερου επιπέδου, όσο και για τον δακτύλιο RG . Επιπλέον, υπάρχει μια απευθείας σύνδεση από κάθε κόμβο στον επόμενο RG στον δακτύλιο ανώτερου επιπέδου (δηλ. στον RG που είναι υπεύθυνος για τον κόμβο). Οι κόμβοι στον χαμηλότερου επιπέδου δακτύλιο ελέγχουν την κατάσταση του RG τους (π.χ. σαν μέρος της κανονικής διαδικασίας DHT-ΔΤ σταθεροποίησης) και ανανεώνουν αυτόματα τα index tuples που αποθηκεύει.

Οι RG πρέπει να είναι (α) επαρκώς ισχυροί και διατεθειμένοι να αντέξουν τον επιπλέον φόρτο (αποθήκευσης, επεξεργασίας και επικοινωνίας) και (β) να είναι συνδεδεμένοι για μεγάλα χρονικά διαστήματα, ώστε να παρέχουν εγγυήσεις στον αριθμό βημάτων για ερωτήματα εύρους και να αποφεύγονται οι μεγάλες μεταφορές εξαιτίας των προσχωρήσεων/αποχωρήσεών τους. Συνεπώς, υπάρχει η ανάγκη ύπαρξης ενός μηχανισμού για την αναγνώριση και αξιοποίηση υποψήφιων RG με τρόπο αποδοτικό και διαφανή.⁵ Δεδομένης της λειτουργικότητας που παρέχεται από την αρχική μας (την χωρίς τους RangeGuards) υποδομή, οι RG αναγνωρίζονται και εντοπίζονται όπως περιγράφουμε στη συνέχεια.

⁵Υποθέτουμε ότι οι κόμβοι δεν θα συμπεριφερθούν εχθρικά (αφήνοντας τα μέτρα αντιμετώπισης τέτοιων συμπεριφορών σαν ενδεχόμενη μελλοντική εργασία), όπως ισχύει στις περισσότερες ερευνητικές εργασίες που είναι βασισμένες σε DHT.

3.4.1 Μετρητές Απόδοσης Κόμβων και η Σχέση Απόδοσης Κόμβων

Ο διαχειριστής του κάθε κόμβου επιλέγει αν θα ήθελε ο κόμβος του να είναι υποψήφιος για RangeGuard ή όχι (παρόμοια με ότι γίνεται σήμερα με τους super-peers στις περισσότερες αδόμητες εφαρμογές διαμοιρασμού δεδομένων σε ΔΟΕ). Ένας υποψήφιος κόμβος n , με ταυτότητα $n.id$, παρακολουθεί την ποσότητα $n.a$ αιτήσεων ανάκτησης ή/και δρομολόγησης που εξυπηρετεί. Αυτή η πληροφορία ανανεώνεται περιοδικά, κάθε $\mathcal{E}$ δευτερόλεπτα (περίοδος που επίσης ονομάζουμε εποχή). Κρατώνται λοιπόν, δυο μετρητές επίδοσης κόμβων - node performance counters (NPCs), οι $n.a_c$ και $n.a_p$ για τις αιτήσεις που εξυπηρετούνται κατά τη διάρκεια της παρούσας και της προηγούμενης εποχής αντίστοιχα.

Αυτή η πληροφορία είναι αποθηκευμένη στο σύστημα ως μια σχέση επίδοσης κόμβων (node performance relation) τεσσάρων στηλών $\mathcal{NPR} : \{n.id, n.a_p, \mathcal{U}, status\}$, με πρωτεύων κλειδί το $n.id$, και δεικτοδοτημένη με βάση το $n.a_p$. Η $status$ είναι μια διαδικτική μεταβλητή, που παίρνει την τιμή *true* αν ο κόμβος είναι μέλος του RangeGuard. Ο $\mathcal{U}$ είναι ένας μετρητής, που αυξάνεται με κάθε ανανέωση της tuple και μετατοπίζεται (shift) προς τα αριστερά (υποθέτοντας μια little-endian αρχιτεκτονική) (α) σε κάθε ανανέωση ή (β) κάθε $\mathcal{E} + \delta$ δευτερόλεπτα, με το χρονόμετρο να μηδενίζεται με κάθε ανανέωση. Η δ είναι μια ποσότητα που εξαρτάται από μετρήσιμα χαρακτηριστικά (π.χ. round-trip / ping time) της από σημείο σε σημείο σύνδεσης ανάμεσα στον κόμβο n και τον κόμβο που έχει αποθηκευμένο το index tuple με την μεταπληροφορία του n . Ο $\mathcal{U}$ συμβολίζει τον χρόνο για τον οποίο ένας κόμβος είναι συνδεδεμένος στο δίκτυο. Πιστεύουμε ότι ο χρόνος που ένας κόμβος είναι συνδεδεμένος στο δίκτυο και ο αριθμός των αιτήσεων που εξυπηρετήσε σε αυτό το χρόνο, είναι αρκετές ενδείξεις για την ισχύ και την καταλληλότητα ενός κόμβου για τον δακτύλιο RG . Πιο πολύπλοκες μετρικές θα μπορούσαν να χρησιμοποιηθούν, πάντα με βάση το σκεπτικό της αποθήκευσης των πλειάδων αυτών στο δαχτυλίδι του κάτω επιπέδου· η εύρεση τέτοιων μετρικών είναι ένα ανοιχτό ερώτημα και αντικείμενο τρέχουσας και μέλλουσας δουλειάς. Αν ένας κόμβος επιθυμεί να πάψει να είναι υποψήφιος RG , αρκεί να θέσει μια χαμηλή τιμή (π.χ. 0) για το a_p του και να σταματήσει να ανανεώνει αυτήν την πληροφορία (έτσι ώστε η $\mathcal{U}$ τιμή του να μειωθεί λόγω της απραξίας). Επίσης, να σημειώσουμε ότι η $\mathcal{NPR}$ πλειάδα του κόμβου n είναι αποθηκευμένη στον ίδιο τον κόμβο, καθώς το πρωτεύων κλειδί αυτής της σχέσης είναι το $n.id$.

Το Κόστος Συντήρησης της NPR Οι υποψήφιοι RG κόμβοι πρέπει να ανανεώνουν τα $\mathcal{NPR}$ index tuple τους - σημειώνουμε ότι οι $\mathcal{NPR}$ πλειάδες είναι επίσης δεικτοδοτειμένες με το $n.a_p$ πεδίο τους. Αυτή η διαδικασία απαιτεί 2 DHT-ΔΤ lookups δακτυλίων για κάθε εποχή $\mathcal{E}$: ένα lookup για την διαγραφή του index tuple της παλιάς τιμής του $n.a_p$ και ένα για την εισαγωγή του index tuple της νέας τιμής του $n.a_p$. Το συνολικό κόστος, μετρημένο σε βήματα, είναι $O(\log(N))$ (καθώς κάθε lookup χρειάζεται $O(\log(N))$ βήματα), ενώ η κατανάλωση σε εύρος ζώνης δικτύου είναι ελάχιστη δεδομένου του μικρού μεγέθους αυτών των index tuples. Εναλλακτικά, μπορούμε είτε να κρατήσουμε μια σύνδεση με τον κόμβο που αποθήκευσε τελευταίος το σχετικό index tuple και να αρχίσουμε την αναζήτηση από εκεί, είτε να ακολουθήσουμε μια soft-state προσέγγιση. Επιπλέον το συνολικό κόστος μπορεί να ρυθμιστεί μέσω της παραμέτρου $\mathcal{E}$, έτσι μπορούμε να ανταλλάξουμε την φρεσκάδα του $\mathcal{NPR}$ index για bandwidth και βήματα.

3.4.2 Είσοδος στο RangeGuard

Ένας κόμβος που θα ήθελε να προσχωρήσει στο RangeGuard χρησιμοποιεί τον RG του σαν τον ((bootstrap)) κόμβο του για τον RangeGuard δακτύλιο. Ο RG είναι υπεύθυνος για την ανάκτηση των μεταπληροφοριών του υποψήφιου κόμβου και για να ελέγξει ότι είναι αρκετά ισχυρός (δηλ. ότι έχει εξυπηρετήσει περισσότερες αιτήσεις από ένα προκαθορισμένο κάτω όριο) και ότι έχει μείνει συνδεδεμένος για αρκετό χρονικό διάστημα (βασισμένος στην αντίστοιχη τιμή του $\mathcal{U}$) ώστε

να του επιτραπεί η είσοδος στο RangeGuard. Αν πληρούνται όλες οι προϋποθέσεις⁶, εκτελείται το κανονικό DHT-ΔΤ πρωτόκολλο προσχώρησης και ο υποψήφιος κόμβος προάγεται στον δακτύλιο RG , διαφορετικά το πρωτόκολλο τερματίζει. Στη συνέχεια, ο κόμβος ανανεώνει το πεδίο *status* στην εγγραφή που του αντιστοιχεί στην $\mathcal{NPR}$ σχέση στον χαμηλότερου επιπέδου δακτύλιο, ώστε να αντικατοπτρίζεται η προαγωγή του σε κατάσταση RG και ενημερώνει για την ύπαρξή του τους κόμβους που βρίσκονται στο τόξο της υπευθυνότητάς του. Εναλλακτικά, αυτό το βήμα μπορεί να αφαιρεθεί ως μέρος της διαδικασίας σταθεροποίησης/συντήρησης του χαμηλότερου επιπέδου δακτυλίου. Το κόστος της προσχώρησης στον RangeGuard δακτύλιο αποτελείται από: (α) το κόστος της επικοινωνίας και αποστολής της σχετικής $\mathcal{NPR}$ πλειάδας στον RG που είναι υπεύθυνος για τον προσχωρούντα κόμβο, και (β) το κόστος του κανονικού πρωτοκόλλου προσχώρησης του DHT-ΔΤ δακτυλίου, για τον RG δακτύλιο. Συνεπώς, το κόστος προσχώρησης στον δακτύλιο RG , σε αριθμό αλμάτων, είναι $O(\log(M))$, ενώ η επιπλέον κατανάλωση σε bandwidth είναι ελάχιστη (δεδομένου του αναμενόμενου μικρού μεγέθους του RG δακτυλίου και του πολύ μικρού μεγέθους των $\mathcal{NPR}$ πλειάδων).

Υπάρχουν δυο τρόποι για να γίνει δεκτός ένας κόμβος στο RangeGuard: (α) να προαχθεί από έναν RG που επιθυμεί να μοιραστεί μέρος του φόρτου του, ή (β) να προθυμοποιηθεί να πάρει κάποια περιοχή στο χώρο διευθύνσεων όπου δεν υπάρχει κάποιος RG .

A. Προαγωγή Λόγω ανωμαλιών στα δεδομένα ή στην κατανομή προσπέλασης, ένας RangeGuard μπορεί να καταλήξει υπερφορτωμένος με εισερχόμενες αιτήσεις. Επιπλέον, είναι πιθανό για μια περιοχή στο RangeGuard να έχει λίγους κόμβους (π.χ. μπορούμε να σκεφτούμε τον RangeGuard δακτύλιο στη φάση της δημιουργίας του). Σε τέτοιες περιπτώσεις, ένα μέλος του RangeGuard μπορεί να ζητήσει την υποστήριξη κάποιων υποψηφίων για το RangeGuard προάγωντάς τους σε κατάσταση RG .

Με την υποδομή που περιγράψαμε προηγουμένως, όταν ένας RG θέλει να προάγει έναν κόμβο σε κατάσταση RangeGuard για μια περιοχή γύρω από ένα σημείο p - δηλ. η ταυτότητα του κόμβου βρίσκεται σε απόσταση το πολύ ϵ από ένα σημείο p στον χαμηλότερου επιπέδου δακτύλιο, ο μετρητής προσπέλασής του είναι μεγαλύτερος από a στην προηγούμενη εποχή και η τιμή $\mathcal{U}$ είναι μεγαλύτερη του u - απλώς εκτελεί το εξής ερώτημα εύρους:

select id **from** $\mathcal{NPR}$ **where** $\mathcal{U} > u$ **and** $\alpha_p > a$ **and** $0 \leq id - p < \epsilon$

Το σύνολο που θα προκύψει σαν αποτέλεσμα αυτού του ερωτήματος θα περιέχει τις ταυτότητες των υποψηφίων RG για την περιοχή του ενδιαφέροντος. Είναι στη συνέχεια δουλειά του RG από τον οποίο προήλθε το ερώτημα, να διαλέξει τον καλύτερο υποψήφιο, να τον ενημερώσει για την προαγωγή του και να ξεκινήσει το πρωτόκολλο προσχώρησης ώστε να τον εντάξει στον RG δακτύλιο.

B. Προθυμοποίηση Οι υποψήφιοι RG μπορούν να βρίσκονται σε οποιαδήποτε περιοχή στον χαμηλότερου επιπέδου δακτύλιο. Για λόγους ανωμαλιών, είτε των δεδομένων, είτε της κατανομής προσπέλασης, οι οποίοι είναι παρόμοιοι με αυτούς που προκαλούν τους RangeGuards να ζητήσουν υποστήριξη, είναι πιθανόν για κάποιες περιοχές να έχουμε χαμηλό φορτίο δεδομένων/προσπέλασης σε βαθμό που το RangeGuard που είναι υπεύθυνο για αυτές δεν χρειάστηκε ποτέ υποστήριξη. Αυτό θα μπορούσε να έχει ως αποτέλεσμα μεγάλα τόξα/περιοχές του δακτυλίου χαμηλότερου επιπέδου να αντιστοιχισθούν σε έναν μόνο RangeGuard κόμβο, που να απέχει αρκετά βήματα, στο δακτύλιο χαμηλότερου επιπέδου, από τους πρώτους κόμβους αυτού του τόξου. Αν και αυτό δεν είναι πρόβλημα όταν το σύστημα είναι σε σταθερή κατάσταση, μπορεί όμως να αυξήσει τον χρόνο που χρειάζεται ένας κόμβος αυτού του τόξου για να βρει έναν καινούριο RG , αν ο παρών RG φύγει από το σύστημα αφύσικα.

⁶Τα κάτω όρια κατά πάσα πιθανότητα διαφέρουν ανάλογα με τη σημασιολογία που απαιτείται από το RangeGuard. Ο ακριβής υπολογισμός θεωρητικών κάτω ορίων είναι ένα ορθογώνιο πρόβλημα που το αφήνουμε για μελλοντική εργασία.

Ετσι, επιτρέπουμε στους υποψήφιους RG κόμβους του δακτυλίου χαμηλότερου επιπέδου να προθυμοποιηθούν για μια RG θέση. Αν ένας υποψήφιος RG διαπιστώσει την ύπαρξη μιας κατάστασης σαν αυτής που περιγράψαμε προηγουμένως (δηλ. να υπάρχει μεγάλη απόσταση ανάμεσα στον ίδιο και στον RG του), μπορεί να επικοινωνήσει με τον τελευταίο και να ζητήσει να προαχθεί σε κατάσταση RG . Ο RG είναι υπεύθυνος να κάνει τους απαραίτητους α_p και $\mathcal{U}$ ελέγχους και να δεχθεί ή όχι τον υποψήφιο στο RangeGuard.

3.4.3 Έξοδος από το RangeGuard

Παρομοίως, ένας RG μπορεί να αποφασίσει αν φύγει από το RangeGuard αν βρεθεί σε μια κατάσταση όπου το τόξο της υπευθυνότητάς του γίνει πολύ μικρό (λόγω του ότι υπάρχουν κοντά του υποψήφιοι RG που προάγονται σε RG κατάσταση), ή ο φόρτος που έχει να αντιμετωπίσει σαν RG γίνεται μικρότερος από ένα προκαθορισμένο κάτω όριο (π.χ. μια εκτίμηση του φορτίου που πρέπει να έχει βασιζόμενη σε ομοιόμορφο φόρτο δεδομένων/προσπέλασης). Ένας RG που επιθυμεί να φύγει από τον RangeGuard δακτύλιο, ακολουθεί τα εξής βήματα: (1) ακολουθεί το DHT-ΔΤ πρωτόκολλο αποχώρησης, μεταφέροντας τα RG δεδομένα του στον κατάλληλο κόμβο ή κόμβους στον δακτύλιο RG , (2) ανανεώνει το πεδίο *status* στην εγγραφή του στην $\mathcal{NPR}$ σχέση στον δακτύλιο χαμηλότερου επιπέδου, ώστε να δηλώσει ότι δεν είναι πλέον RangeGuard και (3) προαιρετικά, ενημερώνει τους κόμβους που είναι συνδεδεμένοι σε αυτόν στον RG δακτύλιο ώστε να ανανεώσουν τις συνδέσεις τους, ή διαφορετικά φεύγει και αυτό το αφήνει να γίνει σαν μέρος της διαδικασίας σταθεροποίησης/συντήρησης του δακτυλίου RG .

Επισημαίνουμε ότι η προσέγγιση μας χρησιμοποιεί κανονικές διαδικασίες DHT-ΔΤ δακτυλίου για τη συγκρότηση και τη διατήρηση του RangeGuard δακτυλίου. Με εξαίρεση το δεύτερο βήμα, η διαδικασία που περιγράψαμε προηγουμένως είναι το κανονικό DHT-ΔΤ πρωτόκολλο αποχώρησης. Επιπλέον σημειώνουμε ότι ο RG που φεύγει δεν χρειάζεται να ενημερώσει τους κόμβους στο τόξο υπευθυνότητάς του για τον καινούριο τους RG , καθώς αυτό θα επιτευχθεί κατά τη διάρκεια της διαδικασίας σταθεροποίησης του του δακτυλίου χαμηλότερου επιπέδου. Συνεπώς το κόστος αποχώρησης ενός κόμβου από το RangeGuard είναι ίσο με το κόστος εκτέλεσης του κανονικού DHT-ΔΤ πρωτοκόλλου αποχώρησης για τον RG δακτύλιο, ενώ η μεταφορά των δεδομένων είναι ελάχιστη εξαιτίας του πολύ μικρού μεγέθους των $\mathcal{NPR}$ πλειάδων και του μεγέθους του RG δακτυλίου. Από την άλλη πλευρά, υπάρχει η απαίτηση για αρκετούς RangeGuard κόμβους με αυξημένες ικανότητες. Αυτό δεν είναι μη ρεαλιστικό, καθώς πολλοί κόμβοι σε πραγματικές εφαρμογές έχουν αποδειχθεί να είναι περισσότερο ισχυροί. Με την έννοια και την αξιοποίηση των RangeGuards μπορούμε να εκμεταλλευτούμε αυτήν την ετερογένεια σε ισχύ για να επιτύχουμε μεγαλύτερη αποτελεσματικότητα στην επεξεργασία ερωτημάτων εύρους.

3.4.4 Επεξεργασία Ερωτημάτων Εύρους Χρησιμοποιώντας RGs

Με τους RangeGuards στο προσκήνιο, ένα ερώτημα $(a_i.v_{low}, a_i.v_{high})$ στο χαρακτηριστικό a_i θα σταλεί από τον αιτούντα κόμβο απευθείας (1 βήμα) στον RG που είναι υπεύθυνος για τα δεδομένα του αιτούντα κόμβου. Μετά από αυτό το σημείο οι RangeGuards αναλαμβάνουν να συγκεντρώσουν την ζητούμενη πληροφορία, χρησιμοποιώντας τον DHT-ΔΤ αλγόριθμο που περιγράψαμε νωρίτερα, με τη διαφορά ότι τώρα όλες οι λειτουργίες πραγματοποιούνται στον RangeGuard δακτύλιο (Σχήμα 3.5). Με την τοποθέτηση των δεδομένων στο χαμηλότερο δακτύλιο να αντικατοπτρίζεται στο RangeGuard δακτύλιο, οι ζητούμενες index tuples θα βρίσκονται ανάμεσα στον RG_l υπεύθυνο για το $a_i.v_{low}$ και τον RG_h υπεύθυνο για το $a_i.v_{high}$. Αυτός ο αλγόριθμος χρειάζεται 1 βήμα δρομολόγησης για να προσεγγιστεί ο RangeGuard δακτύλιος, άλλα $O(\log(M))$ βήματα στον RG δακτύλιο και τόσα βήματα δρομολόγησης όσοι RangeGuards υπάρχουν ανάμεσα στους RG_l και RG_h . Επιπλέον ο όρος $O(\log(M))$ μπορεί να βελτιωθεί σε $O(1)$, χρησιμοποιώντας τεχνικές

Σχήμα 3.5: Επεξεργασία Ερωτημάτων Εύρους με και χωρίς τους Κόμβους RG

παρόμοιες με αυτές που παρουσιάζονται στο [53] ή στο [95].

Καθώς κατά πάσα πιθανότητα θα υπάρχουν πολύ λιγότεροι *RGs* στο σύστημα από ότι υπάρχουν κόμβοι, και οι RangeGuards είναι πιο ισχυροί (από άποψη υπολογιστικών δυνατοτήτων, bandwidth δικτύου και χρόνου σύνδεσης στο δίκτυο) από ότι ο μέσος κόμβος του συστήματος, αυτή η αρχιτεκτονική είναι σημαντικά αποδοτικότερη από αυτή που παρουσιάστηκε νωρίτερα. Συγκεκριμένα, για $5\% \times N$ *RGs*, παρόλο που η χειρίστη περίπτωση αριθμού αλμάτων παραμένει $O(N)$, τώρα έχει μια αρκετά σημαντικά χαμηλότερη σταθερά (δηλ. ένα 5% - ή 20 φορές - χαμηλότερο αριθμό αλμάτων). Σημειώνουμε ότι απαιτούμε μόνο ένα $5\% \times N$ των κόμβων να είναι ισχυροί και αλτρουιστές στο σύστημά μας· όπως σχετικές εργασίες [5, 114] έχουν δείξει, μπορούμε να περιμένουμε ότι κατά μέσο όρο ένα 5% του πλήθους των κόμβων σε μεγάλης κλίμακας ΔΟΕ διαμοιρασμού δεδομένων, όπως τα Gnutella και Kazaa, είναι επαρκώς ισχυροί και αλτρουιστικοί κόμβοι. Συνεπώς με την εκμετάλλευση της πλήρους ισχύος όλων αυτών των κόμβων, μπορούμε να επιτύχουμε υψηλότερα κέρδη σε απόδοση απ'ότι περιγράψουμε πιο πάνω.

3.4.5 Μετατροπές στο Δίκτυο DHT-ΔΤ

Μέχρι στιγμής, έχουμε επεκτείνει το υποκείμενο DHT-ΔΤ σύστημα στα εξής πεδία. Πρώτα δώσαμε πρωτόκολλα που επιτρέπουν την προσχώρηση/αποχώρηση των κόμβων στον RangeGuard δακτύλιο, ενώ παρέχουμε εγγυήσεις για την ορθή λειτουργία του συστήματος συνολικά, όπως επίσης και μια μέθοδο για την ανακάλυψη και χρήση των υποψηφίων *RGs* (που περιγράφηκαν με λεπτομέρεια στις Ενότητες 3.4.2 και 3.4.3. Επίσης μετατρέψαμε το πρωτόκολλο επεξεργασίας ερωτημάτων, ώστε να χρησιμοποιήσουμε και να εκμεταλλευτούμε την επιπλέον λειτουργικότητα που προσφέρεται από το RangeGuard (που περιγράφηκε στην Ενότητα 3.4.4). Όσον αφορά την κατάσταση δρομολόγησης, έχουμε προσθέσει ένα ακόμα πεδίο σε κάθε κόμβο ώστε να δείχνει στον υπεύθυνο του *RG*. Για λόγους ανεκτικότητας σε λάθη και ταχύτερης ανάκαμψης από *RGs* που «πέφτουν» ή φεύγουν, μπορούμε να διατηρούμε συνδέσεις με τους επόμενους k *RGs*. Σημειώνουμε, ότι το μέγεθος της κατάστασης δρομολόγησης παραμένει $O(\log N)$.

Με την επιπλέον πληροφορία στους πίνακες δρομολόγησης των κόμβων, πρέπει επίσης να ρυθμίσουμε την διαδικασία σταθεροποίησης ώστε να περιλαμβάνει την εγγραφή *RG* στο σύνολο των συνδέσεων που πρέπει να δοκιμαστούν. Όπως και στην κανονική διαδικασία σταθεροποίησης, ένας κόμβος θέτει ένα ερώτημα για το ID του στον *RG* δακτύλιο. Η σχετική πληροφορία είναι από σχεδιασμού αποθηκευμένη στον υπεύθυνο *RG*· έτσι, η απάντηση σε αυτό το ερώτημα θα ξεκινήσει από τον *RG* που είναι υπεύθυνος για το τόξο στο οποίο βρίσκεται ο αιτών κόμβος. Αν ο υπεύθυνος *RG* έχει αλλάξει (π.χ. λόγω του ότι περισσότεροι υποψήφιοι *RGs* προσχώρησαν στο RangeGuard), ο κόμβος θα πάρει την απάντηση από ένα διαφορετικό *RG* και έτσι θα ανανεώσει την *RG* σύνδεσή του. Τέλος, αν και οι k *RG* σύνδεσμοι ενός κόμβου αποσυνδεθούν από το σύστημα συγχρόνως, τότε ο κόμβος μπορεί να θέσει ερώτημα στον χαμηλότερου επιπέδου δακτύλιο για έναν κοντινό του

RangeGuard (δηλ. έναν κόμβο του οποίου το πεδίο status έχει την τιμή *true*).

3.4.6 Κατανομή Φόρτου

Για να προσομοιώσουμε την 95%-5% παρατήρηση του [5] (δηλ. το ότι 5% όλων των κόμβων εξυπηρετούν το 95% όλων των αιτήσεων του συστήματος), βάζουμε τους κόμβους στον DHT-ΔΤ δακτύλιο να στρίψουν ένα μη δίκαιο κέρμα και να αποστείλουν τα ερωτήματα στον *RG* δακτύλιο με πιθανότητα 0.95, ενώ τα επεξεργάζονται στον DHT-ΔΤ δακτύλιο με πιθανότητα 0.05. Εκτός από την σχετική μέριμνα από το DHT-ΔΤ, ο φόρτος μοιράζεται περαιτέρω στον *RG* δακτύλιο και από τα πρωτόκολλα προσχώρησης/αποχώρησης.

Όσον αφορά την προσχώρηση/αποχώρηση, υπενθυμίζουμε ότι οι *RGs* μπορούν να ζητήσουν την υποστήριξη από τους υποψήφιους *RG* κόμβους όταν είναι υπερφορτωμένοι και μπορούν να αποφασίσουν να φύγουν από τον *RG* δακτύλιο αν ο φόρτος τους είναι πολύ χαμηλός ή το τόξο της υπευθυνότητας του είναι πολύ μικρό. Στην πρώτη περίπτωση, ο *RG* που καλεί για βοήθεια μπορεί να διαλέξει ανάμεσα σε διάφορους υποψήφιους *RGs*, που επιστρέφουν σαν αποτέλεσμα του σχετικού ερωτήματος. Καθώς αυτός ο κόμβος γνωρίζει ποιό μέρος του τόξου της υπευθυνότητάς του προκαλεί τον μεγαλύτερο φόρτο, μπορεί να διαλέξει τον κατάλληλο υποψήφιο *RG* για να μοιραστεί αυτόν τον φόρτο. Όπως ήδη αναφέραμε, αναμένουμε ότι οι υποψήφιοι RangeGuards είναι ομοιόμορφα κατανεμημένοι στον δακτύλιο χαμηλότερου επιπέδου. Συνεπώς οι RangeGuards που ζητούν βοήθεια θα έχουν καλή πιθανότητα να βρουν ένα υποψήφιο RangeGuard στο τόξο του ενδιαφέροντός τους.

Όσον αφορά την τοποθέτηση των δεδομένων, αν η δημοτικότητα μιας τιμής δεν εξαρτάται από την θέση της στο πεδίο τιμών των attributes (π.χ. μία τιμή v δεν είναι η πιο δημοφιλής για όλα τα attributes στο σύστημα) τότε το να έχουμε πολλαπλά attributes αντιστοιχισμένα στον ίδιο δακτύλιο μεταφράζεται στο να έχουμε πολλαπλά δημοφιλή αντικείμενα διασκορπισμένα σε όλους τους κόμβους του συστήματος. Σε αντίθετη περίπτωση, μια τυχαία αλλά εύκολα υπολογίσιμη offset τιμή (π.χ. η cryptographic hash του ονόματος του attribute) μπορεί να προστεθεί (mod το μέγιστο ID κειμένου) σε όλες τις τιμές στο πεδίο τιμών του attribute. Αυτό μας παρέχει αρκετή τυχαιότητα στην τοποθέτηση των δεδομένων για να εγγυηθούμε παρόμοια αποτελέσματα, όπως θα δούμε στην Ενότητα 5.1.2. Σημειώνουμε ότι για ένα N -κόμβων σύστημα και την χειρίστης κλήσης κατανομή (δηλ. μια τιμή επιλέγεται με πιθανότητα 1 και οι υπόλοιπες με πιθανότητα 0), απαιτούνται N attributes για να έχουμε ισορροπημένο φόρτο, υποθέτοντας την βέλτιστη περίπτωση κατανομής φόρτου βασιζόμενοι μόνο στα παραπάνω δεδομένα. Ωστόσο, αφού ο *RG* δακτύλιος είναι πολύ μικρότερος από τον δακτύλιο DHT-ΔΤ, ο απαιτούμενος αριθμός από attributes είναι πολύ μικρότερος (δηλ. $M \ll N$ attributes για έναν M -κόμβων *RG* δακτύλιο).

Συνάμα, εξαιτίας αυτής της διαφοράς στα μεγέθη των δακτυλίων *RG* και DHT-ΔΤ, κάθε *RG* κόμβος είναι υπεύθυνος για τιμές που είναι ανατεθειμένες σε πολλαπλούς κόμβους στον DHT-ΔΤ δακτύλιο, κάτι που οδηγεί σε μια ακόμα πιο ομοιόμορφη κατανομή του φόρτου στον *RG* δακτύλιο. Επιπλέον μπορούμε εύκολα να εφαρμόσουμε τεχνικές ισορρόπησης φόρτου για τον DHT-ΔΤ δακτύλιο, ώστε να μοιράσουμε ακόμα καλύτερα τον φόρτο τόσο στους χαμηλότερου επιπέδου όσο και στους *RG* κόμβους (π.χ. με ιδεατούς κόμβους [108], ή μεταφορά κόμβων με βάση το φόρτο τους [71], όταν χρησιμοποιούμε OP-Chord [125]· με το να βασιζόμαστε στην εξισορρόπηση φόρτου του ίδιου του δικτύου επικάλυψης, όταν χρησιμοποιούμε SkipNets [56] ή Skip Graphs [8], κ.τ.λ.).

Κεφάλαιο 4

Κατανεμημένες Συνόψεις Κατακερματισμού

Την τελευταία δεκαετία είμαστε μάρτυρες μίας πληθώρας εφαρμογών διαχείρισης δεδομένων παγκόσμιας κλίμακας οι οποίες περιλαμβάνουν έναν μεγάλο αριθμό από κόμβους απλωμένους σε όλο το Διαδίκτυο. Κυριότερα παραδείγματα τέτοιων εφαρμογών είναι:

- *Εφαρμογές διαμοιρασμού αρχείων* σε δίκτυα ομοτίμων εταιρών, όπως το Gnutella και το BitTorrent [120]. Τα δίκτυα αυτά έχουν εκατομμύρια χρήστες (κόμβους) οι οποίοι διαθέτουν αποθηκευτικό χώρο και δικτυακούς πόρους για την αναζήτηση και την ανάκτηση αρχείων και παρουσιάζουν έντονη δυναμικότητα στον πληθυσμό και την κατάσταση του δικτύου, με χρήστες να συνδέονται και να αποσυνδέονται με γρήγορους ρυθμούς.
- *Εφαρμογές διαμοιρασμού επιστημονικών δεδομένων σε πλέγμα* [47] όπως αστρονομικές παρατηρήσεις ή μοντέλα και πειράματα σε βιοχημικά δίκτυα στις επιστήμες της ζωής. Στις περιπτώσεις αυτές, τα δεδομένα τυπικά απαιτούν μία σαφώς υψηλότερης στάθμης συνέπεια, σε σύγκριση με τις υπόλοιπες εφαρμογές ΔΟΕ, καθώς και υποστήριξη για προχωρημένα και πολύπλοκα ερωτήματα.
- *Δίκτυα αισθητήρων* διαφόρων ειδών [36], τα οποία για παράδειγμα μετρούν περιβαλλοντικές συνθήκες ή παρατηρούν την κίνηση ενός δικτύου (οδικού ή υπολογιστικού). Οι εφαρμογές αυτές απαιτούν υποστήριξη για συναθροιστικά ερωτήματα πάνω από υβριδικές υποδομές συμπεριλαμβανομένων ασύρματων δικτύων και δικτύων βασισμένων στο Διαδίκτυο.
- Η κατανεμημένη ανάλυση δεδομένων επί δομημένων εγγραφών, ως τμήμα εφαρμογών συνένωσης δεδομένων από σχεσιακές, XML ή RDF πηγές [75, 122]. Μία μεγάλη πρόκληση σε αυτή την κατεύθυνση θα μπορούσε να είναι η ανάλυση σε πραγματικό χρόνο δεδομένων κίνησης στο Διαδίκτυο με σκοπό την καταπολέμηση (και στην ιδανική περίπτωση, την πρόληψη) επιθέσεων, ανεπιθύμητης αλληλογραφίας και άλλων ανωμαλιών [57]. Αυτή η κλάση εφαρμογών έρχεται με ένα πλούσιο ρεπερτόριο τύπων ερωτημάτων, συμπεριλαμβανομένων σχεσιακών τελεστών όπως η συνένωση (join) και η ομαδοποίηση (group-by) [44].

Όλες αυτές οι περιοχές εφαρμογών Διαδικτυακής κλίμακας έχουν την ανάγκη κατανεμημένων συναθροιστικών ερωτημάτων επί ενός μεγάλου αριθμού κόμβων δικτύου. Σε πολλές περιπτώσεις, τα συναθροιστικά ερωτήματα αυτά έχουν μία στατιστική φύση και άρα μπορούν ορισμένες φορές να εκτιμηθούν με ικανοποιητική ακρίβεια και χωρίς τον υπολογισμό του πλήρους, ακριβούς αποτελέσματος: το παράδειγμα αυτό είναι γνωστό ως *προσεγγιστική επεξεργασία ερωτημάτων*

(approximate query processing (AQP)) [4, 18, 58, 68]. Για παράδειγμα, σε εφαρμογές διαμοιρασμού αρχείων ΔΟΕ, μπορεί κανείς να ενδιαφέρεται για την εκτίμηση του συνολικού αριθμού διακριτών αρχείων που είναι διαθέσιμα στο δίκτυο σε οποιαδήποτε χρονική στιγμή – μία συναθροιστική ερώτηση τύπου *Count-Distinct*. Στις ηλεκτρονικές επιστήμες, ο συνολικός αριθμός φασματογραφιών ακτίνων X για τριπλά συστήματα αστερών διαθέσιμων οπουδήποτε σε ένα δικτυακό πλέγμα θα μπορούσε να είναι μία ενδιαφέρουσα μετρική, η οποία θα απαιτούσε μία κατανεμημένη λειτουργία καταμέτρησης τύπου *Count*. Για την ανάλυση καταγραφών Διαδικτυακής κίνησης, ο συνολικός όγκος δεδομένων που μεταδόθηκαν από ή προς πελάτες σε ένα συγκεκριμένο διάστημα διευθύνσεων IP μπορεί να υπολογιστεί από ένα κατανεμημένο ερώτημα τύπου *Sum*. Τέλος, τα πολύπλοκα και προχωρημένα ερωτήματα τύπου *join-and-group* επί δομημένων δεδομένων τυπικά απαιτούν την επιλογή του πλάνου εκτέλεσης ελάχιστου κόστους· αυτού του τύπου η βελτιστοποίηση ερωτημάτων με τη σειρά της βασίζεται σε επαρκώς ακριβείς εκτιμήσεις για την επιλεκτικότητα των περιορισμών του ερωτήματος (δηλ. του πληθάριθμου των ενδιάμεσων αποτελεσμάτων). Αν τα ίδια τα δεδομένα είναι ευρέως κατανεμημένα, αντιμετωπίζουμε το πρόβλημα του υπολογισμού *ιστογραμμάτων* και άλλων στατιστικών συνόψεων επί ενός μεγάλου αριθμού κόμβων δικτύου, καθένας από τους οποίους αποθηκεύει μόνο κάποια τμήματα του συνόλου, πιθανώς επικαλυπτόμενα με αυτά των υπόλοιπων κόμβων.

Σε αυτή τη νέα εποχή δικτύων δεδομένων ομοτίμων εταίρων διαδικτυακής κλίμακας, η ανάγκη για έναν κατανεμημένο μηχανισμό καταμέτρησης ανακύπτει σε πολλές περιπτώσεις. Συνήθως, η ποσότητα που θέλουμε να μετρήσουμε περιλαμβάνει και διπλότυπα και ο υποψήφιος αλγόριθμος πρέπει να προσφέρει ιδιότητες μη καταμέτρησής τους. Επιπλέον, η ευρείας κλίμακας κατανομή των δεδομένων και των λειτουργιών απαιτεί αντίστοιχα πλήρως αποκεντριοποιημένες μεθόδους εκτέλεσης παραδοσιακά κεντροποιημένων λειτουργιών ενώ υπονοεί και την παντελή έλλειψη γνώσης καθολικών/συνολικών ιδιοτήτων του συστήματος. Συνεπώς, ο υπολογισμός τέτοιου είδους μετρικών (π.χ. αριθμός των εγγράφων στο δίκτυο, μεγέθη σχέσεων σε μία βάση δεδομένων, κατανομή των τιμών των εγγράφων) σε συστήματα ΔΟΕ, με τρόπο κλιμακώσιμο, αποδοτικό και ακριβή, έχει αγνοηθεί συστηματικά.

Παράλληλα, με την εργασία μας αυτή προσπαθούμε να μεταφέρουμε στατιστικά εργαλεία τα οποία χρησιμοποιούνται σε κεντροποιημένα συστήματα διαχείρισης βάσεων δεδομένων, στον ευρέως κατανεμημένο κόσμο των συστημάτων διαχείρισης δεδομένων ΔΟΕ. Τα στοιχεία που καθιστούν τον ερευνητικό αυτό τομέα τόσο διαφορετικό σε σχέση με τις παραδοσιακές κατανεμημένες, αυτόνομες ή/και *federated* βάσεις δεδομένων, είναι η έλλειψη υψηλής ποιότητας διαχειριστών στους κόμβους του συστήματος και η ιδιαίτερα ασταθής φύση των ΔΟΕ υπό πολύ δυναμική συμπεριφορά των κόμβων του. Στο περιβάλλον των ΔΟΕ, οι πλειάδες δεδομένων διασκορπίζονται σε όλους τους κόμβους του δικτύου, συνήθως με τέτοιο τρόπο που να κατανέμει τον αποθηκευτικό φόρτο (και ίσως και τον φόρτο επεξεργασίας) δίκαια ανάμεσα στους συμμετέχοντες. Μπορεί κανείς εύκολα να καταλάβει ότι, σε ένα περιβάλλον αυτής της κλίμακας και αυτής της δυναμικής, ο υπολογισμός και η διατήρηση καθολικών στατιστικών υπό εγγυήσεις ακρίβειας και κόστους και όλα αυτά με σεβασμό προς την αυτονομία των κόμβων, είναι ένα πραγματικά πού δύσκολο έργο.

Σε κεντροποιημένα περιβάλλοντα, η διαχείριση στατιστικών για την επεξεργασία συναθροιστικών ερωτημάτων, την εκτίμηση της επιλεκτικότητας και την προσεγγιστική επεξεργασία ερωτημάτων, έχει φτάσει σε ένα επαρκώς ώριμο στάδιο. Για κατανεμημένα δεδομένα, ωστόσο, κατανοούμε σαφώς λιγότερο τα ζητήματα που ανακύπτουν· και για ευρέως κατανεμημένα, Διαδικτυακής κλίμακας περιβάλλοντα, η δημιουργία και η διατήρηση κατανεμημένων στατιστικών συνόψεων και ο υπολογισμός ακριβών εκτιμήσεων για τα παραπάνω είδη ερωτημάτων, θέτει ακόμα μεγάλες προκλήσεις. Στο τμήμα αυτό απαντούμε στις προκλήσεις αυτές, προσφέροντας λύσεις για κάποια από τα ζητήματα αυτά.

Το σχέδιο του συστήματος στο οποίο αποσκοπούμε φαίνεται στο σχήμα 4.1. Υποθέτουμε ότι λειτουργούμε πάνω από ένα δομημένο δίκτυο επικάλυψης, όπως είτε τα παραδοσιακά δίκτυα κατα-

Σχήμα 4.1: Δομικά Στοιχεία του Συστήματος Στόχου. Όλοι οι αλγόριθμοι που παρουσιάζονται εδώ, χτίζονται πάνω από μία κατανεμημένη σύνοψη, η οποία λειτουργεί πάνω από ένα σύστημα DHT

νεμημένων πινάκων κατακερματισμού είτε τα νεώτερα δίκτυα DHT διατήρησης τοπικότητας. Τα δεδομένα που αποθηκεύονται στο ΔΟΕ διαμοιράζονται με χρήση συναρτήσεων κατακερματισμού (και πιθανότητα αντιγράφονται) ανάμεσα στους κόμβους του δικτύου επικάλυψης. Αρχικά εισαγάγουμε τις Κατανεμημένες Συνόψεις Κατακερματισμού (DHS) [97, 99]: έναν κατανεμημένο μηχανισμό καταμέτρησης βασισμένο στις συνόψεις κατακερματισμού [33, 35]. Η προτεινόμενη λύση υποστηρίζει αποδοτική καταμέτρηση σε πολλές διαστάσεις με ισορροπημένο φόρτο, τόσο καταμετρώντας διπλότυπα όσο και χωρίς αυτά. Χρησιμοποιούμε τις DHS ως δομικό στοιχείο των λύσεών μας, ενώ παράλληλα παρουσιάζουμε και συγκρίνουμε εναλλακτικούς σχεδιασμούς και υλοποιήσεις για την δημιουργία κατανεμημένων συνόψεων.

Εν συνεχεία, εκμεταλλευόμαστε βασικές στατιστικές δομές (όπως οι συνόψεις κατακερματισμού και το DHS) για να υπολογίσουμε εκτιμήσεις υψηλής ακρίβειας για τα περισσότερα γνωστά συναθροιστικά ερωτήματα, σε δίκτυα διαχείρισης δεδομένων ΔΟΕ. Κατόπιν θα βασιστούμε πάνω στις δομές αυτές ως συστατικά για να παράγουμε και να χρησιμοποιήσουμε, με τρόπο αποδοτικό και κλιμακώσιμο, υψηλότερου επιπέδου στατιστικές δομές, όπως διαφόρων τύπων ιστογράμματα, με σκοπό την βελτιστοποίηση ερωτημάτων. Η ελπίδα μας είναι πως, καταφέροντας τα παραπάνω, θα καταστεί δυνατή η μεταφορά μέρους του μεγάλου πλούτου ερευνητικών αποτελεσμάτων σχετικών με την επεξεργασία και βελτιστοποίηση ερωτημάτων σε κεντροποιημένα και παραδοσιακά κατανεμημένα συστήματα διαχείρισης βάσεων δεδομένων, στον χώρο των ΔΟΕ.

4.1 Προσεγγίσεις Κατανεμημένων Συνόψεων

Ας υποθέσουμε ότι επιθυμούμε να υπολογίσουμε μία σύνοψη κατακερματισμού για ένα πολυσύνολο A κατανεμημένο κατά μήκος ενός δικτύου ομοτίμων. Διακρίνουμε δύο κύριες κατευθύνσεις στην αντιμετώπιση αυτού του προβλήματος : (α) τη «συντηρητική», απλοϊκή αλλά δημοφιλή

προσέγγιση με «ραντεβού» και (β) τον απόλυτα αποκεντριοποιημένο και πλήρως κλιμακώσιμο τρόπο των DHS.

4.1.1 Η Προσέγγιση Ραντεβού

Η προσέγγιση με ραντεβού είναι αυτό που κανείς θα αποκαλούσε τη φυσική εξέλιξη των client-server αρχιτεκτονικών στον κατανεμημένο κόσμο των δικτύων ομοτίμων. Κόμβοι που αποθηκεύουν αντικείμενα που ανήκουν σε ένα σύνολο A αρχικά υπολογίζουν ένα αναγνωριστικό ραντεβού (για παράδειγμα, εισάγοντας το «A» στην συνάρτηση κατακερματισμού του δικτύου επικάλυψης). Στη συνέχεια, υπολογίζουν τοπικά την επιλεγμένη σύνοψη και στέλνουν το αποτέλεσμα στον κόμβο του οποίου το αναγνωριστικό είναι πλησιέστερο στο παραπάνω αναγνωριστικό (αποκαλούμενο «κόμβο ραντεβού»). Ο κόμβος ραντεβού είναι υπεύθυνος για το συνδυασμό των επιμέρους συνόψεων (με bitwise OR) σε μια καθολική σύνοψη για το A . Οι ενδιαφερόμενοι κόμβοι μπορούν στη συνέχεια να ανακτήσουν την καθολική σύνοψη για το A ερωτώντας τον κόμβο ραντεβού. Όπως είναι προφανές, το κόστος αριθμού βημάτων για την εισαγωγή ενός αντικειμένου από ένα κόμβο σε αυτή τη συνολική σύνοψη, καθώς και το κόστος για να ανακτήσει τη συνολική σύνοψη είναι $O(\log N)$.

Αν και ξεκάθαρη, απλή και ιδιαίτερα αποδοτική από πλευράς αριθμού βημάτων, αυτή η λύση υποφέρει από δύο κύρια προβλήματα κλιμάκωσης. Πρώτον, ο κόμβος ραντεβού επιβαρύνεται με δυσανάλογα υψηλότερο κόστος επικοινωνίας από τους υπόλοιπους κόμβους. Στη χειρότερη περίπτωση που όλοι οι κόμβοι (ή ένα μεγάλο τμήμα τους) στο σύστημα αποθηκεύσουν αντικείμενα από το A – ή ένα μεγάλο τμήμα από κόμβους επιθυμήσουν να ανακτήσουν την καθολική σύνοψη – τότε ο κόμβος ραντεβού θα πρέπει να εξυπηρετήσει $O(N)$ εισερχόμενες συνδέσεις και την αντίστοιχη επιβάρυνση σε εύρος ζώνης.

Η εύκολη διέξοδος είναι να χρησιμοποιήσουμε πολλαπλούς κόμβους ραντεβού ανά κατανεμημένη σύνοψη ώστε να κατανείμουμε το φορτίο μεταξύ τους. Αυτό μπορεί να συμβεί με δύο τρόπους: είτε (α) οι κόμβοι κατανείμουν τα αντικείμενά τους κατά μήκος των κόμβων ραντεβού κατά το χρόνο εισαγωγής και συντήρησης, αλλά πληρώνουν το επιπλέον κόστος σε αριθμό βημάτων του να υποχρεούνται να επισκεφθούν κάθε ένα τους για να ανακατασκευάσουν τη συνολική σύνοψη ή (β) ενημερώνουν πάντα όλους τους κόμβους ραντεβού κατά την εισαγωγή (proactive replication), συνεπώς πληρώνοντας αυτό το επιπλέον κόστος κατά τις εισαγωγές, αλλά απαιτείται να επισκεφθούν μόνο τους κόμβους ραντεβού μια μοναδική φορά για να ανακτήσουν την καθολική σύνοψη. Δεύτερον, στα πλαίσια της παραπάνω παρατήρησης, προκειμένου να υπολογιστεί η καθολική σύνοψη για πολλαπλά σύνολα, θα πρέπει κανείς να επικοινωνήσει με τόσους κόμβους ραντεβού όσα και τα σύνολα. Επομένως, το συνολικό κόστος σε μηνύματα για τον υπολογισμό πολλαπλών συνόψεων αυξάνεται γραμμικά με τον αριθμό των συνόψεων. Παρόλα αυτά, για λόγους πληρότητας, δε θα εγκαταλείψουμε αυτή την προσέγγιση, καθώς φαίνεται να είναι αρκετά δημοφιλής στη σχετική βιβλιογραφία.

Ας υποθέσουμε πως η σύνοψη κατακερματισμού μας αποτελείται από ένα μοναδικό λ -bit διάνυσμα $B[\cdot]$. Μια αφελής υλοποίηση βασισμένη σε DHT θα ανέθετε καθεμία από τις λ αυτές θέσεις σε ένα κόμβο στο δίκτυο (η όπως λέγεται προσέγγιση με ραντεβού, η οποία αναφέρεται επίσης και στο [35]) και θα χρησιμοποιούσε αυτούς τους κόμβους για να αποθηκεύσει τις τιμές των bits με κατανεμημένο τρόπο (σχήμα 4.2). Παρόλα αυτά, αυτή η σχεδίαση έχει πολλά σοβαρά ελαττώματα: (α) μόνο λ από τους N κόμβους (με $\lambda \ll N$) στο δίκτυο επιβαρύνονται με την εργασία της συντήρησης των τιμών των θέσεων των διανυσμάτων, οδηγώντας σε σοβαρή ανισορροπία φόρτου για τους κόμβους αυτούς· (β) σαν συνέπεια της εξ. 2.1, υπάρχει σοβαρή ανισορροπία ακόμα και μεταξύ αυτών των ίδιων κόμβων· και (γ) με (μέγιστο) 2^L αντικείμενα κατανεμημένα σε λ κόμβους, οι λειτουργίες προσχώρησης/αποχώρησης (join/leave) για κάθε ένα από αυτούς τους κόμβους θα έχει σαν αποτέλεσμα τη διασπορά πληροφορίας για $\frac{2^L}{\lambda}$ αντικείμενα – ένα απαγορευτικό κόστος, ανεξάρτητα από το μέγεθος των δεδομένων που διατηρούμε ανά αντικείμενο.

Σχήμα 4.2: Παράδειγμα Προσέγγισης Ραντεβού. Ο κόμβος 48 ορίζεται ως ο κόμβος-ραντεβού και όλοι οι υπόλοιποι κόμβοι του στέλνουν τα σχετικά δεδομένα τους

Προφανώς, η οριακή περίπτωση κατά την οποία αποθηκεύεται μία ολόκληρη σύνοψη κατακερματισμού (δηλαδή όλες οι θέσεις των bits σε όλα τα διανύσματα) σε ένα μοναδικό κόμβο ραντεβού για μια δεδομένη ποσότητα, βρίσκεται σε ακόμα χειρότερη κατάσταση σε σύγκριση με τους παραπάνω περιορισμούς, καθώς ένας μοναδικός κόμβος θα επιβαρυνθεί με όλο το φόρτο – ξεχωρά από το ότι θα αποτελούσε μοναδικό σημείο αστοχίας και θα κατέρριπτε την βασική σχεδιαστική δικλείδα των ΔΟΕ περί αποκεντροποιημένης λειτουργίας. Επιπλέον, προκειμένου να υπολογίσουμε την προσέγγιση από συνόψεις κατακερματισμού για πολλαπλές ποσότητες σε ένα σενάριο βασισμένο σε ραντεβού, θα πρέπει κανείς να επικοινωνήσει με ισάριθμους κόμβους ραντεβού, συνεπώς αντιμετωπίζοντας μια αύξηση στο συνολικό κόστος σε αριθμό βημάτων γραμμική με τον αριθμό των υπολογιζόμενων μετρικών. Προφανώς, η κατανομή των συνόψεων κατακερματισμού κατά μήκος ενός ΔΟΕ DHT πρέπει να προσεγγιστεί από τελείως διαφορετική γωνία. Προς αυτή την κατεύθυνση, προτείνουμε τις Κατανεμημένες Συνόψεις Κατακερματισμού· έναν πολύ αποδοτικό, κλιμακώσιμο και πλήρως αποκεντροποιημένο μηχανισμό επίλυσης του προβλήματος αυτού.

4.1.2 Η Προσέγγιση DHS

Ο πίνακας 4.1 συνοψίζει τους συμβολισμούς που θα χρησιμοποιήσουμε για το υπόλοιπο αυτού του μέρους (με μικρά ελληνικά γράμματα να συμβολίζουν παραμέτρους των DHS και κεφαλαία λατινικά να χρησιμοποιούνται για παραμέτρους σχετικές με το ΔΟΕ DHT). Αρχικά θα περιγράψουμε αλγόριθμους και πρωτόκολλα για συνόψεις κατακερματισμού με ένα μοναδικό διάνυσμα $B[\cdot]$ (δηλ. $\beta = 1$), επεκτείνοντας τη σχεδιάσή μας για πολλαπλά διανύσματα αργότερα.

Προτείνουμε λοιπόν την τμηματοποίηση του χώρου αναγνωριστικών των κόμβων, $[0, 2^L]$, σε λ συνεχή, μη επικαλυπτόμενα διαστήματα $\mathcal{I}_i = [thr(i), thr(i+1))$, $i \in [0, \lambda)$, όπου: $thr(i) = 2^{L-i-1}$. Χρησιμοποιώντας αυτή την τμηματοποίηση, το bit i του $B[\cdot]$ αντιστοιχίζεται σε αναγνωριστικά κόμβων τυχαία (ομοιόμορφα) επιλεγμένα από το $\mathcal{I}_i$. Δηλαδή, για κάθε λειτουργία των DHS (εισαγωγή, ενημέρωση, επερώτηση, κλπ.) σχετική με μία συγκεκριμένη θέση bit i της κατανεμημένης σύνοψης κατακερματισμού, οι κόμβοι επιλέγουν ένα τυχαίο αναγνωριστικό στο $\mathcal{I}_i$ και επικοινωνούν με τον κόμβο που ευθύνεται για αυτό το αναγνωριστικό. Το Σχήμα 4.3(α) απεικονίζει αυτή την αντιστοίχιση και το σχήμα 4.1 συνοψίζει το βασικό αλγόριθμο.

Υπενθυμίζουμε (εξ. 2.1) πως για ένα πολυσύνολο I αντικειμένων, το bit i ενός διανύσματος τυγχάνει επίσκεψης $I \cdot 2^{-i-1}$ φορές. Με τα λ -bit διανύσματα που χρησιμοποιούνται στις DHS, αυτό

Παράμετροι DHT	
L	: Μήκος (σε bits) των κλειδιών του DHT (συνήθως 160 bits)
N	: Αριθμός κόμβων στο δίκτυο επικάλυψης
I	: Αριθμός διακριτών αντικειμένων στο καταναμημένο πολυσύνολο (μέγιστο 2^{80} αντικείμενα)
I_n	: Αριθμός διακριτών αντικειμένων προσφερόμενων από τον κόμβο n
Παράμετροι DHS	
β	: Αριθμός διανυσμάτων στην DHS (συνήθως 256 διανύσματα)
λ	: Μήκος (σε bits) των διανυσμάτων στην DHS (μέγιστο 80 bits· συνήθως 40 bits)
μ	: Αριθμός μετρικών/διαστάσεων στην DHS (μέγιστο 2^{80} μετρικές)
τ	: Μέγιστος αριθμός επαναλήψεων ανά έλεγχο θέσης DHS (συνήθως 5 βήματα)
α	: Βαθμός αντιγραφής δεδομένων της DHS
σ_{type}	: Μέγεθος (σε bytes) των μηνυμάτων τύπου $type$ της DHS

Πίνακας 4.1: Περίληψη Σημειολογίας

Σχήμα 4.3: Αντιστοίχιση Θέσεων DHS σε Κόμβους του ΔΟΕ

Αλγόριθμος 4.1 Αλγόριθμος Αντιστοίχισης: Τυχαία επιλογή αναγνωριστικού στο ζητούμενο διαμέρισμα και πρόσβαση στον αντίστοιχο υπεύθυνο κόμβο

```

1 Algorithm mapBitToNode(Bit bit) {
2   ID targetID = this.computeRandomIdForBit(bit);
3   Node targetNode = DHT.lookup(targetID);
4   return targetNode;
5 }
```

μεταφράζεται σε ένα μέγιστο από (περίπου) 2^λ διακριτά αντικείμενα σε κάθε πιθανό πολυσύνολο, ή σε ένα μέγιστο από $2^{\lambda-I-1}$ αντικείμενα που αντιστοιχίζονται στη θέση i του διανύσματος. Τώρα, παρατηρήστε πως το διάστημα (τόξο) $\mathcal{I}_i$ για τη θέση του bit i αποτελείται από $|\mathcal{I}_i| = 2^{L-i-1}$ αναγνωριστικά, οδηγώντας σε μια ομοιόμορφη κατανομή της πληροφορίας κατά μήκος όλων των κόμβων στο δίκτυο. Επιπλέον, αυτή η αντιστοίχιση παραμένει ίδια για όλες τις μετρικές/ποσότητες που καταμετρούνται και όλα τα διανύσματα της κατανεμημένης σύνοψης κατακερματισμού (βλπ. Σχήμα 4.3(β)). Αυτό μας επιτρέπει να ορίσουμε ή να ελέγξουμε μια συγκεκριμένη θέση bit για πολλαπλές μετρικές και/ή πολλαπλά διανύσματα με μια μοναδική λειτουργία, επικοινωνώντας με μόνο ένα κόμβο στο αντίστοιχο τόξο του χώρου αναγνωριστικών (όπως θα δούμε, στην ενότητα 4.4).

4.2 Δρομολόγηση Μηνυμάτων DHS

Αυτή η ενότητα συζητά τις εργασίες δρομολόγησης των DHS. Πρώτα παρουσιάζουμε τα δύο βασικά πρωτόκολλα δρομολόγησης μηνυμάτων που χρησιμοποιούνται όταν πρέπει να επικοινωνήσουμε με πολλαπλές θέσεις bit της κατανεμημένης σύνοψης κατακερματισμού. Στη συνέχεια περιγράφουμε μια βελτιστοποίηση δρομολόγησης βασισμένη στην εκμετάλλευση των εισαγωγών στους πίνακες δρομολόγησης των κόμβων του ΔΟΕ DHT για την επιτάχυνση των εργασιών των DHS.

4.2.1 Επαναληπτική και Αναδρομική Δρομολόγηση Μηνυμάτων

Όπως θα δούμε σύντομα, υπάρχουν διάφοροι αλγόριθμοι των DHS που απαιτούν την επικοινωνία με κόμβους που αντιστοιχούν σε θέσεις πολλαπλών bit σαν ένα μέρος μιας υψηλότερου επιπέδου λειτουργίας της DHS (όπως η εισαγωγή δεδομένων ή ο υπολογισμός της εκτίμησης της κατανεμημένης σύνοψης). Τότε δημιουργείται το θέμα του πώς πρέπει να δρομολογηθούν τα μηνύματα για μια τέτοια λειτουργία πολλαπλών θέσεων: είτε θα τα χειριστούμε ως μοναδιαίες λειτουργίες, διατρέχοντας το σύνολο των θέσεων των bit και επικοινωνώντας με τους κατάλληλους κόμβους κάθε φορά ή σαν μια μεγάλη λειτουργία που πρέπει να πραγματοποιηθεί σε ένα μοναδικό πέρασμα γύρω από τον κύκλο των αναγνωριστικών των κόμβων.

Το πρώτο σχήμα, αποκαλούμενο Επαναληπτική Δρομολόγηση Bit (*Iterative Bit Routing - IBR*), ακολουθεί το σκεπτικό των αλγορίθμων δρομολόγησης που περιγράφονται στις πρώιμες εργασίες στον χώρο των ΔΟΕ DHT, αποτελούμενο από την επαναληπτική επίσκεψη κόμβων πλησιέστερα στο επιθυμητό αναγνωριστικό με σκοπό την ανάκτηση της διεύθυνσης του επόμενου κόμβου με τον οποίο πρέπει να επικοινωνήσουμε, έως ότου ο επιθυμητός κόμβος βρεθεί. Υποθέστε πως ένας αρχικός κόμβος n_s επιθυμεί να εκτελέσει μια συγκεκριμένη λειτουργία DHS πολλαπλών bit, για θέσεις που βρίσκονται αποθηκευμένες σε ένα διάνυσμα από bit. Ο αρχικός κόμβος επαναλαμβάνει την ίδια διαδικασία κατά μήκος του διανύσματος και για κάθε θέση bit επιλέγει ένα τυχαίο αναγνωριστικό στο αντίστοιχο τόξο του χώρου αναγνωριστικών και ρωτά τον κόμβο που ευθύνεται για

Σχήμα 4.4: Παράδειγμα Δρομολόγησης στο DHS για Πράξεις Πολλαπλών Θέσεων. Ο κόμβος 48 εκτελεί μία πράξη η οποία προσπελαύνει τις θέσεις 0, 1 και 2

αυτό το ID να εκτελέσει τη ζητούμενη λειτουργία. Αυτό το σχέδιο απεικονίζεται στο σχήμα 4.4(α') και συνοψίζεται στον αλγόριθμο ψευδοκώδικα του σχήματος 4.2.

Μετέπειτα ερευνητική δραστηριότητα στον χώρο της δρομολόγησης σε περιβάλλον δικτύων DHS [27, 112] απέδειξε ότι η επαναληπτική δρομολόγηση αντιμετωπίζει πολλά προβλήματα απόδοσης και ανοχής σε λάθη σε πραγματικές εφαρμογές, προτείνοντας την αναδρομική δρομολόγηση ως καλύτερη λύση. Έτσι, υλοποιήσαμε επίσης και ένα σχήμα Αναδρομική Δρομολόγησης Θέσεων (*Recursive Bit Routing* ή *RBR*). Σύμφωνα με αυτό το σχήμα, το οποίο απεικονίζεται στο σχήμα 4.4(β') και συνοψίζεται στο σχήμα 4.3, αρχικά ο n_s ταξινομεί τις θέσεις των bits σε αύξουσα σειρά και τις αποθηκεύει σε ένα διάνυσμα. Κατόπιν, επιλέγει ένα τυχαίο αναγνωριστικό από το τόξο αναγνωριστικών που αντιστοιχεί στη λιγότερο σημαντική από αυτές τις θέσεις και ζητά από τον κόμβο που είναι υπεύθυνος για το αναγνωριστικό αυτό να εκτελέσει την επιθυμητή ενέργεια. Ο κόμβος αυτός, τότε, εκτελεί τοπικά την ζητούμενη λειτουργία και αναδρομικά προωθεί το αποτέλεσμα της τοπικής εκτέλεσης καθώς και ό,τι άλλο του στάλθηκε (π.χ. δεδομένα εισόδου, ορισμό λειτουργίας, κτλ.) σε ένα τυχαία επιλεγμένο αναγνωριστικό στο τόξο αναγνωριστικών για την αμέσως επόμενη θέση στο διάνυσμα θέσεων. Όταν ο αλγόριθμος έχει επεξεργαστεί όλες τις θέσεις στο διάνυσμα, ο τελευταίος κόμβος στην αλυσίδα της αναδρομής μπορεί είτε να απαντήσει απευθείας στον n_s ή να επιστρέψει το τελικό αποτέλεσμα στον προτελευταίο κόμβο στην αλυσίδα της αναδρομής, κάνοντας έτσι (ουσιαστικά) μία ανάστροφη αναδρομή προς τον n_s . Το σχήμα αυτό είναι πιο ανθεκτικό σε διαμερίσεις του δικτύου και άλλα προβλήματα τέτοιας φύσεως, ενώ επιτρέπει στο υπόστρωμα δρομολόγησης να εκμεταλλευτεί πιθανές βελτιστοποιήσεις στους τοπικούς πίνακες δρομολόγησης των κόμβων (για παράδειγμα, τα συστήματα Pastry και Kademlia επιλέγουν ανάμεσα στους υποψήφιους κόμβους για τους πίνακες δρομολόγησης τους βασιζόμενα σε στατιστικά της δικτυακής σύνδεσής τους με τους κόμβους αυτούς, όπως π.χ. η καθυστέρηση (latency) ή ο round-trip χρόνος, κτλ.). Επιπλέον, προλειαίνει το έδαφος για την βελτιστοποίηση κρυφών αντιγράφων bits (bit caching), που θα παρουσιάσουμε σε λίγο.

Αλγόριθμος 4.2 Αλγόριθμος *IBR*: Επαναληπτική εκτέλεση μίας λειτουργίας *op* του DHS για πολλές θέσεις bits

```

1 Algorithm IterativeBitRouting (Operation op) {
2   while (op.getBits().hasMoreBits()) {
3     Bit nextBit = op.getBits().getNextBit();
4     Node targetNode = this.mapBitToNode(nextBit);
5     Result partialResult = targetNode.executeOperation(op);
6     this.processResult(partialResult);
7   }
8 }

```

Αλγόριθμος 4.3 Αλγόριθμος *RBR*: Αναδρομική εκτέλεση μίας λειτουργίας *op* στο DHS για πολλές θέσεις bits. Το *result* αποθηκεύει το μερικό αποτέλεσμα κάθε βήματος της αναδρομής

```

1 Algorithm RecursiveBitRouting (Operation op, Result result) {
2   Result myResult = this.executeOperation(op);
3   myResult.merge(result);
4   if (op.getBits().hasMoreBits()) { // Forward recursion
5     Bit nextBit = op.getBits().getNextBit();
6     Node nextNode = this.mapBitToNode(nextBit);
7     op.addNodeToPath(this);
8     nextNode.recursiveBitRouting(op, myResult, nextBit);
9   } else if (op.respondDirectlyToSender) {
10    op.getSender().processResult(myResult);
11  } else if (op.respondThroughBackwardRecursion) {
12    this.recurseBack(op, myResult);
13  }
14 }
15 recurseBack(Operation op, Result result) {
16   Node prevNode = op.getPrevNodeInPath();
17   // Do any extra work (e.g. result caching) here
18   if (prevNode == null) // Backward recursion reached  $n_s$ 
19     this.processResult(result);
20   else // Continue with backward recursion
21     prevNode.recurseBack(op, result);
22 }

```

4.2.2 Σύντομες Διαδρομές Δρομολόγησης

Ένα ενδιαφέρον «παραπροϊόν» του βήματος τυχαίας επιλογής στον αλγόριθμο αντιστοίχησης θέσεων bits του DHS σε αναγνωριστικά κόμβων (καθώς και σε όλους τους προαναφερθέντες αλγόριθμους δρομολόγησης) είναι ότι, για κάθε δεδομένη θέση δεν ενδιαφερόμαστε για τον συγκεκριμένο κόμβο με τον οποίο θα επικοινωνήσουμε, αρκεί το τόξο αναγνωριστικών για το οποίο αυτός είναι υπεύθυνος να επικαλύπτεται με ή να εμπεριέχεται στο τόξο αναγνωριστικών που αντιστοιχεί στην εκάστοτε θέση. Αυτό μας επιτρέπει να εκμεταλλευτούμε τις συνδέσεις προς σχεδόν τυχαίους κόμβους του δικτύου επικάλυψης DHT στους πίνακες δρομολόγησης των κόμβων, τις λίστες υποψηφίων και τις πιθανές δομές καταγραφής λειτουργιών των κόμβων. Έτσι, αντί να επιλέγουμε ένα τυχαίο αναγνωριστικό στο τόξο προορισμού και μετά να κάνουμε μία κλασσική δρομολόγηση DHT κόστους $O(\log N)$ βημάτων, μπορούμε πρώτα να βρούμε όλους τους κόμβους στις παραπάνω δομές δεδομένων των οποίων το αναγνωριστικό είναι εντός του τόξου προορισμού και α επιλέξουμε τυχαία έναν από αυτούς ως τον κόμβο προορισμού «σύντομης διαδρομής» ή «συντόμευσης». Αν δεν υπάρχει κανένας τέτοιος κόμβος, επανερχόμαστε στην κανονική διαδικασία δρομολόγησης με τυχαία επιλογή αναγνωριστικού και κανονική δρομολόγηση στο DHT για το επόμενο βήμα και να ξαναδοκιμάσουμε τη διαδικασία εύρεσης συντόμευσης στον επόμενο κόμβο. Αυτό μας επιτρέπει να μετατρέψουμε τις περισσότερες από τις λειτουργίες που βασίζονται στο DHT σε λειτουργίες ενός ή δύο βημάτων, χωρίς εμφανή επιρροή στην κατανομή των προσβάσεων σε κόμβους του δικτύου επικάλυψης.

4.3 Εισαγωγές στο DHS

Θα ασχοληθούμε τώρα με τους αλγορίθμους εισαγωγής αντικειμένων στο DHS. Πρώτα θα περιγράψουμε τη διαδικασία εισαγωγής μεμονωμένων αντικειμένων (*Single-Item Insertion* ή *SII*), η οποία χρησιμοποιείται όταν τα αντικείμενα καταφθάνουν το ένα μετά το άλλο και ενδιαφερόμαστε για on-line υπολογισμό του αποτελέσματος (όπως για παράδειγμα σε εφαρμογές ροών (streams)). Κατόπιν, θα παρουσιάσουμε ένα σχήμα πολλαπλών εισαγωγών (*Bulk Item Insertion* ή *BII*), το οποίο ταιριάζει καλύτερα σε σενάρια διαμοιρασμού και διαχείρισης δεδομένων, καθώς και έναν αποδοτικό αλγόριθμο διατήρησης soft-state αντιγράφων. Τέλος, παρουσιάζουμε ένα σχήμα δρομολόγησης το οποίο χρησιμοποιεί την διαχειριστική κίνηση του DHT για να μεταδώσει μηνύματα του DHS, «κρύβοντας» έτσι τελείως το (ήδη χαμηλό) κόστος δικτύου του DHS.

4.3.1 Μεμονωμένες Εισαγωγές

Έστω o ένα αντικείμενο προς εισαγωγή στο DHS. Προχωρούμε ως εξής:

1. Υπολογίζουμε το $h(o)$ χρησιμοποιώντας μια κατάλληλη συνάρτηση κατακερματισμού της επιλογής μας,
2. Χρησιμοποιούμε την έξοδο της συνάρτησης κατακερματισμού ως είσοδο στον τελεστή $\rho(\cdot)$ για να υπολογίσουμε την θέση i του bit που θα θέσουμε,
3. Επιλέγουμε ένα τυχαίο αναγνωριστικό στο διάστημα $[thr(i), thr(i - 1))$ και
4. Στέλνουμε ένα «single-set-bit» (ή *SSB*) μήνυμα στον κόμβο που είναι υπεύθυνος για το αναγνωριστικό αυτό.

Το σχήμα 4.5 συνοψίζει την διαδικασία αυτή. Το μήνυμα SSB αποτελείται από το ζεύγος: $\langle metric_id, bit \rangle$, όπου $metric_id$ είναι ένα μοναδικό αναγνωριστικό για την μετρική που εκτιμούμε (π.χ. τμήμα της εξόδου της συνάρτησης κατακερματισμού με το όνομα του πολυσυνόλου

Σχήμα 4.5: Παράδειγμα Προσέγγισης Βασισμένης στο DHS. Οι κόμβοι εισάγουν τις πλειάδες τους στο DHS και ο φόρτος μοιράζεται στους κόμβους του δικτύου επικάλυψης

ως είσοδο – π.χ. «πληθάριθμος μοναδικών αρχείων στο δίκτυο επικάλυψης» – χρησιμοποιώντας κάποια κρυπτογραφική τέτοια συνάρτηση) και $bit = i$ είναι η θέση στο κατανεμημένο διάνυσμα του bit που θα τεθεί στην τιμή 1. Για παράδειγμα, για την χειρότερη περίπτωση όπου έχουμε 160 bits για το πεδίο $metric_id$ (π.χ. αν χρησιμοποιούσαμε την συνάρτηση SHA-1) και $\lambda = 80$ bits στα διανύσματα της σύνοψης κατακερματισμού (άρα 7 bits ή ≈ 1 byte για το πεδίο bit), το μέγεθος σ_{SSB} των δεδομένων ενός μηνύματος SSB είναι μικρότερο από 22 bytes. Στην μέση περίπτωση, χρησιμοποιώντας αναγνωριστικά μετρικών μήκους 64 bits και 40 bits ανά διάνυσμα (δηλ. $\lambda = 40$) – επιτρέποντας έτσι την καταμέτρηση $\approx 10^{12}$ διακριτών αντικειμένων ανά μετρική και $\approx 10^{19}$ διαφορετικά αναγνωριστικά $metric_id$ – η παραπάνω ποσότητα μειώνεται σε 5-9 bytes ανά μήνυμα.

Το κόστος βημάτων δικτύου για την εισαγωγή ενός αντικειμένου σε ένα DHT/DHS N κόμβων είναι $O(\log N)$ ($\frac{\log N}{2}$ αναμενόμενο [121]), όπως μας εγγυάται το υπόστρωμα DHT, μεταφραζόμενο σε μία συνολική κατανάλωση εύρους ζώνης της τάξης του $O(\sigma_{SSB} \cdot \log N)$. Για ένα δίκτυο επικάλυψης ενός εκατομμυρίου κόμβων και με τα παραπάνω νούμερα χειρότερης περίπτωσης, το παραπάνω μεταφράζεται σε κόστος δικτύου της τάξης των 20 βημάτων (10 αναμενόμενα) και 440 συνολικά bytes (220 bytes αναμενόμενα). Επιπλέον, σημειώνουμε ότι σε σύγκριση με το κόστος εισαγωγής ενός αντικειμένου στο DHT, το κόστος μίας εισαγωγής στο DHS είναι αμελητέο· η εισαγωγή ενός αντικειμένου στο DHT απαιτεί επίσης $O(\log N)$ βήματα, αλλά απαιτεί μεταφορά σαφώς περισσότερων (μεγαλύτερων) δεδομένων – είτε εξαιτίας της μεταφοράς ολόκληρου του αντικειμένου ή (ενός τμήματος) των πλειάδων δεικτοδότησής του ή των δεικτών προς τον πηγαίο κόμβο.

Ένας κόμβος που λαμβάνει ένα τέτοιο μήνυμα :

1. Ελέγχει αν υπάρχει τοπικά αποθηκευμένη πληροφορία για το ζητούμενο $metric_id$.
2. Αν όχι, δημιουργεί μία νέα, κενή σύνοψη κατακερματισμού για το $metric_id$ αυτό.
3. Θέτει την αντίστοιχη θέση bit σε 1 και

4. Αρχικοποιεί το πεδίο *time_out* της θέσης αυτής

4.3.2 Μαζικές Εισαγωγές

Οι μεμονωμένες εισαγωγές είναι καλή επιλογή για streaming εφαρμογές. Ωστόσο, δεν αποδίδουν εξίσου καλά σε εφαρμογές διαμοιρασμού/διαχείρισης δεδομένων. Σε αυτές τις περιπτώσεις, κάθε κόμβος συνήθως αποθηκεύει και διαχειρίζεται εκατοντάδες ή χιλιάδες αντικείμενων τα οποία εισάγει μονομιάς στο δίκτυο επικάλυψης και τα οποία ανανεώνονται είτε σπάνια είτε κατά συστάδες. Χαρακτηριστικό παράδειγμα τέτοιου τύπου εφαρμογών είναι ο διαμοιρασμός αρχείων, όπου οι κόμβοι εισάγουν πληροφορία για όλα τα αρχεία τους με την είσοδό τους στο δίκτυο επικάλυψης, η τιμή κάθε αντικείμενου (δηλ. τα περιεχόμενα κάθε αρχείου) δεν αλλάζουν σχεδόν ποτέ για τη διάρκεια ζωής του αρχείου και οι όποιες ανανεώσεις γίνονται ομαδοποιημένα (συνήθως ως τμήμα μιας περιοδικής διαδικασίας ελέγχου των διαμοιραζόμενων φακέλων για νέα ή διαγραφμένα αρχεία).

Σε αυτές τις περιπτώσεις, το να στέλνει κάθε κόμβος ένα χωριστό μήνυμα για κάθε αντικείμενό του δεν έχει λογική. Για τέτοιου είδους εφαρμογές προτείνουμε τις «μαζικές εισαγωγές» (bulk insertions). Υπενθυμίζουμε ότι το σχήμα αντιστοίχισης θέσεων bits σε αναγνωριστικά κόμβων είναι το ίδιο για όλες τις μετρικές· αυτό έχει ως αποτέλεσμα κάθε δεδομένη θέση bit να αντιστοιχίζεται στο ίδιο τόξο του χώρου αναγνωριστικών για όλες τις μετρικές (βλπ. σχήμα 4.3(β')), επιτρέποντάς μας έτσι να ομαδοποιούμε πολλαπλά μηνύματα SSB για διαφορετικές μετρικές, αν όλα απευθύνονται στην ίδια θέση bit. Στο σενάριο αυτό, ένας κόμβος που θέλει να εισάγει ένα σύνολο αντικειμένων στο DHS, ακολουθεί την εξής διαδικασία:

1. Δημιουργεί ένα τοπικό hash sketch για κάθε μετρική στο πολυσύνολο εισόδου.
2. Εισαγάγει όλα τα αντικείμενα του πολυσυνόλου αυτού στις αντίστοιχες συνόψεις.
3. Αν χρησιμοποιούμε επαναληπτική δρομολόγηση,
 - (α') Για κάθε θέση bit i το οποίο έχει την τιμή 1 σε καθεμιά από τις τελικές συνόψεις
 - i. Επιλέγει ένα τυχαίο αναγνωριστικό κόμβου στο διάστημα $[thr(i), thr(i - 1))$ και
 - ii. Στέλνει ένα μήνυμα «μαζικής εισαγωγής» («bulk-set-bit») – BSB στον κόμβο που είναι υπεύθυνος για το αναγνωριστικό αυτό.
4. Αλλιώς, αν χρησιμοποιούμε αναδρομική δρομολόγηση,
 - (α') Επιλέγει ένα τυχαίο αναγνωριστικό κόμβου στο διάστημα που αντιστοιχεί στο λιγότερο σημαντικό bit και
 - (β') Στέλνει ένα μήνυμα «μαζικής εισαγωγής» στον κόμβο που είναι υπεύθυνος για το αναγνωριστικό αυτό. Ο κόμβος προορισμού θα προωθήσει τότε αναδρομικά το BSB μήνυμα έως ότου η αναδρομή έχει επισκεφθεί όλα τα τόξα που αντιστοιχούν σε θέσεις bits με τιμή 1.

Ένα μήνυμα μαζικής εισαγωγής για μία δεδομένη θέση i αποτελείται από ένα σύνολο από πλειάδες της μορφής $\langle metric_id, bit \rangle$, μία για κάθε διαφορετική μετρική στο πολυσύνολο εισόδου της οποίας η σύνοψη κατακερματισμού έχει την τιμή 1 στη θέση i , μετά το βήμα 2 στον παραπάνω αλγόριθμο. Για μ μετρικές και για μία τυχαία θέση bit, το μέγεθος αυτού του τύπου μηνύματος είναι μ φορές το μέγεθος ενός SSB μηνύματος στην χειρότερη περίπτωση (δηλ. όταν και οι μ μετρικές έχουν την τιμή 1 στο bit i). Έτσι, συνολικά, προκειμένου ένας κόμβος να εισαγάγει όλα τα I_n αντικείμενά του χρησιμοποιώντας μία μαζική εισαγωγή, απαιτούνται το πολύ λ μηνύματα BSB (ένα για κάθε θέση bit), καθένα από τα οποία θα έχει μέγεθος αντίστοιχο μ μηνυμάτων SSB

στην χειρότερη περίπτωση (δηλ. όταν και τα λ bits και των μ μετρικών έχουν την τιμή 1). Αυτό μεταφράζεται σε απόδοση σε αριθμό βημάτων $O(\lambda \cdot \log N)$ · από την άλλη, ο αναμενόμενος αριθμός βημάτων είναι $\log I_n \cdot \frac{\log N}{2}$, αφού με I_n αντικείμενα ανά κόμβο, μόνο $\log I_n < \lambda$ bits θα έχουν τιμή 1 κατά μέσο όρο. Οι κόμβοι που λαμβάνουν τέτοιου είδους μηνύματα ακολουθούν την ίδια διαδικασία που περιγράφηκε παραπάνω για την περίπτωση των μοναδιαίων εισαγωγών, μόνο που στην τρέχουσα περίπτωση ενημερώνουν τις τοπικές επαυξημένες συνόψεις κατακερματισμού τους για όλες τις μετρικές που περιλαμβάνονται στο BSB μήνυμα.

Τα δύο σχήματα εισαγωγών που περιγράφηκαν παραπάνω μπορούν να συνυπάρχουν με φυσικό τρόπο. Έτσι, κάθε κόμβος μπορεί να εισαγάγει τα αντικείμενά του και με τους δύο τρόπους ή ακόμα να εναλλάσσει ανάμεσά τους. Αν, για παράδειγμα, το DHS χρησιμοποιείται σε μια εφαρμογή διαμοιρασμού δεδομένων αλλά ένας κόμβος θελήσει να εισαγάγει ή να ανανεώσει έναν μικρό αριθμό αντικειμένων, μπορεί να επιλέξει πολλαπλές μεμονωμένες εισαγωγές, αν το άθροισμα των μεγεθών των μηνυμάτων SSB είναι μικρότερο από το μέγεθος του αντίστοιχου BSB μηνύματος. Αντιστρόφως, σε εφαρμογές ροών, μπορεί ένας κόμβος να επιλέξει να ομαδοποιήσει πολλές εισαγωγές/ανανεώσεις και να χρησιμοποιήσει μαζικές λειτουργίες, αν η είσοδος έχει καταγιγιστική μορφή.

4.3.3 Ανανεώσεις/Διαγραφές στο DHS

Επιλέγουμε μία soft-state προσέγγιση για την διαχείριση των δεδομένων του DHS: κάθε θέση bit με τιμή 1 συνοδεύεται από ένα πεδίο *time_out*, το οποίο καθορίζει ένα διάστημα εγκυρότητας (time-to-live interval) για το αντίστοιχο bit και κάθε κόμβος περιοδικά επανεισαγάγει τα αντικείμενά του στο DHS. Η διαγραφή των δεδομένων που αποθηκεύονται στο DHS γίνεται με έμμεσο τρόπο: μία θέση bit που δεν έχει ανανεωθεί για μία περίοδο *time_out* τίθεται στην τιμή 0. Ως εκ τούτου, οι συνόψεις κατακερματισμού που αναφέρθηκαν στο βήμα 2 παραπάνω αποτελούν επαυξημένη μορφή της δομής δεδομένων που περιγράφηκε στην ενότητα 2.4.2, αποτελούμενη από το πεδίο *metric_id* συν ένα σύνολο από β διανύσματα, καθένα εκ των οποίων περιέχει λ 32-bit ακεραίους¹, οι οποίοι καταδεικνύουν την τιμή *time_out* της αντίστοιχης θέσης bit, με θέσεις με *time_out* τιμή 0 να θεωρούνται ότι έχουν επίσης τιμή 0. Ονομάζουμε τη δομή δεδομένων αυτή *επαυξημένη σύνοψη κατακερματισμού* για να την διαχωρίζουμε από τις κανονικές συνόψεις κατακερματισμού οι οποίες περιγράφηκαν στην ενότητα 2.4.2. Ως παράδειγμα, στην χειρότερη περίπτωση με αναγνωριστικά μετρικών των 160 bits και 80 bits ανά διάνυσμα, μία τέτοια δομή δεδομένων θα απαιτεί περίπου 340 bytes αποθηκευτικού χώρου. Επίσης, σημειώνουμε ότι ένας κόμβος τον οποίο δεν έχουμε επισκεφθεί κατά την διαδικασία εισαγωγής αντικειμένων (π.χ. επειδή δεν επιλέχθηκε κάποιο αναγνωριστικό στην περιοχή ευθύνης του κατά το βήμα τυχαίας επιλογής) δεν θα αποθηκεύει δεδομένα για την αντίστοιχη μετρική.

Η παράμετρος *time_out* μπορεί να θεωρηθεί ως το αντίστοιχο του aggregation window σε περιβάλλοντα ροών πληροφορίας, ενώ για συστήματα διαχείρισης δεδομένων μπορεί να ρυθμιστεί ανάλογα με τον λόγο εισαγωγών/ανανεώσεων και ερωτημάτων. Η ακριβής τιμή της παραμέτρου αυτής εξαρτάται σε μεγάλο βαθμό από την εκάστοτε εφαρμογή και θέτει μία ενδιαφέρουσα σχεδιαστική επιλογή. Προφανώς μεγαλύτερες τιμές θα απαιτούν ως αποτέλεσμα λιγότερες ανανεώσεις ανά μονάδα χρόνου για την διατήρηση των δεδομένων του DHS. Από την άλλη, μία μικρότερη τιμή θα επιτρέπει γρηγορότερη προσαρμογή σε απότομες αλλαγές της τιμής της εκτιμώμενης μετρικής, αλλά θα αυξάνει τις απαιτήσεις σε δικτυακούς πόρους. Σε κάθε περίπτωση, όπως φαίνεται και από τα προαναφερθέντα παραδείγματα χειρότερων περιπτώσεων, οι απαιτήσεις σε αποθηκευτικό χώρο και εύρος ζώνης δικτύου ανά κόμβο του DHS είναι πολύ χαμηλές, οπότε ανάλογα την εφαρμογή ακόμα και ένας μεγάλος ρυθμός ανανέωσης μπορεί να μεταφράζεται σε μικρή κατανάλωση εύρους ζώνης δικτύου.

¹ Μεγέθους 32-bits (=4 bytes), σύμφωνα με τον τύπο δεδομένων *time_t* του προτύπου POSIX.1

Μια ενδιαφέρουσα εναλλακτική είναι να επιτρέπουμε στους κόμβους να θέτουν την τιμή της παραμέτρου *time_out* αυτόνομα για τα αντικείμενά τους. Για να επιτευχθεί αυτό, πρέπει να προσθέσουμε και ένα επιπλέον πεδίο *time_out* στα SSB/BSB μηνύματα. Κάθε κόμβος θα μπορεί τότε να επιλέγει αυτόνομα την τιμή του πεδίου αυτού σε κάθε εισαγωγή αντικειμένου. Οι κόμβοι προορισμού θα «θέτουν» τότε το αντίστοιχο bit στην κατάλληλη τοπική επαυξημένη σύνοψη κατακερματισμού, δίνοντας στο πεδίο *time_out* την μέγιστη τιμή ανάμεσα στην τρέχουσα τιμή του και την τιμή του στο εισερχόμενο μήνυμα. Η προσθήκη αυτή αυξάνει το σ_{SSB} στα 26 bytes στην χειρότερη περίπτωση (όπως περιγράφηκε νωρίτερα) και στα 9–13 bytes στη μέση περίπτωση.

4.3.4 Δημιουργία Αντιγράφων σε Γειτονικούς Κόμβους

Σε πολλές κατηγορίες εφαρμογών η ανεκτικότητα σε λάθη ή/και η διαθεσιμότητα των δεδομένων είναι πιο σημαντικές από την καθαρή απόδοση του συστήματος σε αριθμό βημάτων ή κατανάλωση εύρους ζώνης δικτύου. Τέτοιες εφαρμογές απαιτούν διατήρηση αντιγράφων (replication) της πληροφορίας του DHS. Στο στάδιο αυτό, το σύστημά μας προσφέρει δύο επιλογές: μπορούμε είτε να στηριχθούμε στο χρησιμοποιούμενο υπόστρωμα DHT για να μας προσφέρει τη δυνατότητα αυτή, όπως ισχύει με τα περισσότερα σύγχρονα τέτοια δίκτυα επικάλυψης, είτε να υλοποιήσουμε μία δική μας πολιτική για replication. Στη δεύτερη περίπτωση, όταν εισάγουμε (ή ανανεώνουμε) πληροφορία για κάποιο bit στο DHS, επισκεπτόμενοι κάποιον κόμβο στο διάστημα αναγνωριστικών του bit αυτού, στέλνουμε το ίδιο μήνυμα SSB/BSB και σε έναν αριθμό α επόμενων/προηγούμενων κόμβων, χρησιμοποιώντας μοναδιαία άλματα. Με α αντίγραφα για κάθε bit, το κόστος βημάτων χειρότερης περίπτωσης για εισαγωγές μεμονωμένων αντικειμένων είναι $O(\log N + \alpha)$ (αναμενόμενο $\frac{\log N}{2} + \alpha$)· για την περίπτωση μαζικών εισαγωγών, το κόστος γίνεται $O(\lambda \cdot (\log N + \alpha))$ (αναμενόμενο $\log I_n \cdot (\frac{\log N}{2} + \alpha)$). Όπως θα δούμε και στην πειραματική μας μελέτη, ένα σχήμα αντιγραφής αυτού του είδους είναι επιπλέον ευεργετικό κατά τα στάδια επερωτήσεων στο DHS.

4.3.5 Απόκρυψη Μηνυμάτων DHS στην Κίνηση Διαχείρισης του DHT

Υπενθυμίζουμε ότι οι κόμβοι του δικτύου επικάλυψης DHT χρησιμοποιούν μηνύματα τύπου PING/PONG και περιοδικούς ελέγχους για να ανιχνεύσουν σφάλματα στις συνδέσεις τους με άλλους κόμβους και κενά στους πίνακες δρομολόγησής τους, καθώς και ότι περιοδικά αναδημιουργούν τους τελευταίους ως μέρος του πρωτοκόλλου σταθεροποίησής τους (stabilization protocol). Σε τυπικά, ευρέως καταναεμημένα πραγματικά συστήματα βασισμένα σε δίκτυα επικάλυψης DHT, τα μηνύματα αυτά στέλνονται κάθε 20–30 δευτερόλεπτα ανά σύνδεσμο, ενώ το πρωτόκολλο σταθεροποίησης εκτελείται κάθε 15–20 λεπτά της ώρας. Βασιζόμενοι στην ίδια παρατήρηση με παραπάνω, η ενσωμάτωση (piggybacking) της διαχειριστικής κίνησης του DHS (επονομαζόμενη *MTP*) κατά κάποιο τρόπο “κρύβει” μηνύματα σχετικά με το DHS μέσα σε μηνύματα διαχείρισης του υποστρώματος DHT. Ανάλογα με τα πρότυπα χρήσης μίας τυχαίας εφαρμογής του DHS, οι λειτουργίες του DHS μπορούν να εκμεταλλευτούν αυτού του είδους την επικοινωνία ανάμεσα στους κόμβους του δικτύου και να στείλουν τα δεδομένα τους μαζί με τα μηνύματα του DHT υποστρώματος. Όπως έχουμε ήδη δει, τα μηνύματα εισαγωγής/ανανέωσης του DHS έχουν αμελητέο μέγεθος. Επομένως, μία τέτοιου είδους ενσωμάτωσή τους σε DHT μηνύματα θα καταστήσει την διαχείριση του DHS πρακτικά άνευ κόστους, χωρίς παρατηρήσιμη αύξηση στην συνολική χρησιμοποίηση του εύρους ζώνης του δικτύου.

4.4 Αύξηση της Ακρίβειας του DHS

Μέχρι τώρα έχουμε ασχοληθεί με μονού-πίνακα/διανύσματος συνόψεις κατακερματισμού. Μολαταύτα οι εκτιμήσεις που αποκτάμε από τέτοια απλά σχήματα, τυπικά μπορούν να αποκλίνουν

από την πραγματικότητα κατά μία δυαδική τάξη μεγέθους [33,35]. Όπως αναφέραμε στην ενότητα 2.4.2 η ακρίβεια των παραγόμενων εκτιμήσεων από τις συνόψεις κατακερματισμού μπορεί να βελτιωθεί με την χρήση πολλαπλών πινάκων/διανυσμάτων. Η επέκταση των παραπάνω αλγορίθμων για $\beta > 1$ (το β είναι δύναμη του 2) είναι προφανής. Η εισαγωγή ενός αντικειμένου o πραγματοποιείται διαλέγοντας ένα από τα β διανύσματα, χρησιμοποιώντας τα τελευταία $\log \beta$ bits της $h(o)$, ενώ στην συνέχεια χρησιμοποιούμε τα υπολειπόμενα $\lambda - \log \beta$ bits σαν είσοδο στην συνάρτηση $\rho(\cdot)$ προκειμένου να επιλέξουμε το bit στην θέση i το οποίο θα πάρει τιμή 1 σε αυτό το συγκεκριμένο διάνυσμα. Σημειώνουμε εδώ ότι ακόμα δεν έχουμε αλλάξει το σχήμα αντιστοίχισης από τις τιμές των bits σε αναγνωριστικά κόμβων (βλέπε Σχήμα 4.3(β))· αυτό μας επιτρέπει την περαιτέρω ταυτόχρονη ομαδοποίηση SSB/BSB μηνυμάτων από διαφορετικά διανύσματα, αν όλα είχαν κάποιο bit με τιμή 1 στην ίδια θέση.

Τα SSB μηνύματα πρέπει τώρα να επεκταθούν με ένα πεδίο *vector_id*, ώστε να γίνεται διακριτό ποιο διάνυσμα ανανεώνεται για μία δεδομένη μετρική. Για ένα μέγιστο 1024 διανυσμάτων για κάθε σύνοψη κατακερματισμού, το πεδίο *vector_id* πρέπει να έχει εύρος 10 bits, επομένως το σ_{SSB} θα είναι περίπου 24 bytes (6-10 bytes στην μέση περίπτωση με 256 διανύσματα). Αντίθετα, ένα BSB μήνυμα για την i -οστή θέση bit τώρα περιέχει ένα σύνολο από $\langle metric_id, bit, \langle bit_vector \rangle \rangle$ πλειάδες. Το τελευταίο έχει β bits πλάτος και περιέχει την τιμή 1 για κάθε διάνυσμα που το bit στην i -οστή θέση είναι επίσης 1. Στην χειρότερη των περιπτώσεων, το μέγεθος των μηνυμάτων αυτού του τύπου θα είναι 148 bytes ανά μετρική, αν όλα τα 1024 διανύσματα της δεδομένης μετρικής έχουν 1 σε μία συγκεκριμένη θέση. Στην μέση περίπτωση (πχ 256 32-64-bit διανύσματα) σε αντίθεση, το σ_{BSB} θα είναι περίπου 28 bytes ανά μετρική. Τέλος, ο αποθηκευτικός χώρος που απαιτείται για κάθε εκτιμώμενη μετρική πολλαπλασιάζεται επί β . Κάτι τέτοιο είναι περίπου 320 Kbytes για κάθε επαυξημένη σύνοψη κατακερματισμού στην χειρότερη των περιπτώσεων (160-bit metric IDs, 1024 80-bit διανύσματα), ενώ στην μέση περίπτωση είναι περίπου 65kbytes (32-64-bit metric IDs, 256 32-64-bit διανύσματα).

Να σημειωθεί ότι, ενώ κάθε στοιχείο θέτει μόνο ένα bit, ακόμα και με πολλαπλά διανύσματα ανά σύνοψη κατακερματισμού, λόγω της στοχαστικής μέσου όρου τεχνικής των [33, 35] το κόστος της εισαγωγής σε αριθμό βημάτων παραμένει το ίδιο, τόσο για μία και μόνο εισαγωγή όσο και για τις περιπτώσεις των μαζικών εισαγωγών, ανεξαρτήτως του αριθμού των διανυσμάτων και/ή των μετρικών που χρησιμοποιούνται. Αφετέρου το κόστος σε εύρος ζώνης δικτύου είναι επίσης ανεξάρτητο από τον αριθμό των διανυσμάτων και/ή των μετρικών για την περίπτωση της εισαγωγής ενός στοιχείου ενώ υπάρχει μία μικρή αύξηση στην περίπτωση των μαζικών εισαγωγών.

4.5 Καταμέτρηση με το DHS

Με αυτή την υποδομή διαθέσιμη θα επικεντρωθούμε τώρα στον σχεδιασμό και στην υλοποίηση του υποσυστήματος εκτίμησης των DHS. Πρώτα θα παρουσιάσουμε αλγόριθμους για να συγκεντρώσουμε τα δεδομένα των bits της DHS από τους κόμβους του υποστρώματος, με σκοπό να ανακατασκευάσουμε την κατανομημένη σύνοψη κατακερματισμού και να υπολογίσουμε την επιθυμητή εκτίμηση. Έπειτα θα συζητήσουμε την πολυδιάστατη καταμέτρηση (δηλ. την ταυτόχρονη εκτίμηση πολλαπλών ποσοτήτων) και θα παρουσιάσουμε ένα απλό σχήμα κρυφής αποθήκευσης bits, το οποίο μειώνει αισθητά και το κόστος αριθμού βημάτων των ερωτημάτων αλλά και το φορτίο στους κόμβους του υποστρώματος. Θα συζητήσουμε επίσης τα θέματα που προκύπτουν εξαιτίας του σταδίου της τυχαίας επιλογής στους αλγόριθμους εισαγωγής και ελέγχου bits και θα δώσουμε μία balls-and-bins θεωρητική ανάλυση.

Υπενθυμίζουμε ότι η εκτίμηση του αριθμού των διακριτών στοιχείων σε ένα υπερσύνολο χρησιμοποιώντας συνόψεις κατακερματισμού αποτελείται από (α) τον υπολογισμό των θέσεων $R^{(i)}$ των πιο σημαντικών 1-bit θέσεων για τον superLogLog εκτιμητή ή των λιγότερο σημαντικών 0-bits

Αλγόριθμος 4.4 Αλγόριθμος Επεξεργασίας Μηνυμάτων «single-check-bit(s)»

```
1 Algorithm processSCB (SCBMessage msg) {  
2   HashSketch augmentedHashSketch = storage.getHashSketch (msg.getMetricId ());  
3   if (localHashSketch != null) {  
4     if (msg.hasBit ())  
5       return augmentedHashSketch.isBitSet (msg.getBit ());  
6     else  
7       return augmentedHashSketch.getStandardHashSketch ();  
8   }  
9 }
```

για τον PCSA εκτιμητή, στο $B[\cdot]$, (β) τον υπολογισμό του («συντετμημένου» – truncated) αριθμητικού μέσου και (γ) τη χρησιμοποίηση της εξίσωσης 2.2 (superLogLog) ή της εξίσωσης 2.4 (PCSA) για τον υπολογισμό μίας εκτίμησης του πληθάριθμου των πολυσυνόλων που μας ενδιαφέρουν. Το δεύτερο και τρίτο είναι προφανή και έχουμε ασχοληθεί μαζί τους κατά το πρώτο βήμα.

4.5.1 Βασικός Έλεγχος Θέσεων

Για κάθε δεδομένη μετρική, ο έλεγχος για το καθεστώς των θέσεων των ψηφίων της αντίστοιχης κατανεμημένης σύνοψης κατακερματισμού ακολουθεί την ίδια ιδέα της εισαγωγής ενός στοιχείου· με άλλα λόγια, αν ένας κόμβος θέλει να ελέγξει αν ένα bit είναι set ή όχι, για μία συγκεκριμένη μετρική, επιλέγει πρώτα ένα τυχαίο ID από το σύνολο των αναγνωριστικών που αντιστοιχούν σε αυτή την θέση bit και στην συνέχεια στέλνει ένα «single-check-bit(s)» μήνυμα στον κόμβο που είναι υπεύθυνος για αυτό το ID. Ένα SCB αποτελείται από ένα ζεύγος $\langle metric_id, bit \rangle$, όπως ακριβώς τα SSB μηνύματα. Οι κόμβοι έχουν την επιλογή να παραλείψουν το πεδίο bit, ζητώντας σε αυτή την περίπτωση το σύνολο των bit δεδομένων για την συγκεκριμένη μετρική που είναι αποθηκευμένη στον κόμβο προορισμού. Ακολουθώς, όταν ληφθεί ένα SCB μήνυμα, εκτελούνται τα βήματα που περιγράφονται στο σχήμα 4.4. Εν συντομία, η απάντηση είτε αποτελείται από ένα και μόνο bit το οποίο αντιπροσωπεύει την κατάσταση της θέσης του bit που ζητήθηκε για την πρώτη περίπτωση, είτε περιλαμβάνει μία πλήρη σύνοψη κατακερματισμού (υπολογισμένη από αντιστοιχίες τιμών *time_out* στην επαυξημένη σύνοψη κατακερματισμού σε τιμές bit) για τον PCSA εκτιμητή, είτε τις θέσεις των πιο σημαντικών ψηφίων με τιμή 1 στα διανύσματα για τον εκτιμητή του superLogLog. Επιπρόσθετα ο ανταποκρινόμενος κόμβος περιλαμβάνει στην απάντηση τα ID που ανταποκρίνονται στο άνω και κάτω όριο της περιοχής στην οποία είναι υπεύθυνος. Η πληροφορία αυτή χρησιμοποιείται έπειτα κατά τη διάρκεια ζωής του τρέχοντος ερωτήματος με στόχο να αποφύγουμε να ξαναεπισκεφτούμε τους ίδιους κόμβους κατά την διάρκεια του βήματος της τυχαίας επιλογής. Μία κενή απόκριση σημαίνει ότι ο κόμβος προορισμού δεν έχει αποθηκευμένη καμία πληροφορία για την αιτούμενη μετρική και/ή bit. Οι κόμβοι μπορούν επιπλέον να ελέγξουν την κατάσταση πολλαπλών μετρικών με ένα και μοναδικό «bulk-check-bit(s)» (BCB) μήνυμα. Ένα BCB μήνυμα αποτελείται από ένα σύνολο SCB μηνυμάτων, ένα για κάθε ενδιαφερόμενη μετρική. Οι κόμβοι μπορούν να επιλέξουν να παραλείψουν το πεδίο bit σε οποιαδήποτε συμπεριλαμβανόμενο SCB μήνυμα, με τα ίδια επακόλουθα όπως και παραπάνω. Ξανά, ο κόμβος προορισμού εκτελεί την διαδικασία που περιγράφεται στο σχήμα 4.4 για κάθε μετρική στο BCB μήνυμα.

4.5.2 Σειριακός και Δυναδικός Έλεγχος Θέσεων

Ο απλούστερος αλγόριθμος για να βρούμε τις $R^{(i)}$ θέσεις των bit που απαιτούνται για τον υπολογισμό της σύνοψης κατακερματισμού θα έπρεπε να επισκεφτούμε καθένα από τα διαστήμα-

Αλγόριθμος 4.5 Αλγόριθμος *estimateSBP*: Εκτίμηση του αριθμού διακριτών στοιχείων σε ένα πολυσύνολο με χρήση DHS με σειριακό έλεγχο θέσεων. Το $R[\beta]$ αποθηκεύει τις θέσεις των λιγότερο σημαντικών 0-bits για τον PCSA ή των πιο σημαντικών 1-bits για τον superLogLog εκτιμητή αντίστοιχα

```

1 Algorithm estimateSBP (Metric metric_id) {
2   HashSketch hashSketch;
3   for (int bit = 0; bit < λ; bit++) {
4     Node targetNode = this.mapBitToNode (bit);
5     Result bitData = targetNode.processSCB(new SCBMessage (metric_id) );
6     hashSketch.merge (bitData);
7     if (hashSketch.bitIsZeroForAllVectors (bit))
8       break;
9   }
10  int R[β] = hashSketch.computeRValues ();
11  return  $\tilde{\alpha}_\beta \cdot \beta_0 \cdot 2^{\frac{1}{\beta_0}} \cdot \sum^* (R[\cdot])$ 
12 }
```

τα που αντιστοιχούν στις λ θέσεις bit των β κατανεμημένων $B^{(i)}[\cdot]$ διανυσμάτων και να ελέγχουμε κάθε πότε υπάρχει ένα στοιχείο καταγεγραμμένο εκεί. Σε ένα περιβάλλον ΔΟΕ, το δικτυακό κόστος – σε όρους τόσο σε αριθμού βημάτων αλλά και σε κατανάλωση εύρους – είναι μακράν ο κυριότερος λειτουργικός παράγοντας, έτσι ένας τέτοιος αλγόριθμος θα ήταν υπερβολικός. Μολαταύτα διαπιστώσαμε ότι μπορούμε να αποφύγουμε πολλές από αυτές τις επαναλήψεις αν ρυθμίσουμε την σειρά με την οποία οι θέσεις των bit ερευνώνται. Προς αυτή την κατεύθυνση, έχουμε προσδιορίσει και υλοποιήσει δύο εναλλακτικές, ονόματι Σειριακός Έλεγχος Θέσεων (Sequential Bit Probing ή SBP) και Δυαδικό Έλεγχος Θέσεων ή Έλεγχος Θέσεων Δυαδικής Αναζήτησης (Binary-search Bit Probing ή BBP). Όπως και τα ονόματα υποδηλώνουν, στην SBP οι θέσεις των bit ερευνώνται με αριθμητική σειρά, ενώ στην BBP ακολουθούν το πρότυπο της δυαδικής αναζήτησης.

Για τον PCSA εκτιμητή το $R^{(i)}$, $i \in [0, \beta)$ αποθηκεύει τις θέσεις των κατώτερων μηδενικών bit για κάθε διάνυσμα, έτσι διαισθητικά θα ερευνούσαμε διαδοχικά τις θέσεις των bit κατά αύξουσα σειρά, ξεκινώντας από το bit 0 (λιγότερο σημαντικό bit) και πηγαίνοντας μέχρι την θέση $\lambda - 1$ (περισσότερο σημαντικό bit), μέχρι ότου να υπάρξει τουλάχιστον ένα μηδενικό bit σε κάθε διάνυσμα. Από την άλλη στον εκτιμητή του superLogLog μας ενδιαφέρουν τα υψηλότερα 1-bits, έτσι λογικά θα προχωρούσαμε στην αντίθετη κατεύθυνση: ερευνώντας τις θέσεις των bit κατά φθίνουσα σειρά αρχίζοντας από την $\lambda - 1$ μέχρι είτε να φτάσουμε στη θέση 0 ή να υπάρξει τουλάχιστον ένα μηδενικό bit σε κάθε διάνυσμα. Και οι δύο αλγόριθμοι δουλεύουν τέλεια στην πράξη. Ωστόσο ο τελευταίος έχει ένα ανεπαίσθητο ελάττωμα: αρχίζει να ερευνά από το μικρότερο διάστημα αναγνωριστικών (που αντιστοιχεί στην θέση bit $\lambda - 1$), η οποία συνήθως αποτελείται από έναν μικρό αριθμό από κόμβους (ένα ή δύο στην χειρότερη περίπτωση). Αυτό οδηγεί σε μία ανομοιομορφία φόρτου όσον αφορά την επεξεργασία ερωτημάτων (η ισορροπία στο φόρτο εισαγωγής και αποθήκευσης μένει ανεπηρέαστη), με τους περισσότερο φορτωμένους κόμβους να μοιράζονται σχεδόν το ίδιο συσσωρευμένο φορτίο όπως και στην περίπτωση ενός κόμβου ραντεβού. Ο PCSA εκτιμητής δεν χωλαίνει από αυτό το ζήτημα, αφού πρώτα ερευνά τα μεγαλύτερα διαστήματα αναγνωριστικών, εξαπλώνοντας έτσι το φορτίο σε πολύ περισσότερους κόμβους (τους μισούς από τον συνολικό αριθμό) και τερματίζει όσο το δυνατό πιο γρήγορα.

Για να αντιμετωπίσουμε αυτό το πρόβλημα, προτείνουμε την αλλαγή στον αλγόριθμο που ερευνά τα bit, ώστε να ξεκινά από το bit στην θέση 0 και να κινείται προς τις υψηλότερες θέσεις bit και για τους δύο εκτιμητές, τελειώνοντας όταν βρεθεί μία θέση bit για την οποία όλα τα διανύσματα

έχουν ένα μηδενικό bit (ο τελικός αλγόριθμος σειριακού ελέγχου και για τον PCSA και για τον superLogLog εκτιμητή συνοψίζεται στο σχήμα 4.5). Η διαίσθηση πίσω από αυτή την επιλογή βασίζεται στην παρακάτω εμπειρική λογική παρατήρηση: για μία δεδομένη θέση bit i και για μία σύνοψη κατακερματισμού με β διανύσματα $B^{(i)}[\cdot]$ η πιθανότητα να υπάρχει ένα μηδενικό bit στην θέση k για όλα τα β διανύσματα – δηλ. $B^{(m)}[k] = 0, \forall m \in [0, \delta)$ – και να υπάρχει ένα μη μηδενικό bit σε κάποια θέση $l > k$ είναι αμελητέα για $\beta \gg 1$ και γίνεται ακόμα μικρότερη για μεγαλύτερο k (υποκείμενο σε στατιστικές αυξομειώσεις στις τιμές του $R^{(i)}$)². Συν τοις άλλοις, ειδικά για τον superLogLog εκτιμητή, ακόμα και όταν υπάρξουν τέτοιες εξαιρετικά σπάνιες περιπτώσεις (δεν έχουμε ανακαλύψει ακόμα καμία έπειτα από μερικές εκατομμύρια εκτελέσεις των αλγορίθμων σε δίκτυα διαφόρων μεγεθών και με εισόδους σύνολα δεδομένων διαφορετικών μεγεθών, πεδίων τιμών και κατανομών), θα απορριφθούν στο βήμα της αποκοπής.

Ο δεύτερος αλγόριθμος ελέγχου θέσεων, ο Έλεγχος Δυαδικής Αναζήτησης, βασίζεται στην ίδια παραπάνω παρατήρηση. Ο αλγόριθμος εξελίσσεται σε τρία στάδια, όπως διαφαίνεται στο σχήμα 4.6. Καταρχήν, ερευνά για τις θέσεις των bit που ανήκουν στο διάστημα $[0, \lambda)$ με ένα τρόπο που μοιάζει σαν δυαδική αναζήτηση, μέχρις ότου να βρει την μικρότερη θέση bit p_h για την οποία όλα τα διανύσματα έχουν ένα μηδενικό bit. Ακολούθως, αν χρησιμοποιείται ο PCSA εκτιμητής, ερευνά τις θέσεις των bits που ανήκουν στο $[0, p_h)$ για την υψηλότερη θέση bit p_l για την οποία όλα τα διανύσματα έχουν ένα μη μηδενικό bit. Τέλος, επανέρχεται σε σειριακή αναζήτηση για θέσεις των bit που ανήκουν στο διάστημα (p_l, p_h) για τον PCSA και στο διάστημα $(p_l, 0]$ για τον superLogLog εκτιμητή. Ο αλγόριθμος καταγράφει τις θέσεις των bit που ερευνά κατά την διάρκεια αυτών των φάσεων έτσι ώστε οι θέσεις των bits που έχουμε επισκεφτεί στην διάρκεια κάποιας φάσης, να παραλειφθούν στα επόμενα στάδια. Το αποτέλεσμα στο δίκτυο είναι μία μείωση στο ολικό εκτιμώμενο κόστος μετάβασης και ελαφρώς καλύτερη κατανομή του φόρτου.

Ας πάρουμε για παράδειγμα ένα σύνολο δεδομένων που αποτελείται από 2,500,000 μοναδικές εγγραφές και ας υποθέσουμε ότι θα χρησιμοποιήσουμε 512 32-bit διανύσματα στην σύνοψη κατακερματισμού. Η υψηλότερη θέση bit με τιμή 1 στην μέση κατάσταση στην σύνοψη κατακερματισμού με αυτό το σύνολο δεδομένων είναι περίπου 16, όπως υπολογίστηκε από την εξίσωση 2.3. Με 32 bits ανά διάνυσμα σύνοψης κατακερματισμού, αναμένουμε ο BBP να ερευνά κατά μέσο όρο 5-6 θέσεις bit κατά τα αρχικά στάδια του αλγορίθμου (μέχρις ότου να εντοπίσει την χαμηλότερη θέση bit με όλα τα διανύσματα μηδέν, π.χ. τη θέση 17) και τόσα bits κατά το τρίτο στάδιο όσα είναι απαραίτητα μέχρις ότου εντοπίσει τουλάχιστον ένα μη μηδενικό bit για κάθε διάνυσμα (το δεύτερο στάδιο είναι ενεργό μόνο στον PCSA εκτιμητή). Αυτό μπορεί να μεταφραστεί σε 12-21 θέσεις bit να ερευνώνται κατά την διάρκεια της εκτίμησης, ενώ ο κλασσικός σειριακός αλγόριθμος ελέγχου ερευνά 18 θέσεις bit (τις θέσεις από 0 ως 17). Αναμένουμε ότι ο BBP θα είναι καλύτερος από τον SBP για μεγαλύτερα σύνολα τιμών και διανύσματα και να είναι τουλάχιστον τόσο καλός όσο ο SBP στις μικρότερες περιπτώσεις.

Σημειώνουμε ότι στην χειρότερη περίπτωση η απόδοση σε αριθμό βημάτων και για τους δύο αλγόριθμους είναι ανεξάρτητη από τον αριθμό των διανυσμάτων και από τον αριθμό των στοιχείων στο υπόστρωμα: $O(\lambda \cdot \log N)$ βήματα στην χειρότερη περίπτωση, με ανεμενόμενη³ απόδοση $O(\log \frac{I}{\beta} \cdot \log N)$ και για τους δύο αλγόριθμους έρευνας. Αυτή η παρατήρηση συνιστά μία βασική ιδιότητα των DHS και ένα από τα μεγαλύτερα πλεονεκτήματά τους: το κόστος αριθμού βημάτων καταμέτρησης παραμένει το ίδιο, ανεξάρτητα από τον αριθμό των διανυσμάτων και των διαστάσεων (μετρικών). Αυτό είναι έτσι γιατί η αντιστοίχιση από θέσεις bit στο χώρο αναγνωριστικών κόμβων είναι η ίδια για όλα τα διανύσματα και όλες τις διαστάσεις: έτσι επισκεπτόμενοι ένα και μόνο κόμβο σε ένα τέτοιο διάστημα, ο αλγόριθμος καταμέτρησης είναι ικανός να ερευνήσει για την κατάσταση των αντίστοιχων θέσεων bit για όλα τα διανύσματα και για όλες τις διαστάσεις/μετρικές, συνεπώς καθιστώντας την εκτίμηση πολλαπλών μετρικών ανέξοδη (όσον αφορά τον αριθμό βημάτων).

²Ένα παρόμοιο συμπέρασμα υπονοείται επίσης και από την ανάλυση του [33].

³Σημειώνουμε ότι $\log \frac{I}{\beta} < \log I \leq \lambda \leq L = \log N$.

Αλγόριθμος 4.6 Αλγόριθμος *estimateBBP*: Εκτίμηση του αριθμού διακριτών στοιχείων σε ένα πολυσύνολο με DHS με έλεγχο θέσεων δυαδικής αναζήτησης

```

1 Algorithm estimateBBP(Metric metric_id) {
2   HashSketch hashSketch;
3   int l = 0, overallLow = 0, h = λ, overallHigh = λ;
4   // Search for the lowest all-0 bit in [0,λ)
5   for (int bit = (l + h) / 2; l < h; bit = (l + h) / 2) {
6     Node targetNode = this.mapBitToNode(bit);
7     Result bitData = targetNode.processSCB(new SCBMessage(metric_id));
8     hashSketch.merge(bitData);
9     if (hashSketch.bitIsZeroForAllVectors(bit))
10      h = bit;
11     else
12      l = bit + 1;
13   }
14   overallHigh = l; // Bits above this are all 0
15   // Search for the highest all-1 bit in [0,overallHigh)
16   l = 0; h = overallHigh;
17   for (int bit = (l + h) / 2; l < h; bit = (l + h) / 2) {
18     if (hashSketch.hasInfoForBit(bit) == false) { // Skip the probe if we
19       have already done it
20     }
21     Node targetNode = this.mapBitToNode(bit);
22     Result bitData = targetNode.processSCB(new SCBMessage(metric_id));
23     hashSketch.merge(bitData);
24     if (hashSketch.bitIsOneForAllVectors(bit))
25       l = bit + 1;
26     else
27       h = bit;
28   }
29   overallLow = l; // Bits below this are all 1
30   // Revert to the standard procedure for [overallLow,overallHigh)
31   for (int bit = overallLow + 1; bit < overallHigh; bit++) {
32     if (hashSketch.hasInfoForBit(bit) == false) { // Skip the probe if we
33       have already done it
34     }
35     Node targetNode = this.mapBitToNode(bit);
36     Result bitData = targetNode.processSCB(new SCBMessage(metric_id));
37     hashSketch.merge(bitData);
38   }
39   int R[β] = hashSketch.computeRValues();
40   return  $\tilde{\alpha}_\beta \cdot \beta_0 \cdot 2^{\frac{1}{\beta_0}} \cdot \sum^* (R[i])$ 
41 }
```

4.5.3 Λανθάνουσα Αποθήκευση Bits

Από τον σχεδιασμό η επιλογή του σχήματος ελέγχου θέσεων είναι ορθογώνια ως προς τον μηχανισμό που δρομολογεί τα μηνύματα· αυτό σημαίνει ότι μπορούμε να έχουμε και επαναληπτικές αλλά και αναδρομικές εκδοχές των αλγορίθμων σειριακού και δυαδικού ελέγχου, πιθανότατα προσθέτοντας στο σκληνικό και συντομεύσεις δρομολόγησης, με διαφορετικά χαρακτηριστικά και αντιπραγματισμούς για κάθε συνδυασμό (περισσότερα για αυτό στην παράγραφο για την αξιολόγηση της απόδοσης). Ένας αρκετά ενδιαφέρον συνδυασμός είναι αυτός του σειριακού ελέγχου θέσεων με την αναδρομική δρομολόγηση bits, όπως έχει παρουσιαστεί στην ενότητα 4.2. Αυτό το σχήμα προσδίδει στις DHS μία απλή αλλά πολύ χρήσιμη δυνατότητα: κρυφή αποθήκευση (caching) των bits του αποτελέσματος του ερωτήματος στο ανάστροφο μονοπάτι αναδρομής (ονομάζουμε την δυνατότητα αυτή Bit Caching ή BC). Θέματα χαμηλού επιπέδου υλοποίησης – όπως η στρατηγική αντικατάστασης στην cache, ο χρόνος ζωής ενός αντικειμένου κ.λ.π. – θεωρούνται ορθογώνια για μας και γι' αυτό αυτή η εργασία δεν ασχολείται μαζί τους. Αυτό που έχει πραγματική σημασία, είναι το γεγονός ότι με το πέρασμα του χρόνου όλο και λιγότερες διαδικασίες καταμέτρησης θα πρέπει να διατρέχουν όλες τις θέσεις από το bit 0 προς το bit που περιέχει την τελευταία ενδιαφέρουσα θέση bit για τον εκτιμητή που χρησιμοποιούμε την δεδομένη στιγμή. Επιπρόσθετα, καθώς κάτω στο σχήμα SBP+RBR όλα τα ερωτήματα αρχίζουν από το bit στην θέση 0 και προχωρούν από το ένα διάστημα αναγνωριστικών στο επόμενο με αναδρομική μορφή, η πιθανότητα επιτυχίας στην cache σε κάθε βήμα αυξάνει καθώς το μέγεθος των διαστημάτων αναγνωριστικών μειώνεται. Αυτή η προσθήκη έχει ευεργετικά αποτελέσματα στο κόστος αριθμού βημάτων των ερωτημάτων, καθώς επίσης και στην εξισορρόπηση του φορτίου ανάμεσα στους κόμβους· χρησιμοποιώντας caching το φορτίο των ερωτημάτων μεταφέρεται από τους λίγους κόμβους που αποτελούν τα διαστήματα των bits σε υψηλές θέσεις στους υπόλοιπους κόμβους του δικτύου επικάλυψης. Στην παράγραφο της αξιολόγησης της απόδοσης θα παρουσιάσουμε αυτή την προσθήκη και τα σαφώς θετικά της αποτελέσματα στα χαρακτηριστικά του συστήματος.

4.5.4 Λάθη και Επαναλήψεις

Τα λάθη στην εκτίμηση που επιστρέφονται από τον αλγόριθμο καταμέτρησης των DHS οφείλονται σε: (α) στατιστικές αποκλίσεις εκ μέρους του θεωρητικού υποστρώματος της σύνοψης κατακερματισμού και (β) bits τα οποία δεν γίνονται set κατά την διάρκεια της έρευνας των κόμβων από τον αλγόριθμο καταμέτρησης. Όσον αφορά τις συνόψεις κατακερματισμού, τα [33,35] παρέχουν μία ενδελεχή ανάλυση των στατιστικών χαρακτηριστικών τους. Το να απαριθμήσουμε τις αποδείξεις που βρίσκονται σε αυτές τις εργασίες είναι πέρα από την εμβέλεια αυτής της διατριβής. Οι αναγνώστες που ενδιαφέρονται μπορούν να αναφερθούν στις [33,35] και απλά σημειώνουμε εδώ ότι η δεδομένη απόκλιση προσεγγίζει το $\frac{1.05}{\sqrt{\beta}}$ για την [33] και το $\frac{0.78}{\sqrt{\beta}}$ για την [35].

Όσον αφορά τη δεύτερη περίπτωση, αφού κάθε θέση bit των διανυσμάτων είναι ομοιόμορφα κατανομημένη σε ένα διάστημα του συνόλου των αναγνωριστικών των κόμβων, ίσως θα πρέπει να επισκεφτούμε πολλαπλούς κόμβους σε κάθε διάστημα μέχρις ότου να εντοπίσουμε μία αποθηκευμένη πληροφορία για ένα στοιχείο (που ανταποκρίνεται στο ότι ένα bit είναι set). Όσον αφορά αυτό, ο αλγόριθμος καταμέτρησης των DHS πρώτα επιλέγει έναν τυχαίο κόμβο στο διάστημα αναγνωριστικών που εξετάζουμε και ερευνά για οποιαδήποτε σχετική DHS πλειάδα. Αν μια τέτοιου είδους πληροφορία δεν είναι διαθέσιμη σε αυτό τον κόμβο, ο αλγόριθμος συνεχίζει και επισκέπτεται τους προηγούμενους/επόμενους κόμβους από αυτόν που εξετάσαμε μέσα στο συγκεκριμένο διάστημα αναγνωριστικών μέχρις ότου είτε να εντοπιστεί κάποια σχετική πλειάδα, είτε να φτάσει κάποιο άνω όριο προσπαθειών. Αυτή η επαναληπτική φάση υπάρχει με σκοπό να αντισταθμίσει το εξής: όταν καταγράψουμε I στοιχεία σε ένα διάστημα αναγνωριστικών το οποίο αντιστοιχεί σε N ή περισσότερους κόμβους τότε θα υπάρχουν κόμβοι οι οποίοι δεν θα έχουν αποθηκευμένες

σχετική πληροφορία· ακόμα και όταν το εξεταζόμενο διάστημα αποτελείται από αριθμό κόμβων μικρότερο του I , κάποιοι από αυτούς ίσως δεν έχουν αποθηκευμένες σχετική πληροφορία, εξαιτίας της τυχαίας επιλογής του εξεταζόμενου κόμβου (και όταν αποθηκεύουμε και όταν ανακτούμε πληροφορίες σχετικές με τις DHS). Για τους δικούς μας αλγόριθμους αυτό σημαίνει ότι όταν επισκεπτόμαστε τυχαία ένα κόμβο που κρατάει κάποιο bit, αν αυτό είναι μηδενικό, δεν μπορούμε να είμαστε ακόμα σίγουροι, έτσι πρέπει να ξαναπροσπαθήσουμε μέχρι να βρούμε ένα bit που να είναι set. Το ερώτημα είναι πόσες φορές πρέπει να ξαναπροσπαθήσουμε, πριν να σταματήσουμε, ώστε να μη σφάλουμε με ελεγχόμενη πιθανότητα.

Στρεφόμαστε τώρα στον υπολογισμό του άνω ορίου των επαναλήψεων ανά θέση bit των διανυσμάτων της DHS. Ας υποθέσουμε ότι I' αντικείμενα έχουν καταταξιωθεί ομοιόμορφα σε N' κάδους (π.χ. αντιστοιχήθηκαν σε ένα διάστημα N' κόμβων στην DHS). Η διαδικασία καταμέτρησης της προηγούμενης ενότητας αντιστοιχεί σε ομοιόμορφη και ανεξάρτητη επιλογή ενός κάδου από το σύνολο κάδων χωρίς επανάθεση και έλεγχο για το αν ο επιλεγμένος κάδος περιέχει κάποιο αντικείμενο. Η πιθανότητα $P(X = t)$ t άδειοι κάδοι να επιλεγούν στους πρώτους t ελέγχους, ισούται με:

$$P(X = t) = \left(\frac{N' - t}{N'} \right)^{I'} . \quad (4.1)$$

Απόδειξη: 'Όταν αναθέτουμε ένα αντικείμενο σε έναν από N' κάδους με ομοιόμορφο τρόπο, η πιθανότητα να επιλέξουμε κάποιον συγκεκριμένο κάδο είναι $\frac{1}{N'}$ και η πιθανότητα να μην τον επιλέξουμε είναι $\frac{N'-1}{N'}$. Επομένως, αφού τοποθετήσουμε I' αντικείμενα, ένας κάδος θα είναι άδειος με πιθανότητα $\left(\frac{N'-1}{N'} \right)^{I'}$. Αυτό επίσης καθορίζει την πιθανότητα να επιλέξουμε έναν άδειο κάδο στην πρώτη δοκιμή. Τώρα, η πιθανότητα ένας από τους εναπομείναντες $N' - 1$ κάδους να είναι άδειος (και η πιθανότητα να επιλέξουμε έναν άδειο κάδο στην δεύτερη προσπάθειά μας), είναι $\left(\frac{N'-2}{N'-1} \right)^{I'}$, δεδομένου ότι η πρώτη δοκιμή μας κατέληξε σε άδειο κάδο. Σημειώνουμε ότι το να επιλέξουμε τον αμέσως επόμενο κάδο από αυτόν που επιλέξαμε στο προηγούμενο βήμα είναι ισοδύναμο με το να επιλέγουμε κάποιον από τους $N' - 1$ κάδους με ομοιόμορφη πιθανότητα, αφού τα αντικείμενα τοποθετήθηκαν στους κάδους με ομοιόμορφη κατανομή. Στην t -οστή δοκιμή, η πιθανότητα να επιλέξουμε έναν άδειο κάδο θα είναι $\left(\frac{N'-t}{N'-t-1} \right)^{I'}$. Αφού κάθε επανάληψη είναι ανεξάρτητη από τις υπόλοιπες η πιθανότητα να επιλέξουμε t άδειους κάδους στις πρώτες t επαναλήψεις είναι: $\left(\frac{N'-1}{N'} \right)^{I'} \cdots \left(\frac{N'-t+1}{N'-t+2} \right)^{I'} \cdot \left(\frac{N'-t}{N'-t+1} \right)^{I'} = \left(\frac{1}{N'} \right)^{I'} \cdot \left(\frac{(N'-1) \cdot (N'-2) \cdots (N'-t+1)}{(N'-1) \cdot (N'-2) \cdots (N'-t+1)} \right)^{I'} \cdot (N' - t)^{I'} = \left(\frac{N'-t}{N'} \right)^{I'} = P(X = t)$. $\diamond$

Λύνοντας την εξίσωση 4.1 ως προς t , παίρνουμε ότι, για να επιλέξουμε ένα μη κενό κάδο με πιθανότητα τουλάχιστον p , πρέπει να κάνουμε τουλάχιστον: $t \leq \tau = \lceil N' \cdot (1 - p^{\frac{1}{I'}}) \rceil$ κάδους/κόμβους. Θέτοντας $\xi = \frac{I'}{N'}$, παίρνουμε: $\tau = \lceil N' \cdot (1 - p^{\frac{1}{\xi \cdot N'}}) \rceil$. 'Όταν χρησιμοποιούμε πολλαπλά (β) διανύσματα, τα αντικείμενα διαμοιράζονται στα διανύσματα αυτά, με αποτέλεσμα $\frac{I'}{\beta}$ αντικείμενα να εισάγονται στους N' κάδους και επομένως η τελευταία εξίσωση γίνεται: $\tau_\beta = \lceil N' \cdot (1 - p^{\frac{\beta}{\xi \cdot N'}}) \rceil$. Τέλος, αν πάρουμε και τα αντίγραφα υπόψιν και θεωρώντας έναν βαθμό αντιγραφής α , παίρνουμε:

$$\tau_\beta^\alpha = \lceil N' \cdot (1 - p^{\frac{\beta}{\alpha \cdot \xi \cdot N'}}) \rceil . \quad (4.2)$$

Σημειώνουμε και πάλι ότι N' είναι ο αριθμός των κόμβων που είναι υπεύθυνοι για και μόνο θέση ενός και μόνο διανύσματος (δηλ. ανήκουν στο ίδιο διάστημα του χώρου αναγνωριστικών), I' είναι το πλήθος των αντικειμένων που αντιστοιχίζονται στο διάστημα αυτό και ξ είναι ο λόγος των δύο αυτών ποσοτήτων. Αυτό σημαίνει ότι υπάρχει ένα διαφορετικό βέλτιστο τ_β για κάθε διάστημα του χώρου αναγνωριστικών, με μικρότερου μεγέθους διαστήματα να έχουν μικρότερες τιμές για το

τ_β (δηλ. τα διαστήματα που αντιστοιχούν στο λιγότερο σημαντικό bit των διανυσμάτων, θα έχουν την μεγαλύτερη τιμή για το τ_β).

Η προκαθορισμένη τιμή του τ_β που χρησιμοποιούμε στις DHS για όλα τα διαστήματα, είναι 5 (σταθερή) και επαρκεί για να εγγυηθεί ότι, αν υπάρχει ένας μη κενός κόμβος σε οποιοδήποτε διάστημα, θα τον βρούμε με πιθανότητα τουλάχιστον 99% όταν ο αριθμός των αντικειμένων που αντιστοιχήθηκαν στο συγκεκριμένο διάστημα του χώρου αναγνωριστικών είναι μεγαλύτερος ή ίσος του αριθμού των κόμβων στο διάστημα (δηλ. $I \geq \beta \cdot N$). Προφανώς, η προκαθορισμένη τιμή αυτή επαρκεί και για περιπτώσεις που μετρούμε μεγαλύτερους πληθάρθιμους από τον παραπάνω. Ωστόσο, όταν μετρούμε μικρότερους πληθάρθιμους, μπορούμε να επιλέξουμε είτε (α) να αυξήσουμε την τιμή του τ_β , σύμφωνα με την εξίσωση 4.2, είτε (β) να χρησιμοποιήσουμε ένα μικρότερο DHT/DHS δίκτυο επικάλυψης για την συγκεκριμένη λειτουργία, ακολουθώντας το σκεπτικό των super-nodes των υβριδικών ΔΟΕ – μία τάση που τελευταίως έχει αρχίσει να κερδίζει πολλούς υποστηρικτές και στον κόσμο των δομημένων δικτύων, είτε (γ) να χρησιμοποιήσουμε αντίγραφα της πληροφορίας της DHS, όπως περιγράψαμε στην ενότητα 4.3.4. Επομένως, το κόστος αριθμού βημάτων χειρότερης περίπτωσης γίνεται $O(\lambda \cdot (\log N + \tau))$, όπου τ είναι το άνω όριο του αριθμού επαναλήψεων κατά την φάση ελέγχου θέσεων. Για σταθερό τ , όπως χρησιμοποιούμε στις DHS, η πολυπλοκότητα αριθμού βημάτων γίνεται και πάλι $O(\lambda \cdot \log N)$. Σημειώνουμε ότι το κόστος της καταμέτρησης είναι σε κάθε περίπτωση ανεξάρτητο του αριθμού των διανυσμάτων.

4.6 Κάτω Φράγματα

Είναι κοινή πρακτική ανάμεσα στις εργασίες που εκτιμούν join sizes [43, 79, 80] και ροπές συχνότητας (frequency moments) εν γένει [6], να θεωρούν ένα λογικό (κάτω) όριο όσον αφορά την θεμελιώδη οντότητα που εκτιμάται. Αυτό πηγάζει κυρίως από το γεγονός ότι το να εκτιμάς μικρές ποσότητες είναι πολύ πιο δύσκολο από την εκτίμηση μεγάλων ποσοτήτων και τα τελικά αποτελέσματα είναι σημαντικά βελτιωμένα όσον αφορά τα όρια λάθους στην εκτίμηση. Μπορούμε και εμείς να θεωρήσουμε ένα τέτοιο κάτω όριο στην διάταξη μας και στην πραγματικότητα να το ενσωματώσουμε στην αντιστοίχιση της θέσης bit με τον ακόλουθο τρόπο. Έστω ότι θέτουμε ένα κάτω όριο το C_{LB} στον πληθάρθιμο του πολυσυνόλου εισόδου. Βασίζόμενοι στην εξίσωση 2.3, ενδιαφερόμαστε μόνο για τις θέσεις των bit πάνω από την $\lambda_0 = \log\left(\frac{C_{LB}}{\beta}\right) - 3$. Κατ' αυτόν τον τρόπο αλλάζουμε την αντιστοίχιση από θέσεις bit σε διαστήματα του χώρου αναγνωριστικών ώστε αυτή η θέση bit να αντιστοιχίζεται στο πρώτο μισό του δακτυλίου (όπως στην περίπτωση του bit στην θέση 0 στην κανονική αντιστοίχιση) και τα bits που βρίσκονται χαμηλότερα από αυτή την θέση (λιγότερο σημαντικά) αγνοούνται. Δηλαδή έχουμε σαν αποτέλεσμα την αντιστοίχιση του i -οστού DHT διαστήματος στο $(\lambda_0 + i)$ -οστό bit. Καθώς τα λιγότερο συχνά bits αντιστοιχίζονται σε μεγαλύτερα διαστήματα, αυτή η τεχνική απαιτεί είτε συνύπαρξη με την τεχνική αντιγράφων της παραγράφου 4.3.4, είτε ένα υψηλότερο όριο επαναλήψεων (βλπ. ενότητα 4.5.4). Από την άλλη μεριά υπάρχει το πλεονέκτημα της καλύτερης κατανομής φορτίου, εξαιτίας της αποφόρτισης των υψηλότερων (αν και λιγότερο συχνά επισκεπτόμενων) θέσεων bit σε μεγαλύτερα DHT διαστήματα και της μείωσης του κόστους αριθμού βημάτων εισαγωγής και/ή ερωτημάτων αφού λιγότερα bits πρέπει να γίνουν set ή να ερευνηθούν.

4.7 Επιλογή Μήκους Πινάκων και Συνάρτησης Διασποράς

Όπως έχει αναφερθεί και προηγουμένως (ενότητα 2.4.2), οι συνόψεις κατακερματισμού αποτελούνται από μία συνάρτηση κατακερματισμού και ένα διάνυσμα από bits που καταγράφει την έξοδο της προγενέστερης, αφού έχει επεξεργαστεί με την συνάρτηση $\rho(\cdot)$, για κάθε στοιχείο εισό-

Αλγόριθμος 4.7 Εκτίμηση του Αριθμού Διακριτών Στοιχείων σε ένα Πολυσύνολο με Χρήση DHS.
Τα β , λ και τ περιγράφονται στον πίνακα 4.1

```

1  Algorithm estimate() {
2      /*  $R[]$ : bit positions of the most significant 1-bits. */
3      int  $R[\beta] = \{-1, -1, \dots, -1\}$ ; bool allBitmapsSet = false;
4
5      for (int bit =  $\lambda - 1$ ; bit > 0 && !allBitmapsSet; bit--) {
6          ID randomID = this.computeRandomIdForBit(bit);
7          Node initialNode = \DHT{\}.lookup(randomID);
8          Node currentNode = initialNode;
9          bool goingCW = true;
10         for (int retry = 0; retry <  $\tau$  && !allBitmapsSet && currentNode != null;
11             retry++) {
12             allBitmapsSet = true;
13             for (int bitmap = 0; bitmap <  $\beta$ ; bitmap++)
14                 if ( $R[\text{bitmap}] < 0$ ) {
15                     if (currentNode.storesBitDataFor(bitmap, bit))
16                          $R[\text{bitmap}] = \text{bit}$ ;
17                     else
18                         allBitmapsSet = false;
19                 }
20             if (!allBitmapsSet) {
21                 if (goingCW)
22                     if (currentNode.successor.isInArcForBit(bit))
23                         currentNode = currentNode.successor;
24                     else {
25                         goingCW = false;
26                         currentNode = initialNode;
27                     }
28             }
29             if (!goingCW) {
30                 if (currentNode.predecessor.isInArcForBit(bit))
31                     currentNode = currentNode.predecessor;
32                 else
33                     currentNode = null;
34             }
35         }
36     }
37     return  $\tilde{\alpha}_\beta \cdot \beta_0 \cdot 2^{\frac{1}{\beta_0}} \cdot \sum^*(R[\cdot])$ 
38 }

```

δου. Για το πρώτο μέρος μπορούμε να χρησιμοποιήσουμε ή κάποια από τις πολλές κρυπτογραφικές συναρτήσεις κατακερματισμού της βιβλιογραφίας (πχ MD-4, MD-5, SHA-1, RMD-160, κ.λ.π.) ή να μεταχειριστούμε απευθείας οποιαδήποτε συνάρτηση χρησιμοποιεί το υπόστρωμα του DHT για να παράγει αναγνωριστικά κόμβων και αντικειμένων. Ας υποθέσουμε ότι η έξοδος της συνάρτησης κατακερματισμού έχει μήκος L bits: τότε όλα τα bits εξόδου της συνάρτησης κατακερματισμού είναι ψευδοτυχαία αφού αυτό είναι ένα θεμελιώδες αξίωμα κάθε τυχαίας/κρυπτογραφικής συνάρτησης κατακερματισμού. Αυτό επίσης εκπληρώνει την βασική προϋπόθεση του [33] και (κυρίως) του [35] σχετικά με συναρτήσεις κατακερματισμού που χρησιμοποιούνται για την εισαγωγή στοιχείων στις συνόψεις κατακερματισμού. Δεδομένου ότι το μήκος λ του διανύσματος της σύνοψης κατακερματισμού πρέπει να είναι μικρότερο από L bits, μπορούμε να χρησιμοποιήσουμε ένα μέρος της εξόδου από την συνάρτηση κατακερματισμού που επιλέξαμε για να εισάγουμε στοιχεία στην κατανεμημένη σύνοψη κατακερματισμού. Για το υπόλοιπο αυτού του μέρους, θα υποθέσουμε ότι θα χρησιμοποιούμε τα χαμηλότερα λ bits της εξόδου της συνάρτησης κατακερματισμού που μας παρέχεται από την υλοποίηση του υποστρώματος του ΔΟΕ DHT.

Από την άλλη, το μήκος λ των διανυσμάτων είναι ένα θέμα συζήτησης. Καθώς έχουμε να κάνουμε με κατανεμημένα, αυτόνομα συστήματα, πρέπει να είναι σταθερό την χρονική στιγμή που αναπτύσσουμε την κατανεμημένη σύνοψη κατακερματισμού. Όμως, το λ είναι μία συγκεκριμένη για την εφαρμογή παράμετρος η τιμή της οποίας εξαρτάται από την χρησιμοποιούμενη συνάρτηση κατακερματισμού και το μέγιστο πληθάριθμο που θα εκτιμηθεί. Ειδικότερα, το λ πρέπει να είναι μικρότερο από την έξοδο της συνάρτησης κατακερματισμού που χρησιμοποιούμε μείον τον λογάριθμο του πλήθους των διανυσμάτων της σύνοψης κατακερματισμού και μεγαλύτερο από το στρογγυλοποιημένο προς τα άνω άθροισμα του 3 συν τον λογάριθμο του πληθικού του πληθάριθμου I που θα εκτιμηθεί δια τον αριθμό των διανυσμάτων στην σύνοψη κατακερματισμού (βλέπε εξίσωση 2.3). Για παράδειγμα, για να υπολογίσουμε τον πληθάριθμο ενός πολυσυνόλου που περιλαμβάνει ένα εκατομμύριο διακριτά στοιχεία με μία σύνοψη κατακερματισμού ενός διανύσματος, θα χρειαζόμαστε τουλάχιστον 23 bits, ενώ για ένα τρισεκατομμύριο διακριτών στοιχείων θα χρειαζόμαστε 43 bits ανά διάνυσμα· για 512 διανύσματα στην σύνοψη κατακερματισμού, αυτά τα νούμερα μειώνονται σε 14 και 34 bits αντίστοιχα. Επιπροσθέτως, σημειώνουμε ότι μία τιμή του λ (ακόμα και κατά πολύ) μεγαλύτερη από το ελάχιστο που δίνεται από την εξίσωση 2.3 δεν έχει αρνητικό αποτέλεσμα στην ακρίβεια της υπολογιζόμενης εκτίμησης (όλο κι' όλο το αποτέλεσμα θα είναι κάποια υψηλά bits ποτέ να μην πάρουν τιμή 1) και γι' αυτό, λόγω της αντιστοίχησης θέσεων bit σε αναγνωριστικά κόμβων, δεν είναι απαραίτητο να επιλέξουμε μία τιμή του λ κοντά στην παραπάνω. Επιπροσθέτως, με L -bit κλειδιά στο DHT και 2^L πιθανές τιμές αναγνωριστικών, το Birthday Paradox μειώνει τον αριθμό των στοιχείων σε κάθε χώρο DHT – και ως εκ τούτου το μέγιστο εκτιμώμενο πληθάριθμο – σε $2^{\frac{L}{2}}$. Για παράδειγμα τα περισσότερα δίκτυα επικάλυψης DHT χρησιμοποιούν μία 160-bit συνάρτηση κατακερματισμού (συνήθως την SHA-1), έτσι το λ δεν θα υπερβεί ποτέ τα 83 bits, πέφτοντας στα 76 bits για 512 διανύσματα στην σύνοψη κατακερματισμού.

Η παραπάνω ανάλυση οδηγεί στον ακόλουθο απλό αλγόριθμο: αν κατά την διάρκεια του σχεδιασμού/ανάπτυξης της εφαρμογής υπάρχει κάποια – έστω και αόριστη – ιδέα σχετικά με τον μέγιστο πληθάριθμο που θα εκτιμηθεί από την δεδομένη σύνοψη κατακερματισμού κατά την προκειμένη περίπτωση (ακόμα και μέσα σε μερικές δυαδικές τάξεις μεγέθους) και έχουμε καταλήξει στον αριθμό των διανυσμάτων στην σύνοψη κατακερματισμού, μπορούμε να υπολογίσουμε το λ από την εξίσωση 2.3 όπως στο πρώτο παραπάνω παράδειγμα· ή μπορούμε να χρησιμοποιήσουμε το άνω όριο του I όπως υποδηλώνεται από το Birthday Paradox και να υπολογίσουμε το λ όπως στο δεύτερο παράδειγμα.

Παράμετροι Σύνοψης Κατακερματισμού	⇒ Συνάρτηση Κατακερματισμού	($h(\cdot)$)
	⇒ Αριθμός Διανυσμάτων	(β)
	⇒ Αριθμός Bits ανά Διάνυσμα	(λ)
	⇒ Εκτιμητής PCSA vs. superLogLog	
Δρομολόγηση	⇒ Επαναληπτική vs. Αναδρομική Δρομολόγηση Θέσεων	(IBR/RBR)
	⇒ Συντομεύσεις Δρομολόγησης	(RS)
Εισαγωγή/Ανανέωση	⇒ Μοναδικού Αντικειμένου vs. Μαζικές Εισαγωγές	(SI/BI)
	⇒ Αντιγραφή στην Γειτονιά Προορισμού	(TNR)
	⇒ Απόκρυψη στην Διαχειριστική Κίνηση	(MTP)
Καταμέτρηση	⇒ Σειριακός vs. Δυναδικός Έλεγχος Θέσεων	(SBP/BBP)
	⇒ Κρυφή Αποθήκευση Bits Αποτελεσμάτων	(BC)
	⇒ Όριο Επαναλήψεων	(τ)

Πίνακας 4.2: Σύνοψη Σχεδιαστικών Επιλογών: Παράμετροι Ρυθμίσεων των DHS

4.8 Σύνοψη Σχεδιαστικών Επιλογών

Σε αυτή την ενότητα περιγράψαμε τις Κατανεμημένες Συνόψεις Κατακερματισμού, την δική μας προσέγγιση για να κατανειμούμε τα δεδομένα και της διάφορες λειτουργίες των υπολογισμών μίας σύνοψης κατακερματισμού πάνω από ένα δίκτυο επικάλυψης ΔΟΕ κατανεμημένων πινάκων κατακερματισμού με κλιμακώσιμο και αποδοτικό τρόπο. Η αρχιτεκτονική μας προσφέρει ένα πλήθος από επιλογές και εναλλακτικές για ένα σχεδιαστή συστήματος ώστε να το προσαρμόσει στις ανάγκες του όταν αναπτύσσει ένα στιγμιότυπο των DHS. Πρώτα υπάρχουν παράμετροι σχετικές με την βάση της σύνοψης κατακερματισμού: η συνάρτηση κατακερματισμού, ο αριθμός και το μήκος σε bits των διανυσμάτων και ο ακριβής αλγόριθμος εκτίμησης (superLogLog vs PCSA) . Έπειτα κάποιος μπορεί να επιλέξει ανάμεσα σε δύο πρωτόκολλα δρομολόγησης bit – ειδικότερα, επαναληπτική και αναδρομική δρομολόγηση bit – και κατά πόσο να χρησιμοποιήσει συντομεύσεις στην δρομολόγηση ή όχι. Όσον αφορά τις εισαγωγές και τις ανανεώσεις των στοιχείων, υπάρχει η επιλογή μεμονωμένων ή μαζικών εισαγωγών, ενώ κάποιος μπορεί επιλεκτικά να δημιουργήσει αντίγραφα στην γειτονία προορισμού ή/και να αποκρύψει τα μηνύματα αυτά στην διαχειριστική κίνηση του ΔΟΕ DHT για να μειώσει περαιτέρω το συνολικό κόστος διατήρησης. Όσον αφορά το κομμάτι της καταμέτρησης, υπάρχουν επίσης δύο υποψήφιοι αλγόριθμοι έρευνας bit: σειριακός και δυναδικός έλεγχος θέσεων· υπάρχει επίσης η δυνατότητα του caching των bits των αποτελεσμάτων της εκτίμησης (σε συνδυασμό με αναδρομική δρομολόγηση) και η παράμετρος για το όριο των επαναλήψεων. Ο πίνακας 4.2 συνοψίζει το τελικό χώρο σχεδιαστικών επιλογών.

4.9 Κατανεμημένη Επεξεργασία Συναθροιστικών Ερωτημάτων

Το κεφάλαιο αυτό ασχολείται με την εκμετάλλευση της ισχύος των Κατανεμημένων Συνόψεων Κατακερματισμού, με σκοπό την υποστήριξη ενός πλήθους βασικών συναθροιστικών ερωτημάτων σε ένα σύστημα διαχείρισης δεδομένων Διαδικτυακής κλίμακας βασισμένο σε ΔΟΕ. Εκτός κι αν αναφέρουμε κάτι διαφορετικό, χρησιμοποιούμε τον όρο «κατανεμημένη σύνοψη» για να αναφερθούμε στην κατανεμημένη έκδοση μίας σύνοψης κατακερματισμού, είτε είναι βασισμένη σε προσέγγιση ραντεβού είτε στην αρχιτεκτονική DHS. Έτσι, πρώτα θα δείξουμε πως μπορούν να υπολογιστούν πολλά συναθροιστικά ερωτήματα, όπως τα Count, Count-Distinct, Sum και Average, με αποκεντρωποιημένο τρόπο· κατόπιν, στο επόμενο κεφάλαιο θα συζητήσουμε τεχνικές που χρησιμοποιούν αυτά τα εργαλεία ώστε να κατασκευάσουν και να διατηρήσουν ιστογράμματα πάνω από τα αποθηκευμένα δεδομένα [98].

4.9.1 Count-Distinct

Τόσο οι συνόψεις κατακερματισμού με βάση την προσέγγιση ραντεβού όσο και το DHS μπορούν να εφαρμοστούν άμεσα για την εκτίμηση του αριθμού των διακριτών αντικειμένων ενός πολυσυνόλου. Ας υποθέσουμε ότι θέλουμε να εκτιμήσουμε τον αριθμό των διακριτών τιμών σε μία στήλη C μίας σχέσης R αποθηκευμένης στο διαδικτυακής κλίμακας σύστημά μας διαχείρισης δεδομένων. Στο σενάριο της προσέγγισης ραντεβού ο αλγόριθμος προχωρά ως εξής: (α) οι κόμβοι που αποθηκεύουν πλειάδες της R πρώτα υπολογίζουν το αναγνωριστικό του ραντεβού χρησιμοποιώντας τη συνάρτηση κατακερματισμού του DHT με είσοδο τα αναγνωριστικά των R και C . (β) κάθε τέτοιος κόμβος προσθέτει τις σχετικές τιμές που αποθηκεύει σε μία τοπικά συμπληρούμενη σύνοψη και (γ) στέλνει την σύνοψη αυτή στον κόμβο που είναι υπεύθυνος για το αναγνωριστικό του ραντεβού. Ο τελευταίος είναι τότε υπεύθυνος για να συγκεντρώσει όλες τις συνόψεις και να ανασκευάσει (μέσω bitwise OR) την συνολική σύνοψη. Οποιοσδήποτε κόμβος θελήσει στη συνέχεια να υπολογίσει τον πληθύνον των πλειάδων της $R.C$ μπορεί να ρωτήσει τον κόμβο ραντεβού για την συνολική εκτίμηση (βλπ. Σχήμα 4.2). Ο αριθμός βημάτων στην χειρότερη περίπτωση για να υπολογιστεί η συνολική σύνοψη είναι $O(N \log N)$ σε ένα δίκτυο N κόμβων – ή $O(b \cdot N \log N)$ χρήσης εύρους ζώνης δικτύου, με b bytes ανά τοπική σύνοψη – ενώ η ερώτηση για την συνολική σύνοψη απαιτεί $O(\log(N))$ βήματα.

Παρατηρείστε ότι, όταν καταμετρούμε διακριτά αντικείμενα, μία λύση σύνοψης βασισμένης στην προσέγγιση ραντεβού είναι πολύ αποδοτικότερη από την απλή προσέγγιση του να στέλνουν οι κόμβοι λίστες από ζεύγη τιμής/αναγνωριστικού πλειάδας για τις πλειάδες που αποθηκεύουν στον κόμβο ραντεβού· ο κόμβος ραντεβού θα χρειαζόταν χώρο ανάλογο της εκτιμώμενης ποσότητας αφού θα έπρεπε να διατηρεί πλήρη κατάλογο των ζευγών που θα λάμβανε για να μπορεί να διακρίνει διπλότυπα, ενώ οι συνόψεις κατακερματισμού μειώνουν σημαντικά την απαίτηση σε χώρο (περίπου στην τάξη του λογαρίθμου της εκτιμώμενης ποσότητας), ενώ κάτι αντίστοιχο ισχύει και για την κατανάλωση εύρους ζώνης δικτύου για την μεταφορά των λιστών των ζευγών αυτών στον κόμβο ραντεβού.

Στην περίπτωση των DHS, οι κόμβοι που αποθηκεύουν πλειάδες της R τις εισάγουν στην DHS ως εξής:

1. εισάγουν κάθε σχετική πλειάδα σε μία τοπική σύνοψη χρησιμοποιώντας την τιμή της πλειάδας στην στήλη C ($C.v_i$), όπως περιγράφηκε στην ενότητα 4,
2. επιλέγουν ένα τυχαίο αναγνωριστικό στο διάστημα που αντιστοιχεί σε κάθε bit που θα πρέπει να τεθεί στην τιμή 1 κατά το προηγούμενο βήμα και
3. στέλνουν ένα μήνυμα «set-to-1» στους κόμβους που είναι υπεύθυνοι για τα αναγνωριστικά αυτά (βλπ. Σχήμα 4.5).

Ας υποθέσουμε ότι το DHS χρησιμοποιεί L bits ανά πίνακα. Τότε, ο αριθμός μηνυμάτων χειρότερης περίπτωσης για να συμπληρωθεί η DHS είναι $O(N \cdot L \cdot \log(N))$, το οποίο μεταφράζεται σε συνολική κατανάλωση εύρους ζώνης δικτύου $O(b' \cdot N \cdot L \cdot \log(N))$, με b' bytes ανά «set-to-1» μήνυμα. Υπό πραγματικές συνθήκες, ένα τέτοιο μήνυμα θα συμπεριλάμβανε απλά το αναγνωριστικό της μετρούμενης ποσότητας (δηλ. το «αναγνωριστικό μετρικής» και τον αριθμό του πίνακα και της θέσης του bit που πρέπει να τεθεί στην τιμή 1. Η ερώτηση της συμπληρωμένης DHS απαιτεί $O(L \cdot \log(N))$ μηνύματα στην χειρότερη περίπτωση.

Συγκρίνοντας τις δύο προσεγγίσεις, μπορούμε να δούμε ότι η συμπλήρωση της DHS απαιτεί έναν παράγοντα L περισσότερων μηνυμάτων και έναν παράγοντα L περισσότερου εύρους ζώνης⁴ στην χειρότερη περίπτωση, ενώ το ερώτημα της κατανομημένης σύνοψης απαιτεί επίσης L φορές περισσότερα μηνύματα στην χειρότερη περίπτωση, σε σχέση με την προσέγγιση ραντεβού. Ωστόσο,

⁴Διαιρούμενα με N για να βρεθούν τα αντίστοιχα κόστη ανά κόμβο.

σημειώνουμε ότι: (α) στην προσέγγιση ραντεβού, ο κόμβος ραντεβού λαμβάνει όλο τον φόρτο της καταμέτρησης και αποτελεί μοναδικό σημείο αποτυχίας και σημείο συσσωρευμένης καθυστέρησης, ενώ η προσέγγιση DHS είναι πλήρως αποκεντριοποιημένη και εξισορροπημένη όσον αφορά τον φόρτο των κόμβων και (β) όταν εκτιμούμε πολλαπλές ποσότητες ταυτόχρονα, το κόστος αριθμού βημάτων για την συμπλήρωση και την ερώτηση της DHS παραμένει το ίδιο με αυτό της περίπτωσης μίας ποσότητας, ενώ το αντίστοιχο κόστος για την προσέγγιση ραντεβού αυξάνει γραμμικά με το πλήθος των εκτιμώμενων ποσοτήτων.

4.9.2 Count

Για να καταμετρηθούν όλα τα αντικείμενα μίας στήλης *συμπεριλαμβανομένου των διπλοτύπων*, προχωρούμε ως άνω, προσθέτοντας τα αναγνωριστικά των πλειάδων στην αντίστοιχη σύνοψη αντί για την τιμή της κάθε πλειάδας. Επιπλέον, αν πανομοιότυπες πλειάδες που αποθηκεύονται σε διαφορετικούς κόμβους θα έπρεπε επίσης να καταμετρηθούν στο τελικό αποτέλεσμα, οι κόμβοι προσθέτουν και το δικό τους αναγνωριστικό στην είσοδο της συνάρτησης εισόδου στην κατανεμημένη σύνοψη.

4.9.3 Node Cardinality

Μία από τις μετρικές που χρησιμοποιήθηκαν από τον βελτιστοποιητή του συστήματος System-R και μία πολύ σημαντική μετρική για την εκτέλεση ερωτήματος σε πραγματικό σύστημα, είναι ο αριθμός των «σελίδων δεδομένων» ανά σχέση ή ανά index. Η μετρική αυτή καθορίζει τον αριθμό των προσβάσεων σε δευτερεύοντα αποθηκευτικό χώρο που θα πρέπει να κάνει ο επεξεργαστής ερωτημάτων κατά την πλήρη σάρωση της σχέσης/index. Προτείνουμε μία αναλογία ανάμεσα στους κόμβους ενός ΔΟΕ και τα blocks ενός δίσκου δευτερεύουσας αποθήκευσης. Στα παραδοσιακά, κεντριοποιημένα περιβάλλοντα, ο αριθμός των προσβάσεων στον δίσκο αποτελεί τον κυριότερο παράγοντα κόστους. Επιπλέον, είναι κοινώς γνωστό ότι η καθυστέρηση κατά την πρόσβαση στο δίσκο οφείλεται κυρίως στον χρόνο αναζήτησης (seek time)· η δρομολόγηση προς έναν κόμβο στο δίκτυο επικάλυψης αντιστοιχεί με μία αναζήτηση σε ένα υποσύστημα δίσκων, ενώ η πρόσβαση αντικειμένων σε έναν κόμβο αντιστοιχεί με μία πρόσβαση σε ένα block δίσκου. Για την ακρίβεια, σύμφωνα με τη βιβλιογραφία και το γενικότερο σκεπτικό στα ΔΟΕ, το κόστος επικοινωνίας κυριαρχεί κατά πολύ έναντι των άλλων παραγόντων κόστους, στο σημείο που κάθε τοπική λειτουργία μπορεί να θεωρηθεί (και συνήθως είναι) αμελητέου κόστους στις αντίστοιχες εξισώσεις.

Για να υπολογίσουμε τον αριθμό αυτό, κάθε κόμβος που αποθηκεύει πλειάδες για την σχέση/στήλη που μας ενδιαφέρει, εισάγει το αναγνωριστικό του στην κατανεμημένη σύνοψη. Από τη στιγμή που τα αναγνωριστικά των κόμβων είναι μοναδικά στο ΔΟΕ, η πράξη αυτή είναι το αντίστοιχο ενός συναθροιστικού ερωτήματος Count-Distinct επί μίας φανταστικής στήλης με τα αναγνωριστικά των κόμβων που αποθηκεύουν δεδομένα για την συγκεκριμένη στήλη. Ως εκ τούτου, μπορούμε να χρησιμοποιήσουμε οποιαδήποτε από τις τεχνικές που περιγράψαμε νωρίτερα (δηλ. συνόψεις κατακερματισμού με κόμβους ραντεβού ή σύστημα DHS) για να εκτιμήσουμε αυτή την ποσότητα.

Όταν εφαρμόζεται σε μία ολόκληρη σχέση R , η μετρική «πληθάριθμου κόμβων» αντιστοιχεί στον αριθμό των κόμβων στο σύστημα που αποθηκεύουν πλειάδες της R . Κανείς μπορεί εύκολα να σκεφτεί σημαντικές περιπτώσεις στις οποίες μία τέτοια μετρική μπορεί να αποδειχθεί χρήσιμη. Για παράδειγμα, σε μετέπειτα ενότητες θα δείξουμε πως να υπολογίζονται ιστογράμματα επί ιδιοτήτων μίας σχέσης κατανεμημένης σε ένα σύστημα διαχείρισης δεδομένων διαδικτυακής κλίμακας. Ένας πιθανός βελτιστοποιητής θα χρειαστεί, εκτός από ιστογράμματα της κατανομής συχνοτήτων ιδιοτήτων-τιμών και πληροφορία για το πόσοι κόμβοι αποθηκεύουν πλειάδες που αντιστοιχούν σε κάθε συγκεκριμένο κελί ή που έχουν μία συγκεκριμένη τιμή, για να καταλήξουν σε μία σωστή από-

φαση εκτέλεσης του εκάστοτε ερωτήματος. Με το να χτίζουμε ιστογράμματα πληθάριθμου κόμβων, μπορούμε να απαντήσουμε σε τέτοια ζητήματα χρησιμοποιώντας κλασσικές στατιστικές τεχνικές από τη σχετική βιβλιογραφία. Τέτοιου είδους στατιστικές, μαζί με μία μέθοδο υπολογισμού επικαλύψεων ανάμεσα στα σύνολα δεδομένων των διάφορων κόμβων (όπως η λύση του [85]), θα βελτιώσει σημαντικά την αποδοτικότητα, το κόστος και την ποιότητα της επεξεργασίας ερωτημάτων σε τέτοιες κατανεμημένες συνθήκες.

4.9.4 Sum και Average

Οι Considine, et al. [22] εισήγαγαν την έννοια των «συνόψεων αθροίσματος»: για να υπολογίσουμε το άθροισμα δύο ποσοτήτων α και β , προσθέτουμε α τυχαίες τιμές σε μία πρώτη σύνοψη κατακερματισμού, β τυχαίες τιμές σε μία δεύτερη σύνοψη κατακερματισμού, παίρνουμε το bitwise OR τους και το αποτέλεσμα της εκτίμησης της τελικής σύνοψης προσεγγίζει το άθροισμα των αρχικών τιμών. Στην εργασία [22] οι συγγραφείς προτείνουν έναν αποδοτικό τρόπο συμπλήρωσης των δύο αρχικών συνόψεων κατακερματισμού σε πολυ-λογαριθμικό χρόνο στον αριθμό των τυχαίων τιμών που πρέπει να εισαχθούν.

Η εφαρμογή της λύσης αυτής στο σενάριο των κατανεμημένων συνόψεων μας (προσέγγιση ραντεβού ή DHS) είναι μάλλον προφανής: κάθε κόμβος υπολογίζει τοπικά το άθροισμα των τιμών της στήλης που μας ενδιαφέρει για τις πλειάδες που αποθηκεύει, συμπληρώνει μία τοπική σύνοψη κατακερματισμού χρησιμοποιώντας τον αλγόριθμο των [22] και τότε είτε στέλνει την σύνοψη αυτή στον κόμβο ραντεβού στην πρώτη περίπτωση ή στέλνει μηνύματα «set-to-1» στους κατάλληλους κόμβους στη DHS στην δεύτερη περίπτωση. Ο αριθμός μηνυμάτων, η κατανάλωση εύρους ζώνης δικτύου και το επιπλέον κόστος ερωτήματος/εκτίμησης είναι όπως και στις περιπτώσεις που περιγράφηκαν νωρίτερα. Με τα παραπάνω εργαλεία ανά χείρας, ο υπολογισμός του μέσου όρου (Average) των τιμών μίας στήλης αποτελείται από την εκτίμηση των ερωτημάτων Sum και Count των τιμών της στήλης και την διαίρεση των αποτελεσμάτων.

4.9.5 Minimum και Maximum

Όταν χρησιμοποιείται ένα δίκτυο επικάλυψης DHT-ΔΤ, ο υπολογισμός των ελάχιστων και μέγιστων τιμών μίας συγκεκριμένης στήλης μπορεί να γίνει απλά με δύο lookups στο δίκτυο. Ωστόσο, η τεχνική αυτή μετατρέπει τους κόμβους που αποθηκεύουν τις τιμές αυτές σε σημεία συσσωρευμένης δραστηριότητας, στην οποία περίπτωση απαιτούνται χωριστοί μηχανισμοί εξισορρόπησης φόρτου [1, 46, 56, 88, 102]. Μία εκτίμηση των τιμών αυτών θα μπορούσε επίσης να υπολογιστεί αν υπήρχε κάποιο ιστόγραμμα επί της στήλης που μας ενδιαφέρει.

4.10 Ιστογράμματα σε Συστήματα Διαχείρισης Δεδομένων ΔΟΕ

Στην ενότητα αυτή θα προσπαθήσουμε να χρησιμοποιήσουμε τα εργαλεία που περιγράφηκαν ως τώρα και να παρουσιάσουμε: (α) τεχνικές υλοποίησης ιστογράμματα Ίσου Πλάτους (Equi-Width) και Ίσου Πλάτους Μετακινούμενου Μέσου Όρου (Average Shifted) και (β) μία μέθοδο κατασκευής ιστογραμμάτων Ίσου Βάθους (Equi-Depth), Μέγιστης Διαφοράς (MaxDiff(V,F)) και Συμπιεσμένων (Compressed(V,F)), πάνω από ένα σύστημα διαχείρισης δεδομένων Διαδικτυακής κλίμακας.

4.10.1 Ιστογράμματα Equi-Width

Η βασική ιδέα είναι να διαμερίσουμε το πεδίο τιμής της κάθε στήλης σε ένα σταθερό αριθμό (π.χ. B) διαστημάτων/κελιών ίσου πλάτους και να χρησιμοποιήσουμε μία διαφορετική «μετρική»

για κάθε κελί ώστε να εκτιμήσουμε τον αριθμό των πλειάδων που ανήκουν σε καθένα από αυτά. Αυτή η τεχνική αντιστοιχεί σε ένα κατανεμημένο ιστόγραμμα Ίσου Πλάτους (συμβολίζεται με DEWH) επί της κάθε στήλης. Αν χρησιμοποιούμε μία κατανεμημένη σύνοψη βασισμένη σε DHS, η τεχνική αυτή απαιτεί $O(N \cdot L \cdot \log(N))$ συνολικά μηνύματα στην χειρότερη περίπτωση για να εισάγουν όλοι οι κόμβοι όλα τα αντικείμενά τους στην DHS και $O(L \cdot \log(N))$ βήματα για την ανακατασκευή του ιστογράμματος (υπενθυμίζουμε ότι η καταμέτρηση σε πολλαπλές διαστάσεις με τις DHS δεν επιφέρει επιπλέον κόστος αριθμού βημάτων). Για μία προσέγγιση ραντεβού, τα κόστη αυτά είναι $O(N \cdot B \cdot \log(N))$ και $O(B \log(N))$ αντίστοιχα, αφού θα υπάρχουν (κατά μέσο όρο) B διαφορετικοί κόμβοι ραντεβού. Μία ακόμα πιο ακραία περίπτωση θα ήταν να αποθηκεύσουμε όλα τα κελιά του ιστογράμματος σε έναν μόνο κόμβο ραντεβού· στην περίπτωση αυτή, το κόστος πλήθους βημάτων για την εισαγωγή και την ανακατασκευή του ιστογράμματος είναι $O(\log N)$, όπως είδαμε πιο πριν, αλλά το πρόβλημα μη εξισορροπημένου φόρτου γίνεται ακόμα χειρότερο, όπως θα δούμε και στην πειραματική μας μελέτη.

4.10.2 Ιστογράμματα Average Shifted Equi-Width

Εξαιτίας της ιδιότητας των DHS να μετρούν σε πολλές διαστάσεις χωρίς επιπλέον κόστος αριθμού βημάτων, μπορούμε επιπλέον να υλοποιήσουμε μία κατανεμημένη έκδοση των ιστογραμμάτων Μετακινημένου Μέσου Ίσου Πλάτους (Average-Shifted Equi-Width ή ASH), ονομαζόμενη Distributed ASH ή DASH, χρησιμοποιώντας την τεχνική που περιγράψαμε παραπάνω για τα τυπικά κατανεμημένα ιστογράμματα Ίσου Πλάτους. Κάθε DASH θα αποτελείται από πολλά DEWH με το ίδιο πλάτος στα κελιά τους αλλά με διαφορετική αρχική τιμή στο πεδίο τιμών της στήλης.

Σε μία προσέγγιση βασισμένη σε DHS, κάθε κόμβος κατασκευάζει μία τοπική σύνοψη κατακερματισμού για κάθε κελί κάθε ιστογράμματος Ίσου Πλάτους που αποτελεί το ASH και στέλνει δέσμες από μηνύματα «set-to-1» για κάθε μη μηδενικό bit στις τελικές συνόψεις κατακερματισμού, τα οποία περιλαμβάνουν τα αναγνωριστικά των μετρικών/κελιών του ASH για τα οποία το κάθε bit πρέπει να πάρει την τιμή 1. Επομένως, το κόστος αριθμού βημάτων για την συμπλήρωση του DASH είναι το ίδιο όπως και στην προηγούμενη περίπτωση, ενώ το κόστος ανακατασκευής του ιστογράμματος είναι και πάλι $O(L \cdot \log(N))$.

Από την άλλη, όταν χρησιμοποιούμε DEWH βασισμένα σε προσέγγιση ραντεβού ως τη βάση μας, το κόστος εισαγωγής των αντικειμένων είναι $O(N \cdot S \cdot B \cdot \log N)$ βήματα/μηνύματα για S μετακινημένα DEWH και το κόστος ανακατασκευής του ιστογράμματος είναι $O(S \cdot B \cdot \log N)$ βήματα, αφού στην περίπτωση αυτή θα υπάρχουν (κατά μέσο όρο) $S \cdot B$ διαφορετικοί κόμβοι ραντεβού. Και πάλι, θα μπορούσαμε να πάρουμε την προσέγγιση ραντεβού στα άκρα και να αποθηκεύσουμε όλα τα κελιά όλων των μετακινημένων ιστογραμμάτων στον ίδιο κόμβο ραντεβού, έτσι ώστε να μειώσουμε το παραπάνω κόστος σε $O(\log(N))$, επιφέροντας ωστόσο μία ακόμα πιο άδικη κατανομή φόρτου απ' ότι στην προηγούμενη προσέγγιση.

4.10.3 Υπολογισμός Πολυπλοκότερων Ιστογραμμάτων

Οι κατανεμημένες συνόψεις, όπως περιγράφηκαν παραπάνω, είναι καλές στο να εκτιμούν το πλήθος αντικειμένων που ανήκουν σε ένα προκαθορισμένο σύνολο (π.χ. εύρος τιμών), αλλά δεν μπορούν να καταμετρήσουν αντικείμενα που ανήκουν σε σύνολα των οποίων τα άκρα αλλάζουν δυναμικά. Αυτό καθιστά τις κατανεμημένες συνόψεις ακατάλληλες για απευθείας χρήση στην κατασκευή ιστογραμμάτων εκτός των Ίσου Πλάτους ή Μετακινημένου Μέσου Ίσου Πλάτους.

Προς αυτή την κατεύθυνση, πρώτα παρατηρούμε ότι αν και οι ερευνητές συνήθως δεν προτιμούν τα ιστογράμματα Ίσου Πλάτους έναντι πιο πολύπλοκων μορφών, όπως τα Συμπίεσμένα ή τα Ιστογράμματα Μέγιστης Διαφοράς, υπάρχουν καταστάσεις όπου τα πρώτα λειτουργούν το ίδιο

καλά ή ακόμα και καλύτερα από τα δεύτερα [14], ενώ υπάρχουν περιπτώσεις όπου ακόμα κι ένα ιστόγραμμα ενός κελιού είναι αρκετό [29].

Από την άλλη, μελετώντας τη σχετική βιβλιογραφία, παρατηρήσαμε ότι τα πιο πολύπλοκα ιστογράμματα συνήθως απαιτούν σαφώς μικρότερο αριθμό κελιών για να παρέχουν μία καλή προσέγγιση της κατανομής τιμών της εκάστοτε στήλης σε σχέση με τα ιστογράμματα Ίσου Πλάτους [107]. Επιπλέον, υπενθυμίζουμε ότι χρησιμοποιώντας τις DHS, μπορούμε να χτίσουμε ιστογράμματα (Μετακινημένου Μέσου) Ίσου Πλάτους με πολλά κελιά χωρίς παραπάνω κόστος, εξαιτίας της ιδιότητας παράλληλης καταμέτρησης των DHS.

Στις επόμενες υποενότητες θα χρησιμοποιούμε είτε DEWH είτε DASH ιστογράμματα ως βάση μας και θα προσπαθούμε να εξάγουμε πιο πολύπλοκα ιστογράμματα από αυτά. Η γενική προσέγγιση βασίζεται στην δημιουργία κάποιου DEWH/DASH ιστογράμματος και στην διενέργεια υπολογισμών επ' αυτών για τον υπολογισμό των πολυπλοκότερων τύπων ιστογραμμάτων. Πιο αναλυτικά, η λύση μας αποτελείται από τα εξής βήματα:

1. Πρώτα, χτίζουμε ένα σχετικά μεγάλο ιστόγραμμα DEWH/DASH – για παράδειγμα, κάποιο που να έχει 10-με-20 φορές περισσότερα κελιά από το τελικό ζητούμενο ιστόγραμμα.
2. Στη συνέχεια, χρησιμοποιούμε την υπόθεση ομοιόμορφης διασποράς για να προβάλουμε συχνότητες σε τιμές μέσα σε κάθε κελί.
3. Τέλος, χρησιμοποιούμε αυτά τα ζεύγη τιμών-συχνοτήτων ως προσέγγιση της πραγματικής κατανομής τιμών/συχνοτήτων και εκτιμούμε τα κελιά και τις συχνότητες του τελικού ιστογράμματος.

Ιστογράμματα Equi-Depth

Ο υπολογισμός ενός ιστογράμματος Ίσου Βάθους c κελιών από ένα μεγαλύτερο ιστόγραμμα Ίσου Πλάτους γίνεται σε τρία στάδια

1. Χρησιμοποιούμε την υπόθεση ομοιόμορφης διασποράς για να προβάλουμε συχνότητες σε τιμές μέσα σε κάθε κελί.,
2. Εξετάζουμε το σύνολο των τιμών κάθε κελιού και
3. Βρίσκουμε μία διαμέριση των τιμών αυτών σε c υποσύνολα, έτσι ώστε η σειρά των τιμών να παραμένει η ίδια και όλα τα υποσύνολα να περιέχουν (σχεδόν) το ίδιο άθροισμα συχνοτήτων τιμών.

Ιστογράμματα MaxDiff(V,F)

Ένα ιστόγραμμα $\text{MaxDiff}(V,F)$ c κελιών μπορεί να υπολογιστεί από ένα μεγαλύτερο ιστόγραμμα DEWH/DASH, ως εξής:

1. Χρησιμοποιούμε την υπόθεση ομοιόμορφης διασποράς για να προβάλουμε συχνότητες σε τιμές μέσα σε κάθε κελί,
2. Δημιουργούμε μία ταξινομημένη λίστα των διαφορών των συχνοτήτων γειτονικών τιμών,
3. Επιλέγουμε τις $c - 1$ μεγαλύτερες διαφορές ως όρια των κελιών,
4. Ομαδοποιούμε τις τιμές μέσα σε κάθε κελί και
5. Υπολογίζουμε την συχνότητα κάθε κελιού ως το άθροισμα των συχνοτήτων των τιμών του κελιού.

Ιστογράμματα Compressed(V,F)

Για να υπολογίσουμε αυτόν τον τύπο ιστογράμματος, προχωρούμε όπως και στην περίπτωση των ιστογραμμάτων MaxDiff(V,F), μόνο που τώρα :

1. Εντοπίζουμε τις τιμές με τις $c - k$ υψηλότερες συχνότητες,
2. Τις τοποθετούμε σε μοναδιαία κελιά και
3. Αναθέτουμε τις υπόλοιπες τιμές σε k κελιά ίσου βάθους όπως και στην περίπτωση του ιστογράμματος Ίσου Βάθους παραπάνω.

4.10.4 Συζήτηση

Το πλάτος (και, συνεπώς, το πλήθος) των κελιών του ιστογράμματος στο πρώτο βήμα του αλγορίθμου της ενότητας 4.10.3, καθορίζει την ακρίβεια και την «λεπτότητα» των επόμενων βημάτων εξαγωγής του τελικού ιστογράμματος. Για να μη χάσουμε πληροφορία θα έπρεπε να υπάρχει ένα χωριστό κελί για κάθε τιμή στο πεδίο τιμών της στήλης ενδιαφέροντος (δηλ. τα κελιά θα έπρεπε να έχουν πλάτος 1). Από την άλλη, αν το πεδίο τιμών της στήλης είναι πολύ μεγάλο (π.χ. εκατομμυρίων τιμών), το να έχουμε μοναδιαία κελιά δεν θα είναι πρακτικό. Επιπλέον, η χρήση των DHS εισάγει μία ανταλλαγή ανάμεσα στο πλήθος των αντικειμένων που αντιστοιχούν σε κάθε κελί και στην ακρίβεια της αντίστοιχης εκτίμησης, με περισσότερα αντικείμενα να έχουν ως αποτέλεσμα καλύτερη ακρίβεια.

Από τη στιγμή που οι συχνότητες των κελιών προβάλλονται σε τιμές στο κελί χρησιμοποιώντας την υπόθεση ομοιόμορφης διασποράς, η προτεινόμενη μέθοδος χάνει σε ακρίβεια όσο μεγαλώνει το πλάτος των κελιών. Αυτό ισχύει ειδικά για τα ιστογράμματα τύπου Compressed(V,F) και MaxDiff(V,F) των οποίων η κατασκευή βασίζεται στον ακριβή εντοπισμό των θέσεων υψηλότερων συχνοτήτων ή υψηλότερων διαφορών συχνοτήτων αντίστοιχα. Για παράδειγμα, αν η κατανομή τιμών απέχει πολύ από την ομοιόμορφη και υπάρχει μία πολύ δημοφιλής τιμή σε ένα κελί πολλών τιμών, τότε η προτεινόμενη μέθοδος αντιστοίχισης θα αναθέσει ένα κλάσμα της συνολικής συχνότητας του κελιού σε κάθε διακριτή τιμή ομοιόμορφης διασποράς στο κελί, αποτυγχάνοντας έτσι να υπολογίσει επακριβώς την αντιστοίχιση συχνοτήτων σε τιμές στο κελί αυτό.

Θα ξαναδούμε την ανταλλαγή αυτή στην ενότητα πειραματικής μελέτης. Όπως επίσης θα δούμε εκεί, αυτό το ζήτημα επηρεάζει σχεδόν αποκλειστικά τα Συμπιεσμένα ιστογράμματα και τα ιστογράμματα Μέγιστης Διαφοράς· για τα ιστογράμματα Ίσου Βάθους, ένα αρχικό ιστόγραμμα με 10 έως 20 φορές περισσότερα κελιά καταφέρει σχεδόν τέλεια ακρίβεια με πολύ μικρό κόστος. Από την άλλη, για τα Συμπιεσμένα και τα Μέγιστης Διαφοράς ιστογράμματα, χρειαζόμαστε ένα DEWH/DASH ιστόγραμμα με λίγες τιμές – 1 έως 5 – ανά κελί. Υπάρχει ένα μη προφανές ζήτημα εδώ σχετικά με τις απαιτήσεις σε εύρος ζώνης δικτύου· το μέγεθος των δεδομένων που ανακτούμε κατά την ανακατασκευή του βασικού DEWH/DASH ιστογράμματος αυξάνει γραμμικά στον αριθμό των κελιών του ιστογράμματος. Προς αυτή την κατεύθυνση, σκεφτόμαστε να προσθέσουμε ένα άρθρωμα στην μηχανή εξαγωγής μας, το οποίο θα είναι υπεύθυνο για την αυτόματη ρύθμιση του πλήθους και του πλάτους των κελιών ή/και των μετακινήσεων στο βασικό ιστόγραμμα, ανάλογα με το μέγεθος του πεδίου τιμών, τον τελικό τύπο ιστογράμματος και ένα άνω όριο κατανάλωσης εύρους ζώνης δικτύου το οποίο θα καθορίζεται από τον χρήστη. Η τρέχουσα υλοποίηση του αρθρώματος αυτού είναι σε πρώιμα στάδια και παραμένει θέμα μελλοντικής έρευνας. Ως εκ τούτου, δεν συζητούμε το θέμα αυτό περισσότερο.

4.11 Εφαρμογές στην Βελτιστοποίηση Ερωτημάτων ΔΟΕ

Όπως αναφέραμε ήδη, θεωρούμε τους κόμβους και τις προσβάσεις στο δίκτυο ως blocks και προσβάσεις στην δευτερεύουσα μνήμη αντίστοιχα σε ένα κεντρικοποιημένο σύστημα διαχείρισης βάσεων δεδομένων. Επομένως, ένας βελτιστοποιητής ερωτημάτων για βάσεις δεδομένων σε ΔΟΕ θα πρέπει, εκτός απ' το να λαμβάνει υπόψιν του τα στατιστικά των τιμών των δεδομένων που είναι αποθηκευμένα στο δίκτυο επικάλυψης ΔΟΕ και στατιστικά επί του πληθυσμού των κόμβων, όπως ο «αριθμός των κόμβων ανά σχέση» (σε αναλογία με τον «αριθμό των σελίδων δεδομένων ανά σχέση»). Επομένως, προτείνουμε την αποθήκευση, μαζί με τις συχνότητες των τιμών και τον πληθάρημο κόμβων κάθε κελιού και την χρησιμοποίηση τόσο του αριθμού των αντικειμένων όσο και του αριθμού των κόμβων που εμπλέκονται σε κάθε ερώτημα, για την περαιτέρω βελτιστοποίηση της επεξεργασίας ερωτημάτων. Αυτή η εργασία αντιστοιχεί χονδρικά στην κατασκευή «ιστογραμμάτων κόμβων» επιπλέον των ιστογραμμάτων συχνότητας τιμών.

Με την υποδομή που παρουσιάσαμε ως τώρα, μπορούμε να υπολογίσουμε αρκετές πληροφορίες που απαιτούνται από τους βελτιστοποιητές, όπως συζητείται στο [16]. Πιο αναλυτικά:

1. Τον αριθμό των πλειάδων ανά σχέση. Η ποσότητα αυτή αντιστοιχεί στον πληθάρημο της σχέσης και μπορεί τώρα να υπολογιστεί με ένα απλό ερώτημα Count ή/και Count-Distinct, ανάλογα με το αν θέλουμε τα καταμετρήσουμε διπλότυπα ή όχι.
2. Τον αριθμό φυσικών σελίδων ανά σχέση. Η ποσότητα αυτή αντιστοιχεί στον «πληθάρημο κόμβων» της σχέσης, ο οποίος μπορεί επίσης να υπολογιστεί άμεσα, όπως είδαμε παραπάνω.
3. Στατιστικές πληροφορίες για τις στήλες των σχέσεων, υπό την μορφή:
 - (α') Ιστογραμμάτων. Νωρίτερα στην ενότητα αυτή συζητήσαμε τεχνικές υπολογισμού πολλών τύπων ιστογραμμάτων επί των αποθηκευμένων δεδομένων.
 - (β') Πλήθους διακριτών τιμών σε κάθε στήλη. Και πάλι, η ποσότητα αυτή μπορεί να υπολογιστεί με ένα ερώτημα Count-Distinct στην στήλη ενδιαφέροντος.
 - (γ') Της ελάχιστης και της μέγιστης ή της δεύτερης μικρότερης και δεύτερης μεγαλύτερης, τιμής. Όπως συζητήσαμε παραπάνω, αυτού του είδους οι πληροφορίες συνήθως παρέχονται από δίκτυα επικάλυψης DHT-ΔΤ, πάνω από τα οποία έχουμε θεωρήσει ότι λειτουργούμε. Διαφορετικά, μπορούμε να χρησιμοποιήσουμε ιστογράμματα και την τεχνική εξαγωγής μας για να υπολογίσουμε μία εκτίμηση των τιμών αυτών για κάθε κελί ή για τη συνολική στήλη.
4. Πληροφορία για την συσχέτιση ανάμεσα σε τιμές στηλών, υπό την μορφή είτε:
 - (α') Πολυδιάστατων ιστογραμμάτων, τα οποία μπορούν να δημιουργηθούν χρησιμοποιώντας την υποδομή που παρουσιάσαμε ως τώρα, καθορίζοντας τα διαχωριστικά των πολυδιάστατων ιστογραμμάτων από πριν και αναθέτοντας μία διαφορετική μετρική DHS σε κάθε τέτοιο κελί, σε αντίθεση με τα μονοδιάστατα κελιά που είχαμε ως τώρα είτε
 - (β') Ένα μονοδιάστατο ιστόγραμμα στην κύρια στήλη, συν τον συνολικό αριθμό διακριτών συνδυασμών τιμών των στηλών που υπάρχουν στα δεδομένα, για δείκτες πολλών στηλών. Αυτό μπορεί επίσης να υπολογιστεί με την υποδομή μας, ακόμα και σε ανά-κελί βάση, με το να έχουμε τους κόμβους να εισάγουν συνδυασμούς τιμών σε κατάλληλες μετρικές DHS.

Ακόμα, μπορούμε να παρέχουμε μηχανισμούς για την εκτίμηση της επιλεκτικότητας περιορισμών εύρους. Για να το επιτύχουμε αυτό, χρησιμοποιούμε την υπόθεση ομοιόμορφης διασποράς με μία παραλλαγή. Υπενθυμίζουμε ότι οι κατανεμημένες συνόψεις μπορούν να καταμετρήσουν

τόσο τον συνολικό αριθμό αντικειμένων σε ένα πολυσύνολο συμπεριλαμβανομένων των διπλοτύπων και τον αριθμό των διακριτών τιμών, ανάλογα με το αν εισάγουμε τις τιμές ή τα κλειδιά των πλειάδων. Αφού ασχολούμαστε με ερωτήματα εύρους στο σημείο αυτό, υποθέτουμε ότι το δίκτυο επικάλυψης τα υποστηρίζει αποδοτικά, έτσι μπορούμε να βρούμε την ελάχιστη και τη μέγιστη τιμή ανά κελί ρωτώντας τους κόμβους που είναι υπεύθυνοι για τις τιμές που αντιστοιχούν στα όρια των κελιών· αν από την άλλη δεν υπάρχει αυτή η πρόσθετη λειτουργικότητα ή δεν θέλουμε να κάνουμε τις δύο αυτές πρόσθετες προσβάσεις στο δίκτυο, μπορούμε να υποθέσουμε ότι η ελάχιστη και η μέγιστη τιμή κάθε κελιού συμπίπτουν με τα όρια του κελιού ή να τις υπολογίσουμε όπως παραπάνω.

Δεδομένου ενός περιορισμού εύρους $\alpha \leq X \leq \beta$, προχωράμε ως εξής:

1. Πρώτα καθορίζουμε τα κελιά του ιστογράμματος που σχετίζονται με το ερώτημα εύρους (δηλ. τα κελιά με τα οποία επικαλύπτεται ο περιορισμός εύρους).
2. Χρησιμοποιώντας την DHS, καθορίζουμε την συχνότητα κάθε κελιού που σχετίζεται με το ερώτημα, ακολουθώντας τους αλγόριθμους που περιγράψαμε νωρίτερα.
3. Χρησιμοποιώντας την υπόθεση ομοιόμορφης διασποράς [107], καθορίσουμε τις τιμές κάθε κελιού και τις αντίστοιχες συχνότητές τους.
4. Ανάμεσα από αυτές τις τιμές, βρίσκουμε εκείνες που ανήκουν στον περιορισμό εύρους και
5. Επιστρέφουμε το άθροισμα των συχνοτήτων τους ως το εκτιμώμενο μέγεθος του αποτελέσματος του ερωτήματος εύρους.

Επιπλέον, θα μπορούσαμε να υπολογίσουμε μία εκτίμηση του μεγέθους του self-join μίας σχέσης χρησιμοποιώντας Συμπίεσμένα ιστογράμματα (εξαιτίας της ακαταλληλότητας των βασικών τύπων ιστογραμμάτων ως εκτιμητές όταν οι τιμές των σχέσεων που συνενώνονται είναι συσχετισμένες [29]). Αυτό που πρέπει να κάνουμε είναι:

1. Να χρησιμοποιήσουμε την DHS για να ανακατασκευάσουμε το βασικό ιστόγραμμα και να υπολογίσουμε από αυτό το ιστόγραμμα τύπου Compressed(V,F),
2. Να ελέγξουμε επαναληπτικά όλα τα κελιά του ιστογράμματος και
3. Να πάρουμε το άθροισμα των τετραγώνων των συχνοτήτων των κελιών.

Εξαιτίας της ιδιότητας καταμέτρησης των DHS, η ανακατασκευή του ιστογράμματος έχει ένα κόστος αριθμού βημάτων χειρότερης περίπτωσης $O(L \log(N))$. Προς το παρόν, εξετάζουμε τρόπους εφαρμογής των μεθόδων μας σε άλλα κλασσικά σενάρια βελτιστοποίησης ερωτημάτων, όπως οι προβολές, οι απλές και οι πολυ- συνενώσεις (multi-way joins), κτλ.

Κεφάλαιο 5

Πειραματική Αξιολόγηση

5.1 Δίκτυα Επικάλυψης Βασιζόμενα σε Αλτρουιστές

Θα παρουσιάσουμε αρχικά την πειραματική αξιολόγηση των λύσεων που σχετίζονται με τον εντοπισμό και την χρησιμοποίηση των αλτρουιστικών κόμβων του δικτύου. Καθώς οι βελτιώσεις στην δρομολόγηση που επιτυγχάνονται από το PLANES είναι προφανείς, θα εστιάσουμε στο κόστος διατήρησης των στατιστικών για τους κόμβους και στην αποδοτικότητα του μηχανισμού επίβλεψης. Αρχικά, λοιπόν, εξετάζουμε την απόδοση του AltSeAI, υπό συνθήκες ενός πραγματικού σεναρίου λειτουργίας. Κατόπιν, στρεφόμαστε προς την αρχιτεκτονική RangeGuard και εξετάζουμε τις βελτιώσεις που επιτυγχάνονται από αυτή στο κόστος αριθμού βημάτων για την εκτέλεση ερωτημάτων εύρους, κοιτάζοντας παράλληλα και την κατανομή του φόρτου στο δαχτυλίδι των RG κόμβων.

5.1.1 Απόδοση του AltSeAI

Η πειραματική μας διάταξη θεωρεί ότι το AltSeAI λειτουργεί στο περιβάλλον ενός ΔΟΕ διαμοιρασμού αρχείων μουσικής, με το μέγεθος των αρχείων (σε Mbytes) να είναι ομοιόμορφα κατανομημένο στο διάστημα 3–10 (με μέσο όρο περίπου 6.5 Mbytes). Το προσομοιούμενο δίκτυο αποτελείται από 2500 κόμβους, οι οποίοι διαμοιράζονται 50000 μοναδικά αρχεία, αντιγραφμένα στο δίκτυο ακολουθώντας μία κατανομή Zipf με παράμετρο $\alpha = 0.7$ και 1.2 [119] (με συνολικά 50200 και 51350 αρχεία αντίστοιχα). Στην συνέχεια προσομοιώνονται 200000 αιτήσεις/ερωτήματα. Έχουμε εξετάσει την απόδοση του συστήματός μας και για μεγαλύτερα δίκτυα (με παρόμοια αποτελέσματα) αλλά με όχι αντίστοιχα πολλά ερωτήματα, λόγω περιορισμών μνήμης και υπολογιστικής ισχύος και επομένως παρουσιάζουμε αποτελέσματα μόνο για το δίκτυο των 2500 κόμβων. Τα ερωτήματα φτάνουν στο σύστημα ακολουθώντας μία κατανομή Poisson, έτσι ώστε κάθε κόμβος να εκτελεί κατά μέσο όρο 5 μεταφορτώσεις ανά ημέρα χρόνου προσομοίωσης. Τα αντικείμενα που ζητούνται επιλέγονται επίσης από μία κατανομή Zipf, με παράμετρο $\alpha = 1.2$.

Ο πληθυσμός των κόμβων αποτελείται από 90% (70%) «λαθρεπιβάτες» και από 10% (30%) αλτρουιστές, με συνδέσεις δικτύου οι οποίες κυμαίνονται (ομοιόμορφα) από 33.6kbps (χρήστες με συνδέσεις dial-up) έως 256kbps (πρώιμοι χρήστες DSL) για τους πρώτους και από 256kbps έως 2Mbps (T1) για τους τελευταίους. Επιπλέον, όλοι οι κόμβοι μπορεί να αποτύχουν ή να αρνηθούν να εξυπηρετήσουν κάποια αίτηση με πιθανότητα 0.2, καθώς και να διαγράψουν ή να πάψουν να διαμοιράζονται ένα αρχείο με πιθανότητα 0.1. Η εξεύρεση ενός κόμβου που διαμοιράζεται ένα αρχείο και η μεταφόρτωσή του θεωρούνται αμφότερες λειτουργίες ενός βήματος. Όλες οι λειτουργίες του AltSeAI εκτελούνται πάνω από ένα σύστημα DHT, επομένως κάθε μεταφορά στο AltSeAI θεωρείται πως κοστίζει $O(\log N) = 11$ βήματα κατά μέσο όρο (για ένα δίκτυο 2500 κόμβων).

Σχήμα 5.1: Απόδοση του Μηχανισμού Καταγραφής

Όπως θα δούμε σε λίγο, παρά αυτή την «αδικία» εις βάρος του, το AltSeAI επιφέρει αμελητέο κόστος δικτύου. Επιπλέον, αν το σύστημά μας λειτουργούσε σε περιβάλλον δικτύων αποθήκευσης-και-προώθησης, όπως είναι τα συστήματα FreeNet και AChord, το παρατηρούμενο κόστος δικτύου και αποθηκευτικού χώρου θα ήταν τάξεις μεγέθους μικρότερο. Οι κόμβοι υπολογίζουν τη βαθμολογία τους χρησιμοποιώντας τη διαφορά $|F_d| - |F_o|$. Τέλος, οι κόμβοι ανακατευθύνουν εισερχόμενες αιτήσεις με πιθανότητα 0, 1 και 0.5, όταν η βαθμολογία τους είναι κάτω από το κατώτερο όριό τους, πάνω από το ανώτερο όριό τους ή ανάμεσα σε αυτές τις δύο τιμές αντίστοιχα.

Τα αποτελέσματα της προσομοίωσης αποτυπώνονται στα σχήματα 5.1 και 5.2. Κατά την διάρκεια της προσομοίωσης, ο μηχανισμός παλαιώσης των Αποδείξεων Συναλλαγής ήταν απενεργοποιημένος, εξ ου και η γραμμική αύξηση του αποθηκευτικού κόστους (σχήμα 5.2). Ακόμα κι έτσι, όμως, το AltSeAI επιφέρει κατά μέσο όρο ένα επιπλέον κόστος αποθήκευσης της τάξης του 0.16%. Επιπλέον, το κόστος δικτύου σταθεροποιείται περίπου στο 0.7%, ενώ περισσότερο από το 90% των κόμβων έχουν καταγραφεί από τον μηχανισμό μας ύστερα από τις πρώτες 15000 αιτήσεις (σχήμα 5.1).

5.1.2 Απόδοση του RangeGuard

Για το δεύτερο μέρος, θα χρησιμοποιήσουμε ένα DHT-ΔΤ που αναπτύξαμε οι ίδιοι, ονομαζόμενο OP-Chord [125], ως το δίκτυο επικάλυψης της επιλογής μας. Το OP-Chord βασίζεται στο σύστημα Chord, αντικαθιστώντας την συνάρτηση διασποράς SHA-1 του δεύτερου για την δημιουργία των αναγνωριστικών των αντικειμένων με μία συνάρτηση διασποράς διατήρησης της τοπικότητας, ενώ διέπεται από τις ίδιες αρχές λειτουργίας όπως και τα υπόλοιπα συστήματα DHT-ΔΤ. Επεκτείναμε τον βασικό προσομοιωτή του Chord (διαθέσιμο στη διεύθυνση <http://www.pdos.lcs.mit.edu/chord/>), προσθέτοντας υποστήριξη για δείκτες πλειάδων και ερωτήματα εύρους και υλοποιώντας τις αρχιτεκτονικές μας OP-Chord και RangeGuard. Επιλέξαμε να εξετάσουμε δύο διαστάσεις του συστήματος: (α) την απόδοσή του σε κόστος αριθμού βημάτων

Σχήμα 5.2: Δικτυακό και Αποθηκευτικό Κόστος του AltSeAI

για τον αλγόριθμό μας για επεξεργασία ερωτημάτων εύρους, και (β) την κατανομή αποθηκευτικών απαιτήσεων και προσβάσεων στους συμμετέχοντες κόμβους RG κατά τη διάρκεια της επεξεργασίας ερωτημάτων εύρους, υπό ρεαλιστικές κατανομές πρόσβασης.

Αριθμός Βημάτων Στα πειράματα χρησιμοποιήθηκε μία σχέση μίας στήλης, με το χαρακτηριστικό αυτό να λαμβάνει τιμές ανάμεσα σε 5000 ακραίους, ακολουθώντας μία κατανομή Zipf με παράμετρο $\theta = 0.7$ επί του πεδίου τιμών του $\mathcal{D}$ [114]. Τα ερωτήματα εύρους δημιουργήθηκαν χρησιμοποιώντας μία διαφορετική κατανομή Zipf επί του πεδίου τιμών (και πάλι με $\theta = 0.7$) για το κάτω άκρο τους, κι έναν ομοιόμορφα κατανεμημένο εύρος διαστήματος S το οποίο κυμαινόταν από 1% έως 50% του πεδίου τιμών της στήλης. Αναφέρουμε αποτελέσματα μίας σειράς 50000 ερωτημάτων σε ένα σύστημα αποτελούμενο από $N = 1000$ κόμβους, $M = 50$ κόμβους RG (περίπου 5% του N) και 50000 πλειάδες (τα αποτελέσματα είναι παρόμοια και για άλλες τιμές των παραπάνω παραμέτρων).

Σημεία Αναφοράς Απόδοσης Συγκρίναμε την απόδοση κόστους αριθμού βημάτων της αρχιτεκτονικής RangeGuard απέναντι (α) στο απλό Chord (PC), ως εκπροσώπου των παραδοσιακών DHT, (β) σε ένα φανταστικό, επαυξημένο σύστημα Chord (EC), όπου για κάθε εύρος $\mathcal{R}$ το σύστημα γνωρίζει τα αναγνωριστικά των n' κόμβων που αποθηκεύουν τις τιμές στο $\mathcal{R}$, (γ) στην αρχιτεκτονική OP-Chord και (δ) σε ένα υβριδικό σύστημα όπου 95% των ερωτημάτων επεξεργάζονται από το δαχτυλίδι RG και το υπόλοιπο 5% ανατίθενται στο δαχτυλίδι OP-Chord (βλπ. ενότητα 3.4.6).

Ας υποθέσουμε ότι έχουμε ένα ερώτημα r για το εύρος ακεραίων $\langle v_{low}, v_{high} \rangle$, καθώς και ότι οι δείκτες προς τις ζητούμενες πλειάδες είναι αποθηκευμένα σε n' κόμβους, σύμφωνα με την ανάθεση του απλού Chord και σε k κόμβους διαχειριζόμενους από k' RangeGuards, σύμφωνα με το DHT-ΔΤ σχήμα μας. Τότε, για να εκτελεστεί το ερώτημα αυτό:

- PC : απαιτούνται $|r|$ ερωτήματα ή $O(|r| \log(N))$ βήματα.

Σχήμα 5.3: Αριθμός Βημάτων ανά Ερώτημα Εύρους (Λογαριθμική Αποτύπωση)

- *EC*: πρέπει να εκτελεστούν $|n'|$ ερωτήματα ή $O(|n'| \log(N))$ βήματα.
- *OP*: πρέπει πρώτα να εντοπίσουμε τον κόμβο που αποθηκεύει την τιμή v_{low} και κατόπιν να ακολουθήσουμε $k-1$ δείκτες επόμενου κόμβου, κάνοντας συνολικά $O(\log(N)+k)$ βήματα.
- *RG*: πρέπει να δρομολογήσουμε το ερώτημα προς τον πλησιέστερο κόμβο *RG* (1 βήμα), ύστερα στον κόμβο *RG* που αποθηκεύει την τιμή v_{low} ($O(\log(M))$ βήματα) και κατόπιν να ακολουθήσουμε $k' - 1$ δείκτες επόμενου κόμβου, με συνολικό κόστος $O(\log(M) + k')$ βημάτων.
- *OP + RG*: επιλέγουμε ανάμεσα στα *OP* και *RG* χρησιμοποιώντας ένα «άδικο κέρμα», με τα κόστη αριθμού βημάτων που αναφέρθηκαν παραπάνω.

Το σχήμα 5.3 συνοψίζει τον μετρούμενο αριθμό βημάτων ανά ερώτημα εύρους. Τα παραδοσιακά συστήματα DHT δεν είχαν σχεδιαστεί με ερωτήματα εύρους κατά νου κι έτσι το Chord έχει πολύ φτωχή απόδοση. Το φανταστικό, επαυξημένο Chord μειώνει τον απαιτούμενο αριθμό βημάτων κατά περίπου 20% σε σχέση με το απλό Chord. Ωστόσο, απαιτεί την ύπαρξη καθολικής γνώσης για να υλοποιηθεί· κάθε κόμβος θα πρέπει να γνωρίζει την ακριβή κατανομή των πλειάδων στους κόμβους του συστήματος, καθώς και αν υπάρχουν πλειάδες με κάθε τυχαία τιμή του πεδίου τιμών. Από την άλλη, χρησιμοποιώντας πρώτα το DHT-ΔΤ μας και εν συνεχεία την αρχιτεκτονική RangeGuard, το κόστος μειώνεται κατά έναν παράγοντα από 50 έως 500 φορές σε σχέση με το *PC*, 10 έως 110 φορές σε σχέση με το *EC* και 5 έως 20 φορές σε σχέση με το *OP* για διάφορα μεγέθη εύρους τιμών, με την απόδοση του *OP + RG* να ακολουθεί κατά πόδας.

Κατανομή Φόρτου Η τυχαία τιμή έναρξης της αντιστοίχισης στο OP-Chord και η επικάλυψη πολλαπλών χαρακτηριστικών/στηλών έχουν θετική επίδραση στην κατανομή του φόρτου πρόσβασης/αποθήκευσης στο σύστημά μας. Για να υποστηρίξουμε αυτή μας την πρόταση, εκτελέσαμε το ακόλουθο πείραμα. Έστω ένα δαχτυλίδι OP-Chord. Προσθέτουμε κόμβους σε ψευδοτυχαίες θέσεις, προσομοιώνοντας έτσι την αντιστοίχιση που προκύπτει από την χρήση της συνάρτησης διασποράς

Σχήμα 5.4: Λόγος Υψηλότερου-Χαμηλότερου Φόρτου

SHA-1 και αφήνουμε το σύστημα να σταθεροποιηθεί. Εν συνεχεία, προσθέτουμε 20000 πλειάδες πολλαπλών στηλών. Οι τιμές των χαρακτηριστικών κάθε πλειάδας επιλέγονται από 40000 ακέραιες τιμές, ακολουθώντας μία κατανομή Zipf με παράμετρο $\theta = 0.7$. Από την άλλη, αν θεωρήσουμε μία ομοιόμορφη κατανομή των τιμών εισόδου (σε αντίθεση με την παραπάνω Zipf κατανομή), τότε τα παρακάτω αποτελέσματα ισχύουν για Zipf κατανομή προσβάσεων στα δεδομένα. Μεταβάλλουμε (α) τον αριθμό των στηλών ανά πλειάδα, από 1 (η περίπτωση των κλασσικών μονοδιάστατων σχέσεων που υποστηρίζεται από το σύστημα Chord) έως και 400 στήλες και (β) τον αριθμό των κόμβων του ΔΟΕ από 1000, σε 5000, 10000 και 20000. Σημειώνουμε ότι, για παράδειγμα, στην περίπτωση των 20000 κόμβων, αν επρόκειτο για κόμβους *RG*, θα ήταν αρκετοί για να διαχειριστούν ένα δίκτυο 400000 κόμβων, υπό την αναλογία του 5% που περιγράφηκε νωρίτερα.

Το σχήμα 5.4 δείχνει τον λόγο ανάμεσα στον υψηλότερο και τον χαμηλότερο φόρτο στο σύστημα. Φυσιολογικά, ο βέλτιστος λόγος φόρτου θα ήταν 1, στην οποία περίπτωση όλοι οι κόμβοι του συστήματος θα είχαν ακριβώς τον ίδιο φόρτο. Για κατανομή τιμών Zipf με παράμετρο $\theta = 0.7$ και 1000 κόμβους στο σύστημα, ο λόγος αυτός για μονόστηλες σχέσεις είναι 7.5, πέφτοντας στο 1.97 για 8 και στο 1.06 για 400 στήλες. Έχουμε σημειώσει στα σχήματα τον λόγο για τις περιπτώσεις με μία μόνο στήλη (πρόκειται για τα σημεία με την ένδειξη «load =») καθώς και τον αριθμό των στηλών που απαιτείται για να πέσει ο λόγος αυτός κάτω από την τιμή 2 (σημεία με την ένδειξη «# of attributes»). Με τους κόμβους να τοποθετούνται στον χώρο αναγνωριστικών χρησιμοποιώντας την συνάρτηση διασποράς SHA-1 του Chord, μπορούμε να θεωρήσουμε ότι οι κόμβοι *RG* θα είναι σχεδόν ομοιόμορφα κατανεμημένοι στο δαχτυλίδι. Επομένως, η παραπάνω κατάσταση ισχύει τόσο για το δαχτυλίδι του κάτω επιπέδου όσο και για το δαχτυλίδι του RangeGuard στην αρχιτεκτονική μας. Ωστόσο, παρατηρήστε την αύξηση στο τελευταίο όσο αυξάνονται οι κόμβοι του δικτύου. Καθώς αναμένουμε οι κόμβοι *RG* να αποτελούν μικρό ποσοστό του συνολικού πληθυσμού κόμβων, τα παραπάνω αποτελέσματα δείχνουν πως, για το δαχτυλίδι *RG*, η κατανομή φόρτου θα είναι εντός αποδεκτών ορίων, ακόμα και χωρίς τους υπόλοιπους μηχανισμούς που συζητήθηκαν στην ενότητα 3.4. Για πολύ μεγαλύτερα δίκτυα, χρειαζόμαστε έναν πολύ μεγάλο αριθμό στηλών για να επιτευχθεί μία καλή κατανομή φόρτου ή/και κάποια πιο εμπειριστατωμένη λύση εξισορρόπηση φόρτου, όπως ο μηχανισμός που περιγράφουμε στο [102].

5.2 Κατανεμημένες Συνόψεις Κατακερματισμού

Θα παρουσιάσουμε τώρα μία σε βάθος αξιολόγηση της απόδοσης και της κλιμακωσιμότητας των αλγορίθμων και των πρωτοκόλλων μας για κατανεμημένη καταμέτρηση και υπολογισμό συναθροιστικών ερωτημάτων και ιστογραμμάτων. Αρχικά θα δώσουμε μία περιγραφή της μεθοδολογίας που ακολουθήσαμε και στη συνέχεια θα αναλύσουμε τα αποτελέσματά μας.

5.2.1 Μεθοδολογία

Αρχικά, στην εργασία [97] είχαμε υλοποιήσει ένα DHT τύπου Chord και το σύστημα DHS σε έναν προσομοιωτή συμβάντων, γραμμένο εκ του μηδενός στη γλώσσα προγραμματισμού C++. Ο προσομοιωτής αυτός υποστήριζε όλους τους βασικούς τελεστές των δικτύων DHT – με άλλα λόγια, προσθήκη κόμβων, αφαίρεση/αποτυχία κόμβων, καθώς και προσθήκη και αφαίρεση δεδομένων – και όλη την λειτουργικότητα του DHS και των συν αυτώ δομών. Χρησιμοποιήσαμε τον προσομοιωτή αυτόν για την αρχική αξιολόγηση της απόδοσης του DHS και για έλεγχο της ορθότητας των προτεινόμενων λύσεων, καθώς και για να παρατηρήσουμε πως συμπεριφέρονται και κλιμακώνονται οι τελευταίες σε ένα πλήρως ελεγχόμενο περιβάλλον. Κατόπιν υλοποιήσαμε το DHS πάνω από το FreePastry [38], μία κοινώς διαθέσιμη υλοποίηση του DHT Pastry [32] από τους ίδιους τους δημιουργούς του DHT αυτού, στην γλώσσα προγραμματισμού Java. Στο Pastry (και στο FreePastry) η δρομολόγηση ανάμεσα σε οποιοδήποτε τυχαίο ζεύγος κόμβων απαιτεί $O(\log_{2^b} N)$ βήματα, όπου b ένας θετικός ακέραιος. Ρυθμίσαμε το FreePastry έτσι ώστε οι πίνακες δρομολόγησής του να είναι $O(\log N)$ εγγραφών (δηλ. $b = 1$), ώστε τα αποτελέσματά μας να είναι απευθείας συγκρίσιμα με αυτά που θα παίρναμε αν η υλοποίηση είχε βασιστεί σε κάποιο από τα υπόλοιπα συστήματα DHT. Και οι δύο υλοποιήσεις έδωσαν παρόμοια αποτελέσματα, εξ ου και παρουσιάζουμε μόνο αυτά που προέκυψαν από την δεύτερη, ώστε να δείξουμε την απόδοση των λύσεών μας σε μία πραγματική υλοποίησή τους. Ο κώδικας της υλοποίησης αποτελείται από περίπου 5,500 γραμμές κώδικα Java¹ και είναι διαθέσιμη ελεύθερα στον παγκόσμιο ιστό [37].

Σε όλες τις περιπτώσεις, η αξιολόγηση έγινε ακολουθώντας τα εξής βήματα :

1. Προσθέσαμε κόμβους στο ΔΟΕ και αφήσαμε αρκετό χρόνο ώστε αυτό να φτάσει σε σταθερή κατάσταση. Προσομοιώσαμε δίκτυα αποτελούμενα από 1000, 2500, 5000 και 10000 κόμβους έτσι ώστε να καταδείξουμε την κλιμακωσιμότητα της προσέγγισής μας και ως παράδειγμα κατανεμημένων συστημάτων ΔΟΕ μεσαίας κλίμακας· ωστόσο, μεγαλύτερα δίκτυα προφανώς υποστηρίζονται τόσο από τις λύσεις μας όσο και από την υλοποίησή μας.
2. Δημιουργήσαμε τα δεδομένα που θα αποθηκευτούν στους κόμβους. Σύμφωνα με τη συνήθη πρακτική στην σχετική βιβλιογραφία, δημιουργήσαμε συνθετικά σύνολα δεδομένων τα οποία (α) μπορούν να δοκιμάσουν το σύστημά μας για διάφορες κατανομές τιμών εισόδου, από σχεδόν ομοιόμορφες ως ιδιαίτερα επικλινείς (skewed) και (β) αντιστοιχούν σε κατανομές τιμών που έχουν παρατηρηθεί και σε πραγματικά συστήματα [114]. Συγκεκριμένα, οι τιμές των δεδομένων πάρθηκαν είτε με βάση την κατανομή Zipf με παράμετρο $\theta = 1.2$ (αλλά και 0.7 και 1.0 για την περίπτωση των ιστογραμμάτων και των συναθροιστικών ερωτημάτων) ή από μία ομοιόμορφη κατανομή. Το πλήθος των αντικειμένων ήταν σε κάθε περίπτωση $250 \times$ το πλήθος των κόμβων του ΔΟΕ (δηλ. 250000 αντικείμενα για ΔΟΕ 1000 κόμβων, 625000 αντικείμενα για ΔΟΕ 2500 κόμβων, κ.ο.κ.), έτσι ώστε να δείξουμε και την κλιμακωσιμότητα της προσέγγισής μας σε σχέση με το μέγεθος του κατανεμημένου πολυσυνόλου.
3. Αναθέσαμε με τυχαίο τρόπο τα αντικείμενα σε κόμβους στο ΔΟΕ. Αυτή η ανάθεση αντιστοιχεί στα δεδομένα που συνεισφέρει κάθε κόμβος του δικτύου.

¹ Η μέτρηση έγινε με το γνωστό πρόγραμμα «SLOCCount» του David A. Wheeler.

4. Κατόπιν όλοι οι κόμβοι εισήγαγαν στην κατανεμημένη σύνοψη όλες τις σχετικές πληροφορίες για τα αντικείμενα που τους ανατέθηκαν στο προηγούμενο βήμα.
5. Τέλος, επιλέξαμε τυχαίους κόμβους οι οποίοι και εκτέλεσαν τους αλγορίθμους μας για αναδόμηση ιστογραμμάτων και υπολογισμό συναθροιστικών ερωτημάτων ή του πληθάριθμου των αντικειμένων.

Για την αξιολόγηση του DHS χρησιμοποιήσαμε 128 και 512 πίνακες ανά σύνοψη κατακερματισμού και 32 bits ανά πίνακα, ενώ για το δεύτερο τμήμα το πλήθος των πινάκων κυμαινόταν από 64 έως 512 και χρησιμοποιήθηκαν 40 bits ανά πίνακα. Σημειώνουμε εδώ ότι τα αναμενόμενα αποτελέσματα εξαρτώνται άμεσα από το μήκος των πινάκων και ότι με 32 bits μπορούμε να μετρήσουμε πάνω από τρισεκατομμύρια αντικειμένων. Τα αποτελέσματα που θα παρουσιάσουμε σε λίγο προέκυψαν ύστερα από πολλές επαναλήψεις για κάθε περίπτωση, ώστε να αποκλείσουμε στατιστικά λάθη.

Εν ολίγοις, ενδιαφερόμαστε και παρουσιάζουμε αποτελέσματα για τους παρακάτω τομείς:

1. Κόστος πλήθους βημάτων για την εισαγωγή αντικειμένων στο DHS.
2. Φόρτος εισαγωγής ανά εισαγόμενο αντικείμενο και συνολική κατανομή φόρτου εισαγωγής.
3. Κόστος πλήθους βημάτων για την εκτέλεση ερωτημάτων (π.χ. για τον υπολογισμό της εκτίμησης του πληθάριθμου του κατανεμημένου πολυσυνόλου).
4. Φόρτος των κόμβων ανά ερώτημα και συνολική κατανομή φόρτου ερωτημάτων.
5. Ακρίβεια των υπολογιζόμενων εκτιμήσεων, ως το μέσο λάθος σε σχέση με την αντίστοιχη κεντριοποιημένη εκτίμηση.

Πιο συγκεκριμένα, ενδιαφερόμαστε κυρίως για διαφάνεια κατανομής, επομένως θέλουμε πρωτίστως να δείξουμε ότι οι προτεινόμενες λύσεις (α) είναι όσο ακριβείς και οι αντίστοιχες κεντριοποιημένες, (β) επιφέρουν χαμηλό επιπλέον κόστος κατά την εκτέλεσή τους και (γ) κλιμακώνονται καλά με το μέγεθος του δικτύου και το σύνολο των δεδομένων των κόμβων. Για την περίπτωση των DHS και για την επεξεργασία συναθροιστικών ερωτημάτων, ενδιαφερόμασταν κυρίως για τον αριθμό των βημάτων που απαιτούνται για την εισαγωγή δεδομένων και την εκτίμηση, την ακρίβεια της ίδιας της εκτίμησης, καθώς και την εξισορροπημένη κατανομή φόρτου στους κόμβους του δικτύου. Αρχικά προσθέσαμε στο FreePastry την απαιτούμενη λογική για να μας επιτρέψει να μετρήσουμε το πλήθος των βημάτων που χρειάστηκε κάθε μήνυμα για να φτάσει στον προορισμό του και την αντίστοιχη χρήση εύρους ζώνης δικτύου. Παρουσιάζουμε αποτελέσματα αριθμού βημάτων για διάφορες τιμές των παραμέτρων της βασικής αρχιτεκτονικής DHS, καθώς και για τρία συναθροιστικά ερωτήματα: συγκεκριμένα, για τα Count, Sum και Average, ως παραδείγματα συναθροιστικών ερωτημάτων που βασίζονται σε κλασσικές συνόψεις κατακερματισμού, σε αθροιστικές συνόψεις κατακερματισμού και σε συνδυασμό των δύο αυτών τύπων συνόψεων αντίστοιχα. Ακόμα, αναφέρουμε το μέσο σφάλμα εκτίμησης, υπολογιζόμενο ως το ποσοστό κατά το οποίο η κατανεμημένη εκτίμηση διαφέρει από την πραγματική τιμή του εκτιμώμενου συναθροιστικού ερωτήματος που θα παίρναμε αν τα δεδομένα βρίσκονταν σε κεντριοποιημένο περιβάλλον (δηλ. αν ήταν αποθηκευμένα σε έναν και μοναδικό κόμβο).

Για τα ιστογράμματα, μας απασχολεί κυρίως η ακρίβεια των εκτιμώμενων συχνοτήτων κελιών και των ορίων των κελιών (για τα εξαγόμενα ιστογράμματα), καθώς και το κόστος της ανακατασκευής του βασικού DEWH/DASH ιστογράμματος σε αριθμό βημάτων, κατανάλωση εύρους ζώνης και κατανομή φόρτου. Για τον αριθμό βημάτων και την χρήση εύρους ζώνης, προχωρήσαμε όπως και παραπάνω. Για να μετρήσουμε την ακρίβεια των εκτιμώμενων ιστογραμμάτων, πρώτα υπολογίσαμε τα ιστογράμματα της επιλογής μας επί των αποθηκευμένων δεδομένων με κεντριοποιημένο

τρόπο, κατόπιν ανακατασκευάσαμε και εξάγαμε τα αντίστοιχα κατανεμημένα ιστογράμματα. Η ακρίβεια και πάλι εκφράζεται ως το ποσοστό κατά το οποίο οι εκτιμώμενες συχνότητες ανά κελί ή τα όρια των κελιών διαφέρουν από τις αντίστοιχες τιμές των κεντριοποιημένων ιστογραμμάτων.

Άλλη μία ζωτικής σημασίας μετρική για κατανεμημένα συστήματα όπως τα δίκτυα δεδομένων ΔΟΕ είναι η κατανομή του φόρτου στους συμμετέχοντες κόμβους. Μετράμε τον φόρτο σε κάθε κόμβο ως το πλήθος των φορών που ο εκάστοτε κόμβος είναι ο αποδέκτης μίας εισαγωγής ή ενός ερωτήματος. Υλοποιήσαμε την επιπλέον λογική στο FreePastry ώστε να μπορούμε να μετρήσουμε τις παραπάνω μετρικές· συγκεκριμένα τον Φόρτο Κόμβου Εισαγωγής (Node Insertion Hits) και τον Φόρτο Κόμβου Ερωτήματος (Node Query Hits). Ας υποθέσουμε ότι l_i , $1 \leq i \leq N$ είναι ο φόρτος στον i -οστό κόμβο και ότι μ_l και σ_l είναι ο μέσος όρος και η τυπική απόκλιση του δείγματος των φορτίων αυτών αντίστοιχα. Δηλαδή:

$$\mu_l = \frac{1}{N} \cdot \sum_{i=1}^N l_i$$

και

$$\sigma_l = \sqrt{\frac{1}{N-1} \cdot \sum_{i=1}^N (l_i - \mu_l)^2}.$$

Χρησιμοποιήσαμε μία πλειάδα μετρικών για να δείξουμε την επιρροή των διαφόρων προσεγγίσεων που είδαμε νωρίτερα στο τμήμα αυτό. Πιο συγκεκριμένα, χρησιμοποιήσαμε:

- Τον Συντελεστή Gini [28], όπως προτάθηκε στο [103], ο οποίος υπολογίζεται ως:

$$GC = \frac{1}{2 \cdot N^2 \cdot \mu_l} \sum_{i=1}^N \sum_{j=1}^N (|l_i - l_j|).$$

Με άλλα λόγια, ο GC είναι η μέση τιμή των απόλυτων διαφορών κάθε δυνατού ζεύγους τιμών φόρτου. Ο Συντελεστής Gini λαμβάνει τιμές στο διάστημα $[0, 1]$, όπου μία τιμή GC 0.0 είναι η καλύτερη δυνατή κατάσταση, με 1.0 να είναι η χειρότερη. Ο Συντελεστής Gini χονδρικά αντικατοπτρίζει το ποσοστό της έλλειψης εξισορρόπησης φόρτου στο σύστημα, έτσι ώστε για παράδειγμα μία τιμή GC 0.25 μεταφράζεται σε $\approx 75\%$ του συνολικού φόρτου να είναι ισοκατανεμημένος σε όλους τους κόμβους του συστήματος.

- Τον Συντελεστή Fairness Index [66]:

$$FI = \frac{\left(\sum_{i=1}^N l_i\right)^2}{N \cdot \sum_{i=1}^N l_i^2}.$$

Ο Συντελεστής Fairness Index (επίσης γνωστός και ως «Συντελεστής Fairness Index του Jain» από το όνομα του πρώτου συγγραφέα στο [66]) λαμβάνει τιμές στο διάστημα $(0, 1]$, με τις τιμές 0 και 1 να είναι αντίστοιχα η χειρότερη και η καλύτερη δυνατή. Ο FI είναι κατά κάποιον τρόπο το αντίστροφο του GC με την έννοια του ότι αντικατοπτρίζει το ποσοστό της εξισορρόπησης φόρτου στο σύστημα, έτσι ώστε μία τιμή FI 0.25 να μεταφράζεται στο ότι ο φόρτος ισοκατανέμεται στο 25% των κόμβων του συστήματος.

- Τον μέγιστο και το συνολικό φόρτο των κόμβων του δικτύου.

Ως μέτρο σύγκρισης, (α) στην απλή προσέγγιση ραντεβού, η τιμή του GC για την κατανομή φόρτου εισαγωγής και ερωτημάτων είναι σχεδόν ίση με 1, και (β) ένα απλό δίκτυο DHT με τους κόμβους και τα αντικείμενα να παίρνουν ψευδομοιόμορφες τιμές και τα δεύτερα να αντιστοιχίζονται στους

	Δρομολόγηση	Συντήρηση		Ερωτήματα		Επιρροή
A/A	Δρομολόγηση Bits	Αντικείμενα ανά Πράξη	Αντίγραφα	Έλεγχος Bits	Όριο Επ/ψεων	Σ : Συντήρηση Ε : Ερώτημα
Σενάρια Ροών						
1	IBR	Single	0	SBP	5	-
2	IBR+RS	Single	0	SBP	5	Σ+E
3	IBR	Single	0	BBP	5	E
4	RBR	Single	0	SBP	5	Σ+E
5	RBR+RS	Single	0	SBP	5	Σ+E
Σενάρια Διαχείρισης Δεδομένων						
6	RBR	Bulk	0	SBP	5	Σ
7	RBR	Bulk	0	SBP+BC	5	E
8	RBR+RS	Bulk	0	SBP+BC	5	Σ+E
9	RBR	Bulk	5	SBP	0	Σ+E
10	RBR+RS	Bulk	5	SBP	5	Σ+E

Πίνακας 5.1: Εκτιμώμενες Ρυθμίσεις Συστήματος

πρώτους χρησιμοποιώντας consistent hashing [70], έχει τιμή GC κατανομής φόρτου εισαγωγής ≈ 0.7 .

5.2.2 Αποτελέσματα DHS

Εξαιτίας του πλάτους του χώρου σχεδίασης, εντοπίσαμε τους πιο ενδιαφέροντες συνδυασμούς σχεδιαστικών επιλογών και παρουσιάζουμε αποτελέσματα μόνο γι' αυτούς. Ξεκινάμε από το σύνολο ρυθμίσεων του [97] και κατόπιν ρυθμίζουμε σταδιακά τις διάφορες παραμέτρους που συζητήθηκαν στο τμήμα αυτό, δείχνοντας πώς καθεμία από αυτές επηρεάζει το αποτέλεσμα και βελτιώνει την ως τώρα απόδοση. Θα περιγράψουμε εν συντομία καθέναν από αυτούς τους συνδυασμούς ρυθμίσεων και ύστερα θα συζητήσουμε την επιρροή τους στις διάφορες μετρικές απόδοσης. Οι συνδυασμοί αυτοί συνοψίζονται στον πίνακα 5.1. Σε όλες τις περιπτώσεις χρησιμοποιούμε τον εκτιμητή superLogLog.

Ο πρώτος συνδυασμός ρυθμίσεων (Περίπτωση 1), συμβολιζόμενος ως DHS_0 , αντιστοιχεί στις ρυθμίσεις του [97] και αποτελεί το αρχικό μας σημείο εκκίνησης. Εν συντομία, το DHS_0 χρησιμοποιεί επαναληπτική δρομολόγηση bits χωρίς συντομεύσεις, μεμονωμένες εισαγωγές χωρίς αντίγραφα και σειριακό έλεγχο των bits χωρίς κρυφή αποθήκευση και με όριο επαναλήψεων 5 βημάτων. Σε ένα δεύτερο βήμα, ενεργοποιούμε τις συντομεύσεις δρομολόγησης (Περίπτωση 2) και επαναλαμβάνουμε τις μετρήσεις μας, ώστε να εξετάσουμε το αποτέλεσμα της βελτιστοποίησης αυτής σε έναν κατά τ' άλλα απλό συνδυασμό ρυθμίσεων της DHS. Αναφέρουμε αποτελέσματα τόσο για τα χαρακτηριστικά εισαγωγής όσο και για τα χαρακτηριστικά επερώτησης των περιπτώσεων αυτών.

Το επόμενο βήμα (Περίπτωση 3) προσθέτει τον έλεγχο δυαδικής αναζήτησης στη θέση του σειριακού ελέγχου των bits του αλγορίθμου DHS_0 . Καθώς αυτή η αλλαγή επηρεάζει μόνο το κομμάτι ελέγχου των bits, θα δείξουμε αποτελέσματα μόνο για ερωτήματα. Στην περίπτωση 4 ξεκινάμε και πάλι από το DHS_0 και ενεργοποιούμε την αναδρομική δρομολόγηση bits. Επιπλέον, προσθέτουμε και συντομεύσεις δρομολόγησης σε αυτό το σύνολο ρυθμίσεων (Περίπτωση 5), για να εξετάσουμε την συνεργασία αναδρομικής δρομολόγησης και δρομολόγησης σύντομης διαδρομής. Κατόπιν, στρέφουμε την προσοχή μας σε περιπτώσεις που χρησιμοποιούν μαζικές εισαγωγές. Πρώτα, η Περίπτωση 6 εξετάζει το αποτέλεσμα των μαζικών εισαγωγών στο κόστος αριθμού βημάτων και στην κατανομή φόρτου στην DHS. Εν συντομία, η περίπτωση αυτή είναι ίδια με την Περίπτωση 4, με

τη διαφορά της αντικατάστασης μεμονωμένων εισαγωγών με μαζικές εισαγωγές. Αναφέρουμε αποτελέσματα μόνο για τα χαρακτηριστικά εισαγωγής για την περίπτωση αυτή, καθώς η συμπεριφορά της στα ερωτήματα είναι ίδια με αυτή της Περίπτωσης 4. Το μεγάλο πλεονέκτημα της αναδρομικής δρομολόγησης είναι ότι προετοιμάζει το έδαφος για την κρυφή αποθήκευση bits, η οποία εξετάζεται στην Περίπτωση 7. Εξετάζουμε περαιτέρω την επιρροή των συντομεύσεων δρομολόγησης σε αυτές τις ρυθμίσεις στην Περίπτωση 8. Κατόπιν, εξετάζουμε το αποτέλεσμα της διατήρησης αντιγράφων στους γείτονες του κόμβου-παραλήπτη τόσο στο κόστος αριθμού βημάτων όσο και στην κατανομή φόρτου ερωτημάτων και εισαγωγών. Η Περίπτωση 9 χτίζει πάνω από την Περίπτωση 6, θέτοντας το όριο επαναλήψεων στην τιμή 0 και δημιουργώντας αντιθέτως 5 αντίγραφα ανά μαζική εισαγωγή, όπως εξηγήσαμε στην ενότητα 4.3.4. Ως τελευταίο βήμα (Περίπτωση 10), ενεργοποιούμε τις συντομεύσεις δρομολόγησης σε αυτό το σύνολο ρυθμίσεων και θέτουμε το όριο επαναλήψεων ξανά στην αρχική τιμή του των 5 βημάτων, για να εξετάσουμε το αποτέλεσμα των συντομεύσεων σε ένα σύνολο ρυθμίσεων με αντίγραφα και επαναλήψεις.

Χαρακτηριστικά Εισαγωγών Το πρώτο σύνολο αποτελεσμάτων αφορά τους αλγόριθμους και τα πρωτόκολλα εισαγωγής αντικειμένων των DHS. Πρώτα εξετάζουμε το κόστος βημάτων εισαγωγής αντικειμένων (ανά αντικείμενο) και ύστερα κοιτάμε την κατανομή του φόρτου εισαγωγής στους κόμβους στο δίκτυο επικάλυψης.

Κόστος Αριθμού Βημάτων Εισαγωγών Στα σχήματα 5.5(α') και 5.5(γ') σχεδιάζουμε το κόστος αριθμού βημάτων εισαγωγής ανά αντικείμενο για τις Περιπτώσεις 1 και 4 αντίστοιχα. Καθώς στις περιπτώσεις αυτές κάθε εισαγωγή αντικειμένου έχει ως αποτέλεσμα την αποθήκευση πληροφορίας bits DHS σε έναν (πιθανώς) διαφορετικό κόμβο κάθε φορά, το κόστος αριθμού βημάτων ακολουθεί στενά το αναμενόμενο μέσο μήκος διαδρομής του δομημένου δικτύου επικάλυψης (περίπου $\approx 0.5 \cdot \log N$ βήματα σε ένα δίκτυο N κόμβων διαμέτρου $O(\log N)$ [121]). Τα σχήματα 5.5(β') και 5.5(δ'), από την άλλη, δείχνουν το κόστος αριθμού βημάτων για τις Περιπτώσεις 2 και 5 αντίστοιχα, που αντιστοιχούν στις Περιπτώσεις 1 και 4 αντίστοιχα με ενεργοποιημένη την δρομολόγηση σύντομων διαδρομών. Όπως μπορούμε να δούμε, τώρα η εισαγωγή αντικειμένων στην DHS απαιτεί κατά μέσο όρο ≈ 2.3 βήματα, ανεξάρτητα από τον αριθμό των κόμβων στο δίκτυο επικάλυψης. Αυτό σημαίνει χονδρικά ότι, όταν στέλνουμε ένα μήνυμα προς έναν τυχαίο κόμβο στο διάστημα προορισμού, βρίσκουμε μία συντόμευση προς το διάστημα αυτό το πολύ στο δεύτερο βήμα προς τον κόμβο-παραλήπτη. Αυτό έχει ως αποτέλεσμα να μειώνεται δραματικά το κόστος αριθμού βημάτων, αλλά επηρεάζει εμφανώς αρνητικά την κατανομή φόρτου, όπως θα δούμε παρακάτω.

Το Σχήμα 5.5(ε') (Περίπτωση 6) δείχνει γιατί οι μαζικές εισαγωγές είναι η καλύτερη επιλογή όταν μπορούμε/πρέπει να εισάγουμε πολλά αντικείμενα κάθε φορά (όπως συνήθως συμβαίνει σε σενάρια διαχείρισης δεδομένων). Όπως μπορούμε να δούμε, το αναμενόμενο μικρό κόστος αριθμού βημάτων εισαγωγής επαληθεύεται ξεκάθαρα: το μέσο κόστος εισαγωγής αριθμού βημάτων ανά αντικείμενο έχει πέσει στα 0.12–0.18 βήματα: μία μείωση κατά έναν παράγοντα 40 έως 45 φορές σε σύγκριση με τον αλγόριθμο του DHS_0 και 13 έως 18 φορές σε σύγκριση ακόμα και με τις περιπτώσεις με συντομεύσεις δρομολόγησης. Τα νούμερα αυτά μεταφράζονται σε 30–45 βήματα (ή 6–7 DHT lookups) για να εισάγει ένας οποιοσδήποτε κόμβος όλα τα αντικείμενά του στην DHS. Η Περίπτωση 8 προσθέτει συντομεύσεις δρομολόγησης στην προηγούμενη Περίπτωση και μειώνει ακόμα περισσότερο το κόστος αριθμού βημάτων σε περίπου 0.05 βήματα ανά αντικείμενο, μία μείωση κατά έναν παράγοντα περίπου 2.5–3.5 φορές. Αυτό σημαίνει ότι, για να προσθέσει ένας οποιοσδήποτε κόμβος όλα τα αντικείμενά του στην DHS, θα χρειαστεί περίπου 12.5 βήματα ή 2–3 DHT lookups.

Κατόπιν εξετάζουμε την επιρροή των αντιγράφων στις περιπτώσεις μαζικών εισαγωγών. Με έναν παράγοντα αντιγραφής 5, χρησιμοποιώντας τον αλγόριθμο δημιουργίας αντιγράφων στην γειτονιά προορισμού που περιγράφηκε στην ενότητα 4.3.4, το κόστος εισαγωγής ανά αντικείμενο

(α) Περίπτωση 1 (DHS_0)

(β) Περίπτωση 2 (DHS_0 + Συντομεύσεις Δρομολόγησης)

(γ) Περίπτωση 4 (DHS_0 + Αναδρομική Δρομολόγηση)

(δ) Περίπτωση 5 (Περίπτωση 4 + Συντομεύσεις Δρομολόγησης)

(ε) Περίπτωση 6 (Περίπτωση 4 + Βυλκ Ινσέρτιονς)

(ς) Περίπτωση 8 (Περίπτωση 6 + Συντομεύσεις Δρομολόγησης)

(ζ) Περίπτωση 9 (Περίπτωση 6 + 5-Αντίγραφα)

(η) Περίπτωση 10 (Περίπτωση 9 + Συντομεύσεις Δρομολόγησης)

Σχήμα 5.5: Μέσος Αριθμός Βημάτων Εισαγωγής ανά Αντικείμενο Εισόδου

σχεδόν διπλασιάζεται, όπως μπορούμε να δούμε στο Σχήμα 5.5(ζ) για την Περίπτωση 9. Το τελικό αποτέλεσμα είναι ότι πλέον χρειάζονται περίπου 60–90 βήματα (ή 12–14 DHT lookups) για να εισάγει κάθε κόμβος όλα τα αντικείμενά του στην DHS. Όταν προσθέτουμε και συντομεύσεις δρομολόγησης στην εικόνα (Περίπτωση 10), τα νούμερα αυτά μειώνονται σε περίπου 0.2 βήματα ανά αντικείμενο ή 50 βήματα (10 DHT lookups) ανά κόμβο. Αυτή η μικρότερου μεγέθους μείωση (κατά έναν παράγοντα $1.2\text{--}1.8\times$) σε σύγκριση με την αντίστοιχη μείωση της προηγούμενης περίπτωσης, οφείλεται στο γεγονός ότι οι συντομεύσεις δρομολόγησης μπορούν μόνο να βοηθήσουν με τις μακρινές δρομολογήσεις (π.χ. όταν στέλνουμε δεδομένα από έναν κόμβο σε κάποιον άλλο σε ένα απομακρυσμένο διάστημα του χώρου αναγνωριστικών), αλλά δεν μπορεί να αποφύγει τα βήματα προς τους γείτονες του κόμβου προορισμού που θα γίνουν κατά τη δημιουργία αντιγράφων.

Τα νούμερα αριθμού βημάτων που αναφέρουμε παραπάνω είναι στην πραγματικότητα πολύ μικρά, αποδεικνύοντας έτσι το μικρό κόστος εισαγωγής της λύσης μας. Ως σημείο αναφοράς, σημειώνουμε ότι από μόνη της η περιοδική διεργασία σταθεροποίησης/συντήρησης των πινάκων δρομολόγησης των κόμβων στα περισσότερα δημοφιλή δίκτυα επικάλυψης DHT, απαιτούν περίπου 160 lookups κάθε λίγα (π.χ. 15) λεπτά, συν το (πολύ συχνότερο) κόστος συντήρησης των συνδέσμων προς γειτονικούς στον χώρο αναγνωριστικών κόμβων. Το γεγονός αυτό μας επιτρέπει να εφαρμόσουμε την τεχνική απόκρυψης της ενότητας 4.3.5 για να κρύψουμε τελείως το κόστος εισαγωγής/συντήρησης των DHS στην διαχειριστική κίνηση του DHT.

Χαρακτηριστικά Φόρτου Εισαγωγής Υπενθυμίζουμε ότι για τις Περιπτώσεις 1, 2, 4 και 5, η οποίες χρησιμοποιούν μεμονωμένες εισαγωγές, κάθε εισαγωγή αντικειμένου έχει ως αποτέλεσμα ένας (πιθανότατα διαφορετικός) κόμβος προορισμού να αποθηκεύει πληροφορία DHS bit και άρα ο ανά αντικείμενο φόρτος είναι ακριβώς 1. Αυτό φαίνεται ξεκάθαρα στα σχήματα 5.6(α'), 5.6(β'), 5.6(γ') και 5.6(δ') αντίστοιχα. Οι μαζικές εισαγωγές (Περιπτώσεις 6 και 8) μειώνουν τα νούμερα αυτά σε περίπου 0.033 (σχήματα 5.6(ε') και 5.6(φ')) – μία μείωση κατά έναν παράγοντα περίπου $30\times$ · αυτό είναι άμεση απόρροια του ότι μόνο ένα μήνυμα εισαγωγής χρειάζεται για να προσθέσει ένας οποιοσδήποτε κόμβος όλα τα αντικείμενά του στην DHS, έτσι όσο περισσότερα αντικείμενα συμπεριλαμβάνονται ανά μαζική εισαγωγή τόσο μικρότερος ο φόρτος ανά αντικείμενο. Τα παραπάνω νούμερα μεταφράζονται σε ένα συνολικό φόρτο περίπου 8 ανά κόμβο· με άλλα λόγια, κάθε κόμβος αποθηκεύει δεδομένα σε περίπου 8 άλλους κόμβους ή αντίστροφα ότι κάθε κόμβος ήταν ο κόμβος προορισμού για 8 περίπου εισαγωγές. Τέλος, με έναν παράγοντα αντιγραφής ίσο με 5, ο φόρτος εισαγωγής προφανώς πολλαπλασιάζεται με περίπου 5, όπως μπορούμε να δούμε στα σχήματα 5.6(ζ) και 5.6(η') για τις Περιπτώσεις 9 και 10 αντίστοιχα, αλλά παραμένει ακόμα πολύ χαμηλός: περίπου 45 «χτυπήματα» για το δίκτυο επικάλυψης 1000 κόμβων και περίπου 50 «χτυπήματα» για το δίκτυο των 10000 κόμβων.

Τέλος εξετάζουμε την κατανομή του φόρτου εισαγωγής στους κόμβους του δικτύου επικάλυψης. Χρησιμοποιούμε τον Συντελεστή Gini, όπως προτάθηκε στο [103] και περιγράφηκε νωρίτερα. Στις Περιπτώσεις 1 και 4, η κατανομή του φόρτου εισαγωγής ήταν άψογη, δίνοντας μία μέση τιμή 0.4 του Συντελεστή Gini (σχήματα 5.7(α') και 5.7(γ')), η οποία αυξάνεται σε περίπου 0.6–0.7 όταν ενεργοποιούμε τη δρομολόγηση σύντομης διαδρομής (σχήματα 5.7(β') και 5.7(δ')). Η Περίπτωση 6, η οποία χρησιμοποιεί μαζικές εισαγωγές, δίνει μία ελαφρώς υψηλότερη τιμή GC (σχήμα 5.7(ε')), μόλις πάνω από 0.73. Η Περίπτωση 8, η οποία προσθέτει και συντομεύσεις δρομολόγησης, παράγει τον ελαφρώς χειρότερο μέσο βαθμό GC ≈ 0.9 (σχήμα 5.7(φ')). Η δημιουργία αντιγράφων βελτιώνει την τιμή του GC σε περίπου 0.69 κατά μέσο όρο, το οποίο ανεβαίνει σε 0.7–0.8 με συντομεύσεις δρομολόγησης (σχήματα 5.7(ζ) και 5.7(η') αντίστοιχα).

Χαρακτηριστικά Ερωτημάτων Το επόμενο σύνολο αποτελεσμάτων σχετίζεται με την ίδια την εκτίμηση (επερώτηση) και τους αντίστοιχους αλγόριθμους και πρωτόκολλα των DHS. Όπως και στην περίπτωση των εισαγωγών, πρώτα εξετάζουμε το κόστος ερωτημάτων αριθμού βημάτων και

(α) Περίπτωση 1 (DHS_0)

(β) Περίπτωση 2 (DHS_0 + Συνοτμεύσεις Δρομολόγησης)

(γ') Περίπτωση 4 (DHS_0 + Αναδρομική Δρομολόγησης)

(δ') Περίπτωση 5 (Περίπτωση 4 + Συνοτμεύσεις Δρομολόγησης)

(ε') Περίπτωση 6 (Περίπτωση 4 + Βυλκ Ινσερτιονς)

(ε') Περίπτωση 8 (Περίπτωση 6 + Συνοτμεύσεις Δρομολόγησης)

(ζ') Περίπτωση 9 (Περίπτωση 6 + 5-Αντίγραφα)

(η') Περίπτωση 10 (Περίπτωση 9 + Συνοτμεύσεις Δρομολόγησης)

Σχήμα 5.6: Μέσος Φόρτος Εισαγωγής ανά Αντικείμενο Εισόδου

(α) Περίπτωση 1 (DHS_0)

(β) Περίπτωση 2 (DHS_0 + Συνοτμεύσεις Δρομολόγησης)

(γ') Περίπτωση 4 (DHS_0 + Αναδρομική Δρομολόγηση)

(δ) Περίπτωση 5 (Περίπτωση 4 + Συνοτμεύσεις Δρομολόγησης)

(ε) Περίπτωση 6 (Περίπτωση 4 + Βυλκ Ινσερτιονς)

(ς) Περίπτωση 8 (Περίπτωση 6 + Συνοτμεύσεις Δρομολόγησης)

(ζ) Περίπτωση 9 (Περίπτωση 6 + 5-Αντίγραφα)

(η) Περίπτωση 10 (Περίπτωση 9 + Συνοτμεύσεις Δρομολόγησης)

Σχήμα 5.7: Κατανομή Φόρτου Εισαγωγής: Συντελεστής Gini

στη συνέχεια κοιτάμε την κατανομή του φόρτου αυτού στους κόμβους του δικτύου. Τέλος, εξετάζουμε και την ακρίβεια των υπολογιζόμενων εκτιμήσεων, σε σύγκριση με αντίστοιχες εκτιμήσεις σε κεντρικοποιημένο περιβάλλον (δηλ. αν όλα τα αντικείμενα ήταν αποθηκευμένα σε έναν κόμβο).

Κόστος Αριθμού Βημάτων Ερωτημάτων Όσον αφορά τα ερωτήματα, οι Περιπτώσεις 1 και 3 (σχήματα 5.8(α') και 5.8(γ')) χρειάζονται κατά μέσο όρο περίπου 50–70 βήματα (δηλ. το ισοδύναμο περίπου 10 DHT lookups) για να υπολογίσουν την εκτίμηση με 128 διανύσματα και περίπου 160–250 βήματα (ή 32–40 DHT lookups) με 512 διανύσματα (σχήματα 5.8(α') και 5.8(δ')), με το μεγαλύτερο κόστος της δεύτερης περίπτωσης να οφείλεται στις περισσότερες επαναλήψεις· με περισσότερα διανύσματα στην DHS, λιγότερα bits παίρνουν την τιμή 1 ανά διάνυσμα και επομένως λιγότερες πλειάδες DHS διασκορπίζονται στο δίκτυο και έτσι η πιθανότητα να επιλέξουμε τυχαία κάποιον κόμβο που δεν αποθηκεύει σχετική πληροφορία αυξάνεται. Οι συντομεύσεις δρομολόγησης (Περίπτωση 2) μειώνουν τα νούμερα αυτά στα 40–50 βήματα για 128 διανύσματα και 150–250 βήματα για 512 βήματα. Τα νούμερα αυτά επαληθεύουν την ανάλυσή μας για κόστος αριθμού βημάτων ερωτημάτων $O(\lambda \log N)$ στην χειρότερη περίπτωση.

Το σχήμα 5.8(δ') δείχνει τον αριθμό βημάτων ανά ερώτημα για την Περίπτωση 4. Τώρα οι εκτιμήσεις παίρνουν κατά μέσο όρο περίπου 80–120 βήματα για 128 διανύσματα (η το ισοδύναμο 16–18 DHT lookups), με την περίπτωση των 512 διανυσμάτων να πέφτει στα 100–160 βήματα (ή 20–32 DHT lookups). Οι συντομεύσεις δρομολόγησης (Περίπτωση 5) μειώνει κι άλλο τα νούμερα αυτά σε 40–60 βήματα για 128 διανύσματα και 65–90 βήματα για 512 διανύσματα (σχήμα 5.8(ε')).

Κατόπιν, εξετάζουμε τις Περιπτώσεις 9 και 10 (σχήματα 5.8(ε') και 5.8(ζ')), οι οποίες χρησιμοποιούν τους αλγόριθμους δημιουργίας αντιγράφων, μαζί με συντομεύσεις δρομολόγησης για την δεύτερη περίπτωση. Όπως μπορούμε να δούμε, η δημιουργία αντιγράφων έχει θετική επιρροή και για το κόστος των επερωτήσεων, καθώς υπάρχει μία μείωση της τάξης του 30%–40% για την Περίπτωση 9 σε σχέση με την Περίπτωση 4 (την ισοδύναμη περίπτωση χωρίς συντομεύσεις δρομολόγησης). Από την άλλη, το κόστος βημάτων της Περιπτώσεως 5 (δηλ. της Περιπτώσεως 4 συν συντομεύσεις δρομολόγησης) είναι ήδη όσο χαμηλό μπορούμε να επιτύχουμε με αυτά τα εφόδια· επομένως η δημιουργία αντιγράφων σε αυτή την περίπτωση δεν έχει σχεδόν κανένα αποτέλεσμα στο μέσο πλήθος βημάτων, όπως μπορούμε να δούμε στο σχήμα 5.8(ζ') σε σχέση με το σχήμα 5.8(ε').

Αφήσαμε τελευταίες τις περιπτώσεις που χρησιμοποιούν κρυφή αποθήκευση των bits των αποτελεσμάτων (δηλ. τις Περιπτώσεις 7 και 8). Τα σχήματα 5.9(α') και 5.9(β') αντίστοιχα δείχνουν το μέσο αριθμό βημάτων ερωτημάτων για τις δύο αυτές περιπτώσεις, στο πέρασμα του χρόνου. Όπως μπορούμε να δούμε, υπάρχει μία απότομη πτώση στο μέσο αριθμό βημάτων για τα πρώτα 25–50 ερωτήματα καθώς η cache «ξεσταίνεται», ενώ στη συνέχεια ο αριθμός αυτός σταθεροποιείται σε περίπου 5–20 βήματα για την Περίπτωση 7 και περίπου 1 βήμα για την Περίπτωση 8 ύστερα από 500 ερωτήματα. Θα θέλαμε να σημειώσουμε ότι δεν υπάρχει κάποιου είδους αλγόριθμος αντικατάστασης στα πειράματα αυτά. Αυτή ήταν ηβελημένη επιλογή, έτσι ώστε να δείξουμε την επιρροή της κρυφής αποθήκευσης στον αριθμό βημάτων ερωτημάτων (και στην κατανομή φόρτου ερωτημάτων, όπως θα δούμε σύντομα). Αν υποθέσουμε ότι θέτουμε ένα διάστημα «ζωής» για τα αντικείμενα στην cache του κάθε κόμβου και δεδομένου ενός ρυθμού άφιξης ερωτημάτων, μπορούμε να υπολογίσουμε πόσα ερωτήματα θα απαντηθούν κατά μέσο όρο από την cache και έτσι να βρούμε απευθείας τα αντίστοιχα κέρδη στον αριθμό βημάτων από τα σχήματα αυτά.

Κατανομή Φόρτου Ερωτημάτων Τα σχήματα 5.10 και 5.11 δείχνουν τον συνολικό φόρτο που κάθε ερώτημα προσθέτει στο σύστημα, για διάφορους συνδυασμούς παραμέτρων (υπολογιζόμενος ως ο μέσος όρος ανά «παράθυρο» 25 ερωτημάτων). Ο κοινός παρονομαστής όλων των περιπτώσεων που δεν χρησιμοποιούν κρυφή αποθήκευση, εκτός από την Περίπτωση 9 (δηλ. τα σχήματα 5.10(α')–5.10(ε') και 5.10(ζ')) είναι ότι ο συνολικός φόρτος είναι μεγαλύτερος με 512 bitmaps από όταν χρησιμοποιούνται 128 bitmaps. Ο λόγος για τον οποίο παρατηρείται αυτό το φαινόμενο είναι

(α) Περίπτωση 1 (DHS_0)

(β) Περίπτωση 2 (DHS_0 + Συντομεύσεις Δρομολόγησης)

(γ) Περίπτωση 3 (DHS_0 + Διαδικός Έλεγχος Θέσεων)

(δ) Περίπτωση 4 (DHS_0 + Αναδρομική Δρομολόγηση)

(ε) Περίπτωση 5 (Περίπτωση 4 + Συντομεύσεις Δρομολόγησης)

(ρ) Περίπτωση 9 (Περίπτωση 4 + 5-Αντίγραφα/0-Επαναλήψεις)

(ζ) Περίπτωση 10 (Περίπτωση 9 + Συντομεύσεις Δρομολόγησης)

Σχήμα 5.8: Μέσος Αριθμός Βημάτων ανά Ερώτημα – Περιπτώσεις χωρίς Λανθάνουσα Αποθήκευση Bits Αποτελεσμάτων

(α) Περίπτωση 7 (Περίπτωση 4 + Λανθάνουσα Αποθήκευση Αποτελεσμάτων)

(β) Περίπτωση 8 (Περίπτωση 7 + Συντομεύσεις Δρομολόγησης)

Σχήμα 5.9: Μέσος Αριθμός Βημάτων ανά Ερώτημα – Περιπτώσεις με Λανθάνουσα Αποθήκευση Αποτελεσμάτων

ο ίδιος για τον οποίο παρατηρούμε και μεγαλύτερο αριθμό βημάτων για τις περιπτώσεις αυτές: περισσότερα bitmaps έχουν ως αποτέλεσμα λιγότερα αντικείμενα να ανατίθενται σε κάθε bitmap και αυτό με τη σειρά του αυξάνει την πιθανότητα να επισκεφθούμε έναν κόμβο ο οποίος δεν αποθηκεύει σχετική πληροφορία, κάτι που μεταφράζεται σε επαναλήψεις. Αυτός είναι επίσης ο λόγος για τον οποίο η Περίπτωση 9 στο σχήμα 5.10(ε'), έχοντας μηδενικό όριο επαναλήψεων, δεν υποφέρει από αυτό το πρόβλημα, ενώ αντιθέτως κατορθώνει να έχει έναν αρκετά χαμηλότερο συνολικό φόρτο ανά ερώτημα αλλά ελαφρώς χειρότερη ακρίβεια, όπως θα δούμε σύντομα. Τέλος, τα σχήματα 5.11(α') και 5.11(β') δείχνουν το φόρτο ανά ερώτημα για τις περιπτώσεις με κρυφή αποθήκευση (Περίπτωση 7 και 8 αντίστοιχα). Τα παραπάνω συμπεράσματα για την επίδραση του αριθμού των bitmaps και των συντομεύσεων δρομολόγησης στον φόρτο κάθε ερωτήματος ισχύουν και εδώ, αλλά τώρα βλέπουμε επιπλέον τον τρόπο με τον οποίο η κρυφή αποθήκευση μειώνει μεσοπρόθεσμα τον φόρτο ανά ερώτημα. Μπορούμε ακόμα να δούμε ότι οι συντομεύσεις δρομολόγησης δεν επηρεάζουν την απόδοση του DHS όσον αφορά αυτή τη μετρική: αν και μειώνουν τον αριθμό των βημάτων για τις μακρινές αναζητήσεις, δεν μεταβάλλουν τον αριθμό των κόμβων που πρέπει να επισκεφθεί κάθε ερώτημα.

Θα εξετάσουμε τώρα την επίδραση των διάφορων παραμέτρων στην κατανομή του φόρτου των ερωτημάτων στους κόμβους του δικτυακού υποστρώματος. Τα πρώτα τρία σχήματα δείχνουν την εξέλιξη του δείκτη GC της κατανομής του φόρτου ερωτημάτων καθώς ερωτήματα εκτελούνται στο σύστημα, για τις Περιπτώσεις 1, 2 και 3 (σχήματα 5.12(α'), 5.12(β') και 5.12(γ') αντίστοιχα). Απαιτούνται περίπου 100 με 200 ερωτήματα για να σταθεροποιηθεί ο δείκτης GC για τις περιπτώσεις με 512 bitmaps (οι καμπύλες για τις περιπτώσεις με 128 bitmaps είναι κοντά στο 1 αλλά πέφτουν στο 0.9-0.95 ύστερα από περίπου 2000 ερωτήματα, κάτι που δεν δείχνουμε σε σχήμα για λόγους παρουσίασης). Ο λόγος για τον οποίο οι περιπτώσεις αυτές έχουν το προβάδισμα οφείλεται στο γεγονός ότι λιγότερα bits παίρνουν την τιμή 1 σε κάθε bitmap του DHS, έχοντας έτσι ως αποτέλεσμα ο έλεγχος των θέσεων των bits να τερματίζει νωρίτερα και σε μεγαλύτερα τμήματα του χώρου αναγνωριστικών: σημειώνουμε εδώ ότι τα τελευταία (πιο σημαντικά) bits αντιστοιχίζονται σε λίγους κόμβους (με το πιο σημαντικό bits να αποθηκεύεται σε έναν ή δύο κόμβους). Ωστόσο, αυτή η κατάσταση μπορεί εύκολα να αποφευχθεί χρησιμοποιώντας όρια (thresholding – βλπ. ενότητα 4.6). Ακόμα, σημειώνουμε ότι (α) οι συντομεύσεις δρομολόγησης έχουν αρνητική επίδραση στην δίκαιη κατανομή φόρτου και (β) οι δύο περιπτώσεις που δεν χρησιμοποιούν συντομεύσεις καταφέρνουν τιμή του δείκτη GC της τάξης του 0.73-0.85, η οποία είναι εφάμιλλη της αντίστοιχης τιμής της κατανομής φόρτου που καταφέρνουν δημοφιλή υποστρώματα DHT υπό προϋποθέσεις ομοιόμορφης κατανομής τιμών και δημοτικότητας των αντικειμένων (≈ 0.7).

Η κατανομή του φόρτου για τις Περιπτώσεις 4 και 5 φαίνεται στα σχήματα 5.13(α') και 5.13(β'). Σε σύγκριση με τις προηγούμενες περιπτώσεις, μπορούμε να δούμε ότι η αναδρομική δρομολόγηση βελτιώνει αισθητά την εξισορρόπηση φόρτου: το σύστημα τώρα καταφέρνει τιμή δείκτη GC ίση με 0.87 στην χειρότερη περίπτωση (ακόμα και με 128 bitmaps) και 0.68 στην καλύτερη περίπτωση (σχήμα 5.13(α')). Ακόμα, σημειώνουμε ότι η κλίση της καμπύλης για τις περιπτώσεις με 128 bitmaps (ειδικά για την Περίπτωση 4) είναι ακόμα σχετικά μεγάλη στο σημείο των 500 ερωτημάτων, που σημαίνει ότι η τιμή του δείκτη GC δεν έχει σταθεροποιηθεί ακόμα. Εξετάσαμε περαιτέρω τις περιπτώσεις αυτές για αρκετές χιλιάδες ερωτημάτων (δεν δείχνουμε αντίστοιχο σχήμα χάριν παρουσίασης) και βρήκαμε ότι ο δείκτης GC καταλήγει περίπου στην τιμή 0.65 για 512 bitmaps και 0.75 για 128 bitmaps.

Η επίδραση των αντιγράφων (replication) στην κατανομή φόρτου φαίνεται στα σχήματα 5.14(α') και 5.14(β'), στα οποία σχεδιάζεται η τιμή του δείκτη GC για τις Περιπτώσεις 9 και 10 αντίστοιχα. Για την τελευταία, η τιμή του δείκτη GC που επιτυγχάνεται για την κατανομή του φόρτου ερωτημάτων είναι στο διάστημα 0.82 έως 0.91 για 512 bitmaps και από 0.93 έως 0.96 για 128 bitmaps. Για την Περίπτωση 9, βλέπουμε καθαρά ότι η καμπύλη της τιμής του δείκτη GC είναι ακόμα αρκετά απότομη στο σημείο των 500 ερωτημάτων: ύστερα από μερικές εκατοντάδες

(α) Περίπτωση 1 (DHS₀)

(β) Περίπτωση 2 (DHS₀ + Συντομεύσεις Δρομολόγησης)

(γ) Περίπτωση 3 (DHS₀ + Διαδικός Έλεγχος Θέσεων)

(δ) Περίπτωση 4 (DHS₀ + Αναδρομική Δρομολόγηση)

(ε) Περίπτωση 5 (Περίπτωση 4 + Συντομεύσεις Δρομολόγησης)

(ρ) Περίπτωση 9 (Περίπτωση 4 + 5-Αντίγραφα/0-Επαναλήψεις)

(ζ) Περίπτωση 10 (Περίπτωση 9 + Συντομεύσεις Δρομολόγησης)

Σχήμα 5.10: Μέσος Φόρτος Κόμβων ανά Ερώτημα – Περιπτώσεις χωρίς Λανθάνουσα Αποθήκευση Αποτελεσμάτων

(α) Περίπτωση 7 (Περίπτωση 4 + Λανθάνουσα Αποθήκευση Αποτελεσμάτων)

(β) Περίπτωση 8 (Περίπτωση 7 + Συντομεύσεις Δρομολόγησης)

Σχήμα 5.11: Μέσος Φόρτος Κόμβων ανά Ερώτημα - Περιπτώσεις με Λανθάνουσα Αποθήκευση Αποτελεσμάτων

(α) Περίπτωση 1 (DHS₀)

(β) Περίπτωση 2 (DHS₀ + Συνομιτούσεις Δρομολόγησης)

(γ) Περίπτωση 3 (DHS₀ + Διαδικός Έλεγχος Θέσεων)

Σχήμα 5.12: Κατανομή Φόρτου Ερωτημάτων (Επαναληπτική Δρομολόγηση) – Συντελεστής Gini

(α) Περίπτωση 4 (DHS₀ + Αναδρομική Δρομολόγηση)

(β) Περίπτωση 5 (Περίπτωση 4 + Συντομεύσεις Δρομολόγησης)

Σχήμα 5.13: Κατανομή Φόρτου Ερωτημάτων (Αναδρομική Δρομολόγηση) - Συντελεστής Gini

(α) Περίπτωση 9 (Περίπτωση 4 + 5-Αντίγραφα/0-Επαναλήψεις)

(β) Περίπτωση 10 (Περίπτωση 9 + Συντομεύσεις Δρομολόγησης)

Σχήμα 5.14: Κατανομή Φόρτου Ερωτημάτων (Διατήρηση Αντιγράφων) – Συντελεστής Gini

ερωτημάτων η τιμή αυτή σταθεροποιείται στα ίδια περίπου επίπεδα με αυτά τις Περίπτωσης 4 παραπάνω. Και πάλι είναι προφανής η ευεργετική επίδραση της αναδρομικής δρομολόγησης (σε σύγκριση για παράδειγμα με την Περίπτωση 1) καθώς και η αρνητική επίδραση της χρήσης συντομεύσεων δρομολόγησης στην κατανομή φόρτου.

Αφήσαμε και πάλι τις περιπτώσεις που χρησιμοποιούν κρυφή αποθήκευση για το τέλος. Η κρυφή αποθήκευση είναι (μαζί με το thresholding) ο καλύτερος τρόπος κατανομής του φόρτου στους κόμβους. Στο σχήμα 5.15(α) βλέπουμε την τιμή του δείκτη GC για την Περίπτωση 7. Όπως φαίνεται, οι περιπτώσεις με 512 bitmaps φτάνουν σε τιμή του GC της τάξης του 0.45 ύστερα από 500 ερωτήματα, με τις περιπτώσεις με 128 bitmaps να έχουν φτάσει στο 0.55-0.75 στο αντίστοιχο σημείο, ενώ και οι δύο κατηγορίες καταλήγουν σε μία τιμή της τάξης του 0.4 ύστερα από μερικές ακόμα εκατοντάδες ερωτημάτων. Ακόμα και χρησιμοποιώντας συντομεύσεις δρομολόγησης (σχήμα 5.15(β)), ο δείκτης GC φτάνει σχεδόν στην τιμή 0.55-0.66 ύστερα από 500 ερωτήματα για 512 bitmaps και περίπου στο 0.75-0.85 για 128 bitmaps (και οι δύο κατηγορίες καταλήγουν περίπου

Α/Α	$\beta = 128$		$\beta = 512$	
	Ομοιόμορφη	Zipf 1.2	Ομοιόμορφη	Zipf 1.2
Περίπτωση 1	0.00	0.00	0.00	0.00
Περίπτωση 2	0.01	0.00	0.26	0.25
Περίπτωση 3	0.03	0.04	0.24	0.23
Περίπτωση 4	0.00	0.00	0.00	0.00
Περίπτωση 5	0.17	0.12	0.54	0.52
Περίπτωση 6	0.00	0.02	0.04	0.05
Περίπτωση 7	0.05	0.01	0.01	0.08
Περίπτωση 8	1.62	1.54	2.33	2.41
Περίπτωση 9	0.87	0.92	2.54	2.48
Περίπτωση 10	2.11	2.56	3.27	3.49

Πίνακας 5.2: Μέσο Λάθος Εκτίμησης (%)

στο 0.55 ύστερα από μερικές ακόμα εκατοντάδες ερωτημάτων).

Ακρίβεια Εκτίμησης Τέλος, εξετάζουμε την ακρίβεια των υπολογιζόμενων εκτιμήσεων σε σχέση με έναν κεντριοποιημένο εκτιμητή. Στις περισσότερες περιπτώσεις το μέσο σφάλμα ήταν κάτω του 1%, ενώ έφτανε μέχρι και 3.5% στη χειρότερη περίπτωση, αποδεικνύοντας έτσι τον ισχυρισμό μας για διαφάνεια κατανομής. Ο πίνακας 5.2 συνοψίζει τα αποτελέσματά μας για διάφορους συνδυασμούς παραμέτρων συστήματος. Σημειώνουμε την αρνητική επίπτωση των συντομεύσεων δρομολόγησης (τα αντίστοιχα νούμερα φαίνονται με πλάγια γράμματα) και το ελαφρώς μεγαλύτερο σφάλμα στις περιπτώσεις με 512 διανύσματα σε σχέση με αυτές με 128 διανύσματα εξαιτίας του κάπως χαμηλού ορίου αριθμού επαναλήψεων.

5.2.3 Συναθροιστικά Ερωτήματα

Σε αυτό το τμήμα της πειραματικής μελέτης μας, ενδιαφερόμαστε κυρίως για το κόστος μηνυμάτων και την ακρίβεια των προσεγγίσεών μας ως βάσεις για τον υπολογισμό συναθροιστικών ερωτημάτων σε μία σχεσιακή βάση δεδομένων πάνω από ένα ΔΟΕ. Αρχικά δημιουργήσαμε πίνακες μίας στήλης με ακέραιες τιμές στο διάστημα $[0, 1000)$, ακολουθώντας είτε μία ομοιόμορφη κατανομή (συμβολίζεται σαν κατανομή Zipf με παράμετρο θ ίση με 0.0) ή μία κατανομή Zipf με παράμετρο θ ίση με 0.7, 1.0 και 1.2 ([51, 111]). Οι πίνακες περιείχαν 300,000 εγγραφές ο καθένας, κατανεμημένες σε όλους τους κόμβους το δικτύου. Τονίζουμε ότι χρησιμοποιούμε πίνακες μίας στήλης για λόγους παρουσίασης και ευκολίας στη μοντελοποίηση, ως μέσο επεξεργασίας ερωτημάτων σε στατιστικές δομές επί στηλών δεικτοδότησης (index attributes) ενός (πιθανότητα κατά πολύ) μεγαλύτερου πίνακα. Εξετάσαμε το σύστημά μας και για μεγαλύτερες περιπτώσεις (περισσότερες και μεγαλύτερες εγγραφές ανά πίνακα, τιμές κινητής υποδιαστολής, κτλ.) με παρόμοια αποτελέσματα.

Λάθη Εκτίμησης και Κόστη Μηνυμάτων Τα σχήματα 5.16(α') και 5.16(β') δείχνουν το μέσο λάθος της εκτίμησης για το ερώτημα Count χρησιμοποιώντας προσέγγιση «Ραντεβού» και DHS αντίστοιχα. Και στις δύο περιπτώσεις το όποιο λάθος προκαλείται από την χρήση των συνόψεων κατακερματισμού, έτσι και οι δύο προσεγγίσεις παρουσιάζουν το ίδιο μέσο σφάλμα. Όπως ήταν αναμενόμενο, όσο περισσότερα τα bitmaps της σύνοψης τόσο καλύτερη η ακρίβεια. Για την περίπτωση του ερωτήματος Sum το σφάλμα είναι ελαφρώς μεγαλύτερο λόγω του πρόσθετου σφάλματος προσέγγισης που εισαγάγει η χρήση της τεχνικής άθροισης των [22] (σχ. 5.16(δ) και 5.16(γ')).

Σχήμα 5.15: Κατανομή Φόρτου Ερωτημάτων (Λανθάνουσα Αποθήκευση Αποτελεσμάτων) – Συντελεστής Gini

(α) Ερώτημα Count – Προσέγγιση Ραντεβού

(β) Ερώτημα Count – Προσέγγιση DHS

(γ) Ερωτήματα Sum/Average – Προσέγγιση Ραντεβού

(δ) Ερωτήματα Sum/Average – Προσέγγιση DHS

Σχήμα 5.16: Μέσος Όρος % Λάθους Εκτίμησης για Ερωτήματα Count και Sum/Average σε Σχέση με Κεντρικοποιημένες Προσεγγίσεις

Τέλος, το μέσο σφάλμα για τα ερωτήματα Average ήταν σχεδόν το ίδιο με αυτό των ερωτημάτων Sum.

Μετρήσαμε επίσης και δείχνουμε τον μέσο αριθμό μηνυμάτων ανά κόμβο για την εισαγωγή όλων των πλειάδων στην καταμεμημένη σύνοψη, καθώς και τον αριθμό μηνυμάτων για τον υπολογισμό των ερωτημάτων. Τα σχήματα 5.17(α') και 5.17(β') δείχνουν το κόστος εισαγωγής για όλα τα ερωτήματα. Το σχήμα 5.18(α') δείχνει το κόστος μηνυμάτων/αλμάτων των ερωτημάτων για την προσέγγιση ραντεβού· η καμπύλη αυτή είναι ίδια για όλα τα ερωτήματα, καθώς σε κάθε περίπτωση υπάρχει ένας και μοναδικός κόμβος ραντεβού. Τέλος τα σχήματα 5.18(β'), 5.18(γ') και 5.18(δ') δείχνουν το κόστος αλμάτων για τον υπολογισμό των ερωτημάτων με την προσέγγιση DHS για τα ερωτήματα Count, Sum και Average αντίστοιχα. Όπως μπορούμε να δούμε, το κόστος μηνυμάτων ανά κόμβο είναι υψηλότερο για την προσέγγιση DHS κατά έναν παράγοντα περίπου $8 \times$ τόσο για την εισαγωγή αντικειμένων όσο και για τον υπολογισμό των ερωτημάτων. Δεδομένου ότι στην προσέγγιση ραντεβού κάθε κόμβος κάνει μία αναζήτηση στο DHT για να εισάγει όλες τις πλειάδες του και για να υπολογίσει το αποτέλεσμα της καταμεμημένης σύνοψης, το νούμερο αυτό μεταφράζεται σε περίπου 8 αναζητήσεις DHT ανά κόμβο για την προσέγγιση DHS. Ο αριθμός μηνυμάτων για τον υπολογισμό των ερωτημάτων Sum και Average στην περίπτωση της προσέγγισης DHS είναι μικρότερος σε σχέση με τα ερωτήματα Count (σχήματα 5.18(γ') και 5.18(δ')) καθώς περισσότερα bits τίθενται στην τιμή 1 ως αποτέλεσμα του αλγορίθμου των [22] και έτσι είναι ευκολότερο για τον αλγόριθμο αναζήτησης του DHS να εντοπίσει τα bits αυτά.

Κατανομή Φόρτου Το επιπλέον κόστος μηνυμάτων της προσέγγισης DHS μας αποζημιώνει στον τομέα της δίκαιης κατανομής φόρτου στους κόμβους του δικτύου. Τα σχήματα 5.19(α'), 5.19(β'), 5.20(α') και 5.20(β') δείχνουν τον συντελεστή Gini και τον συντελεστή Fairness Index του φόρτου

(α) Αριθμός μηνυμάτων εισαγωγής – Προσέγγιση Ραν- (β) Αριθμός μηνυμάτων εισαγωγής – Προσέγγιση DHS τεβού

Σχήμα 5.17: Αριθμός Μηνυμάτων/Αλμάτων για την Εισαγωγή Αντικειμένων στην Κατανεμημένη Σύνοψη Χρησιμοποιώντας Προσέγγιση Ραντεβού Χωρίς Διατήρηση Αντιγράφων και Προσέγγιση DHS

(α') Όλα τα ερωτήματα – Προσέγγιση Ραντεβού

(β') Ερωτήματα Count – Προσέγγιση DHS

(γ') Ερωτήματα Sum – Προσέγγιση DHS

(δ') Ερωτήματα Average – Προσέγγιση DHS

Σχήμα 5.18: Αριθμός Μηνυμάτων/Αλμάτων για τον Υπολογισμό των Ερωτημάτων Count, Sum και Average Χρησιμοποιώντας Προσέγγιση Ραντεβού Χωρίς Διατήρηση Αντιγράφων και Προσέγγιση DHS

(α) Συντελεστής Gini – Προσέγγιση Ραντεβού

(β) Συντελεστής Fairness Index – Προσέγγιση Ραντεβού

Σχήμα 5.19: Συντελεστής Gini (όσο μικρότερο τόσο καλύτερο) και Συντελεστής Fairness Index (όσο μεγαλύτερο τόσο καλύτερο – λογαριθμική κλίμακα στον κατακόρυφο άξονα) του Φόρτου των Κόμβων για την Εισαγωγή Πλειάδων στην Κατανομημένη Σύνοψη στην Προσέγγιση Ραντεβού

(α) Συντελεστής Gini – Προσέγγιση DHS

(β) Συντελεστής Fairness Index – Προσέγγιση DHS

Σχήμα 5.20: Συντελεστής Gini (όσο μικρότερο τόσο καλύτερο) και Συντελεστής Fairness Index (όσο μεγαλύτερο τόσο καλύτερο – λογαριθμική κλίμακα στον κατακόρυφο άξονα) του Φόρτου των Κόμβων για την Εισαγωγή Πλειάδων στην Κατανομημένη Σύνοψη για Ερωτήματα Count στην Προσέγγιση DHS

Σχήμα 5.21: Συντελεστής Gini (όσο μικρότερο τόσο καλύτερο) και Συντελεστής Fairness Index (όσο μεγαλύτερο τόσο καλύτερο) του Φόρτου των Κόμβων για την Εισαγωγή Πλειάδων στην Κατανομή Σύνοψη για Ερωτήματα Sum και Average στην Προσέγγιση DHS (λογαριθμική κλίμακα στον κατακόρυφο άξονα)

Σχήμα 5.22: Χρονική Εξέλιξη των Συντελεστών Gini και Fairness Index για τον Υπολογισμό των Τριών Ειδών Κατανομημένων Ερωτημάτων

στους κόμβους του δικτύου κατά την εισαγωγή των πλειάδων για το ερώτημα Count (τα σχήματα για την προσέγγιση ραντεβού είναι πρακτικά πανομοιότυπα και για τα τρία είδη ερωτημάτων που ελέγχουμε και επομένως επαναχρησιμοποιούμε τα σχήματα 5.19(α') και 5.19(β')). Υπενθυμίζουμε ότι ο φόρτος σε έναν οποιοδήποτε κόμβο μετράται ως το πλήθος των επισκέψεων στον κόμβο κατά τη διάρκεια της εισαγωγής ή/και του υπολογισμού του ερωτήματος. Παρατηρούμε μία βελτίωση δύο τάξεων μεγέθους περίπου και για τις δύο μετρικές. Αυτό ισχύει επίσης και για τα ερωτήματα Sum και Average (σχ. 5.21)· η προσέγγιση DHS υπερτερεί της προσέγγισης ραντεβού κατά έναν παράγοντα $\approx 200\times$ και $\approx 40\times$ όσον αφορά τις μετρικές GC και FI αντίστοιχα.

Τέλος, το σχήμα 5.22 δείχνει την χρονική εξέλιξη των συντελεστών Gini και Fairness Index καθώς ερωτήματα εκτελούνται στο σύστημα. Με την προσέγγιση ραντεβού υπάρχει ένας και μόνο κόμβος που επιβαρύνεται με όλο τον φόρτο των ερωτημάτων και επομένως οι παράγοντες GC και FI είναι σταθεροί στο χρόνο και περίπου ίσοι ή χειρότεροι από ≈ 0.999 και ≈ 0.001 αντίστοιχα. Η προσέγγιση DHS συγκλίνει στην τιμή ≈ 0.5 και για τις δύο μετρικές GC και FI , το οποίο ισοδυναμεί με τις τιμές που έχουν οι μετρικές αυτές για τις αποστάσεις ανάμεσα σε διαδοχικούς κόμβους στον χώρο τιμών των αναγνωριστικών και είναι επομένως οι καλύτερες εφικτές τιμές από έναν αλγόριθμο ο οποίος χρησιμοποιεί τυχαία ανάθεση αντικειμένων στους κόμβους του δικτύου.

(α') Συντελεστής Gini (όσο μικρότερο τόσο καλύτερο)

(β') Συντελεστής Fairness Index (όσο μεγαλύτερο τόσο καλύτερο)

Σχήμα 5.23: Συντελεστής Gini και Συντελεστής Fairness Index του Φόρτου των Κόμβων για την Εισαγωγή Αντικειμένων στην Κατανεμημένη Σύνοψη για το Ερώτημα Count για Συνεχώς Αυξανόμενο Αριθμό Κόμβων Ραντεβού που Διαμοιράζονται τον Κατανεμημένο Υπολογισμό (είτε μέσω αντιγράφων ή λόγων πολλαπλών παράλληλων ερωτημάτων)

(α') Απλή αντιγραφή - Κόστος μηνυμάτων επεξεργασίας ερωτημάτων

(β') Προορατική αντιγραφή - Κόστος μηνυμάτων εισαγωγής πλειάδων

Σχήμα 5.24: Κόστος Μηνυμάτων για το Ερώτημα Count Χρησιμοποιώντας Προσέγγιση Ραντεβού με 50 Κόμβους Ραντεβού

Προσέγγιση Ραντεβού με Αντίγραφα Οι υποστηρικτές της προσέγγισης ραντεβού ίσως πιστέψουν ότι οι συντελεστές GC και FI μπορούν να βελτιωθούν αν χρησιμοποιηθούν πολλαπλοί κόμβοι για την αποθήκευση της κατανεμημένης σύνοψης χρησιμοποιώντας κάποιο από τα σχήματα διατήρησης αντιγράφων που αναφέραμε νωρίτερα: απλή αντιγραφή (κάθε κόμβος επιλέγει τυχαία έναν από τους κόμβους ραντεβού και εισάγει όλα τα αντικείμενά του σε αυτόν) ή «προορατική» αντιγραφή (κάθε κόμβος εισαγάγει όλα τα αντικείμενά του σε όλους τους κόμβους ραντεβού), αντιπροσωπεύοντας τα δύο άκρα σενάρια κατανομής φόρτου για την προσέγγιση ραντεβού. Το σχήμα 5.23 δείχνει τους συντελεστές GC και FI για έναν αυξανόμενο αριθμό κόμβων ραντεβού (replicas). Όπως μπορούμε να δούμε, ακόμα και για 2,000 κόμβους ραντεβού, η τιμή του συντελεστή GC δεν πέφτει κάτω από το 0.725 ενώ η τιμή του FI αυξάνεται ως το πολύ 0.3.

Αυτό ίσως να φαίνεται παράλογο αρχικά για τον μη εξοικειωμένο αναγνώστη· θα περίμενε κανείς ότι ο φόρτος θα ήταν πλήρως μοιρασμένος όταν τα αντίγραφα θα ήταν τόσα όσοι και οι κόμβοι του δικτύου (δηλ. 1000). Ωστόσο, η μέθοδος επιλογής των κόμβων είναι που κάνει τη διαφορά· καθώς κάθε κόμβος δεν (μπορεί να) γνωρίζει εξ αρχής τα αναγνωριστικά όλων των κόμβων του συστήματος (ακόμα κι αν υπήρχε η δυνατότητα αυτή, η πληροφορία αυτή θα απαιτούσε συχνές ανανεώσεις καθώς κόμβοι θα εισέρχονταν και θα αποχωρούσαν), η καλύτερη εναλλακτική είναι να

επιλέγονται τυχαία αναγνωριστικά (με ομοιόμορφη κατανομή) από τον χώρο αναγνωριστικών των κόμβων. Αν και τα αναγνωριστικά των πραγματικών κόμβων επιλέγονται με σχεδόν ομοιόμορφη κατανομή (εξαιτίας της χρήσης κρυπτογραφικών συναρτήσεων κατακερματισμού για τον υπολογισμό τους), αυτό δε συνεπάγεται ότι οι αποστάσεις ανάμεσα σε διαδοχικούς κόμβους είναι ίσες· για την ακρίβεια, συμβαίνει το ακριβώς αντίθετο. Για παράδειγμα, σε ένα σύστημα με 1000 κόμβους και χώρο αναγνωριστικών 32 bits, η ελάχιστη, μέση και μέγιστη απόσταση ανάμεσα σε δύο οποιουδήποτε διαδοχικούς κόμβους ήταν $\approx 2^{11}$, 2^{22} και 2^{25} αντίστοιχα, με τυπική απόκλιση $\approx 2^{22}$. Σημειώνουμε ότι αυτή η κατανομή των αποστάσεων έχει τιμή συντελεστών GC και $FI \approx 0.5$. Επομένως, το να επιλέγουμε ανάμεσα σε αυτούς τους κόμβους υπολογίζοντας αρχικά ένα τυχαίο (ομοιόμορφα) αναγνωριστικό και στη συνέχεια εντοπίζοντας τον κόμβο που είναι υπεύθυνος γι' αυτό, οδηγεί σε μία μη ομοιόμορφη κατανομή του φόρτου.

Ας υποθέσουμε τώρα ότι υπήρχε ένας τρόπος κάθε κόμβος να γνωρίζει τα αναγνωριστικά όλων των υπόλοιπων κόμβων του δικτύου σε κάθε χρονική στιγμή. Ακόμα και σε αυτή την περίπτωση, θα απαιτούνται τουλάχιστον ≈ 275 κόμβοι ραντεβού (είτε με απλή είτε με προορατική αντιγραφή), για να επιτύχει η προσέγγιση ραντεβού τις τιμές συντελεστών GC και FI της προσέγγισης DHS. Ωστόσο, στην περίπτωση αυτή το κόστος μηνυμάτων για την εισαγωγή πλειάδων ή/και τον υπολογισμό του κατανεμημένου ερωτήματος θα ήταν απαγορευτικό. Ως σημείο σύγκρισης, το σχήμα 5.24 δείχνει το κόστος μηνυμάτων για την εισαγωγή πλειάδων με προορατική αντιγραφή και το κόστος υπολογισμού ερωτημάτων Count με απλή αντιγραφή, χρησιμοποιώντας κατανεμημένες συνόψεις βασισμένες σε προσέγγιση ραντεβού, για ένα σαφώς μικρότερο σενάριο με 50 κόμβους ραντεβού/αντίγραφα.

Ως τώρα φαίνεται να υπάρχει μία σαφής σχεδιαστική διελκυστίδα ανάμεσα στις μεθόδους που βασίζονται σε ραντεβού και στο DHS· οι πρώτες είναι καλύτερες όταν αυτό που μας ενδιαφέρει είναι το απόλυτο πλήθος μηνυμάτων, ενώ οι δεύτερες είναι η προφανής επιλογή όταν μας ενδιαφέρει η δίκαιη κατανομή του φόρτου. Αυτό είναι όμως μόνο η μισή αλήθεια· ας παρατηρήσουμε τα σχήματα 5.23 και 5.24 από μία διαφορετική οπτική γωνία: αυτή των διαφορετικών πολλαπλών κατανεμημένων ερωτημάτων αντί για αυτή των αντιγράφων της ίδιας κατανεμημένης σύνοψης. Ας υποθέσουμε, για παράδειγμα, ότι έπρεπε να υπολογίσουμε πολλαπλά ερωτήματα Count, Sum, Average κτλ. πάνω από το ίδιο ΔΟΕ. Εκτός και αποθηκεύσουμε όλες τις συνόψεις σε ένα και μόνο κόμβο – ένα εφιαλτικό σενάριο από την άποψη της κατανομής φόρτου και επομένως και της κλιμακωσιμότητας και της αντοχής του συστήματος σε λάθη – η μόνη άλλη επιλογή είναι να αναθέσουμε κάθε τέτοια σύνοψη σε διαφορετικό, τυχαία επιλεγμένο κόμβο ραντεβού. Το σενάριο αυτό, όμως, είναι πανομοιότυπο με το να έχουμε μία σύνοψη με προορατική αντιγραφή και τόσα αντίγραφα όσα και τα ερωτήματα που θέλουμε να υπολογίσουμε. Το τελικό συμπέρασμα είναι ότι αν και στην περίπτωση μεμονωμένων ερωτημάτων η προσέγγιση ραντεβού φαίνεται να είναι ο νικητής στον τομέα του απόλυτου κόστους μηνυμάτων, τα πλεονεκτήματά της εξαφανίζονται όταν απαιτείται ταυτόχρονος υπολογισμός πολλαπλών ερωτημάτων.

Αυτό μπορούμε να το δούμε καθαρότερα στο σχήμα 5.25 το οποίο δείχνει τον συνολικό φόρτο στους κόμβους του δικτύου κατά τον υπολογισμό ερωτημάτων για 10, 20, 50 και 100 ερωτήματα Count (ή ισοδύναμα για ένα ερώτημα Count και προορατική αντιγραφή με παράγοντα αντιγραφής 1%, 2%, 5% και 10% αντίστοιχα – η κατάσταση για τις περιπτώσεις απλής αντιγραφής θα ήταν ακόμα χειρότερη). Παρατηρήστε ότι ο κατακόρυφος άξονας είναι σε λογαριθμική κλίμακα. Το σχήμα αυτό αποδεικνύει με τον καλύτερο δυνατό τρόπο την γραμμική αύξηση στον συνολικό φόρτο υπολογισμού ερωτημάτων σε σχέση με τον αριθμό των υπολογιζόμενων ερωτημάτων για την προσέγγιση ραντεβού. Σημειώνουμε ότι δείχνουμε μόνο μία καμπύλη για την προσέγγιση DHS για κάθε περίπτωση πλήθους bitmaps καθώς ο συνολικός φόρτος υπολογισμού και το συνολικό κόστος μηνυμάτων είναι τα ίδια ανεξάρτητα από τον αριθμό των ερωτημάτων που υπολογίζονται, εξαιτίας των ιδιοτήτων του DHS που αναφέραμε στην ενότητα 4.

Σχήμα 5.25: Συνολικός Φόρτος Υπολογισμού Ερωτημάτων Όταν Πολλαπλά Ερωτήματα Επεξεργάζονται Ταυτόχρονα από το Σύστημα

Κλιμακωσιμότητα Δεδομένων των παραπάνω, εστιάζουμε τώρα στην μελέτη της κλιμακωσιμότητας της προσέγγισης DHS. Όπως έγινε προφανές ως τώρα, οι λύσεις που βασίζονται στο DHS έχουν σαφές προβάδισμα στον τομέα της κλιμακωσιμότητας σε μεγάλους αριθμούς ταυτόχρονα υπολογιζόμενων ερωτημάτων και στην κατανομή του σχετικού φόρτου στους κόμβους του δικτύου. Οι λύσεις αυτές επιπλέον παρουσιάζουν εξαιρετική κλιμακωσιμότητα όσον αφορά τον αριθμό των κόμβων και των πλειάδων στο σύστημα. Στην ενότητα αυτή επαναλαμβάνουμε τις μετρήσεις μας για τα ερωτήματα Count μεταβάλλοντας τον αριθμό των κόμβων στο σύστημα από 1000 σε 2500, 5000 και 10000 κόμβους. Το πλήθος των πλειάδων μεταβάλλεται επίσης από $30 \times$ σε $300 \times$ το πλήθος των κόμβων (δηλ. 30,000 και 300,000 πλειάδες για 1000 κόμβους, 75,000 και 750,000 πλειάδες για 2500 κόμβους κοκ.)

Τα σχήματα 5.26(α'), 5.27(α') και 5.28(α') δείχνουν τον μέσο φόρτο εισαγωγής ανά κόμβο, τον μέγιστο φόρτο εισαγωγής και το κόστος μηνυμάτων αντίστοιχα για την περίπτωση $30 \times$ πλειάδων – δηλαδή, το μέσο αριθμό επισκέψεων σε κάθε κόμβο κατά την εισαγωγή όλων των πλειάδων στην κατανομημένη σύνοψη, τον μέγιστο αντίστοιχο αριθμό επισκέψεων και τον μέσο αριθμό μηνυμάτων/αλμάτων που πρέπει να στείλει κάθε κόμβος για να εισάγει όλες τις πλειάδες του στο DHS – για διάφορους συνδυασμούς κατανομής τιμών εισόδου και αριθμού bitmaps στο DHS. Παρομοίως, τα σχήματα 5.26(β'), 5.27(β') και 5.28(β') δείχνουν τις ίδιες μετρικές για την περίπτωση $300 \times$ πλειάδων, όπως μπορούμε να δούμε, ο φόρτος εισαγωγής ανά κόμβο είναι σχεδόν σταθερός, ενώ ο μέγιστο φόρτος αυξάνεται λογαριθμικά σε σχέση με τον αριθμό των κόμβων και υπογραμμικά σε σχέση με τον αριθμό των αντικειμένων/πλειάδων (οι αντίστοιχες γραφικές παραστάσεις για την περίπτωση της προσέγγισης ραντεβού θα έδειχναν μία γραμμική εξάρτηση από τον αριθμό των κόμβων και ανεξαρτησία από τον αριθμό των αντικειμένων). Επιπλέον, το μέσο πλήθος μηνυμάτων ανά κόμβο αυξάνεται σχεδόν λογαριθμικά ως προς το πλήθος των κόμβων και των πλειάδων στο σύστημα.

Παρομοίως, τα σχήματα 5.29(α') και 5.30(α') δείχνουν τον μέσο αριθμό κόμβων και το μέσο κόστος αλμάτων για τον υπολογισμό ερωτημάτων Count για την περίπτωση $30 \times$ πλειάδων, ενώ η περίπτωση $300 \times$ πλειάδων φαίνεται στα σχήματα 5.29(β') και 5.30(β'). Από τα σχήματα αυτά μπορούμε να συμπεράνουμε ότι το μέσο κόστος μηνυμάτων για τον υπολογισμό του ερωτήματος είναι ανεξάρτητο του αριθμού των αντικειμένων ενώ αυξάνεται λογαριθμικά στο πλήθος των κόμβων του συστήματος. Ο μέσος φόρτος επεξεργασίας ερωτημάτων επίσης αυξάνεται λογαριθμικά στο πλήθος των κόμβων. Το μέσο λάθος εκτίμησης και οι τιμές των συντελεστών Gini και Fairness Index ήταν όμοιοι με αυτούς των σχημάτων 5.16(β'), 5.20(α') και 5.20(β') αντίστοιχα. Τέλος, τα σχήματα για τα υπόλοιπα ερωτήματα ήταν παρόμοια με αυτά που παρουσιάσαμε για τα ερωτήματα Count και ως εκ τούτου παραλείπονται.

(α) Πλήθος πλειάδων = $30 \times$ πλήθος κόμβων

(β) Πλήθος πλειάδων = $300 \times$ πλήθος κόμβων

Σχήμα 5.26: Μέσος Φόρτος Εισαγωγής ανά Κόμβο για Ερωτήματα Count Χρησιμοποιώντας το DHS, για Διάφορες Τιμές του Πλήθους των Κόμβων στο Δίκτυο: ένα δεκαπλασιασμός του αριθμού των αντικειμένων οδηγεί σε μία μόλις κατά $\approx 60\%$ αύξηση στο πλήθος των επισκέψεων σε κάθε κόμβο κατά την φάση εισαγωγής

(α) Πλήθος πλειάδων = $30 \times$ πλήθος κόμβων

(β) Πλήθος πλειάδων = $300 \times$ πλήθος κόμβων

Σχήμα 5.27: Μέγιστος Φόρτος Εισαγωγής για Ερωτήματα Count Χρησιμοποιώντας το DHS, για Διάφορες Τιμές του Πλήθους των Κόμβων στο Δίκτυο: ένα δεκαπλασιασμός του αριθμού των αντικειμένων οδηγεί σε μία κατά $\approx 6 \times$ αύξηση στο μέγιστο πλήθος επισκέψεων σε οποιονδήποτε κόμβο κατά την φάση εισαγωγής, ενώ παραμένει ακόμα μία τάξη μεγέθους μικρότερο από το αντίστοιχο πλήθος επισκέψεων της προσέγγισης ραντεβού

(α) Πλήθος πλειάδων = $30 \times$ πλήθος κόμβων

(β) Πλήθος πλειάδων = $300 \times$ πλήθος κόμβων

Σχήμα 5.28: Μέσος Αριθμός Μηνυμάτων/Αλμάτων ανά Κόμβο για Εισαγωγή Πλειάδων για Ερωτήματα Count Χρησιμοποιώντας το DHS, για Διάφορες Τιμές του Πλήθους των Κόμβων στο Δίκτυο· ένα δεκαπλασιασμός του αριθμού των αντικειμένων οδηγεί σε μία μόλις κατά $\approx 35\%$ αύξηση στο πλήθος των μηνυμάτων κάθε κόμβου για την εισαγωγή όλων των πλειάδων του στο DHS

(α) Πλήθος πλειάδων = $30 \times$ πλήθος κόμβων

(β) Πλήθος πλειάδων = $300 \times$ πλήθος κόμβων

Σχήμα 5.29: Μέσος Αριθμός Κόμβων που Επισκέφθηκε ο Αλγόριθμος Υπολογισμού των Ερωτημάτων Count Χρησιμοποιώντας το DHS, για Διάφορες Τιμές του Πλήθους των Κόμβων στο Δίκτυο

(α) Πλήθος πλειάδων = $30 \times$ πλήθος κόμβων

(β) Πλήθος πλειάδων = $300 \times$ πλήθος κόμβων

Σχήμα 5.30: Μέσο Κόστος Αλμάτων ανά Ερώτημα για Ερωτήματα Count Χρησιμοποιώντας το DHS, για Διάφορες Τιμές του Πλήθους των Κόμβων στο Δίκτυο

(α) Μέσο σφάλμα συχνότητων κελιών

(β) Μέσο σφάλμα θέσης ορίων κελιών (μόνο για ιστογράμματα Equi-Depth)

Σχήμα 5.31: Αποτελέσματα Υπολογισμού Ιστογραμμάτων: 1,000 κόμβοι, 10,000,000 αντικείμενα, τιμές στο διάστημα $[0, 1,000,000)$, 100 κελιά στο αρχικό ιστόγραμμα DHS, 10 κελιά ανά υπολογιζόμενο ιστόγραμμα

Ιστογράμματα Θα θέλαμε να τονίσουμε εκ νέου ότι ο κεντρικός στόχος μας είναι η διαφανής κατανομημένη λειτουργία. Ως εκ τούτου δεν μας απασχολούν επί τούτου οι στατιστικές ιδιότητες των ιστογραμμάτων ή το αν ένας συγκεκριμένος τύπος ιστογραμμάτων είναι ο βέλτιστος ή όχι· τα θέματα αυτά έχουν μελετηθεί εκτενώς στη βιβλιογραφία. Αντ' αυτού, συγκρίνουμε τα αποτελέσματά μας με ιστογράμματα των ίδιων τύπων και χαρακτηριστικών τα οποία λειτουργούν σε κεντρικοποιημένο περιβάλλον.

Τα πειράματά μας προχωρούν ως εξής: (α) πρώτα δημιουργούμε τα δεδομένα εισόδου και υπολογίζουμε τα ιστογράμμά τους με κεντρικοποιημένο τρόπο· (β) κατόπιν, ρυθμίζουμε και δημιουργούμε ένα δίκτυο 1000 κόμβων στο οποίο εισαγάγουμε πλειάδες δεδομένων όπως και νωρίτερα· (γ) τέλος, επιλέγουμε τυχαίους κόμβους οι οποίοι ανασυνθέτουν ή/και υπολογίζουν τα ίδια ιστογράμματα, ακολουθώντας τις μεθόδους της ενότητας 4.10. Τονίζουμε ότι στο στάδιο αυτό ασχολούμαστε μόνο με μονοδιάστατα ιστογράμματα.

Αρχικά συμπληρώνουμε και ανασυνθέτουμε ένα Equi-Width ιστόγραμμα 100 κελιών χρησιμοποιώντας το DHS και κατόπιν το χρησιμοποιούμε για να υπολογίσουμε ιστογράμματα Equi-Depth, Compressed(V,F) και MaxDiff(V,F) 10 κελιών. Στο τμήμα αυτό της πειραματικής μας μελέτης χρησιμοποιούμε μόνο 256 bitmaps ανά κατανομημένη σύνοψη κατακερματισμού, καθώς το πλήθος αυτό προσφέρει ένα καλό σημείο ισορροπίας ανάμεσα στο κόστος και την ακρίβεια, όπως έδειξαν τα προηγούμενα αποτελέσματα. Όλοι οι πίνακες/σχέσεις έχουν μία αριθμητική στήλη η οποία παίρνει ακέραιες τιμές με βάση είτε μία ομοιόμορφη κατανομή είτε μία κατανομή Zipf με παράμετρο θ ίση με 0.7, 1.0 και 1.2 ([51, 111]).

Για κάθε υπολογιζόμενο ιστόγραμμα μετράμε το μέσο λάθος συχνότητας ανά κελί σε σχέση με την πραγματική κατανομή συχνότητας. Για την περίπτωση των ιστογραμμάτων Equi-Depth δείχνουμε επιπλέον το επανομαζόμενο «λάθος θέσης»: το λάθος στην εκτίμηση των ορίων των κελιών. Για τα τελευταία δύο είδη ιστογραμμάτων δείχνουμε ακόμα το recall rate – το ποσοστό των κορυφαίων συχνότητων ή των μοναδιαίων κελιών αντίστοιχα που υπολογίσαμε σωστά από το σύνολο των 10 κελιών. Τέλος, για τα Compressed ιστογράμματα δείχνουμε επιπλέον το «λάθος ταξινόμησης», δηλαδή το ποσοστό των τιμών των κορυφαίων συχνότητων τις οποίες ο αλγόριθμος υπολόγισε με τη σωστή σειρά. Όπως και πριν, οι τιμές που αναφέρουμε υπολογίζονται ως οι μέσες τιμές πολλαπλών εκτελέσεων των πειραμάτων μας.

Αρχικά χρησιμοποιήσαμε έναν πίνακα 10,000,000 πλειάδων, με τιμές στο διάστημα $[0, 1,000,000)$. Το σχήμα 5.31(α) συνοψίζει τα αποτελέσματα από αυτό το σετ πειραμάτων. Όσον αφορά το κόστος αριθμού αλμάτων και την κατανομή φόρτου, ισχύουν όσα είδαμε στις περιπτώσεις

των πολλαπλών ταυτόχρονων ερωτημάτων, καθώς ο προτεινόμενος τρόπος υλοποίησης των ιστογραμμάτων Equi-Width είναι χρησιμοποιώντας μία διαφορετική «μετρική» DHS ή ένα διαφορετικό αναγνωριστικό ραντεβού για κάθε κελί του ιστογράμματος. Αυτό σημαίνει ότι μία προσέγγιση ραντεβού θα έπρεπε είτε (α) να χρησιμοποιήσει πολλαπλούς κόμβους ραντεβού, υποφέροντας έτσι από υπερβολικό κόστος αριθμού αλμάτων/μηνυμάτων και συνολικού φόρτου ερωτημάτων όπως στα σχήματα 5.24 και 5.25 ή (β) να αποθηκεύσει όλα τα κελιά των ιστογραμμάτων σε ένα μόνο κόμβο, δημιουργώντας έτσι σοβαρή ανισορροπία στην κατανομή του φόρτου στο σύστημα. Επομένως παραλείπουμε εντελώς την περίπτωση ιστογραμμάτων βασισμένων στην προσέγγιση ραντεβού και εστιάζουμε σε ιστογράμματα βασισμένα στο DHS.

Όσον αφορά την ακρίβεια, το μέσο λάθος συχνότητας κελιών είναι σχεδόν αμελητέο για τα ιστογράμματα Equi-Width, Average-Shifted και Equi-Depth. Το πολύ μικρό λάθος στην περίπτωση των ιστογραμμάτων Equi-Depth είναι κάπως παραπλανητικό· καθώς όλα τα κελιά έχουν σχεδόν την ίδια συχνότητα υπάρχει πολύ μικρό περιθώριο για λάθος. Αυτό που έχει μεγαλύτερη σημασία στην περίπτωση αυτή είναι το λάθος θέσης των ορίων των κελιών (σχ. 5.31(β)). Και σε αυτή την περίπτωση, οι αλγόριθμοί μας είχαν σχεδόν αμελητέο σφάλμα, ακόμα και στην περίπτωση της εισόδου με κατανομή Zipf με παράμετρο θ ίση με 1.2. Στο πεδίο του κόστους, ο μέσος αριθμός αλμάτων/μηνυμάτων για την ανασύνθεση του αρχικού ιστογράμματος με το DHS ήταν ≈ 26 άλματα (δηλ. περίπου 6 αναζητήσεις DHT), ενώ η μέση κατανάλωση εύρους ζώνης ήταν ≈ 60 -70 kbytes.

Στο σχήμα 5.31(α') δε συμπεριλάβαμε καθόλου αποτελέσματα για ιστογράμματα Max-Diff(V,F) και Compressed(V,F) καθώς αποδείχθηκαν προβληματικά. Ο κύριος λόγος γι' αυτό ήταν ότι, με 100 κελιά στο αρχικό ιστόγραμμα και 1 εκατομμύριο τιμές στο πεδίο τιμών των στηλών, 10,000 τιμές αντιστοιχούν σε κάθε κελί του ιστογράμματος. Αυτό καθιστά σχεδόν αδύνατη την εκτίμηση, με βάση μόνο τον αριθμό των μοναδικών τιμών κάθε κελιού και την συνολική συχνότητα του κελιού, της αντιστοιχίας συχνοτήτων σε τιμές, η οποία απαιτείται για τα ιστογράμματα αυτών των τύπων.

Στο σημείο αυτό αλλάξαμε τις ρυθμίσεις των πειραμάτων μας προς μια διαφορετική κατεύθυνση: χρησιμοποιήσαμε ξανά δίκτυα 1000 κόμβων και δημιουργήσαμε τα ίδια ιστογράμματα όπως και πριν χρησιμοποιώντας πίνακες 10,000,000 πλειάδων, οι οποίες όμως τώρα έπαιρναν τιμές στο διάστημα $[0, 1000)$ ενώ το αρχικό ιστόγραμμα είχε και αυτό 1000 κελιά. Αυτό αντιστοιχεί στο να έχουμε έναν χωριστό μετρητή DHS για κάθε δυνατή τιμή στο πεδίο τιμών των στηλών μας.

Τα αποτελέσματα φαίνονται στα σχήματα 5.32(α')-5.32(δ). Όπως μπορούμε να δούμε, το μέσο λάθος συχνοτήτων κελιών είναι στα ίδια επίπεδα με πριν (σχ. 5.32(α')) και το ίδιο ισχύει για το λάθος θέσης ορίων κελιών για τα ιστογράμματα Equi-Depth (σχ. 5.32(β')). Όσον αφορά το μέσο λάθος ταξινόμησης και το μέσο recall rate (σχ. 5.32(γ') και 5.32(δ') αντίστοιχα), παρατηρούμε μία σταθερή τάση: και στις δύο περιπτώσεις οι αλγόριθμοί μας χάνουν σε ακρίβεια όταν οι τιμές προέρχονται από ομοιόμορφη κατανομή και βελτιώνονται όσο μεγαλώνει η «κλίση» της κατανομής. Πρέπει όμως να τονίσουμε ότι τα ιστογράμματα αυτά χρησιμοποιούνται κυρίως με δεδομένα κατανομής τιμών με μεγάλη κλίση· αν οι τιμές των πλειάδων είναι κατανεμημένες ομοιόμορφα στο πεδίο τιμών τους, δεν κερδίζουμε τίποτα χρησιμοποιώντας τέτοιου είδους ιστογράμματα (ή εν τέλει ιστογράμματα γενικά)!

Ωστόσο, η πλέον ενδιαφέρουσα παρατήρηση σε αυτό το σύνολο πειραμάτων αφορά το κόστος ανασύνθεσης του αρχικού ιστογράμματος: σε όλα τα πειράματα το κόστος αυτό δεν ξεπέρασε τα 30 άλματα (περίπου 27 άλματα κατά μέσο όρο). Η μέση κατανάλωση εύρους ζώνης δικτύου για την ανασύνθεση του αρχικού ιστογράμματος ήταν περίπου 600-700 kbytes. Όπως αναφέραμε και παραπάνω, η βασική σχεδιαστική επιλογή εδώ σχετίζεται με τις απαιτήσεις σε εύρος ζώνης δικτύου· με 1000 κελιά σε σχέση με τα 100 κελιά που είχαμε πριν, κάθε κόμβος που επισκεπτόμαστε κατά την ανασύνθεση του ιστογράμματος μπορεί να απαντήσει με έως και 10 φορές περισσότερα δεδομένα. Υπενθυμίζουμε ότι τα δεδομένα αυτά είναι απλές πλειάδες του τύπου <αναγνωριστικό μετρικής, διάνυσμα bits> των οποίων το μέγεθος είναι πολύ μικρό (24-84 bytes ανά πλειάδα στην

(α') Μέσο λάθος συχνότητας κελιών

(β') Μέσο λάθος θέσης ορίων κελιών - Ιστογράμματα Equi-Depth

(γ') Μέσο λάθος ταξινόμησης - Ιστογράμματα Compressed(V,F)

(δ') Μέσο recall rate - Ιστογράμματα Compressed(V,F) και MaxDiff(V,F)

Σχήμα 5.32: Αποτελέσματα Υπολογισμού Ιστογραμμάτων: 1,000 κόμβοι, 10,000,000 πλειάδες, τιμές στο διάστημα $[0, 1,000)$, 1,000 κελιά στο αρχικό ιστόγραμμα DHS, 10 κελιά ανά υπολογιζόμενο ιστόγραμμα

υλοποίησή μας).

Η διελκυστίνδα αυτή γίνεται σαφώς πιο εμφανής όταν αντιμετωπίζουμε μεγάλα πεδία τιμών πλειάδων. Για παράδειγμα, αν οι τιμές των πλειάδων προέρχονται από το διάστημα $[0, 1,000,000)$ και σχεδόν όλες οι τιμές του διαστήματος αυτού αντιπροσωπεύονται στις πλειάδες και απαιτούσαμε ένα αρχικό ιστόγραμμα ενός εκατομμυρίου κελιών, τότε το κόστος σε εύρος ζώνης δικτύου για την ανασύνθεσή του θα ήταν σχεδόν 600-700 Mbytes! Μία πρώτη παρατήρηση στο σημείο αυτό είναι ότι ο αριθμός αυτός αναφέρεται σε ασυμπίεστα δεδομένα DHS· με χρήση κατάλληλων αλγορίθμων συμπίεσης θα μπορούσαμε να μειώσουμε δραστικά την κατανάλωση εύρους ζώνης κατά μία με δύο τάξεις μεγέθους. Εκτός από αυτό, (α) εξ όσων γνωρίζουμε, τα περισσότερα πραγματικά συστήματα βάσεων δεδομένων χτίζουν δομές δεικτοδότησης και ιστογράμματα για στήλες με μερικές (δεκάδες) χιλιάδες διαφορετικών τιμών, (β) στα περισσότερα ερωτήματα δε θα χρειαστούν πληροφορίες από όλα τα κελιά του ιστογράμματος παρά μόνο από μερικά από αυτά και (γ) ακόμα κι αν τα παραπάνω δεν ισχύουν για κάποια περίπτωση, μπορούμε ακόμα να υπολογίσουμε ακριβή ιστογράμματα Equi-Width, ASH και Equi-Depth με μικρό κόστος αριθμού μηνυμάτων/αλμάτων και χαμηλή κατανάλωση εύρους ζώνης δικτύου.

Συζήτηση Θα θέλαμε να καταδείξουμε την ανάγκη και την καταλληλότητα των τεχνικών που παρουσιάσαμε, υλοποιήσαμε και εξετάσαμε στην εργασία αυτή, για την επίλυση βασικών προβλημάτων στο πεδίο των βάσεων δεδομένων πάνω από δίκτυα ομοτίμων εταιρών. Τα παραπάνω πειραματικά αποτελέσματα έδειξαν ότι ο αριθμός αλμάτων/μηνυμάτων και το κόστος εύρους ζώνης δικτύου για την ανασύνθεση του βασικού ιστογράμματος DEWH ή DASH και τον υπολογισμό των πιο περίπλοκων ιστογραμμάτων μπορεί να είναι αρκετά χαμηλά. Σημειώνουμε λοιπόν ότι: (α) αφού του ένας κόμβος έχει πληρώσει το κόστος της ανασύνθεσης/υπολογισμού ενός ιστογράμματος, η επιλογή του βέλτιστου πλάνου εκτέλεσης ενός ερωτήματος είναι τοπική λειτουργία και (β) τα παραπάνω νούμερα αναφέρονται στο κόστος ανασύνθεσης ολόκληρου του ιστογράμματος· η επεξεργασία των ερωτημάτων μπορεί να απαιτεί τον υπολογισμό της συχνότητας συγκεκριμένων μόνο κελιών, ανάλογα με τους περιορισμούς του ερωτήματος.

Τώρα ας υποθέσουμε ότι ένας βελτιστοποιητής ερωτημάτων, έχοντας τα ιστογράμματα αυτά, είναι (τουλάχιστον) σε θέση να επιλέξει το βέλτιστο πλάνο εκτέλεσης ενός ερωτήματος. Στην περίπτωση αυτή το κέρδος σε εύρος ζώνης δικτύου και σε χρόνο απόκρισης θα είναι σίγουρα σημαντικό [13,60]. Για παράδειγμα, στο [60] οι συγγραφείς εξετάζουν multi-way joins σε ένα σύστημα σαφώς μικρότερης κλίμακας από το δικό μας (256 κόμβοι και σχέσεις 256,000 πλειάδων ή 100 πλειάδες ανά κόμβο). Η βέλτιστη στρατηγική εκτέλεσης των ερωτημάτων στην περίπτωση three-way join ερωτημάτων θα έχει ως αποτέλεσμα την μεταφορά 47Mbytes δεδομένων, σε αντίθεση με τα 71Mbytes που μεταφέρονται από τον μηχανισμό των FREddies και τα δύο εκ των οποίων είναι τάξεις μεγέθους μεγαλύτερα από τα περίπου 0.7Mbytes που απαιτούνται για την ανασύνθεση ενός σχετικού ιστογράμματος από το DHS. Έτσι, για παράδειγμα, αν το σύστημα PIER συμπεριλάμβανε ιστογράμματα DHS στην μηχανή επεξεργασίας ερωτημάτων του, θα είχε τη δυνατότητα επιλογής του βέλτιστου πλάνου εκτέλεσης ερωτημάτων με ένα σαφώς λογικό επιπλέον κόστος, με αποτέλεσμα σημαντικά κέρδη σε κατανάλωση εύρους ζώνης δικτύου και χρόνου εκτέλεσης. Το γεγονός αυτό δίνει μία ένδειξη του αντίκτυπου της ενσωμάτωσης του DHS σε μηχανές επεξεργασίας ερωτημάτων διαδικτυακής κλίμακας.

Κεφάλαιο 6

Συμπεράσματα & Μελλοντικές Επεκτάσεις

Στην εργασία αυτή προσπαθήσαμε να αντιμετωπίσουμε θεμελιώδη ζητήματα σχεδίασης και υλοποίησης αλγορίθμων και πρωτοκόλλων για κατανεμημένα συστήματα καθώς και αρχιτεκτονικές τέτοιων συστημάτων, τα οποία παρέχουν την απαραίτητη υποδομή για τον υπολογισμό συγκεκριμένων καθολικών ιδιοτήτων που περιγράφουν την κατάσταση ενός δικτύου ομοτίμων εταίρων. Με αυτή την λογική κινηθήκαμε σε δύο κατευθύνσεις: (α) στην ποσοτικοποίηση και την εκμετάλλευση της ανομοιογένειας κόμβων ενός δικτύου ομοτίμων και (β) σε υπολογιστικές εκτιμήσεις για καθολικές ιδιότητες του δικτύου ομοτίμων, με στόχο την διευκόλυνση της επεξεργασίας και της βελτιστοποίησης πολύπλοκων επερωτημάτων σε συστήματα διαχείρισης δεδομένων διαδικτυακής κλίμακας.

Στα δίκτυα επικάλυψης κοινωνικοοικονομικοί παράγοντες (όπως η διαθεσιμότητα πόρων, η επίπεδη τιμολόγηση, η ηθική τοποθέτηση κ.α.) επηρεάζουν την συμπεριφορά ενός κόμβου (και άρα την τοπολογία του δικτύου) και πρέπει να ληφθούν υπόψιν. Η εγωιστική/αλτρουιστική συμπεριφορά των κόμβων είναι βασισμένη σε αυτούς τους κοινωνικοοικονομικούς παράγοντες και έχουν αναφερθεί πολλές φορές στην βιβλιογραφία των δικτύων ομοτίμων. Στο πρώτο κομμάτι αυτής της διδακτορικής διατριβής προτείνουμε την αρχιτεκτονική AESOP: ένα νέο παράδειγμα για την δόμηση δικτύων βασισμένων σε κατανεμημένους πίνακες κατακερματισμού που εμπεριέχει τέτοια συμπεριφορά στην αρχιτεκτονική του σχεδίαση. Το σύστημα AESOP αποτελείται από μία πολυεπίπεδη αρχιτεκτονική, η οποία αποτελείται από ένα επίπεδο που αξιοποιεί τους αλτρουιστές, ένα επίπεδο το οποίο καταγράφει/μετρά και ελέγχει με μικρό κόστος την συμπεριφορά συσχετιζόμενων κόμβων και τέλος από ένα επίπεδο το οποίο διαχειρίζεται την συνεργατικότητα τους. Περιγράφουμε την ανάγκη μίας τέτοια δόμησης και δείχνουμε πώς να επιτύχουμε σημαντικά βελτιωμένη από πλευράς απόδοσης δρομολόγηση σε παρόμοια δίκτυα τόσο για την περίπτωση που η τοπολογία του δικτύου παραμένει σταθερή αλλά και για την περίπτωση της δυναμικής αλλαγής της τοπολογίας – χωρίς να μεταφέρουμε επιπλέον κόστος σε άλλες λειτουργίες του συστήματος – και παράλληλα εισάγοντας σημαντικά κέρδη στην απόδοση σε μετρικές απαιτούμενων βημάτων, στο μέγεθος των πινάκων δρομολόγησης και τις απαιτήσεις για συντήρηση και σταθερότητα.

Παρουσιάσαμε διάφορες αρχιτεκτονικές και αλγορίθμους, που προσφέρουν αντιστάθμιση ανάμεσα στην αύξηση της ταχύτητας δρομολόγησης και του πλήθους των αλτρουιστικών κόμβων και του μεγέθους των πινάκων δρομολόγησης τους, καθώς και μεταξύ του μήκους του μονοπατιού δρομολόγησης στην σταθερή κατάσταση και της διασύνδεσης του δικτύου των αλτρουιστών κόμβων. Το τελικό συμπέρασμα είναι ότι απαιτείται ένα πολύ μικρό ποσοστό αλτρουιστών κόμβων ενώ επιτυγχάνεται μία βελτίωση στην περίπτωση της δρομολόγησης σε σταθερή κατάσταση κατά

έναν παράγοντα $2\text{-}4\times$ και κατά μερικές τάξεις μεγέθους στην δυναμική περίπτωση. Ταυτόχρονα το μέγεθος των πινάκων δρομολόγησης ελαττώνεται κατά ένα παράγοντα $2\times$, το οποίο οδηγεί το δίκτυο σε καλύτερη σταθερότητα. Ταυτόχρονα, αυξάνεται και η σταθερότητα δρομολόγησης λόγω της μικρότερης κατάστασης δρομολόγησης και της απομόνωσης των παθογενών εγωιστικών κόμβων μέσα σε μικρές συστάδες από κόμβους. Λόγω της παραπάνω απλότητας, πιστεύουμε ότι το προτεινόμενο παράδειγμα είναι βιώσιμο και πραγματοποιήσιμο και το προτείνουμε σαν την νέα μέθοδο για την δημιουργία των δικτύων ομοτίμων στο μέλλον. Τέλος παρουσιάσαμε κάποια ανοιχτά/άλυτα προβλήματα, των οποίων η επίλυση θα εξασφαλίσει σημαντικά κέρδη στην απόδοση. Παράλληλα, αναπτύσσουμε ένα λογισμικό επίπεδο καταγραφής/παρακολούθησης, το οποίο μπορεί με συνέπεια και καλή απόδοση να ανακαλύψει τους αλτρουιστές κόμβους του δικτύου [96]. Επιπλέον αναπτύσσουμε μία σουίτα από πρωτόκολλα για την ενθυσία της υποδομής μας με την αρχιτεκτονική PLANES. Τα δύο αυτά αρθρώματα οδηγούν στην δημιουργία αυτο-οργανούμενων, αλτρουιστικών εμπνευσμένων, δικτύων ομοτίμων εταίρων που βασίζονται στην αρχιτεκτονική PLANES διατηρώντας την υψηλά επιθυμητή ιδιότητα για απόδοση που προσφέρει η συγκεκριμένη αρχιτεκτονική, όπως παρουσιάζεται εδώ.

Επιπλέον αντιμετωπίζουμε το πρόβλημα της επεξεργασίας ερωτημάτων εύρους σε δομημένα δίκτυα ομοτίμων. Η προσέγγισή μας επεκτείνει την υπάρχουσα έρευνα σε δίκτυα ομοτίμων που βασίζονται σε κατανεμημένους πίνακες κατακερματισμού. Η προσέγγισή είναι βασισμένη σε μία νέα αρχιτεκτονική, η οποία διαχειρίζεται την αξιοποίηση ισχυρών κόμβων στο δίκτυο, που ονομάζονται RangeGuards και στους οποίους ανατίθεται επιπλέον λειτουργικότητα για την βελτίωση της ταχύτητας κατά την επεξεργασία των ερωτημάτων εύρους. Η αρχιτεκτονική βασίζεται πάνω σε (α) έναν μηχανισμό αποδοτικής αναγνώρισης και ταυτοποίησης ισχυρών κόμβων και (β) μηχανισμούς διαχείρισης τους κατά την διάρκεια της επεξεργασίας του ερωτήματος εύρους. Η πειραματική μελέτη μας, αποδεικνύουν ότι σημαντικές βελτιώσεις μπορούν να επιτευχθούν από την προτεινόμενη αρχιτεκτονική. Ίσως το πιο σημαντικό, είναι ότι αποφεύγονται οι κίνδυνοι και η αναποτελεσματικότητα του να βασίζομαστε σε αδύναμους κόμβους για επεξεργασία ερωτημάτων εύρους, αναφερόμενοι στην υπολογιστική, αποθηκευτική και επικοινωνιακή τους ισχύ.

Κατόπιν, παρουσιάζουμε τις Κατανεμημένες Συνόψεις Κατακερματισμού· έναν καινοτόμο, πλήρως αποκεντρωμένο, κλιμακώσιμο και αποδοτικό μηχανισμό, ικανό να παρέχει εκτιμήσεις πάνω στον πληθώρα των πολυσυνόλων που μπορεί να υπάρχουν σε πληροφοριακά συστήματα σε επίπεδο διαδικτύου. Το DHS, απ' όσο γνωρίζουμε, είναι το πρώτο σύστημα που ικανοποιεί του κεντρικούς στόχους για αποδοτικότητα, διαβάθμιση, ζύγιση φόρτου σε επίπεδο αποθήκευσης και πρόσβασης, υψηλή ακρίβεια και (μη-) ευαισθησία σε διπλότυπα, χωρίς τη διατήρηση επιπλέον δομών δεικτοδότησης. Τα χαρακτηριστικά αυτά το καθιστούν ικανό να γίνει η προτιμώμενη τεχνική καταμέτρησης για δίκτυα δεδομένων διαδικτυακής κλίμακας.

Δείξαμε πως μπορούν να χτιστούν DHS χρησιμοποιώντας τους αλγορίθμους εκτίμησης συνόψεων κατακερματισμού PCSA και super-LogLog. Υπολογίσαμε αναλυτικά τα επιπλέον σφάλματα εκτίμησης που εισάγονται λόγω του μεγάλου εύρους κατανομής που εμπεριέχεται στην τεχνική μας. Υλοποιήσαμε τις DHS και τις μελετήσαμε πειραματικά όσον αφορά τόσο το σφάλμα εκτίμησης όσο και τα χαρακτηριστικά απόδοσής τους. Τα πειραματικά αποτελέσματα τεκμηριώνουν τους ισχυρισμούς μας για μικρά σφάλματα και μικρό επιπλέον κόστος για αποθήκευση και εύρος ζώνης, ενώ ταυτόχρονα δείχνουν την απόδοση τους και την κλιμακωσιμότητα της διαδικασίας μέτρησης.

Ο χώρος σχεδίασης που προκύπτει είναι αρκετά μεγάλος, με πολλές παραμέτρους που μπορούν να πειραχτούν με βάση την προτίμηση του σχεδιαστή του συστήματος. Για λόγους παρουσίασης, παρείχαμε και συζητήσαμε πειραματικά αποτελέσματα για μόνο μερικές δυνατές ρυθμίσεις και παρουσιάσαμε την ανταλλαγή μεταξύ της απόδοσης του πλήθους μεταβάσεων, την ακρίβεια της πρόβλεψης και τη δίκαιη κατανομή φόρτου. Μία υλοποίηση των DHS σε πραγματικές συνθήκες δεν θα έκανε τόσο ξεκάθαρες επιλογές, όσον αφορά τις λεπτομέρειες της σχεδίασης. Για παράδειγμα μία ενδιαφέρουσα επιλογή ρυθμίσεων για το σενάριο ροής θα ήταν η χρησιμοποίηση μίας

υβριδικής μορφής με επαναληπτική δρομολόγηση, εισαγωγές ενός μόνο αντικειμένου, διατήρηση αντιγράφων και συντομεύσεις δρομολόγησης για την εισαγωγή αντικειμένων, καθώς και απλή επαναληπτική δρομολόγηση χωρίς συντομεύσεις και με έλεγχο θέσεων δυαδικής αναζήτησης για το κομμάτι του υπολογισμού της εκτίμησης. Από την άλλη πλευρά, για την περίπτωση συστημάτων διαχείρισης δεδομένων, μία ρύθμιση με μαζικές εισαγωγές κρυμμένες στην διαχειριστική κίνηση του επιπέδου δρομολόγησης, καθώς και σειριακό αναδρομικό έλεγχο με διατήρηση κρυφών αντιγράφων του αποτελέσματος, μοιάζει να είναι η καλύτερη επιλογή. Σκοπεύουμε να δοκιμάσουμε κι άλλους τέτοιους συνδυασμούς στο μέλλον.

Επιπλέον έχουν αναπτύξει μία νέα πλατφόρμα για κατανεμημένες στατιστικές συνόψεις που είναι ιδιαίτερα ταιριαστή για το περιβάλλον των ΔΟΕ. Η σκέψη μας ήταν να ξεκινήσουμε με τις περισσότερες γνωστές τεχνικές για κεντριοποιημένα συστήματα και να τις επεκτείνουμε για κατανεμημένα συστήματα με ιδιαίτερη προσοχή στην δυνατότητα κλιμάκωσης. Η συνεισφορά μας έγκειται στην χρησιμοποίηση συγκεκριμένων στατιστικών δομών, όπως οι συνόψεις κατακερματισμού και οι DHS και στην ανάπτυξη νέων μεθόδων επεξεργασίας σημαντικών τύπων συναθροιστικών ερωτημάτων σε περιβάλλον ΔΟΕ. Με βάση αυτή την υποδομή, δείξαμε πως μπορούν να χτιστούν υψηλότερου επιπέδου συνόψεις βασισμένες σε δίκτυα DHT, όπως ιστογράμματα τύπου Equi-Width, Equi-Depth, Average-Shifted, Compressed και MaxDiff, τα οποία είναι από τα πιο δημοφιλή και αποδεκτά στην βιβλιογραφία των παραδοσιακών τεχνικών διατήρησης στατιστικών για βελτιστοποίηση ερωτημάτων σε συστήματα διαχείρισης σχεσιακών βάσεων δεδομένων. Η υλοποίησή μας και η αναλυτική αξιολόγηση της απόδοσης των λύσεών μας, δείχνουν ότι αυτές οι δομές μπορούν να κατασκευαστούν και να χρησιμοποιηθούν αποδοτικά και ότι κλιμακώνονται καλά για αυξανόμενα μεγέθη του δικτύου, ενώ επιτυγχάνουν υψηλή ακρίβεια. Η υλοποίηση των λύσεών μας, πάνω από το δημοφιλές και σταθερό δίκτυο DHT FreePastry, είναι διαθέσιμη υπό μορφή ανοιχτού κώδικα στον παγκόσμιο ιστό [39].

Υπάρχουν ακόμα κάποιες κατευθύνσεις για μελλοντική έρευνα. Πρώτα θα θέλαμε να εξετάσουμε την σχεδίαση και υλοποίηση ενός μηχανισμού αυτόματης βελτιστοποίησης/παραμετροποίησης για τη μηχανή εξαγωγής ιστογραμμάτων μας· τα προβλήματα αυτά έχουν αντιμετωπιστεί για συστήματα κεντριοποιημένων βάσεων δεδομένων [3], αλλά δεν έχουν ερευνηθεί από την ερευνητική επιστημονική κοινότητα των δικτύων ομοτίμων. Η ενσωμάτωση των λύσεών μας με μηχανές επεξεργασίας ερωτημάτων στο διαδίκτυο στο πνεύμα της [61] έρχεται επόμενη στην λίστα μας· πιστεύουμε ότι το αποτέλεσμα θα μας φέρει πιο κοντά στον στόχο μας για πλήρως αυτο-οργανούμενες λύσεις ΔΟΕ για προχωρημένη διαχείριση δεδομένων. Τρίτον, σκοπεύουμε να εξετάσουμε την υλοποίηση περαιτέρω τύπων συνόψεων, συναθροιστικών ερωτημάτων και ιστογραμμάτων. Τέλος, η χρήση των εργαλείων μας για προσεγγιστική επεξεργασία ερωτημάτων, στο πνεύμα των [19, 68], θα ήταν μία πολύ ενδιαφέρουσα κατεύθυνση έρευνας.

Βιβλιογραφία

- [1] K. Aberer, A. Datta, M. Hauswirth, and R. Schmidt. Indexing data-oriented overlay networks. In *Proc. 31st International Conference on Very Large Databases (VLDB)*, 2005.
- [2] K. Aberer, M. Hauswirth, M. Puceva, and R. Schmidt. Improving data access in P2P systems. *IEEE Internet Computing*, 6(1), Jan/Feb 2002.
- [3] A. Aboulmaga and S. Chaudhuri. Self-tuning histograms: Building histograms without looking at data. In *Proc. ACM SIGMOD International Conference on Management of Data (SIGMOD)*, 1999.
- [4] S. Acharaya, P. B. Gibbons, V. Poosala, and S. Ramaswamy. The Aqua approximate query answering system. In *Proc. ACM SIGMOD International Conference on Management of Data (SIGMOD)*, 1999.
- [5] E. Adar and B. A. Huberman. Free riding on Gnutella. Technical report, Xerox PARC, 2000.
- [6] N. Alon, P. B. Gibbons, Y. Matias, and M. Szegedy. Tracking join and self-join sizes in limited storage. In *Proc. 18th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS)*, 1999.
- [7] A. Andrzejak and Z. Xu. Scalable, efficient range queries for grid information services. In *Proc. 2nd IEEE International Conference on Peer-to-Peer Computing (P2P)*, 2002.
- [8] J. Aspnes and G. Shah. Skip Graphs. In *Proc. 14th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2003.
- [9] R. Avnur and J. M. Hellerstein. Eddies: Continuously adaptive query processing. In *Proc. ACM SIGMOD International Conference on Management of Data (SIGMOD)*, 2000.
- [10] Ö. Babaoğlu, H. Meling, and A. Montresor. Anthill: A framework for the development of agent-based peer-to-peer systems. In *Proc. 22th IEEE International Conference on Distributed Computing and Systems (ICDCS)*, 2002.
- [11] M. Bawa, H. Garcia-Molina, A. Gionis, and R. Motwani. Estimating aggregates on a peer-to-peer network, 2003. Submitted (<http://dbpubs.stanford.edu/pub/2003-24>).
- [12] M. Bawa, A. Gionis, H. Garcia-Molina, and R. Motwani. The price of validity in dynamic networks. In *Proc. ACM SIGMOD International Conference on Management of Data (SIGMOD)*, 2004.

- [13] A. Bharambe, M. Agrawal, and S. Seshan. Mercury: Supporting scalable multi-attribute range queries. In *Proc. ACM Symposium on Communications Architectures and Protocols (SIGCOMM)*, 2004.
- [14] B. Blohsfeld, D. Korus, and B. Seeger. A comparison of selectivity estimators for range queries on metric attributes. In *Proc. ACM SIGMOD International Conference on Management of Data (SIGMOD)*, 1999.
- [15] A. Carzaniga, D. S. Rosenblum, and A. L. Wolf. Design and evaluation of a wide-area event notification service. *ACM Transactions on Computer Systems (TOCS)*, 19(3), 2001.
- [16] S. Chaudhuri. An overview of query optimization in relational systems. In *Proc. 17th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS)*, 1998.
- [17] S. Chaudhuri, R. Motwani, and R. Narasayya. Random sampling for histogram construction: How much is enough? In *Proc. ACM SIGMOD International Conference on Management of Data (SIGMOD)*, 1998.
- [18] S. Chaudhuri, G. Das, and V. R. Narasayya. A robust, optimization-based approach for approximate answering of aggregate queries. In *Proc. ACM SIGMOD International Conference on Management of Data (SIGMOD)*, 2001.
- [19] S. Chaudhuri, G. Das, and V. R. Narasayya. Optimized stratified sampling for approximate query processing. *ACM Transactions on Database Systems (TODS)*, 32(2):9, 2007.
- [20] Y. Chawathe, et al. Making Gnutella-like P2P systems scalable. In *Proc. ACM Symposium on Communications Architectures and Protocols (SIGCOMM)*, 2003.
- [21] I. Clarke, O. Sandberg, B. Wiley, and T. W. Hong. Freenet: A distributed anonymous information storage and retrieval system. In *Proc. ICSI Workshop on Design Issues in Anonymity and Unobservability*, 2000.
- [22] J. Considine, F. Li, G. Kollios, and J. Byers. Approximate aggregation techniques for sensor databases. In *Proc. 20th International Conference on Data Engineering (ICDE)*, 2004.
- [23] G. Cormode and S. Muthukrishnan. An improved data stream summary: The count-min sketch and its applications. In *Proc. 6th Latin American Symposium on Theoretical Informatics (LATIN)*, 2004.
- [24] G. Cormode and S. Muthukrishnan. Space efficient mining of multigraph streams. In *Proc. 24th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS)*, 2005.
- [25] G. Cormode and M. Garofalakis. Sketching streams through the net: Distributed approximate query tracking. In *Proc. 31st International Conference on Very Large Databases (VLDB)*, 2005.
- [26] A. Crainiceanu, P. Linga, J. Gehrke, and J. Shanmugasundaram. Querying peer-to-peer networks using P-trees. In *Proc. 7th International Workshop on the Web and Databases (WebDB)*, 2004.
- [27] F. Dabek, J. Li, E. Sit, J. Robertson, M. F. Kaashoek, and R. Morris. Designing a DHT for low latency and high throughput. In *Proc. 1st USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, 2004.

- [28] C. Damgaard and J. Weiner. Describing inequality in plant size or fecundity. *Ecology*, 81:1139–1142, 2000.
- [29] A. Dobra. Histograms revisited: When are histograms the best approximation method for aggregates over joins? In *Proc. 24th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS)*, 2005.
- [30] A. Dobra, M. Garofalakis, J. Gehrke, and R. Rastogi. Sketch-based multi-query processing over data streams. In *Proc. 9th International Conference on Extending Database Technology (EDBT)*, 2004.
- [31] J. R. Douceur. The Sybil attack. In *Proc. 1st International Workshop on Peer-to-Peer Systems (IPTPS)*, 2002.
- [32] P. Druschel and A. Rowstron. Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. In *Proc. IFIP/ACM IFIP/ACM International Conference on Distributed Systems Platforms (Middleware)*, 2001.
- [33] M. Durand and P. Flajolet. Loglog counting of large cardinalities. In *Proc. 11th Annual European Symposium on Algorithms (ESA)*, 2003.
- [34] L. Fan, P. Cao, J. Almeida, and A. Z. Broder. Summary cache: A scalable wide-area web cache sharing protocol. In *Proc. ACM Symposium on Communications Architectures and Protocols (SIGCOMM)*, 1998.
- [35] P. Flajolet and G. N. Martin. Probabilistic counting algorithms for data base applications. *Journal of Computer and System Sciences*, 31(2):182–209, October 1985.
- [36] M. J. Franklin, J. M. Hellerstein, and S. Madden. Thinking big about tiny databases. *IEEE Data Engineering Bulletin*, 30(3):37–48, 2007.
- [37] FreeDHS. Homepage, 2006. <http://netcins.ceid.upatras.gr/DHS.php>.
- [38] FreePastry. Homepage, 2002. <http://freepastry.org/FreePastry/>.
- [39] FreeSHADE, 2006. <http://netcins.ceid.upatras.gr/SHADE.php>.
- [40] P. Ganesan, M. Bawa, and H. Garcia-Molina. Online balancing of range-partitioned data with applications to peer-to-peer systems. In *Proc. International Conference on Very Large Data Bases (VLDB)*, 2004.
- [41] P. Ganesan, B. Yang, and H. Garcia-Molina. Multi-dimensional indexing in peer-to-peer systems. In *Proc. 7th International Workshop on the Web and Databases (WebDB)*, 2004.
- [42] S. Ganguly, M. Garofalakis, and R. Rastogi. Processing set expressions over continuous update streams. In *Proc. ACM SIGMOD International Conference on Management of Data (SIGMOD)*, 2003.
- [43] S. Ganguly, P. B. Gibbons, Y. Matias, and A. Silberschatz. Bifocal sampling for skew-resistant join size estimation. In *Proc. ACM SIGMOD International Conference on Management of Data (SIGMOD)*, 1996.
- [44] M. Garofalakis, editor. Special Issue on In-Network Query Processing. *IEEE Data Engineering Bulletin*, 28(1), 2005.

- [45] Gnutella. Homepage, 2001. <http://gnutella.wego.com/>.
- [46] B. Godfrey, K. Lakshminarayanan, S. Surana, R. M. Karp, and I. Stoica. Load balancing in dynamic structured p2p systems. In *Proc. 23rd Conference of the IEEE Communications Society (INFOCOM)*, 2004.
- [47] J. Gray, D. T. Liu, M. A. Nieto-Santisteban, A. S. Szalay, David J. DeWitt, and Gerd Heber. Scientific data management in the coming decade. *SIGMOD Record*, 34(4):34–41, 2005.
- [48] S. Gribble, A. Halevy, Z. Ives, M. Rodrig, and D. Suciu. What can peer-to-peer do for databases, and vice versa? In *Proc. 4th International Workshop on the Web and Databases (WebDB)*, 2001.
- [49] S. Guha, N. Koudas, and K. Shim. Data-streams and histograms. In *Proc. 33rd Annual ACM Symposium on Theory of Computing (STOC)*, 2001.
- [50] K. Gummadi, R. Gummadi, S. Gribble, S. Ratnasamy, S. Shenker, and I. Stoica. The impact of DHT routing geometry on resilience and proximity. In *Proc. ACM Symposium on Communications Architectures and Protocols (SIGCOMM)*, 2003.
- [51] K. P. Gummadi, R. J. Dunn, S. Saroiu, S. D. Gribble, H. M. Levy, and J. Zahorjan. Measurement, modeling, and analysis of a peer-to-peer file-sharing workload. In *Proc. 19th ACM Symposium on Operating Systems Principles (SOSP)*, pages 314–329, 2003.
- [52] A. Gupta, D. Agrawal, and A. El Abbadi. Approximate range selection queries in peer-to-peer systems. In *Proc. 1st ACM SIGMOD/VLDB Biennial Conference on Innovative Data Systems Research (CIDR)*, 2003.
- [53] A. Gupta, B. Liskov, and R. Rodrigues. One hop lookups for peer-to-peer overlays. In *Proc. 9th Workshop on Hot Topics in Operating Systems (HotOS)*, 2003.
- [54] M. Hadjieleftheriou, J. W. Byers, and G. Kollios. Robust sketching and aggregation of distributed data streams. Technical Report 2005-011, Computer Science Dept., Boston University, 2005.
- [55] M. Harren, J. M. Hellerstein, R. Huebsch, B. T. Loo, S. Shenker, and I. Stoica. Complex queries in DHT-based peer-to-peer networks. In *Proc. 1st International Workshop on Peer-to-Peer Systems (IPTPS)*, 2002.
- [56] N. Harvey, M. Jones, S. Saroiu, M. Theimer, and A. Wolman. Skipnet: A scalable overlay network with practical locality properties. In *Proc. 4th USENIX Symposium on Internet Technologies and Systems (USITS)*, 2003.
- [57] J. M. Hellerstein, T. Condie, M. Garofalakis, B. T. Loo, P. Maniatis, T. Roscoe, and N. Taft. Public health for the internet (phi). In *Proc. 3rd ACM SIGMOD/VLDB Biennial Conference on Innovative Data Systems Research (CIDR)*, 2007.
- [58] J. M. Hellerstein, P. J. Haas, and H. J. Wang. Online aggregation. In *Proc. ACM SIGMOD International Conference on Management of Data (SIGMOD)*, 1997.
- [59] R. Huebsch, J. M. Hellerstein, N. Lanham, B. T. Loo, S. Shenker, and I. Stoica. Querying the internet with PIER. In *Proc. 29th International Conference on Very Large Data Bases (VLDB)*, 2003.

- [60] R. Huebsch and S. Jeffery. FREddies: DHT-based adaptive query processing via Federated Eddies. Technical Report UCB/CSD-4-1339, U. of California at Berkeley, 2004.
- [61] R. Huebsch, B. N. Chun, J. M. Hellerstein, B. T. Loo, P. Maniatis, T. Roscoe, S. Shenker, I. Stoica, and A. R. Yumerefendi. The architecture of PIER: an internet-scale query processor. In *Proc. 2nd Biennial Conference on Innovative Data Systems Research (CIDR)*, 2005.
- [62] Y. Ioannidis. The history of histograms. In *Proc. 29th International Conference on Very Large Data Bases (VLDB)*, 2003.
- [63] Y. Ioannidis and S. Christodoulakis. Optimal histograms for limiting worst-case error propagation in the size of join results. *ACM Transactions on Database Systems (TODS)*, 18(4):709–748, December 1993.
- [64] Y. Ioannidis and V. Poosala. Balancing histogram optimality and practicality for query result size estimation. In *Proc. ACM SIGMOD International Conference on Management of Data (SIGMOD)*, 1995.
- [65] Z. Ives, N. Khandelwal, A. Kapur, and M. Cakir. ORCHESTRA: Rapid, collaborative sharing of dynamic data. In *Proc. 2nd Biennial Conference on Innovative Data Systems Research (CIDR)*, 2005.
- [66] R. Jain, D. Chiu, and W. Hawe. A quantitative measure of fairness and discrimination for resource allocation in shared computer systems. Technical Report TR-301, DEC, 1984.
- [67] M. Jelasity and A. Montresor. Epidemic-style proactive aggregation in large overlay networks. In *Proc. 24th International Conference on Distributed Computing Systems (ICDCS)*, 2004.
- [68] C. M. Jermaine, S. Arumugam, A. Pol, and A. Dobra. Scalable approximate query processing with the dbo engine. In *Proc. ACM SIGMOD International Conference on Management of Data (SIGMOD)*, 2007.
- [69] P. Kalnis, W. S. Ng, B. C. Ooi, and K. L. Tan. Answering similarity queries in peer-to-peer networks. *Information Systems*, 31, March 2006.
- [70] D. Karger, E. Lehman, T. Leighton, M. Levine, D. Lewin, and R. Panigrahy. Consistent hashing and random trees: Distributed caching protocols for relieving hot spots on the world wide web. In *Proc. 29th Annual ACM Symposium on Theory of Computing (STOC)*, 1997.
- [71] D. Karger and M. Ruhl. New algorithms for load balancing in P2P systems. In *Proc. 3rd International Workshop on Peer-to-Peer Systems (IPTPS)*, 2004.
- [72] D. Kempe, A. Dobra, and J. Gehrke. Computing aggregate information using gossip. In *Proc. 44th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2003.
- [73] G. Koloniari and E. Pitoura. Content-based routing of path queries in peer-to-peer systems. In *Proc. 9th International Conference on Extending Database Technology (EDBT)*, 2004.
- [74] A. Kossmann. The state of the art in distributed query processing. *ACM Computing Surveys*, 32(4), December 2000.

- [75] N. Koudas, editor. Special Issue on Data Quality. *IEEE Data Engineering Bulletin*, 29(2), 2006.
- [76] P. Krishnan. *Online Prediction Algorithms for Databases and Operating Systems*. PhD thesis, Brown University, April 1995.
- [77] J. Li, B. T. Loo, J. M. Hellerstein, M. F. Kaashoek, D. Karger, and R. Morris. On the feasibility of peer-to-peer web indexing and search. In *Proc. 2nd International Workshop on Peer-to-Peer Systems (IPTPS)*, 2003.
- [78] D. Liben-Nowell, H. Balakrishnan, and D. Karger. Observations on the dynamic evolution of peer-to-peer networks. In *Proc. 1st International Workshop on Peer-to-Peer Systems (IPTPS)*, 2002.
- [79] R. Lipton and J. F. Naughton. Query size estimation by adaptive sampling. In *Journal of Computer and System Sciences*, vol. 51, issue 1, pp. 18-25, 1995.
- [80] R. J. Lipton, J. F. Naughton, and D. A. Schneider. Practical selectivity estimation through adaptive sampling. In *Proc. ACM SIGMOD International Conference on Management of Data (SIGMOD)*, 1990.
- [81] Q. Lv, S. Ratnasamy, and S. Shenker. Can heterogeneity make Gnutella scalable? In *Proc. 1st International Workshop on Peer-to-Peer Systems (IPTPS)*, 2002.
- [82] D. Malkhi, M. Naor, and D. Ratajczak. Viceroy: A scalable and dynamic emulation of the butterfly. In *Proc. 21st ACM Symposium on Principles of Distributed Computing (PODC)*, 2002.
- [83] G. Manku. Routing networks for Distributed Hash Tables. In *Proc. 22nd ACM Symposium on Principles of Distributed Computing (PODC)*, 2003.
- [84] P. Maymoukhnov and D. Mazières. Kademlia: A peer-to-peer information system based on the XOR metric. In *Proc. 1st International Workshop on Peer-to-Peer Systems (IPTPS)*, 2002.
- [85] S. Michel, M. Bender, N. Ntarmos, P. Triantafillou, G. Weikum, and C. Zimmer. Discovering and exploiting keyword and attribute-value co-occurrences to improve P2P routing indices. In *Proc. 15th ACM Conference on Information and Knowledge Management (CIKM)*, **Recipient of the Best Interdisciplinary Paper Award**, 2006.
- [86] S. Michel, P. Triantafillou, and G. Weikum. KLEE: A framework for distributed top-k query algorithms. In *Proc. 31st International Conference on Very Large Databases (VLDB)*, 2005.
- [87] Mojonation. Homepage. <http://www.mojonation.com/>.
- [88] A. Mondal, K. Goda, and M. Kitsuregawa. Effective load-balancing via migration and replication in spatial grids. In *Proc. 14th International Conference on Database and Expert Systems Applications (DEXA)*, 2003.
- [89] A. Montresor, M. Jelasity, and Ö. Babaoğlu. Robust aggregation protocols for large-scale overlay networks. In *Proc. International Conference on Dependable Systems and Networks (DSN)*, 2004.
- [90] A. Montresor, H. Meling, and Ö. Babaoğlu. Messor: Load-balancing through a swarm of autonomous agents. In *Proc. Workshop on Agent and Peer-to-Peer Systems*, 2002.

- [91] J. I. Munro and M. S. Paterson. Selection and sorting with limited storage. *Theoretical Computer Science*, pages 315–323, 1980.
- [92] M. Muralikrishna and D. J. DeWitt. Equi-depth histograms for estimating selectivity factors for multi-dimensional queries. In *Proc. ACM SIGMOD International Conference on Management of Data (SIGMOD)*, 1988.
- [93] W. S. Ng, B. C. Ooi, K. L. Tan, and A. Zhou. PeerDB: A P2P-based system for distributed data sharing. In *Proc. 19th International Conference on Data Engineering (ICDE)*, 2003.
- [94] N. Ntarmos, T. Pitoura, and P. Triantafillou. Range query optimization leveraging peer heterogeneity in DHT data networks. In *Proc. International Workshop on Databases, Information Systems and Peer-to-Peer Computing (DBISP2P)*, 2005.
- [95] N. Ntarmos and P. Triantafillou. AESOP: Altruism-endowed self-organizing peers. In *Proc. International Workshop on Databases, Information Systems and Peer-to-Peer Computing (DBISP2P)*, 2004.
- [96] N. Ntarmos and P. Triantafillou. SeAl: Managing accesses and data in peer-to-peer sharing networks. In *Proc. IEEE International Conference on Peer-to-Peer Computing (P2P)*, 2004.
- [97] N. Ntarmos, P. Triantafillou, and G. Weikum. Counting at large: Efficient cardinality estimation in internet-scale data networks. In *Proc. 22nd International Conference on Data Engineering (ICDE)*, 2006.
- [98] N. Ntarmos, P. Triantafillou, and G. Weikum. Statistical structures for internet-scale data management. *Journal on Very Large Data Bases (VLDBJ)* (to appear), 2009.
- [99] N. Ntarmos, P. Triantafillou, and G. Weikum. Distributed hash sketches: Scalable, efficient cardinality estimation for distributed multisets. *ACM Transactions on Computer Systems (TOCS)*, 27(1), February 2009.
- [100] C. R. Palmer, G. Sigamos, M. Faloutsos, C. Faloutsos, and P. B. Gibbons. The connectivity and fault-tolerance of the internet topology. In *Proc. Workshop on Network-Related Data Management (NRDM)*, 2001.
- [101] V. Papadimos, D. Maier, and K. Tufte. Distributed query processing and catalogs for peer-to-peer systems. In *Proc. 1st ACM SIGMOD/VLDB Biennial Conference on Innovative Data Systems Research (CIDR)*, 2003.
- [102] T. Pitoura, N. Ntarmos, and P. Triantafillou. Replication, load balancing, and efficient range query processing in dht data networks. In *Proc. 10th International Conference on Extending Database Technology (EDBT)*, 2006.
- [103] T. Pitoura and P. Triantafillou. Load distribution fairness in p2p data management systems. In *Proc. 23rd International Conference on Data Engineering (ICDE)*, 2007.
- [104] V. Poosala, V. Ganti, and Y. Ioannidis. Approximate query answering using histograms. *IEEE Data Engineering Bulletin*, 22(4), December 1999.
- [105] V. Poosala and Y. Ioannidis. Estimation of query-result distribution and its application in parallel-join load balancing. In *Proc. 22nd International Conference on Very Large Data Bases (VLDB)*, 1996.

- [106] V. Poosala and Y. Ioannidis. Selectivity estimation without the attribute value independence assumption. In *Proc. 23rd International Conference on Very Large Data Bases (VLDB)*, 1997.
- [107] V. Poosala, Y. Ioannidis, P. Haas, and E. Shekita. Improved histograms for selectivity estimation of range predicates. In *Proc. ACM SIGMOD International Conference on Management of Data (SIGMOD)*, 1996.
- [108] A. Rao, K. Lakshminarayanan, S. Surana, R. Karp, and I. Stoica. Load balancing in structured P2P systems. In *Proc. 2nd International Workshop on Peer-to-Peer Systems (IPTPS)*, 2003.
- [109] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. A scalable content-addressable network. In *Proc. ACM Symposium on Communications Architectures and Protocols (SIGCOMM)*, 2001.
- [110] S. Ratnasamy, S. Shenker, and I. Stoica. Routing algorithms for DHTs: Some open questions. In *Proc. 1st International Workshop on Peer-to-Peer Systems (IPTPS)*, 2002.
- [111] P. Reynolds and A. Vahdat. Efficient peer-to-peer keyword searching. In *Proc. IFIP/ACM IFIP/ACM International Conference on Distributed Systems Platforms (Middleware)*, pages 21–40, 2003.
- [112] S. Rhea, D. Geels, T. Roscoe, and J. Kubiatowicz. Handling churn in a DHT. In *Proc. of the USENIX Annual Technical Conference (USENIX)*, 2004.
- [113] O.D. Sahin, A. Gupta, D. Agrawal, and A.E. Abbadi. Query processing over peer-to-peer data sharing systems. Technical Report UCSB/CSD-2002-28, UC Santa Barbara, 2002.
- [114] S. Saroiu, P. K. Gummadi, and S. D. Gribble. A measurement study of peer-to-peer file sharing systems. In *Proc. Multimedia Computing and Networking Conference (MMCN)*, 2002.
- [115] D. W. Scott. Average shifted histograms: Effective nonparametric density estimators in several dimensions. *The Annals of Statistics*, 13(3):1024–1040, 1985.
- [116] P. G. Selinger, M. M. Astrahan, D. D. Chamberlin, R. A. Lorie, and T. G. Price. Access path selection in a relational database management system. In *Proc. ACM SIGMOD International Conference on Management of Data (SIGMOD)*, 1979.
- [117] S. Sen and J. Wang. Analyzing peer-to-peer traffic accross large networks. *IEEE/ACM Transactions on Networking (TON)*, 12(2):219–232, 2004.
- [118] G. P. Shapiro and C. Connell. Accurate estimation of the number of tuples satisfying a condition. In *Proc. ACM SIGMOD International Conference on Management of Data (SIGMOD)*, 1984.
- [119] K. Sripanidkulchai. The popularity of gnutella queries and its implications on scalability, Feb. 2001. White paper.
- [120] R. Steinmetz and K. Wehrle, editors. *Peer-to-Peer Systems and Applications*. Springer, September 2005.

- [121] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. Chord: A scalable Peer-To-Peer lookup service for internet applications. In *Proc. ACM Symposium on Communications Architectures and Protocols (SIGCOMM)*, 2001.
- [122] D. Suci, editor. Special Issue on Web-Scale Data, Systems, and Semantics. *IEEE Data Engineering Bulletin*, 29(4), 2006.
- [123] Y. Tao, G. Kollios, J. Considine, F. Li, and D. Papadias. Spatio-temporal aggregation using sketches. In *Proc. 20th International Conference on Data Engineering (ICDE)*, 2004.
- [124] P. Triantafillou and A. Economides. Subscription summarization: A new paradigm for efficient publish/subscribe systems. In *Proc. 24th International Conference on Distributed Computing Systems (ICDCS)*, 2004.
- [125] P. Triantafillou and T. Pitoura. Towards a unifying framework for complex query processing over structured peer-to-peer data networks. In *Proc. International Workshop on Databases, Information Systems and Peer-to-Peer Computing (DBISP2P)*, 2003.
- [126] R. van Renesse, K. P. Birman, and W. Vogels. Astrolabe: A robust and scalable technology for distributed system monitoring, management, and data mining. *ACM Transactions on Computer Systems (TOCS)*, 21(2):164–206, May 2003.
- [127] S. Wang, B. C. Ooi, A. K.H. Tung, and L. Xu. Efficient skyline query processing on peer-to-peer networks. In *Proc. 23rd IEEE International Conference on Data Engineering (ICDE)*, 2007, 2007.
- [128] B. Wilcox-O’Hearn. Experiences deploying a large-scale emergent network. In *Proc. 1st International Workshop on Peer-to-Peer Systems (IPTPS)*, 2002.
- [129] A. N. Wilschut and P. M.G. Apers. Dataflow query execution in a parallel main-memory environment. *Distributed and Parallel Databases*, 1(1), 1993.
- [130] P. Yalagandula and M. Dahlin. A scalable distributed information management system. In *Proc. ACM Symposium on Communications Architectures and Protocols (SIGCOMM)*, 2004.
- [131] B. Yang and H. Garcia-Molina. Comparing hybrid peer-to-peer systems. In *Proc. International Conference on Very Large Data Bases (VLDB)*, 2001.
- [132] B. Y. Zhao, J. D. Kubiatowicz, and A. D. Joseph. Tapestry: An infrastructure for fault-tolerant wide-area location and routing. Technical Report UCB/CSD-01-1141, U. of California at Berkeley, CSD, 2001.