ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΑΤΡΩΝ
Τμήμα Μηχανικών Ηλεκτρονικών Υπολογιστών και Πληροφορικής

ΑΝΑΠΤΥΞΗ ΣΥΣΤΗΜΑΤΩΝ ΔΗΜΟΣΙΕΥΣΕΩΝ/ΣΥΝΔΡΟΜΩΝ
ΣΕ ΔΟΜΗΜΕΝΑ ΔΙΚΤΥΑ ΟΜΟΤΙΜΩΝ ΕΤΑΙΡΩΝ

ΙΩΑΝΝΗΣ ΑΙΚΑΤΕΡΙΝΙΔΗΣ

Δίπλωμα - Τμήμα Ηλεκτρονικών Μηχ/κών και Μηχ/κών Η/Υ, Πολυτεχνείο Κρήτης - 2001
Μ.Δ.Ε. - Τμήμα Ηλεκτρονικών Μηχ/κών και Μηχ/κών Η/Υ, Πολυτεχνείο Κρήτης - 2003

Διδακτορική Διατριβή

Προς το Τμήμα Μηχανικών Ηλεκτρονικών Υπολογιστών και Πληροφορικής
Πολυτεχνικής Σχολής, του Πανεπιστημίου Πατρών
στα πλαίσια εκπλήρωσης των υποχρεώσεων για την απονομή
Διδακτορικού Διπλώματος

Επιβλέπων: Καθηγητής Τριανταφύλλου Παναγιώτης

Πάτρα, Ελλάδα
Απρίλιος 2008

Η Επταμελής Εξεταστική Επιτροπή βεβαιώνει ότι αυτή είναι η εγκεκριμένη έκδοση της διδακτορικής διατριβής με θέμα

ΑΝΑΠΤΥΞΗ ΣΥΣΤΗΜΑΤΩΝ ΔΗΜΟΣΙΕΥΣΕΩΝ/ΣΥΝΔΡΟΜΩΝ
ΣΕ ΔΟΜΗΜΕΝΑ ΔΙΚΤΥΑ ΟΜΟΤΙΜΩΝ ΕΤΑΙΡΩΝ

Η Επταμελής Επιτροπή

Καθ. Παναγιώτης Τριανταφύλλου (Επιβλέπων)

Καθ. Εμμανουήλ Βαρβαρίγος

Καθ. Ευστράτιος Γαλλόπουλος

Αναπλ. Καθ. Ιωάννης Γαροφαλάκης

Αναπλ. Καθ. Εμμανουήλ Κουμπάρκης

Αναπλ. Καθ. Ευαγγελία Πιτουρά

Καθ. Αθανάσιος Τσακαλίδης

© Ιωάννης Αικατερινίδης - Απρίλιος 2008

Περίληψη

Τα τελευταία χρόνια οι εφαρμογές συνεχούς μετάδοσης ροών πληροφορίας στο διαδίκτυο έχουν γίνει ιδιαίτερα δημοφιλείς. Με τον συνεχώς αυξανόμενο ρυθμό εισόδου νέων αντικειμένων πληροφορίας, γίνεται ολοένα και πιο επιτακτική η ανάγκη για την ανάπτυξη Πληροφορικών Συστημάτων που να μπορούν να προσφέρουν στους χρήστες τους μόνο εκείνες τις πληροφορίες που τους ενδιαφέρουν, φιλτράροντας τεράστιους όγκους από άσχετες για τον κάθε χρήστη, πληροφορίες. Ένα μοντέλο διάδοσης πληροφορίας ικανό να ενσωματώσει τέτοιου είδους ιδιότητες, είναι το μοντέλο δημοσιεύσεων/συνδρομών (publish/subscribe). Η πιο ενδιαφέρουσα κατηγορία συστημάτων δημοσιεύσεων/συνδρομών είναι τα λεγόμενα content-based publish/subscribe systems, που δίνουν στο χρήστη μεγαλύτερη εκφραστικότητα κατά τη δήλωση των προτιμήσεών του. Στα content-based pub/sub συστήματα, όπου επικεντρώνεται και η συγκεκριμένη διατριβή, οι χρήστες γνωστοποιούν τα ενδιαφέροντα τους δηλώνοντας συνδρομές με συγκεκριμένους περιορισμούς/συνθήκες πάνω στην ίδια την πληροφορία που διακινείται. Στη συνέχεια χρησιμοποιώντας αυτά τα «φίλτρα» παραδίδεται από το σύστημα στον κάθε συνδρομητή μόνο η σχετική με τα ενδιαφέροντά του πληροφορία. Συνήθεις τύποι τελεστών που μπορούν να χρησιμοποιηθούν πάνω στη πληροφορία κατά τον ορισμό των ενδιαφερόντων των χρηστών περιλαμβάνουν (α) τους τελεστές ισότητας, $=$, $\leq$, $\geq$, και διαστημάτων τιμών για αριθμητικούς τύπους και (β) τους τελεστές ισότητας, προθέματος, επιθέματος και υποσυμβολοσειράς (substring) σε δεδομένα τύπου συμβολοσειράς. Τα τελευταία χρόνια έχουν δημοσιευθεί πολλές ερευνητικές εργασίες που αφορούν στην κεντριοποιημένη προσέγγιση του μοντέλου δημοσιεύσεων συνδρομών, επικεντρώνοντας κυρίως στην ανάπτυξη δομών δεικτοδότησης για τις συνδρομές και κατάλληλους αλγόριθμους αποδοτικής προσπέλασης των δομών αυτών για κάθε νεοεισερχόμενη πληροφορία με στόχο την ανεύρεση σχετικών συνδρομών. Κατά την περίοδο έναρξης αυτής της διατριβής αποτελούσε ανοικτό πρόβλημα η ανάπτυξη κατάλληλων αρχιτεκτονικών και αλγορίθμων που να μπορούν να προσφέρουν μια πλήρως κατανεμημένη, κλιμακώσιμη και αποδοτική λύση για την εφαρμογή διαδικτυακής κλίμακας content-based pub/sub συστημάτων.

Ένα ιδιαίτερα δημοφιλές μοντέλο κατανεμημένων εφαρμογών είναι το μοντέλο ομοτίμων με τα πλέον πολύ δημοφιλή δίκτυα ομοτίμων (peer-to-peer networks). Αυτά τα δίκτυα είναι δίκτυα overlay, (όπου κυριαρχεί η έννοια των λογικών συνδέσεων μεταξύ των κόμβων, με μία σύνδεση μεταξύ δύο κόμβων να απο-

τελεί στη πραγματικότητα ένα μονοπάτι στο tcp/ip δίκτυο). Κυρίαρχες έννοιες στα p2p δίκτυα είναι αυτές της κοινοκτημοσύνης των διαθέσιμων πόρων, της συνεργατικότητας και της παντελούς απουσίας κεντρικού ελέγχου. Η κλιμάκωση, η αξιοπιστία, η επεκτασιμότητα και η διαθεσιμότητα των δικτύων ομοτίμων εταιρών τα καθιστούν ιδιαίτερα δημοφιλή για την ανάπτυξη εφαρμογών που ακριβώς αναζητούν αυτές τις ιδιότητες. Στα δομημένα δίκτυα ομοτίμων, υπάρχει η έννοια της δομής με βάση την οποία δημιουργείται η τοπολογία και η συνεκτικότητα του δικτύου, η οποία διασφαλίζεται με τη βοήθεια προηγμένων κατανεμημένων δομών δεικτοδότησης, όπως για παράδειγμα των κατανεμημένων πινάκων κατακερματισμού (DHT - Distributed Hash Table). Τα αντικείμενα και οι κόμβοι καταλαμβάνουν ένα χώρο στην τοπολογία του δικτύου με βάση το κλειδί τους που είναι η τιμή που επιστρέφει η συνάρτηση κατακερματισμού. Κάθε κόμβος διαθέτει ένα περιορισμένου μεγέθους πίνακα δρομολόγησης, όπου με τους κατάλληλους αλγορίθμους εξασφαλίζεται ο εντοπισμός ενός κόμβου/αντικειμένου σε $O(\log N)$ βήματα για ένα δίκτυο με N κόμβους.

Βασική συνεισφορά μας στο χώρο είναι η εφαρμογή του μοντέλου δημοσιεύσεων/συνδρομών βασισμένου στο περιεχόμενο (content-based publish/subscribe) πάνω στα δίκτυα ομοτίμων ώστε να μπορέσουμε να προσφέρουμε στους χρήστες υψηλή εκφραστικότητα κατά την δήλωση των ενδιαφερόντων τους, λειτουργώντας σε ένα πλήρως κατανεμημένο και κλιμακώσιμο περιβάλλον. Ο κορμός των προτεινόμενων λύσεων σε αυτή τη διατριβή είναι: (α) η ανάπτυξη αλγορίθμων για την αποθήκευση των κλειδιών των δημοσιεύσεων σε κατάλληλους κόμβους του δικτύου με βάση τις συνθήκες στο περιεχόμενο που έχουν δηλωθεί και (β) αλγορίθμων δρομολόγησης δημοσιεύσεων στο διαδίκτυο έτσι ώστε να «συναντούν» αυτούς τους κόμβους οι οποίοι περιέχουν συνδρομές που ικανοποιούνται από την πληροφορία της δημοσίευσης.

Αρχικά αναπτύξαμε αποδοτικές τεχνικές και σχετικούς αλγορίθμους που επιτυγχάνουν κατανεμημένη σύζευξη των συνδρομών με τις δημοσιεύσεις και παράδοση αυτών στους τελικούς χρήστες πάνω από οποιοδήποτε δομημένο δίκτυο ομοτίμων εταιρών τύπου DHT. Στη συνέχεια επωφεληθήκαμε από τα συγκεκριμένα χαρακτηριστικά ορισμένων δικτύων ομοτίμων (και συγκεκριμένα αυτών που εκτελούν δρομολόγηση με βάση το πρόθεμα των κλειδιών, όπως τα δίκτυα Pastry, Tapestry, Bamboo) ώστε να επιτύχουμε ακόμα καλύτερη απόδοση κατά την επεξεργασία των συνδρομών και των δημοσιεύσεων. Στα πλαίσια αυτής της εργασίας αναπτύξαμε λύσεις που επιτρέπουν την επεξεργασία συνδρομών/δημοσιεύσεων με περιορισμούς/τιμές τόσο σε αριθμητικούς τύπους δεδομένων όσο και σε συμβολοσειρές. Ειδικότερα, οι προτεινόμενες λύσεις επιτρέπουν στους χρήστες την εφαρμογή τελεστών $=$, $\leq$, $\geq$, και διαστημάτων τιμών σε αριθμητικούς τύπους και επίθεμα, πρόθεμα, ισότητα και substring σε συμβολοσειρές. Εξ όσων γνωρίζουμε, αυτή είναι η μοναδική εργασία που προσφέρει αυτές τις υπηρεσίες σε περιβάλλοντα p2p. Παράλληλα, δεδομένου του ότι ένα δίκτυο ομοτίμων αποτελεί ένα ιδιαίτερα δυναμικό περιβάλλον με συχνές αναχωρήσεις/αφίξεις κόμβων, προτείναμε τεχνικές που διατηρούν την λειτουργικότητα και την συνεκτικότητα των δομών μας,

ακόμα και υπό την παρουσία αναχωρήσεων/αφίξεων, αλλά και σφαλμάτων/αποτυχιών στο δίκτυο.

Οι προτεινόμενοι αλγόριθμοι υλοποιήθηκαν και εξετάσθηκαν ενδελεχώς με προσομοίωση μελετώντας την απόδοσή τους με βάση μετρικές όπως: η δίκαιη κατανομή του φόρτου στους κόμβους του δικτύου από τη διακίνηση μηνυμάτων κατά την επεξεργασία των συνδρομών/δημοσιεύσεων, ο συνολικός αριθμός μηνυμάτων που διακινούνται, ο συνολικός όγκος επιπλέον πληροφορίας που απαιτούν οι αλγόριθμοι να εισέλθει στο δίκτυο (network bandwidth), και ο χρόνος που απαιτείται για την ανεύρεση των συνδρομών που συζευγνύουν με κάθε δημοσίευση.

Στην οικογένεια μου και στην αγαπημένη μου Ελένη.

Ευχαριστίες

Η παρούσα διατριβή δεν θα ήταν δυνατόν να ολοκληρωθεί χωρίς την συμπαράσταση και υποστήριξη αρκετών ανθρώπων. Θα ήθελα να ευχαριστήσω θερμά των επιβλέποντα Καθηγητή Τριανταφύλλου (Peter) Παναγιώτη, που όλα αυτά τα χρόνια με βοήθησε όσο κανένας άλλος. Ο χαρακτήρας του, η διδασκαλία του που με κέρδισε όταν ακόμα ήμουν προπτυχιακός φοιτητής και το πάθος του για έρευνα, με οδήγησαν στο να αγαπήσω αυτό το χώρο και την επιστήμη των υπολογιστών γενικότερα. Τον ευχαριστώ για όσα μου πρόσφερε όλα αυτά τα χρόνια.

Θα ήθελα να ευχαριστήσω την συμβουλευτική επιτροπή αποτελούμενη από τους Καθ. Αθανάσιο Τσακαλίδη και Αναπλ. Καθ. Ιωάννη Γαροφαλάκη για την υποστήριξη και καθοδήγηση που μου προσέφεραν. Επίσης, θα ήθελα να ευχαριστήσω τα υπόλοιπα μέλη της επταμελούς εξεταστικής επιτροπής, Καθ. Ευστράτιο Γαλλόπουλο, Καθ. Εμμανουήλ Βαρβαρίγο, Αναπλ. Καθ. Εμμανουήλ Κουμπάρακη και Αναπλ. Καθ. Ευαγγελία Πιτουρά για την τιμή που μου έκαναν.

Η συμπαράσταση και η βοήθεια που μου προσέφεραν όλα αυτά τα χρόνια οι συνάδελφοι και φίλοι Νίκος Ντάρμος, Γρηγόρης Πράσιнос, Μάνος Καπρίτσος, και Πιτουρά Θεώνη ήταν ανεκτίμητη. Τους ευχαριστώ θερμά.

Τέλος, θα ήθελα να ευχαριστήσω την οικογένεια μου για την αγάπη, κατανόηση και την υποστήριξη που μου προσέφερε όλα αυτά τα χρόνια. Ένα μεγάλο ευχαριστώ στους φίλους μου. Ιδιαίτερα θα ήθελα να ευχαριστήσω την Ελένη Χρυσάφοπούλου που όλα αυτά τα χρόνια μου προσέφερε αμέριστη συμπαράσταση, κατανόηση και αγάπη, βοηθώντας με στο δύσκολο έργο μου.

Η έρευνα που οδήγησε σε αυτή τη διατριβή χρηματοδοτήθηκε εν μέρη από το Ευρωπαϊκό Ταμείο (ΕΚΤ), Επιχειρησιακό Πρόγραμμα Εκπαίδευση και Αρχική Επαγγελματική Κατάρτιση (ΕΠΕΑΕΚ II), και ειδικότερα από το Πρόγραμμα ΠΥΘΑΓΟΡΑΣ I. Επίσης χρηματοδοτήθηκε από το ερευνητικό πρόγραμμα DELIS IST-2004-001907 (Dynamically Evolving Large-Scale Information Systems) μέσω του προγράμματος της Ευρωπαϊκής Ένωσης IST FP6.

Περιεχόμενα

Κατάλογος Πινάκων	xv
Κατάλογος Σχημάτων	xvii
Λίστα Συντμήσεων	xxi
Λίστα Συμβόλων	xxiii
1 Εισαγωγή	1
1.1 Συνεισφορά στη Περιοχή	4
1.2 Δομή της Διατριβής	6
2 Ανασκόπηση Σχετικής Έρευνας	7
2.1 Συστήματα Δικτύων Ομοτίμων Εταίρων	7
2.2 Μη Δομημένα Δίκτυα Ομοτίμων Εταίρων	8
2.2.1 Το Δίκτυο Freenet	8
2.2.2 Το Δίκτυο Gnutella	9
2.2.3 Άλλα Μη δομημένα Δίκτυα Ομοτίμων Εταίρων	11
2.3 Δομημένα Δίκτυα Ομοτίμων Εταίρων	12
2.3.1 Το Δίκτυο Pastry	13
2.3.2 Το Δίκτυο Chord	15
2.3.3 Το Δίκτυο CAN	17
2.4 Συστήματα Δημοσιεύσεων Συνδρομών	18
2.4.1 Κατηγοριοποίηση Συστημάτων Δημοσιεύσεων/Συνδρομών Βασισμένα στο Περιεχόμενο	20
2.5 Ανασκόπηση Γνωστών Συστημάτων Δημοσιεύσεων/Συνδρομών Βασισμένα στο Περιεχόμενο	23
2.5.1 Τα Συστήματα Scribe και Bayeux	23
2.5.2 Το Σύστημα SIENA	24
2.5.3 Το Σύστημα REBECA	25
2.5.4 Gryphon	26
2.5.5 Το Σύστημα Hermes	27
2.5.6 Το Σύστημα Meghdoot	28
2.5.7 Τα Συστήματα MEDYM και Kyra	29
2.6 Διαχείριση Πληροφορίας σε Κατανεμημένο Περιβάλλον	30
3 Το Μοντέλο	33
3.1 Περιγραφή του Μοντέλου Διασύνδεσης Χρηστών	33
3.2 Περιγραφή του Μοντέλου Δημοσιεύσεων/Συνδρομών	35
4 <i>Pygacanthus</i>: Σύστημα Δημοσιεύσεων/Συνδρομών Ανεξάρτητο του Υποκείμενου DHT Δικτύου	39
4.1 Επεξεργασία Συνδρομών με Ιδιοχαρακτηριστικά Αριθμητικού Τύπου	40
4.1.1 Stateless Προσεγγίσεις	40
4.1.2 Stateful Προσεγγίσεις	49
4.2 Επεξεργασία Συνδρομών με Ιδιοχαρακτηριστικά Τύπου Συμβολοσειράς	54
4.2.1 Η Stateless Προσέγγιση	54

4.2.2	Η Stateful Προσέγγιση	55
4.3	Επεξεργασία και Σύζευξη των Δημοσιεύσεων με τις Συνδρομές	57
4.3.1	Συλλογή των Αποθηκευμένων Αναγνωριστικών	57
4.3.2	Σύζευξη των Συνδρομών με τις Δημοσιεύσεις	60
4.4	Αυτόνομη Οργάνωση των Δενδρικών Δομών στις Stateful Προσεγγίσεις	64
4.4.1	Άφιξη Κόμβων	64
4.4.2	Αναχώρηση Κόμβων	65
4.5	Θεωρητική Ανάλυση Απόδοσης	65
4.5.1	Επεξεργασία Αριθμητικών Τιμών	66
4.5.2	Επεξεργασία Συμβολοσειρών	68
4.6	Βελτίωση της Πολυδιάστατης Επεξεργασίας των Δημοσιεύσεων	69
4.6.1	Κεντρικοποιημένη Σύζευξη	69
4.6.2	Κατανεμημένη Σύζευξη	70
4.6.3	Υβριδική Σύζευξη	72
4.7	Πειραματική Αξιολόγηση και Μελέτη Απόδοσης	73
4.7.1	Χαρακτηριστικά και Παράμετροι Πειραματικής Διάταξης	73
4.7.2	Απαιτήσεις σε Χώρο Αποθήκευσης	75
4.7.3	Εξισορρόπηση του Φόρτου Αποθήκευσης και Ανάκτησης των SubIDs	75
4.7.4	Κίνηση στο Δίκτυο	78
4.7.5	Πλήθος Μηνυμάτων	79
4.7.6	Πειραματική Αξιολόγηση Κατανεμημένης Επεξεργασίας Δημοσιεύσεων	84
5	<i>PastryStrings</i>: Σύστημα Δημοσιεύσεων/Συνδρομών σε DHT Δίκτυα με Δρομολόγηση κατά Πρόθεμα	91
5.1	Η Αρχιτεκτονική της Υποδομής <i>PastryStrings</i>	91
5.1.1	Βασικά Χαρακτηριστικά των String Trees	93
5.1.2	Επεξεργασία Συνδρομών και Δημοσιεύσεων με Ιδιοχαρακτηριστικά Αριθμητικού Τύπου και Συμβολοσειρές	97
5.2	Εξισορρόπηση του Φόρτου στην Υποδομή <i>PastryStrings</i>	102
5.2.1	Δημιουργία Αντιγράφων Δέντρων	102
5.2.2	Διαμοιρασμός του Φόρτου Αποθήκευσης	103
5.2.3	Μετατόπιση του Πεδίου Ορισμού	103
5.3	Αυτόνομη Οργάνωση της Υποδομής <i>PastryStrings</i> και Αντιμετώπιση Βλαβών	104
5.3.1	Αυτόνομη Οργάνωση της Υποδομής <i>PastryStrings</i>	105
5.3.2	Αντιμετώπιση Βλαβών στην Υποδομή <i>PastryStrings</i>	108
5.4	Θεωρητική Ανάλυση Απόδοσης	114
5.5	Πειραματική Αξιολόγηση και Μελέτη Απόδοσης	115
5.5.1	Χαρακτηριστικά και Παράμετροι Πειραματικής Διάταξης	115
5.5.2	Η Επίδραση του Μεγέθους των Διαστημάτων στην Επεξεργασία των Συνδρομών	116
5.5.3	Εξισορρόπηση του Φόρτου Αποθήκευσης και Ανάκτησης των SubIDs	117
5.5.4	Επεξεργασία και Σύζευξη Δημοσιεύσεων με Συνδρομές	120
5.5.5	Αυτόνομη Οργάνωση και Αντιμετώπιση Βλαβών	122
6	Σύζευξη Δημοσιεύσεων και Συνδρομών με Τελεστές Υποσυμβολοσειράς	127
6.1	Ορισμός του Προβλήματος και των Σχετικών Αλγορίθμων	128
6.1.1	Η Προσέγγιση complete-IND	128
6.1.2	Η Προσέγγιση με Χρήση των <i>ngrams</i> πάνω σε Οποιοδήποτε DHT: <i>ngrams-IND</i>	129
6.1.3	Η Προσέγγιση complete-PS	131
6.1.4	Η Προσέγγιση με Χρήση των <i>ngrams</i> πάνω στην Υποδομή <i>PastryStrings</i> : <i>ngrams-PS</i>	133
6.2	Η Επέκταση των <i>ngams</i> Προσεγγίσεων με την Χρήση <i>Bloom</i> Φίλτρων	134
6.2.1	Τεχνική Μείωσης της Κίνησης στο Δίκτυο	138
6.2.2	Τεχνική Αύξησης της Πιθανότητας Σύζευξης	138
6.3	Θεωρητική Ανάλυση Απόδοσης	139
6.4	Πειραματική Αξιολόγηση και Μελέτη Απόδοσης	140

6.4.1	Χαρακτηριστικά και Παράμετροι Πειραματικής Διάταξης	140
6.4.2	Πλήθος Μηνυμάτων	141
6.4.3	Κίνηση στο Δίκτυο	143
6.4.4	Εξισορρόπηση του Φόρτου Πρόσβασης και Αποθήκευσης των SubIDs	143
6.4.5	Σχολιασμός των Χαρακτηριστικών και της Απόδοσης των Προτεινόμενων Προσεγγίσεων	145
7	Επίλογος	147
	Ευρετήριο	150
	Γλωσσάρι	153
	Βιβλιογραφία	159

Κατάλογος Πινάκων

4.1	Αλγόριθμος διάσπασης διαστημάτων τιμών.	48
4.2	Η διαδικασία αποθήκευση των αναγνωριστικών <i>SubIDs</i> των συνδρομών στις οποίες έχουν οριστεί περιορισμοί διαστημάτων αριθμητικών τιμών, με την προσέγγιση <i>virtual Range Tree Forest</i>	49
4.3	Η διαδικασία αποθήκευση των αναγνωριστικών <i>SubIDs</i> των συνδρομών στις οποίες έχουν οριστεί περιορισμοί διαστημάτων αριθμητικών τιμών, με την προσέγγιση <i>stateful Range Tree Forest</i>	52
4.4	Η διαδικασία επεξεργασίας των συνδρομών με ιδιοχαρακτηριστικά τύπου συμβολοσειράς. . .	53
4.5	Συλλογή των <i>SubIDs</i> των συνδρομών που πιθανόν να συζευγνύουν με την δημοσίευση. . . .	60
4.6	Σύζευξη της δημοσίευσης με τις συνδρομές και παράδοση της στους συνδρομητές.	61
4.7	Πολυπλοκότητα επεξεργασίας συνδρομών και δημοσιεύσεων με αριθμητικού τύπου ιδιοχαρακτηριστικά.	66
4.8	Πολυπλοκότητα επεξεργασίας συνδρομών και δημοσιεύσεων με ιδιοχαρακτηριστικά τύπου συμβολοσειράς.	68
4.9	Μέσο (και σταθμισμένο) πλήθος μηνυμάτων που απαιτούνται κατά την επεξεργασία των συνδρομών και των δημοσιεύσεων με ιδιοχαρακτηριστικά αριθμητικού τύπο και συμβολοσειράς. . .	83
5.1	Πολυπλοκότητα επεξεργασίας συνδρομών και δημοσιεύσεων στην υποδομή <i>PastryStrings</i> . . .	114
6.1	Πολυπλοκότητα επεξεργασία δημοσιεύσεων και συνδρομών με τελεστές υποσυμβολοσειράς. .	140
6.2	Χαρακτηριστικά και απόδοση των προτεινόμενων προσεγγίσεων για την επεξεργασία τελεστών υποσυμβολοσειρών.	145

Κατάλογος Σχημάτων

2.1 Αναζήτηση αντικειμένων στο δίκτυο Freenet.	9
2.2 Αναζήτηση αντικειμένων στο δίκτυο Gnutella.	10
2.3 Αναζήτηση αντικειμένων στο δίκτυο Kazaa.	11
2.4 Πίνακας δρομολόγησης στο Pastry, με το leaf set και το neighbor set. Παράδειγμα δρομολόγησης πάνω στο δίκτυο Pastry.	14
2.5 Το δακτυλίδι διευθύνσεων στο δίκτυο Chord αποτελούμενο από 10 κόμβους και 5 αποθηκευμένα αντικείμενα. Μία αίτηση αναζήτηση για το αντικείμενο K54 ξεκινάει από τον κόμβο N8	16
2.6 Δρομολόγηση αιτήσεων στο δισδιάστατο χώρο ($d = 2$) του δικτύου CAN και άφιξη νέου κόμβου. 17	
2.7 Δρομολόγηση δημοσίευσης $n1$ (με τιμή 65) στους συνδρομητές με συνδρομές $s1$ και $s2$ στο σύστημα SIENA.	24
2.8 Παράδειγμα της δενδρικής δομής σύζευξης δημοσιεύσεων με συνδρομές στο σύστημα Gryphon. 26	
2.9 Παράδειγμα ανάπτυξης του multicast δέντρου στο σύστημα Hermes.	28
3.1 Το μοντέλο δικτύου σε ένα σύστημα δημοσιεύσεων/συνδρομών. Οι χρήστες συνδέονται με τους brokers οι οποίοι σχηματίζουν ένα δίκτυο ομοτίμων εταίρων βασισμένο σε DHT.	34
4.1 Επεξεργασία συνδρομών με περιορισμό διαστήματος τιμών και επεξεργασία δημοσίευσης με τιμή που ανήκει στο διάστημα τιμών, με την προσέγγιση <i>OP-DHT</i>	42
4.2 Αποθήκευση διαστήματος τιμών με την προσέγγιση <i>OP-DHT</i>	44
4.3 Ένα δέντρο στο <i>Range Tree Forest</i> με βαθμό $g=2$ (ένα πλήρες δυαδικό δέντρο) και ύψος $H=4$ ικανό να διαχειριστεί διαστήματα στο πεδίο ορισμού $D = [0, 16)$	46
4.4 Υπολογισμός του κατάλληλου μονοπατιού προς τη ρίζα σε ένα δέντρο στο <i>Range Tree Forest</i> με βαθμό $g=2$ και ύψος $H=4$	51
4.5 Σχηματισμό μέρους του δέντρου κατά την επεξεργασία συνδρομών με την προσέγγιση <i>stateful Range Tree Forest</i>	54
4.6 Επεξεργασία συνδρομής και σύζευξη αυτής με σχετική δημοσίευση, με την προσέγγιση <i>OP-DHT</i> . 57	
4.7 Επεξεργασία συνδρομής και σύζευξη αυτής με σχετική δημοσίευση, με την προσέγγιση <i>virtual Range Tree Forest</i>	58
4.8 <i>Κεντριοποιημένη Σύζευξη</i> δημοσίευσης με συνδρομές.	70
4.9 <i>Κατανεμημένη Σύζευξη</i> δημοσίευσης με συνδρομές.	71
4.10 <i>Υβριδική Σύζευξη</i> δημοσίευσης με συνδρομές.	73
4.11 Η κατανομή δημοτικότητας Zipf για $\vartheta = 1.0$	74
4.12 Η καμπύλη Lorenz που απεικονίζει την κατανομή του φόρτου σε ένα δίκτυο ομοτίμων εταίρων, με συντελεστή $G = 0.5$	76
4.13 Εξισορρόπηση του φόρτου μεταβάλλοντας την κλήση της κατανομής της δημοτικότητας των τιμών και το πλήθος των αντιγράφων (replicas) <i>SubIDs</i>	78
4.14 Κίνηση στο δίκτυο κατά την επεξεργασία των δημοσιεύσεων.	79
4.15 Πλήθος των μηνυμάτων κατά την επεξεργασία των δημοσιεύσεων με αριθμητικού τύπου ιδιο-χαρακτηριστικά για διαφορετικούς συνδυασμούς g/H	80
4.16 Πλήθος των μηνυμάτων κατά την επεξεργασία των συνδρομών με αριθμητικού τύπου ιδιο-χαρακτηριστικά για διαφορετικούς συνδυασμούς g/H	81

4.17	Σταθμισμένη μέση τιμή του πλήθους των μηνυμάτων που απαιτούνται κατά την επεξεργασία των δημοσιεύσεων και των συνδρομών με λόγο δημοσιεύσεων/συνδρομών ίσο με 80/20, για αριθμητικού τύπου ιδιοχαρακτηριστικά.	82
4.18	Πλήθος των μηνυμάτων που απαιτούνται κατά την επεξεργασία των δημοσιεύσεων με ιδιοχαρακτηριστικά τύπου συμβολοσειράς μεταβάλλοντας το ύψος του δέντρου (μέγιστο μέγεθος συμβολοσειρών)	82
4.19	Σταθμισμένη μέση τιμή του πλήθους των μηνυμάτων που απαιτούνται κατά την επεξεργασία των δημοσιεύσεων και των συνδρομών (λόγος δημοσιεύσεων/συνδρομών: 80/20).	82
4.20	Πλήθος μεταφερόμενων <i>SubIDs</i> μεταβάλλοντας την κλίση της κατανομής της δημοτικότητας των τιμών των ιδιοχαρακτηριστικών.	86
4.21	Πλήθος μεταφερόμενων <i>SubIDs</i> μεταβάλλοντας την κλίση της κατανομής της δημοτικότητας των ιδιοχαρακτηριστικών. Η κατανομή των τιμών καθορίζεται από $\vartheta = 0.5$	87
4.22	Πλήθος μεταφερόμενων <i>SubIDs</i> μεταβάλλοντας την κλίση της κατανομής της δημοτικότητας των ιδιοχαρακτηριστικών. Η κατανομή των τιμών καθορίζεται από $\vartheta = 1.0$ για τα μη δημοφιλή ιδιοχαρακτηριστικά.	88
4.23	Χρόνος απόκρισης κατά την επεξεργασία δημοσιεύσεων με <i>Κεντρικοποιημένη, Κατανεμημένη, και Υβριδική</i> σύζευξη.	89
5.1	Ανάπτυξη ενός <i>string tree</i> με $\beta = 2$ και αναγνωριστικά μεγέθους 3 χαρακτήρων και αποθήκευση σε αυτό <i>SubIDs</i> συνδρομών με τιμές '010' και '00'	95
5.2	Σύζευξη δημοσίευσης με συνδρομές. Η δημοσίευση έχει τιμή '010' ενώ ήδη είναι αποθηκευμένες δύο συνδρομές με τιμές '00' και '010'.	95
5.3	Το δέντρο T_0 με αποθηκευμένα τα <i>SubIDs</i> για το διάστημα [2,5). Μία δημοσίευση καταφθάνει με τιμή 3 και ακολουθώντας το μονοπάτι μέχρι το φύλλο, εντοπίζει τα αποθηκευμένα <i>SubIDs</i>	99
5.4	Αποθήκευση συνδρομών με ιδιοχαρακτηριστικά τύπου συμβολοσειράς, με την τεχνική της δρομολόγησης με βάση το πρόθεμα.	101
5.5	Ο κόμβος F συνδέεται στην υποδομή <i>PastryStrings</i> στην οποία έχει αναπτυχθεί μέρος του δέντρου T_0 . Το T_0 οργανώνεται αυτόνομα με την άφιξη του F	106
5.6	Αναχώρηση του κόμβου B από την υποδομή <i>PastryStrings</i> . Το δέντρο T_0 αναδιοργανώνεται κατάλληλα.	108
5.7	Αποκατάσταση του δέντρου υπό την παρουσία συχνών βλαβών. Τα T_{node00} και $T_{node000}$ αποσυνδέονται. Το υποδέντρο με ρίζα τον $T_{node0001}$ επανασυνδέεται στο υπόλοιπο δέντρο	113
5.8	Η επίδραση των ιδιοχαρακτηριστικών αριθμητικού τύπου με περιορισμούς διαστήματος στο πλήθος των μηνυμάτων κατά την επεξεργασία των συνδρομών.	116
5.9	Η επίδραση του μεγέθους του διαστήματος τιμών στο πλήθος των παραγόμενων υποδιαστημάτων με την εφαρμογή του αλγόριθμου διάσπασης.	117
5.10	Εξισορρόπηση του φόρτου ανάκτησης και αποθήκευσης των <i>SubIDs</i> μεταβάλλοντας την κατανομή της δημοτικότητας των ιδιοχαρακτηριστικών.	118
5.11	Εξισορρόπηση του φόρτου ανάκτησης και αποθήκευσης των <i>SubIDs</i> μεταβάλλοντας την κατανομή δημοτικότητας των τιμών των ιδιοχαρακτηριστικών.	119
5.12	Εξισορρόπηση του φόρτου ανάκτησης και αποθήκευσης των <i>SubIDs</i>	120
5.13	Πλήθος μηνυμάτων που απαιτούνται κατά την επεξεργασία των συνδρομών και των δημοσιεύσεων.	121
5.14	Ποσοστό των ενεργών εγγραφών των <i>CRT</i> μεταβάλλοντας το β	122
5.15	Ποσοστό των ενεργών εγγραφών των <i>CRT</i> μεταβάλλοντας το μέγιστο μήκος των συμβολοσειρών.	123
5.16	Πλήθος χαμένων <i>SubIDs</i> συναρτήσει του ποσοστού των κόμβων που αποσυνδέονται από το δίκτυο λόγω βλάβης.	124
5.17	Επιρροή του ποσοστού αποτυχίας στην ακεραιότητα της δενδρικής δομής.	125
6.1	Ανάπτυξη των δέντρων κατά την επεξεργασία των συνδρομών και σύζευξη με τη δημοσίευση στη προσέγγιση complete-PS.	132
6.2	Σύζευξη δημοσίευσης με συνδρομές στην προσέγγιση ngrams-PS.	133
6.3	Παράδειγμα της προσέγγισης με ngrams και την χρήση των Bloom φίλτρων.	137

6.4	Πλήθος μηνυμάτων κατά την επεξεργασία των δημοσιεύσεων και των συνδρομών συναρτήσει της δημοτικότητας των ιδιοχαρακτηριστικών.	141
6.5	Κίνηση στο δίκτυο κατά την επεξεργασία των δημοσιεύσεων. Σύγκριση των προσεγγίσεων με ngrams με την επέκταση και χρήση των Bloom φίλτρων.	142
6.6	Κατανομή του φόρτου πρόσβασης και αποθήκευσης συναρτήσει της κατανομής της δημοτικότητας των τιμών των ιδιοχαρακτηριστικών.	144

Λίστα Συντμήσεων

Αγγλικοί Όροι

P2P	Peer-to-Peer
Pub/Sub	Publish/Subscribe information dissemination systems
DHT	Distributed Hash Table

Ελληνικοί Όροι

ΔΟΕ	Δίκτυο Ομοτίμων Εταίρων
Δημ./Συνδ.	Δημοσιεύσεις/Συνδρομές
ΚΠΚ	Κατανεμημένος Πίνακας Κατακερματισμού

Λίστα Συμβόλων

$nodeID$	Το αναγνωριστικό ενός οποιοδήποτε κόμβου του δικτύου ομοτίμων εταιρών
$SubID_i$	Το αναγνωριστικό της συνδρομής i
$xSubID_i$	Επέκταση του αναγνωριστικού $SubID_i$ της συνδρομής i ενσωματώνοντας επιπλέον πληροφορία
T_i	Το δέντρο με αναγνωριστικό i για τη διαχείριση συμβολοσειρών ή αριθμητικών τιμών
$Tnode_i$	Ο κόμβος δέντρου με ετικέτα (ή υποδιάστημα τιμών) i
CRT	Ο πίνακας δρομολόγησης που αφορά στην διατήρηση της συνεκτικότητας των δενδρικών δομών υπό την παρουσία σφαλμάτων
g	Ο βαθμός των δέντρων (πλήθος των δυνατών παιδιών που μπορεί να έχει κάθε $Tnode$)
β	Το πλήθος των διαθέσιμων χαρακτήρων των συμβολοσειρών στην προσέγγιση <i>PastryStrings</i> που καθορίζει τον βαθμό των δέντρων (πλήθος των δυνατών παιδιών που μπορεί να έχει κάθε $Tnode$)
d_i	Το βάθος του κόμβου i στο δέντρο. Η ρίζα του δέντρου βρίσκεται σε βάθος 0.
H	Το ύψος των δέντρων
a_i	Το ιδιοχαρακτηριστικό i από τα A διαθέσιμα ιδιοχαρακτηριστικά του μοντέλου δημοσιεύσεων/συνδρομών
$v(a_i)$	Η τιμή του ιδιοχαρακτηριστικού a_i
D_{a_i}	Το πεδίο ορισμού του ιδιοχαρακτηριστικού αριθμητικού τύπου, a_i
$pr(a_i)$	Η ακρίβεια του ιδιοχαρακτηριστικού αριθμητικού τύπου, a_i
D	Το μέγιστο δυνατό πεδίο ορισμού των αριθμητικών τιμών
$s[1...l_s]$	Συμβολοσειρά s αποτελούμενη από l_s χαρακτήρες
$s[i]$	Χαρακτήρας στη θέση i της συμβολοσειράς $s[1...l_s]$
$v_i = v[i...(n + i - 1)]$	ngram i της συμβολοσειράς v , αποτελούμενο από n χαρακτήρες
n	Το πλήθος των χαρακτήρων σε κάθε ngram
$prefix(string\ s, int\ j)$	Το πρόθεμα j χαρακτήρων της συμβολοσειράς s
$suffix(string\ s, int\ j)$	Το επίθεμα j χαρακτήρων της συμβολοσειράς s .
L_{ai-num}	Η λίστα αποθήκευσης στους κόμβους του δικτύου των $SubIDs$ των συνδρομών με τελεστή σε αριθμητική τιμή στο ιδιοχαρακτηριστικό a_i

L_{ai-eq}	Η λίστα αποθήκευσης στους κόμβους του δικτύου των $SubIDs$ των συνδρομών με τελεστή ισότητας σε συμβολοσειρά στο ιδιοχαρακτηριστικό a_i
$L_{ai-suff}$	Η λίστα αποθήκευσης στους κόμβους του δικτύου των $SubIDs$ των συνδρομών με τελεστή επιθέματος σε συμβολοσειρά στο ιδιοχαρακτηριστικό a_i
$L_{ai-pref}$	Η λίστα αποθήκευσης στους κόμβους του δικτύου των $SubIDs$ των συνδρομών με τελεστή προθέματος σε συμβολοσειρά στο ιδιοχαρακτηριστικό a_i
RS_i	Το σύνολο RS_i (Result Set), αποτελείται από τα $SubIDs$ που επιστρέφει ο κόμβος i εξαιτίας της σύζευξης τους με τη τιμή της εκάστοτε δημοσίευσης.
BF_i	Το <i>Bloom</i> φίλτρο που επιστρέφει ο κόμβος i
PS_i	Οι θέσεις στο <i>Bloom</i> φίλτρο BF_i του κόμβου i που θα χρησιμοποιηθούν ώστε να εντοπιστούν τα $SubIDs$ που ανήκουν στο σύνολο RS_i και πρέπει να επιστραφούν στον συντονιστή
BFM	Πίνακας του οποίου οι γραμμές είναι τα BF_i των κόμβων που εμπλέκονται στην επεξεργασία ενός ιδιοχαρακτηριστικού τύπου συμβολοσειράς
RF	Το πλήθος των αντιγράφων (<i>replicas</i>) $SubIDs$ που υπάρχουν στο δίκτυο
$DHT.hash()$	Η συνάρτηση κατακερματισμού του δικτύου ομοτίμων εταίρων
N	Το πλήθος των κόμβων στο δίκτυο ομοτίμων εταίρων

1 Εισαγωγή

Στην εποχή μας οι χρήστες των εφαρμογών διαδικτύου κατακλύζονται από μεγάλες ποσότητες πληροφορίας που παράγεται με πολύ υψηλούς ρυθμούς. Αν και μπορεί κάποιος να υποστηρίξει ότι στις μέρες μας οι χρήστες θα πρέπει να είναι ιδιαίτερα ικανοποιημένοι με την ποσότητα πληροφορίας που έχουν στη διάθεση του, στην πραγματικότητα, μόνο ένα πολύ μικρό μέρος αυτής είναι πραγματικά ενδιαφέρον για κάθε χρήστη. Επιπλέον, οι παραγωγοί της πληροφορίας επιθυμούν αυτή να φτάνει σε όσο το δυνατόν περισσότερους χρήστες που έχουν εκδηλώσει το ανάλογο ενδιαφέρον.

Ως εκ τούτου, σε αυτό το περιβάλλον, μπορεί να δει κανείς το σημαντικό θέμα της μετάδοσης της πληροφορίας από δύο διαφορετικές πλευρές: η πρώτη αναφέρεται στην αποφυγή του συνεχούς κατακλυσμού των χρηστών με τα τεράστια ποσά πληροφορίας και τον εξαναγκασμό των χρηστών να επεξεργάζονται και να φιλτράρουν άσχετη με τα ενδιαφέροντα τους πληροφορία. Η δεύτερη αναφέρεται στην εξασφάλιση ότι η παραγόμενη πληροφορία θα φθάσει σε όλους τους χρήστες που πραγματικά έχουν εκδηλώσει ενδιαφέρον. Το μοντέλο διακίνησης και παράδοσης της πληροφορίας που περιλαμβάνεται στα συστήματα δημοσιεύσεων/συνδρομών (publish/subscribe systems) από τη φύση του κατάλληλο ώστε να καλύψει αυτές τις ανάγκες.

Ειδικότερα με το μοντέλο δημοσιεύσεων/συνδρομών που βασίζεται στο περιεχόμενο (content-based publish/subscribe), οι καταναλωτές πληροφοριών έχουν τη δυνατότητα να δηλώσουν τα ενδιαφέροντα τους χρησιμοποιώντας τις συνδρομές, στις οποίες καθορίζουν τα ιδιοχαρακτηριστικά και τις τιμές τους για τις οποίες οι χρήστες ενδιαφέρονται. Ειδικότερα, οι συνδρομές των χρηστών αποτελούνται από ορισμένους περιορισμούς σε συγκεκριμένου τύπου ιδιοχαρακτηριστικά. Τα ιδιοχαρακτηριστικά είναι συνήθως αριθμητικού τύπου ή τύπου συμβολοσειράς. Οι περιορισμοί που δηλώνονται σε αυτά τα ιδιοχαρακτηριστικά μπορεί να είναι για παράδειγμα η ισότητα, η ανισότητα, και τα διαστήματα τιμών για τις αριθμητικά ιδιοχαρακτηριστικά καθώς επίσης και το πρόθεμα (prefix), το επίθεμα (suffix), η υποσυμβολοσειρά (substring), και η ισότητα για τα ιδιοχαρακτηριστικά τύπου συμβολοσειράς.

Έχοντα λοιπόν στη διάθεση τους τις συνδρομές σαν μέσο έκφρασης των ενδιαφερόντων τους, οι χρήστες θα πρέπει αυτόματα να λαμβάνουν μόνο πληροφορία σχετική με τις συνδρομές τους. Αυτή η πληροφορία γεννιέται και καταφθάνει στο σύστημα με την μορφή των δημοσιεύσεων, όπου η καθεμία φέρει τιμές για ένα υποσύνολο των διαθέσιμων ιδιοχαρακτηριστικών. Αναγκαία συνθήκη σε αυτά τα συστήματα ώστε

μία δημοσίευση να παραδοθεί σε ένα συνδρομητή, είναι η σύζευξη των τιμών της δημοσίευσης με τους περιορισμούς που έχουν διατυπωθεί στη συνδρομή.

Στις σύγχρονες εφαρμογές, συνήθως οι καταναλωτές και οι παραγωγοί της πληροφορίας είναι γεωγραφικά κατανεμημένοι, συνθέτοντας έτσι ένα δίκτυο πληροφοριών. Η σημαντικότερη πρόκληση κατά την εφαρμογή συστημάτων δημοσιεύσεων/συνδρομών βασισμένα στο περιεχόμενο πάνω σε τέτοια δίκτυα, είναι ανάπτυξη αποδοτικών αλγορίθμων για την επεξεργασία των συνδρομών και των δημοσιεύσεων. Η επεξεργασία των συνδρομών συνήθως συνίσταται στο να εντοπίσει κανείς τους κατάλληλους κόμβους του δικτύου και να αποθηκεύσει εκεί πληροφορία σχετική με τις συνδρομές. Στη συνέχεια και κατά την επεξεργασία των δημοσιεύσεων, θα πρέπει να εντοπιστούν εκείνη οι κόμβοι που έχουν αποθηκευμένη πληροφορία συνδρομών που συζευγνύουν με τις δημοσιεύσεις, ώστε αυτές να παραδοθούν στους συνδρομητές που έχουν δηλώσει σχετικό ενδιαφέρον.

Εντούτοις, το ίδιο το δίκτυο θέτει επιπρόσθετες σημαντικές προκλήσεις. Συγκεκριμένα, η ανοχή στα σφάλματα θα πρέπει να εξασφαλιστεί ενώ παράλληλα οι εφαρμογές που αξιοποιούν το δίκτυο (όπως για παράδειγμα ένα σύστημα δημοσιεύσεων συνδρομών) θα πρέπει να είναι θωρακισμένες απέναντι στα δυναμικά χαρακτηριστικά των δικτύων που σχετίζονται με βλάβες και συχνές αναχωρήσεις/αφίξεις νέων κόμβων. Επιπλέον, θα πρέπει να εξασφαλισθεί ότι η δρομολόγηση των δημοσιεύσεων και των συνδρομών μέσω του δικτύου θα εκτελείται γρήγορα. Η κλιμάκωση (scalability) επίσης αποτελεί σημαντικό ζήτημα σε τέτοια συστήματα, αφού δίκτυα αποτελούμενα από χιλιάδες κόμβους με εξίσου πολλούς χρήστες μπορεί να προκαλέσουν μια έκρηξη στον αριθμό των συνδρομών και των δημοσιεύσεων. Τέτοια μεγέθη θα πρέπει να μπορούν να αντιμετωπιστούν ομαλά από το δίκτυο, χωρίς σημεία συμφόρησης ή αποτυχίας εξαιτίας του αυξημένου φόρτου, με μία μικρή αύξηση του κόστους δρομολόγησης των αιτήσεων ακόμα και όταν ο αριθμός των κόμβων και των χρηστών αυξάνεται δραματικά.

Τα τελευταία χρόνια, ένα μεγάλο μέρος της σχετικής έρευνας έχει αντιμετωπίσει τα σχετικά με το δίκτυο προβλήματα, παράγοντας αρκετά επικαλυπτικά δίκτυα (overlay networks) που παρέχουν την κλιμάκωση (scalability), ανοχή στα σφάλματα (fault tolerance), και την αποδοτικότητα. Επιπλέον, αυτά τα δίκτυα διευκολύνουν την κατασκευή μεγάλης κλίμακας, δικτύων πληροφοριών που αποκαλούνται δίκτυα ομοτίμων εταίρων (P2P - peer-to-peer). Τα δίκτυα ομοτίμων εταίρων προσφέρουν ένα πλήρως κατανεμημένο περιβάλλον, μπορούν να κλιμακωθούν σε μεγάλο βαθμό με χιλιάδες κόμβους, ενώ μπορούν να οργανώνονται απουσία κεντρικού ελέγχου καθιστώντας τα πλήρως αυτόνομα. Όλοι οι συμμετέχοντες κόμβοι σε αυτά τα δίκτυα έχουν ίσες ευκαιρίες και παρέχουν υπηρεσίες όπου οι πληροφορία ανταλλάσσεται άμεσα μεταξύ τους. Μια δημοφιλής κατηγορία των δικτύων ομοτίμων εταίρων είναι αυτή των δομημένων δικτύων, όπου η τοποθέτηση των αντικειμένων και η τοπολογία του δικτύου ορίζονται ρητά ώστε να εξασφαλίζεται με μεγάλη

πιθανότητα δρομολόγηση και εντοπισμός των αντικειμένων με αριθμό βημάτων πάνω στην τοπολογία που είναι λογαριθμική συνάρτηση του αριθμού των κόμβων. Ανάμεσα στα δομημένα δίκτυα ομοτίμων εταιρών, ξεχωρίζουν αυτά που βασίζονται στους Κατανεμημένους Πίνακες Κατακερματισμού (DHT - Distributed Hash Table) όπως το Pastry [85], το Tapestry [104], το Bamboo [84], το CHORD [88], το CAN [82] κ.α.

Η κατεύθυνση μας προς την ανάπτυξη συστημάτων δημοσιεύσεων/συνδρομών ευρείας κλίμακας, μας οδηγεί στο να απομονώσουμε και να παρακάμψουμε τα προαναφερθέντα ζητήματα που αφορούν την υποδομή του δικτύου με το να υιοθέτηση ένα DHT δίκτυο ομοτίμων εταιρών. Ο πρώτος στόχος μας είναι η ανάπτυξη αλγορίθμων επεξεργασίας των συνδρομών και των δημοσιεύσεων σε ένα πλήρως κατανεμημένο περιβάλλον, που διαχειρίζεται το DHT δίκτυο σαν ένα «μαύρο» κουτί. Αυτό σημαίνει ότι θα πρέπει οι αλγόριθμοι που προτείνουμε να είναι σε θέση να λειτουργούν εκμεταλλευόμενοι μόνο τις διεπαφές και τη λειτουργικότητα που προσφέρει το DHT δίκτυο. Ένα βασικό πρόβλημα που παρουσιάζεται σε αυτή την περίπτωση, έχει να κάνει με το ότι τα δίκτυα DHT έχουν σχεδιαστεί έτσι ώστε να προσφέρουν υποστήριξη για ερωτήσεις ακριβούς αντιστοιχίας (δηλ. δεδομένου ενός κλειδιού να μπορέσουμε να εντοπίσουμε τον κόμβο που έχει αποθηκευμένα αντικείμενα με το ίδιο κλειδί). Ως εκ τούτου, οι αλγόριθμοι μας θα πρέπει να εκμεταλλευτούν αυτή τη λειτουργικότητα ώστε να μπορούν να εκτελεστούν και πιο σύνθετες ερωτήσεις τόσο σε αριθμητικού τύπου δεδομένα όσο και σε συμβολοσειρές πράγμα που είναι σύνθετες σε συστήματα δημοσιεύσεων/συνδρομών βασισμένα στο περιεχόμενο.

Ένα μεγάλο μέρος της πρόσφατης και τρέχουσας έρευνας στοχεύει στην επέκταση της λειτουργικότητας των δικτύων ομοτίμων εταιρών είτε αυτά είναι μη δομημένα, είτε δομημένα βασισμένα σε DHTs ([39, 49, 52–54, 56, 60, 69, 75, 95, 101, 102]). Η σχετική έρευνα έχει παράσχει λύσεις για έναν μεγάλο αριθμό προβλημάτων που σχετίζονται με την αναζήτηση της πληροφορίας, την επεξεργασία σύνθετων ερωτημάτων σε αριθμητικού τύπου δεδομένα με περιορισμούς διαστημάτων τιμών (range queries), την επεξεργασία σύνθετων ερωτημάτων σε δεδομένα τύπου συμβολοσειράς, την εφαρμογή της λειτουργικότητας των βάσεων δεδομένων σε κατανεμημένο περιβάλλον κ.α. Αυτό το γεγονός πιστοποιεί τη σημασία που δίνει η ερευνητική κοινότητά ώστε να είσαι σε θέση να υποστηριχθούν εφαρμογές διαχείρισης της πληροφορίας πάνω σε μεγάλης κλίμακας κατανεμημένα περιβάλλοντα.

Η υποστήριξη σύνθετων ερωτημάτων σε ιδιοχαρακτηριστικά τύπου συμβολοσειρά και αριθμητικού τύπου που περιλαμβάνουν τελεστές/περιορισμούς προθέματος (prefix), επιθέματος (suffix), υποσυμβολοσειράς (substring), ισότητας και διαστημάτων τιμών (ranges), μπορεί να είναι πολύ χρήσιμη σε ένα πλήθος εφαρμογών διαχείρισης της πληροφορίας, που στηρίζονται στο μοντέλο δημοσιεύσεων/συνδρομών. Μερικές αντιπροσωπευτικές εφαρμογές που ανήκουν σε αυτή την οικογένεια είναι:

(i) *Information Feeds*. Σε αυτές τις εφαρμογές οι εκδότες μπορούν να είναι, παραδείγματος χάριν,

διεθνή πρακτορεία ειδήσεων (που σχηματίζουν ένα γεωγραφικά μεγάλο δίκτυο) που δημοσιεύουν άρθρα. Οι συνδρομητές δηλώνουν τα ενδιαφέροντά τους με κατάλληλες συνδρομές βασισμένες στο περιεχόμενο που περιλαμβάνουν ιδιοχαρακτηριστικά με αριθμητικού τύπου περιορισμούς (π.χ. εύρος ημερομηνιών εμφάνισης των ειδησεογραφικών άρθρων) και ιδιοχαρακτηριστικά με περιορισμούς πάνω σε συμβολοσειρές όπως λέξεις που αποτελούν πρόθεμα των τίτλων των άρθρων. Γενικά, σε αυτήν την κατηγορία εμπίπτουν συστήματα διάδοσης πληροφοριών στα πρότυπα των RSS feeds (π.χ. δημοσίευση τιμών μετοχών χρηματιστηρίων).

(ii) *Grid Computing*. Φανταστείτε ένα δίκτυο αποτελούμενο από συστάδες υπολογιστικών μηχανών (computing clusters) με ετερογενή χαρακτηριστικά (όσον αφορά τους τύπους των λειτουργικών συστημάτων, των ικανοτήτων του υλικού των υπολογιστών, των εκδόσεων του λογισμικού, κ.λπ.). Οι δημοσιεύσεις διοχετεύονται στο σύστημα και αποτελούν μια περιγραφή/περίληψη των χαρακτηριστικών του κόμβου που είναι διατεθειμένος να προσφέρει υπολογιστική ισχύ στο σύνολο. Οι χρήστες που επιθυμούν να εκτελέσουν τα προγράμματά τους και ενδιαφέρονται για τις συγκεκριμένα χαρακτηριστικά υπολογιστικών μηχανών, κοινοποιούν τις προθέσεις τους με την έκδοση των συνδρομών τους που ορίζουν για παράδειγμα τις αποδεκτές ταχύτητες CPU, τις ελάχιστες απαιτήσεις σε μνήμη, το επιθυμητό λειτουργικό σύστημα κ.α.

(iv) *Bio-Informatics Applications*. Το δίκτυο σε αυτές τις εφαρμογές μπορεί να αποτελείται από κόμβους διαχείρισης των βάσεων δεδομένων ερευνητικών αποτελεσμάτων στο χώρο της βιολογίας, που μπορούν να ανήκουν σε συγκεκριμένα κυβερνητικά ή ιδιωτικά ερευνητικά ινστιτούτα. Οι δημοσιεύσεις προέρχονται σε αυτήν την περίπτωση από τους ερευνητές που εκθέτουν τα συγκεκριμένα συμπεράσματα τους στις πρωτεϊνικές ακολουθίες ενώ από την άλλη πλευρά βρίσκονται οι ερευνητές, δημοσιεύοντας τις συνδρομές τους με ερωτήματα πάνω σε συγκεκριμένα προθέματα, επιθέματα, υποσυμβολοσειρές συγκεκριμένων ακολουθιών χαρακτήρων (πρωτεϊνικές ακολουθίες).

1.1 Συνεισφορά στη Περιοχή

Κατά την άποψή μας, είναι ιδιαίτερα σημαντική η ύπαρξη μιας περιεκτικής αρχιτεκτονικής που βασίζεται στο μοντέλο δημοσιεύσεων συνδρομών, εύκολα εφαρμόσιμης πάνω από οποιοδήποτε DHT δίκτυο ομοτίμων εταίρων, ικανή να υποστηρίξει τόσο ιδιοχαρακτηριστικά αριθμητικού τύπου όσο και ιδιοχαρακτηριστικά τύπου συμβολοσειράς. Επιπλέον, θα πρέπει να είναι ικανή να μπορεί να υποστηρίξει αποδοτικά την κατανεμημένη επεξεργασία και σύζευξη των δημοσιεύσεων με τις συνδρομές, με τον αριθμό βημάτων πάνω στο δίκτυο (hops) να είναι λογαριθμική συνάρτηση του αριθμού των κόμβων που το αποτελούν.

Δεδομένου ότι οι πιθανές εφαρμογές που θα λειτουργήσουν με την βοήθεια αυτής της αρχιτεκτονικής

θα βρίσκονται σε ένα ιδιαίτερα δυναμικό περιβάλλον (όπως είναι αυτό των δικτύων ομοτίμων εταιρών), θα πρέπει να εξασφαλίσουμε ότι η υποδομή που θα αναπτύξουμε θα προσαρμόζεται στα χαρακτηριστικά του δικτύου, θα λειτουργεί σε ένα πλήρως κατανεμημένο περιβάλλον με ικανότητα αυτόνομης οργάνωσης, θα προσαρμόζεται στην κλιμάκωση που χαρακτηρίζει αυτά τα δίκτυα ενώ παράλληλα θα είναι ανεκτική στα σφάλματα που μπορεί να προκύψουν.

Με την εργασία που παρουσιάζουμε σε αυτή τη διατριβή, προσφέρουμε ένα σύνολο από λύσεις στο πρόβλημα της εφαρμογής του μοντέλου δημοσιεύσεων/συνδρομών πάνω σε δομημένα δίκτυα ομοτίμων εταιρών που βασίζονται σε DHTs. Η ανάπτυξη μιας ενοποιημένης αρχιτεκτονικής, ικανής να υποστηρίξει αποδοτικά σύνθετους τελεστές σε ιδιοχαρακτηριστικά αριθμητικού τύπου και συμβολοσειρών, αποτελεί πρόκληση δεδομένης της λειτουργικότητας που προσφέρουν τα DHT δίκτυα και ιδιαίτερα του έμφυτου περιορισμού τους να υποστηρίζουν μόνο τελεστές ισότητας (δηλ. δεδομένου ενός αναγνωριστικού ενός αντικειμένου να μπορούμε να εντοπίσουμε τον κόμβο που έχει αποθηκευμένα αντικείμενα με το ίδιο αναγνωριστικό).

Επεκτείνουμε λοιπόν και παρουσιάζουμε εδώ το μοντέλο δημοσιεύσεων/συνδρομών βασισμένο στο περιεχόμενο (content-based publish/subscribe) πάνω στα δίκτυα ομοτίμων εταιρών ώστε να μπορέσουμε να προσφέρουμε στους χρήστες υψηλή εκφραστικότητα κατά την δήλωση των ενδιαφερόντων τους, λειτουργώντας σε ένα πλήρως κατανεμημένο και κλιμακώσιμο περιβάλλον. Ο κορμός των προτεινόμενων λύσεων σε αυτή τη διατριβή είναι: (i) η ανάπτυξη αλγορίθμων για την αποθήκευση των αναγνωριστικών των δημοσιεύσεων σε κατάλληλους κόμβους του δικτύου με βάση τις συνθήκες στο περιεχόμενο που έχουν δηλωθεί και (ii) η ανάπτυξη αλγορίθμων δρομολόγησης των δημοσιεύσεων στο δίκτυο έτσι ώστε να «συναντούν» αυτούς τους κόμβους που περιέχουν συνδρομές που ικανοποιούνται από την πληροφορία της δημοσίευσης.

Αρχικά αναπτύξαμε αποδοτικές τεχνικές και σχετικούς αλγορίθμους που επιτυγχάνουν κατανεμημένη σύζευξη των συνδρομών με τις δημοσιεύσεις και παράδοση αυτών στους τελικούς χρήστες πάνω από οποιοδήποτε δομημένο δίκτυο ομοτίμων εταιρών τύπου DHT. Στη συνέχεια επωφεληθήκαμε από τα συγκεκριμένα χαρακτηριστικά ορισμένων δικτύων ομοτίμων (και συγκεκριμένα αυτών που εκτελούν δρομολόγηση με βάση το πρόθεμα των κλειδιών, όπως τα δίκτυα Pastry [85], Tapestry [103], Bamboo [84]) ώστε να βελτιώσουμε ακόμα περισσότερο την απόδοση. Στα πλαίσια αυτής της εργασίας αναπτύξαμε λύσεις που επιτρέπουν την επεξεργασία συνδρομών/δημοσιεύσεων με περιορισμούς/τιμές τόσο σε αριθμητικούς τύπους δεδομένων όσο και σε συμβολοσειρές. Ειδικότερα, οι προτεινόμενες λύσεις επιτρέπουν στους χρήστες την εφαρμογή τελεστών ισότητας, $\leq$, $\geq$, και διαστημάτων τιμών σε αριθμητικούς τύπους και επίθεμα (suffix), πρόθεμα (prefix), ισότητα και υποσυμβολοσειρά (substring) σε δεδομένα τύπου συμβολοσειρές. Παράλληλα, δεδομένου του ότι ένα δίκτυο ομοτίμων αποτελεί ένα ιδιαίτερα δυναμικό περιβάλλον με συχνές αναχωρήσεις/αφίξεις κόμβων, προτείνουμε τεχνικές που διατηρούν την λειτουργικότητα και την συνεκτι-

κότητα των δομών μας, ακόμα και υπό την παρουσία συχνών αναχωρήσεων/αφίξεων κόμβων, αλλά και σφαλμάτων/αποτυχιών στο δίκτυο.

Οι προτεινόμενοι αλγόριθμοι υλοποιήθηκαν και εξετάστηκαν ενδελεχώς με προσομοίωση μελετώντας την απόδοσή τους με βάση μετρικές όπως: η ισοκατανομή του φόρτου στους κόμβους του δικτύου από τη διακίνηση μηνυμάτων κατά την επεξεργασία των συνδρομών/δημοσιεύσεων, ο συνολικός αριθμός μηνυμάτων που διακινούνται, ο συνολικός όγκος επιπλέον πληροφορίας που απαιτούν οι αλγόριθμοι να διατηρείται στο δίκτυο, και ο χρόνος που απαιτείται για την ανεύρεση των συνδρομών που συζευγνύουν με κάθε δημοσίευση.

1.2 Δομή της Διατριβής

Η διατριβή είναι οργανωμένη ως εξής. Στο κεφάλαιο 2 παρουσιάζεται η σχετική έρευνα των τελευταίων χρόνων στην περιοχή των συστημάτων δημοσιεύσεων/συνδρομών και των δικτύων ομοτίμων εταίρων, όπου περιγράφονται και ταξινομούνται οι διάφοροι τύποι και τα ιδιαίτερα χαρακτηριστικά τους. Στο κεφάλαιο 3 παρουσιάζεται το βασικό μοντέλο δημοσιεύσεων/συνδρομών που ακολουθούμε στη διατριβή, τους τρόπους έκφρασης και σύνταξης των δημοσιεύσεων και των συνδρομών και τέλος το μοντέλο διασύνδεσης των χρηστών και τα βασικά λειτουργικά χαρακτηριστικά της υποδομής που προτείνουμε.

Στα κεφάλαια 4 και 5 παρουσιάζουμε τις λύσεις που προτείνουμε για την ανάπτυξη ενός πλήρως λειτουργικού συστήματος δημοσιεύσεων/συνδρομών πάνω από οποιοδήποτε DHT δίκτυο και πάνω από το δίκτυο Patsry αντίστοιχα. Ακολουθώντας, στο κεφάλαιο 6 επεκτείνουμε το μοντέλο που παρουσιάσαμε στα προηγούμενα κεφάλαια ώστε να μπορεί να υποστηρίξει τον τελεστή υποσυμβολοσειράς σε δεδομένα τύπου συμβολοσειράς. Τέλος, το κεφάλαιο 7 παρουσιάζει τα συμπεράσματα της όλης μελέτης.

2 Ανασκόπηση Σχετικής Έρευνας

2.1 Συστήματα Δικτύων Ομοτίμων Εταίρων

Τα Δίκτυα Ομοτίμων Εταίρων (Δ.Ο.Ε ή Peer-to-Peer, P2P) έχουν γίνει ιδιαίτερα δημοφιλή τα τελευταία χρόνια. Στις αρχές του αιώνα εμφανίστηκαν τα δίκτυα Napster [2], Gnutella, Kazaa [3] και Freenet [38] τα οποία έγιναν ιδιαίτερα δημοφιλή μιας και πρωταρχικός τους στόχος ήταν ο διαμοιρασμός και διακίνηση αρχείων, επιτρέποντας σε χιλιάδες χρήστες σε όλο τον πλανήτη να ανταλλάζουν πληροφορία.

Γενικά, τα δίκτυα ομοτίμων εταίρων, αποτελούν ένα υπόστρωμα (όπου κυριαρχεί η έννοια των λογικών συνδέσεων μεταξύ των κόμβων, με μία σύνδεση μεταξύ δύο κόμβων να αποτελεί στη πραγματικότητα ένα μονοπάτι στο tcp/ip δίκτυο) πάνω στο οποίο μπορεί κανείς να αναπτύξει αποδοτικά, κατανεμημένα συστήματα και εφαρμογές μεγάλης κλίμακας, σε ένα πλήρως κατανεμημένο περιβάλλον ικανό να οργανώνεται αυτόνομα. Στα δίκτυα ομοτίμων εταίρων κυριαρχούν οι έννοιες της κοινοκτημοσύνης των διαθέσιμων πόρων, της συνεργατικότητας και της παντελούς απουσίας κεντρικού ελέγχου. Η κλιμάκωση, η αξιοπιστία, η επεκτασιμότητα και η διαθεσιμότητα των δικτύων ομοτίμων εταίρων τα καθιστούν ιδιαίτερα δημοφιλή για την ανάπτυξη εφαρμογών που ακριβώς αναζητούν αυτές τις ιδιότητες.

Τα δίκτυα ομοτίμων εταίρων κατατάσσονται σε δύο μεγάλες κατηγορίες με βάση τον τρόπο που οργανώνονται οι κόμβοι σε αυτά τα δίκτυα :

- Στα *Μη Δομημένα Δ.Ο.Ε* δεν υπάρχει κανένας περιορισμός στον τρόπο που σχηματίζεται η τοπολογία του δικτύου. Στα πρώτα δίκτυα που αναπτύχθηκαν και ανήκουν σε αυτή την κατηγορία υπήρχε ένα είδος κεντρικού ελέγχου, όμως η πλειονότητα των κόμβων σχημάτιζαν ένα δίκτυο ομοτίμων εταίρων με τυχαία τοπολογία.
- Στα *Δομημένα Δ.Ο.Ε*, η τοπολογία του δικτύου και κατ' επέκταση η τοποθέτηση της πληροφορίας ελέγχεται συνήθως με κάποιου είδους κατανεμημένη δομή δεδομένων, όπως για παράδειγμα ο Κατανεμημένος Πίνακας Κατανεμημένος. Και εδώ οι κόμβοι είναι απόλυτα ίσοι μεταξύ τους, ενώ σχηματίζουν συγκεκριμένες ιδεατές διατάξεις (π.χ. δακτυλίδια, υπερκύβους) με βάση τα αναγνωριστικά των κόμβων.

2.2 Μη Δομημένα Δίκτυα Ομοτίμων Εταίρων

Σε αυτή την κατηγορία οι κόμβοι οργανώνονται σε τυχαίους γράφους, με επίπεδες τοπολογίες (flat topologies) με όλους τους κόμβους να είναι ίσοι μεταξύ τους, ή με ιεραρχικές τοπολογίες (Hierarchical topologies) όπου ένα μικρό σύνολο κόμβων (υπερκόμβοι, super-peers) χαρακτηρίζεται από αυξημένες αρμοδιότητες και δυνατότητες.

Σε αυτά τα δίκτυα δεν υπάρχει στενή σχέση μεταξύ της τοπολογίας και των σημείων (κόμβων) όπου αποθηκεύεται η πληροφορία. Η αναζήτηση και ο εντοπισμός των αντικειμένων επιτυγχάνεται κυρίως με τεχνικές flooding και random walks. Έτσι, οι αιτήσεις ανάκτησης αντικειμένων πλημμυρίζουν το δίκτυο, με κάθε κόμβο να εξυπηρετεί την αίτηση με βάση την πληροφορία που έχει αποθηκευμένη και στην συνέχεια να την προωθηθεί σε επόμενους κόμβους. Η συγκεκριμένη τεχνική δεν θεωρείται ιδιαίτερα αποδοτική, μιας και αιτήσεις αναζήτησης πληροφορίας που είναι ελάχιστα κατανεμημένη στο δίκτυο, θα πρέπει να ταξιδέψουν σε όλο το δίκτυο μέχρι να καταφέρουν να την εντοπίσουν.

2.2.1 Το Δίκτυο Freenet

Το δίκτυο Freenet [38] αναπτύχθηκε με σκοπό την αποθήκευση και τον διαμοιρασμό αρχείων. Κάθε αρχείο διαχειρίζεται από το δίκτυο χρησιμοποιώντας ένα κλειδί που κατασκευάζεται με βάση το περιεχόμενο του αρχείου και πληροφορία που δίνει ο ίδιος ο χρήστης. Το Freenet αποτελεί χαρακτηριστικό παράδειγμα αδόμητου δικτύου ομοτίμων εταίρων, όπου τα αντικείμενα (αρχεία σε αυτή την περίπτωση) και οι κόμβοι στους οποίους τελικά θα αποθηκευθούν δεν καθορίζονται με κάποιο γενικό κανόνα.

Το δίκτυο Freenet προσφέρει κλιμάκωση, ανοχή σε λάθη, ενώ παράλληλα επικεντρώνεται σε θέματα που σχετίζονται με τη βιωσιμότητα των αντικειμένων στο δίκτυο καθώς και την ανωνυμία τόσο των χρηστών που αρχικά προσέφεραν κάποιο αρχείο όσο και αυτών που το κατέχουν και το διαμοιράζουν.

Τα αρχεία στο Freenet χαρακτηρίζονται από το αναγνωριστικό τους που προκύπτει με τη χρήση της SHA-1 συνάρτησης κατακερματισμού. Κατά την αναζήτηση αρχείων (όπως μπορείτε να δείτε στο Σχήμα 2.1, όπου ο κόμβος A στέλνει την αίτηση αναζήτησης στον κόμβο B), ο χρήστης στέλνει στο δίκτυο μία αίτηση με το αναγνωριστικό του αρχείου. Όταν ένας κόμβος λάβει μια τέτοια αίτηση, ελέγχει τον τοπικό αποθηκευτικό του χώρο για την ύπαρξη του αρχείου και αν αυτό βρίσκεται εκεί το επιστρέφει στον χρήστη που το ζήτησε, ακολουθώντας το αντίστροφο μονοπάτι που ακολούθησε η αίτηση αναζήτησης. Διαφορετικά, προωθεί την αίτηση σε έναν από τους κόμβους που βρίσκονται στον τον πίνακα δρομολόγησης του. Ο πίνακας δρομολόγησης αποτελείται από τις διευθύνσεις των γειτονικών κόμβων ενώ παράλληλα κρατείται και πληροφορία σχετικά με τα αρχεία που έχουν αποθηκευμένα. Η επιλογή του επόμενου κόμβου που θα

Σχήμα 2.1: Αναζήτηση αντικειμένων στο δίκτυο Freenet.

σταλεί η αίτηση βασίζεται στο αναγνωριστικό του αρχείου και τα αναγνωριστικά των κόμβων στον πίνακα δρομολόγησης, επιλέγοντας αυτόν που ταιριάζει περισσότερο. Όταν τελικά το αρχείο εντοπιστεί, και κατά την αποστολή του, όλοι οι ενδιαμέσοι κόμβοι στο μονοπάτι που ακολουθεί το αρχείο προς τον τελικό κόμβο, αποθηκεύουν στον πίνακα δρομολόγησης τους πληροφορίες σχετικά με το συγκεκριμένο αρχείο ώστε να διευκολυνθούν μελλοντικές αναζητήσεις.

Κάθε αίτηση στο Freenet συνοδεύεται από ένα αναγνωριστικό ώστε να αποφευχθούν οι κυκλικές διαδρομές στο δίκτυο. Επίσης, συνοδεύεται και από έναν μετρητή TTL (Time To Live) που μειώνεται σε κάθε επιτυχημένη προώθηση της αίτησης, ώστε τελικά να κρατηθεί η κίνηση στο δίκτυο (που αφορά την αναζήτηση) σε χαμηλά επίπεδα.

Η εισαγωγή νέων αρχείων στο σύστημα γίνεται με τη δημοσίευση του αναγνωριστικού του, στους υπόλοιπους ακολουθώντας παρόμοια τεχνική με αυτή της αναζήτησης αρχείων, όπου κάθε κόμβος ενημερώνει τον πίνακα δρομολόγησης του με την προέλευση του αρχείου.

Η ανωνυμία των χρηστών επιτυγχάνεται με το να θεωρείται ο κάθε κόμβος που έχει πληροφορία για ένα αρχείο, σαν ο κόμβος που κατέχει το αρχείο. Έτσι είναι αδύνατο να γνωρίζει κανείς ποιος πραγματικά είναι ο κάτοχος (που πρώτος εισήγαγε το αρχείο στο δίκτυο). Επίσης, κάθε κόμβος που λαμβάνει μια αίτηση αναζήτησης δεν γνωρίζει αν ο κόμβος που του έστειλε την αίτηση είναι αυτός που ενδιαφέρεται για το αρχείο, ή κάποιος ενδιαμέσος στο μονοπάτι αναζήτησης.

2.2.2 Το Δίκτυο Gnutella

Το Gnutella είναι ένα κατακευματισμένο πρωτόκολλο αναζήτησης και εντοπισμού αντικειμένων σε μία επίπεδη τοπολογία. Κάθε κόμβος στο δίκτυο που εκμεταλλεύεται το πρωτόκολλο Gnutella μπορεί να είναι είτε απλός χρήστης είτε εξυπηρετητής την ίδια στιγμή. Χαρακτηριστικό τους είναι ότι δεν υπάρχει κεντρικός

Σχήμα 2.2: Αναζήτηση αντικειμένων στο δίκτυο Gnutella.

έλεγχος ούτε αναφορικά με την τοπολογία που σχηματίζουν οι κόμβοι αλλά ούτε και σχετικά με το που αποθηκεύονται τα αντικείμενα. Οι κόμβοι σχηματίζουν το δίκτυο ακολουθώντας ορισμένους χαλαρούς κανόνες και αφού πρώτα γνωρίζουν για την ύπαρξη ενός τουλάχιστον κόμβου μέλους του δικτύου (συνήθως μέσω ενός σχετικού με το δίκτυο ιστοτόπου). Το πρωτόκολλο Gnutella έχει χρησιμοποιηθεί ευρέως και για αυτό έχουν παρουσιαστεί κατά καιρούς σχετικές εργασίες ([35, 48, 66])

Για τον εντοπισμό ενός αντικειμένου, ο κόμβος στον οποίο έφτασε η αίτηση αναζήτησης την προωθεί σε όλους τους γείτονες, ακολουθώντας την τεχνική του flooding όπως μπορείτε να δείτε στο Σχήμα 2.2. Η αίτηση συνεχίζει να προωθείται σε όλους τους κόμβους που βρίσκονται σε ακτίνα R γύρω από τον αρχικό κόμβο. Η τεχνική αυτή θεωρείται ιδιαίτερα ανεκτική στην συχνή άφιξη και αναχώρηση των κόμβων (κάτι που χαρακτηρίζει τα δίκτυα ομοτίμων εταίρων). Ωστόσο, δεν θεωρείται ιδιαίτερα αποδοτική όσον αφορά τον αριθμό των μηνυμάτων που παράγονται σε κάθε αναζήτηση, αυξάνοντας τον φόρτο πρόσβασης των κόμβων, ενώ παράλληλα όταν ο αριθμός των κόμβων αυξάνεται τότε ο αριθμός μηνυμάτων αυξάνεται δραματικά.

Κατά τον εντοπισμό σχετικών με την αναζήτηση αντικειμένων, αυτά ακολουθούν το αντίστροφο μονοπάτι προς τον κόμβο που εκτέλεσε την αναζήτηση. Σε αντίθεση με το Freenet, το Gnutella δεν προσφέρει κανένα είδος ανωνυμίας μιας και τα μηνύματα στο δίκτυο περιέχουν τις IP διευθύνσεις των κόμβων που έχουν κάποιο αντικείμενο, οπότε κάποιος χρήστης μπορεί να γνωρίζει ποιος κατέχει τι. Τελευταίες εκδόσεις του Gnutella περιλαμβάνουν την έννοια των υπερκόμβων (super-peers) που διευκολύνουν στην δρομολόγηση των αιτήσεων στο δίκτυο. Παρά τα αρνητικά στοιχεία του Gnutella που σχετίζονται κυρίως με το θέμα της κλιμάκωσης και του τρόπου που εκτελείται η αναζήτηση, έχουν κατά καιρούς προταθεί εργασίες ([35, 64]) που μπορούν να βελτιώσουν την απόδοση του δραματικά.

Σχήμα 2.3: Αναζήτηση αντικειμένων στο δίκτυο Kazaa.

2.2.3 Άλλα Μη δομημένα Δίκτυα Ομοτίμων Εταίρων

Napster

Το δίκτυο Napster ([2]) αποτέλεσε ένα από τα πρώτα δίκτυα ομοτίμων εταίρων. Δεν μπορεί να θεωρηθεί ότι αποτελεί ένα πλήρως καταναμημένο δίκτυο, μιας και υπάρχει ένα σύνολο από εξυπηρετητές (στους οποίους συνδέονται οι χρήστες) που διατηρούν πληροφορία σχετικά με το ποια αντικείμενα κατέχουν. Η αναζήτηση λοιπόν σε αυτή την περίπτωση προωθείται σε αυτούς τους εξυπηρετητές που αναλαμβάνουν να επιστρέψουν πίσω τις διευθύνσεις των χρηστών που διατηρούν τα αντικείμενα που ζητήθηκαν. Οι χρήστες σχηματίζουν ένα δίκτυο ομοτίμων εταίρων που στη συνέχεια αξιοποιείται ώστε να παραδοθεί το αντικείμενο στον τελικό χρήστη.

Δεδομένης της ύπαρξης κεντρικής υπηρεσίας, το Napster δεν μπορεί να αντεπεξέλθει στην αύξηση του αριθμού των χρηστών μιας και όλες οι αιτήσεις αναζήτησης θα πρέπει να αποσταλούν στους κεντρικούς εξυπηρετητές.

Kazaa

Το Kazaa ([3]) βασίζεται στο δίκτυο Fasttrack που πρωτοεμφανίστηκε το 2001. Αποτελεί κλασσικό παράδειγμα ιεραρχικής αρχιτεκτονικής. Στα *Ιεραρχικά* δίκτυα ομοτίμων εταίρων υπάρχει η έννοια της ιεραρχίας με ορισμένους κόμβους να έχουν αυξημένες αρμοδιότητες αλλά και πόρους (αυξημένο εύρος ζώνης, αποθηκευτικό χώρο, υπολογιστική ισχύ) και με την πλειονότητα των χρηστών να είναι συνδεδεμένοι με αυτούς και να διαμοιράζονται πληροφορία (Σχήμα 2.3). Οι υπερκόμβοι (super-peers) οργανώνονται σε ένα δίκτυο ομοτίμων εταίρων, όπου ο κάθε υπερκόμβος εξυπηρετεί ένα σύνολο των χρηστών που είναι συνδεδεμένοι με το δίκτυο, και χρησιμοποιούνται για να διευκολύνουν την αναζήτηση αντικειμένων.

Οι απλοί χρήστες μεταφέρουν πληροφορία σχετικά με τα αντικείμενα που κατέχουν στους υπερκόμβους

(στα πρότυπα λειτουργίας του Napster). Όλες οι αιτήσεις αναζήτησης, επίσης μεταφέρονται στους υπερ-κόμβους, οι οποίοι εκτελούν αναζήτηση στα πρότυπα του Gnutella στο δίκτυο που σχηματίζουν μεταξύ τους, ώστε να εντοπίσουν το σύνολο των χρηστών που κατέχουν ένα συγκεκριμένο αντικείμενο. Αυτή η πληροφορία επιστρέφεται στον χρήστη που εκτέλεσε την αναζήτηση που στη συνέχεια συνδέεται με τους υπόλοιπους χρήστες, ώστε τελικά να ανακτηθεί το αντικείμενο.

BitTorrent

Τα δίκτυα που βασίζονται στο πρωτόκολλο BitTorrent ([4]) σχηματίζουν και αυτά ένα ιεραρχικού τύπου δίκτυο ομοτίμων εταίρων, με το tracker (ένα ιστότοπο που περιέχει αρχεία - torrents - με μεταπληροφορία) να διατηρεί πληροφορία σχετικά με τα διαμοιραζόμενα αρχεία και τους χρήστες που τα κατέχουν. Οι χρήστες σχηματίζουν ένα δίκτυο ομοτίμων εταίρων, και αφού συνδεθούν με τον tracker και πάρουν πληροφορία σχετικά με το αντικείμενο που αναζητούν (το torrent αρχείο που αντιστοιχεί στο αντικείμενο που αναζητούν), ξεκινάνε τον διαμοιρασμό του με το να δέχονται αλλά και να προσφέρουν κομμάτια του αρχείου.

Το πρωτόκολλο BitTorrent είναι εφοδιασμένο με ορισμένα ενδιαφέροντα χαρακτηριστικά που το έχουν κάνει ιδιαίτερα δημοφιλές τα τελευταία χρόνια. Το πρωτόκολλο είναι έτσι σχεδιασμένο ώστε να αποθαρρύνει τους free-riders δηλαδή τους χρήστες που δεν είναι διατεθειμένοι να προσφέρουν τους πόρους τους στο δίκτυο. Επίσης οι κόμβοι που προσφέρουν πληροφορία με υψηλούς ρυθμούς επίσης είναι σε θέση να λαμβάνουν πληροφορία με το ίδιο υψηλούς ρυθμούς.

2.3 Δομημένα Δίκτυα Ομοτίμων Εταίρων

Στα δομημένα δίκτυα ομοτίμων εταίρων, υπάρχει η έννοια της δομής με βάση την οποία δημιουργείται η τοπολογία και η συνεκτικότητα του δικτύου, η οποία διασφαλίζεται με τη βοήθεια προηγμένων κατανεμημένων δομών δεικτοδότησης, όπως για παράδειγμα των Κατανεμημένων Πινάκων Κατακερματισμού (DHT - Distributed Hash Table).

Σε ένα DHT κάθε κόμβος έχει ένα μοναδικό αναγνωριστικό *nodeID* το οποίο έχει επιλεγθεί τυχαία από ένα πολύ μεγάλο χώρο διευθύνσεων. Κάθε αντικείμενο (για παράδειγμα κάποιο έγγραφο, ένα αρχείο μουσικής κτλ.) αντιστοιχίζεται με ένα κλειδί που αποτελεί ένα μοναδικό αναγνωριστικό για το αντικείμενο, που και αυτό έχει επιλεγθεί από τον ίδιο χώρο διευθύνσεων που χρησιμοποιείται για τα αναγνωριστικά των κόμβων. Η βασική ιδέα είναι η αποθήκευση των αντικείμενα στους κόμβους των οποίων το αναγνωριστικό είναι ίδιο ή αριθμητικά πολύ κοντά στο κλειδί του αντικειμένου. Έτσι, κατά την αναζήτηση, αρκεί να

επισκεφθούμε τον κόμβο με αναγνωριστικό ίσο με αυτό του αντικειμένου.

Τα περισσότερα DHTs εγγυώνται τη δρομολόγηση ενός μηνύματος στον κατάλληλο κόμβο (με βάση το αναγνωριστικό του) σε $O(\log(N))$ βήματα (hops) σε ένα δίκτυο με N κόμβους (αυτή η διαδικασία συνήθως αποκαλείται DHT *lookup*()). Επίσης η ποσότητα πληροφορίας που διατηρείται σε κάθε κόμβο και σχετίζεται με την δρομολόγηση (πίνακας δρομολόγησης, λίστες γειτονικών κόμβων κτλ.) είναι λογαριθμική συνάρτηση του αριθμού των κόμβων N .

Σε αυτή τη κατηγορία ανήκουν τα πιο γνωστά δίκτυα ομοτίμων εταίρων, όπως το CAN [82], CHORD [88], Pastry [85], Tapestry [103], Bamboo [84], Skip-Net [55] και άλλα. Σε αυτή την ενότητα θα παρουσιάσουμε τη βασική λειτουργικότητα και χαρακτηριστικά των δημοφιλών δικτύων CHORD, Pastry και CAN, μιας και τα δύο πρώτα θα μας βοηθήσουν στην παρουσίαση των αλγορίθμων που έχουν αναπτυχθεί στα πλαίσια αυτής της διατριβής.

2.3.1 Το Δίκτυο Pastry

Το p2p δίκτυο Pastry [85], όπως επίσης και το Tapestry [103] αλλά και το Bamboo [84], στηρίζονται στον αλγόριθμο δρομολόγησης και εντοπισμού αντικειμένων που πρώτος πρότεινε ο Plaxton το 1997 [80].

Ο Plaxton παρουσιάζει στην εργασία [80] μία κατανεμημένη δομή δεδομένων (που είναι επίσης γνωστή και ως Plaxton Mesh) που χρησιμοποιείται για την αποδοτική δρομολόγηση μηνυμάτων και εντοπισμό αντικειμένων που είναι αποθηκευμένα σε δίκτυο αποτελούμενο από χιλιάδες κόμβους, χρησιμοποιώντας σταθερού μεγέθους πίνακες δρομολόγησης. Θεωρώντας την ύπαρξη ενός στατικού δικτύου, ο πίνακας δρομολόγησης σε κάθε κόμβο, αποτελείται από πολλαπλά επίπεδα. Σε κάθε επίπεδο i διατηρούνται «δείκτες» προς τους κόμβους των οποίων το αναγνωριστικό, $nodeID$, έχει το ίδιο επίθεμα μήκους i ψηφίων με το $nodeID$ του κόμβου που κατέχει τον συγκεκριμένο πίνακα δρομολόγησης. Η δρομολόγηση μηνυμάτων με τη χρήση αυτών των πινάκων, γίνεται με τον να ελέγχουμε ένα ψηφίο του αναγνωριστικού του αντικειμένου σε κάθε βήμα i , να κοιτάμε στο αντίστοιχο επίπεδο $i + 1$ στον πίνακα δρομολόγησης, και να προωθούμε το μήνυμα στον κόμβο του οποίου η εγγραφή βρέθηκε εκεί. Αυτός ο μηχανισμός εξασφαλίζει ότι ένας κόμβος μπορεί να εντοπιστεί κάνοντας το πολύ $m = \log_{\beta}(N)$ βήματα (hops), όπου N είναι ο μέγιστος αριθμός των κόμβων που μπορεί να υπάρξουν στο δίκτυο¹, β είναι η αριθμητική βάση των αναγνωριστικών (π.χ. εάν $\beta = 2$, τότε προκύπτουν αναγνωριστικά με δυαδικές τιμές) και m είναι ο αριθμός ψηφίων από τα οποία αποτελείται το κάθε αναγνωριστικό. Το μέγεθος των πινάκων δρομολόγησης είναι σταθερό και ίσο με $\beta \times \log_{\beta}(N)$. Το ότι ο αριθμός βημάτων για τον εντοπισμό ενός κόμβου με συγκεκριμένο αναγνωριστικό είναι ίσος με το μέγεθος των αναγνωριστικών, οφείλεται στον τρόπο με το οποίο γίνεται η δρομολόγηση με

¹ Στην ουσία το N είναι το μέγεθος του χώρου διευθύνσεων που χρησιμοποιούν οι κόμβοι αλλά και τα αντικείμενα για τα αναγνωριστικά τους.

Part of the Routing Table of Pastry node with id 37A0F1 (digits in hexadecimal, x is an arbitrary suffix)								
0x	1x	2x	3x	4x	...	...	Ex	Fx
30x	31x	32x	...	37x	38x	...	3Ex	3Fx
370x	371x	372x	...	37Ax	37Bx	...	37Ex	37Fx
37A0x	37A1x	37A2x	...	37ABx	37ACx	37ADx	37AEx	37AFx

Leaf Set and Neighborhood Set of Pastry node id 37A0F1			
Leaf Set			
37A001	37A011	37A022	37A033
37A044	34A066	34A066	37A077
37A0F2	37A0F4	37A0F6	37A0F8
37A0FA	37A0FB	37A0FC	37A0FE
Neighborhood Set			
1A223B	1B3467	2454D0	2670AB
3612AB	37890A	390AF0	3912CD
46710A	477810	4881AB	490CDE
279DE0	290A0B	510A0C	5213EA
11345B	122167	16228A	19902D
221145	267221	28989C	199ABC

Σχήμα 2.4: Πίνακας δρομολόγησης στο Pastry, με το leaf set και το neighbor set. Παράδειγμα δρομολόγησης πάνω στο δίκτυο Pastry.

την χρήση των πινάκων δρομολόγησης και των έλεγχου ενός ψηφίου του αναγνωριστικού σε κάθε βήμα.

Το δίκτυο Pastry [85], αποτελεί μία επέκταση της δομής που προτάθηκε από τον Plaxton. Το Pastry προσφέρει όπως άλλωστε και όλα τα δομημένα p2p δίκτυα, ευρωστία, κλιμάκωση και αποδοτική δρομολόγηση σε δυναμικά περιβάλλοντα με συνεχείς αφίξεις και αναχωρήσεις κόμβων. Ο μηχανισμός δρομολόγησης στο Pastry είναι παρόμοιος με αυτόν που έχει προταθεί από τον Plaxton, με πίνακες δρομολόγησης που αποτελούνται από $\beta \times \log_{\beta}(N)$ εγγραφές (με $\log_{\beta}(N)$ επίπεδα/γραμμές και β στήλες, Σχήμα 2.4), και έχουν σαν αποτέλεσμα τον εντοπισμό ενός κόμβου (ή τη δρομολόγηση ενός μηνύματος με συγκεκριμένο αναγνωριστικό) σε $\log_{\beta}(N)$ βήματα. Στην περίπτωση του Pastry το β ορίζεται να είναι ίσο με $\beta = 2^b$ όπου το b έχει συνήθως την τιμή 4, με αποτέλεσμα το ορισμό των αναγνωριστικών στο δεκαεξαδικό σύστημα αρίθμησης.

Η διαφορά με το μηχανισμό που πρότεινε ο Plaxton, είναι ο εντοπισμός του επόμενου κόμβου με βάση το πρόθεμα μήκους i ψηφίων στο βήμα i (αντί για επίθεμα), ενώ παράλληλα οι εγγραφές σε κάθε πίνακα δρομολόγησης περιέχουν τον πιο κοντινό κόμβο σε κάθε περίπτωση με βάση μία μετρική απόστασης (συνήθως βασισμένη στο χρόνο απόκρισης).

Επιπλέον, με στόχο την αξιόπιστη δρομολόγηση, υπό την παρουσία βλαβών στο δίκτυο ή ξαφνικών αναχωρήσεων των κόμβων, υπάρχει η έννοια του leaf set σε κάθε κόμβο (Σχήμα 2.4), που αποτελείται από εγγραφές με δείκτες προς του L κόμβους των οποίων το αναγνωριστικό είναι αριθμητικά κοντινό στο αναγνωριστικό του τρέχοντα κόμβου. Επίσης, στο Pastry υπάρχει και η έννοια του «γειτονικού» κόμβου, όπου είναι ένα σύνολο από δείκτες σε M κοντινούς κόμβους με βάση μία μετρική απόστασης (neighbor set,

στο Σχήμα 2.4), και χρησιμοποιείται για την διατήρηση ιδιοτήτων χωρικής συσχέτισης. Το Tapestry [103] και το Bamboo [84], αποτελούν και αυτά δίκτυα με παρόμοια χαρακτηριστικά και λειτουργικότητα με το Pastry.

Στο Σχήμα 2.4 μπορείτε να δείτε τα πρώτα τέσσερα επίπεδα του πίνακα δρομολόγησης του κόμβου με αναγνωριστικό $37A0F1$. Παρατηρήστε ότι το επίπεδο $i = 2$ (ξεκινώντας από το επίπεδο $i = 0$) περιέχει εγγραφές προς τους κόμβους με αναγνωριστικό που έχει κοινό το πρόθεμα μήκους 2 χαρακτήρων ($37*$) με τον κόμβο $37A0F1$, ενώ ο επόμενος χαρακτήρας μπορεί να είναι ένας από τους β διαθέσιμους. Η δρομολόγηση αιτήσεων ανάκτησης ή εντοπισμού ενός αντικειμένου με αναγνωριστικό id στο Pastry γίνεται με τον έλεγχο του προθέματος μήκους i χαρακτήρων και του τοπικού πίνακα του κόμβου (στο επίπεδο i) και την προώθηση της αίτησης στον αντίστοιχο κόμβο. Στην συνέχεια ελέγχουμε το $i + 1$ επίπεδο μέχρι να καταλήξουμε στον τελικό κόμβο. Στο Σχήμα 2.4, μπορείτε να δείτε ένα παράδειγμα εντοπισμού του αντικειμένου με $id = B57B2D$ από τον κόμβο με αναγνωριστικό $37A0F1$. Η διαδικασία δρομολόγησης ξεκινάει με τον έλεγχο του πίνακα δρομολόγησης του $37A0F1$ στο επίπεδο $i = 0$ και προώθηση της αίτησης στον κόμβο που είναι πιο κοντά και το αναγνωριστικό του ξεκινάει με τον ψηφίο B ($B24EA3$). Σε αυτό τον κόμβο ελέγχεται ο πίνακας δρομολόγησης του στο επίπεδο $i = 1$ για τον πιο κοντινό κόμβο με αναγνωριστικό που έχει κοινό πρόθεμα 1 ψηφίου με τον $B24EA3$ ενώ το επόμενο ψηφίο είναι ίσο με το αντίστοιχο ψηφίο του id του αντικειμένου (5). Η αίτηση λοιπόν προωθείται στον κόμβο $B532A7$. Η διαδικασία αυτή συνεχίζεται αυξάνοντας το i κατά ένα κάθε φορά, μέχρι να φτάσουμε στον τελικό κόμβο. Στο παράδειγμα αυτό θα παρατηρήσετε ότι ο κόμβος με αναγνωριστικό ίσο με $B57B2D$ δεν βρίσκεται στο δίκτυο, οπότε η αίτηση καταλήγει στο κόμβο με αναγνωριστικό αριθμητικά πιο κοντά στο $B24EA3$ που είναι ο $B573D6$.

2.3.2 Το Δίκτυο Chord

Ένα επίσης ιδιαίτερα δημοφιλές δίκτυο ομοτίμων εταίρων είναι το δίκτυο Chord [88], το οποίο εξασφαλίζει τη χρόνο απόκρισης(δομή» του με την παράθεση των κόμβων σε ένα νοητό δακτυλίδι (Σχήμα 2.5), με βάση την τιμή του αναγνωριστικού τους $nodeID$. Τα αναγνωριστικά τόσο των κόμβων όσο και των αντικειμένων αντλούνται τυχαία από ένα μεγάλο χώρο διευθύνσεων 2^m (για παράδειγμα $[0, 2^{160})$ για αναγνωριστικά αποτελούμενα από 160 bits) ώστε να αποφευχθούν συγκρούσεις. Οι τιμές σε αυτό το χώρο μπορεί να θεωρηθεί ότι αποτελούν σημεία ενός κυκλικού χώρου στον οποίο είναι ταξινομημένα τα αναγνωριστικά (Σχήμα 2.5). Έτσι, δοσμένου ενός αναγνωριστικού αντικειμένου, το Chord αντιστοιχίζει το αντικείμενο αυτό με μία θέση στον κυκλικό χώρο, που στη ουσία καταλαμβάνει ο κόμβος του οποίου το αναγνωριστικό είναι ίσο με αυτό του αντικειμένου. Αν αυτός ο κόμβος δεν υπάρχει ακόμα στο δίκτυο, τότε το αντικείμενο

Σχήμα 2.5: Το δακτυλίδι διευθύνσεων στο δίκτυο Chord αποτελούμενο από 10 κόμβους και 5 αποθηκευμένα αντικείμενα. Μία αίτηση αναζήτησης για το αντικείμενο K54 ξεκινάει από τον κόμβο N8 .

αυτό αντιστοιχίζεται με τον επόμενο (αριθμητικά) υπαρκτό κόμβο στον κυκλικό χώρο (successor).

Στο Chord, επιτυγχάνεται ακριβής αντιστοίχιση μεταξύ των αναγνωριστικών των κόμβων (*nodeID*) και των αναγνωριστικών των αντικειμένων με την χρήση μίας «συνεπούς» συνάρτησης κατακερματισμού (Consistent Hashing [58]). Η έννοια του consistent hashing έχει να κάνει με την αντιστοίχιση των αντικειμένων σε κόμβους και με την διατήρηση αυτής της αντιστοίχιας ακόμα και όταν οι κόμβοι αποχωρούν από το δίκτυο. Υποθέστε ότι ένα αντικείμενο με αναγνωριστικό k αντιστοιχίζεται με το τον κόμβο A (δηλ. $A = k$). Εάν ο κόμβος A αποχωρήσει από το δίκτυο τότε θα πρέπει να υπάρχει κάποιος κόμβος, έστω B ($B = \text{successor}(A)$), που πλέον θα διαχειρίζεται αιτήσεις και αντικείμενα που προορίζονταν για τον A . Έτσι το αντικείμενο k θα πρέπει να μεταφερθεί στον B κατά την αποχώρηση του A , ώστε μελλοντικές αιτήσεις αναζήτησης του, να καταλήξουν στον B και άρα και στο k .

Στο Σχήμα 2.5 παρουσιάζεται το δακτυλίδι διευθύνσεων ενός δικτύου Chord με $m = 6$ (άρα ο χώρος διευθύνσεων περιέχει $2^m = 64$ τιμές), αποτελούμενο από 10 κόμβους που έχουν αποθηκευμένα 5 αντικείμενα. Κατά την αποθήκευση του αντικειμένου με αναγνωριστικό $K10$, θα πρέπει να εντοπιστεί ο κόμβος $N10$ και να αποθηκευτεί εκεί. Δεδομένου όμως ότι ο $N10$ είναι απών, επιλέγουμε τον successor του που είναι ο κόμβος $N14$ ($N14 = \text{successor}(N10)$) για να αποθηκευτεί το αντικείμενο.

Κάθε κόμβος στο δίκτυο Chord διατηρεί πίνακα δρομολόγησης (finger table) με δείκτες προς m άλλους κόμβους. Σε κάθε γραμμή i του πίνακα δρομολόγησης του κόμβου με αναγνωριστικό *nodeID* υπάρχει δείκτης προς τον successor του κόμβους με αναγνωριστικό $\text{nodeID} + 2^{i-1}$. Στο Σχήμα 2.5 μπορείτε να δείτε

Σχήμα 2.6: Δρομολόγηση αιτήσεων στο δισδιάστατο χώρο ($d = 2$) του δικτύου CAN και άφιξη νέου κόμβου.

το finger table του κόμβους με $nodeID=8$. Επιπλέον με τον πίνακα δρομολόγησης και για την αντιμετώπιση των σφαλμάτων που μπορεί να προκύψουν, κάθε κόμβος διατηρεί και μία λίστα με επόμενους r successors του.

Το Chord, όπως άλλωστε και όλα τα δίκτυα ομοτίμων που ανήκουν στην οικογένεια των δομημένων δικτύων, είναι ιδιαίτερα αποδοτικό κατά την δρομολόγηση και εντοπισμό ενός κόμβου με βάση το αναγνωριστικό του. Απαιτούνται για αυτό το σκοπό $O(\log(N))$ μηνύματα στη χειρότερη περίπτωση με την χρήση των finger tables των κόμβων σε κάθε βήμα της δρομολόγησης. Στο Σχήμα 2.5 μπορείτε να δείτε ένα παράδειγμα δρομολόγησης αίτησης αναζήτησης του αντικειμένου με αναγνωριστικό 54 από τον κόμβο με $nodeID=8$.

2.3.3 Το Δίκτυο CAN

Το δίκτυο CAN (Content Addressable Network, [82]) είναι δομημένο πάνω σε ένα πολυδιάστατο χώρο Καρτεσιανών συντεταγμένων με d διαστάσεις (Σχήμα 2.6). Κάθε κόμβος είναι υπεύθυνος για μία συγκεκριμένη ζώνη του d -διάστατου χώρου. Ο διαμοιρασμός και η ανάθεση των ζωνών γίνεται δυναμικά καθώς νέοι κόμβοι καταφθάνουν στο σύστημα. Κάθε κόμβος γνωρίζει τις συντεταγμένες της ζώνης του καθώς και τις συντεταγμένες των γειτονικών ζωνών, πληροφορία που είναι απαραίτητη κατά την δρομολόγηση των αιτήσεων.

Τα αντικείμενα στο δίκτυο CAN συνοδεύονται από ένα μοναδικό κλειδί ή αναγνωριστικό. Στη συνέχεια χρησιμοποιώντας μία συνάρτηση κατακερματισμού, αντιστοιχίζονται τα αναγνωριστικά των αντικειμένων με

μοναδικά σημεία του του πολυδιάστατου χώρου. Εφόσον οι κόμβοι είναι υπεύθυνοι για συγκεκριμένες ζώνες, τα αντικείμενα τελικά αποθηκεύονται σε εκείνους τους κόμβους των οποίων η ζώνη περιλαμβάνει τις συντεταγμένες που αντιστοιχούν στο αντικείμενο.

Ο εντοπισμός ενός αντικειμένου με συγκεκριμένο αναγνωριστικό (και άρα συντεταγμένες) από ένα κόμβο X (Σχήμα 2.6) πραγματοποιείται με την αποστολή της αίτησης στον κόμβο που διαχειρίζεται την γειτονική ζώνη του X που βρίσκεται προς της κατεύθυνση του σημείου που αντιστοιχεί στο αντικείμενο. Κάθε κόμβος στο CAN διατηρεί πίνακα δρομολόγησης με $2 \times d$ εγγραφές, ενώ η δρομολόγηση μίας αίτησης απαιτεί την εκτέλεση $d \times N^{1/d}$ βημάτων σε ένα δίκτυο αποτελούμενο από N κόμβους.

Κατά την εισαγωγή ενός νέο κόμβου στο δίκτυο CAN (π.χ. κόμβος Z στο Σχήμα 2.6) , επιλέγεται τυχαία ένα σημείο στον πολυδιάστατο χώρο και προωθείται ένα ειδικού τύπου μήνυμα προς τον κόμβο που κατέχει την ζώνη που περικλείει το σημείο αυτό (κόμβος Q στο Σχήμα 2.6). Όταν αυτό το μήνυμα φθάσει στον προορισμό του, η ζώνη μοιράζεται σε δύο μέρη, όπου το ένα αναλογεί πλέον στον νέο κόμβο. Οι πίνακες δρομολόγησης όλων των γειτονικών κόμβων ενημερώνονται κατάλληλα. Η αντίστροφη διαδικασία ακολουθείται όταν ένας κόμβος αποφασίσει να αναχωρήσει από το δίκτυο. Η απόδοση του CAN μπορεί να βελτιωθεί με την ύπαρξη r διαφορετικών πολυδιάστατων χώρων, όπου οι κόμβοι συμμετέχουν σε κάθε ένα από τους r χώρους.

Το δίκτυο CAN έχει αξιοποιηθεί από συστήματα διαχείρισης και αποθήκευσης δεδομένων όπως τα OceanStore [65], Farsite [23], και Publius [99].

2.4 Συστήματα Δημοσιεύσεων Συνδρομών

Τα τελευταία χρόνια, η διαχείριση πληροφορίας σε μεγάλης κλίμακας κατανεμημένα συστήματα, όπως αυτά των δικτύων ομοτίμων εταιρών, έχει αντιμετωπιστεί με δύο διαφορετικές προσεγγίσεις. Η πρώτη, εφαρμόζει παραδοσιακές τεχνικές διαχείρισης βάσεων δεδομένων και προσφέρει τη σχετική με αυτό το περιβάλλον λειτουργικότητα (όπως για παράδειγμα το PIER [56], και άλλες σχετικές εργασίες [16, 24, 74]). Σε αυτή την ερευνητική περιοχή, η πληροφορία (για παράδειγμα οι πίνακες μίας βάσεως δεδομένων) είναι αποθηκευμένη στο δίκτυο κατάλληλα, έτσι ώστε τα εισερχόμενα ερωτήματα (ερωτήματα SQL) να εντοπίζουν και να ανακτούν την πληροφορία.

Η δεύτερη ερευνητική περιοχή, εστιάζει στην εφαρμογή του μοντέλου δημοσιεύσεων/συνδρομών, για την κατανεμημένη διαχείριση της πληροφορίας. Σε αυτό το μοντέλο, οι χρήστες (συνδρομητές ή subscribers) εκφράζουν τα ενδιαφέροντα τους (με τη μορφή συνδρομών) τα οποία στη συνέχεια αποθηκεύονται κατάλληλα στο δίκτυο. Από την άλλη πλευρά, υπάρχουν οι χρήστες που παίζουν τον ρόλο του εκδότη

(publishers), διοχετεύοντας στο σύστημα πληροφορία, η οποία εφαρμόζοντας κατάλληλους αλγορίθμους εντοπίζει και ενημερώνει τους συνδρομητές που έχουν εκδηλώσει σχετικό ενδιαφέρον. Γενικά, ένας συνδρομητής ενδιαφέρεται για ένα μικρό ποσοστό των δημοσιεύσεων που γεννιούνται στο σύστημα. Η βασική διαφορά λοιπόν του μοντέλου δημοσιεύσεων/συνδρομών με το μοντέλο κατανεμημένης βάσης δεδομένων, είναι ότι στο πρώτο οι συνδρομές (ή αλλιώς ερωτήματα στη γλώσσα των βάσεων δεδομένων) αποθηκεύονται στο δίκτυο και η πληροφορία με τη μορφή δημοσιευμάτων επεξεργάζεται κατάλληλα ώστε να εντοπιστούν οι συνδρομητές των οποίων τα ενδιαφέροντα συζευγνύουν με τα δημοσιεύματα.

Τύποι Συστημάτων Δημοσιεύσεων/Συνδρομών

Γενικά υπάρχουν δύο βασικές κατηγορίες ([45]) συστημάτων δημοσιεύσεων/συνδρομών: (i) αυτά που είναι βασισμένα σε ομάδες (group-based) ή σε θεματικές ενότητες (topic-based) και (ii) αυτά που είναι βασισμένα στο ίδιο το περιεχόμενο (content-based).

Η πρώτη γενιά συστημάτων δημοσιεύσεων/συνδρομών αποτελούνταν από group-based συστήματα ([1, 21, 36]). Σε αυτά υπήρχε η έννοια των groups που ήταν ορισμένα κατάλληλα από το σύστημα. Οι συνδρομητές, δήλωναν τα ενδιαφέροντα τους με το να εγγράφονται σε ένα ή περισσότερα groups. Στη συνέχεια οι εκδότες διοχέτευαν τα μηνύματα τους στο κατάλληλο με τη πληροφορία που περιείχαν group, με αποτέλεσμα αυτά τελικά να φτάνουν στους τελικούς συνδρομητές. Μία ελαφρώς διαφορετική προσέγγιση, αποτέλεσαν αργότερα τα συστήματα δημοσιεύσεων/συνδρομών βασισμένα σε θεματικές ενότητες (topics). Τα μηνύματα πλέον χαρακτηρίζονταν από μία συμβολοσειρά που περιέγραφε τη θεματική περιοχή του μηνύματος. Οι συνδρομητές μπορούσαν πλέον να εκφραστούν καλύτερα, με το να δηλώνουν για παράδειγμα τα ενδιαφέροντα τους όχι μόνο με απόλυτη ταύτιση της θεματικής περιοχής αλλά χρησιμοποιώντας το κοινό πρόθεμα όλων των θεματικών εννοιών που τους ενδιέφεραν. Στη κατηγορία των topic ή group-based συστημάτων δημοσιεύσεων/συνδρομών ανήκουν το Scribe [33], το Bayeux [105], το daMulticast [13] το TIB/RV [73] και [83].

Αργότερα, στις αρχές του αιώνα εμφανίστηκαν και επικράτησαν τα συστήματα δημοσιεύσεων/συνδρομών βασισμένα στο περιεχόμενο (content-based). Τα content-based συστήματα επικράτησαν επειδή ακριβώς προσφέρουν μεγαλύτερη ευελιξία κατά την διατύπωση των ενδιαφερόντων των χρηστών, με το να εφαρμόζουν συγκεκριμένους τελεστές πάνω στο ίδιο το περιεχόμενο. Δίνεται έτσι μεγαλύτερη ευελιξία και έλεγχος στον συνδρομητή που πλέον μπορεί να ορίσει πιο σύνθετες και εκφραστικότερες συνδρομές, όπως για παράδειγμα «ενημέρωσε με για την τιμή της μετοχής της Google όταν αυτή υπερβεί τα 500 \$». Συγκεκριμένα, υπάρχει η έννοια των ιδιοχαρακτηριστικών (attributes) τα οποία συνθέτουν τόσο τις συνδρομές όσο και τις δημοσιεύσεις. Κάθε ιδιοχαρακτηριστικό έχει συγκεκριμένο τύπο (κυρίως συμβολοσειρά ή α-

ριθμητική τιμή) και μία συγκεκριμένη τιμή. Τα υπό δημοσίευση μηνύματα περιέχουν ένα υποσύνολο από τα διαθέσιμα από το σύστημα ιδιοχαρακτηριστικά με συγκεκριμένες τιμές. Από την άλλη, οι συνδρομές περιέχουν και αυτές ένα υποσύνολο των ιδιοχαρακτηριστικών, με συγκεκριμένους τελεστές πάνω στις τιμές τους. Πιθανοί τελεστές είναι για παράδειγμα το ισότητα, πρόθεμα (prefix), επίθεμα (suffix), υποσυμβολοσειρά (substring) για ιδιοχαρακτηριστικά τύπου συμβολοσειράς και ισότητα, ανισότητα, διαστήματα τιμών για αριθμητικούς τύπους. Κάτω από αυτό το μοντέλο, ένα μήνυμα συζευγνύεται με μία συνδρομή, εάν όλοι οι τελεστές των ιδιοχαρακτηριστικών της συνδρομής ικανοποιούνται από τις τιμές των ιδιοχαρακτηριστικών του μηνύματος. Η μεγαλύτερη πρόκληση σε ένα πλήρως καταναμημένο σύστημα δημοσιεύσεων/συνδρομών βασισμένο στο περιεχόμενο, είναι η ανάπτυξη αποδοτικών καταναμημένων αλγορίθμων επεξεργασίας και σύζευξης των συνδρομών με τις δημοσιεύσεις, αλλά και παράδοσης των τελευταίων στους τελικούς χρήστες.

Σε αυτή την ερευνητική περιοχή, που εστιάζει και η παρούσα διατριβή, έχουν παρουσιαστεί τα τελευταία χρόνια αρκετές λύσεις όπως το SIENA [31, 32], το REBECA [47], το Gryphon [11], το Hermes [76], το Meghdoot [52, 53], το MEDYM [26], το JEDI [40], Elvin [86] και οι λύσεις που παρουσιάστηκαν στην σχετική βιβλιογραφία [14, 15, 29, 46, 57, 70, 89, 91, 93, 94, 100].

Ας θεωρηθεί ότι για το υπόλοιπο κείμενο, οποτεδήποτε αναφερόμαστε σε συστήματα δημοσιεύσεων/συνδρομών, εννοούμε τα συστήματα δημοσιεύσεων/συνδρομών βασισμένα στο περιεχόμενο μιας και είναι πιο γενικός και επικρατέστερος τύπος. Άλλωστε, τα συστήματα δημοσιεύσεων/συνδρομών βασισμένα σε θεματικές ενότητες μπορούν να θεωρηθούν σαν υποσύνολο των συστημάτων βασισμένα στο περιεχόμενο, αν θεωρήσει κανείς ότι κάθε θεματική περιοχή μπορεί να θεωρηθεί ότι είναι η τιμή ενός μόνο ιδιοχαρακτηριστικού.

2.4.1 Κατηγοριοποίηση Συστημάτων Δημοσιεύσεων/Συνδρομών Βασισμένα στο Περιεχόμενο

Σχετικές εργασίες που έχουν παρουσιαστεί στο πλαίσιο των συστημάτων δημοσιεύσεων/συνδρομών βασισμένα στο περιεχόμενο, μπορούν να κατηγοριοποιηθούν ανάλογα με : (i) το τύπο του δικτύου που χρησιμοποιείται, (ii) την ικανότητα που έχουν αυτά τα συστήματα να διαχειριστούν πολύπλοκα ερωτήματα (συνδρομές με σύνθετους περιορισμούς) πάνω σε συμβολοσειρές και αριθμητικές τιμές, και (iii) ανάλογα με το αν διατηρούν ή όχι πίνακες δρομολόγησης ή σχετική πληροφορία ανεξάρτητη με αυτή που χρησιμοποιεί το ίδιο το δίκτυο.

Τύπος Δικτύου

Σχετικά με τον τύπο του δικτύου πάνω στο οποίο μπορεί να κτιστεί ένα σύστημα δημοσιεύσεων/συνδρομών, η σχετική βιβλιογραφία έχει να επιδείξει συστήματα τα οποία εντάσσονται σε δύο κατηγορίες ανάλογα με το αν χρησιμοποιούν ένα δίκτυο ομοτίμων εταίρων ή όχι.

Από αυτά τα συστήματα που επωφελούνται από την ύπαρξη ενός δικτύου ομοτίμων εταίρων, άλλα μπορούν να εφαρμοστούν πάνω από οποιοδήποτε τέτοιο δίκτυο και άλλα μπορούν να εφαρμοστούν σε συγκεκριμένα δίκτυα, επειδή ακριβώς αξιοποιούν τα ιδιαίτερα χαρακτηριστικά τους.

(i) Δίκτυα Υπολογιστών Αποκλειστικής Χρήσης από το Σύστημα Δημοσιεύσεων/Συνδρομών - Non-P2P

Οι πρώτες ολοκληρωμένες δουλειές στην ερευνητική περιοχή των συστημάτων δημοσιεύσεων/συνδρομών, είχαν αναπτύξει και χρησιμοποιούσαν ένα αποκλειστικό δίκτυο από συνήθως καλά συνδεδεμένους κόμβους (υπολογιστικές μηχανές). Αυτό το μοντέλο προϋποθέτει ότι όλοι αυτοί οι κόμβοι είναι ιδιαίτερα ισχυροί υπολογιστές μιας και αναλαμβάνουν να εφαρμόσουν τους κατανεμημένους αλγόριθμους σύζευξης των δημοσιεύσεων με τις συνδρομές. Οι χρήστες, συνδρομητές ή εκδότες, συνδέονται σε αυτού το δίκτυο και δημοσιεύουν τα ενδιαφέροντα ή τα μηνύματα τους.

Αντιπροσωπευτικές εργασίες αυτής της κατηγορίας, προερχόμενες από τα πρώτα χρόνια ανάπτυξης του κατανεμημένου μοντέλου δημοσιεύσεων συνδρομών, αποτελούν οι [11, 15, 26, 31, 32, 40, 47, 93, 94].

(ii) Ανεξάρτητα από το DHT Δίκτυο Ομοτίμων Εταίρων - DHT-Independent

Αρκετές μεταγενέστερες προσεγγίσεις κινήθηκαν προς την κατεύθυνση της ενσωμάτωσης του μοντέλου των δημοσιεύσεων/συνδρομών σε δίκτυα ομοτίμων εταίρων βασισμένα σε DHTs, έτσι ώστε να επωφεληθούν από τα θετικά στοιχεία αυτών, όπως η κλιμάκωση, η αποδοτικότητα και η αυτόνομη οργάνωση τους.

Στην κατηγορία των DHT-Independent προσεγγίσεων, ανήκουν αυτές που χειρίζονται το DHT δίκτυο σαν ένα «μαύρο κουτί», στηριζόμενοι μόνο στις γνωστές προσφερόμενες διεπαφές και συγκεκριμένα στην *lookup()* λειτουργικότητα. Αυτή προσφέρει την δυνατότητα να εντοπιστεί ο κόμβος και να προωθηθεί σε αυτόν ένα μήνυμα ή αντικείμενο με βάση το αναγνωριστικό του. Σχετικές με την κατηγορία αυτή εργασίες είναι οι [5, 14, 19, 50, 81, 89, 92].

(iii) Εξαρτώμενα από το DHT Δίκτυο Ομοτίμων Εταίρων - DHT-Dependent

Σε αυτή τη κατηγορία ανήκουν επίσης συστήματα που έχουν κτιστεί πάνω σε DHT δίκτυα ομοτίμων εταίρων. Ωστόσο, εκμεταλλεύονται τα ιδιαίτερα χαρακτηριστικά που μπορεί να έχουν με αποτέλεσμα να μπορούν να εφαρμοστούν μόνο σε συγκεκριμένα DHTs. Σε αυτές εργασίες, συνήθως, η επεξεργασία των συνδρομών και των δημοσιεύσεων είναι στενά συνδεδεμένη με την αρχιτεκτονική ενός συγκεκριμένου DHT. Για παράδειγμα, μπορεί να εξαρτάται από τον αλγόριθμο δρομολόγησης που έχει εφαρμοστεί, ή

ακόμα μπορεί να χρησιμοποιεί πληροφορία που το ίδιο το DHT δεν προσφέρει μέσα από τις διεπαφές του. Αντιπροσωπευτικά μέλη αυτής της κατηγορίας αποτελούν τα [6, 41, 51, 53, 76, 87, 90, 91, 98].

Λειτουργικότητα και Εκφραστικότητα

Ένα σημαντικό χαρακτηριστικό των συστημάτων δημοσιεύσεων/συνδρομών με βάση το περιεχόμενο, σχετίζεται με το αν προσφέρουν την δυνατότητα διαχείρισης τόσο αριθμητικών τύπων όσο και συμβολοσειρών.

Αυτή η δυνατότητα είναι ξεκάθαρη σε συστήματα που δεν αναπτύσσονται πάνω σε δίκτυα ομοτίμων εταιρών (Non-P2P) ή που εκμεταλλεύονται τα ιδιαίτερα χαρακτηριστικά ενός DHT (DHT-Dependent) όπως για παράδειγμα οι ακόλουθες εργασίες [6, 30, 76, 91]. Σε αυτές τις περιπτώσεις, συνήθως η υποστήριξη αριθμητικών τύπων και συμβολοσειρών, επιτυγχάνεται με την διατήρηση επιπλέον πληροφορίας σε κάθε κόμβο με τη μορφή «φίλτρων» συνδρομών που αντιστοιχίζεται με τους πίνακες δρομολόγησης ή τον αλγόριθμο δρομολόγησης του κάθε κόμβου.

Από την άλλη πλευρά, σε μία DHT-Independent προσέγγιση, όπου κανείς καλείται να χρησιμοποιήσει τις διεπαφές που προσφέρει το DHT και συγκεκριμένα την *lookup()* λειτουργικότητα, η υποστήριξη σύνθετων ερωτημάτων σε αριθμητικές τιμές και συμβολοσειρές είναι μία πρόκληση. Σε τέτοια συστήματα, ερωτήματα ισότητας τιμών (exact-match) είναι σχετικά εύκολο να υλοποιηθούν, όμως δεν συμβαίνει το ίδιο για πιο σύνθετα ερωτήματα. Έτσι η υποστήριξη αυτών, πρέπει να επιτευχθεί με την ανάπτυξη σχετικών αλγορίθμων και τεχνικών που στηρίζονται αποκλειστικά και μόνο στις κλήσεις/διεπαφές που προσφέρει το DHT. Σχετικές εργασίες προς αυτή την κατεύθυνση έχουν κατά καιρούς παρουσιαστεί και ασχολούνται αποκλειστικά με σύνθετα ερωτήματα σε αριθμητικές τιμές [14, 15, 19, 53, 92], και σε συμβολοσειρές [5, 7, 8, 98].

Διατήρηση Πληροφορίας Σχετική με Δρομολόγηση

Μία σημαντική σχεδιαστική επιλογή έχει να κάνει με το αν οι κόμβοι που ανήκουν στο DHT πρέπει ή όχι να διατηρούν πληροφορία δρομολόγησης (διαφορετική από αυτή που απαιτείται από τους αλγορίθμους δρομολόγησης του ίδιου του DHT). Αν αυτό είναι αναγκαίο, αναφερόμαστε σε *stateful* προσεγγίσεις, ενώ αν δεν είναι σε *stateless*.

Η επιπλέον πληροφορία δρομολόγησης συνήθως είναι απαραίτητη ώστε να βελτιωθεί η απόδοση κατά την επεξεργασία των δημοσιεύσεων και/ή των συνδρομών που σχετίζεται με τον συνολικό αριθμό βημάτων (hops) που απαιτούνται. Αυτή η πληροφορία δρομολόγησης κτίζεται κατά την επεξεργασία των συνδρομών και συγκεκριμένα όταν αυτά κατανέμονται στο δίκτυο. Έτσι, μία συνδρομή που προωθείται από τον κόμβο n_1 στον n_2 σχετίζεται με αυτή τη σύνδεση. Στη συνέχεια όταν καταφθάσει ένα μήνυμα στον κόμβο n_1 , αυτό προωθείται στον κόμβο n_2 μόνο εάν ο n_1 έχει λάβει μία συνδρομή από τον n_2 που συζευγνύει με το μήνυμα.

Στην ουσία λοιπόν, στις *stateful* προσεγγίσεις αυτό που συνήθως γίνεται, είναι η ανάπτυξη ενός multicast δέντρου με τα φύλλα του να αποτελούνται από τους συνδρομητές και με τα μηνύματα να τα διατρέχουν και σε κάθε βήμα να φιλτράρονται κατάλληλα, ώστε να φτάνουν τελικά μόνο στους κατάλληλους συνδρομητές (αυτούς που με τις συνδρομές τους έχουν δηλώσει ενδιαφέρον για τα συγκεκριμένα μηνύματα). Σε αυτή την κατηγορία ανήκουν οι ακόλουθες εργασίες [6, 76, 89–91, 98].

Στην κατηγορία των *stateless* προσεγγίσεων, ανήκουν αυτές που ακολουθούν τη τεχνική των κόμβων «ραντεβού». Με κατάλληλους αλγορίθμους, αυτές οι προσεγγίσεις εγγυώνται ότι δημοσιεύσεις και συνδρομές που συζευγνύουν θα συναντηθούν στον ίδιο κόμβο. Από εκεί και πέρα η σύζευξη γίνεται τοπικά και οι κατάλληλοι συνδρομητές ενημερώνονται για τη δημοσίευση που συζευγνύει με τα ενδιαφέροντα τους. Σε αυτή τη κατηγορία ανήκουν οι εργασίες [5, 14, 41, 50, 53, 87, 92].

2.5 Ανασκόπηση Γνωστών Συστημάτων

Δημοσιεύσεων/Συνδρομών Βασισμένα στο Περιεχόμενο

2.5.1 Τα Συστήματα Scribe και Bayeux

Το Scribe [33] είναι ένα κλιμακώσιμο σύστημα δημοσιεύσεων/συνδρομών βασισμένο σε θεματικές ενότητες (topic-based). Λειτουργεί σε ένα πλήρως καταναεμημένο περιβάλλον πάνω στο δίκτυο Pastry [85]. Ως εκ τούτου, εκμεταλλεύεται την αξιοπιστία, την ανοχή σε σφάλματα και την αυτόνομη οργάνωση του δικτύου Pastry. Όλοι οι κόμβοι που συμμετέχουν στο Scribe είναι πλήρως αυτόνομοι και επεξεργάζονται τις δημοσιεύσεις και τις συνδρομές βασιζόμενοι αποκλειστικά σε τοπική πληροφορία. Κάθε κόμβος, μπορεί να είναι συνδρομητής, πάροχος πληροφορίας, ή να συμμετέχει σε multicast δέντρα που σκοπό έχουν την παράδοση των δημοσιεύσεων στους ενδιαφερόμενους συνδρομητές.

Στο Scribe, κάθε θεματική ενότητα (Subject ή Topic) συνοδεύεται από ένα τυχαίο αναγνωριστικό. Ακολουθώντας την προσέγγιση του κόμβου «ραντεβού», ο κόμβος με αναγνωριστικό που είναι αριθμητικά κοντά στο αναγνωριστικό της θεματικής ενότητας, είναι υπεύθυνος για την διαχείριση των δημοσιεύσεων και των συνδρομών που σχετίζονται με αυτή. Έτσι όλες η δημοσιεύσεις (αλλά και οι συνδρομές ή αλλιώς οι αιτήσεις συμμετοχής στη θεματική ενότητα) καταλήγουν στον κόμβο «ραντεβού» της θεματικής ενότητας, που στην συνέχεια είναι υπεύθυνος για την παράδοση τους στους συνδρομητές.

Κάθε συνδρομητής που επιθυμεί να εγγραφεί σε μία θεματική ενότητα, στέλνει ένα ειδικού τύπου μήνυμα (join subject) προς τον κόμβο «ραντεβού» της θεματικής ενότητας. Το μονοπάτι πάνω στο δίκτυο Pastry που σχηματίζεται κατά την δρομολόγηση αυτού του μηνύματος, αποτελεί τελικά ένα μονοπάτι στο

κατηγορία του *filtering-based routing* [68] και σκοπό έχει να σταματήσει όσο το δυνατόν συντομότερα τη μετάδοση μίας δημοσίευσης που δεν ενδιαφέρει κανένα συνδρομητή. Τα μονοπάτια παράδοσης των δημοσιεύσεων στους συνδρομητές, σχηματίζονται χρησιμοποιώντας πληροφορία που είναι τοπικά αποθηκευμένη σε κάθε κόμβο (φίλτρα) που σχηματίζεται από την κατάλληλη επεξεργασία των συνδρομών. Η πληροφορία δρομολόγησης σε ένα κόμβο σχετίζεται με τους γειτονικούς του κόμβους και τους συνδρομητές που μπορούν να προσπελαστούν μέσω αυτών. Αυτή η πληροφορία επιτρέπει τη δημιουργία αντίστροφων μονοπατιών (προς τους συνδρομητές) που μπορούν να ακολουθήσουν στη συνέχεια οι δημοσιεύσεις

Επιπλέον στο SIENA παρουσιάζεται μία τεχνική περιορισμού της μετάδοσης των συνδρομών με σκοπό να σχηματιστούν τα αντίστροφα μονοπάτια, με το να εκμεταλλεύεται την σχέση μεταξύ των συνδρομών. Δηλαδή, μία συνδρομή s_1 περικλείει την συνδρομή s_2 εάν όλες οι δημοσιεύσεις που συζευγνύουν με την συνδρομή s_2 , συζευγνύουν και με την s_1 (όπως μπορείτε να δείτε στο Σχήμα 2.7 όπου η συνδρομή s_2 με περιορισμό διαστήματος $[60, 70]$ καλύπτεται από τη συνδρομή s_1 με περιορισμό διαστήματος $[50, 100]$). Εκμεταλλευόμενοι αυτή τη παρατήρηση οι συνδρομές ταξιδεύουν μέχρι ένα ορισμένο σημείο στο δίκτυο όπου συναντούν ένα κόμβο που κατέχει ένα φίλτρο κατασκευασμένο από συνδρομές που περικλείουν τη συνδρομή.

Μια ακόμα τεχνική, που αποτελεί χαρακτηριστικό γνώρισμα του SIENA είναι οι διαφημίσεις. Μια διαφήμιση ορίζει το είδος των δημοσιεύσεων που ένα πάροχος πληροφορίας πρόκειται να προσφέρει στο σύστημα, και χρησιμοποιείται για την δημιουργία των μονοπατιών που τελικά θα ακολουθήσουν οι δημοσιεύσεις μέχρι τους τελικούς συνδρομητές.

Το σύστημα SIENA αποτελεί σημείο αναφοράς στην δρομολόγησης με βάση το περιεχόμενο και έχουν υιοθετηθεί οι λύσεις που παρουσιάζονται εκεί από άλλες ερευνητικές εργασίες. Υποστηρίζει αποδοτικά δημοσιεύσεις και συνδρομές με αριθμητικές τιμές και συμβολοσειρές, ενώ μπορεί να θεωρηθεί σαν μία *stateful* προσέγγιση εξαιτίας της πληροφορίας σχετικά με τη δρομολόγηση που πρέπει να διατηρείται σε κάθε κόμβο.

2.5.3 Το Σύστημα REBECA

Το σύστημα δημοσιεύσεων/συνδρομών REBECA [47], ενσωματώνει μία αντικειμενοστρεφή υπηρεσία παράδοσης δημοσιεύσεων, υλοποιημένη σε Java. Οι κόμβοι που συμμετέχουν στο σύστημα σχηματίζουν ένα ακυκλικό γράφο. Η δρομολόγηση των δημοσιεύσεων στους συνδρομητές ακολουθεί το μοντέλο *filtering-based* όπως και στο SIENA και έχει επεκταθεί ώστε να περιλαμβάνει (i) απλή δρομολόγηση που αντιστοιχεί σε *flooding* των συνδρομών σε ολόκληρο το δίκτυο ώστε οι δημοσιεύσεις να φιλτράρονται στους κόμβους όπου γεννιούνται, (ii) *identity routing* που αποφεύγει την προώθηση μιας συνδρομής σε επόμενο κόμβο

*, δείχνει ότι οι συνδρομές διαμέσου της συγκεκριμένης διακλάδωσης δεν ενδιαφέρονται για την τιμή του ιδιοχαρακτηριστικού ρ_1 . Εάν ακολουθήσουμε τη δεξιότερη διαδρομή, θα καταλήξουμε σε μία συνδρομή που έχει ως περιορισμούς τους $\rho_1=1 \wedge \rho_2=2 \wedge \rho_3=3 \wedge \rho_5=3$.

Η σύζευξη μίας δημοσίευσης επιτυγχάνεται ξεκινώντας από την ρίζα του δέντρου και με βάση τις τιμές των ιδιοχαρακτηριστικών του, καταλήγει τελικά στα φύλλα που αντιστοιχούν σε συνδρομές που συζευγνύουν με τη δημοσίευση. Ξεκινώντας από τη ρίζα και ελέγχοντας το ιδιοχαρακτηριστικό ρ_1 προχωράμε αριστερά ή δεξιά από τη ρίζα στον επόμενο κόμβο. Σε κάθε ενδιάμεσο κόμβο του δέντρου, ακολουθούμε τη ακμή που η ετικέτα της ικανοποιεί την τιμή του ιδιοχαρακτηριστικού της δημοσίευσης που αντιστοιχεί στο συγκεκριμένο βάθος, ή ακολουθούμε την ακμή με ετικέτα *. Με αυτό τον κανόνα, ξεκινάει μια παράλληλη αναζήτηση στο εσωτερικό του δέντρου. Για παράδειγμα, αν εκτελέσουμε τον αλγόριθμο ταύτισης στο δέντρο του Σχήματος 2.8 για την δημοσίευση με ιδιοχαρακτηριστικά $\rho_1=1, \rho_2=2, \rho_3=3, \rho_4=1, \rho_5=2$, τότε θα επισκεφθούμε όλους τους κόμβους με μαύρο χρώμα και θα βρούμε τέσσερις συνδρομές (τα φύλλα του δέντρου με μαύρο χρώμα).

2.5.5 Το Σύστημα Hermes

Το σύστημα Hermes αποτελεί μια ολοκληρωμένη λύση επεξεργασίας δημοσιεύσεων και συνδρομών συνδυάζοντας το topic-based μοντέλο με το content-based. Είναι λοιπόν ένας συνδυασμός των δύο βασικών μοντέλων των συστημάτων δημοσιεύσεων συνδρομών. Το Hermes βασίζεται στη προσέγγιση του κόμβου «ραντεβού» με το μοντέλο δημοσιεύσεων/συνδρομών με βάση τη θεματική ενότητα.

Το δίκτυο πάνω στο οποίο λειτουργεί το Hermes αποτελείται από brokers ωστόσο έχουν προταθεί και προσεγγίσεις που εκμεταλλεύονται άλλου είδους αρχιτεκτονικές [77] ώστε να αντιμετωπίσουν προβλήματα σχετικά με την αναχώρηση κόμβων από το δίκτυο ή την ύπαρξη σφαλμάτων σε αυτό.

Κάθε συνδρομή ανήκει σε συγκεκριμένη θεματική ενότητα (ή αλλιώς χαρακτηρίζεται από συγκεκριμένο τύπο) και αντιστοιχίζεται με τον κόμβο του δικτύου του οποίου το αναγνωριστικό είναι αυτό που προκύπτει (ή τουλάχιστον είναι αριθμητικά κοντά) από την τιμή που επιστρέφει μια συνάρτηση κατακερματισμού με το όνομα της θεματικής ενότητας ως όρισμα. Όλες λοιπόν οι δημοσιεύσεις και οι συνδρομές που ανήκουν στην ίδια θεματική ενότητα καταλήγουν σε αυτό τον κόμβο «ραντεβού». Στην συνέχεια και με σκοπό να παραδοθεί η δημοσίευση στους τελικούς συνδρομητές, ακολουθείται μία προσέγγιση όμοια με αυτή που παρουσιάστηκε στα πλαίσια του SIENA. Δημιουργούνται έτσι μονοπάτια που αποτελούν μέρη του multicast δέντρου με ρίζα τον κόμβο «ραντεβού» και φύλλα τους συνδρομητές.

Στο Σχήμα 2.9 μπορείτε να δείτε σχηματικά την ανάπτυξη του multicast δέντρου. Σε αυτό το παράδειγμα μπορείτε να δείτε 4 brokers, δύο παραγωγούς πληροφορίας ($P1$ και $P2$), δύο συνδρομητές ($S1$ και $S2$), και

πολυδιάστατο κόμβο και έτσι τελικά η σύζευξη τους με τις συνδρομές επιτυγχάνεται στους κόμβους που διαχειρίζονται ζώνες που περικλείουν αυτά τα σημεία.

Παρόμοια προσέγγιση ακολουθεί και η εργασία [14], που όμως μπορεί να εφαρμοστεί πάνω σε οποιοδήποτε δίκτυο ομοτίμων εταίρων. Επίσης το σύστημα Mercury [17, 18] παρουσιάζει αρκετές ομοιότητες με το Meghdoot αναφορικά με το μοντέλο δημοσιεύσεων/συνδρομών. Στο Mercury [17] υποστηρίζονται συνδρομές με περιορισμούς διαστημάτων τιμών ενώ παράλληλα αντιμετωπίζονται και θέματα σχετικά με την ισοκατανομή του φόρτου στο δίκτυο. Σε σχετική εργασία [18] το Mercury αξιοποιεί το μοντέλο δημοσιεύσεων/συνδρομών για την διαχείριση και την λειτουργία διαδικτυακών παιχνιδιών.

2.5.7 Τα Συστήματα MEDYM και Kyra

Τα συστήματα MEDYM [26] και Kyra [27] βασίζονται και αποτελούν επέκταση της προσέγγισης DADI - Discovery Analysis and Dissemination of Information που παρουσιάστηκε το 2005 [28].

Το σύστημα Kyra [27], αναπτύσσεται πάνω σε ένα δίκτυο από brokers και μπορεί να θεωρηθεί ότι συνδυάζει την προσέγγιση του κόμβου «ραντεβού» με τις filter-based προσεγγίσεις όπως ακριβώς και το Hermes. Η τοπολογία του δικτύου αναπτύσσεται με τέτοιο τρόπο, ώστε γειτονικοί κόμβοι (με βάση μία μετρική απόστασης) να σχηματίζουν ομάδες. Κάθε ομάδα αναλαμβάνει να διαχειρίζεται μία ζώνη στο πεδίο ορισμού των συνδρομών, ενώ ένας από τους κόμβους της ομάδας παίζει το ρόλο του κόμβου «ραντεβού» για την ζώνη της ομάδας. Χρησιμοποιείται στην συνέχεια η filter-based προσέγγιση για την ανάπτυξη ενός δέντρου μέσα σε κάθε ομάδα, που θα αναλάβει να προωθήσει τη δημοσίευση στους κόμβους της και τελικά στους συνδρομητές που έχουν δηλώσει ενδιαφέρον. Το πλεονεκτήματα αυτής της προσέγγισης σε σχέση με μία απλή filter-based δρομολόγηση, είναι η μειωμένη ποσότητα πληροφορίας σχετικά με την δρομολόγηση σε κάθε κόμβο και η σχετικά καλή κατανομή του φόρτου. Τα μειονεκτήματα της έχουν να κάνουν κυρίως με το ότι πρέπει να έχουμε πρότερη γνώση των χαρακτηριστικών των συνδρομών και των δημοσιεύσεων, οι επεξεργασία ιδιοχαρακτηριστικών τύπου συμβολοσειράς είναι περιορισμένη και τέλος δεν αντιμετωπίζεται το θέμα της αυτόνομης οργάνωσης του δικτύου.

Μία διαφορετική προσέγγιση είναι το MEDYM [26], όπου ο κάθε broker στο δίκτυο γνωρίζει για όλους τους υπόλοιπους (το δίκτυο είναι πλήρως συνδεδεμένο) και τις συνδρομές που ο καθένας έχει αποθηκευμένες. Χρησιμοποιείται λοιπόν εδώ η τεχνική του flooding ώστε οι συνδρομές να φτάσουν τελικά στους παραγωγούς της πληροφορίας και να μπορεί έτσι, να επιτευχθεί εκεί η σύζευξη με τις δημοσιεύσεις. Η παράδοση των δημοσιεύσεων πραγματοποιείται με την ανάπτυξη ενός multicast δέντρου (είτε σε επίπεδο εφαρμογής ή σε επίπεδο δικτύου).

2.6 Διαχείριση Πληροφορίας σε Κατανεμημένο Περιβάλλον

Μία ερευνητική περιοχή που παρουσιάζει αρκετές ομοιότητες με αυτή των συστημάτων δημοσιεύσεων/συνδρομών, είναι αυτή της επεξεργασίας των συνεχών ροών δεδομένων (continuous data streams), σε περιβάλλοντα όπως για παράδειγμα τα συστήματα παρακολούθησης της κίνησης σε δίκτυα υπολογιστών, τα δίκτυα αισθητήρων, τα συστήματα ελέγχου οδικής κυκλοφορίας, κ.α. Τα «συνεχή ερωτήματα» (continuous queries) [12] παίζουν βασικό ρόλο σε τέτοια περιβάλλοντα. Εν ολίγοις, ένα continuous query εκδίδεται μία φορά, διευκρινίζοντας τα στοιχεία της ροής δεδομένων που το ενδιαφέρουν (κάτι αντίστοιχο με τις δημοσιεύσεις). Το σύστημα είναι αρμόδιο για την επεξεργασία και αποθήκευση αυτών των continuous queries ώστε να παρακολουθούν συνεχώς τη ροή δεδομένων και να φιλτράρουν μόνο την επιθυμητή πληροφορία. Σε αυτή την ερευνητική περιοχή ανήκει και το PeerCQ [51] το οποίο όμως εκμεταλλεύεται την ετερογένεια των κόμβων όπως επίσης και η εργασία [71] κατά την εξυπηρέτηση ερωτημάτων με διαστήματα τιμών. Στη προσέγγιση που προτείνεται στην εργασία [37], εκμεταλλεύεται η ετερογένεια της πληροφορίας για την αποδοτική επεξεργασία των ερωτημάτων.

Στην ερευνητική περιοχή των κατανεμημένων βάσεων δεδομένων (και των continuous data streams) έχει προταθεί το μοντέλο WP [62, 63] για την έκφραση σύνθετων ερωτημάτων, το οποίο μπορεί να εφαρμοστεί και στα συστήματα δημοσιεύσεων/συνδρομών. Επέκταση του WP μοντέλου, αποτελεί το AWP [62, 96, 97] και στη συνέχεια το AWPS [96] που επιτρέπει την εφαρμογή περιορισμών ομοιότητας σε συστήματα δημοσιεύσεων/συνδρομών βασισμένα στο περιεχόμενο. Στην εργασία [98] παρουσιάζεται μία νέα προσέγγιση (MAPS) που επιτρέπει την πιθανοτική σύζευξη των δημοσιεύσεων με τις συνδρομές, όπου είναι πιθανό για ένα ποσοστό των συζευγνυόμενων συνδρομών να μην παραδίδεται η δημοσίευση στους αντίστοιχους συνδρομητές. Αυτή η σχεδιαστική επιλογή στην προσέγγιση MAPS, όπου η σύζευξη συνδρομών και δημοσιεύσεων δεν διέπεται από αυστηρούς κανόνες, αποσκοπεί στη βελτίωση (μείωση) της κίνησης που παράγεται στο δίκτυο. Στην ίδια εργασία παρουσιάζεται και η εφαρμογή του AWPS μοντέλου στην προσέγγιση DHTric πάνω στο δίκτυο ομοτίμων εταίρων Chord και συγκρίνεται με τη προσέγγιση MAPS επισημαίνοντας τα υπέρ και τα κατά της κάθε μίας.

Στην προσέγγιση PHT προτείνεται μία κατανεμημένη δενδρική δομή πάνω σε ένα δίκτυο ομοτίμων εταίρων, ικανή να υποστηρίξει ερωτήματα με περιορισμούς προθέματος σε συμβολοσειρές και διαστημάτων τιμών σε αριθμητικές τιμές. Η συγκεκριμένη προσέγγιση μπορεί να εφαρμοστεί πάνω σε οποιοδήποτε δίκτυο βασισμένο σε DHT, ωστόσο παρουσιάζει ορισμένα μειονεκτήματα όσον αφορά τον αριθμό των μηνυμάτων κατά την επεξεργασία των ερωτημάτων.

Στο σύστημα pFilter [90] παρουσιάζεται ένα μοντέλο ανάκτησης πληροφορίας πάνω σε δομημένα δίκτυα ομοτίμων εταίρων. Στην εργασία [57] αναπτύσσεται μία δενδρική δομή πάνω σε δίκτυα ομοτίμων εταίρων,

που αξιοποιείται για την επεξεργασία ερωτημάτων ισότητας και διαστημάτων τιμών (range queries). Στο ίδιο μοτίβο κινείται και η προσέγγιση RST [50] όπου παρουσιάζονται ταυτόχρονα και τεχνικές εξισορρόπησης του φόρτου πρόσβασης και αποθήκευση, ενώ παράλληλα η συσχέτιση της δενδρικής δομής με τους κόμβους του δικτύου μπορεί να μεταβληθεί δυναμικά ανάλογα με τα χαρακτηριστικά της πληροφορίας που αποθηκεύεται στο δίκτυο και των ερωτημάτων προς αυτή.

Μία ακόμα εργασία που σχετίζεται με την υποστήριξη σύνθετων ερωτημάτων στο πλαίσιο των συστημάτων δημοσιεύσεων συνδρομών βασισμένα στο περιεχόμενο είναι η [20], όπου αναπτύσσονται μία κατανεμημένη δομή R-Tree. Στην προσέγγιση που παρουσιάζεται στην εργασία [75], παρουσιάζεται μία ιεραρχική δομή για την επεξεργασία των ερωτημάτων που όμως απαιτεί κάποιου είδους προσπάθεια ώστε να εξαχθεί η κατάλληλη κατηγοριοποίηση από την πληροφορία.

Στην εργασία [87] προτείνουν έναν αλγόριθμο αναζήτησης λέξεων κλειδιών στο δίκτυο ομοτίμων εταιρών SkipNet [55]. Το δίκτυο SkipNet ανήκει στην κατηγορία των δομημένων δικτύων ομοτίμων εταιρών και χρησιμοποιεί την κατανεμημένη εφαρμογή της δομής δεδομένων Skip List. Τα κίνητρα των συγγραφέων στην εργασία [87], σχετίζονται με την αναζήτηση συμβολοσειρών (λέξεων κλειδιών) που με τη σειρά τους συνδέονται με συγκεκριμένα αντικείμενα (σε μορφή κειμένου) με τη χρήση ανεστραμμένων πινάκων. Οι τεχνικές που παρουσιάζονται εκεί μπορούν να εφαρμοστούν και στο μοντέλο δημοσιεύσεων/συνδρομών. Συγκεκριμένα, χρησιμοποιούν την τεχνική MLP (Multi-Level Partitioning) για την ταξινόμηση των κόμβων σε μία πολυεπίπεδη ιεραρχία. Σε κάθε επίπεδο υπάρχει ένα σύνολο από κόμβους. Οι ανεστραμμένοι πίνακες διαμοιράζονται στις ομάδες κόμβων με βάση το αντικείμενο (που αντιστοιχεί σε κάθε λέξη), και ανάμεσα στους κόμβους κάθε ομάδας με βάση τη λέξη κλειδί. Η τεχνική αυτή αποσκοπεί στην μείωση της κίνησης στο δίκτυο και του αριθμού των μηνυμάτων που απαιτούνται, αποφεύγοντας την διασπορά των αιτήσεων αναζήτησης σε ολόκληρο το δίκτυο.

Στην εργασία [60] παρουσιάζεται μία επέκταση των φίλτρων Bloom [22], που σκοπό έχει την αποδοτική εξυπηρέτηση ερωτημάτων σε ένα δίκτυο ομοτίμων εταιρών που επεξεργάζεται την πληροφορία υπό μορφή XML αντικειμένων (τα οποία γενικά μπορούν να χρησιμοποιηθούν για την περιγραφή της πληροφορίας ή των υπηρεσιών που προσφέρει κάθε κόμβος). Παράλληλα, αξιοποιείται η πληροφορία σχετικά με τα αντικείμενα που έχει αποθηκευμένα κάθε κόμβος, ώστε αυτοί να οργανωθούν ιεραρχικά σε ομάδες με βάση την ομοιότητα που παρουσιάζουν τα αποθηκευμένα σε αυτούς αντικείμενα. Στην εργασία [61] παρουσιάζεται μία εκτενής μελέτη γνωστών προσεγγίσεων που σχετίζονται με την διαχείριση πληροφορίας δομημένης υπό μορφή XML σε δομημένα και μη δομημένα δίκτυα ομοτίμων εταιρών. Πρόσφατες εργασίες στο χώρο της διαχείρισης XML αντικειμένων σε κατανεμημένο περιβάλλον, περιλαμβάνουν προσεγγίσεις όπου οι δημοσιεύσεις είναι XML αντικείμενα, ενώ οι συνδρομές εκφράζονται με XPath ή XQuery (YFilter [43],

XFilter [10], και Xtrie [34]).

Στην εργασία [78] παρουσιάζονται τεχνικές εξισορρόπησης τους φόρτου στο δίκτυο κατά την επεξεργασία ερωτημάτων με περιορισμούς διαστημάτων τιμών, αξιοποιώντας κατάλληλα τεχνικές που εκμεταλλεύονται την ύπαρξη και διατήρηση πολλαπλών αντιγράφων της πληροφορίας στο δίκτυο. Προς την κατεύθυνση της εξισορρόπησης του φόρτου στο δίκτυο κινείται και η εργασία [79].

Κατά καιρούς έχουν προταθεί προσεγγίσεις που συνδυάζουν το μοντέλου δημοσιεύσεων/συνδρομών με βάση τις θεματικές ενότητες (topic-based) και με βάση το περιεχόμενο (content-based). Οι συγγραφείς στην [89] χρησιμοποιούν ένα σύστημα δημοσιεύσεων/συνδρομών βασισμένο σε θεματικές ενότητες (Scribe [33]) που εφαρμόζεται πάνω στο δίκτυο ομοτίμων εταίρων Pastry [85]. Στην προσέγγισή τους οι δημοσιεύσεις και οι συνδρομές ταξινομούνται αυτόματα σε θεματικές ενότητες με βάση το περιεχόμενό τους, χρησιμοποιώντας ένα κατάλληλο μοντέλο (εξαρτώμενο από την εφαρμογή που τελικά θα χρησιμοποιηθεί στο σύστημα). Ένα πιθανό μειονέκτημα αυτής της προσέγγισης έχει να κάνει με τον κατάλληλο σχεδιασμό του μοντέλου αυτόματης κατάταξης των συνδρομών και των δημοσιεύσεων σε θεματικές ενότητες, μιας και αυτό παίζει σημαντικό ρόλο. Επίσης, μειονέκτημα αποτελεί και η ύπαρξη false positives, όπου δημοσιεύσεις θεωρείται ότι συζευγνύουν με ένα υποσύνολο των συνδρομών, κάτι που όμως στη πραγματικότητα δεν ισχύει. Προς την ίδια κατεύθυνση (του συνδυασμού του μοντέλου δημοσιεύσεων/συνδρομών με βάση τις θεματικές ενότητες και το περιεχόμενο) κινείται και η προσέγγιση που παρουσιάζεται σε σχετική εργασία [77].

Μία συνήθης τακτική για την σύζευξη των δημοσιεύσεων και των συνδρομών σε ένα πλήρως καταναμεμένο περιβάλλον με βάση το περιεχόμενό τους, είναι η ανάπτυξη multicast δέντρων. Σε σχετικές εργασίες που κατά καιρούς έχουν δημοσιευθεί σε αυτό το τομέα [15, 26, 30, 32, 40, 55, 76, 90, 91, 94], οι συνδρομές ταξιδεύουν σε ολόκληρο το δίκτυο, σχηματίζοντας multicast δέντρα, που στη συνέχεια αξιοποιούν οι δημοσιεύσεις ώστε να παραδοθούν στους συνδρομητές. Κατά συνέπεια, και εφόσον στην πλειονότητα αυτών των προσεγγίσεων χρησιμοποιείται μία flooding προσέγγιση για την διασπορά των συνδρομών στο δίκτυο, απαιτείται η αποστολή $O(N)$ μηνυμάτων σε ένα δίκτυο αποτελούμενο από N κόμβους. Κατά καιρούς έχουν προταθεί εναλλακτικές τεχνικές που στοχεύουν στη μείωση του αριθμού των μηνυμάτων και του όγκου της πληροφορίας που διακινείται στο δίκτυο, με την εφαρμογή «περιλήψεων» των συνδρομών [93, 94, 100].

3 Το Μοντέλο

3.1 Περιγραφή του Μοντέλου Διασύνδεσης Χρηστών

Η υποδομή ενός συστήματος δημοσιεύσεων/συνδρομών αποτελείται από τρία βασικά μέρη :

- Μία ή περισσότερες γεννήτριες μηνυμάτων που διοχετεύουν μηνύματα/δημοσιεύσεις στο σύστημα (που αποκαλούνται παραγωγοί, πάροχοι δημοσιεύσεων - producers).
- Ένα σύνολο από διαμεσολαβητές (brokers) που σκοπό έχουν να διοχετεύουν τα μηνύματα/δημοσιεύσεις στους κατάλληλους μόνο χρήστες.
- Έναν ή περισσότερους συνδρομητές (subscribers) που δηλώνουν τα ενδιαφέροντα τους για συγκεκριμένα μηνύματα/δημοσιεύσεις με κατάλληλη διατύπωση συνδρομών, και στη συνέχεια δέχονται μόνο αυτά τα μηνύματα που συζηγνύουν με τις συνδρομές τους.

Η αρχιτεκτονική ενός τυπικού δικτύου που υποστηρίζει τον μοντέλο δημοσιεύσεων/ συνδρομών, αποτελείται από κόμβους διαμεσολαβητές (brokers) και τους χρήστες που μπορεί να είναι είτε συνδρομητές είτε παραγωγοί δημοσιεύσεων (Σχήμα 3.1).

Οι διαμεσολαβητές είναι δυνατόν είτε να σχηματίζουν ένα καλά συνδεδεμένο δίκτυο στο οποίο συνδέονται οι χρήστες (Non-P2P προσεγγίσεις), ή να αποτελούν μέρος ενός δικτύου ομοτίμων εταιρών (P2P προσεγγίσεις). Στην τελευταία περίπτωση, ο ρόλος των χρηστών (που μπορεί να διοχετεύουν συνδρομές ή δημοσιεύσεις στο σύστημα) και των διαμεσολαβητών είναι αλληλένδετος, με την έννοια ότι ένα κόμβος του δικτύου, μπορεί να παίζει και τους τρεις ρόλους ταυτόχρονα (παραγωγός-συνδρομητής-διαμεσολαβητής).

Στη παρούσα διατριβή υιοθετούμε την P2P προσέγγιση, όπου οι διαμεσολαβητές σχηματίζουν ένα δομημένο δίκτυο ομοτίμων εταιρών βασισμένο σε κατανεμημένο πίνακα κατακερματισμού. Οι χρήστες συνδέονται στο δίκτυο και διοχετεύουν συνδρομές ή δημοσιεύσεις.

Σε ένα κατανεμημένο σύστημα δημοσιεύσεων/συνδρομών όπως αυτό, υπάρχουν δύο βασικές εργασίες που πρέπει να εκτελεστούν. Η πρώτη είναι η σύζευξη των μηνυμάτων ή δημοσιεύσεων με τις συνδρομές και η δεύτερη είναι η παράδοση των δημοσιεύσεων στους τελικούς χρήστες που έχουν δηλώσει το ανάλογο

Σχήμα 3.1: Το μοντέλο δικτύου σε ένα σύστημα δημοσιεύσεων/συνδρομών. Οι χρήστες συνδέονται με τους brokers οι οποίοι σχηματίζουν ένα δίκτυο ομοτίμων εταίρων βασισμένο σε DHT.

ενδιαφέρον. Στα πλαίσια αυτής της διατριβής επικεντρωνόμαστε στην πρώτη εργασία, ενώ για την δεύτερη μπορούμε να ανατρέξουμε στη σχετική βιβλιογραφία [15, 31, 32, 40, 76].

Η σύζευξη των δημοσιεύσεων με τις συνδρομές γίνεται σε δύο βήματα ακολουθώντας το μοντέλο του κόμβου «ραντεβού». Πρώτα από όλα, οι συνδρομές επεξεργάζονται κατάλληλα και το αναγνωριστικό τους (το οποίο περιλαμβάνει χρήσιμη πληροφορία που θα παρουσιαστεί παρακάτω) αποθηκεύεται σε προσεκτικά επιλεγμένους κόμβους του δικτύου. Αυτή η διαδικασία θα αναφέρεται από εδώ και στο εξής ως **επεξεργασία συνδρομών**. Στη συνέχεια, όταν κάποια δημοσίευση εισέλθει στο δίκτυο, προωθείται κατάλληλα στον κόμβο αυτό που είναι πιθανό να βρίσκονται αποθηκευμένα τα αναγνωριστικά των συνδρομών που συζευγνύουν με τη δημοσίευση, όπου και πραγματοποιείται η σύζευξη. Η διαδικασία αυτή θα αναφέρεται από εδώ και στο εξής ως **επεξεργασία δημοσιεύσεων**. Έχοντας λοιπόν τη λίστα των αναγνωριστικών μπορούμε να τα επεξεργαστούμε περαιτέρω και να ενημερώσουμε τους συνδρομητές που έχουν διατυπώσει αυτές τις συνδρομές.

Για το υπόλοιπο του κειμένου, όταν παρουσιάζονται οι εκφράσεις αποθήκευση συνδρομής ή αποθήκευση αναγνωριστικού συνδρομής, ας θεωρηθεί ότι αναφερόμαστε στην ίδια ακριβώς διαδικασία, την αποθήκευση του αναγνωριστικού της συνδρομής.

3.2 Περιγραφή του Μοντέλου Δημοσιεύσεων/Συνδρομών

Οι δημοσιεύσεις σε ένα σύστημα δημοσιεύσεων/συνδρομών βασισμένο στο περιεχόμενο μπορούν να εκφραστούν με διάφορα μοντέλα. Κατά καιρούς έχουν χρησιμοποιηθεί σε ανάλογες εργασίες: (i) το μοντέλο βασισμένο σε συστοιχίες δεδομένων (Tuple-based), όπου η δημοσίευση αποτελείται από ένα σύνολο συμβολοσειρών, κάτι αντίστοιχο με τις εγγραφές σε μία βάση δεδομένων, (ii) το μοντέλο βασισμένο σε αντικείμενα (Object-based) που δανείζεται την έννοια των αντικειμένων που μπορεί κανείς να συναντήσει στις σύγχρονες γλώσσες προγραμματισμού και (iii) το μοντέλο βασισμένο σε ιδιοχαρακτηριστικά (Attribute-based) όπου μία δημοσίευση αποτελείται από ένα σύνολο ζευγών ιδιοχαρακτηριστικού-τιμής.

Στη παρούσα διατριβή θα υιοθετήσουμε το Attribute-based μοντέλο ([30]). Έτσι τόσο οι δημοσιεύσεις όσο και οι συνδρομές αποτελούνται από ένα υποσύνολο του συνόλου των A διαθέσιμων ιδιοχαρακτηριστικών που υπάρχουν στο σύστημα. Κάθε ιδιοχαρακτηριστικό a_i (με $1 \leq i \leq A$), έχει ένα συγκεκριμένο όνομα, τύπο και τιμή $v(a_i)$. Ένα μήνυμα ορίζεται σαν ένα σύνολο από k ζεύγη <ιδιοχαρακτηριστικό, τιμή> (με $k \leq A$).

Κάθε συνδρομή ορίζεται με ένα υποσύνολο από ιδιοχαρακτηριστικά (επιλεγμένα από το ίδιο σύνολο ιδιοχαρακτηριστικών που χρησιμοποιήθηκαν για τη διατύπωση των μηνυμάτων) των οποίων οι τιμές είναι στην ουσία περιορισμοί (constraints) ή τελεστές σε τιμές που εκφράζουν τα ενδιαφέροντα του συνδρομητή. Εάν ο τύπος του ιδιοχαρακτηριστικού είναι συμβολοσειρά, τότε οι επιτρεπτοί περιορισμοί είναι: (i) πρόθεμα ή prefix όπως για παράδειγμα το ιδιοχαρακτηριστικό IP στη συνδρομή Sub1 στο παρακάτω παράδειγμα, (ii) επίθεμα ή suffix όπως για παράδειγμα το ιδιοχαρακτηριστικό MB στη συνδρομή Sub1, (iii) υποσυμβολοσειρά ή substring όπως για παράδειγμα το ιδιοχαρακτηριστικό $OSDISTR$ στη συνδρομή Sub3, και (iv) ισότητα όπως για παράδειγμα το ιδιοχαρακτηριστικό OS στη συνδρομή Sub2.

Για ιδιοχαρακτηριστικά αριθμητικού τύπου, οι επιτρεπτοί τελεστές ή περιορισμοί στις τιμές τους μπορεί να είναι η ισότητα ή οι τελεστές $\leq, \geq, \neq$ που στη γενική περίπτωση μπορούν να εκφραστούν σαν ένα οποιοδήποτε διάστημα τιμών ($v(a_i)=[l, h]$) που αποτελεί μέρος του μέγιστου δυνατού διαστήματος τιμών που μπορεί να υποστηρίξει ένα ιδιοχαρακτηριστικό και συμβολίζεται με D_{ai} (Domain ή πεδίο ορισμού). Το πεδίο ορισμού ενός ιδιοχαρακτηριστικού a_i , ορίζεται από το κάτω και άνω όριο του $D_{ai} = [S_{ai}, E_{ai}]$ και την ακρίβεια του $pr(a_i)$ που χρησιμοποιείται για τον ορισμό πεπερασμένου πλήθους τιμών ανά διάστημα¹.

Η αναγκαία συνθήκη για τη σύζευξη μίας συνδρομής με μία δημοσίευση είναι

1. όλα τα ιδιοχαρακτηριστικά που έχουν χρησιμοποιηθεί στην συνδρομή να εμφανίζονται στη δημοσίευση

¹ Ας θεωρηθεί ότι από εδώ και στο εξής αναφερόμαστε σε αριθμητικού τύπου ιδιοχαρακτηριστικά που αποτελούνται από ακέραιους αριθμούς. Η γενίκευση του μοντέλου ώστε να μπορούν να χρησιμοποιηθούν ιδιοχαρακτηριστικά με πραγματικούς αριθμούς μπορεί να γίνει σχετικά εύκολα.

2. οι περιορισμοί των ιδιοχαρακτηριστικών της συνδρομής να ικανοποιούνται από τις τιμές των αντίστοιχων ιδιοχαρακτηριστικών της δημοσίευσης

Στο παράδειγμα που ακολουθεί μόνο η συνδρομή Sub2, συζευγνύει με τη δημοσίευση Event1, μιας και ικανοποιούνται οι περιορισμοί της συνδρομής από τις τιμές της.

$$\text{Sub1} = \{\text{SubID}_1:: \text{MB} : '*\text{GSR}', \text{IP} : '150.140.*'\}$$

$$\text{Sub2} = \{\text{SubID}_2:: \text{OS} : '\text{Linux}', \text{CPU} : [1600, 2400]\}$$

$$\text{Sub3} = \{\text{SubID}_3:: \text{OSDISTR} : '*\text{-BSD*}', \text{CPU} : [2000, 2400]\}$$

$$\text{Event1} = \{\text{OS} : '\text{Linux}', \text{CPU} : 2000\}$$

Πέρα από την υποστήριξη των τελεστών prefix και suffix για ιδιοχαρακτηριστικά τύπου συμβολοσειράς, θα πρέπει να σημειωθεί ότι μπορούμε να υποστηρίξουμε και τον τελεστή containment, για παράδειγμα 'A*C' που μπορεί πολύ εύκολα να θεωρηθεί σαν ένας διπλός περιορισμός αποτελούμενος από prefix 'A*' και suffix '*C' τελεστές

Κάθε συνδρομή στο μοντέλο που παρουσιάζουμε, συνοδεύεται και από ένα αναγνωριστικό συνδρομής, *SubID*, που αποτελείται από τρία μέρη: Το πρώτο μέρος c_1 , είναι στην ουσία το αναγνωριστικό του κόμβου στον οποίο είναι συνδεδεμένος ο χρήστης που διατύπωσε τη συγκεκριμένη συνδρομή. Αυτός ο κόμβος συχνά θα αποκαλείται *κόμβος πηγή* ή *αφετηρία* της συνδρομής, και είναι επιφορτισμένος με το να διατηρεί πληροφορία σχετικά με τη συνδρομή, όπως για παράδειγμα την διεύθυνση IP του χρήστη που διατύπωσε τη συνδρομή κ.α. Το δεύτερο μέρος c_2 , αποτελεί ένα αναγνωριστικό της συνδρομής σε εμβέλεια κόμβου και σκοπό έχει να μπορεί να εντοπιστεί η συνδρομή ανάμεσα στις αποθηκευμένες συνδρομές σε ένα κόμβο. Το τρίτο μέρος c_3 είναι ο αριθμός των ιδιοχαρακτηριστικών που έχουν οριστεί στη συνδρομή και χρησιμοποιείται από τον αλγόριθμο σύζευξης, όπως θα δούμε παρακάτω. Ανάλογα με τον αλγόριθμο σύζευξης, μπορεί να είναι αναγκαίο να προσθέσουμε επιπλέον πεδία στον αναγνωριστικό της συνδρομής. Ωστόσο αυτό θα αναφερθεί ρητά καθώς και θα εξηγηθεί η χρήση αυτών των επιπλέον πεδίων στα αντίστοιχα κεφάλαια.

Σε ένα σύστημα δημοσιεύσεων/συνδρομών η βασική λειτουργία που πρέπει να επιτευχθεί, είναι η σύζευξη των δημοσιεύσεων με τις συνδρομές και στη συνέχεια η ενημέρωση των κατάλληλων συνδρομητών. Οι βασικές λειτουργίες που χρησιμοποιούνται στη κατανεμημένη σύζευξη των συνδρομών με τις δημοσιεύσεις στην υποδομή που προτείνουμε είναι:

1. Τα αναγνωριστικά των συνδρομών (*SubIDs*) αποθηκεύονται σε κατάλληλους κόμβους τους δικτύου (*επεξεργασία συνδρομών-subscription processing*) οι οποίοι έχουν επιλεγεί εφαρμόζοντας σχετικούς με αυτό το βήμα αλγόριθμους.

2. Κατά την άφιξη μίας δημοσίευσης στο σύστημά, αυτοί οι κόμβοι που έχουν αποθηκευμένα αναγνωριστικά συνδρομών που συζευγνύουν με το μήνυμα εντοπίζονται κατάλληλα και τα αποθηκευμένα αναγνωριστικά (*SubIDs*) συλλέγονται (*επεξεργασία δημοσιεύσεων-event processing*).
3. Σε κάθε κόμβο, τα *SubIDs* αποθηκεύονται σε 4 λίστες σχετικές με κάθε ιδιοχαρακτηριστικό του μοντέλου μας, με βάση τον τύπο του τελεστή που έχει χρησιμοποιηθεί για κάθε ιδιοχαρακτηριστικό: για το ιδιοχαρακτηριστικό a_i , η λίστα $L_{ai-pref}$ έχει αποθηκευμένα *SubIDs* συνδρομών με περιορισμούς τύπου προθέματος (prefix) στην τιμή του ιδιοχαρακτηριστικού a_i , η λίστα $L_{ai-suff}$ είναι αφιερωμένη στους περιορισμούς τύπου επιθέματος (suffix), η λίστα L_{ai-eq} είναι αφιερωμένη στους περιορισμούς τύπου ισότητας, και τέλος η λίστα L_{ai-num} είναι αφιερωμένη στους περιορισμούς που εφαρμόζονται σε αριθμητικές τιμές (π.χ. διαστήματα τιμών, ισότητες κ.α.).

Συνήθως, ο εντοπισμός ενός κόμβου στον οποίο πρόκειται να αποθηκευτεί το αναγνωριστικό *SubID* μιας συνδρομής επιτυγχάνεται με την κλήση της συνάρτησεως κατακερματισμού που προσφέρει το ίδιο το DHT, αφού χρησιμοποιήσουμε (στη γενική περίπτωση) την τιμή (συμβολοσειρά ή αριθμητική τιμή) που ορίζεται στα ιδιοχαρακτηριστικά μίας συνδρομής, ως όρισμα. Η τιμή που επιστρέφει η συνάρτηση κατακερματισμού ($DHT.hash()$) είναι ένα αναγνωριστικό κόμβου (nodeID) το οποίο ανήκει στον χώρο διευθύνσεων αναγνωριστικών των κόμβων. Η έξοδος αυτής της συνάρτησης ακολουθεί μία ομοιόμορφη κατανομή στο χώρο διευθύνσεων των αναγνωριστικών των κόμβων.

Έτσι, εάν το αποτέλεσμα της κλήσης της συνάρτησης κατακερματισμού είναι n , για την τιμή $v(a_i)$ του ιδιοχαρακτηριστικού a_i (δηλ. $n=DHT.hash(v(a_i))$), τότε η τιμή n είναι ομοιόμορφα κατανεμημένη στον χώρο διευθύνσεων των αναγνωριστικών των κόμβων. Στη γενική περίπτωση λοιπόν, το *SubID* της συνδρομής θα πρέπει να αποθηκευτεί στον κόμβο με αναγνωριστικό ίσο με n . Εάν αυτός ο κόμβος δεν υπάρχει ακόμα στο δίκτυο, τότε το ίδιο το DHT εγγυάται ότι η αίτηση αποθήκευσης θα προωθηθεί στον κόμβο του δικτύου του οποίου το αναγνωριστικό είναι αριθμητικά πιο κοντά στο n (από τις διεπαφές για παράδειγμα που προσφέρει το Chord αυτός ο κόμβος είναι ο $successor(n)$).

4 *Pyracanthus*: Σύστημα Δημοσιεύσεων/Συνδρομών Ανεξάρτητο του Υποκείμενου DHT Δικτύου

Όπως έχει ήδη αναφερθεί οι DHT-independent λύσεις είναι ιδιαίτερα επιθυμητές, επειδή ακριβώς μπορούν να εφαρμοστούν πάνω από οποιοδήποτε δίκτυο ομοτίμων εταίρων βασισμένο σε DHT. Επιπλέον, η ανάπτυξη κατανεμημένων συστημάτων διαχείρισης της πληροφορίας στα πρότυπα του μοντέλου δημοσιεύσεων συνδρομών, είναι μία σχετικά εύκολη υπόθεση αφού η ενσωμάτωση του μοντέλου στο DHT δίκτυο απαιτεί την χρήση μόνο των επαφών που προσφέρει το ίδιο το DHT (συγκεκριμένα την συνάρτηση *lookup()* που χρησιμοποιείται για τον εντοπισμό των κόμβων και τη δρομολόγηση σε αυτούς μηνυμάτων με βάση το αναγνωριστικό τους).

Από πλευρά σχεδιασμού, το να προσφέρει κανείς τη δυνατότητα διατύπωσης σύνθετων ερωτημάτων τόσο σε αριθμητικές τιμές όσο και σε συμβολοσειρές, είναι μία πρόκληση δεδομένου ότι πρέπει να χρησιμοποιήσει μόνο τις διεπαφές που προσφέρει το DHT, ενώ παράλληλα η λύση που θα προσφέρει θα πρέπει να είναι όσο το δυνατό πιο αποδοτική.

Προς αυτή την κατεύθυνση λοιπόν, παρουσιάζουμε σ' αυτή την ενότητα την υποδομή *Pyracanthus*¹ η οποία θέλουμε να πιστεύουμε ότι αποτελεί την πρώτη δουλειά που είναι ικανή να προσφέρει ικανοποιητική από πλευράς απόδοσης, υποστήριξη σύνθετων ερωτημάτων σε αριθμητικές τιμές και συμβολοσειρές. Επιπλέον στην *Pyracanthus* παρουσιάζονται τόσο stateless όσο και stateful προσεγγίσεις, με χαρακτηριστικά που δείχνουν κάτω από ποιες προϋποθέσεις είναι προτιμότερο να χρησιμοποιηθούν.

Συγκεκριμένα, η υποδομή *Pyracanthus* είναι εφοδιασμένη με τρεις διαφορετικές προσεγγίσεις για την επεξεργασία συνδρομών με περιορισμούς σε αριθμητικές τιμές. Από αυτές οι δύο είναι stateless λύσεις ενώ η τρίτη είναι stateful. Για την επεξεργασία συνδρομών με περιορισμούς πάνω σε συμβολοσειρές, προτείνουμε δύο προσεγγίσεις, μία stateless και μία stateful.

¹ Ο Πυράκανθος (*Pyracanthus*) είναι ένας όμορφος αειθαλής θάμνος, του οποίου η πυκνή δομή και τα ισχυρά αγκάθια τον καθιστούν ένα τέλειο «φίλτρο». Συνήθως, χρησιμοποιείται για τη διακόσμηση φρακτών, όπου τα όμορφα άνθη του προσελκύουν διάφορα είδη άγριας ζωής, από τα έντομα μέχρι όμορφα πουλιά.

4.1 Επεξεργασία Συνδρομών με Ιδιοχαρακτηριστικά Αριθμητικού Τύπου

Σε αυτή την ενότητα παρουσιάζουμε τρεις διαφορετικές μεθόδους επεξεργασίας συνδρομών με περιορισμούς σε αριθμητικές τιμές. Παρουσιάζουμε τόσο stateless προσεγγίσεις, όπου δεν απαιτείται επιπρόσθετη διατήρηση πληροφορίας σχετική με τη δρομολόγηση σε κάθε κόμβο (εκτός βέβαια από αυτή που απαιτείται από το ίδιο το DHT), και μία stateful.

Σχετικά με τις stateless προσεγγίσεις, παρουσιάζουμε δύο διαφορετικές λύσεις που παρουσιάζουν ενδιαφέροντα χαρακτηριστικά: (i) την προσέγγιση *OP-DHT* η οποία στηρίζεται στη χρήση μίας συνάρτησης κατακερματισμού με χαρακτηριστικά διατήρησης τάξης (αντί της χρήσης της συνήθους συνάρτησης που προσφέρει κάθε DHT), που χρησιμοποιείται για την αποθήκευση των αντικειμένων (στην περίπτωση μας των αναγνωριστικών - *SubID*- των συνδρομών) και (ii) την προσέγγιση *virtual Range Tree Forest*, που στηρίζεται στη χρήση ενός δάσους από συγκεκριμένες δεντρικές δομές (παρόμοιες με το *segment tree* [42] και το *Range Search Tree* [50]) με σκοπό να αντιστοιχιστούν κατάλληλα τα διαστήματα αριθμητικών τιμών με τα αναγνωριστικά των κόμβων.

Επιπλέον, παρουσιάζουμε την stateful παραλλαγή της προσέγγισης *Range Tree Forest*, όπου η διαφορά της με την *virtual Range Tree Forest* είναι ότι πλέον η δεντρική δομή κτίζεται πάνω στο δίκτυο, με τους κόμβους αυτού να παίζουν το ρόλο των κόμβων των δέντρων, με σκοπό την βελτίωση της απόδοσης κατά την επεξεργασία των δημοσιεύσεων.

4.1.1 Stateless Προσεγγίσεις

Συνάρτηση Κατακερματισμού με Διατήρηση Τάξης *OP-DHT*

Υποθέστε ότι για ιδιοχαρακτηριστικό a_i μίας συνδρομής ορίζεται το εξής διάστημα τιμών: $v(a_i) = R = [l, h]$. Θεωρούμε ότι οι τιμές μεταξύ του l και του h είναι πεπερασμένες. Αυτό ισχύει όταν το συγκεκριμένο ιδιοχαρακτηριστικό υποστηρίζει ακέραιους αριθμούς. Ωστόσο, μπορούμε να καταλήξουμε σε πεπερασμένο πλήθος τιμών ανά διάστημα πραγματικών αριθμών, εάν ορίσουμε κατάλληλα την ακρίβεια $pr(a_i)$ του πεδίου ορισμού $D_{ai} = [S_{ai}, E_{ai}]$ του ιδιοχαρακτηριστικού. Σε αυτή την περίπτωση, το πλήθος των πεπερασμένων τιμών ενός διαστήματος πραγματικών τιμών για το ιδιοχαρακτηριστικό a_i είναι $|R| = \frac{h-l}{pr(a_i)}$.

Κατά την επεξεργασία ενός διαστήματος τιμών, θα πρέπει να αποθηκεύσουμε κατάλληλα στο δίκτυο το *SubID* για κάθε μια από τις $|R|$ τιμές, και άρα θα πρέπει να εκτελέσουμε $|R|$ βήματα. Μία απλή προσέγγιση, θα ήταν να χρησιμοποιήσουμε την συνάρτηση κατακερματισμού του DHT, και για κάθε μια από τις τιμές του διαστήματος, να χρησιμοποιήσουμε την τιμή σαν όρισμα στην συνάρτηση κατακερματισμού, να

εντοπίσουμε τον κόμβο με το αντίστοιχο αναγνωριστικό (που είναι ίσο με την τιμή που θα μας επέστρεφε η συνάρτηση) και να αποθηκεύσουμε εκεί το *SubID* της συνδρομής. Στη συνέχεια, όταν καταφθάσει μία δημοσίευση με αριθμητική τιμή στο ίδιο ιδιοχαρακτηριστικό, που συμπεριλαμβάνεται στο διάστημα τιμών της συνδρομής, θα μπορούσαμε εύκολα να εντοπίσουμε ένα από τα αποθηκευμένα *SubIDs* χρησιμοποιώντας με τον ίδιο τρόπο την συνάρτηση κατακερματισμού.

Είναι εύκολο να καταλάβει κανείς, ότι σε αυτή τη μέθοδο, η επεξεργασία της συνδρομής με περιορισμό διαστήματος R , θα απαιτεί την εκτέλεση $|R| \times O(\log(N))$ βημάτων, επειδή απαιτούνται $|R|$ lookups πάνω στο DHT. Αυτή η απόδοση μπορεί να θεωρηθεί ανεπίτρεπτη, εάν για παράδειγμα τα διαστήματα τιμών είναι πολύ μεγάλα ή/και η ακρίβεια που έχει οριστεί σε ένα ιδιοχαρακτηριστικό είναι υψηλή (δηλ. ο αριθμός των πεπερασμένων τιμών ανά διάστημα είναι πολύ μεγάλος).

Μπορούμε να βελτιώσουμε την απόδοση κατά την επεξεργασία των συνδρομών με το να επεκτείνουμε την λειτουργικότητα του DHT και να εφαρμόσουμε μία συνάρτηση κατακερματισμού με διατήρηση τάξης (OPHF) αντί της συνήθους συνάρτησης κατακερματισμού. Το κίνητρο μας προς αυτή την επιλογή, είναι η αντιστοίχιση διαδοχικών αριθμητικών τιμών με κόμβους του δικτύου που καταλαμβάνουν διαδοχικές θέσεις στο χώρο διευθύνσεων από όπου επιλέγονται τα αναγνωριστικά τους. Έτσι, και δεδομένου ότι τα περισσότερα DHTs διατηρούν στους πίνακες δρομολόγησης τους δείκτες προς τους αριθμητικά πιο κοντά σε αυτούς κόμβους (με βάση το αναγνωριστικό τους), μπορούμε να βελτιώσουμε δραματικά την απόδοση αφού πλέον δεν πρέπει να εκτελούμε $O(\log(N))$ βήματα για κάθε τιμή αλλά μόνο ένα (εκμεταλλευόμενοι τους δείκτες προς τους αριθμητικά επόμενους κόμβους). Για λόγους παρουσίασης, ας χρησιμοποιήσουμε το DHT Chord σαν το δίκτυο πάνω στο οποίο θα κτίσουμε την υποδομή *Pyracanthus* με χώρο διευθύνσεων αποτελούμενο από 2^m διαφορετικές τιμές που σχηματίζουν το «δακτυλίδι» του Chord.

OP-DHT στο Δίκτυο Ομοτίμων Εταίρων Chord

Θα χρησιμοποιήσουμε μια συνάρτηση κατακερματισμού με διατήρηση τάξης και χώρο διευθύνσεων αποτελούμενο από 2^m τιμές, ώστε να αντιστοιχίσουμε διαδοχικές αριθμητικές τιμές σε διαδοχικούς, με βάση το αναγνωριστικό τους, κόμβους στο «δακτυλίδι» του Chord. Έχοντας στη διάθεση μας μία τέτοια συνάρτηση και δεδομένου ότι πρέπει να επεξεργαστούμε κατάλληλα ένα διάστημα τιμών $R = [l, h]$ που ορίζεται στο ιδιοχαρακτηριστικό μίας συνδρομής, θα πρέπει να εκτελέσουμε $O(\log(N))$ βήματα για να εντοπίσουμε τον πρώτο κόμβο όπου θα αποθηκευτεί το *SubID* (στην λίστα L_{a-num}) για την μικρότερη τιμή l , του διαστήματος R . Στη συνέχεια θα πρέπει να εκτελέσουμε $|R|$ βήματα και να αποθηκεύσουμε το *SubID* για τις υπόλοιπες τιμές του διαστήματος. Με αυτή τη προσέγγιση, θα πρέπει να εκτελέσουμε συνολικά $|R| + O(\log(N))$ βήματα, μιας και διαδοχικοί κόμβοι στο χώρο διευθύνσεων των αναγνωριστικών των κόμβων, είναι άμεσα συνδεδεμένοι μεταξύ τους και έτσι απαιτείται ένα μόνο βήμα για να κινηθούμε μεταξύ

Σχήμα 4.1: Επεξεργασία συνδρομών με περιορισμό διαστήματος τιμών και επεξεργασία δημοσίευσης με τιμή που ανήκει στο διάστημα τιμών, με την προσέγγιση *OP-DHT*.

αυτών (αυτή η ιδιότητα παρέχεται από τα περισσότερα γνωστά DHTs όπως για παράδειγμα το Chord [88], το Pastry [85], το Bamboo [84] κ.α.). Στη συνέχεια, όταν καταφθάσει μία δημοσίευση με τιμή στο ίδιο ιδιοχαρακτηριστικό που ανήκει στο διάστημα τιμών R , θα χρησιμοποιήσουμε την ίδια συνάρτηση κατακερματισμού και τελικά θα εντοπίσουμε ένα από τους κόμβους που έχει ήδη αποθηκευμένο το *SubID* της συνδρομής. Στο Σχήμα 4.1 μπορεί να δει κανείς σχηματικά την διαδικασία επεξεργασία μίας συνδρομής με περιορισμό διαστήματος τιμών $[l, h]$ και την επεξεργασία μίας δημοσίευσης με τιμή που ανήκει στο διάστημα τιμών $[l, h]$.

Εάν v είναι μία οποιαδήποτε τιμή στο διάστημα $[l, h]$, και το πεδίο ορισμού του ιδιοχαρακτηριστικού a_i είναι $D_{ai} = [S_{ai}, L_{ai}]$, τότε η συνάρτηση κατακερματισμού με διατήρηση τάξης (OPHF) ορίζεται ως εξής:

$$\text{hash}(v) = (s_o(a_i) + \frac{v - S_{ai}}{L_{ai} - S_{ai}} \times 2^m) \bmod 2^m$$

$$s_o(a_i) = \text{DHT.hash}(a_i)$$

Η σταθερά $s_o(a_i)$ χρησιμοποιείται για επιλέξουμε τυχαία τον κόμβο που θα είναι υπεύθυνος για τη μικρότερη τιμή του πεδίου ορισμού ενός ιδιοχαρακτηριστικού, a_i , ώστε διαφορετικά ιδιοχαρακτηριστικά με το ίδιο πεδίο ορισμού να αντιστοιχίζονται σε διαφορετικές ομάδες κόμβων του δικτύου (ή διαφορετικά τόξα στο κυκλικό χώρο διευθύνσεων του Chord, όπως μπορείτε να δείτε στο Σχήμα 4.1, όπου φαίνονται τα τρία διαφορετικά τόξα από κόμβους στους οποίους αντιστοιχίζονται οι τιμές τριών ιδιοχαρακτηριστικών: a_1 , a_2 και a_3). Η $\text{DHT.hash}()$ είναι η συνάρτηση κατακερματισμού που παρέχει το Chord (για παράδειγμα η SHA-1).

Το ακριβές πλήθος βημάτων που απαιτείται για την επεξεργασία ενός διαστήματος τιμών R με πλήθος τιμών $|R|$ που ορίζεται σε ένα ιδιοχαρακτηριστικό a_i , εξαρτάται από το μέγεθος του πεδίου ορισμού του ιδιοχαρακτηριστικού D_{ai} και από τον τρόπο που όλες οι δυνατές τιμές του πεδίου ορισμού αντιστοιχίζονται με τους κόμβους του δικτύου χρησιμοποιώντας την συνάρτηση κατακερματισμού διατήρησης τάξης.

Ας υποθέσουμε προς το παρόν ότι το πεδίο ορισμού D_{ai} αποτελείται από ακέραιους αριθμούς². Υπάρχουν τρία πιθανά σενάρια που καθορίζουν τη σχέση του μεγέθους του πεδίου ορισμού και του αριθμού των κόμβων N που σχηματίζουν το δίκτυο ομοτίμων εταίρων.

- $|D_{ai}| = N$: αυτή είναι η περίπτωση που μόλις περιγράψαμε. Όταν το πλήθος τιμών του πεδίου ορισμού είναι ίσο με τον αριθμό των κόμβων, προκύπτει ένας-προς-ένα αντιστοίχιση μεταξύ κόμβων και τιμών. Κάτω από αυτές τις προϋποθέσεις, ένα διάστημα τιμών R θα απαιτεί την εκτέλεση $|R|$ βημάτων (αφού πρώτα βέβαια εκτελέσουμε $O(\log(N))$ βήματα για να εντοπίσουμε τον πρώτο κόμβο που αντιστοιχίζεται με την μικρότερη τιμή του διαστήματος R).
- $|D_{ai}| > N$: σ' αυτή την περίπτωση, ένα κόμβος θα είναι υπεύθυνος για περισσότερες από μία τιμές του πεδίου ορισμού. Συγκεκριμένα, ένας κόμβος θα αντιστοιχίζεται με $\frac{|D_{ai}|}{N}$ διαδοχικές τιμές και έτσι ένα διάστημα R θα απαιτεί την εκτέλεση $\frac{|R|}{\frac{|D_{ai}|}{N}}$ βημάτων.
- $|D_{ai}| < N$: αυτή αποτελεί την πιο δύσκολη στο χειρισμό περίπτωση, επειδή δύο διαδοχικές τιμές στο πεδίο ορισμού είναι δυνατόν να αντιστοιχίζονται με μη διαδοχικούς κόμβους στο χώρο διευθύνσεων των αναγνωριστικών των κόμβων. Έτσι, δεδομένου ότι δύο μη διαδοχικοί κόμβοι δεν είναι άμεσα συνδεδεμένοι, μπορεί να χρειαστεί να εκτελέσουμε $O(\log(N))$ βήματα για κινηθούμε μεταξύ δύο κόμβων που είναι υπεύθυνοι για δύο διαδοχικές τιμές του πεδίου ορισμού. Σ' αυτή την περίπτωση, για να εγγυηθούμε την απευθείας σύνδεση των κόμβων που διαχειρίζονται διαδοχικές τιμές, θα πρέπει να κτίσουμε ένα δεύτερο DHT, με μέγεθος χώρου αναγνωριστικών των κόμβων ίσο με το πλήθος τιμών του πεδίου ορισμού D_{ai} .

Επιλέγουμε λοιπόν $|D_{ai}|$ κόμβους του δικτύου και τους αναγκάζουμε να σχηματίσουν ένα δεύτερο DHT για την διαχείριση του σχετικά μικρού πεδίου ορισμού του ιδιοχαρακτηριστικού a_i . Έτσι, ένα κόμβος που διαχειρίζεται τις τιμές του ιδιοχαρακτηριστικού a_i θα συμμετέχει σε δύο διαφορετικά DHTs και έτσι θα διατηρεί πίνακες δρομολόγησης και για τα δύο. Θα μπορούσαμε βέβαια, με στόχο τη μείωση του κόστους συντήρησης του επιπλέον DHT, να εφαρμόσουμε ένα απλό πίνακα δρομολόγησης που να περιέχει μόνο δείκτες προς των επόμενο αριθμητικά κόμβο που ανήκει στο ίδιο DHT. Έτσι η δρομολόγηση αιτήσεων προς τον κόμβο που είναι υπεύθυνος για την μικρότερη τιμή

²Όπως έχουμε ήδη αναφέρει, η υποστήριξη ενός πεδίου ορισμού με πραγματικούς αριθμούς μπορεί να γίνει με κατάλληλο ορισμό της ακρίβειας $pr(a_i)$, ώστε τελικά να προκύψει πεδίο ορισμού με πεπερασμένο πλήθος τιμών.

Σχήμα 4.2: Αποθήκευση διαστήματος τιμών με την προσέγγιση OP-DHT.

του διαστήματος R θα γίνεται στο πρωτεύον DHT (με την εκτέλεση $O(\log(N))$ βημάτων) και στην συνέχεια η αίτηση θα προωθείται στο δευτερεύον DHT, όπου θα χρειαστεί να εκτελεστούν το πολύ $|R|$ βήματα.

Τα τελευταία χρόνια έχουν παρουσιαστεί εργασίες που σχετίζονται με τον υπολογισμό των χαρακτηριστικών των δικτύων ομοτίμων εταιρών με ιδιαίτερα αποδοτικό τρόπο ([59, 67, 72]). Ένα από αυτά τα χαρακτηριστικά είναι για παράδειγμα το πλήθος των κόμβων που συμμετέχουν σ' αυτό μια δεδομένη χρονική στιγμή. Μπορούμε λοιπόν να υιοθετήσουμε μία από αυτές τις λύσεις και να εκτιμήσουμε το πλήθος των κόμβων N και στη συνέχεια να αποφασίσουμε για το αν θα πρέπει να σχηματίσουμε ένα δεύτερο DHT για την επεξεργασία ενός ιδιοχαρακτηριστικού με σχετικά μικρό μέγεθος πεδίου ορισμού.

Παράδειγμα 1 Επεξεργασία συνδρομής με περιορισμό διαστήματος τιμών στο OP-DHT.

Ας υποθέσουμε ότι έχουμε ένα σχετικά απλό δίκτυο Chord με αναγνωριστικά κόμβων αποτελούμενα από 3 bits, και άρα ικανό να υποστηρίξει το ποπύ $2^3 = 2^3 = 8$ κόμβους. Έστω λοιπόν, ότι ένας χρήστης συνδεδεμένος στον κόμβο 3 (Σχήμα 4.2) ορίζει μία συνδρομή αποτελούμενη από ένα ιδιοχαρακτηριστικό με πεδίο ορισμού ακέραιων τιμών. Εάν ο περιορισμός διαστήματος τιμών που έχει οριστεί είναι $v(a_i)=R = (l, h) = (0, 4)$, τότε ακολουθώντας την απλή λύση που προτείναμε στην αρχή αυτής της ενότητας, θα πρέπει να εκτελέσουμε $|R| \times O(\log(N)) = 3 \times O(\log(N))$ βήματα για να αποθηκεύσουμε το SubID σε $|R|$ κόμβους (σε αυτό το παράδειγμα το $|R|$ είναι ίσο με 3).

Στην OP-DHT προσέγγιση θα πρέπει να εκτελέσουμε $O(\log(N))$ βήματα για να εντοπίσουμε τον πρώτο κόμβο (τον 6 στην περίπτωση μας), και στη συνέχεια την εκτέλεση 2 ακόμα βημάτων για την αποθήκευση του SubID στους κόμβους 7 και 0. Άρα η συνολική απόδοση είναι $(|R|-1) + O(\log(N)) = 2 + O(\log(N))$ βήματα.

◇

Η Προσέγγιση *Virtual Range Tree Forest*

Η βασική ιδέα πίσω από την *Virtual Range Tree Forest* προσέγγιση, είναι η αντιστοίχιση κάθε δυνατού διαστήματος τιμών R που ορίζεται σε μία συνδρομή, με ένα σύνολο από κόμβους (του οποίου το μέγεθος είναι πολύ μικρότερο από $|R|$) με τη βοήθεια μίας δενδρικής δομής δεδομένων. Το *SubID* της συνδρομής αποθηκεύεται σε αυτούς τους κόμβους, και στη συνέχεια, όταν καταφθάσει στο σύστημα μας μία δημοσίευση, χρησιμοποιούμε την ίδια δεντρική δομή για να εντοπίσουμε αυτούς τους κόμβους και να συλλέξουμε τα αποθηκευμένα *SubIDs* των συνδρομών που συζευγνύουν με τη δημοσίευση.

Η δενδρική δομή που πρόκειται να χρησιμοποιήσουμε εδώ, αποτελεί μία παραλλαγή των *segment tree* [42]. Συγκεκριμένα, είναι μία γενίκευση της δενδρικής δομής που έχει παρουσιαστεί στην εργασία [50] και σκοπό έχει την αντιστοίχιση διαστημάτων τιμών στους κόμβους του δέντρου (Σχήμα 4.3). Στην προσέγγιση μας θα χρησιμοποιήσουμε ένα πλήθος από τέτοια δέντρα (ένα για κάθε ιδιοχαρακτηριστικό αριθμητικού τύπου που έχει οριστεί στο μοντέλο δημοσιεύσεων/συνδρομών), και έτσι από εδώ και στο εξής θα αναφερόμαστε σε ένα δάσος από δέντρα ή αλλιώς σε ένα *Range Tree Forest*. Ένα από αυτά τα δέντρα που ανήκει στο *Range Tree Forest*, είναι υπεύθυνο για την διαχείριση του εύρους τιμών D_{a_i} του ιδιοχαρακτηριστικού a_i . Συγκεκριμένα, κάθε κόμβος του δέντρου είναι υπεύθυνος για ένα διάστημα τιμών που ανήκει στο πεδίο ορισμού του ιδιοχαρακτηριστικού για το οποίο έχει χρησιμοποιηθεί.

Κάθε δέντρο χαρακτηρίζεται από δύο παραμέτρους:

- **βαθμός ή degree ή g :** ορίζει το πλήθος των παιδιών που είναι δυνατό να έχει κάθε κόμβος του δέντρου. Για την τιμή $g=2$ προκύπτει το γνωστό πλήρες δυαδικό δέντρο.
- **ύψος ή height ή H :** ορίζει το πλήθος των επιπέδων από το οποίο αποτελείται ένα τέτοιο δέντρο. Κάθε επίπεδο ορίζεται από το βάθος του d , όπου η ρίζα του δέντρου βρίσκεται σε βάθος $d=0$ και τα φύλλα του δέντρου σε βάθος $d=d_{max}=H$.

Κάθε δέντρο συμβολίζεται με T_{a_i} όπου a_i είναι το ιδιοχαρακτηριστικό για το οποίο το συγκεκριμένο δέντρο χρησιμοποιείται. Κάθε κόμβος του δέντρου συμβολίζεται με $Tnode_{d|p}$ όπου d είναι το βάθος στο οποίο βρίσκεται ο κόμβος και p είναι η θέση του κόμβου στο συγκεκριμένο βάθος, με το κόμβο που βρίσκεται αριστερά να έχει $p=0$ (για παράδειγμα ο κόμβος $Tnode_{3|6}$ βρίσκεται σε βάθος 3 και είναι ο έβδομος κόμβος από αριστερά σ' αυτό το βάθος). Ένας ακόμα συμβολισμός που θα χρησιμοποιηθεί για τους κόμβους του δέντρου είναι $Tnode_{[l,h]}$ όπου $[l, h]$ είναι το διάστημα τιμών για το οποίο είναι υπεύθυνος ο συγκεκριμένος κόμβος. Στο Σχήμα 4.3, μπορεί κανείς να δει ένα τέτοιο δέντρο το οποίο ανήκει στο *Range Tree Forest* ικανό να υποστηρίξει διαστήματα τιμών που ανήκουν στο πεδίο ορισμού $D = [0, 16)$.

Για την αντιστοίχιση διαστημάτων τιμών με *Tnodes* ακολουθούμε ένα ιεραρχικό σχήμα: η ρίζα του

Σχήμα 4.3: Ένα δέντρο στο *Range Tree Forest* με βαθμό $g=2$ (ένα πλήρες δυαδικό δέντρο) και ύψος $H=4$ ικανό να διαχειριστεί διαστήματα στο πεδίο ορισμού $D = [0, 16)$.

δέντρου είναι υπεύθυνη για το μεγαλύτερο δυνατό διάστημα (ολόκληρο το πεδίο ορισμού του ιδιοχαρακτηριστικού) και τα φύλλα είναι υπεύθυνα για το μικρότερο δυνατό διάστημα αποτελούμενο από μία μόνο τιμή. Κάθε ενδιάμεσος κόμβος, είναι υπεύθυνος για διάστημα τιμών που προκύπτει από την ένωση των διαστημάτων τιμών για τα οποία είναι υπεύθυνα τα παιδιά του. Η ένωση των διαστημάτων τιμών των κόμβων που βρίσκονται στο ίδιο βάθος, καλύπτει το πεδίο ορισμού του ιδιοχαρακτηριστικού. Το μέγεθος του διαστήματος που διαχειρίζεται ένας κόμβος σε βάθος d είναι ίσο με g^{H-d} . Δεδομένης αυτής της παρατήρησης, εάν ένας κόμβος είναι υπεύθυνος για το διάστημα $[l, h)$, τότε τα g παιδιά του θα είναι υπεύθυνα για τα διαστήματα:

$$[l, l + g^{H-d}], [l + g^{H-d}, l + 2 \times g^{H-d}], \dots, [l + (g-1) \times g^{H-d}, h]$$

Γενικά, ο κόμβος $Tnode_{d|p}$ είναι υπεύθυνος για το διάστημα:

$$[p \times g^{H-d}, p \times g^{H-d} + g^{H-d})$$

Η σωστή επιλογή για τον βαθμό του δέντρου (g) όπως επίσης και για το ύψος του (H), εξαρτάται από το μέγεθος του πεδίου ορισμού $|D_{ai}|$ του ιδιοχαρακτηριστικού a_i . Η αναγκαία συνθήκη είναι $|D_{ai}| \leq g^H$ (δεδομένου του βαθμού g , και του πεδίου ορισμού D_{ai} , το ύψος H πρέπει να είναι: $H \geq \log_g(D_{ai})$ και δεδομένου του ύψους H , και του πεδίου ορισμού D_{ai} , ο βαθμός g πρέπει να είναι: $g \geq 10^{\frac{\log(|D_{ai}|)}{H}}$). Παρακάτω, θα δείξουμε πως ρυθμίζοντας κατάλληλα τις τιμές των g και H μπορούμε να επηρεάσουμε την απόδοση κατά την επεξεργασία των συνδρομών ή των δημοσιεύσεων, ανάλογα με τις απαιτήσεις της εφαρμογής που αξιοποιεί την υποδομή δημοσιεύσεων/συνδρομών που παρουσιάζουμε εδώ. Αυτός είναι και ένας από τους λόγους που επιλέξαμε να χρησιμοποιήσουμε αυτή τη γενικευμένη δεντρική δομή.

Σε γενικές γραμμές η λειτουργικότητα που προσφέρεται με την αξιοποίηση αυτών των δέντρων είναι : (i) η ανάθεση διαστημάτων τιμών στους κόμβους του δέντρου και (ii) ο τεμαχισμός ενός διαστήματος τιμών σε υποδιαστήματα που αντιστοιχίζονται με τους κόμβους του δέντρου. Δεδομένης αυτής της λειτουργικότητας, όταν σε μία συνδρομή με αναγνωριστικό $SubID_j$, ορίζεται περιορισμός διαστήματος τιμών στο ιδιοχαρακτηριστικό a_i , κατά την επεξεργασία της :

1. αρχικά τεμαχίζουμε το διάστημα σε ένα κατάλληλα διαμορφωμένο σύνολο μη αποκαλυπτομένων υποδιαστημάτων, χρησιμοποιώντας ένα αλγόριθμο διάσπασης και τον δέντρο T_{ai} .
2. για κάθε ένα υποδιάστημα, χρησιμοποιούμε το όνομα του ιδιοχαρακτηριστικού a_i καθώς και το ίδιο το υποδιάστημα σαν όρισμα στην συνάρτηση κατακερματισμού που προσφέρει το DHT. Η τιμή που μας επιστρέφει η συνάρτηση κατακερματισμού, είναι το αναγνωριστικό του κόμβου που θα πρέπει να εντοπίσουμε ώστε να αποθηκεύσουμε το $SubID$ της συνδρομής.

Κατά την επεξεργασία των δημοσιεύσεων, και όταν στο σύστημα μας καταφθάσει μία δημοσίευση στην οποία έχει οριστεί μία αριθμητική τιμή στο ιδιοχαρακτηριστικό a_i :

1. εντοπίζουμε τον κόμβο-φύλλο του δέντρου T_{ai} , που είναι υπεύθυνο για το μικρότερο δυνατό διάστημα που περικλείει την αριθμητική τιμή της δημοσίευσης.
2. ακολουθούμε το μονοπάτι από την ρίζα του δέντρου προς το συγκεκριμένο φύλλο. Για κάθε $Tnode$ που βρίσκεται σ' αυτό το μονοπάτι, χρησιμοποιούμε το διάστημα για το οποίο είναι υπεύθυνο μαζί με το όνομα του ιδιοχαρακτηριστικού σαν όρισμα στη συνάρτηση κατακερματισμού, ώστε να εντοπίσουμε τους κόμβους του δικτύου που είναι πιθανό να έχουν αποθηκευμένα τα $SubIDs$ συνδρομών που συζευγνύουν με την δημοσίευση.

Διάσπαση Διαστημάτων και Αντιστοίχιση Αυτών με τους Κόμβους του Δέντρου

Στην προσέγγιση του Gao [50], παρουσιάζεται ένας αλγόριθμος διάσπασης ενός διαστήματος τιμών R με μέγεθος $|R|$ σε $2 \times \lfloor \log_2(|R|) \rfloor$ μη επικαλυπτόμενα υποδιαστήματα τιμών. Στην περίπτωση μας, και δεδομένου ότι τα δέντρα στο *Range Tree Forest* είναι πλήρη δέντρα με βαθμό $g \geq 2$, τροποποιήσαμε αυτό τον αλγόριθμο και τον παρουσιάζουμε στον Πίνακα 4.1. Χρησιμοποιώντας τον αλγόριθμό μας, μπορούμε να διασπάσουμε ένα διάστημα R το πολύ σε $2 \times g \times \lfloor \log_g |R| \rfloor - g$ υποδιαστήματα τα οποία δεν επικαλύπτονται και η ένωσή τους μας δίνει το διάστημα R . Αν το πλήθος τους είναι k και συμβολίζονται με r_i τότε :

$$r_1 \cup r_2 \cup r_3 \cup \dots \cup r_k = R$$

Range Decomposition Algorithm

Notation

Range $R : [l, h)$

SR : List of Sub-Ranges

g : degree of each tree node

```
1. rangeDecomposition ( $R, SR, g$ )
2.  $|R| = h - l$ 
3.  $d = \lfloor \log(|R|) \rfloor$ 
4. while(  $d \geq 0$ )
5.   if ( $l \% g^d == 0$ )
6.      $SR.push\_back([l, l + g^d))$ 
7.      $R = [l + g^d, h)$ 
8.   rangeDecomposition ( $R, SR, g$ )
9.   return
10.  else
11.     $d = d - 1$ 
```

Πίνακας 4.1: Αλγόριθμος διάσπασης διαστημάτων τιμών.

και

$$\forall i, j \ r_i \cap r_j = \emptyset$$

Γενικά θα μπορούσαμε να αντιστοιχίσουμε ένα διάστημα R σε υποδιαστήματα με δύο εκ διαμέτρου αντίθετους τρόπους. Από τη μία θα μπορούσαμε να αντιστοιχίσουμε το R μόνο στη ρίζα ενώ από την άλλη μόνο σε κόμβους φύλλα. Η πρώτη προσέγγιση προφανώς δεν είναι αποδοτική από πλευράς εξισορρόπησης φόρτου, μιας και όλες οι αιτήσεις θα κατέληγαν στη ρίζα του δέντρου. Η δεύτερη επίσης δεν είναι αποδοτική, επειδή θα πρέπει να επικοινωνήσουμε με $|R|$ κόμβους (εφόσον στα φύλλα αντιστοιχίζονται διαστήματα μεγέθους 1, τότε για να καλύψουμε το διάστημα R απαιτούνται $|R|$ υποδιαστήματα). Με τον αλγόριθμο που παρουσιάζουμε στον Πίνακα 4.1 προσπαθούμε από τη μία να μειώσουμε το πλήθος των υποδιαστημάτων r_i , διατηρώντας την ισότητα $|R| = \sum_{i=1}^k |r_i|$.

Ο αλγόριθμος που παρουσιάζεται στον Πίνακα 4.1 ξεκινάει προσπαθώντας να εντοπίσει το υποδιάστημα (που αντιστοιχίζεται με κάποιο κόμβο του δέντρου) που αποτελεί υποσύνολο του διαστήματος R και βρίσκεται στο μικρότερο δυνατό βάθος. Αφού βρεθεί, έστω r_1 , επαναλαμβάνουμε την ίδια διαδικασία με το διάστημα $R \setminus r_1$. Η διαδικασία επαναλαμβάνεται μέχρι να υπολογίσουμε και το τελευταίο υποδιάστημα r_i .

Για παράδειγμα, με το συγκεκριμένο αλγόριθμο και έχοντας υπόψη το δέντρο του Σχήματος 4.3 με ύψος $H=4$ και βαθμό $g=2$, μπορούμε να τεμαχίσουμε το διάστημα $[3, 13)$ στα υποδιαστήματα $[3, 4)$, $[4, 8)$, $[8, 12)$ και $[12, 13)$. Ξεκινώντας με το αρχικό διάστημα, προσπαθούμε να βρούμε το υποδιάστημα r_1 που αποτελεί υποσύνολο του R και βρίσκεται στο μικρότερο βάθος. Αυτό είναι το $r_1 = [3, 4)$ σε βάθος 4. Συνεχίζουμε με το διάστημα $R' = R \setminus r_1 = [4, 13)$, για το οποίο εντοπίζουμε το υποδιάστημα στο μικρότερο βάθος με κάτω όριο το 4, το $r_2 = [4, 8)$. Η διαδικασία συνεχίζεται με το διάστημα $R'' = R' \setminus r_2 = [8, 13)$

Storing Subscriptions - Ranges over the virtual Range Tree Forest

Notation

$SubID$: subscription identifier

a_i : attribute i , $v(a_i)$: value of attribute a_i

$DHT.hash()$: DHT's hash function (SHA-1 in Chord)

L_{a_i-num} : List of stored $SubIDs$ for attribute a_i that define numerical predicate

1. For every numerical attribute a_i
2. If a_i has an equality constraint
3. compute the smallest sub-range r including $v(a_i)$
4. store $SubID$ in L_{a_i-num} in node $DHT.hash(r|a_i)$
5. If a_i has range constraint
6. decompose the range to a set of sub-ranges
7. for each sub-range r_j
8. store $SubID$ in L_{a_i-num} in node $DHT.hash(r_j|a_i)$

Πίνακας 4.2: Η διαδικασία αποθήκευση των αναγνωριστικών $SubIDs$ των συνδρομών στις οποίες έχουν οριστεί περιορισμοί διαστημάτων αριθμητικών τιμών, με την προσέγγιση virtual Range Tree Forest.

όπου υπολογίζουμε το $r_3 = [8, 12)$ και τέλος το $r_4 = [12, 13)$. Για αυτά τα υποδιαστήματα είναι υπεύθυνοι οι κόμβοι $Tnode_{4|3}$, $Tnode_{2|1}$, $Tnode_{2|2}$, και $Tnode_{4|12}$ αντίστοιχα, όπως μπορεί να δει κανείς στο Σχήμα 4.3.

Πρέπει να σημειωθεί, ότι δεν υπάρχει ανάγκη να κτίσουμε τα δέντρα μας ή να διατηρούμε τη συνεκτικότητα μεταξύ των κόμβων σε επίπεδο υλοποίησης, αλλά και ούτε να διατηρούμε πληροφορία σχετικά με τα δέντρα στο ίδιο το δίκτυο. Αυτό που πραγματικά απαιτείται για να υπολογίσουμε το σύνολο των υποδιαστημάτων που προκύπτουν από ένα δεδομένο διάστημα, είναι τα χαρακτηριστικά των δέντρων στο Range Tree Forest (ο βαθμός g και το ύψος H του) και το αλγόριθμο διάσπασης.

Στον πίνακα 4.2 μπορεί να βρει κανείς λεπτομέρειες σχετικά με την επεξεργασία συνδρομών με ιδιοχαρακτηριστικά αριθμητικού τύπου και περιορισμούς διαστημάτων τιμών.

4.1.2 Stateful Προσεγγίσεις

Η Προσέγγιση Stateful Range Tree Forest

Η προσέγγιση virtual Range Tree Forest βασίζεται στη χρήση μίας δενδρική δομής και στη συσχέτισή της με το DHT δίκτυο ομοτίμων εταίρων. Έτσι η προσπέλαση ενός κόμβου στο δέντρο, συνεπάγεται τον εντοπισμό και την προσπέλαση του κόμβου του δικτύου που αντιστοιχίζεται με τον κόμβο του δέντρου. Συνήθως, στα περισσότερα DHTs αυτή η διαδικασία συνεπάγεται την διενέργεια $O(\log(N))$ βημάτων, ως αποτέλεσμα της κλήσης της συνάρτησης $lookup()$ που προσφέρει το DHT για τον εντοπισμό ενός κόμβου με συγκεκριμένο αναγνωριστικό. Μπορούμε να βελτιώσουμε (μειώσουμε) τον αριθμό των βημάτων, με το να κτίσουμε το δέντρο απ ευθεία πάνω στο DHT δίκτυο. Σε αυτή την stateful προσέγγιση, οι κόμβοι του

δέντρου στην ουσία φιλοξενούνται στους κόμβους του δικτύου. Κάθε κόμβος στο δίκτυο, είναι δυνατό να φιλοξενεί περισσότερους από ένα κόμβους του δέντρου ($Tnodes$). Οι κόμβοι στο δέντρο, όπως και πριν, προσδιορίζονται από το διάστημα για το οποίο είναι υπεύθυνοι (π.χ. $Tnode_{[3,4]}$) ή από το βάθος και τη θέση τους σ' αυτό (π.χ. $Tnode_{8|10}$).

Η βασική λειτουργικότητα είναι παρόμοια με αυτή που παρουσιάστηκε στην *virtual Range Tree Forest* προσέγγιση, μόνο που πλέον δημιουργούμε ένα πραγματικό δέντρο πάνω στο δίκτυο, με τους κόμβους του δέντρου να διατηρούν δείκτες (για παράδειγμα τη διεύθυνση IP) προς άλλους κόμβους του δικτύου που φιλοξενούν τα παιδιά τους. Κατά την επεξεργασία των συνδρομών με περιορισμούς διαστήματος τιμών, αρχικά τεμαχίζουμε το διάστημα στο κατάλληλο σύνολο από υποδιαστήματα (εφαρμόζοντας τον αλγόριθμο που παρουσιάζεται στον Πίνακα 4.1). Για κάθε υποδιάστημα κτίζουμε μόνο ένα τμήμα του δέντρου και συγκεκριμένα το μονοπάτι από τον κόμβο που είναι υπεύθυνος για το υποδιάστημα μέχρι τη ρίζα του δέντρου.

Προκειμένου να κτίσουμε αυτό το μονοπάτι (για κάθε υποδιάστημα), ξεκινάμε από τη ρίζα του δέντρου, και εντοπίζουμε τον κόμβο του δικτύου που θα πρέπει να τη φιλοξενήσει. Το αναγνωριστικό αυτού του κόμβου είναι το αποτέλεσμα που επιστρέφει η συνάρτηση κατακερματισμού, εάν περάσουμε σαν όρισμα το διάστημα που διαχειρίζεται η ρίζα μαζί με το όνομα a_i του ιδιοχαρακτηριστικού στο οποίο αναφερόμαστε. Στη συνέχεια, κτίζουμε εκεί την ρίζα του δέντρου μαζί με τον πίνακα δρομολόγησης της, που αποτελείται από g δείκτες (αρχικά μη ορισμένους), ένα για το κάθε πιθανό παιδί της. Το επόμενο βήμα σχετίζεται με την εύρεσή του κόμβου που θα φιλοξενήσει το παιδί της, και επιτυγχάνεται με το ίδια μεθοδολογία που ακολουθήσαμε για την εύρεση του κόμβου που θα φιλοξενήσει την ρίζα, χρησιμοποιώντας τώρα το αμέσως μικρότερο διάστημα από το σύνολο διαστημάτων που διαχειρίζονται οι $Tnodes$ στο μονοπάτι μας. Αφού εντοπιστεί αυτός ο κόμβος, κτίζεται εκεί το παιδί και ενημερώνεται κατάλληλα ο πίνακας δρομολόγησης του πατέρα, ώστε να δείχνει στο νέο παιδί του. Η διαδικασία αυτή συνεχίζεται μέχρι να φτάσουμε στον κόμβο όπου θα φιλοξενήσει τον κόμβο του δέντρου που είναι υπεύθυνος για το υποδιάστημα, όπου και αποθηκεύουμε τελικά το *SubID* της συνδρομής. Το κτίσιμο των σχετικών μονοπατιών συνεχίζεται και για τα υπόλοιπα υποδιαστήματα, με αποτέλεσμα μετά το τέλος αυτής της διαδικασίας να έχει σχηματιστεί ένα τμήμα του δέντρου για το ιδιοχαρακτηριστικό a_i , και να έχει αποθηκευτεί κατάλληλα το *SubID* της συνδρομής.

Ένα θέμα που προκύπτει από αυτή τη διαδικασία σχετίζεται με το τρόπο υπολογισμού των διαστημάτων για τα οποία είναι υπεύθυνοι οι $Tnodes$ που ανήκουν σε κάθε ένα από αυτά τα μονοπάτια. Με άλλα λόγια, δεδομένου ενός διαστήματος τιμών R που αντιστοιχίζεται με ένα κόμβο στο δέντρο $Tnode_R$, θα πρέπει να μπορούμε να υπολογίσουμε τα διαστήματα των $Tnodes$ που βρίσκονται στο μονοπάτι από τον $Tnode_R$ έως

Σχήμα 4.4: Υπολογισμός του κατάλληλου μονοπατιού προς τη ρίζα σε ένα δέντρο στο *Range Tree Forest* με βαθμό $g=2$ και ύψος $H=4$.

τη ρίζα του δέντρου. Η λύση που προτείνουμε είναι η εξής: ας υποθέσουμε ότι το διάστημα $R = [l, h)$, αντιστοιχίζεται στο δέντρο με ύψος H και βαθμό g , με τον κόμβο $Tnode_{[l,h)}$. Όπως ήδη έχει αναφερθεί, p είναι η θέση του κόμβου $Tnode_{[l,h)}$ στο βάθος στο οποίο βρίσκεται (με τον κόμβο που βρίσκεται αριστερά να είναι στην θέση $p=0$) και $|r_i|$ ($= h_i - l_i$) είναι το μέγεθος του διαστήματος $r_i = [l_i, h_i)$. Δεδομένου ότι το διάστημα που εξετάζουμε είναι το $R = [l, h)$ (δηλ. ο $Tnode_{[l,h)}$ αποτελεί το τέρμα του μονοπατιού) τότε τα υπόλοιπα διαστήματα r_i που αντιστοιχίζονται με τους κόμβους του μονοπατιού μέχρι τη ρίζα, δίνονται από την παρακάτω σχέση:

$$r_i = [p_i \times |r_i|, p_i \times |r_i| + |r_i|) \quad 1 \leq i \leq H - \lfloor \log g |r_0| \rfloor$$

όπου:

$$|r_i| = g \times |r_{i-1}|, \quad |r_0| = |R| = h - l$$

$$p_i = \lfloor \frac{p_{i-1}}{g} \rfloor, \quad p_0 = \lfloor l / |r_0| \rfloor$$

Για παράδειγμα, εάν υποθέσουμε ότι επιθυμούμε να χρησιμοποιήσουμε το δέντρο του Σχήματος 4.4 (με $g=2$ και $H=4$) και το υπό εξέταση διάστημα είναι το $R = [10, 11)$, τότε τα υπόλοιπα διαστήματα που αντιστοιχίζονται με τους $Tnodes$ από τον $Tnode_{[10,11)}$ μέχρι τη ρίζα, προκύπτουν ως εξής: οι αρχικές τιμές της παραπάνω σχέσεις είναι: $|r_0| = |R| = 1$ και $p_0 = \lfloor l / |r_0| \rfloor = 10 / 1 = 10$. Το i μεταβάλλεται από 1 έως $H - \lfloor \log g |r_0| \rfloor = 4 - 0 = 4$. Έτσι, εφαρμόζοντας την παραπάνω σχέση προκύπτουν τα εξής διαστήματα τιμών:

Storing Subscriptions - Ranges over the stateful Range Tree Forest

Notation

$SubID$: subscription identifier, a_i : attribute i , $v(a_i)$: value of attribute a_i

$DHT.hash()$: DHT's hash function (SHA-1 in Chord)

L_{ai-num} : List of stored $SubIDs$ for attribute a_i that define numerical predicate

$Tnode_{[l,h]}$: tree node responsible for the sub-range $[l,h]$

1. For every numerical attribute a_i
2. If a_i has an equality constraint
3. compute the smallest sub-range $R = [l, h]$ which includes the value $v(a_i)$
4. compute the set of sub-ranges r_i in the path from $Tnode_{r_0}$ to the root
5. construct that tree path in the DHT network
6. store $SubID$ in L_{ai-num} in $Tnode_r$ which is hosted in node $DHT.hash(R|a_i)$
7. If a_i has range constraint
8. decompose the range to a set of sub-ranges
9. for each sub-range R
10. compute the set of sub-ranges r_i in the path from $Tnode_r$ to the root
11. construct that tree path in the DHT network
12. store $SubID$ in L_{ai-num} in $Tnode_r$ which is hosted in node $DHT.hash(R|a_i)$

Πίνακας 4.3: Η διαδικασία αποθήκευση των αναγνωριστικών $SubIDs$ των συνδρομών στις οποίες έχουν οριστεί περιορισμοί διαστημάτων αριθμητικών τιμών, με την προσέγγιση stateful Range Tree Forest.

$$\begin{array}{cc|c}
 |r_0| = 1 & p_0 = 10 & r_0 = R = [10, 11) \\
 |r_1| = 2 & p_1 = 5 & r_0 = [5 \times 2, 5 \times 2 + 2) = [10, 12) \\
 |r_2| = 4 & p_2 = 2 & r_0 = [2 \times 4, 2 \times 4 + 4) = [8, 12) \\
 |r_3| = 8 & p_3 = 1 & r_0 = [1 \times 8, 1 \times 8 + 8) = [8, 16) \\
 |r_4| = 16 & p_4 = 0 & r_0 = [0 \times 16, 0 \times 16 + 16) = [0, 16)
 \end{array}$$

Έτσι λοιπόν, θα πρέπει να κτίσουμε και να ακολουθήσουμε το μονοπάτι $Tnode_{[0,16)}$ (η ρίζα του δέντρου), $Tnode_{[8,16)}$, $Tnode_{[8,12)}$, $Tnode_{[10,12)}$, και τελικά $Tnode_{[10,11)}$, όπως μπορείτε να δείτε και από το Σχήμα 4.4.

Στον Πίνακα 4.3 μπορεί να βρει κανείς λεπτομέρειες σχετικά με την επεξεργασία συνδρομών με περιορισμούς διαστήματος με την προσέγγιση stateful Range Tree Forest.

Παράδειγμα 2 Επεξεργασία συνδρομών με περιορισμούς διαστήματος με την προσέγγιση stateful Range Tree Forest.

Ας υποθέσουμε ότι καταφθάνει στο σύστημα μας μία συνδρομή με περιορισμό διαστήματος ορισμένο στο ιδιοχαρακτηριστικό a_i , με τιμή $v(a_i)=[l, h) = [2, 8)$. Το δέντρο το οποίο πρόκειται να χρησιμοποιηθεί για την επεξεργασία διαστημάτων τιμών σχετικών με το συγκεκριμένο ιδιοχαρακτηριστικό, ας θεωρήσουμε ότι έχει ύψος $H=4$ και βαθμό $g=2$ (όπως ακριβώς το δέντρο που παρουσιάζεται στο Σχήμα 4.4). Αρχικά, τεμαχίζουμε

Storing Subscriptions - Strings

Notation

$SubID$: subscription identifier, a_i : attribute i , $v(a_i)$: value of attribute a_i

$DHT.hash()$: DHT's hash function (SHA-1 in Chord)

$L_{a_i-eq}, L_{a_i-prefix}, L_{a_i-suffix}$: List of stored $SubIDs$ for attribute a_i that define equality, prefix, or suffix predicates

1. For every string attribute a_i
2. If a_i has an equality constraint
3. store $SubID$ in L_{a_i-eq} in node $DHT.hash(v(a_i))$
4. If a_i has an prefix constraint
5. store $SubID$ in $L_{a_i-prefix}$ in node $DHT.hash(v(a_i))$
6. If a_i has a suffix constraint
7. store $SubID$ in $L_{a_i-suffix}$ in node $DHT.hash(v(a_i))$

Πίνακας 4.4: Η διαδικασία επεξεργασίας των συνδρομών με ιδιοχαρακτηριστικά τύπου συμβολοσειράς.

το διάστημα $[2, 8)$ εφαρμόζοντας τον σχετικό αλγόριθμο που παρουσιάζεται στον Πίνακα 4.1, και προκύπτουν τα υποδιαστήματα $[2, 4)$ και $[4, 8)$. Για το πρώτο υποδιάστημα, εντοπίζουμε τον κόμβο του δικτύου που πρόκειται να φιλοξενήσει τον $Tnode_{[2,4)}$. Το αναγνωριστικό αυτού του κόμβου είναι η έξοδος της συνάρτησης κατακερματισμού του DHT, αφού περάσουμε σαν όρισμα της το υποδιάστημα και το όνομα του ιδιοχαρακτηριστικού a_i . Σε αυτό τον κόμβο, $N5$ στο παράδειγμα μας όπως μπορείτε να δείτε στο Σχήμα 4.5, κατασκευάζουμε τον $Tnode_{[2,4)}$ και αποθηκεύουμε το αναγνωριστικό $SubID$ της συνδρομής. Το επόμενο βήμα, ασχολείται με την κατασκευή του μονοπατιού από τον $Tnode_{[2,4)}$ έως τη ρίζα του δέντρου. Υπολογίζουμε τα διαστήματα για τα οποία είναι υπεύθυνοι οι κόμβοι που ανήκουν σ' αυτό το μονοπάτι, χρησιμοποιώντας τη σχέση που παρουσιάσαμε προηγουμένως, και καταλήγουμε στα διαστήματα $[0, 4)$, $[0, 8)$, και $[0, 16)$. Για κάθε ένα από αυτά εντοπίζουμε τον κατάλληλο κόμβο στο δίκτυο ($N3$, $N2$, και $N1$ σ' αυτό το παράδειγμα) και κτίζουμε εκεί τους $Tnode_{[0,4)}$, $Tnode_{[0,8)}$, και $Tnode_{[0,16)}$ ενώ παράλληλα ορίζουμε κατάλληλα και τους δείκτες στους πίνακες δρομολόγησης τους, ώστε να δείχνουν προς τα παιδιά τους, όπως φαίνεται στο Σχήμα 4.5.

Αφού ολοκληρώσουμε την επεξεργασία του πρώτου υποδιαστήματος, συνεχίζουμε με το δεύτερο $[4, 8)$, όπου εντοπίζουμε τον αντίστοιχο κόμβο, $N4$, κατασκευάζουμε εκεί τον $Tnode_{[4,8)}$ όπου και αποθηκεύουμε το $SubID$ της συνδρομής. Στη συνέχεια, κτίζουμε το μονοπάτι μέχρι τη ρίζα του δέντρου αποτελούμενο από τους κόμβους $Tnode_{[0,8)}$ και $Tnode_{[0,16)}$ που όπως μπορείτε να δείτε στο Σχήμα 4.5 έχουν ήδη σχηματιστεί σαν αποτέλεσμα της επεξεργασίας του πρώτου υποδιαστήματος. Έτσι, το μόνο που πρέπει να κάνουμε, είναι να ενημερώσουμε τον $Tnode_{[0,8)}$ σχετικά με την ύπαρξη του νέου του παιδιού $Tnode_{[4,8)}$, ώστε να ενημερώσει ανάλογα τον πίνακα δρομολόγησης του με ένα δείκτη στον κόμβο $N4$.

Όπως μπορείτε να δείτε στο Σχήμα 4.5, έχει σχηματιστεί ένα μέρος του δέντρου για το ιδιοχαρακτηριστικό a_i , και είναι πλέον έτοιμο να δεχθεί την επεξεργασία νέων δημοσιεύσεων. ◇

Σχήμα 4.5: Σχηματισμό μέρους του δέντρου κατά την επεξεργασία συνδρομών με την προσέγγιση stateful Range Tree Forest.

4.2 Επεξεργασία Συνδρομών με Ιδιοχαρακτηριστικά Τύπου

Συμβολοσειράς

4.2.1 Η Stateless Προσέγγιση

Η βασική ιδέα πίσω από την stateless προσέγγιση για την επεξεργασία συνδρομών με ιδιοχαρακτηριστικά τύπου συμβολοσειράς στην υποδομή *Pyracanthus* (που αρχικά έχει παρουσιαστεί στο [5]), είναι η αποθήκευση των αναγνωριστικών των συνδρομών, *SubIDs*, σε εκείνους τους κόμβους των οποίων το αναγνωριστικό, *nodeID*, είναι η τιμή που επιστρέφει η συνάρτηση κατακερματισμού όταν το όρισμα της είναι οι τιμές που ορίζονται στα ιδιοχαρακτηριστικά της συνδρομής. Στη συνέχεια, η σύζευξη των δημοσιεύσεων με τις αποθηκευμένες συνδρομές, μπορεί να επιτευχθεί απλά με τον εντοπισμό αυτών των κόμβων. Είναι προφανές, ότι και σε αυτή την προσέγγιση ακολουθούμε το μοντέλο του κόμβου «ραντεβού», όπου οι δημοσιεύσεις συναντούν τις συνδρομές και τελικά επιτυγχάνεται η σύζευξη τους.

Κάθε κόμβος στο δίκτυο μας, διατηρεί τρεις λίστες ανά ιδιοχαρακτηριστικό τύπου συμβολοσειράς. Αυτές είναι οι $L_{ai-pref}$, $L_{ai-suff}$, και L_{ai-eq} στις οποίες αποθηκεύονται τα *SubIDs* των συνδρομών που φέρουν περιορισμό προθέματος, επιθέματος, ή ισότητας αντίστοιχα για το ιδιοχαρακτηριστικό a_i . Η διαδικασία επεξεργασίας μιας συνδρομής ξεκινάει με τον έλεγχο του είδους του τελεστή (πρόθεμα, επίθεμα, ή ισότητα) που ορίζεται σε κάθε ιδιοχαρακτηριστικό a_i (Πίνακας 4.4).

Εάν έχει οριστεί ισότητα, τότε το *SubID* θα πρέπει να αποθηκευτεί στη λίστα L_{ai-eq} του κόμβου του

οποίου το αναγνωριστικό είναι ίσο (ή αριθμητικά κοντά) στη τιμή που επιστρέφει η συνάρτηση κατακερματισμού αφού περάσουμε την τιμή του ιδιοχαρακτηριστικού $v(a_i)$, σαν όρισμα. Για τους τελεστές προθέματος και επιθέματος, το *SubID* αποθηκεύεται στις λίστες $L_{ai-pref}$ και $L_{ai-suff}$ του αντίστοιχου κόμβου. Για την αποθήκευση των *SubIDs* που ορίζουν ένα από αυτούς τους τρεις τελεστές σε συμβολοσειρές, απαιτούνται $O(\log(N))$ βήματα (όσα δηλαδή απαιτούνται για την εκτέλεση ενός DHT lookup).

Παράδειγμα 3 Αποθήκευση συνδρομών με ιδιοχαρακτηριστικό τύπου συμβολοσειράς στη *stateful* προσέγγιση.

Ας υποθέσουμε ότι ένας χρήστης δηλώνει τα ενδιαφέροντα του με τις παρακάτω συνδρομές *Sub1* και *Sub2*:

$$\begin{aligned} Sub1 &= \{SubID_1 \mid a_1:MB = '*LNZ', a_2:IP = '150.*'\} \\ Sub2 &= \{SubID_2 \mid a_3:OS = 'Linux', a_4:OSVER = 'Debian*'\} \end{aligned}$$

Όπως μπορείτε να δείτε, στη συνδρομή *Sub1* και στο ιδιοχαρακτηριστικό της *MB* (a_1) έχει οριστεί ένα τελεστής προθέματος με τιμή $v(a_1) = '*LNZ'$. Για το συγκεκριμένο ιδιοχαρακτηριστικό, το *SubID₁* θα αποθηκευτεί στη λίστα $L_{a1-suff}$ του κόμβου με $nodeID = DHT.hash('LNZ')$. Παρόμοια, για το δεύτερο ιδιοχαρακτηριστικό, *IP* (a_2) με τελεστή προθέματος, και τιμή $v(a_2) = '150.*'$, αποθηκεύουμε το *SubID₁* στη λίστα $L_{a2-pref}$ του κόμβου με $nodeID = DHT.hash('150.')$.

Για τη δεύτερη συνδρομή, στο ιδιοχαρακτηριστικό *OS* (a_3) ορίζεται τελεστής ισότητας, με τιμή $v(a_3) = 'Linux'$ πράγμα που έχει ως αποτέλεσμα την αποθήκευση του *SubID₂* στον κόμβο με $nodeID = DHT.hash('Linux')$, στην λίστα L_{a3-eq} . Τέλος, για το ιδιοχαρακτηριστικό *OSVER* (a_4), ενεργούμε ανάλογα, και αποθηκεύουμε το *SubID₂* στη λίστα $L_{a4-pref}$ του κόμβου με $nodeID = DHT.hash('Debian')$. $\diamond$

4.2.2 Η *Stateful* Προσέγγιση

Σε αυτή τη προσέγγιση, αναπτύσσουμε και κτίζουμε πάνω στο DHT δίκτυο μία δενδρική δομή με παρόμοιο τρόπο με αυτό που παρουσιάστηκε στην ενότητα 4.1.2. Η μόνη διαφορά είναι ότι πλέον χρησιμοποιούμε αυτή τη δενδρική δομή για την επεξεργασία συμβολοσειρών. Έτσι, οι κόμβοι του δέντρου πλέον, διαχειρίζονται τμήματα συμβολοσειρών σε αναλογία με την διαχείριση διαστημάτων για την επεξεργασία αριθμητικού τύπου ιδιοχαρακτηριστικών στην *stateful Range Tree Forest* προσέγγιση. Για παράδειγμα, η ετικέτα ενός κόμβου σε αυτό το δέντρο μπορεί να είναι η συμβολοσειρά '1b1' (και άρα αυτός ο κόμβος συμβολίζεται με $T_{node1b1}$) που πιθανόν να αποτελεί και την τιμή του ιδιοχαρακτηριστικού μιας συνδρομής όπου το *SubID* της είναι αποθηκευμένο εκεί.

Κάθε δέντρο σε αυτή τη προσέγγιση, έχει συγκεκριμένο ύψος H καθώς και συγκεκριμένο βαθμό g . Το ύψος του δέντρου καθορίζει τον μέγιστο αριθμό χαρακτήρων που μπορεί να έχει μία συμβολοσειρά, ενώ ο βαθμός του καθορίζει το πλήθος των διαθέσιμων χαρακτήρων που μπορούμε να χρησιμοποιήσουμε σε κάθε συμβολοσειρά. Ένα βασικό χαρακτηριστικό που προσδιορίζει αυτό το δέντρο είναι ότι οι ετικέτες των παιδιών ενός κόμβου έχουν σαν πρόθεμα την ετικέτα του πατέρα τους, και επιπλέον ένα από του g διαθέσιμους χαρακτήρες. Υπάρχει λοιπόν και εδώ η έννοια της επικάλυψης της ετικέτας του πατέρα σε σχέση με τις ετικέτες των παιδιών του, σε επίπεδο συμβολοσειράς εκφραζόμενη με τον τελεστή προθέματος, σε αναλογία με την επικάλυψη του διαστήματος του πατέρα με τα διαστήματα των παιδιών στην προσέγγιση *stateful Range Tree Forest*, σε επίπεδο αριθμητικών τιμών με τον τελεστή ένωσης.

Όπως και πριν, για να εντοπίσουμε τον κόμβο του δικτύου που θα φιλοξενήσει για παράδειγμα τον κόμβο $Tnode_{lbl}$, χρησιμοποιούμε την συνάρτηση κατακερματισμού του DHT με όρισμα την συμβολοσειρά 'lbl' (που είναι η τιμή $v(a_i)$ του ιδιοχαρακτηριστικού a_i που ορίζεται στη συνδρομή) μαζί με το όνομα του ιδιοχαρακτηριστικού a_i .

Κατά την διαδικασία κατασκευής του μονοπατιού από τον $Tnode_{lbl}$ μέχρι την ρίζα του δέντρου, καλούμαστε και εδώ να υπολογίσουμε τις ετικέτες των ενδιάμεσων κόμβων. Δεδομένου του τύπου αυτού του δέντρου, είναι σχετικά εύκολο να υπολογίσουμε το σύνολο αυτών των συμβολοσειρών. Αν αναλογιστούμε ότι ο πατέρας ενός $Tnode_{lbl}$ έχει σαν ετικέτα το πρόθεμα $l - 1$ χαρακτήρων της ετικέτας 'lbl' αποτελούμενη από l χαρακτήρες, τότε το σύνολο των συμβολοσειρών αποτελείται από τα προθέματα μήκους i χαρακτήρων της ετικέτας 'lbl', με το i να κυμαίνεται από την τιμή 0^3 έως την τιμή l . Για παράδειγμα, εάν η τιμή του ιδιοχαρακτηριστικού a_i είναι $v(a_i) = 'abcd'$, τότε θα πρέπει να κατασκευάζουμε το μονοπάτι από τον κόμβο $Tnode_{abcd|a_i}$ (όπου θα αποθηκευτεί και το αναγνωριστικό της συνδρομής) έως την ρίζα $Tnode_{a_i}$, αποτελούμενο από τους κόμβους: $Tnode_{abcd|a_i}$, $Tnode_{abc|a_i}$, $Tnode_{ab|a_i}$, $Tnode_{a|a_i}$, και $Tnode_{a_i}$. Η επιλογή ενός τέτοιου δέντρου με τα συγκεκριμένα χαρακτηριστικά για την επεξεργασία συμβολοσειρών, θα γίνει πιο κατανοητή κατά την παρουσίαση της μεθοδολογίας που θα ακολουθήσουμε για την επεξεργασία των δημοσιεύσεων (Ενότητα 4.3.1) και συγκεκριμένα του τρόπου που ανακαλύπτουμε τα *SubIDs* των συνδρομών που συζευγνύουν με την τιμή της δημοσίευσης και έχουν περιορισμούς προθέματος ή επιθέματος.

Ανάλογα με τον τύπο του τελεστή που ορίζεται στη συνδρομή, το *SubID* της αποθηκεύεται στη λίστα L_{ai-eq} , $L_{ai-pref}$ ή $L_{ai-suff}$ όταν πρόκειται για ισότητα, πρόθεμα ή επίθεμα στον κατάλληλο (ανάλογα με την τιμή της συνδρομής) κόμβο του δέντρου. Κατά την διαδικασία αυτή κατασκευάζεται και το μονοπάτι προς τη ρίζα, εάν δεν υπάρχει ήδη.

³Το πρόθεμα μήκους 0 χαρακτήρων προορίζεται για την ετικέτα της ρίζας του δέντρου, όπου αποτελείται μόνο από το όνομα του ιδιοχαρακτηριστικού για το οποίο το δέντρο είναι αφιερωμένο.

Σχήμα 4.6: Επεξεργασία συνδρομής και σύζευξη αυτής με σχετική δημοσίευση, με την προσέγγιση *OP-DHT*.

4.3 Επεξεργασία και Σύζευξη των Δημοσιεύσεων με τις Συνδρομές

4.3.1 Συλλογή των Αποθηκευμένων Αναγνωριστικών

Αριθμητικού Τύπου Ιδιοχαρακτηριστικά και Stateless Προσεγγίσεις

Η επεξεργασία δημοσιεύσεων με αριθμητικές τιμές στα ιδιοχαρακτηριστικά τους με την προσέγγιση *OP-DHT*, επιτυγχάνεται χρησιμοποιώντας την ίδια συνάρτηση κατακερματισμού με διατήρηση τάξης που χρησιμοποιήσαμε για την επεξεργασία των συνδρομών. Συγκεκριμένα, χρησιμοποιούμε την τιμή της δημοσίευσης σαν όρισμα, και εντοπίζουμε τον κόμβο του οποίου το αναγνωριστικό είναι η έξοδος της συνάρτησης. Από αυτό τον κόμβο, συλλέγουμε όλα τα *SubIDs* που είναι αποθηκευμένα στη λίστα L_{ai-num} (για το συγκεκριμένο κάθε φορά ιδιοχαρακτηριστικό a_i) και τα αποθηκεύουμε στην λίστα $L_{ai-NUMERICAL}$ που θα χρησιμοποιηθεί κατά την διαδικασία σύζευξης (*Matching Phase* στον Πίνακα 4.6). Όπως μπορείτε να δείτε στο Σχήμα 4.6, το *SubID* μίας δημοσίευσης έχει αποθηκευτεί σε διαδοχικούς κόμβους, ξεκινώντας από τον P_i έως τον P_j για το διάστημα $[10, 50]$. Στη συνέχεια, μία δημοσίευση με τιμή ίση με 30 καταφθάνει στο σύστημα, εντοπίζουμε τον κόμβο που διαχειρίζεται αυτή την τιμή χρησιμοποιώντας την συνάρτηση κατακερματισμού με διατήρηση τάξης, από όπου συλλέγουμε το αποθηκευμένο *SubID*.

Στην προσέγγιση *virtual Range Tree Forest* και συγκεκριμένα κατά τη διαδικασία συλλογής των *SubIDs*, (*SubID Lists Collection Phase* στον Πίνακα 4.5⁴) υπολογίζουμε το μικρότερο δυνατό διάστημα που συμπεριλαμβάνει την τιμή της δημοσίευσης και διαχειρίζεται από ένα φύλλο του δέντρου. Για παράδειγμα,

⁴Στον Πίνακα 4.5 παρουσιάζουμε τον αλγόριθμο συλλογής των *SubIDs* και σύζευξης των δημοσιεύσεων με τις συνδρομές, με την υποδομή *Pyramidus* ρυθμισμένη ώστε να διαχειρίζεται αριθμητικού τύπου ιδιοχαρακτηριστικά με την προσέγγιση *virtual Range Tree Forest* και ιδιοχαρακτηριστικά με συμβολοσειρές με την *stateless* προσέγγιση

Σχήμα 4.7: Επεξεργασία συνδρομής και σύζευξη αυτής με σχετική δημοσίευση, με την προσέγγιση virtual Range Tree Forest.

εάν η τιμή στη δημοσίευση είναι ίση με 3, τότε το μικρότερο δυνατό διάστημα που περικλείει την τιμή 3 στο δέντρο του Σχήματος 4.7 είναι το διάστημα $[3, 4)$. Στο συγκεκριμένο Σχήμα, μπορείτε να δείτε ότι ήδη έχει αποθηκευθεί το *SubID* της συνδρομής Sub1 όπου είχε οριστεί περιορισμός διαστήματος με τιμή $[2, 5)$. Χρησιμοποιώντας τον σχετικό αλγόριθμο διάσπασης διαστήματος, το $[2, 5)$ έχει διασπαστεί στα υποδιαστήματα $[2, 4)$ και $[4, 5)$ και έτσι το *SubID*₁ έχει αποθηκευθεί στους *Tnode*_{2|1} και *Tnode*_{3|4} (ή στους *Tnode*_{[2,4)} και *Tnode*_{[4,5)} αντίστοιχα).

Στη συνέχεια, υπολογίζουμε τα διαστήματα για τα οποία είναι υπεύθυνοι οι κόμβοι από τη ρίζα μέχρι τον *Tnode*_{[3,4)}: $[3, 4)$, $[2, 4)$, $[0, 4)$, και $[0, 8)$. Κάθε ένα από αυτά τα διαστήματα μαζί με το όνομα του ιδιοχαρακτηριστικού χρησιμοποιούνται σαν όρισμα στην συνάρτηση κατακερματισμού ώστε να εντοπίσουμε τους κόμβους που έχουν αποθηκευμένα *SubIDs* συνδρομών που συζευγνύουν με την τιμή της δημοσίευσης. Από αυτούς τους κόμβους συλλέγουμε τα *SubIDs* που βρίσκονται στη λίστα L_{a-num} . Στο συγκεκριμένο παράδειγμα του Σχήματος 4.7, συλλέγουμε το *SubID*₁ από τον κόμβο *Tnode*_{2|1}.

Αριθμητικού Τύπου Ιδιοχαρακτηριστικά και η Stateful Προσέγγιση

Στην προσέγγιση stateful Range Tree Forest η λογική είναι ακριβώς η ίδια όπως και στην προσέγγιση virtual Range Tree Forest. Η μόνη διαφορά είναι ότι το κάθε ένα δέντρο κτίζεται πάνω στο δίκτυο ομοτίμων εταίρων, με τους κόμβους του δέντρου να φιλοξενούνται σε κόμβους του δικτύου.

Τα οφέλη που μπορούμε να αποκομίσουμε από αυτή τη σχεδιαστική επιλογή, εστιάζονται στο γεγονός ότι οι κόμβοι στο μονοπάτι που ακολουθεί η δημοσίευση βρίσκονται ένα βήμα μακριά ο ένας από τον άλλο

(σε αντίθεση με τα $O(\log(N))$ βήματα που απαιτούνται στην virtual *Range Tree Forest* προσέγγιση), κάτι που μπορεί να μειώσει δραματικά το συνολικό αριθμό βημάτων που απαιτούνται.

Ιδιοχαρακτηριστικά με Συμβολοσειρές και η Stateless Προσέγγιση

Στην stateless προσέγγιση για την επεξεργασία ιδιοχαρακτηριστικών με συμβολοσειρές, ξεκινάμε τη συλλογή των *SubIDs* με την εξέταση κάθε ενός ιδιοχαρακτηριστικού a_i , εντοπίζοντας αρχικά τον κόμβο που έχει αποθηκευμένα τα *SubIDs* των συνδρομών που έχουν δηλώσει περιορισμό ισότητας στην τιμή $v(a_i)$ (με το να χρησιμοποιήσουμε την συνάρτηση κατακερματισμού και την τιμή $v(a_i)$ και να εντοπίσουμε τον κόμβο με το συγκεκριμένο αναγνωριστικό). Στη συνέχεια συλλέγουμε τα *SubIDs* που βρίσκονται σε αυτό τον κόμβο στη λίστα L_{ai-eq} και τα μεταφέρουμε στη λίστα $L_{ai-EQUALITY}$ που θα χρησιμοποιηθεί κατά την διαδικασία σύζευξης (*Matching Phase* στον Πίνακα 4.6).

Εκτός από τους περιορισμούς ισότητας, θα πρέπει να εντοπίσουμε και τις συνδρομές που έχουν περιορισμούς προθέματος ή επιθέματος στις τιμές των αντιστοίχων ιδιοχαρακτηριστικών της δημοσίευσης. Ο εντοπισμός των συνδρομών με περιορισμό προθέματος στο ιδιοχαρακτηριστικό a_i επιτυγχάνεται με το να επικοινωνήσουμε με του κόμβους με αναγνωριστικό $nodeID = DHT.hash(prefix(v(a_i), j))$ με το j να κυμαίνεται από 1 έως l (όπου l είναι το μέγεθος της συμβολοσειράς $v(a_i)$). Περισσότερες πληροφορίες μπορεί να βρει κανείς στον Πίνακα 4.5). Η συνάρτηση $prefix(v(a_i), j)$ επιστρέφει το πρόθεμα μεγέθους j χαρακτήρων από την συμβολοσειρά $v(a_i)$. Από αυτούς τους κόμβους συλλέγουμε τις λίστες $L_{ai-pref}$ όπου τελικά τις αποθηκεύουμε στην λίστα $L_{ai-PREFIX}$.

Η ίδια διαδικασία επαναλαμβάνεται για τη συλλογή των *SubIDs* των συνδρομών με περιορισμό επιθέματος, αφού όμως χρησιμοποιήσουμε τη συνάρτηση $suffix(v(a_i), j)$ όπου επιστρέφει το επίθεμα μεγέθους j χαρακτήρων από την συμβολοσειρά $v(a_i)$. Από τους κόμβους που πρόκειται να εντοπίσουμε σε αυτό το βήμα, συλλέγουμε τις λίστες $L_{ai-suff}$ τις οποίες και αποθηκεύουμε στην λίστα $L_{ai-SUFFIX}$.

Ιδιοχαρακτηριστικά με Συμβολοσειρές και η Stateful Προσέγγιση

Στην stateful προσέγγιση, η επεξεργασία των δημοσιεύσεων ξεκινάει με τον εντοπισμό της ρίζας του κατάλληλου δέντρου για κάθε ιδιοχαρακτηριστικό της δημοσίευσης. Στη συνέχεια ακολουθούμε το μονοπάτι προς τον κόμβο του δέντρου που η ετικέτα του είναι ίση με την συμβολοσειρά της δημοσίευσης. Κατά την διάσχιση αυτή του δέντρου, οι κόμβοι που ανήκουν στο μονοπάτι ενδεχομένως να έχουν αποθηκευμένα *SubIDs* συνδρομών με περιορισμούς προθέματος στην τιμή της δημοσίευσης. Αυτό είναι αποτέλεσμα των ιδιοτήτων που χαρακτηρίζουν αυτά τα δέντρα, και συγκεκριμένα το γεγονός ότι ένας οποιοσδήποτε κόμβος του δέντρου έχει ετικέτα με πρόθεμα την ετικέτα του πατέρα του. Ειδικότερα, ο τελευταίος κόμβος στο

Event Processing

Notation

$prefix(x, j)$: j-characters-long prefix of string x, $suffix(x, j)$: j-characters-long suffix of string x

$N_{a-sub-i}$: number of attributes defined in a subscription i, $N_{list-sub-i}$: number of collected lists that $SubID_i$ is stored

$L_{ai-\{EQUALITY, PREFIX, SUFFIX, NUMERICAL\}}$ List with candidate $SubIDs$ for string equality, prefix, suffix, and numerical predicate matching, $L_{matching}$: List of $SubIDs$ that match the event

SubID Lists Collection Phase

1. for every $event_j$ arriving at the system
2. for every string attribute a_i in the $event_j$
3. go to node $DHT.hash(v(a_i))$
4. get L_{ai-eq} and store it in $L_{ai-EQUALITY}$
5. for $j=1$ to character length of $v(a_i)$
6. go to node $DHT.hash(prefix(v(a_i), j))$
7. get the list $L_{ai-pref}$ of $SubIDs$ and store it to $L_{ai-PREFIX}$
8. go to node $DHT.hash(suffix(v(a_i), j))$
9. get the list $L_{ai-suff}$ of $SubIDs$ and store it to $L_{ai-SUFFIX}$
10. for every numerical attribute a_i in the $event_j$
11. compute the smallest sub-range R including $v(a_i)$
12. compute the set of sub-ranges in the path from $Tnode_r$ towards the root
13. for each such sub-range r_j
14. go to node $DHT.hash(r_j|a_i)$
15. get the list L_{ai-num} of $SubIDs$ and store it to L_{ai-NUM}

Πίνακας 4.5: Συλλογή των $SubIDs$ των συνδρομών που πιθανόν να συζευγνύουν με την δημοσίευση.

μονοπάτι (ο κόμβος $Tnode_{v(a_i)}$) ενδέχεται να έχει αποθηκευμένα και $SubIDs$ με περιορισμούς ισότητας. Έτσι, από όλους κόμβους του μονοπατιού συλλέγουμε τις λίστες $L_{ai-pref}$ ενώ από τον $Tnode_{v(a_i)}$ συλλέγουμε επιπλέον και τη λίστα L_{ai-eq} όπου στη συνέχεια συγκεντρώνονται στις λίστες $L_{ai-PREFIX}$ και $L_{ai-EQUALITY}$ αντίστοιχα.

Με σκοπό να εντοπίσουμε τα $SubIDs$ των συνδρομών με περιορισμούς επιθέματος, αναστρέφουμε την τιμή της δημοσίευσης και ακολουθούμε την ίδια διαδικασία συλλέγοντας τις λίστες $L_{ai-suff}$ από τους κόμβους του νέου μονοπατιού όπου τελικά συγκεντρώνονται στην λίστα $L_{ai-SUFFIX}$ για περαιτέρω επεξεργασία.

4.3.2 Σύζευξη των Συνδρομών με τις Δημοσιεύσεις

Το επόμενο βήμα έχει να κάνει με τη σύζευξη των δημοσιεύσεων με τις συνδρομές (Πίνακας 4.6, *Matching Phase*). Μέχρι αυτό το σημείο, έχουμε συλλέξει στον κόμβο όπου κατέφθασε η δημοσίευση, τέσσερις λίστες με αναγνωριστικά συνδρομών, για κάθε ένα από τα ιδιοχαρακτηριστικά της δημοσίευσης.

Ας θεωρηθεί ότι το αναγνωριστικό $SubID_1$ βρίσκεται σε αρκετές από τις λίστες που έχουμε συλλέξει και ότι η συνδρομή (με αναγνωριστικό $SubID_1$) αποτελείται από $N_{a-sub-1}$ ιδιοχαρακτηριστικά. Η τιμή $N_{a-sub-1}$, αποτελεί μέρος τους αναγνωριστικού $SubID_1$ και συγκεκριμένα είναι το πεδίο c_3 που δηλώνει

Event Matching

Matching Phase

1. for every $SubID_i$ found in the collected Lists
2. count the defined attributes from the c_3 field: $N_{a-sub-i}$
3. count the number of lists where $SubID_i$ is stored: $N_{list-sub-i}$
4. if $N_{a-sub-i}$ equals $N_{list-sub-i}$ we have a match
5. remove $SubID_i$ from all lists
6. store $SubID_i$ in the Matching list $L_{matching}$

Delivery Phase

1. for every $SubID_i$ in the $L_{matching}$ list
2. contact the origin node that keeps the subscription,
in order to inform (deliver the event to) the interested client

Πίνακας 4.6: Σύζευξη της δημοσίευσης με τις συνδρομές και παράδοση της στους συνδρομητές.

τον αριθμό των ιδιοχαρακτηριστικών της συνδρομής (Περισσότερες λεπτομέρειες μπορεί να βρει κανείς στην ενότητα 3.2).

Μπορεί εύκολα να αποδειχθεί ότι μία συνδρομή συζευγνύει με τη δημοσίευση εάν βρίσκεται σε ακριβώς $N_{a-sub-1}$ από αυτές τις λίστες. Αυτό είναι αποτέλεσμα του γεγονότος ότι απαραίτητη συνθήκη για να συζευγνύει μία συνδρομή με μία δημοσίευση είναι να ικανοποιούνται όλοι οι περιορισμοί της συνδρομής από τις τιμές της δημοσίευσης. Το ότι ένα συγκεκριμένο $SubID$ βρίσκεται σε μία λίστα που προκύπτει από την επεξεργασία ενός ιδιοχαρακτηριστικού αποδεικνύει ότι ο περιορισμός της συνδρομής ικανοποιείται από την τιμή της δημοσίευσης για το συγκεκριμένο ιδιοχαρακτηριστικό. Εάν το $SubID$ βρεθεί σε αριθμό λιστών ίσο με τον αριθμό των ιδιοχαρακτηριστικών της συνδρομής, τότε όλοι οι περιορισμοί της συνδρομής έχουν ικανοποιηθεί από τις τιμές της δημοσίευσης, και άρα καταλήγουμε σε σύζευξη.

Τα $SubIDs$ που προκύπτουν από αυτή την διαδικασία αποτελούν τα αναγνωριστικά των συνδρομών που συζευγνύουν με την τιμή της δημοσίευσης. Έτσι επεξεργαζόμαστε περαιτέρω κάθε ένα $SubID$ ώστε να εντοπίσουμε τον αντίστοιχο συνδρομητή και να αποστείλουμε τη συγκεκριμένη δημοσίευση (Πίνακας 4.6, *Delivery Phase*). Δεδομένου του ότι αυτή η μαζική αποστολή της δημοσίευσης (με την βοήθεια για παράδειγμα της τεχνολογίας IP Multicast) προς όλους του παραλήπτες/συνδρομητές διεξάγεται στον κόμβο όπου κατέφθασε η δημοσίευση (ενδεχομένως από ένα χρήστη που είναι άμεσα συνδεδεμένος με αυτό τον κόμβο) και δεδομένου ότι τέτοιοι κόμβοι θα υπάρχουν διασκορπισμένοι σε ολόκληρο το δίκτυο, μας κάνει να πιστεύουμε ότι η διαδικασία παράδοσης της δημοσίευσης δεν θα προκαλεί την δημιουργία σημείων συμφόρησης στο μοντέλο που ακολουθούν οι προσεγγίσεις μας.

Παράδειγμα 4 Σύζευξη δημοσιεύσεων με συνδρομές.

Έστω ότι η υποδομή *Pyracanthus* είναι ρυθμισμένη να λειτουργεί χρησιμοποιώντας την προσέγγιση *virtual Range Tree Forest* για την επεξεργασία ιδιοχαρακτηριστικών αριθμητικού τύπου και την *stateless* προσέγγιση για την επεξεργασία των ιδιοχαρακτηριστικών με συμβολησοιφές. Ας υποθέσουμε ότι οι παρακάτω συνδρομές

είναι αποθηκευμένες στην *Pyracanthus* ενώ καταφθάνουν στο σύστημα δύο δημοσιεύσεις με τις παρακάτω τιμές:

$$\begin{aligned} Sub1 &= \{SubID_1 \mid a_1:MB = '*LNZ', a_2:IP = '150.*'\} \\ Sub2 &= \{SubID_2 \mid a_1:MB = 'TZLNZ', a_2:IP = '160.140.*'\} \\ Sub3 &= \{SubID_3 \mid a_3:OS = 'Linux', a_4:CPU = [2000, 3000]\} \\ Event1 &= \{a_1:MB = 'TZLNZ', a_2:IP = '150.140.130.120'\} \\ Event2 &= \{a_3:OS = 'Linux', a_4:CPU = 2600\} \end{aligned}$$

Η διαδικασία ξεκινάει με τη πρώτη δημοσίευση *Event1* και το πρώτο ιδιοχαρακτηριστικό της *MB* (a_1) όπου προσπαθούμε να εντοπίσουμε τα αναγνωριστικά των συνδρομών με περιορισμό ισότητα στην τιμή του ιδιοχαρακτηριστικού $v(a_1) = '*LNZ'$. Συλλήθουμε λοιπόν την λίστα L_{a1-eq} από τον κόμβο με αναγνωριστικό $DHT.hash('TZLNZ')$, όπου περιέχει μόνο το $SubID_2$. Συνεχίζουμε με την συλλογή των λιστών $L_{a1-pref}$ και $L_{a1-suff}$ που περιέχουν αναγνωριστικά συνδρομών με περιορισμούς προθέματος και επιθέματος αντίστοιχα στην τιμή του ιδιοχαρακτηριστικού a_1 . Αναζητάμε τις $L_{a1-pref}$ λίστες στους κόμβους με αναγνωριστικό ίσο με το αποτέλεσμα της συνάρτησης κατακερματισμού με όρισμα τις τιμές: 'T', 'TZ', 'TZL', 'TZLN' και 'TZLNZ'. Όλες οι λίστες που βρέθηκαν σε αυτούς τους κόμβους δεν περιέχουν κανένα στοιχείο επειδή δεν υπάρχει καμία συνδρομή σε αυτό το παράδειγμα με τέτοιου είδους περιορισμό προθέματος. Συνεχίζουμε με την συλλογή των $L_{a1-suff}$ λιστών επικοινωνώντας με τους κόμβους με αναγνωριστικό που προκύπτει από την χρήση της συνάρτησης κατακερματισμού με όρισμα τις τιμές: 'Z', 'NZ', 'LNZ', 'ZLNZ', και 'TZLNZ' όπου βρίσκουμε το $SubID_1$ στη λίστα $L_{a1-suff}$ του κόμβου με αναγνωριστικό $DHT.hash('LNZ')$. Εξετάζουμε και το δεύτερο ιδιοχαρακτηριστικό της δημοσίευσης (a_2) με την ίδια μεθοδολογία, οπότε τελικά προκύπτουν οι παρακάτω λίστες:

$$\begin{aligned} L_{a1-EQUALITY}.. \{SubID_2\} & \quad L_{a2-EQUALITY}.. \{empty\} \\ L_{a1-PREFIX}.. \{empty\} & \quad L_{a2-PREFIX}.. \{SubID_1\} \\ L_{a1-SUFFIX}.. \{SubID_1\} & \quad L_{a2-SUFFIX}.. \{empty\} \end{aligned}$$

Συνεχίζουμε με το ιδιοχαρακτηριστικό *OS* (a_3) της δεύτερης δημοσίευσης το οποίο είναι τύπου συμβολιοσειράς οπότε επεξεργάζεται με τον ίδιο τρόπο όπως και τα a_1 και a_2 . Για το ιδιοχαρακτηριστικό *CPU* (a_4) που είναι αριθμητικού τύπου, ας υποθέσουμε ότι το πεδίο ορισμού του είναι $D_{a4} = [0, 10000]$ και η ακρίβεια του $pr(a_4) = 1$. Χρησιμοποιούμε την προσέγγιση *virtual Range Tree Forest* και συγκεκριμένα το δέντρο T_{a4} με βαθμό $g=5$ και ύψος $H=6$, για την επεξεργασία του.

Ας δούμε πρώτα σε ποιους κόμβους θα αποθηκευθεί το $SubID_3$ της συνδρομής *Sub3* που έχει περιορισμό στο ιδιοχαρακτηριστικό a_4 . Χρησιμοποιώντας τον αλγόριθμο διάσπασης διαστημάτων και δεδομένων των χαρακτηριστικών του δέντρου, το διάστημα που ορίζεται στην συνδρομή *Sub3* ($v(a_4) = [2000, 3000]$) θα τεμαχιστεί

στα εξής υποδιαστήματα: $[2000, 2125)$, $[2125, 2250)$, $[2250, 2375)$, $[2375, 2500)$, **$[2500, 2625)$** , $[2625, 2750)$, $[2750, 2875)$, και $[2875, 3000)$. Αυτά στη συνέχεια χρησιμοποιούνται μαζί με το όνομα του ιδιοχαρακτηριστικού a_4 σαν ορίσματα στην συνάρτηση κατακερματισμού, ώστε να εντοπιστούν οι κόμβοι όπου θα αποθηκευθεί το αναγνωριστικό $SubID_3$.

Κατά την επεξεργασία του συγκεκριμένου ιδιοχαρακτηριστικού της δημοσίευσης, αρχικά υπολογίζουμε το μικρότερο δυνατό διάστημα τιμών που αντιστοιχίζεται με το δέντρο T_{a_4} και περιλαμβάνει την τιμή 2600, το διάστημα $[2600, 2601)$. Στη συνέχεια, χρησιμοποιώντας τον αλγόριθμο υπολογισμού των διαστημάτων που αντιστοιχίζονται με τους κόμβους που ανήκουν στο μονοπάτι από τον $Tnode_{[2600, 2601)}$ μέχρι τη ρίζα $Tnode_{[0, 15625)}$ (που έχει παρουσιαστεί στην ενότητα 4.1.2), καταλήγουμε στα εξής διαστήματα: $[2600, 2601)$, $[2600, 2605)$, $[2600, 2625)$, **$[2500, 2625)$** , $[2500, 3125)$, $[0, 3125)$, και $[0, 15625)$. Χρησιμοποιούμε λοιπόν αυτά τα διαστήματα κατά ανάλογο με την επεξεργασία της συνδρομής τρόπο, ώστε να επικοινωνήσουμε με τους αντίστοιχους κόμβους και να συλλέξουμε τα αναγνωριστικά $SubIDs$ που είναι αποθηκευμένα στη λίστα L_{a_4-num} .

Όπως μπορείτε να δείτε κατά την επεξεργασία της συνδρομής $Sub3$ χρησιμοποιήσαμε το διάστημα $[2500, 2625)$ για να αποθηκεύσουμε το $SubID_3$. Επίσης κατά την επεξεργασία της δημοσίευσης ένα από τα διαστήματα που χρησιμοποιήσαμε είναι και το $[2500, 2625)$. Έτσι από τον κόμβο με αναγνωριστικό $nodeID=DHT.hash([2500, 2625)|a_4)$ συλλέγουμε την λίστα L_{a_4-num} που περιέχει το $SubID_3$. Μετά λοιπόν την επεξεργασία και των δύο ιδιοχαρακτηριστικών της δημοσίευσης, προκύπτουν οι παρακάτω λίστες:

$$\begin{aligned} L_{a_3-EQUALITY}.. \{SubID_3\} \quad L_{a_4-NUMERICAL}.. \{SubID_3\} \\ L_{a_3-PREFIX}.. \{empty\} \\ L_{a_3-SUFFIX}.. \{empty\} \end{aligned}$$

Το επόμενο βήμα αποτελεί την διαδικασία σύζευξης των συνδρομών των οποίων τα $SubIDs$ έχουμε συλλέξει, με τις δύο δημοσιεύσεις. Από το πεδίο c_3 των $SubIDs$ που έχουμε στη διάθεση μας μπορούμε να δούμε ότι οι συνδρομές $Sub1$, $Sub2$, και $Sub3$ έχουν δηλώσει δύο ιδιοχαρακτηριστικά η κάθε μία. Δεδομένου ότι τα $SubID_1$ και $SubID_3$ έχουν βρεθεί σε δύο λίστες, προκύπτει ότι αυτές οι δύο συνδρομές συζευγνύουν με τις τιμές των δημοσιεύσεων. Από την άληθη, η συνδρομή με αναγνωριστικό $SubID_2$ έχει βρεθεί μόνο σε μία λίστα οπότε δεν συζευγνύει με την δημοσίευση. Έτσι τα $SubID_1$ και $SubID_3$ επεξεργάζονται περαιτέρω ώστε να παραδοθούν στους αντίστοιχους συνδρομητές οι δημοσιεύσεις $Event1$ και $Event2$. $\diamond$

4.4 Αυτόνομη Οργάνωση των Δενδρικών Δομών στις Stateful

Προσεγγίσεις

Σε ένα οποιοδήποτε δίκτυο ομοτίμων εταιρών οι κόμβοι συνδέονται στο δίκτυο και αποχωρούν από αυτό ελεύθερα, χωρίς να υπάρχει κεντρικός έλεγχος. Σε ένα τέτοιο περιβάλλον θα πρέπει να εξασφαλίσουμε ότι οι δενδρικές δομές που διατηρούμε στις stateful προσεγγίσεις της υποδομής *Pyracanthus*, θα οργανώνονται αυτόνομα και θα διατηρούν την συνεκτικότητα τους. Έτσι, αν κάποιος *Tnodes* στο εσωτερικό ενός δέντρου πάψουν να λειτουργούν (είτε επειδή ο κόμβος του δικτύου που τους φιλοξενεί έχει αποχωρήσει με τη θέληση του, ή έχει υποστεί κάποια βλάβη), θα πρέπει να εξασφαλίσουμε ότι τα υποδέντρα που βρίσκονται κάτω από αυτούς θα καταφέρουν να ξανασυνδεθούν στο υπόλοιπο δέντρο.

Στα περισσότερα δίκτυα ομοτίμων εταιρών ή σε επεκτάσεις τους (όπως για παράδειγμα το Past/Pastry [44]), εφαρμόζεται η τεχνική της ανάπτυξης και διατήρησης αντιγράφων. Έτσι, για παράδειγμα στο Past/Pastry, αντιγράφονται τα περιεχόμενα ενός κόμβου σε $k/2$ κόμβους με αριθμητικά μεγαλύτερο αναγνωριστικό και σε $k/2$ κόμβους με μικρότερο. Δεδομένης μιας τέτοιας λειτουργικότητας ολόκληρη η δομή ενός *Tnode* (με τους πίνακες δρομολόγησης του, τα αποθηκευμένα *SubIDs*) θα φιλοξενείται και σε k άλλους κόμβους. Προκειμένου να αντιμετωπίσουμε μία πιθανή βλάβη, θα μπορούσαμε απλά να επισκεφθούμε ένα από τους k κόμβους και να ανακτήσουμε τον *Tnode* που έχουμε χάσει, επιδιορθώνοντας έτσι την συνεκτικότητα του δέντρου. Επιπλέον, μπορούμε να αναπτύξουμε ένα επιπρόσθετο επίπεδο αντιγραφής (replication), όπου πλέον τα δέντρα στην υποδομή *Pyracanthus* θα έχουν πολλαπλά αντίγραφα.

Ο μηχανισμός αυτόνομης οργάνωσης των δενδρικών δομών κατά την άφιξη ή την αναχώρηση των κόμβων, ενεργοποιείται κατά την εξυπηρέτηση των αιτήσεων (είτε αποθήκευσης ενός *SubID*, ή ανάκτησης του). Δηλαδή, κατά την δρομολόγηση μίας αίτησης, όταν φτάσουμε σε ένα σημείο του δέντρου όπου ο κόμβος που δείχνει ο πίνακας δρομολόγησης του *Tnode*, είναι απόν, θα πρέπει να ξεκινήσουμε μίας διαδικασία για την διόρθωσή του.

4.4.1 Άφιξη Κόμβων

Κατά την άφιξη ενός νέου κόμβου, έστω Y , αυτός θα πάρει τη θέση που του αναλογεί στο χώρο διευθύνσεων του DHT δικτύου και πλέον θα εξυπηρετεί τις αιτήσεις που του αντιστοιχούν με βάση το αναγνωριστικό του, που μέχρι πρότινος εξυπηρετούνταν από άλλο κόμβο, έστω Z . Εφόσον το DHT δίκτυο λειτουργεί με consistent hashing, όλα τα αντικείμενα που είναι αποθηκευμένα στον Y και αντιστοιχούν στον Z (στην περίπτωση μας οι *Tnodes* και τα *SubIDs* τους) θα μεταφερθούν στον Z .

Κατά την δρομολόγηση μίας αίτησης πάνω σε ένα από τα δέντρα, και όταν φτάσουμε στον πατέρα ενός

από τους $Tnodes$ που πλέον έχει μεταφερθεί στον Z , ο πίνακας δρομολόγησης του πατέρα θα δείχνει στον Y . Προωθώντας την αίτηση στον Y θα παρατηρήσουμε ότι εκεί δεν υπάρχει ο αντίστοιχος $Tnode$ (μιας και πλέον βρίσκεται στον Z). Για να τον εντοπίσουμε, εκτελούμε ένα DHT lookup χρησιμοποιώντας τη συνάρτηση κατακερματισμού και την ετικέτα του $Tnode$ που αναζητούμε σαν όρισμα, κάτι που θα μας οδηγήσει στον κόμβο Z . Στη συνέχεια, διορθώνεται ο πίνακας δρομολόγησης του πατέρα στην αντίστοιχη θέση ώστε πλέον να δείχνει στον Z , και η αίτηση συνεχίζει να προωθείται στο δέντρο.

4.4.2 Αναχώρηση Κόμβων

Κατά την εθελούσια αναχώρηση ενός κόμβου, έστω X , και δεδομένου ότι το DHT δίκτυο λειτουργεί με consistent hashing, όλα τα αντικείμενα που είναι αποθηκευμένα στον X θα μεταφερθούν στον κόμβο με αναγνωριστικό αριθμητικά κοντινότερο σε αυτόν, έστω Z . Ο κόμβος Z είναι πλέον υπεύθυνος για τις αιτήσεις που προορίζονταν για τον X .

Κατά την δρομολόγηση μίας αίτησης πάνω σε ένα από τα δέντρα, υποθέστε ότι αυτή φτάνει στον κόμβο που φιλοξενεί τον πατέρα του $Tnode$ που βρισκόταν στον X . Κατά την προώθηση της αίτησης στον X θα διαπιστώσουμε ότι ο X πλέον δεν υπάρχει και άρα θα πρέπει να βρούμε τον αντικαταστάτη του. Αυτό επιτυγχάνεται, εκτελώντας ένα DHT lookup χρησιμοποιώντας τη συνάρτηση κατακερματισμού και την ετικέτα του απόντα $Tnode$ σαν όρισμα, κάτι που θα μας οδηγήσει στον κόμβο Z . Έτσι ενημερώνουμε τον πίνακα δρομολόγησης του πατέρα ώστε να δείχνει στον Z όπου πλέον βρίσκεται και λειτουργεί ο $Tnode$ που φιλοξενοούνταν στον X .

Στην περίπτωση όπου ο κόμβος X έχει υποστεί βλάβη και δεν έχει καταφέρει να μεταφέρει στον Z τα αντικείμενα του, θα πρέπει να εξασφαλίσουμε ότι ο Z θα φιλοξενήσει τον $Tnode$ που αποχώρησε. Εάν το DHT δίκτυο εφαρμόζει την τεχνική της αντιγραφής, τότε στον κόμβο Z θα υπάρχει ήδη το $Tnode$ που αποχώρησε με όλα τα αποθηκευμένα σε αυτό $SubIDs$ μιας και ο Z εξ ορισμού ανήκει στο σύνολο k των κόμβων που διατηρούν αντίγραφα του X . Αν δεν προσφέρεται τέτοια λειτουργικότητα, θα πρέπει να κατασκευάσουμε στον Z το κατάλληλο $Tnode$ και να επισκεφθούμε ένα από τα αντίγραφα δέντρα (replica tree) στο οποίο ανήκει ώστε να ανακτήσουμε τα $SubIDs$ που του αναλογούν.

4.5 Θεωρητική Ανάλυση Απόδοσης

Σε αυτή την ενότητα παραθέτουμε την θεωρητική ανάλυση απόδοσης των προσεγγίσεων που μόλις παρουσιάσαμε και συνθέτουν την υποδομή *Pyracanthus* για την επεξεργασία ιδιοχαρακτηριστικών με αριθμητικές τιμές και συμβολοσειρές. Συγκεκριμένα, παρουσιάζουμε την πολυπλοκότητα κατά την επεξεργασία συν-

Επεξεργασία Συνδρομών με Αριθμητικού Τύπου Ιδιοχαρακτηριστικά	
<i>OP-DHT</i>	$O(\log(N)) + R $
<i>Virtual Range Tree Forest</i>	$O(g \times \log g(R) \times \log(N))$
<i>Stateful Range Tree Forest</i>	$O(g \times \log g(R) \times \log(N))$
Επεξεργασία Δημοσιεύσεων με Αριθμητικού Τύπου Ιδιοχαρακτηριστικά	
<i>OP-DHT</i>	$O(\log(N))$
<i>Virtual Range Tree Forest</i>	$H \times O(\log(N))$
<i>Stateful Range Tree Forest</i>	$H + O(\log(N))$

Πίνακας 4.7: Πολυπλοκότητα επεξεργασίας συνδρομών και δημοσιεύσεων με αριθμητικού τύπου ιδιοχαρακτηριστικά.

δρομών και δημοσιεύσεων ως προς τον αριθμό βημάτων που απαιτούνται (hop count ή αλλιώς το πλήθος των μηνυμάτων - message count) για όλες προσεγγίσεις που προορίζονται για την επεξεργασία αριθμητικών τιμών (Πίνακας 4.7) και συμβολοσειρών (Πίνακας 4.8).

Θα πρέπει να σημειωθεί ότι τα αποτελέσματα που παραθέτουμε εδώ αφορούν την επεξεργασία ενός ιδιοχαρακτηριστικού. Είναι προφανές, ότι εφόσον μία δημοσίευση ή συνδρομή αποτελείται από περισσότερα ιδιοχαρακτηριστικά, θα πρέπει να πολλαπλασιάσουμε τα αποτελέσματα που παραθέτουμε εδώ με μία σταθερά ώστε να έχουμε την προβλεπόμενη απόδοση σε επίπεδο συνδρομών ή δημοσιεύσεων.

4.5.1 Επεξεργασία Αριθμητικών Τιμών

Επικεντρώνουμε την προσοχή μας στην επεξεργασία διαστημάτων τιμών μιας και όλες οι άλλες περιπτώσεις αποτελούν υποσύνολο αυτής. Για παράδειγμα ο περιορισμός ισότητας μπορεί να θεωρηθεί και να διαχειριστεί όπως ένα διάστημα R με μέγεθος $|R| = 1$. Επίσης, ο περιορισμός $v(a_i) \geq v$ μπορεί εύκολα να θεωρηθεί σαν ένα διάστημα με τιμή $R = [v, E_{ai}]$ για το ιδιοχαρακτηριστικό a_i με πεδίο ορισμού $D_{ai} = [S_{ai}, E_{ai}]$.

Η Προσέγγιση *OP-DHT*

Κατά την επεξεργασία των συνδρομών και εφόσον ο περιορισμός στο ιδιοχαρακτηριστικό a_i ορίζεται με ένα διάστημα τιμών (π.χ. $v(a_i) = R = [l, h]$ με ακρίβεια $pr(a_i)$), θα πρέπει να επικοινωνήσουμε με $|R| = \frac{h-l}{pr(a_i)}$ κόμβους⁵. Έτσι, με την προσέγγιση *OP-DHT* θα πρέπει να εκτελέσουμε $O(\log(N))$ βήματα για να εντοπίσουμε τον πρώτο κόμβο (υπεύθυνο για την μικρότερη τιμή του υπό εξέταση διαστήματος) και στη συνέχεια $|R|$ επιπλέον βήματα. Άρα, συνολικά απαιτούνται $O(\log(N)) + |R|$ βήματα κατά την επεξεργασία ενός ιδιοχαρακτηριστικού της συνδρομής.

Κατά την επεξεργασία των δημοσιεύσεων θα πρέπει να επικοινωνήσουμε με ένα μόνο κόμβο ώστε να συλλέξουμε τα *SubIDs* συνδρομών που συζευγνύουν με την δημοσίευση. Αυτή η διαδικασία (δηλ. ένα

⁵ Αυτή είναι η χειρότερη δυνατή περίπτωση μιας και είναι πιθανό συνεχόμενες τιμές να αντιστοιχίζονται με το ίδιο αναγνωριστικό κόμβου.

απλό DHT lookup) απαιτεί την εκτέλεση $O(\log(N))$ βημάτων.

Η Προσέγγιση *Virtual Range Tree Forest*

Σε αυτή τη προσέγγιση, κατά την επεξεργασία των συνδρομών, ένα οποιοδήποτε διάστημα R θα πρέπει να διασπασθεί στο κατάλληλο σύνολο από υποδιαστήματα εφαρμόζοντας τον αλγόριθμο διάσπασης διαστημάτων που παρουσιάστηκε στον Πίνακα 4.1. Το πλήθος των υποδιαστημάτων εξαρτάται από το μέγεθος τους διαστήματος $|R|$ και τον βαθμό g του δέντρου που θα χρησιμοποιηθεί και είναι ίσο με $O(2 \times g \times \lfloor \log g |R| \rfloor - g)$. Για κάθε υποδιάστημα θα πρέπει να επικοινωνήσουμε με ένα κόμβο, πράγμα που συνεπάγεται ότι θα πρέπει να εκτελέσουμε συνολικά $O(g \times \log g(|R|) \times \log(N))$ για την επεξεργασία ενός διαστήματος τιμών.

Κατά την επεξεργασία των δημοσιεύσεων, ακολουθούμε ένα μονοπάτι στο δέντρο προς ένα από τα φύλλα του, διατρέχοντας έτσι ολόκληρο το δέντρο. Δεδομένου ότι το ύψος του δέντρου είναι ίσο με H θα πρέπει να επικοινωνήσουμε με $H + 1$ κόμβους και έτσι προκύπτει ότι κατά την επεξεργασία των δημοσιεύσεων απαιτούνται $H \times O(\log(N))$ βήματα.

Ας υποθέσουμε ότι ένα ιδιοχαρακτηριστικό αριθμητικού τύπου a_i έχει πεδίο ορισμού $|D_{ai}|$. Οι μικρότερες δυνατές τιμές για τον βαθμό g και το ύψος H του δέντρου T_{ai} θα πρέπει να είναι τέτοιες ώστε να ικανοποιούν την ανισότητα $|D_{ai}| \leq g^H$ (για δεδομένες τιμές των g και D_{ai} , το H θα πρέπει να είναι τέτοιο ώστε $H \geq \log_g(D_{ai})$ ενώ για δεδομένες τιμές των H και D_{ai} , το g θα πρέπει να είναι τέτοιο ώστε $g \geq 10^{\frac{\log(|D_{ai}|)}{H}}$). Οι προσεγγίσεις που στηρίζονται στο *Range Tree Forest* εξαιτίας του σχεδιασμού τους, επιτρέπουν την επιλογή εναλλακτικών συνδυασμών τιμών για τα H και g έτσι ώστε να βελτιωθεί η απόδοση κατά την επεξεργασία είτε των δημοσιεύσεων είτε των συνδρομών.

Η επιλογή μιας μικρής τιμής για τον βαθμό του δέντρου θα βελτιώσει την απόδοση κατά την επεξεργασία των συνδρομών. Όμως, θα επηρεάσει αρνητικά την απόδοση κατά την επεξεργασία των δημοσιεύσεων μιας και το ύψος του δέντρου θα πρέπει να αυξηθεί ώστε να ικανοποιείται η ανισότητα $|D_{ai}| \leq g^H$. Έτσι, καλείται πλέον ο σχεδιαστής της υποδομής *PyraCanthus* να επιλέξει τις κατάλληλες τιμές των g και H ανάλογα με τις απαιτήσεις της εφαρμογής που πρόκειται να φιλοξενήσει, ώστε να επιτύχει γρηγορότερη επεξεργασία των δημοσιεύσεων ή των συνδρομών.

Η Προσέγγιση *Stateful Range Tree Forest*

Στην *stateful Range Tree Forest* προσέγγιση, όπως και στην *stateless*, θα χρειαστεί να επικοινωνήσουμε με $O(2 \times g \times \lfloor \log g |R| \rfloor - g)$ κόμβους κατά την επεξεργασία των συνδρομών, πράγμα που οδηγεί στην εκτέλεση $O(g \times \log g(|R|) \times \log(N))$ βημάτων.

Επιπλέον, θα πρέπει να λάβουμε υπόψη μας και το κόστος δημιουργίας του μονοπατιού προς την ρίζα

Επεξεργασία Συνδρομών με Ιδιοχαρακτηριστικά Τύπου Συμβολοσειράς	
Stateless Προσέγγιση	$O(\log(N))$
Stateful Προσέγγιση	$O(\log(N))$
Επεξεργασία Δημοσιεύσεων με Ιδιοχαρακτηριστικά Τύπου Συμβολοσειράς	
Stateless Προσέγγιση	$l \times O(\log(N))$
Stateful Προσέγγιση	$l + O(\log(N))$

Πίνακας 4.8: Πολυπλοκότητα επεξεργασίας συνδρομών και δημοσιεύσεων με ιδιοχαρακτηριστικά τύπου συμβολοσειράς.

του δέντρου, το οποίο μεταφράζεται σε $d \times O(\log(N))$ βήματα, όπου d είναι το βάθος στο οποίο βρίσκεται ο κόμβος που είναι υπεύθυνος για ένα συγκεκριμένο υποδιάστημα (που αποτελεί μέρος του συνόλου των υποδιαστημάτων στο οποίο τεμαχίστηκε το διάστημα της συνδρομής). Αυτά τα βήματα προστίθενται στα βήματα που απαιτούνται κατά την επεξεργασία μιας συνδρομής μόνο όταν το μονοπάτι δεν έχει ήδη δημιουργηθεί, πράγμα που προβλέπεται ότι θα συμβαίνει στα πρώτα στάδια λειτουργίας του συστήματος. Ένα μεγάλο ποσοστό των συνδρομών, δεν θα χρειαστεί να κατασκευάσουν το μονοπάτι ή θα πρέπει να κατασκευάσουν ένα μικρό μόνο τμήμα του.

Η επεξεργασία των συνδρομών σε αυτή την προσέγγιση απαιτεί την επικοινωνία με $H + 1$ κόμβους, επειδή ακριβώς θα πρέπει να διατρέξουμε ένα μονοπάτι από την ρίζα μέχρι ένα φύλλο του δέντρου. Έτσι, θα πρέπει να εκτελέσουμε $O(\log(N))$ βήματα για να εντοπίσουμε την ρίζα του δέντρου και H επιπλέον βήματα μέχρι το φύλλο, πράγμα που έχει σαν αποτέλεσμα την εκτέλεση συνολικά $H + O(\log(N))$ βημάτων.

4.5.2 Επεξεργασία Συμβολοσειρών

Η Stateless Προσέγγιση

Η επεξεργασία συνδρομών με ιδιοχαρακτηριστικά τύπου συμβολοσειράς στην stateless προσέγγιση, απαιτεί την εκτέλεση $O(\log(N))$ βημάτων αφού θα πρέπει να επικοινωνήσουμε με ένα μόνο κόμβο και να αποθηκεύσουμε εκεί το αναγνωριστικό *SubID*.

Κατά την επεξεργασία των δημοσιεύσεων και πιο συγκεκριμένα κατά την φάση συλλογής των *SubIDs*, για κάθε ιδιοχαρακτηριστικό a_i , θα πρέπει σε πρώτη φάση να επικοινωνήσουμε με ένα κόμβο για να συλλέξουμε τα *SubIDs* των συνδρομών με περιορισμό ισότητα στη τιμή της δημοσίευσης. Στη συνέχεια, αν υποθέσουμε ότι η τιμή της δημοσίευσης στο ιδιοχαρακτηριστικό a_i αποτελείται από l_i χαρακτήρες, θα πρέπει να επικοινωνήσουμε με l_i κόμβους για να συλλέξουμε τα *SubIDs* συνδρομών με περιορισμό προθέματος και l_i για περιορισμό επιθέματος. Ο συνολικός λοιπόν αριθμός βημάτων που απαιτούνται κατά την επεξεργασία των δημοσιεύσεων είναι $l \times O(\log(N))$ (μιας και για κάθε επικοινωνία με κάποιο κόμβο πρέπει να εκτελέσουμε $O(\log(N))$ βήματα) όπου l είναι το μέσο μέγεθος των συμβολοσειρών.

Η Stateful Προσέγγιση

Η επεξεργασία των συνδρομών απαιτεί την εκτέλεση $O(\log(N))$ βημάτων για τον εντοπισμό του κόμβου όπου θα αποθηκευτεί το *SubID*. Επιπλέον, τουλάχιστον για την επεξεργασία των αρχικών συνδρομών (πρωτού δηλαδή το σύστημα μας φτάσει σε μία κατάσταση ηρεμίας, όπου τα περισσότερα μονοπάτια θα έχουν ήδη κατασκευαστεί), θα πρέπει να εκτελέσουμε και μία σειρά βημάτων για την κατασκευή των μονοπατιών που είναι ίσος με $l \times O(\log(N))$, για συμβολοσειρές μεγέθους l χαρακτήρων. Όπως ήδη έχουμε αναφέρει, αυτό το κόστος κατασκευής των μονοπατιών, δεν επηρεάζει όλες τις συνδρομές παρά μόνο αυτές που πρόκειται να επεξεργαστούν όταν ακόμα τα δέντρα θα βρίσκονται σε κατάσταση σχηματισμού.

Η επεξεργασία των δημοσιεύσεων απαιτεί την εκτέλεση $O(\log(N))$ βημάτων για τον εντοπισμό της ρίζας του δέντρου και στη συνέχεια l βημάτων ακολουθώντας το μονοπάτι μέχρι τον κόμβο του οποίου η ετικέτα είναι ίση με την τιμή του ιδιοχαρακτηριστικού της δημοσίευσης. Στους ενδιάμεσους κόμβους του μονοπατιού αναζητάμε συνδρομές με περιορισμό προθέματος στην τιμή της δημοσίευσης ενώ στον τελικό κόμβο για περιορισμό προθέματος και ισότητας. Την ίδια διαδικασία ακολουθούμε και για την συλλογή των *SubIDs* των συνδρομών με περιορισμό επιθέματος στην τιμή της δημοσίευσης. Έτσι, συνολικά απαιτούνται $l + O(\log(N))$ βήματα για την επεξεργασία των δημοσιεύσεων.

4.6 Βελτίωση της Πολυδιάστατης Επεξεργασίας των

Δημοσιεύσεων

Η επέκταση που ακολουθεί στοχεύει στη μείωση του κόστους επεξεργασίας των *SubIDs* που έχουν συγκεντρωθεί, κατά την διαδικασία σύζευξης της δημοσίευσης με τις συνδρομές. Συγκεκριμένα, επιθυμούμε να καταναείμουμε την επεξεργασία των δημοσιεύσεων σε όσο το δυνατόν περισσότερους κόμβους. Η διαδικασία που προτείνεται εδώ είναι ανεξάρτητη από την προσέγγιση που έχουμε επιλέξει για την επεξεργασία των δημοσιεύσεων/συνδρομών και επηρεάζει μόνο τον τρόπο που συλλέγονται τα *SubIDs*. Έτσι, μπορεί να εφαρμοστεί σε όλες τις προσεγγίσεις που αναπτύχθηκαν στο πλαίσιο της υποδομής *Pyracanthus*.

4.6.1 Κεντριοποιημένη Σύζευξη

Στις προσεγγίσεις που μέχρι τώρα έχουμε παρουσιάσει για την επεξεργασία των δημοσιεύσεων, τα *SubIDs* συλλέγονται και συγκεντρώνονται στον κόμβο όπου γεννήθηκε η δημοσίευση (ή στον κόμβο που είναι συνδεδεμένος ένας χρήστης που δημοσιεύει πληροφορίες) που από εδώ και στο εξής θα αποκαλούμε «συντονιστή». Έτσι, κατά την επεξεργασία της δημοσίευσης, για κάθε ιδιοχαρακτηριστικό της, εντοπίζουμε τους κόμβους

Σχήμα 4.8: Κεντροποιημένη Σύζευξη δημοσίευσης με συνδρομές.

που σχετίζονται με την τιμή του ιδιοχαρακτηριστικού και συλλέγουμε τις λίστες των *SubIDs* στον κόμβο συντονιστή, όπως μπορεί κανείς να δει στο Σχήμα 4.8. Ο συντονιστής στη συνέχεια αναλαμβάνει να εκτελέσει τοπικά τον αλγόριθμο σύζευξης. Επειδή η σύζευξη εκτελείται στον συντονιστή, αποκαλούμε αυτή τη προσέγγιση *Κεντροποιημένη Σύζευξη* (Coordinated Matching).

Ανάλογα με τα χαρακτηριστικά του συστήματος (ιδιαίτερα των κατανομών δημοτικότητας των τιμών των ιδιοχαρακτηριστικών αλλά και των ίδιων των ιδιοχαρακτηριστικών), ένας μεγάλος αριθμός των *SubIDs* που έχουν συγκεντρωθεί στον συντονιστή αντιστοιχούν σε συνδρομές που δεν συζευγνύουν με τη δημοσίευση. Η αποστολή αυτών των *SubIDs* μέσω του δικτύου υπερφορτώνει το ίδιο το δίκτυο και αυξάνει τις υπολογιστικές απαιτήσεις στο συντονιστή προκειμένου να εκτελέσει τοπικά την σύζευξη των δημοσιεύσεων με τις συνδρομές.

4.6.2 Κατανεμημένη Σύζευξη

Μία πρώτη απόπειρα ώστε να αποφύγουμε ή έστω να μειώσουμε τις επιπτώσεις του παραπάνω προβλήματος, είναι να εκτελέσουμε την διαδικασία σύζευξης κατανεμημένα, βήμα προς βήμα, όπως μπορείτε να δείτε στο Σχήμα 4.9. Η βασική ιδέα σε αυτή την προσπάθεια (*Κατανεμημένη Σύζευξη* - Distributed Matching), είναι να ταξινομήσουμε τα ιδιοχαρακτηριστικά της δημοσίευσης με βάση την προβλεπόμενη επιλεκτικότητα τους (selectivity) στηριζόμενοι στις τιμές τους. Το πρόβλημα προσδιορισμού της επιλεκτικότητας της τιμής ενός ιδιοχαρακτηριστικού της δημοσίευσης, αποτελεί μία διαδικασία ιδιαίτερα δαπανηρή. Τα τελευταία χρόνια έχουν παρουσιαστεί εργασίες που σχετίζονται με τον υπολογισμό των χαρακτηριστικών των δικτύων

Σχήμα 4.9: Κατανεμημένη Σύζευξη δημοσίευσης με συνδρομές.

ομοτίμων εταιρών με ιδιαίτερα αποδοτικό τρόπο ([59, 67, 72]). Ένα από αυτά τα χαρακτηριστικά για παράδειγμα θα μπορούσε να είναι το κατά προσέγγιση πλήθος των *SubIDs* που είναι αποθηκευμένα σε κάθε κόμβο, για κάθε ιδιοχαρακτηριστικό a_i . Ωστόσο, εάν θέλουμε να είμαστε πιο ακριβής, θα μπορούσαμε να ξεκινήσουμε μία φάση επικοινωνίας του συντονιστή με όλους τους εμπλεκόμενους κόμβους ανά ιδιοχαρακτηριστικό, ώστε οι τελευταίοι να ενημερώσουν τον συντονιστή για το σύνολο των *SubIDs* που πρόκειται να του στείλουν. Με αυτή την πληροφορία ο συντονιστής θα μπορούσε στη συνέχεια να υπολογίζει την επιλεκτικότητα κάθε ιδιοχαρακτηριστικού (που είναι προφανές ότι εξαρτάται τόσο από την δημοτικότητα του ίδιου του ιδιοχαρακτηριστικού όσο και από την δημοτικότητα της τιμής του) και να τα ταξινομήσει έτσι ώστε να επεξεργαστούμε πρώτα αυτά τα ιδιοχαρακτηριστικά που πρόκειται να επιστρέψουν σχετικά μικρές λίστες από *SubIDs*.

Στην Κατανεμημένη Σύζευξη, η συλλογή των *SubIDs* συνδυάζεται με την σύζευξη όπου στην Κεντροποιημένη Σύζευξη εκτελείται στον συντονιστή. Συγκεκριμένα, έχοντας ταξινομήσει τα ιδιοχαρακτηριστικά της συνδρομής με βάση την επιλεκτικότητά τους, επεξεργαζόμαστε πρώτα τα ιδιοχαρακτηριστικά με μεγάλη επιλεκτικότητα. Στέλνουμε λοιπόν την αίτηση συλλογής των *SubIDs* στους κόμβους που συσχετίζονται με την τιμή του πρώτου ιδιοχαρακτηριστικού. Αφού συγκεντρωθούν τα *SubIDs* σε ένα από αυτούς τους κόμβους, αποστέλλονται στους κόμβους που συσχετίζονται με την τιμή του δεύτερου ιδιοχαρακτηριστικού και η διαδικασία επαναλαμβάνεται μέχρι την επεξεργασία του τελευταίου ιδιοχαρακτηριστικού (αυτού με την μικρότερη επιλεκτικότητα που προβλέπεται ότι θα επιστρέψει τη μεγαλύτερη σε πλήθος *SubIDs* λίστα σε σχέση με όλα τα άλλα ιδιοχαρακτηριστικά).

Σε κάθε βήμα i αυτής της διαδικασίας (όπου ένα βήμα αντιστοιχεί στην επεξεργασία ενός ιδιοχαρακτηριστικού) τα $SubIDs$ που συλλέγονται για το συγκεκριμένο ιδιοχαρακτηριστικό από ένα σύνολο κόμβων, συγκεντρώνονται στη λίστα $LocalList_i$. Τα $SubIDs$ που συγκεντρώθηκαν στα προηγούμενα βήματα βρίσκονται στην λίστα $GlobalList_{i-1}$. Από την λίστα $LocalList_i$ αφαιρούμε τα $SubIDs$ των συνδρομών που έχουν ορίσει κάποιο ιδιοχαρακτηριστικό⁶ που ήδη έχουμε εξετάσει στο βήμα j με $j < i$, αλλά δεν βρίσκονται στην λίστα $GlobalList_{i-1}$. Από την $GlobalList_{i-1}$ αφαιρούμε αυτά τα $SubIDs$ που έχουν ορίσει περιορισμό στο ιδιοχαρακτηριστικό που εξετάζουμε στο βήμα i αλλά δεν βρίσκονται στην λίστα $LocalList_i$. Όλα τα $SubIDs$ που απομένουν και από τις δύο λίστες, συνενώνονται στη λίστα $GlobalList_i$ και αποστέλλονται στους κόμβους που συσχετίζονται με την τιμή του επόμενου στην κατάταξη επιλεκτικότητας ιδιοχαρακτηριστικού.

Για παράδειγμα, η συνδρομή με αναγνωριστικό $SubID_1$ έχει ορίσει περιορισμούς στα ιδιοχαρακτηριστικά a_1 και a_3 . Αν υποθέσουμε ότι η συνδρομή αυτή συζευγνύει με την δημοσίευση και αν θεωρήσουμε ότι επεξεργαζόμαστε πρώτα το a_1 (βήμα 1) και στη συνέχεια το a_3 (βήμα 2), τότε στο βήμα 2, το $SubID_1$ θα έπρεπε να βρίσκεται και στην $GlobalList_1$ αλλά και στην $LocalList_2$. Για μία άλλη συνδρομή με αναγνωριστικό $SubID_2$ και περιορισμούς στα a_1 και a_3 , η εμφάνιση του $SubID_2$ στην $GlobalList_1$ αλλά όχι στην $LocalList_2$ θα σήμαινε ότι η συνδρομή συζευγνύει με την δημοσίευση μερικώς (μόνο στη τιμή του ιδιοχαρακτηριστικού a_1) και άρα δεν μπορεί να θεωρηθεί ότι επιτυγχάνεται σύζευξη. Από την άλλη αν το $SubID_2$ εμφανιζόταν στην $LocalList_2$ αλλά όχι στην $GlobalList_1$ θα σήμαινε ότι η συνδρομή συζευγνύει με την δημοσίευση μόνο στη τιμή του ιδιοχαρακτηριστικού a_2 και άρα ούτε και σε αυτή την περίπτωση θα μπορούσε να θεωρηθεί ότι επιτυγχάνεται σύζευξη.

Η παραπάνω διαδικασία συνεχίζεται μέχρι και την επεξεργασία του τελευταίου ιδιοχαρακτηριστικού, όπου η λίστα $GlobalList_k$ (αν θεωρήσουμε ότι η δημοσίευση αποτελείται από k ιδιοχαρακτηριστικά) αποστέλλεται στον συντονιστή που στη συνέχεια έχοντας στην διάθεση του τα $SubIDs$ των συνδρομών που συζευγνύουν με την δημοσίευση, αναλαμβάνει να αποστείλει τη δημοσίευση στους συνδρομητές.

4.6.3 Υβριδική Σύζευξη

Το αδύνατο σημείο της Κατανεμημένης Σύζευξης είναι η άσκοπη διάδοση από ένα σημείο της διαδικασίας και μετά, αυτών των $SubIDs$ που αντιστοιχούν σε συνδρομές των οποίων τα ιδιοχαρακτηριστικά έχουν ήδη ελεγχθεί και έχουμε καταλήξει σε σύζευξη με την δημοσίευση.

Σε κάθε βήμα i μπορούμε να αποφανθούμε για το αν ένα συγκεκριμένο $SubID$ που ανήκει στην

⁶Πληροφορία σχετικά με το ποια ιδιοχαρακτηριστικά έχουν δηλωθεί σε μία συνδρομή μπορεί να εξαχθεί από το ίδιο το $SubID$ εάν αντικαταστήσουμε το πεδίο c_3 με ένα πίνακα αποτελούμενο από A bits με αρχικές τιμές ίσο με το 0. Κάθε δηλωθέν ιδιοχαρακτηριστικό αναγκάζει την αντίστοιχη θέση του πίνακα να πάρει την τιμή 1.

Σχήμα 4.10: Υβριδική Σύζευξη δημοσίευσης με συνδρομές.

λίστα $GlobalList_i$ μπορεί να σταλεί πίσω στον συντονιστή. Αυτή είναι η περίπτωση όπου η συγκεκριμένη συνδρομή δεν έχει ορίσει περιορισμό σε κανένα από τα ιδιοχαρακτηριστικά που πρόκειται να εξετάσουμε σε επόμενα βήματα j (με $j > i$).

Στην Υβριδική Σύζευξη (Hybrid Matching) εκμεταλλευόμαστε αυτή την παρατήρηση. Όταν φτάσουμε σε ένα βήμα όπου όλοι οι περιορισμοί μιας συνδρομής έχουν ικανοποιηθεί από τις τιμές της δημοσίευσης, αποστέλλουμε το $SubID$ της στον συντονιστή. Τα υπόλοιπα $SubIDs$ ακολουθούν την πορεία της Κατανεμημένης Σύζευξης, όπως μπορείτε να δείτε στο Σχήμα 4.10.

4.7 Πειραματική Αξιολόγηση και Μελέτη Απόδοσης

4.7.1 Χαρακτηριστικά και Παράμετροι Πειραματικής Διάταξης

Εκτελέσαμε μια σειρά πειραμάτων προσομοιώνοντας ένα DHT δίκτυο στο οποίο εφαρμόσαμε τις προσεγγίσεις για την επεξεργασία των συνδρομών και των δημοσιεύσεων που έχουν παρουσιαστεί σαν δομικά στοιχεία της υποδομής *Pyracanthus*. Το DHT δίκτυο αποτελείται από 1000 κόμβους (brokers). Παρόμοια αποτελέσματα προκύπτουν και για δίκτυα με πολύ περισσότερους κόμβους.

Χρησιμοποιήσαμε την κατανομή δημοτικότητας Zipf τόσο για τα ιδιοχαρακτηριστικά του μοντέλου δημοσιεύσεων/συνδρομών όσο και για τις τιμές του πεδίου ορισμού τους. Στην κατανομή Zipf η συχνότητα εμφάνισης (F_i) του i -οστού ταξινομημένου αντικειμένου ορίζεται από την σχέση $F_i = \frac{1}{i^\alpha}$. Τυπικές τιμές

Σχήμα 4.11: Η κατανομή δημοτικότητας Zipf για $\vartheta = 1.0$.

για την παράμετρο ϑ είναι $0.0 \leq \vartheta \leq 1.2$, όπου με υψηλές τιμές της ϑ προκύπτουν κατανομές με αρκετά μεγάλη κλήση, ενώ με μικρές τιμές η κατανομή πλησιάζει την ομοιόμορφη κατανομή (για $\vartheta = 0.0$ η κατανομή είναι ομοιόμορφη). Χαρακτηριστικό της Zipf κατανομής, είναι ότι για μεγάλες τιμές της παραμέτρου ϑ ($\vartheta > 0.8$) ένα πολύ μικρό ποσοστό των αντικειμένων απολαμβάνει το μεγαλύτερο ποσοστό των προσπελάσεων ή εμφανίσεων. Τέτοια συμπεριφορά παρατηρείται γενικά στα συστήματα δημοσιεύσεων/συνδρομών, όπου ένα πολύ μικρό ποσοστό των τιμών των ιδιοχαρακτηριστικών είναι ιδιαίτερα δημοφιλές (όπως ακριβώς συμβαίνει και στις αιτήσεις ανάκτησης των αντικείμενων στον παγκόσμιο ιστό [25]). Στο Σχήμα 4.11 μπορείτε να δείτε την κατανομή Zipf για $\vartheta = 1.0$.

Το μοντέλο δημοσιεύσεων/συνδρομών που εφαρμόσαμε σ' αυτά τα πειράματα, αποτελείται από 10 ιδιοχαρακτηριστικά. Ο ακριβής αριθμός αυτών που εμπλέκονται στη δήλωση των δημοσιεύσεων και των συνδρομών καθορίζεται από την κατανομή Zipf της δημοτικότητας των ιδιοχαρακτηριστικών με τιμή $\vartheta = 0.8$, πράγμα που έχει ως αποτέλεσμα την δήλωση κατά μέσο όρο 3.6 ιδιοχαρακτηριστικών ανά δημοσίευση ή συνδρομή. Τα αποτελέσματα που πήραμε για μεγαλύτερες τιμές του ϑ είναι ανάλογες με αυτά για $\vartheta = 0.8$. Η δημοτικότητα των τιμών των ιδιοχαρακτηριστικών ακολουθεί και αυτή την Zipf κατανομή και μιας και παίζει σημαντικό ρόλο (για παράδειγμα στην κατανομή του φόρτου αποθήκευσης των *SubIDs*, ή του φόρτου προσπέλασης των κόμβων), μεταβάλουμε την κλίση της δίνοντας τιμές στη παράμετρο ϑ που κυμαίνονται από 0.0 μέχρι 1.2.

Τα αποτελέσματα μας προέκυψαν κατά την επεξεργασία περίπου 6,000,000 *SubIDs* και περίπου 1,000,000 αιτήσεων για ανάκτηση αυτών. Ο ακριβής αριθμός εξαρτάται από την κλίση των σχετικών κατανομών δημοτικότητας Zipf (που μεταφράζεται στην τιμή της ϑ). Τα μισά ιδιοχαρακτηριστικά είναι

αλφαριθμητικού τύπου, όπου στο 50% αυτών ορίζονται περιορισμοί διαστημάτων τιμών και στο υπόλοιπο 50% περιορισμοί ισότητας. Από τα ιδιοχαρακτηριστικά τύπου συμβολοσειράς, στο 40% ορίζονται περιορισμοί ισότητας, στο 30% περιορισμοί προθέματος και στο 30% περιορισμοί επιθέματος.

4.7.2 Απαιτήσεις σε Χώρο Αποθήκευσης

Μετρήσαμε τις απαιτήσεις σε αποθηκευτικό χώρο που μπορεί να προκύψουν με την εφαρμογή των προσεγγίσεων *virtual Range Tree Forest* και *OP-DHT* μετρώντας το μέσο πλήθος των αποθηκευμένων *SubIDs* στους κόμβους του δικτύου. Οι απαιτήσεις σε χώρο αποθήκευσης των προσεγγίσεων *virtual* και *stateful Range Tree Forest* είναι ακριβώς οι ίδιες μιας και το ίδιο σύνολο κόμβων πρόκειται να χρησιμοποιηθεί για την αποθήκευση ενός *SubID* και στις δύο περιπτώσεις.

Χρησιμοποιήσαμε διαστήματα τιμών κατά την δήλωση των συνδρομών με μέγεθος ομοιόμορφα κατανοημένο μεταξύ 1 και 200 (που κατά μέσο όρο αντιστοιχεί σε διαστήματα μεγέθους ίσο με 2.5% του πεδίου ορισμού κάθε ιδιοχαρακτηριστικού που είναι το διάστημα $[1, 8000]$). Τα αποτελέσματα που πήραμε για διαφορετικού μεγέθους διαστήματα μπορούν να υπολογιστούν κατά προσέγγιση από τα αποτελέσματα που παρουσιάζουμε εδώ.

Τα αποτελέσματα μας δείχνουν ότι η προσέγγιση *virtual Range Tree Forest* απαιτεί μία τάξη μεγέθους λιγότερο αποθηκευτικό χώρο από την προσέγγιση *OP-DHT*. Η μέση τιμή των αποθηκευμένων *SubIDs* στους κόμβους υπολογίστηκε σε 473.158 με την προσέγγιση *virtual Range Tree Forest* και σε 6703.6 με την *OP-DHT*. Αυτό μπορεί να εξηγηθεί από το γεγονός ότι δεδομένου ενός διαστήματος R θα πρέπει να αποθηκεύσουμε το συγκεκριμένο *SubID* σε περίπου $g \times \log(|R|)$ κόμβους με την προσέγγιση *Range Tree Forest*, ενώ ο αντίστοιχος αριθμός με την *OP-DHT* προσέγγιση είναι $|R|$. Αυτή η ανωτερότητα της *virtual Range Tree Forest* προσέγγισης, συνοδεύεται από όχι και τόσο καλή απόδοση κατά την επεξεργασία των δημοσιεύσεων όπως θα δούμε παρακάτω.

4.7.3 Εξισορρόπηση του Φόρτου Αποθήκευσης και Ανάκτησης των SubIDs

Στα περισσότερα πραγματικά συστήματα δημοσιεύσεων/συνδρομών η κατανομή της δημοτικότητας των τιμών των ιδιοχαρακτηριστικών (και συνεπώς η κατανομή των προσβάσεων στους κόμβους για την ανάκτηση των *SubIDs*) δεν είναι ομοιόμορφη. Έτσι είναι δυνατό να προκληθούν ανισορροπίες στην κατανομή του φόρτου (αποθήκευσης ή πρόσβασης και ανάκτησης) με ένα μικρό ποσοστό των κόμβων να δέχεται τον περισσότερο φόρτο.

Με σκοπό λοιπόν να μετρήσουμε τον βαθμό εξισορρόπησης του φόρτου στο δίκτυο, εκτελέσαμε μια σειρά πειραμάτων, μεταβάλλοντας την κλίση της κατανομής της δημοτικότητας των τιμών των ιδιοχαρακτηριστικών

Σχήμα 4.12: Η καμπύλη Lorenz που απεικονίζει την κατανομή του φόρτου σε ένα δίκτυο ομοτίμων εταίρων, με συντελεστή $G = 0.5$.

με την τιμή της παραμέτρου ϑ να κυμαίνεται από 0.0 έως 1.2 .

Όπως είναι γνωστό μία τεχνική που λειτουργεί αμυντικά στην ανάπτυξη ανισορροπιών είναι η δημιουργία και διατήρηση αντιγράφων της πληροφορίας που διαχειριζόμαστε, που στην περίπτωση μας είναι τα αναγνωριστικά των συνδρομών. Η τεχνική αυτή έχει εφαρμοστεί σε αρκετές εφαρμογές που λειτουργούν πάνω σε DHT δίκτυα ομοτίμων εταίρων (όπως για παράδειγμα το Past/Pastry σύστημα αρχείων [44]). Εφαρμόσαμε λοιπόν την τεχνική της διατήρησης αντιγράφων με βαθμό αντιγραφής (δηλ. το πλήθος των αντιγράφων - replica - ενός *SubID* που αποθηκεύονται στο δίκτυο) ίσο με 0 (κανένα αντίγραφο), 5 και 9.

Σε αυτή την ενότητα δεν παρουσιάζουμε αποτελέσματα σχετικά με την *stateful Range Tree Forest* προσέγγιση, μιας και η απόδοση της είναι ακριβώς ίδια με την *virtual Range Tree Forest* προσέγγιση

Η μετρική απόδοσης μας εδώ είναι ο συντελεστής Gini - G . Οι καμπύλες Lorenz και ο συντελεστής Gini έχουν προταθεί πρόσφατα σαν μια ιδιαίτερα κατάλληλη μετρική για την μέτρηση της εξισορρόπησης του φόρτου εργασίας σε ένα δίκτυο ομοτίμων εταίρων [79]. Οι καμπύλες Lorenz αποτελούν μία συνάρτηση των αθροιστικών τιμών ενός συνόλου αντικειμένων, ως προς την ταξινομημένη λίστα αυτών των αντικειμένων.

Στο περιβάλλον των δικτύων ομοτίμων εταίρων, τα αντικείμενα μπορεί να θεωρηθεί ότι είναι οι κόμβοι του δικτύου και οι τιμές τους ο φόρτος εργασίας σε κάθε κόμβο (Σχήμα 4.12). Όσο μεγαλύτερη είναι η απόσταση της καμπύλης από την διαγώνιο (την γραμμή της ομοιομορφίας όπου όλοι οι κόμβοι έχουν ακριβώς τον ίδιο φόρτο) τόσο μεγαλύτερες ανισορροπίες υπάρχουν. Ο συντελεστής Gini εκφράζει το λόγο της περιοχής που περικλείεται από την γραμμή ομοιομορφίας και την καμπύλη Lorenz προς ολόκληρη την περιοχή που σχηματίζεται από τους άξονες και την γραμμή ομοιομορφίας, και εκφράζεται σαν την μέση τιμή των διαφορών του φόρτου μεταξύ κάθε δυνατού ζευγαριού κόμβων διαιρούμενο με τον μέσο φόρτο :

$$G = \frac{1}{2\mu} \frac{1}{N^2} \sum_{i=1}^N \sum_{j=1}^N |ld_i - ld_j|$$

όπου ld_i είναι ο φόρτος στον κόμβο i , N είναι το πλήθος των κόμβων στο δίκτυο, και μ είναι ο μέσος φόρτος.

Η τιμή του συντελεστή Gini μεταβάλλεται από 0.0 όπου όλοι οι κόμβοι έχουν ίσο φόρτο, έως 1.0 όπου όλοι οι κόμβοι έχουν μηδενικό φόρτο, εκτός από ένα. Σαν μέτρο σύγκρισης, τιμές του G κοντά στο 0.5 θεωρούνται αποδεκτές. Συγκεκριμένα, η κατανομή του φόρτου αποθήκευσης αντικειμένων σε ένα DHT με την χρήση μίας ψευδοτυχαίας συνάρτησης κατακερματισμού (όπως η SHA-1) αντιστοιχεί σε συντελεστή Gini με τιμή ίση με 0.5.

Στο Σχήμα 4.13 παρουσιάζεται ο συντελεστής Gini που σχετίζεται με τον φόρτο αποθήκευσης των *SubIDs* χρησιμοποιώντας (i) 0 αντίγραφα, (ii) 5 αντίγραφα και (iii) 9 αντίγραφα. Χωρίς την ύπαρξη αντιγράφων, η υποδομή *Pyracanthus* αποδίδει ικανοποιητικά για σχετικά μικρές τιμές του ϑ . Όμως για κατανομές δημοτικότητας με περισσότερο απότομη κλίση (π.χ. $\vartheta = 1.2$), προκύπτουν μεγάλες ανισότητες στην κατανομή του φόρτου στο δίκτυο. Με την αποθήκευση 5 αντιγράφων για κάθε αποθηκευμένο *SubID* μπορούμε να μειώσουμε σημαντικά τις ανισότητες (κατά περίπου 1.5 φορά στην τιμή του G), ενώ με την διατήρηση 9 αντιγράφων, μπορούμε να μειώσουμε περαιτέρω τις ανισότητες όπως μπορείτε να δείτε στο Σχήμα 4.13.

Η εξισορρόπηση του φόρτου προσπέλασης ακολουθεί τις ίδιες τάσεις. Θα μπορούσε κανείς να περιμένει ότι η εφαρμογή της *virtual Range Tree Forest* προσέγγισης για την επεξεργασία των δημοσιεύσεων με ιδιοχαρακτηριστικά αριθμητικού τύπου θα προκαλούσε μεγαλύτερες ανισότητες στην κατανομή του φόρτου προσπέλασης σε σχέση με την προσέγγιση *OP-DHT*. Και αυτό, διότι στην *virtual Range Tree Forest* θα πρέπει να προσπελάσουμε όλους τους κόμβους που βρίσκονται στο μονοπάτι από την ρίζα του δέντρου μέχρι ένα από τα φύλλα του, σε αντίθεση με την προσπέλαση ενός μόνο κόμβου στην *OP-DHT* προσέγγιση, πράγμα που θα έχει ως αποτέλεσμα οι κόμβοι που βρίσκονται κοντά στην ρίζα να προσπελάζονται πιο συχνά από τους υπόλοιπους προκαλώντας έτσι μεγαλύτερες ανισότητες στη κατανομή του φόρτου πρόσβασης. Φαίνεται όμως ότι αυτό δεν ισχύει, διότι στην προσέγγιση *Range Tree Forest* ο φόρτος προσπέλασης κατανέμεται σε όλους τους κόμβους του μονοπατιού, ενώ στην *OP-DHT* αυτό μεταφράζεται στην προσπέλαση ενός μόνο κόμβου. Έτσι το αποτέλεσμα είναι ότι ο φόρτος κατανέμεται περισσότερο από ότι θα μπορούσε κάποιος να περιμένει, πράγμα που δικαιολογεί την ελάχιστη χειρότερη απόδοση του, σε σχέση με την *OP-DHT*.

Θα πρέπει να σημειωθεί ότι τα αποτελέσματα μας έδειξαν υψηλότερες μέσες τιμές του φόρτου πρόσβασης

Σχήμα 4.13: Εξισορρόπηση του φόρτου μεταβάλλοντας την κλίση της κατανομής της δημοτικότητας των τιμών και το πλήθος των αντιγράφων (replicas) *SubIDs*.

στην προσέγγιση *Range Tree Forest* σε σύγκριση με την προσέγγιση *OP-DHT* (η μέση και μέγιστη τιμή στην *Range Tree Forest* είναι 1.2 και 1.3 φορές μεγαλύτερη από αυτή στην προσέγγιση *OP-DHT*) πράγμα που υποδηλώνει ότι προσπελάζονται περισσότεροι κόμβοι ανά επεξεργασία δημοσίευσης σε σχέση με την προσέγγιση *OP-DHT*. Έτσι, συμπερασματικά, θα μπορούσαμε να πούμε ότι με την εφαρμογή της *virtual Range Tree Forest* προσέγγισης, αυξάνεται η μέση τιμή του φόρτου με σκοπό να αποφευχθούν σημεία με μεγάλες ανισότητες στο δίκτυο.

4.7.4 Κίνηση στο Δίκτυο

Το σύνολο των *SubIDs* που μεταφέρονται κατά την επεξεργασία της δημοσίευσης στον συντονιστή, αποτελεί μία σημαντική ένδειξη για τις απαιτήσεις σε εύρος ζώνης που μπορεί να έχει κάθε μια από τις προτεινόμενες προσεγγίσεις. Ο αριθμός των *SubIDs* που επιστρέφονται με την εφαρμογή των δύο προσεγγίσεων για την επεξεργασία ιδιοχαρακτηριστικών αριθμητικού τύπου (*virtual Range Tree Forest* και *OP-DHT*) είναι ακριβώς ο ίδιος. Όμως, μετρώντας τον όγκο μεταφοράς δεδομένων (σε KBytes όπως μπορείτε να δείτε στο Σχήμα 4.14), παρατηρήσαμε ότι η *virtual Range Tree Forest* προσέγγιση είναι ελάχιστα χειρότερη από την *OP-DHT* ειδικότερα όταν η κατανομή της δημοτικότητας των τιμών έχει ιδιαίτερα απότομη κλίση. Αυτό οφείλεται στο γεγονός ότι με την *virtual Range Tree Forest* προσέγγιση απαιτείται η επικοινωνία πολύ περισσότερων κόμβων σε σχέση με την *OP-DHT*.

Στο Σχήμα 4.14 επίσης, μπορεί κανείς να παρατηρήσει ότι ο συνολικός όγκος δεδομένων (*SubIDs*)

Σχήμα 4.14: Κίνηση στο δίκτυο κατά την επεξεργασία των δημοσιεύσεων.

που μεταφέρεται πάνω από τις γραμμές μεταφοράς του δικτύου για κάθε επεξεργασία δημοσίευσης, είναι σχετικά μικρός ακόμα και για απότομες κλίσεις της κατανομής της δημοτικότητας (όπου το πλήθος των *SubIDs* που επιστρέφονται στον συντονιστή είναι αρκετά μεγάλο).

4.7.5 Πλήθος Μηνυμάτων

Σε αυτή την ενότητα μετρήσαμε τον αριθμό των μηνυμάτων (ή διαφορετικά αριθμός βημάτων - hops που εκτελούνται στο δίκτυο) που απαιτούνται κατά την εκτέλεση των αλγορίθμων επεξεργασία των δημοσιεύσεων και των συνδρομών που αποτελούνται τόσο από ιδιοχαρακτηριστικά τύπου συμβολοσειράς όσο και αριθμητικού τύπου.

Για τις λύσεις που βασίζονται στις *Range Tree Forest* προσεγγίσεις, μεταβάλλουμε τον βαθμό των δέντρων (g) και το ύψος τους (H) έτσι ώστε κάθε φορά το δέντρο που προκύπτει να μπορεί να καλύψει το εύρος τιμών του κάθε ιδιοχαρακτηριστικού. Για την επεξεργασία των ιδιοχαρακτηριστικών τύπου συμβολοσειράς, μεταβάλλουμε το ύψος του δέντρου στην *stateful* προσέγγιση (ο βαθμός του δέντρου είναι σταθερός και ίσος με 5) και το μέγιστο μέγεθος των συμβολοσειρών στην *stateless* προσέγγιση.

Επεξεργασία Ιδιοχαρακτηριστικών Αριθμητικού Τύπου

Στα Σχήματα 4.15 και 4.16 παρουσιάζονται τα αποτελέσματά μας συναρτήσει του συνδυασμού των g και H κατά την επεξεργασία δημοσιεύσεων και συνδρομών με ιδιοχαρακτηριστικά αριθμητικού τύπου. Για

Σχήμα 4.15: Πλήθος των μηνυμάτων κατά την επεξεργασία των δημοσιεύσεων με αριθμητικού τύπου ιδιοχαρακτηριστικά για διαφορετικούς συνδυασμούς g/H .

παράδειγμα το πρώτο σημείο στον οριζόντιο άξονα αντιστοιχεί στην ανάπτυξη δέντρων με $g = 2$ και $H = 13$, ικανά να υποστηρίξουν πεδίο ορισμού με το πολύ $2^{13} = 8192$ τιμές (σε αυτά τα πειράματα το πεδίο ορισμού των ιδιοχαρακτηριστικών αποτελείται από 8000 ακέραιες τιμές στο διάστημα $[0, 8000)$). Το μέσο μέγεθος των διαστημάτων που χρησιμοποιούνται σε αυτή την προσομοίωση είναι 100, με τα μεγέθη να κατανέμονται ομοιόμορφα ανάμεσα στο 1 και το 200. Πειραματιστήκαμε και με μεγαλύτερα μεγέθη διαστημάτων και αποτελέσματα μας συμφωνούν με την θεωρητική μελέτη που παρουσιάσαμε στην ενότητα 4.5. Η κατανομή της δημοτικότητας των τιμών δεν επηρεάζει τον αριθμό των μηνυμάτων (επηρεάζει αντίθετα το μέγεθος των συνόλων των *SubIDs* που επιστρέφονται στον συντονιστή και συνεπώς τις απαιτήσεις σε εύρος ζώνης), και για αυτό το λόγο επιλέγουμε την τιμή $\vartheta = 0.8$ για την Zipf κατανομή τους, που είναι μία σχετικά αποδεκτή τιμή σύμφωνα με τη σχετική βιβλιογραφία [25].

Όπως έχει ήδη αναφερθεί και φαίνεται και στο Σχήμα 4.15, η επεξεργασία των δημοσιεύσεων με την προσέγγιση *OP-DHT* είναι ιδιαίτερα γρήγορη. Όμως, από την άλλη, η επεξεργασία των συνδρομών είναι πολύ δαπανηρή (Σχήμα 4.16). Οι προσεγγίσεις που βασίζονται στα *Range Tree Forest* (virtual και stateful) απολαμβάνουν πολύ καλή απόδοση κατά την επεξεργασία των συνδρομών και ικανοποιητική απόδοση κατά την επεξεργασία των δημοσιεύσεων, σε σχέση με την προσέγγιση *OP-DHT*. Στο Σχήμα 4.15, μπορείτε επίσης να παρατηρήσετε ότι η stateful *Range Tree Forest* είναι αποδοτικότερη σε σχέση με την virtual *Range Tree Forest* όπως άλλωστε ήταν αναμενόμενο.

Πρέπει επίσης να σημειωθεί ότι με την χρήση των δέντρων (virtual και stateful *Range Tree Forest* προ-

Σχήμα 4.16: Πλήθος των μηνυμάτων κατά την επεξεργασία των συνδρομών με αριθμητικού τύπου ιδιοχαρακτηριστικά για διαφορετικούς συνδυασμούς g/H .

σεγγίσεις), μπορεί κανείς να ρυθμίσει έτσι το σύστημα του, ορίζοντας κατάλληλες τιμές για τα g και H , ώστε να επιτύχει γρήγορη επεξεργασία δημοσιεύσεων ή συνδρομών. Για παράδειγμα, όταν $g = 2$ και $H = 13$ η επεξεργασία των συνδρομών είναι αποδοτικότερη από αυτή των δημοσιεύσεων, ενώ όταν $g = 10$ και $H = 4$ παρατηρούμε το αντίστροφο.

Στο Σχήμα 4.17 παρουσιάζεται η καμπύλη της σταθμισμένης μέσης τιμής των μηνυμάτων που απαιτούνται κατά την επεξεργασία δημοσιεύσεων και συνδρομών, σε ένα σύστημα όπου το 80% του συνόλου των δημοσιεύσεων και των συνδρομών είναι δημοσιεύσεις και το υπόλοιπο 20% είναι συνδρομές. Παρατηρούμε ότι η *OP-DHT* είναι η αποδοτικότερη προσέγγιση σε σχέση με τις υπόλοιπες (μίας και η επεξεργασία των δημοσιεύσεων που στην *OP-DHT* είναι ιδιαίτερα αποδοτική, κυριαρχεί). Σε περιβάλλοντα όμως, όπου είτε οι συνδρομές είναι περισσότερες από τις δημοσιεύσεις είτε τα διαστήματα τιμών είναι ιδιαίτερα μεγάλα, οι *Range Tree Forest* προσεγγίσεις είναι προτιμότερες και ιδιαίτερα η *stateful Range Tree Forest* προσέγγιση.

Επεξεργασία Ιδιοχαρακτηριστικών με Συμβολοσειρές

Κατά την επεξεργασία συνδρομών με συμβολοσειρές, οι δύο προσεγγίσεις που παρουσιάσαμε αποδίδουν το ίδιο, κάτι που αντιστοιχεί σε περίπου 7 μηνύματα (ή βήματα στο δίκτυο) και μπορεί να επιβεβαιωθεί και από την θεωρητική μελέτη που παρουσιάσαμε στον Πίνακα 4.8.

Η επεξεργασία των συνδρομών από την άλλη, δείχνει να είναι πιο αποδοτική με την εφαρμογή της *stateful* προσέγγισης, όπως μπορείτε να δείτε στο Σχήμα 4.18.

Σχήμα 4.17: Σταθμισμένη μέση τιμή του πλήθους των μηνυμάτων που απαιτούνται κατά την επεξεργασία των δημοσιεύσεων και των συνδρομών με λόγο δημοσιεύσεων/συνδρομών ίσο με 80/20, για αριθμητικού τύπου ιδιοχαρακτηριστικά.

Σχήμα 4.18: Πλήθος των μηνυμάτων που απαιτούνται κατά την επεξεργασία των δημοσιεύσεων με ιδιοχαρακτηριστικά τύπου συμβολοσειράς μεταβάλλοντας το ύψος του δέντρου (μέγιστο μέγεθος συμβολοσειρών)

Σχήμα 4.19: Σταθμισμένη μέση τιμή του πλήθους των μηνυμάτων που απαιτούνται κατά την επεξεργασία των δημοσιεύσεων και των συνδρομών (λόγος δημοσιεύσεων/συνδρομών: 80/20).

Στο Σχήμα 4.19 παρουσιάζεται η καμπύλη της σταθμισμένης μέσης τιμής των μηνυμάτων που απαιτούνται κατά την επεξεργασία των δημοσιεύσεων και των συνδρομών, με λόγο δημοσιεύσεων/συνδρομών ίσο με 80/20. Η τάσεις είναι οι ίδιες όπως και στο Σχήμα 4.18, με χαμηλότερες όμως απόλυτες τιμές, μιας και η γρήγορη επεξεργασία των δημοσιεύσεων (που κυριαρχεί εδώ) βοηθάει ώστε να μειωθούν οι τιμές.

Subscription Processing of Numerical and String Attributes (50/50 ratio)	
OP-DHT and Stateless Strings	26.11
OP-DHT and Stateful Strings	26.11
Virtual Range Tree Forest and Stateless Strings	25.22
Virtual Range Tree Forest and Stateful Strings	25.22
Stateful Range Tree Forest and Stateless Strings	25.77
Stateful Range Tree Forest and Stateful Strings	25.77
Event Processing of Numerical and String Attributes (50/50 ratio)	
OP-DHT and Stateless Strings	49.57
OP-DHT and Stateful Strings	29.90
Virtual Range Tree Forest and Stateless Strings	61.46
Virtual Range Tree Forest and Stateful Strings	41.79
Stateful Range Tree Forest and Stateless Strings	56.14
Stateful Range Tree Forest and Stateful Strings	36.47
Event and Subscription Processing (80/20 ratio)	
OP-DHT and Stateless Strings	44.88
OP-DHT and Stateful Strings	29.14
Virtual Range Tree Forest and Stateless Strings	54.21
Virtual Range Tree Forest and Stateful Strings	38.48
Stateful Range Tree Forest and Stateless Strings	50.07
Stateful Range Tree Forest and Stateful Strings	34.33

Πίνακας 4.9: Μέσο (και σταθμισμένο) πλήθος μηνυμάτων που απαιτούνται κατά την επεξεργασία των συνδρομών και των δημοσιεύσεων με ιδιοχαρακτηριστικά αριθμητικού τύπο και συμβολοσειράς.

Σχολιασμός της Απόδοσης των Προτεινόμενων Προσεγγίσεων

Στον Πίνακα 4.9 μπορείτε να δείτε την συνολική απόδοση που θα περίμενε κανείς να παρατηρήσει συνδυάζοντας τις προσεγγίσεις που προτείνουμε για την επεξεργασία αριθμητικών τιμών και συμβολοσειρών σε συνδρομές και δημοσιεύσεις.

Όπως και στα προηγούμενα πειράματά μας, στα μισά ιδιοχαρακτηριστικά ορίζονται ιδιοχαρακτηριστικά τύπου συμβολοσειράς και στα υπόλοιπα αριθμητικού τύπου. Στο πρώτο τμήμα του Πίνακα 4.9 μπορείτε να δείτε την απόδοση κατά την επεξεργασία των συνδρομών για κάθε δυνατό συνδυασμό των λύσεων που προτείνουμε, στο δεύτερο τμήμα παρουσιάζεται η απόδοση κατά την επεξεργασία των δημοσιεύσεων και στο τελευταίο τμήμα η σταθμισμένη απόδοση της επεξεργασίας των συνδρομών και των δημοσιεύσεων με λόγο 80/20 (80% δημοσιεύσεις και 20% συνδρομές). Για τις προσεγγίσεις που σχετίζονται με αριθμητικού τύπου ιδιοχαρακτηριστικά επιλέξαμε να χρησιμοποιήσουμε δέντρα με βαθμό $g = 10$ και ύψος $H = 4$, ενώ για αυτές που σχετίζονται με συμβολοσειρές, έχουμε επιλέξει το ύψος του δέντρου (ή το μέγιστο μήκος των συμβολοσειρών) να είναι ίσο με 6.

Για την επεξεργασία των δημοσιεύσεων, ο συνδυασμός της *OP-DHT* προσέγγισης με την *stateful* προσέγγιση για την επεξεργασία συμβολοσειρών φαίνεται να είναι αποδοτικότερη μιας και απαιτεί την εκτέλεση 29.9 βημάτων. Κατά την συνδυασμένη απόδοση της επεξεργασίας συνδρομών και δημοσιεύσεων, και επειδή

η επεξεργασία των δημοσιεύσεων κυριαρχεί, ο αποδοτικότερος συνδυασμός είναι και εδώ η προσέγγιση *OP-DHT* με την *stateful* προσέγγιση για τις συμβολοσειρές.

Μπορείτε επίσης να παρατηρήσετε ότι οι *stateful* προσεγγίσεις είναι αποδοτικότερες. Όμως σ' αυτές υπάρχει και το επιπλέον κόστος διατήρησης της δεντρικής δομής υπό την παρουσία συχνών αναχωρήσεων και αφίξεων κόμβων στο δίκτυο. Με την υποδομή *Pyracanthus* μπορούμε να συμφιλιώσουμε αυτές τις δύο αντικρουόμενες τάσεις. Για παράδειγμα, κάποιος θα μπορούσε να εφαρμόσει *stateful* προσέγγιση είτε για την επεξεργασία μόνο ιδιοχαρακτηριστικών τύπου συμβολοσειράς ή για ιδιοχαρακτηριστικά αριθμητικού τύπου. Ο συνδυασμός των προσεγγίσεων *stateful Range Tree Forest* για αριθμητικές τιμές με *stateless* για τις συμβολοσειρές, απαιτεί την ανταλλαγή 50 μηνυμάτων, ενώ εάν χρησιμοποιήσουμε την *stateful* προσέγγιση για τις συμβολοσειρές μπορούμε να μειώσουμε τον αριθμό μηνυμάτων στα 34.

Μια πιο προσεκτική ματιά στα αποτελέσματα μας θα μας οδηγήσει στο συμπέρασμα ότι μπορεί να αξίζει να ρυθμίσουμε έτσι την υποδομή *Pyracanthus* με *stateless* λύσεις για την επεξεργασία αριθμητικών τιμών, και *stateful* λύσεις για την επεξεργασία συμβολοσειρών, όπου μπορεί να μειώσει τον συνολικό αριθμό μηνυμάτων σε περίπου 38.

4.7.6 Πειραματική Αξιολόγηση Κατανεμημένης Επεξεργασίας Δημοσιεύσεων

Σε αυτή την ενότητα μελετάμε την απόδοση των αλγορίθμων που αναπτύξαμε στο πλαίσιο της κατανεμημένης σύζευξης των δημοσιεύσεων με τις συνδρομές. Συγκεκριμένα, μελετάμε την απόδοση της *Κατανεμημένης* και *Υβριδικής Σύζευξης* και τις συγκρίνουμε με την *Κεντρικοποιημένη Σύζευξη*.

Μελέτη Απόδοσης Σχετική με το Πλήθος των Μηνυμάτων

Με μία σύντομη ανάλυση μπορεί κανείς να αποφανθεί ότι η *Κεντρικοποιημένη Σύζευξη* είναι αποδοτικότερη σε σχέση με τις άλλες δύο από πλευράς πλήθους μηνυμάτων που θα πρέπει να αποσταλούν κατά την επεξεργασία της δημοσίευσης.

Ας υποθέσουμε ότι κατά την επεξεργασία μίας δημοσίευσης θα πρέπει να επικοινωνήσουμε και να συλλέξουμε τις λίστες με τα *SubIDs* από k κόμβους. Με την προσέγγιση της *Κεντρικοποιημένης Σύζευξης*, ο συντονιστής θα πρέπει να στείλει ένα μήνυμα σε κάθε ένα από τους k κόμβους, και στη συνέχεια αυτοί να του επιστρέψουν ένα μήνυμα με τις λίστες που κατέχουν. Έτσι, συνολικά με την *Κεντρικοποιημένη Σύζευξη* απαιτείται η αποστολή $2 \times k$ μηνυμάτων.

Στην *Κατανεμημένη Σύζευξη*, απαιτούνται συνολικά $k + 1$ μηνύματα. Είναι λιγότερα από την *Κεντρικοποιημένη Σύζευξη* κατά $k - 1$, επειδή στην *Κατανεμημένη Σύζευξη* αποφεύγουμε την επικοινωνία κάθε κόμβου με τον συντονιστή, εκτός από τον τελευταίο που επιστρέφει πίσω στον συντονιστή την λίστα με τα

SubIDs. Τέλος, με την *Υβριδική Σύζευξη* θα πρέπει να αποστέλλουμε $2 \times k$ μηνύματα, ίσο με τον αριθμό μηνυμάτων που απαιτούνται στην *Κεντρικοποιημένη Σύζευξη*. Ωστόσο, η λίστα που τελικά παραλαμβάνει ο συντονιστής περιέχει τα *SubIDs* των συνδρομών που συζευγνύουν με την δημοσίευση και έτσι δεν απαιτείται περαιτέρω επεξεργασία αυτών, όπως συμβαίνει στην *Κεντρικοποιημένη Σύζευξη*.

Απαιτήσεις σε Εύρος Ζώνης

Ίσως μία πιο σημαντική μετρική απόδοσης, αποτελεί το πλήθος των *SubIDs* που μεταδίδονται πάνω στο δίκτυο και κατ' επέκταση μία ένδειξη για τις απαιτήσεις σε κατανάλωση εύρους ζώνης. Συγκεκριμένα, η μετρική απόδοσης είναι το πλήθος των *SubIDs* που αποστέλλονται κατά την επεξεργασία μιας δημοσίευσης (όταν ένα *SubID* αποστέλλεται r φορές, τότε υπολογίζεται σαν r *SubIDs*).

Εκτελέσαμε μια σειρά πειραμάτων σε ένα δίκτυο αποτελούμενο από 128 κόμβους, και εφαρμόσαμε τις τεχνικές κατανεμημένης επεξεργασίας για 10,000 συνδρομές και 30,000 δημοσιεύσεις. Οι τιμές των ιδιοχαρακτηριστικών των συνδρομών και των δημοσιεύσεων αποτελούνται από συμβολοσειρές και εφαρμόσαμε την *Stateless* προσέγγιση για την επεξεργασία τους (ενότητα 4.2.1). Μεταβάλλαμε την κλίση (skewness) της κατανομής της δημοτικότητας των ιδιοχαρακτηριστικών (που καθορίζει πόσα ιδιοχαρακτηριστικά θα οριστούν σε μία δημοσίευση ή σε μία συνδρομή) καθώς και την κλίση της κατανομής της δημοτικότητας των τιμών των ιδιοχαρακτηριστικών. Το πεδίο ορισμού των ιδιοχαρακτηριστικών είναι πολύ μεγαλύτερο από τον αριθμό των κόμβων στο δίκτυο μας. Το πλήθος των ιδιοχαρακτηριστικών που μπορεί να ορίζονται σε μία δημοσίευση ή σε μία συνδρομή μεταβάλλεται από 1 έως 10 και εξαρτάται από την κατανομή της δημοτικότητας των ιδιοχαρακτηριστικών. Η δημοτικότητα τόσο των ιδιοχαρακτηριστικών όσο και των τιμών τους ακολουθεί την κατανομή zipf με παράμετρο ϑ που μεταβάλλεται από 0.1 (πλησιάζοντας την ομοιόμορφη κατανομή) μέχρι 1.0.

Στο Σχήμα 4.20 μεταβάλουμε την κλίση της κατανομής των τιμών των ιδιοχαρακτηριστικών, ενώ η κατανομή της δημοτικότητας των ιδιοχαρακτηριστικών ορίζεται από την τιμή $\vartheta = 0.6$. Παρατηρούμε ότι η αποδοτικότερη προσέγγιση είναι η *Κεντρικοποιημένη Σύζευξη*, που είναι ελάχιστα καλύτερη από την *Υβριδική Σύζευξη*, και σημαντικά καλύτερη από την *Κατανεμημένη Σύζευξη*. Πρέπει να σημειωθεί ότι αν και οι δύο αποδοτικότερες προσεγγίσεις έχουν σχεδόν την ίδια απόδοση, στην *Υβριδική Σύζευξη*, η φάση της σύζευξης των συνδρομών με την δημοσίευση γίνεται κατανεμημένα, με αποτέλεσμα ο συντονιστής να παραλάβει μετά το πέρας της διαδικασίας τα *SubIDs* των συνδρομών που συζευγνύουν με την δημοσίευση (σε αντίθεση με την *Κεντρικοποιημένη Σύζευξη* όπου πρέπει να επεξεργαστούμε περαιτέρω τα *SubIDs* στον συντονιστή ώστε να ανακαλύψουμε ποια *SubIDs* συζευγνύουν με την δημοσίευση). Αυτό το χαρακτηριστικό της *Υβριδική Σύζευξη* προβλέπεται ότι θα αποσυμφορήσει τον συντονιστή από το να εκτελεί την διαδικασία

Σχήμα 4.20: Πλήθος μεταφερόμενων *SubIDs* μεταβάλλοντας την κλίση της κατανομής της δημοτικότητας των τιμών των ιδιοχαρακτηριστικών.

σύζευξης τοπικά. Η *Κατανεμημένη Σύζευξη* είναι η χειρότερη γιατί όπως αναφέραμε και προηγουμένως, αρκετά *SubIDs* μεταφέρονται άσκοπα από κόμβο σε κόμβο ενώ θα μπορούσαν να αποσταλούν πίσω στον συντονιστή στο βήμα όπου επεξεργαζόμαστε και το τελευταίο τους ιδιοχαρακτηριστικό και καταλήγουμε σε σύζευξη.

Στην επόμενη σειρά πειραμάτων, προσπαθούμε να καταλάβουμε κάτω από ποιες προϋποθέσεις η *Κατανεμημένη* και η *Υβριδική* σύζευξη μπορούν να βελτιώσουν την απόδοσή τους, συγκρινόμενες με τη *Κεντρικοποιημένη* σύζευξη. Αρχικά μεταβάλλουμε την κατανομή της δημοτικότητας των ιδιοχαρακτηριστικών, με την παράμετρο ϑ να κυμαίνεται από 0.1 μέχρι 1.0. Η κατανομή της δημοτικότητας των τιμών των ιδιοχαρακτηριστικών ορίζεται από την τιμή $\vartheta = 0.5$. Όλες οι άλλες παράμετροι παραμένουν σταθερές.

Όπως μπορείτε να δείτε στο Σχήμα 4.21, η *Κεντρικοποιημένη Σύζευξη* είναι οριακά καλύτερη από την *Υβριδική Σύζευξη* σε όλες τις περιπτώσεις. Διαισθητικά μπορούμε να εξηγήσουμε αυτή τη συμπεριφορά ως εξής: αρχικά, θα πρέπει να σημειωθεί ότι όσο πιο κοντά στην ομοιόμορφη κατανομή είναι η κατανομή της δημοτικότητας των τιμών (το ϑ πλησιάζει την τιμή 0.0) τόσο μικρότερο είναι το πλήθος των *SubIDs* στις λίστες που προκύπτουν από την ένωση των λιστών $GlobalList_{i-1}$ και $LocalList_i$ σε κάθε βήμα i . Αυτό οφείλεται στο γεγονός ότι διαφορετικές συνδρομές με μεγάλη πιθανότητα, θα έχουν επιλέξει διαφορετικές τιμές για τα κοινά ιδιοχαρακτηριστικά. Έτσι, κάτω από αυτές τις συνθήκες, το φιλτράρισμα σε κάθε βήμα της *Κατανεμημένης* και της *Υβριδικής* σύζευξης θα είναι ιδιαίτερα αποδοτικό. Προκειμένου να εξακριβώσουμε εάν αυτή η παρατήρηση είναι ακριβής, εκτελέσαμε ένα ακόμα πείραμα στο οποίο τα δημοφιλή

Σχήμα 4.21: Πλήθος μεταφερόμενων *SubIDs* μεταβάλλοντας την κλίση της κατανομής της δημοτικότητας των ιδιοχαρακτηριστικών. Η κατανομή των τιμών καθορίζεται από $\vartheta = 0.5$

ιδιοχαρακτηριστικά έχουν ομοιόμορφη κατανομή δημοτικότητας τιμών, ενώ τα λιγότερο δημοφιλή έχουν κατανομή δημοτικότητας τιμών με υψηλή κλίση (το ϑ πλησιάζει την τιμή 1.0).

Όπως μπορείτε να δείτε στο Σχήμα 4.22, για μικρές τιμές της παραμέτρου ϑ όπου στις συνδρομές και στις δημοσιεύσεις ορίζονται πολλά ιδιοχαρακτηριστικά, η *Υβριδική Σύζευξη* είναι αποδοτικότερη σε σχέση με την *Κεντροποιημένη Σύζευξη*, επειδή υπάρχουν πολλά βήματα φιλτραρίσματος και επιπλέον η επεξεργασία των δημοφιλών ιδιοχαρακτηριστικών χαρακτηρίζεται από μικρού μεγέθους λίστες από *SubIDs* κάτι που βοηθάει επιπλέον το φιλτράρισμα. Καθώς η κατανομή της δημοτικότητας των ιδιοχαρακτηριστικών γίνεται πιο απότομη, ο μέσος αριθμός ιδιοχαρακτηριστικών ανά συνδρομή ή δημοσίευση μειώνεται, και συνεπώς δεν υπάρχουν αρκετά βήματα φιλτραρίσματος στην *Υβριδική Σύζευξη* ώστε να δείξει την αξία της. Όπως μπορείτε να δείτε στο Σχήμα 4.22 και οι τρεις προσεγγίσεις τείνουν να έχουν την ίδια απόδοση, όσο η κλίση της κατανομής της δημοτικότητας των ιδιοχαρακτηριστικών αυξάνεται (αύξηση της τιμής του ϑ). Μία ακόμα εξήγηση της καλής απόδοσης της *Υβριδικής Σύζευξης* όταν η τιμή του ϑ είναι σχετικά μικρή στο Σχήμα 4.22, είναι το γεγονός ότι η τομή των λιστών $GlobalList_{i-1}$ και $LocalList_i$ αποτελείται από σχετικά μικρό αριθμό *SubIDs* και άρα η *Υβριδική Σύζευξη* είναι ικανή να φιλτράρει όσο το δυνατό περισσότερο.

Χρόνος Απόκρισης Σύζευξης των Συνδρομών με τη Δημοσίευση

Κατά την διαδικασία επεξεργασίας των δημοσιεύσεων θα πρέπει να επεξεργαστούμε όλες τις λίστες που έχουμε συλλέξει κατά την επεξεργασία κάθε ιδιοχαρακτηριστικού που ορίζεται στην δημοσίευση, ώστε να

Σχήμα 4.22: Πλήθος μεταφερόμενων *SubIDs* μεταβάλλοντας την κλίση της κατανομής της δημοτικότητας των ιδιοχαρακτηριστικών. Η κατανομή των τιμών καθορίζεται από $\vartheta = 1.0$ για τα μη δημοφιλή ιδιοχαρακτηριστικά.

εντοπίσουμε εκείνα τα *SubIDs* που συζευγνύουν με την δημοσίευση. Κατά την *Κατανεμημένη* και την *Υβριδική Σύζευξη* θα πρέπει να περιμένουμε μείωση του χρόνου απόκρισης αυτής της διαδικασίας.

Εκτελέσαμε λοιπόν, μια σειρά πειραμάτων για να ποσοτικοποιήσουμε αυτή τη μείωση. Συγκεκριμένα, χρησιμοποιήσαμε το ίδιο δίκτυο αποτελούμενο από 128 κόμβους με 10,000 συνδρομές και 30,000 δημοσιεύσεις. Οι χρόνοι απόκρισης παρουσιάζονται στο Σχήμα 4.23, και προέκυψαν κατά την εκτέλεση των πειραμάτων σε ένα ηλεκτρονικό υπολογιστή με επεξεργαστή i686 στα 2.6 GHz με 1 GB μνήμη και λειτουργικό σύστημα *Linux*. Σε αυτά τα πειράματα, εισάγουμε την παράμετρο *overlapping_factor*, που καθορίζει το ποσοστό των *SubIDs* που βρίσκονται τόσο στην λίστα $GlobalList_{i-1}$ όσο και στην $LocalList_i$ σε κάθε βήμα i . Μεταβάλλαμε την παράμετρο *overlapping_factor* από 10% (που υποδηλώνει υψηλού βαθμού φιλτράρισμα σε κάθε βήμα i) μέχρι 100% (όπου δεν υφίσταται καθόλου φιλτράρισμα) και μετρήσαμε τον χρόνο απόκρισης της *Κεντροποιημένης*, της *Κατανεμημένης* και της *Υβριδικής Σύζευξης*.

Όπως μπορείτε να δείτε ο χρόνος απόκρισης της *Υβριδικής* και της *Κατανεμημένης* σύζευξης πέφτει κάτω από το 20% του χρόνου απόκρισης της *Κεντροποιημένης* σύζευξης για 10% επικάλυψη (*overlapping_factor* = 10%). Καθώς η επικάλυψη αυξάνεται, πράγμα που υποδηλώνει ότι το φιλτράρισμα σε κάθε βήμα είναι λιγότερο αποδοτικό, μιας και λιγότερα *SubIDs* αφαιρούνται από τις λίστες, η *Κατανεμημένη* σύζευξη πλησιάζει την απόδοση της *Κεντροποιημένης* ενώ η *Υβριδική* παραμένει κάτω από το 50% επειδή οι λίστες των *SubIDs* σε κάθε βήμα προβλέπεται ότι θα είναι μικρότερες σε μέγεθος (θυμηθείται ότι κατά την *Υβριδική*

Σχήμα 4.23: Χρόνος απόκρισης κατά την επεξεργασία δημοσιεύσεων με *Κεντριοποιημένη*, *Κατανεμημένη*, και *Υβριδική* σύζευξη.

Σύζευξη αποστέλλονται πίσω στον συντονιστή εκείνα τα *SubIDs* που δεν πρόκειται να εμφανιστούν σε επόμενα βήματα και συζευγνύουν με την δημοσίευση). Εκτελέσαμε επίσης πειράματα με μεγαλύτερο αριθμό δημοσιεύσεων και συνδρομών ωστόσο, παρατηρήσαμε ανάλογα αποτελέσματα.

5 *PastryStrings*: Σύστημα Δημοσιεύσεων/Συνδρομών σε DHT Δίκτυα με Δρομολόγηση κατά Πρόθεμα

Ένα από τα βασικά πλεονεκτήματα της υποδομής *Pyracanthus* είναι η ανεξαρτησία της από το υποκείμενο DHT δίκτυο. Μπορούμε όμως να θυσιάσουμε αυτή την ευελιξία και να εκμεταλλευτούμε τα ιδιαίτερα χαρακτηριστικά ενός συγκεκριμένου DHT¹ ώστε να βελτιώσουμε ακόμα περισσότερο την απόδοση κατά την επεξεργασία των συνδρομών και των δημοσιεύσεων.

Στη προσέγγιση που παρουσιάζουμε εδώ, που πλέον θα αποκαλούμε *PastryStrings*, εκμεταλλευόμαστε τα ιδιαίτερα χαρακτηριστικά που προσφέρει το DHT δίκτυο *Pastry*. Συγκεκριμένα, εκμεταλλευόμαστε την αρχιτεκτονική του *Pastry* και του αλγόριθμου δρομολόγησης που χρησιμοποιεί, ώστε να προσφέρουμε λογαριθμική απόδοση σχετικά με τον αριθμό των μηνυμάτων που απαιτούνται για την επεξεργασία ιδιο-χαρακτηριστικών αριθμητικού τύπου και συμβολοσειρών. Συγκεκριμένα, στην υποδομή *PastryStrings* η επεξεργασία των συνδρομών απαιτεί $O(\log_\beta(N))$ μηνύματα για τις συμβολοσειρές, ενώ αυτή των ιδιο-χαρακτηριστικών αριθμητικού τύπου $O(\beta \times \log_\beta(|R|) \times \log(N))$ μηνύματα για διαστήματα τιμών R . Η επεξεργασία των δημοσιεύσεων από την άλλη απαιτεί $O(\log_\beta(N))$ μηνύματα. Επιπλέον, αυτό που κάνει την υποδομή *PastryStrings* ξεχωριστή είναι ότι εκμεταλλευόμαστε τον κανόνα εγγύτητας που χρησιμοποιεί το *Pastry* κατά τον σχηματισμό της τοπολογίας του, ώστε να εξασφαλίσουμε ότι συγκεκριμένα μονοπάτια που πρέπει να ακολουθήσουμε πάνω στο *Pastry* αποτελούνται από κόμβους που βρίσκονται σε μικρές αποστάσεις σε επίπεδο φυσικού δικτύου.

Δεδομένου ότι πιθανές εφαρμογές που θα αξιοποιούν την υποδομή *PastryStrings* θα λειτουργούν σε ένα ιδιαίτερα δυναμικό περιβάλλον (με συχνές αναχωρήσεις/βλάβες και αφίξεις κόμβων στο δίκτυο ομοτίμων εταίρων), εξασφαλίζουμε με την ανάπτυξη σχετικών αλγορίθμων, ότι αυτή η υποδομή θα προσαρμόζεται στις δυναμικές του δικτύου, θα λειτουργεί πλήρως κατανεμημένα με δυνατότητα αυτόνομης οργάνωσης και ανοχής σε βλάβες ενώ παράλληλα θα επιτρέπει υψηλού βαθμού κλιμάκωση.

5.1 Η Αρχιτεκτονική της Υποδομής *PastryStrings*

Οι δύο βασικές σχεδιαστικές επιλογές που χαρακτηρίζουν την υποδομή *PastryStrings* είναι:

¹Οποιοδήποτε DHT εκτελεί δρομολόγηση με βάση το πρόθεμα των αναγνωριστικών, όπως τα δίκτυα *Pastry* [85], *Tapestry* [103] και *Bamboo* [84], ανήκει σε αυτή την κατηγορία. Επιλέγουμε για λόγους παρουσίασης το δίκτυο *Pastry*.

1. ένα κατάλληλα ρυθμισμένο δίκτυο Pastry (ή οποιοδήποτε άλλο δίκτυο που στηρίζεται στη δομή του Plaxton) με βάση αλφαβήτου β για τα αναγνωριστικά των κόμβων του, τέτοια ώστε να μπορούμε να αντιστοιχίσουμε οποιοδήποτε λέξη ή (συμβολοσειρά γενικά) με τα αναγνωριστικά των κόμβων. Με άλλα λόγια, χρησιμοποιούμε το ίδιο αλφάβητο με β χαρακτήρες τόσο για τα αναγνωριστικά των κόμβων όσο και για τις συμβολοσειρές².
2. μία δενδρική δομή (που από εδώ και στο εξής θα αποκαλούμε *string trees*) η οποία αναπτύσσεται πάνω στο δίκτυο Pastry και χρησιμοποιείται για την επεξεργασία των συνδρομών και των δημοσιεύσεων με την τεχνική σύζευξης με βάση το πρόθεμα των συμβολοσειρών με παρόμοιο τρόπο που επιτυγχάνεται και η δρομολόγηση στο Pastry. Αυτή η δενδρική δομή έχει πολλές ομοιότητες με τα δένδρα που χρησιμοποιούνται στην υποδομή *Pyracanthus* (Κεφάλαιο 4). Ωστόσο, στην υποδομή *PastryStrings* χρησιμοποιούμε ένα σύνολο από β δένδρα (δομή *string trees*) για την επεξεργασία όλων των ιδιοχαρακτηριστικών και όλων των τύπων δεδομένων (συμβολοσειρές και αριθμητικές τιμές). Επιπλέον, τα δένδρα αυτά όπως θα δείξουμε παρακάτω είναι στενά συνδεδεμένα με την τοπολογία του δικτύου Pastry και του τρόπου που επιτυγχάνεται η δρομολόγηση αιτήσεων σε αυτό.

Κάθε δέντρο στη δομή *string trees* είναι αφιερωμένο σε ένα από τους β χαρακτήρες του αλφαβήτου μας. Κατά την επεξεργασία συνδρομών ή δημοσιεύσεων με ιδιοχαρακτηριστικό τύπου συμβολοσειράς, αρχικά εντοπίζουμε το κατάλληλο δέντρο (το δέντρο που διαχειρίζεται όλες τις συμβολοσειρές που ο πρώτος τους χαρακτήρας είναι ίδιος με αυτόν της συμβολοσειράς που εξετάζουμε) και στη συνέχεια ακολουθούμε ένα μονοπάτι προς ένα κόμβο «ραντεβού» μέσα στο δέντρο. Οι αλγόριθμοι που θα παρουσιάσουμε στη συνέχεια εγγυώνται ότι στον κόμβο «ραντεβού» θα συναντηθούν οι δημοσιεύσεις που συζευγνύν με τις εκεί αποθηκευμένες συνδρομές.

Ένα ιδιαίτερο στοιχείο που χαρακτηρίζει την υποδομή *PastryStrings* είναι ότι τα δένδρα που αναπτύσσονται πάνω στο Pastry, κατασκευάζονται με τέτοιο τρόπο ώστε να εκμεταλλεύονται την πληροφορία που κρατείται στους τοπικούς πίνακες δρομολόγησης των κόμβων, ώστε να επιτύχουμε μίας δενδρική δομή της οποίας οι άμεσα συνδεδεμένοι κόμβοι να βρίσκονται όσο το δυνατόν πιο κοντά (π.χ. με βάση μία μετρική όπως ο χρόνος διάδοσης μετέπιστροφής -round-trip time).

Η αρχιτεκτονική της υποδομής *PastryStrings*, στηρίζεται στο μοντέλο που έχουμε περιγράψει σε προηγούμενη ενότητα, με κόμβους του δικτύου που παίζουν τον ρόλο του διαμεσολαβητή (broker) και τους χρήστες που μπορεί να είναι είτε συνδρομητές είτε να γεννήτριες δημοσιεύσεων, συνδεδεμένους με τους διαμεσολαβητές.

²Μία τυπική τιμή για το β θα μπορούσε να είναι το 64 ώστε το αλφάβητο μας να περιέχει όλους τους χαρακτήρες του Αγγλικού αλφαβήτου (κεφαλαία και πεζά) καθώς και τα 10 ψηφία και ορισμένα σημεία στίξης.

Για λόγους παρουσίασης ας θεωρήσουμε προς το παρόν ότι τόσο οι συνδρομές όσο και οι δημοσιεύσεις αποτελούνται από ένα μόνο ιδιοχαρακτηριστικό. Η επεξεργασία δημοσιεύσεων/συνδρομών με περισσότερα ιδιοχαρακτηριστικά έχει ήδη περιγραφεί στο προηγούμενο κεφάλαιο και συγκεκριμένα στην ενότητα 4.3.2.

5.1.1 Βασικά Χαρακτηριστικά των String Trees

Υπάρχουν δύο τύποι κόμβων στην υποδομή *PastryStrings*. Οι κόμβοι του δικτύου και οι κόμβοι των δέντρων (που αποκαλούμε *Tnodes*). Ένας κόμβος είναι δυνατό να φιλοξενεί περισσότερους του ενός *Tnodes*. Οι κόμβοι έχουν αναγνωριστικά που ορίζονται με σχετικούς μηχανισμούς του *Pastry* ενώ οι *Tnodes* έχουν ετικέτες. Η ετικέτα *lbl* του κόμβου *Tnode_{lbl}* υποδηλώνει την συμβολοσειρά για την οποία είναι υπεύθυνος (συμμετέχοντας την επεξεργασία συνδρομών ή δημοσιεύσεων με τιμές ίδιες με την ετικέτα του).

Κάθε δέντρο που ανήκει στα *string trees*, συμβολίζεται με T_i όπου i ανήκει στο σύνολο S των χαρακτήρων που σχηματίζουν το αλφάβητο μας, και είναι υπεύθυνο για την επεξεργασία όλων των συμβολοσειρών που ο πρώτος τους χαρακτήρας είναι ίδιος με τον χαρακτήρα i . Αυτό συνεπάγεται ότι όλες οι ετικέτες των *Tnodes* που ανήκουν στο δέντρο T_i έχουν τον ίδιο πρώτο χαρακτήρα i . Κάθε δέντρο έχει μέγιστο βάθος d_{max} ίσο με το μέγιστο μέγεθος των συμβολοσειρών (το βάθος της ρίζας κάθε δέντρου είναι ίσο με το 0).

Μια πρώτη απόπειρα σχετικά με την κατασκευή των *string trees* αφορούσε την εκμετάλλευση των πινάκων δρομολόγησης κάθε κόμβου στο δίκτυο *Pastry*. Δεδομένου ότι τα αναγνωριστικά των κόμβων του *Pastry* σχηματίζονται με το ίδιο αλφάβητο με τις συμβολοσειρές που διαχειριζόμαστε, θα μπορούσαμε να συμβουλευτούμε τους πίνακες δρομολόγησης και να εκτελέσουμε δρομολόγηση με βάση το πρόθεμα χρησιμοποιώντας την συμβολοσειρά, ώστε τελικά να καταλήξουμε στον κόμβο όπου θα φιλοξενήσει τον *Tnode* «ραντεβού» με ετικέτα ίδια με την συμβολοσειρά. Κατά την διάρκεια αυτής της διαδικασίας, όλοι οι ενδιάμεσοι κόμβοι θα φιλοξενούσαν τους *Tnodes* του μονοπατιού του δέντρου από την ρίζα μέχρι τον τελικό *Tnode*.

Αυτή η προσέγγιση έχει ορισμένα μειονεκτήματα, όπως η αυξημένη πολυπλοκότητα που εισάγεται κατά την διατήρηση της συνεκτικότητας του δέντρου υπό την παρουσία συχνών αναχωρήσεων/αφίξεων κόμβων στο δίκτυο. Θεωρήστε για παράδειγμα την περίπτωση όπου ο κόμβος A αλλάζει την εγγραφή στον πίνακα δρομολόγησης του που έδειχνε στον B , ώστε πλέον να δείχνει στον C , επειδή ο B αποχώρησε ξαφνικά από το δίκτυο. Σε αυτή την περίπτωση ολόκληρο το τμήμα του δέντρου που έχει ρίζα τον κόμβο B θα αποκοπεί από το υπόλοιπο δέντρο και θα είναι αδύνατο να προσπελαστεί από μελλοντικές αιτήσεις. Έτσι, είτε θα πρέπει ολόκληρο αυτό το υποδέντρο να μεταφερθεί ώστε να «κρέμεται» πλέον κάτω από τον C που συνεπάγεται ότι κάθε *Tnode* που ανήκει σε αυτό το υποδέντρο θα πρέπει να μεταφερθεί σε άλλους κόμβους, είτε θα πρέπει να ανανεώσουμε τον πίνακα δρομολόγησης του C , πράγμα που έχει ως αποτέλεσμα την παρέμβαση στη

διαδικασία σχηματισμού του δικτύου Pastry.

Συνεπώς, μια καλύτερη ιδέα είναι η ανάπτυξη της δενδρικής δομής πάνω στο δίκτυο με την διατήρηση επιπλέον κατάστασης δρομολόγησης σε κάθε $Tnode$. Ο πίνακας δρομολόγησης σε κάθε $Tnode$ έχει σταθερό μέγεθος, ίσο με β εγγραφές με δείκτες σε κάθε ένα από τα παιδιά του. Η συμπλήρωση του πίνακα δρομολόγησης ενός $Tnode$ γίνεται με το να συμβουλευόμαστε τον πίνακα δρομολόγησης του κόμβου που τον φιλοξενεί. Ο πίνακας δρομολόγησης ενός Pastry κόμβου αποτελείται από $\beta \times \log_\beta(N)$ εγγραφές. Κάθε $Tnode$ χρησιμοποιεί σαν υπόδειξη ένα από τα $\log_\beta(N)$ επίπεδα του πίνακα δρομολόγησης του κόμβου που τον φιλοξενεί, με σκοπό να κτίσει τον δικό του πίνακα δρομολόγησης.

Συγκεκριμένα, εάν ένας $Tnode$ βρίσκεται σε βάθος d θα πρέπει να χρησιμοποιήσει το $(d+1) \bmod \log_\beta(N)$ επίπεδο του πίνακα δρομολόγησης του οικοδεσπότη. Δεδομένου ότι το αλφάβητο για τα αναγνωριστικά των κόμβων και των ετικετών των $Tnode$ είναι το ίδιο, ο στόχος μας είναι να εκτελέσουμε δρομολόγηση με βάση το πρόθεμα με τη χρήση των ετικετών των $Tnodes$ όπως ακριβώς επιτυγχάνεται η δρομολόγηση στο Pastry με βάση το πρόθεμα και την χρήση των αναγνωριστικών των κόμβων. Με αυτή την επιλογή (i) εκμεταλλευόμαστε τους μηχανισμούς αυτόνομης οργάνωσης που χρησιμοποιεί το Pastry και (ii) επιτυγχάνουμε μικρό χρόνος RTT - Round Trip Time όπου αυτό είναι δυνατό.

Τα *string trees* δημιουργούνται δυναμικά καθώς νέα αιτήματα για την αποθήκευση συνδρομών με ιδιοχαρακτηριστικά τύπου συμβολοσειράς, καταφθάνουν στο δίκτυο. Δεδομένου ότι υπάρχουν πολλά *string trees* εντοπίζουμε τη ρίζα ενός συγκεκριμένου T_i με τη χρήση της συνάρτησης κατακερματισμού του Pastry και περνώντας σαν όρισμα τον πρώτο χαρακτήρα της δεδομένης συμβολοσειράς που εξετάζουμε. Κατόπιν, ο κόμβος με το συγκεκριμένο αναγνωριστικό θα φιλοξενήσει (ή ήδη φιλοξενεί) τη ρίζα του δέντρου T_i .

Όταν μία δημοσίευση καταφθάνει στο σύστημα μας, με μια συγκεκριμένη τιμή, αρχικά εντοπίζουμε τη ρίζα του κατάλληλου T_i και προωθούμε τη δημοσίευση προς το $Tnode$ που έχει την ίδια ετικέτα με την συμβολοσειρά που ορίζεται στη δημοσίευση. Η δρομολόγηση πραγματοποιείται με τον έλεγχο ενός χαρακτήρα της συμβολοσειράς κάθε φορά και επιλογή της κατάλληλης εγγραφής στον πίνακα δρομολόγησης του εκάστοτε $Tnode$. Όλες οι συνδρομές που βρίσκονται αποθηκευμένες εκεί, θεωρούνται ότι συζευγνύουν με τη δημοσίευση μιας και έχουν δηλώσει την ίδια τιμή με την τιμή της δημοσίευσης.

Το ακόλουθο παράδειγμα επεξηγεί πώς δημιουργούνται τα *string trees* και πώς εντοπίζονται οι συνδρομές που συζευγνύουν με μια δημοσίευση, στο απλό μοντέλο όπου συνδρομές και δημοσιεύσεις αποτελούνται από ένα μόνο ιδιοχαρακτηριστικό.

Παράδειγμα 5 Επεξεργασία συνδρομής και αποθήκευση του αναγνωριστικού της σε ένα *string tree* και σύζευξη της με μία δημοσίευση.

Στο Σχήμα 5.1 παρουσιάζεται ένα απλουστευμένο στιγμιότυπο της υποδομής PastryStrings. Σε αυτό

Σχήμα 5.1: Ανάπτυξη ενός *string tree* με $\beta = 2$ και αναγνωριστικά μεγέθους 3 χαρακτήρων και αποθήκευση σε αυτό *SubIDs* συνδρομών με τιμές '010' και '00'

Σχήμα 5.2: Σύζευξη δημοσίευσης με συνδρομές. Η δημοσίευση έχει τιμή '010' ενώ ήδη είναι αποθηκευμένες δύο συνδρομές με τιμές '00' και '010'.

το παράδειγμα, χρησιμοποιούμε συμβολισμούς μεγέθους 3 χαρακτήρων με αλφάβητο αποτελούμενο από $\beta = 2$ χαρακτήρες. Υποθέστε τώρα ότι ένας χρήστης εκφράζει τα ενδιαφέροντά του με δύο συνδρομές (με τα αναγνωριστικά $SubID_1$ και $SubID_2$) χρησιμοποιώντας το ίδιο ιδιοχαρακτηριστικό και ορίζοντας τιμές '010' και '00' αντίστοιχα.

Σε αυτό το παράδειγμα, υπάρχουν γενικά δύο δέντρα: το T_0 και το T_1 . Και οι δύο συνδρομές σε αυτό το παράδειγμα αφορούν το T_0 . Για να επεξεργαστούμε το αίτημα αποθήκευσης του αναγνωριστικού της συνδρομής που σχετίζεται με την τιμή '010' πρέπει πρώτα να εντοπίσουμε τον κόμβο του Pastry που είναι αρμόδιος για τη φιλοξενία της ρίζας του δέντρου T_0 (ας υποθέσουμε ότι αυτός είναι ο κόμβος με αναγνωριστικό '000'). Κατασκευάζουμε εκεί τη ρίζα, $Tnode_0$ και δεδομένου ότι ο πίνακας δρομολόγησης της είναι κενός χρησιμοποιούμε τον πίνακα δρομολόγησης του κόμβου που την φιλοξενεί και συγκεκριμένα το πρώτο του επίπεδο σαν υπόδειξη (η ρίζα βρίσκεται σε βάθος $d = 0$ και έτσι εξετάζουμε το επίπεδο $(d + 1) \bmod 3 = 1$). Ο πίνακας δρομολόγησης του κόμβου που φιλοξενεί τον $Tnode_0$ στο πρώτο του επίπεδο περιέχει δύο καταχωρήσεις (δεδομένου ότι έχουμε δυαδικό αλφάβητο σε αυτό το παράδειγμα) που αντιγράφονται στον πίνακα δρομολόγησης του $Tnode_0$. Η πρώτη καταχώρηση δείχνει τον κόμβο '001' που θα φιλοξενήσει τον $Tnode_{00}$, ενώ η δεύτερη δείχνει στον '011' που πρόκειται να φιλοξενήσει τον $Tnode_{01}$.

Εφόσον έχουμε συμπληρώσει τον πίνακα δρομολόγησης της ρίζας $Tnode_0$ συνεχίζουμε την διαδικασία αποθήκευσης με τον έλεγχο του δεύτερου χαρακτήρα της συμβολιστικής: '1'. Εξετάζουμε τον πίνακα δρομολόγησης της ρίζας, και επιλέγουμε την εγγραφή που ταιριάζει με το επόμενο ψηφίο της συμβολιστικής, το ψηφίο '1'. Η εγγραφή δείχνει στον κόμβο '011', οπότε στέλνουμε εκεί το αίτημα αποθήκευσης. Επειδή

αρχικά ο κόμβος '011' δεν φιλοξενεί κανένα $Tnode$, κατασκευάζουμε εκεί τον $Tnode_{01}$ και συμπληρώνουμε τον πίνακα δρομολόγησης του χρησιμοποιώντας το δεύτερο επίπεδο του πίνακα δρομολόγησης του κόμβου '011'. Κατόπιν, συνεχίζουμε την διαδικασία με την εξέταση του τελικού χαρακτήρα της συμβολοσειράς, '0'. Συμβουλευόμαστε τον πίνακα δρομολόγησης του $Tnode_{01}$ και συγκεκριμένα την εγγραφή στην στήλη '0' και διαβιβάζουμε το αίτημα στον κόμβο '100'. Κατασκευάζουμε εκεί τον $Tnode_{010}$ και αποθηκεύουμε το αναγνωριστικό $SubID_1$. Το δεύτερο αίτημα αποθήκευσης αντιμετωπίζεται ακολουθώντας τον ίδιο αλγόριθμο, όπως μπορείτε να δείτε στο Σχήμα 5.1, με αποτέλεσμα να αποθηκευτεί το $SubID_2$ στον $Tnode_{00}$ που φιλοξενείται στον κόμβο '001'.

Στο Σχήμα 5.2 παρουσιάζεται ο ένα στιγμιότυπο της υποδομής $PastryStrings$ με τις δύο αποθηκευμένες συνδρομές. Μία δημοσίευση καταφθάνει στο σύστημα με τιμή '010'. Αρχικά, εντοπίζουμε τη ρίζα, $Tnode_0$, του *string tree* που είναι αρμόδιο για το χαρακτήρα '0' και στέλνουμε εκεί τη δημοσίευση. Στον $Tnode_0$ θα συμβουλευτούμε τον πίνακα δρομολόγησης του στη στήλη '1' για το δείκτη προς τον $Tnode_{01}$ όπου και θα προωθήσουμε εκεί τη δημοσίευση. Στον $Tnode_{01}$ ακολουθούμε τον δείκτη στον πίνακα δρομολόγησης του στη στήλη '0', και τελικά καταλήγουμε στον $Tnode_{010}$ που είναι αποθηκευμένο το $SubID_1$. $\diamond$

Έχοντας εισάγει την έννοια των *Range Tree Forest*(T_i), και το πώς η δρομολόγηση εκτελείται σε αυτά, παρατηρούμε ότι συνυπάρχουν δύο διαφορετικά ήδη δρομολόγησης στην υποδομή $PastryStrings$. Συγκεκριμένα:

- **Δρομολόγηση στο Pastry:** πραγματοποιείται με βάση τους πίνακες δρομολόγησης του Pastry και προσφέρει τη συνηθισμένη λειτουργικότητα και διεπαφές που περιγράφονται στη σχετική εργασία [85]. Η δρομολόγηση στο Pastry είναι απαραίτητη για τον εντοπισμό των T_i δέντρων και για τη ανάπτυξη αυτών πάνω στο δίκτυο.
- **Δρομολόγηση στα *string trees*:** εκτελείται μέσα σε κάθε T_i δέντρο και εκμεταλλεύεται τους πίνακες δρομολόγησης των $Tnodes$ προκειμένου να διαβιβαστούν τα αιτήματα (αποθήκευσης ή ανάκτησης $SubIDs$) προς τα φύλλα. Μια χαρακτηριστική λειτουργία είναι η $Tc_forward(msg, key)$, που εκτελείται τοπικά σε κάθε $Tnode$, προωθώντας ένα μήνυμα (msg) στον $Tnode$ που είναι αρμόδιο για την τιμή key .

Πρέπει να τονίσουμε ότι η δρομολόγηση Pastry δεν επηρεάζεται από την $PastryStrings$. Κατά τη δρομολόγηση στα *string trees* χρησιμοποιούνται οι πίνακες δρομολόγησης των $Tnodes$. Ο πίνακας δρομολόγησης των $Tnodes$ αποτελεί στην ουσία ένα επιπλέον επίπεδο δρομολόγησης, που έχει μια εγγραφή προς ένα κοντινό $Tnode$ για κάθε πιθανό χαρακτήρα του αλφαβήτου μας.

5.1.2 Επεξεργασία Συνδρομών και Δημοσιεύσεων με Ιδιοχαρακτηριστικά

Αριθμητικού Τύπου και Συμβολοσειρές

Σε αυτή την ενότητα παρουσιάζουμε την προσέγγιση που ακολουθούμε στην υποδομή *PastryStrings* για την υποστήριξη περιορισμών προθέματος (π.χ. '010*') και επιθέματος (π.χ. τα '*101') σε ιδιοχαρακτηριστικά τύπου συμβολοσειράς καθώς επίσης και διαστημάτων τιμών (που προκύπτουν από τελεστές όπως $\leq$, $\geq$, $\neq$) σε ιδιοχαρακτηριστικά αριθμητικού τύπου.

Οι λύσεις που προτείνουμε εδώ έχουν αρκετές ομοιότητες με τις λύσεις που παρουσιάζονται στην υποδομή *Pyracanthus* και ειδικότερα σε αυτές που σχετίζονται με την ανάπτυξη δενδρικών δομών στο *Range Tree Forest*. Ωστόσο, εδώ υπάρχει ένα σύνολο από β δέντρα (*string trees*) που διαχειρίζονται μόνο συμβολοσειρές και σχετίζονται με την τοπολογία του *Pastry*. Έτσι, ενώ η επεξεργασία των συμβολοσειρών στην *PastryStrings* έχει αρκετές ομοιότητες με την αντίστοιχη stateful προσέγγιση στην *Pyracanthus*, η επεξεργασία των αριθμητικών τιμών παρουσιάζει διαφορές, μιας και πρέπει να μεταφραστούν οι αριθμητικές τιμές σε ετικέτες (συμβολοσειρές) των *Tnodes* στα δέντρα που ανήκουν στο σύνολο *string trees*.

Επεξεργασία Συνδρομών και Δημοσιεύσεων με Ιδιοχαρακτηριστικά Αριθμητικού Τύπου

Στην υποδομή *PastryStrings* υποστηρίζονται ιδιοχαρακτηριστικά με αριθμητικές τιμές και περιορισμούς διαστημάτων τιμών, $\leq$, $\geq$, $\neq$, και $=$. Η βασική ιδέα εδώ είναι ότι κάθε διάστημα τιμών μπορεί να αντιστοιχηθεί κατάλληλα σε ένα σύνολο από *Tnodes* με βάση την ετικέτα τους και τη θέση τους στο δέντρο T_i . Υπάρχουν διάφοροι τρόποι να επιτύχουμε αυτή την αντιστοιχία. Εδώ υιοθετούμε και προσαρμόζουμε στις ανάγκες μας την προσέγγιση *segment tree* [42] και ειδικότερα μία γενίκευση της δενδρικής δομής που έχει παρουσιαστεί στην εργασία των Gao και άλλοι [50], όπου χρησιμοποιήσαμε και στην υποδομή *Pyracanthus*.

Η απαραίτητη λειτουργικότητα αποτελείται από:

1. την αντιστοίχιση διαστημάτων τιμών σε *Tnodes*.
2. διάσπαση ενός διαστήματος τιμών στο κατάλληλο σύνολο υποδιαστημάτων που αντιστοιχίζονται σε *Tnodes*.

Λαμβάνοντας υπόψη αυτήν την λειτουργικότητα, όταν φθάνει μια συνδρομή με περιορισμό διαστήματος τιμών, τεμαχίζουμε αρχικά το διάστημα στα κατάλληλα υποδιαστήματα. Κατόπιν, εντοπίζουμε τον κατάλληλο για κάθε υποδιάστημα *Tnode* του δέντρου T_i και αποθηκεύουμε εκεί το *SubID* της συνδρομής. Όταν μια δημοσίευση φθάσει στο σύστημα μας με συγκεκριμένη αριθμητική τιμή, μετασχηματίζουμε κατάλληλα την τιμή σε μία συμβολοσειρά χρησιμοποιώντας το ίδιο αλφάβητο β χαρακτήρων με τα *string trees*. Στη

συνέχεια εντοπίζουμε το κατάλληλο δέντρο που είναι υπεύθυνο για αυτή τη συμβολοσειρά και ακολουθούμε το μονοπάτι από την ρίζα του δέντρου μέχρι τον $Tnode$ με ετικέτα ίδια με την συμβολοσειρά. Κάθε $Tnode$ σε αυτό το μονοπάτι, είναι δυνατό να έχει αποθηκευμένα $SubIDs$ συνδρομών που συζευγνύουν με την δημοσίευση και συνεπώς συλλέγονται.

Κάθε δέντρο T_i έχει ύψος H (ίσο με μέγιστο βάθος d_{max}) που ορίζεται από το μέγιστο μέγεθος των συμβολοσειρών που μπορεί να διαχειριστεί η $PastryStrings$. Κάθε $Tnode_{lbl}$ βρίσκεται σε βάθος d , με τη ρίζα του δέντρου να βρίσκεται σε βάθος $d=0$. Το μέγιστο πλήθος των παιδιών (δηλ. ο βαθμός g του δέντρου) που μπορεί να έχει κάθε κόμβος $Tnode$ είναι ίσος με τον αριθμό των χαρακτήρων του αλφαβήτου μας (β).

Παρακάτω θα χρησιμοποιήσουμε τη συνάρτηση $num(lbl)$ που επιστρέφει την αριθμητική μετάφραση της συμβολοσειράς lbl στο δεκαδικό σύστημα. Για παράδειγμα, λαμβάνοντας υπόψη ότι το αλφάβητο μας αποτελείται από $\beta = 2$ χαρακτήρες τότε αν η συμβολοσειρά είναι $lbl = '0010'$ η αριθμητική τιμή της είναι $num('0010') = 2$.

Δεδομένου λοιπόν των χαρακτηριστικών των δέντρων ($g=\beta$ και H), μπορούν να αντιστοιχηθούν διαστήματα τιμών μεγέθους μέχρι β^{H+1} σε $Tnodes$ στην υποδομή $PastryStrings$. Ακολουθώντας λοιπόν το μοντέλο δημοσιεύσεων/συνδρομών που παρουσιάστηκε στο Κεφάλαιο 3, τα ιδιοχαρακτηριστικά αριθμητικού τύπου a_i έχουν πεδίο ορισμού D_{ai} . Επίσης, επειδή το β είναι σταθερό, θα πρέπει να επιλέξουμε κατάλληλα το ύψος των δέντρων H έτσι ώστε το πεδίο ορισμού του ιδιοχαρακτηριστικού με το μεγαλύτερο πλήθος τιμών να ικανοποιεί την συνθήκη:

$$D_{ai} \leq \beta^{H+1}$$

Κάθε $Tnode_{lbl}$ (σε βάθος d σε ένα δέντρο T_i) που είναι υπεύθυνο για τη συμβολοσειρά lbl θα πρέπει να είναι επίσης υπεύθυνο για την διαχείριση του διαστήματος τιμών που δίνεται από τον παρακάτω τύπο:

$$[num(lbl) \times \beta^{H-d}, (num(lbl) + 1) \times \beta^{H-d})$$

Στο Σχήμα 5.3 παρουσιάζουμε μια απλουστευμένη έκδοση της $PastryStrings$ υποδομής για την περιγραφή της επεξεργασίας των δημοσιεύσεων και των συνδρομών. Σε αυτό το σχήμα, το $\beta = 2$ και το μέγιστο μήκος των συμβολοσειρών είναι ίσο με 3 ($H = 3$). Όπως μπορείτε να δείτε, ο $Tnode_{001}$, είναι υπεύθυνος για τιμές που ανήκουν στο διάστημα $[2, 4)$. Χρησιμοποιώντας τον τύπο αντιστοίχισης $Tnodes$ σε διαστήματα τιμών και δεδομένου ότι $num('001') = 1$, $\beta = 2$, $H = 4$ και $d = 2$ προκύπτει το διάστημα: $[1 \times 2^{3-2}, (1 + 1) \times 2^{3-2}) = [2, 4)$. Κάθε φύλλο $Tnode$ είναι υπεύθυνο για το μικρότερο δυνατό υποδιάστημα τιμών, ενώ οποιοσδήποτε άλλος $Tnode$ είναι υπεύθυνο για την ένωση των υποδιαστημάτων των παιδιών

Σχήμα 5.3: Το δέντρο T_0 με αποθηκευμένα τα $SubIDs$ για το διάστημα [2,5). Μία δημοσίευση καταφθάνει με τιμή 3 και ακολουθώντας το μονοπάτι μέχρι το φύλλο, εντοπίζει τα αποθηκευμένα $SubIDs$.

του. Η ένωση των υποδιαστημάτων των ριζών των β δέντρων που υπάρχουν στην *PastryStrings*, καλύπτει το μεγαλύτερο πεδίο ορισμού των ιδιοχαρακτηριστικών αριθμητικού τύπου ($D_{ai} \leq \beta^{H+1}$).

Μέχρι τώρα έχουμε περιγράψει, το πως μπορεί εύκολα να ανακαλυφθεί το υποδιάστημα για το οποίο είναι αρμόδιος ένας $Tnode$, με βάση την ετικέτα του. Για την αποθήκευση στα δέντρα μας, του αναγνωριστικού μιας συνδρομής με περιορισμό διαστήματος τιμών, θα πρέπει να τεμαχίσουμε το διάστημα σε υποδιαστήματα (που αντιστοιχούν σε $Tnodes$), και να εντοπίσουμε τους κατάλληλους $Tnodes$ που είναι αρμόδιοι για την διαχείριση αυτών των υποδιαστημάτων. Σε αυτούς τους $Tnodes$ θα πρέπει τελικά να αποθηκεύσουμε το $SubID$ της συνδρομής. Για τον τεμαχισμό των διαστημάτων σε υποδιαστήματα χρησιμοποιούμε τον αλγόριθμο που παρουσιάστηκε στον Πίνακα 4.1 στα πλαίσια της παρουσίασης της υποδομής *Pyracanthus*. Με αυτό τον αλγόριθμο μπορούμε να τεμαχίσουμε ένα διάστημα τιμών R σε $2 \times g \times \lfloor \log |R| \rfloor - g$ υποδιαστήματα τα οποία δεν επικαλύπτονται και η ένωσή τους μας δίνει το διάστημα R . Έτσι, για παράδειγμα, το διάστημα [2,5), με την εφαρμογή αυτού του αλγορίθμου τεμαχίζεται στα υποδιαστήματα [2,4) και [4,5) (στο παράδειγμα του Σχήματος 5.3). Σε αυτή την περίπτωση το $SubID$ της συνδρομής που έχει ορίσει ένα τέτοιο διάστημα θα πρέπει να αποθηκευθεί στους κόμβους $Tnode_{001}$ και $Tnode_{0100}$, όπως φαίνεται στο Σχήμα 5.3.

Προηγουμένως, δείξαμε το μηχανισμό που μας επιτρέπει δεδομένης μιας ετικέτας ενός $Tnode$, να υπολογίσουμε το διάστημα τιμών για το οποίο είναι υπεύθυνος αυτός ο $Tnode$.

Εντούτοις, η αντίστροφη λειτουργία (δηλ., λαμβάνοντας υπόψη το διάστημα τιμών να υπολογίσουμε την ετικέτα lbl του αντίστοιχου κόμβου), επιτυγχάνεται με την ακόλουθη μέθοδο. Με βάση τον τρόπο που τα υποδιαστήματα αντιστοιχίζονται $Tnodes$, μπορούμε να παρατηρήσουμε ότι για δεδομένο υποδιάστημα r , το

$Tnode$ που είναι υπεύθυνο για αυτό βρίσκεται σε βάθος $d = H - \log_\beta |r| - 1$. Δεδομένου ότι ξέρουμε το βάθος του $Tnode_{lbl}(d)$, η ετικέτα του είναι το πρόθεμα $d + 1$ χαρακτήρων της αλφαριθμητικής αναπαράστασης με αλφάβητο β χαρακτήρων, του κάτω ορίου του υποδιαστήματος

$$lbl = prefix(\beta - ary(l), d + 1)$$

όπου $\beta - ary(int\ x)$ είναι η αλφαριθμητική αναπαράσταση μήκους $H + 1$ χαρακτήρων της τιμής x με αλφάβητο β χαρακτήρων, l είναι το κάτω όριο του υποδιαστήματος $r = [l, h)$, και $prefix(string\ s, int\ i)$ είναι το πρόθεμα i χαρακτήρων της συμβολοσειράς s .

Παραδείγματος χάριν, όπως μπορείτε να δείτε στο Σχήμα 5.3, το υποδιάστημα $[2, 4)$ αντιστοιχίζεται σε ένα $Tnode$ σε βάθος $4 - 1 - 1 = 2$, και επειδή η δυαδική αναπαράσταση της χαμηλότερης τιμής (2) του υποδιαστήματος αποτελούμενη από $H + 1 = 4$ χαρακτήρες είναι '0010', προκύπτει ότι η ετικέτα του $Tnode$ που διαχειρίζεται αυτό το υποδιάστημα είναι το πρόθεμα 3 (2+1) χαρακτήρων της συμβολοσειράς '0010', δηλαδή η '001' ($= prefix(\beta - ary(2), 2 + 1) = prefix('0010', 3)$).

Όταν μία δημοσίευση καταφθάνει με συγκεκριμένη τιμή, υπολογίζουμε αρχικά το μικρότερο δυνατό υποδιάστημα που περιλαμβάνει την τιμή. Εάν, παραδείγματος χάριν, η τιμή είναι 3, το μικρότερο υποδιάστημα που περιλαμβάνει το 3, είναι το $[3, 4)$. Κατόπιν, υπολογίζουμε την ετικέτα του $Tnode$ που είναι υπεύθυνο για εκείνο το υποδιάστημα, '0011', και προωθούμε τη δημοσίευση προς τον $Tnode_{0011}$. Όλα τα αποθηκευμένα $SubIDs$ στους $Tnodes$ που βρίσκονται στο μονοπάτι από τη ρίζα μέχρι το φύλλο $Tnode_{0011}$ θεωρείται ότι συζευγνύουν με την δημοσίευση.

Επεξεργασία Συνδρομών και Δημοσιεύσεων με Ιδιοχαρακτηριστικά Τύπου Συμβολοσειράς

Αρχικά παρατηρείστε ότι ένας περιορισμός επιθέματος μπορεί να μετασχηματιστεί εύκολα σε περιορισμό προθέματος κατά την επεξεργασία των συνδρομών, εάν απλά να εξετάσουμε τη συμβολοσειρά από της τελευταίο προς τον πρώτο χαρακτήρα της. Κατά συνέπεια, θα παρουσιάσουμε μόνο πώς επεξεργαζόμαστε περιορισμούς προθέματος στην υποδομή *PastryStrings*.

Υποθέστε ότι έχουμε μια συνδρομή με ένα ιδιοχαρακτηριστικό με τελεστή προθέματος. Προκειμένου να αποθηκευτεί κατάλληλα η συνδρομή ακολουθούμε την ίδια μεθοδολογία όπως όταν πρόκειται για ιδιοχαρακτηριστικό με τελεστή ισότητας. Όταν μία δημοσίευση καταφθάσει στο σύστημα, εντοπίζουμε το κατάλληλο δέντρο T_i που είναι αρμόδιο για τον πρώτο χαρακτήρα της συμβολοσειράς της δημοσίευσης (π.χ. T_a για την συμβολοσειρά 'abc*'), και ακολουθούμε ένα συγκεκριμένο μονοπάτι στο δέντρο από την ρίζα μέχρι τον $Tnode$ του οποίου ετικέτα είναι ίση με την συμβολοσειρά.

Σχήμα 5.4: Αποθήκευση συνδρομών με ιδιοχαρακτηριστικά τύπου συμβολοσειράς, με την τεχνική της δρομολόγησης με βάση το πρόθεμα.

Ακολουθώντας αυτό το μονοπάτι, και δεδομένου ότι εκτελούμε δρομολόγηση πάνω στο δέντρο με βάση το πρόθεμα (στα πρότυπα της δρομολόγησης στο δίκτυο Pastry), όλοι οι $Tnodes$ που ανήκει σε αυτό το μονοπάτι ενδέχεται να έχουν αποθηκευμένα τα αναγνωριστικά των συνδρομών που συζευγνύουν με κάποιο πρόθεμα της συμβολοσειράς της δημοσίευσης.

Παράδειγμα 6 Αποθήκευση συνδρομών με ιδιοχαρακτηριστικό τύπου συμβολοσειράς.

Υποθέστε ότι μια συνδρομή φθάνει στο σύστημα με περιορισμό προθέματος με τιμή '00*'. Εντοπίζουμε αρχικά το κατάλληλο δέντρο για το χαρακτήρα '0', T_0 , (δείτε το Σχήμα 5.4, όπου ο κόμβος '0111' φιλοξενεί τη ρίζα $Tnode_0$ του δέντρου T_0). Κατόπιν το αναγνωριστικό της συνδρομής προωθείται και αποθηκεύεται στον $Tnode_{00}$ στην λίστα L_{a-pref} .

Υποθέστε ότι μία δημοσίευση καταφθάνει στο σύστημα με τιμή '001'. Αρχικά η ρίζα του T_0 , $Tnode_0$ εντοπίζεται. Κατόπιν προωθείται στον κόμβο που φιλοξενεί τον $Tnode_{001}$. Σε κάθε $Tnode$ στην διαδρομή από τη ρίζα στο $Tnode_{001}$, συλλέγουμε τις αποθηκευμένες συνδρομές εάν αυτές υπάρχουν. Όπως μπορείτε να δείτε, υπάρχει μια αποθηκευμένη συνδρομή στον $Tnode_{00}$ και έτσι συλλέγεται από τον συντονιστή μιας και συζευγνύει με την τιμή της δημοσίευσης. ◇

Εκτός από τους τελεστές προθέματος και επιθέματος, η λύση που προτείνουμε μπορεί επίσης να υποστηρίξει τον τελεστή '10*01'. Αυτός ο τελεστής μπορεί να αποσυντεθεί εύκολα στους τελεστές προθέματος

και επιθέματος Έτσι, παραδείγματος χάριν, ο περιορισμός 10×11 μπορεί να αντιμετωπισθεί ως πρόθεμα (δηλ. 10^*) και ως επίθεμα (δηλ. *11) και με την κατάλληλη επεξεργασία μπορούμε να καταλήξουμε σχετικά με το αν υπάρχει σύζευξη ή όχι.

5.2 Εξισορρόπηση του Φόρτου στην Υποδομή *PastryStrings*

Ένας πιθανός περιορισμός της προσέγγισης που μόλις περιγράφηκε είναι ότι ένα πολύ μικρό σύνολο των κόμβων μπορεί να αποτελέσει σημείο συμφόρησης, καθώς αναμένεται ότι θα εξυπηρετεί ένα μεγάλο ποσοστό των αιτήσεων για την επεξεργασία των συνδρομών και των δημοσιεύσεων. Οι κόμβοι που ανήκουν σε αυτήν την κατηγορία είναι οι *Tnodes* που βρίσκονται κοντά στη ρίζα του κάθε T_i δέντρου. Στην υποδομή *PastryStrings* υιοθετούμε δύο ευρέως χρησιμοποιημένες τεχνικές για την αντιμετώπιση αυτού του προβλήματος: (i) δημιουργία και εκμετάλλευση αντιγράφων (replicas) των δενδρικών δομών σε όλο το δίκτυο και (ii) διαμοιρασμός του φόρτου αποθήκευσης που αντιστοιχεί στην επεξεργασία δημοφιλών τιμών. Επιπλέον, μπορούμε να επιτύχουμε επιπρόσθετη κατανομή του φόρτου με την «μετατόπιση» του πεδίου ορισμού κάθε ιδιοχαρακτηριστικού αριθμητικού τύπου, πάνω στο χώρο τιμών που είναι ικανό να υποστηρίξει το σύστημα μας.

5.2.1 Δημιουργία Αντιγράφων Δέντρων

Η δημιουργία αντιγράφων ολόκληρων των δενδρικών δομών (tree replicas) μπορεί να οδηγήσει στην εξισορρόπηση του φόρτου πρόσβασης (ως αποτέλεσμα της διαδικασίας αποθήκευσης των αναγνωριστικών των συνδρομών ή της διαδικασίας εντοπισμού και συλλογής τους), αλλά και του φόρτου αποθήκευσης μιας και τα αντίγραφα κατανέμονται σε ολόκληρο το δίκτυο. Στην υποδομή *PastryStrings* ορίζουμε τον βαθμό αντιγραφής (RF), που ορίζεται ως ο αριθμός των αντιγράφων για καθένα από τα β δέντρα. Για να εκμεταλλευτούμε την ύπαρξη των αντιγράφων, τόσο κατά την επεξεργασία των συνδρομών όσο και των δημοσιεύσεων, χρησιμοποιούμε μία συνάρτηση κατακερματισμού η οποία επιστρέφει τυχαία RF τιμές για μια συγκεκριμένη είσοδο. Κατά συνέπεια, κατά τη φάση που επιθυμούμε να εντοπίσουμε τη ρίζα ενός δέντρου, χρησιμοποιούμε αυτήν την συνάρτηση κατακερματισμού ώστε να κατευθυνθούμε σε μια από τις RF διαφορετικές ρίζες του κάθε δέντρου.

Κατά τη διάρκεια της διαδικασίας επεξεργασίας των συνδρομών, θα μπορούσαμε είτε να επαναλάβουμε τη διαδικασία αποθήκευσης ενός συγκεκριμένου *SubID* RF φορές (μία για κάθε αντίγραφο) είτε να αφήσουμε τον κόμβο που επιλέχτηκε για την αποθήκευση του *SubID* να προωθήσει την αίτηση αποθήκευσης και στα υπόλοιπα αντίγραφα.

5.2.2 Διαμοιρασμός του Φόρτου Αποθήκευσης

Είναι πιθανό κάποιοι $Tnodes$ να υπερφορτωθούν λόγω της συνεχιζόμενης αποθήκευσης των $SubIDs$ των συνδρομών που έχουν ορίσει περιορισμούς σε κάποια δημοφιλή τιμή. Αυτού του είδους τα σημεία συμφόρησης θα πρέπει να αντιμετωπιστούν, ώστε να εξομαλυνθεί ο φόρτος αποθήκευσης των $SubIDs$ μεταξύ των κόμβων.

Καθορίζουμε λοιπόν ένα όριο των αποθηκευμένων $SubIDs$ σε κάθε κόμβο. Όταν κάποιος κόμβος υπερβεί αυτό το όριο, επιλέγουμε τυχαία έναν άλλο $Tnode$ για να αποθηκεύσει τα επιπλέον $SubIDs$. Κάθε $Tnode$ διατηρεί ένα πίνακα με δείκτες προς άλλους $Tnodes$ που έχουν αποθηκευμένα $SubIDs$ για την ίδια τιμή, έτσι ώστε κατά την φάση συλλογής των $SubIDs$ (επεξεργασία δημοσιεύσεων) να είναι δυνατό να να συλλέξουμε όλα τα $SubIDs$ που αντιστοιχούν στην τιμή της δημοσίευσης.

5.2.3 Μετατόπιση του Πεδίου Ορισμού

Στην υποδομή $PastryStrings$ μπορούμε να υποστηρίξουμε εξαιρετικά μεγάλα πεδία ορισμού αποτελούμενα από χιλιάδες τιμές³. Εντούτοις, κάθε ιδιοχαρακτηριστικό αριθμητικού τύπου, a_i , αναμένεται για να έχει ένα πολύ μικρότερο πεδίο ορισμού D_{ai} . Δεδομένου ότι μερικά διαστήματα τιμών αναμένονται να είναι πολύ δημοφιλή, ένα μικρό σύνολο των $Tnodes$ υπολογίζεται ότι θα απορροφήσει το μεγαλύτερο φορτίο των αιτημάτων για αποθήκευση και συλλογή των $SubIDs$. Έχοντας κατά νου αυτή την παρατήρηση, προτείνουμε να διανείμουμε κάθε D_{ai} στο χώρο D προσθέτοντας μια συγκεκριμένη σταθερά b_{ai} , στις τιμές κάθε ιδιοχαρακτηριστικού a_i . Αυτού του είδους η μετατόπιση θα επιτρέψει την κατανομή των τιμών των πεδίων ορισμού κάθε ιδιοχαρακτηριστικού σε διαφορετικούς $Tnodes$, εξισορροπώντας έτσι τον φόρτο.

Παραδείγματος χάριν, ας υποθέσουμε ότι το μοντέλο δημοσιεύσεων/συνδρομών αποτελείται από 2 ιδιοχαρακτηριστικά, ενώ τα δέντρα έχουν βαθμό $g = 2$, και το ύψος τους είναι $H=4$ (έτσι μπορούν να υποστηρίξουν πεδία ορισμού με μέγεθος $|D| = 2^4 = 16$ τιμές). Επίσης, θεωρήστε ότι η σταθερά μετατόπισης για τα δύο ιδιοχαρακτηριστικά είναι $b_{a1} = 0$, $b_{a2} = 8$, ενώ το πεδίο ορισμού τους είναι το ίδιο $D_{a1} = D_{a2} = [0, 8)$. Χωρίς την μετατόπιση του πεδίου ορισμού, και δεδομένου ότι όλες οι αριθμητικές τιμές και για τα δύο ιδιοχαρακτηριστικά θα ανήκουν στο διάστημα $[0, 8)$, το αποτέλεσμα είναι ότι όλα τα αιτήματα ανάκτησης και αποθήκευσης $SubIDs$ θα εξυπηρετηθούν από το δέντρο T_0 (όπως μπορείτε να δείτε στο Σχήμα 5.3).

Με την εφαρμογή της τεχνικής μετατόπισης του πεδίου ορισμού, προσθέτουμε τη σταθερά b_{a1} σε όλες τις τιμές του ιδιοχαρακτηριστικού a_1 , με συνέπεια το πεδίο ορισμού να είναι το ίδιο ($[0, 8)$). Κάνουμε το ίδιο και με το δεύτερο ιδιοχαρακτηριστικό προσθέτοντας τη σταθερά b_{a2} σε όλες τις τιμές του ιδιοχαρακτηριστικού

³Το μέγεθος του χώρου τιμών D που μπορεί να υποστηρίξει η $PastryStrings$ καθορίζεται από το ύψος του δέντρου H και το πλήθος των παιδιών κάθε κόμβου g : $D = g^{H+1}$

a_2 με συνέπεια, το πεδίο ορισμού του να μετατοπιστεί στην περιοχή $[8, 16)$. Κατά συνέπεια, το πρώτο ιδιοχαρακτηριστικό θα εξυπηρετηθεί από το δέντρο T_0 , ενώ το δεύτερο από το T_1 . Η σωστή επιλογή των b_{ai} , εξαρτάται από τις αρχικές ρυθμίσεις του *PastryStrings* (πλήθος ιδιοχαρακτηριστικών, μέγιστο μήκος συμβολοσειράς, πεδία ορισμού D_{ai} κ.λπ.) στοχεύοντας στην κατανομή των πεδίων ορισμού των ιδιοχαρακτηριστικών σε όλους τους κόμβους του δικτύου.

5.3 Αυτόνομη Οργάνωση της Υποδομής *PastryStrings* και Αντιμετώπιση Βλαβών

Η αυτόνομη οργάνωση χαρακτηρίζει πρωτίστως τα δίκτυα ομοτίμων εταιρών. Οι κόμβοι συνδέονται στο δίκτυο και αποχωρούν από αυτό ελεύθερα, χωρίς να υπάρχει κεντρικός έλεγχος. Δεδομένου ότι οι αναχωρήσεις και οι αφίξεις των κόμβων είναι ιδιαίτερα συχνές, η δενδρική δομή που έχουμε αναπτύξει στην υποδομή *PastryStrings* θα πρέπει να είναι σε θέση να οργανώνεται αυτόνομα και να διατηρεί την συνεκτικότητα της. Με άλλα λόγια, αν κάποιοι *Tnodes* στο εσωτερικό ενός δέντρου πάψουν να λειτουργούν (επειδή παρουσιάστηκε κάποιο σφάλμα στον κόμβο του Pastry που τους φιλοξενεί), θα πρέπει να εξασφαλίσουμε ότι τα υποδέντρα που βρίσκονται κάτω από αυτούς θα καταφέρουν να ξανασυνδεθούν στο υπόλοιπο δέντρο. Κάτι τέτοιο είναι απαραίτητο, γιατί διαφορετικά οι δημοσιεύσεις δεν θα φτάσουν ποτέ σε αυτά και έτσι συνδρομές που πιθανόν να συζευγνύουν, να μην ανακαλυφθούν ποτέ.

Στα περισσότερα δίκτυα ομοτίμων εταιρών ή σε επεκτάσεις τους (που μπορούν να αξιοποιηθούν από την υποδομή *PastryStrings*, όπως το όπως για παράδειγμα το Past/Pastry [44]), εφαρμόζεται η τεχνική της ανάπτυξης και διατήρησης αντιγράφων. Έτσι, για παράδειγμα στο Past/Pastry, αντιγράφονται τα περιεχόμενα ενός κόμβου σε $k/2$ κόμβους με αριθμητικά μεγαλύτερο αναγνωριστικό και σε $k/2$ κόμβους με μικρότερο. Δεδομένης μιας τέτοιας λειτουργικότητας ολόκληρη η δομή ενός *Tnode* (με τους πίνακες δρομολόγησης του, τα αποθηκευμένα *SubIDs*) θα φιλοξενείται και σε k άλλους κόμβους. Προκειμένου να αντιμετωπίσουμε μία πιθανή βλάβη, θα μπορούσαμε απλά να επισκεφθούμε ένα από τους k κόμβους και να ανακτήσουμε τον *Tnode* που έχουμε χάσει, επιδιορθώνοντας έτσι την συνεκτικότητα του δέντρου.

Εάν το δίκτυο δεν προσφέρει τέτοιου είδους λειτουργικότητα θα πρέπει να αναπτύξουμε συγκεκριμένους αλγόριθμους αντιμετώπισης βλαβών και διατήρησης της συνεκτικότητας των δέντρων. Προκειμένου να εφαρμοστούν αυτοί οι αλγόριθμοι, θα πρέπει να διατηρήσουμε πρόσθετη κατάσταση δρομολόγησης σε κάθε *Tnode*. Αυτή η πρόσθετη κατάσταση δρομολόγησης αποτελείται από (i) τους πίνακες δρομολόγησης των παιδιών του κάθε *Tnode* (*CRT*, Children Routing Tables), και (ii) έναν δείκτη στον πατέρα του (*FP*, Father Pointer).

Γενικά, οι πίνακες CRT αποτελούνται από $\beta \times \beta$ εγγραφές που μπορεί κάποιος να υποστηρίξει ότι αυτός ο αριθμός είναι πολύ μεγάλος. Εντούτοις, αναμένουμε ότι οι εγγραφές του πίνακα δρομολόγησης κάθε $Tnodes$ θα είναι αραιές (δηλαδή δεν θα έχουν όλες ενεργούς δείκτες), μιας και το κάθε δέντρο δεν αναμένεται να είναι πλήρες. Η πειραματική μελέτη στην ενότητα 5.5.5 επιβεβαιώνει αυτό τον ισχυρισμό.

5.3.1 Αυτόνομη Οργάνωση της Υποδομής *PastryStrings*

Άφιξη Κόμβων

Ο αλγόριθμος που εγγυάται την αυτόνομη ένταξη ενός νέου κόμβου στο δίκτυο *Pastry* εν συντομία είναι ο εξής: ο νέος κόμβος με αναγνωριστικό X στέλνει μήνυμα «άφιξης» σε έναν ήδη γνωστό κόμβο A , ο οποίος προωθηθεί έπειτα αυτό το μήνυμα στον κόμβο Z με αναγνωριστικό αριθμητικά πιο κοντά στο X . Κατόπιν όλοι οι κόμβοι στο μονοπάτι από τον A έως τον Z στέλνουν του πίνακές δρομολόγησης τους στο X , που στη συνέχεια χτίζει τον πίνακα δρομολόγησης του και ενημερώνει τους κοντινούς σε αυτόν κόμβους, για την άφιξή του.

Γενικά, τα δίκτυα ομοτίμων εταίρων είναι πολύ αραιά. Κατά συνέπεια, ένας κόμβος μπορεί να είναι αρμόδιος (με βάση το αναγνωριστικό του) για την επεξεργασία μηνυμάτων που προορίζονται σε έναν απόντα κόμβο. Στην υποδομή *PastryStrings* αυτό οδηγεί στο να φιλοξενεί ένας κόμβος περισσότερους από έναν $Tnodes$. Παραδείγματος χάριν, ο κόμβος C στο Σχήματος 5.5 φιλοξενεί τους $Tnode_{000}$ και $Tnode_{001}$. Εντούτοις, ο $Tnode_{001}$ θα έπρεπε να φιλοξενείται από ένα κόμβο που δεν είναι ακόμα συνδεδεμένος στο δίκτυο. Κατά την άφιξη ενός νέου κόμβου, το σύστημά μας πρέπει να αναδιοργανώσει κατάλληλα τη δενδρική δομή με τη μεταφορά των $Tnodes$ που αναλογούν στον κόμβο που μόλις συνδέθηκε και στη συνέχεια να του παράσχει τα $SubIDs$ για τα οποία είναι υπεύθυνος.

Όταν συνδέεται ένας νέος κόμβος στο *Pastry*, η απόφαση για το ποιοι $Tnodes$ θα πρέπει να μεταναστεύσουν σε αυτόν εξαρτάται από το αναγνωριστικό του νέου κόμβου και τις ετικέτες των $Tnodes$. Η νέα άφιξη θα γίνει αισθητή από τον πατέρα του $Tnode$ που είναι υποψήφιος να μεταφερθεί. Αυτός ο κόμβος θα κινήσει μια διαδικασία για τη μεταφορά του παιδιού του στον νέο κόμβο. Τα βασικά βήματα που πρέπει να ακολουθήσαμε σε αυτή την περίπτωση είναι τα ακόλουθα:

1. δημιουργία του $Tnode$ στον νέο κόμβο (βήμα 2 στο Σχήμα 5.5)
2. μεταφορά των $SubIDs$ στον νέο $Tnode$ (βήμα 3 στο Σχήμα 5.5)
3. αφαίρεση του $Tnode$ από τον κόμβο που μέχρι τώρα τον φιλοξενούσε (βήμα 4 στο Σχήμα 5.5)
4. ενημέρωση του πίνακα δρομολόγησης του πατέρα (βήμα 5 στο Σχήμα 5.5)

Σχήμα 5.5: Ο κόμβος F συνδέεται στην υποδομή *PastryStrings* στην οποία έχει αναπτυχθεί μέρος του δέντρου T_0 . Το T_0 οργανώνεται αυτόνομα με την άφιξη του F .

5. ενημέρωση του πίνακα CRT του παπιού (βήμα 6 στο Σχήμα 5.5)

6. ενημέρωση των παιδιών του $Tnode$ ώστε να αλλάξουν τον δείκτη FP

Στο παράδειγμα που ακολουθεί παρουσιάζεται αυτή η διαδικασία.

Παράδειγμα 7 Αυτόνομη οργάνωση του δικτύου με την άφιξη ενός νέου κόμβου.

Υποθέστε ότι ο κόμβος B (στο *Pastry*) ανιχνεύει την άφιξη του κόμβου F (Σχήμα 5.5, (1)) και ενημερώνει τον πίνακα δρομολόγησης του στο επίπεδο l προκειμένου ο αντίστοιχος δείκτης να δείχνει πλέον στο νέο κόμβο. Εάν φιλοξενεί κάποιον $Tnode$ ο οποίος βρίσκεται σε βάθος $(l - 1) \bmod \log_{\beta}(N)$ στο δέντρο του, (στο Σχήμα 5.5 ο $Tnode_{00}$), τότε ο νέος κόμβος F θα πρέπει να φιλοξενηθεί ένα από τα παιδιά του $Tnode_{00}$ ⁴. Ο νέος κόμβος F θα πρέπει λοιπόν να φιλοξενηθεί το παιδί του κόμβου (B), που αντιλήφθηκε την άφιξη του, που μέχρι εκείνη τη στιγμή φιλοξενούνταν από τον κόμβο C .

Το επόμενο βήμα περιλαμβάνει τη διαδικασία μεταφοράς του παιδιού (στο Σχήμα 5.5 το παιδί $Tnode_{001}$) στον νέο κόμβο F . Η μετανάστευση γίνεται με τη δημιουργία ενός νέου $Tnode_{001}$ στον F (Σχήμα 5.5 (2)) και τη μεταφορά των αποθηκευμένων *SubIDs* εκεί (Σχήμα 5.5, (3)). Κατόπιν διαγράφουμε τον $Tnode_{001}$ από τον κόμβο C (Σχήμα 5.5, (4)) και ενημερώνουμε τον πίνακα δρομολόγησης του πατέρα ($Tnode_{00}$) ώστε πλέον να δείχνει στον νέο κόμβο που φιλοξενεί το παιδί του, τον F . Επιπλέον, δημιουργούμε το δείκτη FP ώστε να δείχνει στον πατέρα του, $Tnode_{00}$, στον κόμβο B (Σχήμα 5.5, (5)).

⁴Εάν δεν είναι κενή η εγγραφή στον πίνακα δρομολόγησης του $Tnode$ που αντιστοιχεί σε αυτό το παιδί, όπως σε αυτό το παράδειγμα.

Δεδομένου ότι ο πίνακας δρομολόγησης του πατέρα του ($Tnode_{00}$) έχει αλλιάξει, θα πρέπει επίσης να ενημερώσουμε τον πατέρα του $Tnode_{00}$ για το νέο πίνακα δρομολόγησης του παιδιού του: κατά συνέπεια ενημερώνουμε τον πίνακα CRT που αποθηκεύεται στον $Tnode_0$ και συγκεκριμένα την εγγραφή που αντιστοιχεί στον πίνακα δρομολόγησης του $Tnode_{00}$ (Σχήμα 5.5, (6)). ◇

Αναχώρηση Κόμβων

Ένας κόμβος στο Pastry μπορεί να αποσυνδεθεί από το δίκτυο με τη θέλησή του ή να αναχωρήσει χωρίς προειδοποίηση λόγω κάποιας βλάβης. Σε αυτή την ενότητα παρουσιάζουμε την αντιμετώπιση της πρώτης περίπτωσης.

Όταν ένας κόμβος στο Pastry προετοιμάζεται να αναχωρήσει, όλοι οι φιλοξενούμενοι σε αυτόν, $Tnodes$ θα πρέπει να μεταναστεύσουν στον κόμβο που φιλοξενεί τον πατέρα τους. Αυτή η μετανάστευση περιλαμβάνει τα ακόλουθα βήματα για κάθε $Tnode$:

1. δημιουργία ενός νέου $Tnode$ στον κόμβο που φιλοξενεί τον πατέρα του
2. μεταφορά στον νέο $Tnode$ των αποθηκευμένων $SubIDs$
3. μεταφορά του πίνακα δρομολόγησης στον νέο $Tnode$
4. ενημέρωση του πίνακα δρομολόγησης του πατέρα
5. ενημέρωση του CRT του παππού
6. ενημέρωση των δεικτών FP των παιδιών του $Tnode$
7. διαγραφή του $Tnode$ από τον κόμβο που πρόκειται να αναχωρήσει

Όταν αυτή η διαδικασία ολοκληρωθεί, ο κόμβος αναχωρεί από το Pastry, και το ίδιο το Pastry παίρνει τότε τον έλεγχο ώστε να ενημερώσει τους πίνακες δρομολόγησης των κόμβων που είχαν δείκτη προς τον κόμβο που μόλις αναχώρησε. Οι πατέρες των $Tnodes$ που φιλοξενούνταν στο κόμβο που αναχώρησε θα αντιληφθούν την απουσία του κόμβου⁵ και θα ξεκινήσουν την διαδικασία διόρθωσης του πίνακα δρομολόγησης τους. Σε αυτή τη φάση, εκτελείται μία δεύτερη μεταφορά των $Tnodes$ όμοια με αυτή που περιγράφηκε στην προηγούμενη ενότητα. Στην πραγματικότητα, ο κόμβος που θα πάρει το ρόλο του κόμβου που αποχώρησε, θα αντιμετωπιστεί ως νέος κόμβος από την υποδομή $PastryStrings$ και έτσι θα ξεκινήσει η δεύτερη μετανάστευση των $Tnodes$.

⁵Είναι μία ιδιότητα που απορρέει από το τρόπο που το Pastry σχηματίζει την τοπολογία του και του τρόπου που κτίζουμε στην υποδομή $PastryStrings$ τα δέντρα πάνω στο Pastry

Σχήμα 5.6: Αναχώρηση του κόμβου B από την υποδομή *PastryStrings*. Το δέντρο T_0 αναδιοργανώνεται κατάλληλα.

5.3.2 Αντιμετώπιση Βλαβών στην Υποδομή *PastryStrings*

Η υποδομή *PastryStrings* θα πρέπει να είναι ικανή να αντιμετωπίσει τις αποτυχίες/βλάβες που είναι δυνατό να παρουσιαστούν στο δίκτυο. Οι κόμβοι στο Pastry μπορεί να αναχωρήσουν χωρίς καμία προειδοποίηση (εξαιτίας κάποιας βλάβης) και έτσι η *PastryStrings* θα πρέπει να είναι σε θέση να αντιληφθεί αυτά τα γεγονότα, και να αναδιοργανώσει την δενδρική δομή.

Σφάλματα Λειτουργίας Κόμβων και Ανάνηψη

Το Pastry παρέχει τους κατάλληλους μηχανισμούς ώστε να αντιληφθεί εάν ένας κόμβος έχει πάψει να λειτουργεί. Συγκεκριμένα, όταν αποτυγχάνουν να επικοινωνήσουν οι άμεσοι γείτονες (στο χώρο διευσθύνσεων των αναγνωριστικών τους) ενός κόμβου με αυτόν, ή κατά τη δρομολόγηση αιτήσεων ένας κόμβος δεν αποκρίνεται, τότε ο κόμβος αυτός θεωρείται ότι δεν είναι πια συνδεδεμένος στο δίκτυο. Σε αυτήν την περίπτωση, οι πίνακες δρομολόγησης των κόμβων που έχουν δείκτη προς αυτό τον κόμβο ενημερώνονται με την εύρεση ενός νέου κόμβου που θα αναλάβει τη θέση του.

Αυτός ο μηχανισμός αντιμετώπισης βλαβών στο Pastry ενεργοποιεί μία διαδικασία στην υποδομή *PastryStrings* (παρόμοια με αυτή που περιγράφηκε στην ενότητα 5.3.1) με σκοπό να αναδιοργανωθεί η δενδρική δομή.

Τα βασικά βήματα που θα πρέπει να ακολουθηθούν σε αυτή την περίπτωση είναι:

1. επιλογή ενός νέου κόμβου ώστε να φιλοξενήσει τον *Tnode* που έχει υποστεί βλάβη (βήμα 2 στο Σχήμα

5.6)

2. δημιουργία του $Tnode$ στο νέο κόμβο (βήμα 2 στο Σχήμα 5.6)
3. ενημέρωση του πίνακα δρομολόγησης του πατέρα (βήμα 3 στο Σχήμα 5.6)
4. μεταφορά του πίνακα δρομολόγησης του $Tnode$ από την αντίστοιχη εγγραφή στο CRT του παππού (βήμα 4 στο Σχήμα 5.6)
5. ενημέρωση των παιδιών του $Tnode$ ώστε να αλλάξουν τον δείκτη FP
6. μεταφορά των $SubIDs$ στον νέο $Tnode$ από ένα αντίγραφο (βήμα 5 στο Σχήμα 5.6)

Μπορεί να είναι δυνατό για έναν κόμβο στο Pastry να μη είναι διαθέσιμος για μια σύντομη χρονική περίοδο, η οποία όμως μπορεί να είναι αρκετή ώστε να θεωρηθεί ότι έχει βλάβη και έχει αποχωρήσει από το δίκτυο. Όταν εκείνος ο κόμβος επανασυνδεθεί στο δίκτυο θα ξεκινήσει η μετανάστευση των $Tnodes$ (που όμως ήδη έχει στην κατοχή του). Κατά τη διάρκεια αυτής της διαδικασίας αποκατάστασης οι $Tnodes$ που θα μεταφερθούν σε αυτόν, στην ουσία θα συγχωνευθούν με εκείνους που ο κόμβος ήδη κατέχει.

Παράδειγμα 8 Αυτόνομη οργάνωση της υποδομής PastryStrings κατά την βλάβη ενός κόμβου.

Υποθέστε τώρα πως ο κόμβος που έχει υποστεί βλάβη είναι ο B που φιλοξενεί τους $Tnode_{00}$ (Σχήμα 5.6, (1)) και πως ο κόμβος A που φιλοξενεί τον πατέρα του ($Tnode_0$), παρατηρεί την απουσία του. Οι σχετικοί μηχανισμοί του Pastry θα αναλάβουν να ενημερώσουν τον πίνακα δρομολόγησης του A ώστε η αντίστοιχη εγγραφή να δείχνει πλέον στον κόμβο, D (που έχει πάρει τη θέση του B). Αυτή η ενέργεια θα αναγκάσει τον πατέρα του $Tnode_{00}$ ($Tnode_0$) να δημιουργήσει ένα νέο $Tnode_{00}$ στον κόμβο D (Σχήμα 5.6, (2)) και να ενημερώσει τον πίνακα δρομολόγησης του δείχνοντας τώρα στον D (Σχήμα 5.6, (3)).

Το επόμενο βήμα αφορά τον $Tnode_{00}$ και την ανακάλυψη των παιδιών του ($Tnode_{000}$ και $Tnode_{001}$). Αυτό επιτυγχάνεται με τη μεταφορά του πίνακα δρομολόγησης του που είναι αποθηκευμένος στο CRT του πατέρα του (Σχήμα 5.6, (4)). Κατόπιν τα παιδιά του έρχονται σε επαφή με αυτόν ώστε να ενημερωθεί ο δείκτης FP . Το τελευταίο βήμα περιλαμβάνει τη διαδικασία μεταφοράς των $SubIDs$ που διαχειρίζεται ο $Tnode_{00}$ από ένα αντίγραφο (Σχήμα 5.6, (5)). ◇

Κάποιος μπορεί να αναρωτηθεί τι συμβαίνει με τις ρίζες των δέντρων και ποιος διατηρεί ένα CRT για αυτές. Μέχρι τώρα, θεωρήσαμε ότι προκειμένου να εντοπιστεί η ρίζα $Tnode_c$ του δέντρου T_c , θα πρέπει να χρησιμοποιήσουμε την συνάρτηση κατακερματισμού του Pastry, περνώντας το χαρακτήρα c ως όρισμα και προώθηση της αίτησης προς τον κόμβο με αναγνωριστικό ίσο με την τιμή που επιστρέφει η συνάρτηση.

Προκειμένου να αντιμετωπίσουμε τις βλάβες των κόμβων που φιλοξενούν κάποια ρίζα, επεκτείνουμε αυτή τη διαδικασία.

Σε κάθε ρίζα αντιστοιχούν k τυχαίοι κόμβοι, $RHolders$ που είναι αρμόδιοι να διατηρούν ένα δείκτη προς τη ρίζα. Κατά την εξυπηρέτηση μιας αίτησης προς ένα συγκεκριμένο δέντρο, για παράδειγμα το δέντρο T_c , αυτή η αίτηση θα φθάσει σε ένα από τους k $RHolder_{c,i}$, που με τη σειρά τους θα προωθήσουν την αίτηση στη ρίζα $Tnode_c$.

Με σκοπό να εντοπίσουμε i -οστό $RHolder_{c,i}$ που θα μας οδηγήσει στη ρίζα $Tnode_c$, ακολουθούμε την παρακάτω διαδικασία. Ας ορίσουμε αρχικά ορισμένες συναρτήσεις που θα μας βοηθήσουν στην παρουσίαση:

- $DHT.hash(label)$: η συνάρτηση κατακερματισμού του Pastry. Επιστρέφει το αναγνωριστικό του κόμβου που αντιστοιχίζεται με τη συμβολοσειρά $label$.
- $alphabet_index(c)$: επιστρέφει τη θέση του χαρακτήρα c στον πίνακα που περιέχει τους β διαθέσιμους χαρακτήρες του αλφαβήτου μας.
- $alphabet_char(j)$: επιστρέφει το χαρακτήρα που βρίσκεται στη θέση j στον πίνακα των β χαρακτήρων.
- $Idx(i, c) = (alphabet_index(c) + i) \bmod \beta$: επιστρέφει μία θέση στον πίνακα χαρακτήρων που εξαρτάται από τον $RHolder$ (i) και το χαρακτήρα που διαχειρίζεται το δέντρο της ρίζας (c)
- $Chr(i, c) = alphabet_char(Idx(i, c)) + c$: επιστρέφει την ένωση του χαρακτήρα c με τον χαρακτήρα στη θέση $Idx(i, c)$ στον πίνακα χαρακτήρων.

Έχοντας στη διάθεση μας τις παραπάνω συναρτήσεις, ο i -οστός $RHolder_{c,i}$ με δείκτη στη ρίζα του δέντρου T_c μπορεί να βρεθεί στον κόμβο με αναγνωριστικό

$$id = DHT.hash(Chr(i, c))$$

Αν για παράδειγμα το αλφάβητο μας αποτελείται από 10 χαρακτήρες ($\beta = 10$ και $alphabet = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]$, και το πλήθος των $RHolders$ που συσχετίζονται με κάθε ρίζα είναι $k = 5$, τότε για να εντοπίσουμε την ρίζα του δέντρου T_2 , θα πρέπει πρώτα να εντοπίσουμε ένα από τους 5 $RHolders$, π.χ. τον 3^ο ($RHolder_{2,3}$). Χρησιμοποιώντας τις συναρτήσεις που μόλις παρουσιάσαμε, προκύπτουν οι εξής τιμές:

$$alphabet_index('2') = 2$$

$$Idx(3, '2') = (2 + 3) \bmod 10 = 5$$

$$Chr(3, '2') = alphabet_char(5) + '2' = '5' + '2' = '52'$$

Χρησιμοποιώντας τη συνάρτηση κατακερματισμού του Pastry και την συμβολοσειράς '52' σαν όρισμα, θα μπορέσουμε να εντοπίσουμε τον κόμβο που φιλοξενεί τον 3^ο *RHolder*. Καθένας από τους *RHolders* είναι αρμόδιος να διατηρεί μαζί με τον δείκτη προς στη ρίζα και τον πίνακα δρομολόγησης της (που είναι στην πραγματικότητα το *CRT* με μόνο μια εγγραφή). Όταν ο κόμβος που φιλοξενεί τη ρίζα υποστεί βλάβη και αποσυνδεθεί από την υποδομή *PastryStrings*, επικοινωνούμε με έναν από τους *RHolders* ώστε να αντιγράψουμε τον πίνακα δρομολόγησης της ρίζας στη νέα ρίζα που πρόκειται να δημιουργήσουμε. Όταν η ρίζα αλλάζει θέση (φιλοξενείται σε νέο κόμβο λόγω βλάβης) τότε όλοι οι *RHolders* της ρίζας ενημερώνουν τους δείκτες τους, ώστε να δείχνουν τώρα τη νέα ρίζα. Η έναρξη της διαδικασίας μεταφοράς της ρίζας πραγματοποιείται κατά την εξυπηρέτηση αιτήσεων προς το συγκεκριμένο δέντρο και όταν ο εκάστοτε *RHolder* αντιληφθεί ότι η ρίζα δεν ανταποκρίνεται στην αίτηση δρομολόγησης.

Είναι δυνατό (όπως θα δούμε στην ενότητα 5.3.2) δύο ή περισσότεροι *RHolders* να επιλέξουν ανεξάρτητα και ταυτόχρονα ένα κόμβο για να φιλοξενήσει τη νέα ρίζα. Σε αυτήν την περίπτωση, ξεκινάμε μια διαδικασία καταμετρημένου συγχρονισμού μεταξύ των *RHolders* που θα οδηγήσει τελικά στην από κοινού επιλογή του κόμβου που θα φιλοξενήσει την ρίζα.

Οργάνωση των Δέντρων υπό την Παρουσία Πολλαπλών Βλαβών

Μέχρι τώρα, περιγράψαμε μια διαδικασία που ενεργοποιείται από τον πατέρα του *Tnode* που έχει υποστεί βλάβη. Εντούτοις, εάν το δίκτυο υποφέρει από πολλαπλές και συνεχιζόμενες βλάβες, είναι πιθανό ολόκληρα υποδέντρα να χάσουν τη συνεκτικότητα τους με το υπόλοιπο δέντρο και να καταστούν έτσι απροσπέλαστα. Παραδείγματος χάριν, στο Σχήμα 5.6, εάν οι κόμβοι *A*, *B*, και *D* αποσυνδεθούν, τότε ο κόμβος *Tnode₀₀₀* που φιλοξενείται στον *C* θα μείνει ορφανός, με συνέπεια τα αποθηκευμένα σε αυτόν *SubIDs* να μην μπορούν να εντοπιστούν. Σε αυτήν την περίπτωση πρέπει να εξασφαλίσουμε ότι αυτός ο *Tnode* (που μπορεί γενικά να είναι η ρίζα ενός ολόκληρου υποδέντρου) θα ανακαλυφθεί και θα επανασυνδεθεί με το υπόλοιπο δέντρο.

Κάθε *Tnode* έρχεται περιοδικά σε επαφή με τον πατέρα του για να εξακριβώσει τη κατάσταση του. Όταν ο πατέρας δεν αποκρίνεται υποθέτουμε ότι ο κόμβος που τον φιλοξενεί έχει υποστεί κάποια βλάβη και έτσι περιμένουμε τον παππού να αντιληφθεί αυτό το γεγονός και να ξεκινήσει τη διαδικασία αποκατάστασης που παρουσιάστηκε στην ενότητα 5.3.2. Εάν, εντούτοις, οι διαδοχικές προσπάθειες να επικοινωνήσουν με τον πατέρα αποτύχουν υποθέτουμε ότι και ο παππούς έχει αποτύχει επίσης, και έτσι το *Tnode* θα πρέπει

να πάρει τον έλεγχο στα χέρια του, προκειμένου να ανακαλύψει και να επανασυνδεθεί με τον πατέρα του.

Το *Tnode* που ανακάλυψε την βλάβη, στέλνει ένα ειδικού τύπου αίτημα δρομολόγησης (*locate_father(label)*) στην υποδομή *PastryStrings* προς το *Tnode* με ετικέτα την ετικέτα του ελλείποντος πατέρα του (που στην ουσία είναι η ίδια η ετικέτα του χωρίς τον τελευταίο χαρακτήρα). Αυτή η αίτηση θα μας οδηγήσει στον εντοπισμό της ρίζας του κατάλληλου δέντρου (στο οποίο ανήκει το *Tnode*), όπου και θα ακολουθήσουμε ένα μονοπάτι προς τον πατέρα του. Κατά τη διάρκεια αυτής της δρομολόγησης του μηνύματος μέσα στο δέντρο, θα κατασκευαστούν νέοι κόμβοι σε περίπτωση που αυτοί δεν υπάρχουν, και τελικά θα φτάσουμε στον πατέρα του. Ο πατέρας στη συνέχεια θα ενημερωθεί για την ύπαρξη του παιδιού του και θα ενημερώσει τον πίνακα δρομολόγησης του ώστε να δείχνει σε αυτό. Αντίστοιχα, ο *Tnode* θα ενημερώσει τον δείκτη *FP* ώστε να δείχνει στον πατέρα του.

Δεδομένου ότι μπορεί να υπάρξουν αιτήσεις αποθήκευσης των *SubIDs* (σαν αποτέλεσμα της επεξεργασίας των συνδρομών) όσο διαρκεί αυτή η διαδικασία, είναι πιθανό να κατασκευαστεί ένας *Tnode* που μπορεί όμως ήδη να υπάρχει στο ορφανό δέντρο. Σε αυτήν την περίπτωση συνεχίζουμε τη διαδικασία αποκατάστασης με τη συγχώνευση του διπλών *Tnodes*. Συγκεκριμένα, όταν ο πατέρας λάβει το ειδικού τύπου αίτημα δρομολόγησης (*locate_father(label)*) από το ορφανό παιδί, και πριν ενημερώσει τον πίνακα δρομολόγησης του, τον ελέγχει ώστε να εντοπίσει αν υπάρχει παιδί με ίδια ετικέτα με το ορφανό παιδί. Εάν ήδη υπάρχει, ο πατέρας θα στείλει ένα μήνυμα συγχώνευσης σε όλα τα *Tnodes* που ανήκουν στο υποδέντρο με ρίζα το διπλό παιδί και έπειτα θα ενημερώσει τον πίνακα δρομολόγησης του ώστε η αντίστοιχη εγγραφή να δείχνει στο ορφανό δέντρο.

Κάθε *Tnode* που λαμβάνει την αίτηση συγχώνευσης από τον πατέρα του, θα στείλει στη συνέχεια μία αίτηση συγχώνευσης (*merge(label)*) με την *label* ίδια με την ετικέτα του. Αυτό το αίτημα θα ταξιδέψει πάνω στην υποδομή *PastryStrings* και τελικά θα φθάσει στο δίδυμο *Tnode* στο ορφανό υποδέντρο. Σε αυτό το σημείο τα δύο ίδια *Tnodes* γνωρίζουν το ένα για το άλλο, και θα ξεκινήσουν τη διαδικασία συγχώνευσης. Το *Tnode* που ανήκει στο ορφανό δέντρο θα συνεχίσει να λειτουργεί ενώ το ο δίδυμος αδελφός του θα αφαιρεθεί από το δίκτυο, αφού πρώτα στείλει τα αποθηκευμένα σε αυτό *SubIDs* στον αδελφό του. Επιλέγουμε να διατηρήσουμε το ορφανό δέντρο επανασυνδέοντας το με το πατέρα του, γιατί αναμένεται ότι θα έχει αποθηκευμένα περισσότερα *SubIDs*.

Ένα ζήτημα που μπορεί να προκύψει κατά τη διάρκεια της διαδικασίας αποκατάστασης σχετίζεται με την ταυτόχρονη έναρξη της διαδικασίας αποκατάστασης από δύο *Tnodes* που βρίσκονται στο ορφανό δέντρο στο ίδιο επίπεδο. Αυτό δεν αποτελεί σημαντικό πρόβλημα, επειδή σε μια τέτοια περίπτωση το πρώτο *locate_father(...)* αίτημα θα εντοπίσει το πατέρα τους, ενώ τα υπόλοιπα απλά θα ακολουθήσουν το μονοπάτι που δημιούργησε το πρώτο αίτημα και θα αγνοηθούν στη συνέχεια.

Σχήμα 5.7: Αποκατάσταση του δέντρου υπό την παρουσία συχνών βλαβών. Τα $Tnode_{00}$ και $Tnode_{000}$ απουσυνδέονται. Το υποδέντρο με ρίζα τον $Tnode_{0001}$ επανασυνδέεται στο υπόλοιπο δέντρο

Παράδειγμα 9 Αποκατάσταση του δέντρου υπό την παρουσία συχνών βλαβών.

Στο Σχήμα 5.7 (αριστερό τμήμα) μπορείτε να δείτε το δέντρο T_0 με δύο κόμβους να έχουν υποστεί βλάβη: $Tnode_{00}$ και $Tnode_{000}$.

Αφού ο $Tnode_{0001}$ έχει διαπιστώσει ότι ο πατέρα του (χρησιμοποιώντας τον δείκτη FP) έχει υποστεί βλάβη, στέλνει μία αίτηση δρομολόγησης για τον εντοπισμό του πατέρα του : $locate_father(000)$. Ο πατέρας $Tnode_{000}$ εντοπίζεται, εντούτοις όπως μπορείτε να παρατηρήσετε, στο μεταξύ ένα αίτημα αποθήκευσης $SubID$ οδήγησε στη δημιουργία του $Tnode_{0001}$ και του $Tnode_{00011}$. Αυτοί οι δύο $Tnodes$ έχουν διδυμα αδέρφια στο ορφανό υποδέντρο με ρίζα τον $Tnode_{0001}$ (λευκοί $Tnodes$ στο Σχήμα 5.7).

Σε αυτήν την περίπτωση, και πριν από την ανανέωση του πίνακα δρομολόγησης του πατέρα (ώστε πλέον να δείχνει στο ορφανό υποδέντρο), ο πατέρας στέλνει ένα αίτημα συγχώνευσης στα παιδιά του (διαστιγμένοι κόμβοι στο Σχήμα 5.7) ώστε να ξεκινήσει η διαδικασία συγχώνευσης των διδυμων παιδιών. Στη συνέχεια, και αφού τα παιδιά ($Tnode_{0001}$ και $Tnode_{00011}$) λάβουν αυτό το μήνυμα, στέλνουν το μήνυμα συγχώνευσης ($merge(0001)$ και $merge(00011)$ αντίστοιχα) ώστε να εντοπίσουν τους διδύμους τους στο ορφανό υποδέντρο, και να τους μεταφέρουν τα αποθηκευμένα $SubIDs$. Έπειτα αυτοί διαγράφονται από το δέντρο. Το αποτέλεσμα αυτής της διαδικασίας είναι η επανασύνδεση του ορφανού υποδέντρο με το υπόλοιπο δέντρο. ◇

Διαθεσιμότητα και Ανάνηψη με την Διατήρηση Αντιγράφων

Γενικά, το Pastry καθώς επίσης και όλα τα DHTs παρέχουν μηχανισμούς αντιμετώπισης και ανάνηψης υπό την παρουσία βλαβών με την διατήρηση αντιγράφων. Στο Pastry και ειδικότερα στο σύστημα διαχείρισης αρχείων Past [44], τα αντικείμενα που αποθηκεύονται σε ένα κόμβο αντιγράφονται σε k κόμβους των οποίων το αναγνωριστικό είναι αριθμητικά κοντά στο αναγνωριστικό του κόμβου.

Επεξεργασία Συνδρομών	
Ιδιοχ. Αριθμητικού Τύπου $O(\beta \times \log_\beta(R) \times \log_\beta(N))$	Ιδιοχ. Τύπου Συμβολοσειρές $l+O(\log_\beta(N))$
Επεξεργασία Δημοσιεύσεων	
Ιδιοχ. Αριθμητικού Τύπου $H+O(\log_\beta(N))$	Ιδιοχ. Τύπου Συμβολοσειρές $l+O(\log_\beta(N))$

Πίνακας 5.1: Πολυπλοκότητα επεξεργασίας συνδρομών και δημοσιεύσεων στην υποδομή *PastryStrings*.

Θα μπορούσαμε να στηριχθούμε στο συγκεκριμένο μηχανισμό του Pastry ώστε να αντιμετωπίσουμε τις πιθανές βλάβες στο δίκτυο. Εντούτοις, όπως ήδη παρουσιάσαμε, προτείναμε νέους αλγορίθμους που λειτουργούν πάνω από το Pastry στην υποδομή *PastryStrings*, βοηθώντας περαιτέρω στην αποκατάσταση των δενδρικών δομών και της λειτουργικότητας γενικά. Επιλέγουμε λοιπόν να αντιγράψουμε ολόκληρα τα δέντρα στο δίκτυο. Δεδομένου ότι ένα αντίγραφο (replica) *Tnode* μπορεί να φιλοξενείται σε έναν κόμβο πολλά βήματα μακριά (από την περιοχή που ανήκει το *Tnode* και παρουσιάζονται οι βλάβες), μπορούμε να εγγυηθούμε ότι συνεχόμενες βλάβες σε μία περιοχή δεν έχουν επιπτώσεις στη δομή των δέντρων μας, και ότι τα δέντρα στη *PastryStrings* θα ανακτήσουν την συνεκτικότητα τους με ενεργοποίηση της διαδικασίας αναδιοργάνωσης που περιγράψαμε.

5.4 Θεωρητική Ανάλυση Απόδοσης

Όπως έχει ήδη αναφερθεί, στο δίκτυο Pastry η δρομολόγηση αιτήσεων απαιτεί την εκτέλεση $O(\log_\beta(N))$ βημάτων σε ένα δίκτυο με N κόμβους (και αλφάβητο β χαρακτήρων για τα αναγνωριστικά των αντικειμένων και των κόμβων).

Η επεξεργασία συνδρομών με ιδιοχαρακτηριστικά τύπου συμβολοσειράς, απαιτεί την εκτέλεση $O(\log_\beta(N))$ βημάτων ώστε να εντοπίσουμε τη ρίζα του κατάλληλου δέντρου και στη συνέχεια το πολύ l βήματα για να εντοπίσουμε τον κατάλληλο *Tnode* όπου θα αποθηκεύσουμε το αναγνωριστικό *SubID*, για συμβολοσειρές με μέγεθος l χαρακτήρων. Οπότε συνολικά απαιτούνται $l+O(\log_\beta(N))$ βήματα.

Κατά την επεξεργασία των συνδρομών με ιδιοχαρακτηριστικά αριθμητικού τύπου, ένα οποιοδήποτε διάστημα R θα πρέπει να διασπασθεί στο κατάλληλο σύνολο από υποδιαστήματα εφαρμόζοντας τον αλγόριθμο διάσπασης διαστημάτων που παρουσιάστηκε στον Πίνακα 4.1. Το πλήθος των υποδιαστημάτων που προκύπτει είναι ίσο με $O(2 \times \beta \times \lfloor \log_\beta |R| \rfloor - \beta)$. Επειδή για κάθε υποδιάστημα θα πρέπει να επικοινωνήσουμε με ένα κόμβο, προκύπτει ότι τελικά θα πρέπει να εκτελέσουμε $O(\beta \times \log_\beta(|R|) \times \log_\beta(N))$ βήματα.

Κατά την επεξεργασία των δημοσιεύσεων, αρχικά θα πρέπει να εντοπίσουμε το κατάλληλο δέντρο, κάτι που απαιτεί την εκτέλεση $O(\log_\beta(N))$ βημάτων. Στη συνέχεια θα πρέπει να εντοπίσουμε τον κατάλληλο *Tnode* μέσα στο δέντρο κάτι που απαιτεί το πολύ H βήματα κατά την επεξεργασία ιδιοχαρακτηριστικών

αριθμητικού τύπου (μιας και πρέπει να φτάσουμε σε κάποιο από τα φύλλα του δέντρου) και $2 \times l$ κατά την επεξεργασία ιδιοχαρακτηριστικών με συμβολοσειρά μήκους l χαρακτήρων (θυμηθείτε ότι για κάθε συμβολοσειρά θα πρέπει να εκτελέσουμε την ίδια διαδικασία μία φορά για την εύρεση των συζευγνυόμενων συνδρομών με περιορισμό προθέματος και μία ακόμα για περιορισμό επιθέματος).

Στον Πίνακα 5.1, μπορεί να δει κανείς συγκεντρωτικά την πολυπλοκότητα αναφορικά με το πλήθος των μηνυμάτων, κατά την επεξεργασία των συνδρομών και των δημοσιεύσεων.

5.5 Πειραματική Αξιολόγηση και Μελέτη Απόδοσης

5.5.1 Χαρακτηριστικά και Παράμετροι Πειραματικής Διάταξης

Εκτελέσαμε μια σειρά πειραμάτων σε ένα δίκτυο που προσομοιώνει την λειτουργικότητα του Pastry με την υποδομή *PastryStrings* να λειτουργεί πάνω σε αυτό. Το δίκτυο μας αποτελείται 1000 κόμβους. Παρόμοια αποτελέσματα προκύπτουν και για δίκτυα με πολύ περισσότερους κόμβους.

Το μοντέλο δημοσιεύσεων/συνδρομών που εφαρμόσαμε σ' αυτά τα πειράματα, αποτελείται από 15 ιδιοχαρακτηριστικά. Ο ακριβής αριθμός αυτών που εμπλέκονται στη δήλωση των δημοσιεύσεων και των συνδρομών καθορίζεται από την κατανομή Zipf της δημοτικότητας των ιδιοχαρακτηριστικών. Η δημοτικότητα των τιμών των ιδιοχαρακτηριστικών ακολουθεί και αυτή την Zipf κατανομή και μιας και παίζει σημαντικό ρόλο (για παράδειγμα στην κατανομή του φόρτου αποθήκευσης των *SubIDs*, ή του φόρτου προσπέλασης των κόμβων), μεταβάλλουμε την κλίση της δίνοντας τιμές στη παράμετρο θ που κυμαίνονται από 0.0 μέχρι 1.4.

Τα αποτελέσματα μας προέκυψαν κατά την επεξεργασία περίπου 3,000,000 *SubIDs* και περίπου 16,000,000 αιτήσεων για ανάκτηση αυτών. Ο ακριβής αριθμός εξαρτάται από την κλίση των σχετικών κατανομών δημοτικότητας Zipf (που μεταφράζεται στην τιμή της θ). Τα μισά ιδιοχαρακτηριστικά είναι αλφαριθμητικού τύπου, όπου στο 50% αυτών ορίζονται περιορισμοί διαστημάτων τιμών και στο υπόλοιπο 50% περιορισμοί ισότητας. Από τα ιδιοχαρακτηριστικά τύπου συμβολοσειράς, στο 40% ορίζονται περιορισμοί ισότητας, στο 30% περιορισμοί προθέματος και στο 30% περιορισμοί επιθέματος.

Το πεδίο ορισμού των ιδιοχαρακτηριστικών αριθμητικού τύπου αποτελείται από 70,000 τιμές (στο διάστημα $[0, 70000]$), ενώ το μέγεθος των διαστημάτων τιμών που ορίζονται στις συνδρομές, εκτός και αν δηλωθεί και διαφορετικό, μεταβάλλεται ομοιόμορφα από 1 μέχρι 20. Το πλήθος των διαθέσιμων χαρακτήρων, β , είναι ίσο με 20 ενώ οι συμβολοσειρές έχουν μέγιστο μέγεθος 5 χαρακτήρων.

Σχήμα 5.8: Η επίδραση των ιδιοχαρακτηριστικών αριθμητικού τύπου με περιορισμούς διαστήματος στο πλήθος των μηνυμάτων κατά την επεξεργασία των συνδρομών.

5.5.2 Η Επίδραση του Μεγέθους των Διαστημάτων στην Επεξεργασία των Συνδρομών

Σε αυτή τη σειρά πειραμάτων (αποτελέσματα μπορείτε να δείτε στο Σχήμα 5.8) μεταβάλλουμε το ποσοστό των ιδιοχαρακτηριστικών με αριθμητικές τιμές (με περιορισμούς ισότητας ή διαστήματος) σε κάθε συνδρομή από 0.0% έως 100%. Η μετρική που χρησιμοποιούμε εδώ είναι ο αριθμός μηνυμάτων που απαιτούνται για την αποθήκευση μιας συνδρομής. Μπορείτε να δείτε ότι καθώς το ποσοστό των ιδιοχαρακτηριστικών ανά συνδρομή με διαστήματα τιμών αυξάνει, το ίδιο παρατηρείται και με τον αριθμό των μηνυμάτων. Στην πραγματικότητα ο αριθμός των μηνυμάτων στην περίπτωση που οι συνδρομές αποτελούνται μόνο από ιδιοχαρακτηριστικά με αριθμητικές τιμές (ποσοστό 100%) είναι τέσσερις φορές μεγαλύτερος από την περίπτωση όπου το ποσοστό είναι 0%. Αυτό είναι αναμενόμενο, μιας και τα διαστήματα τιμών πρέπει να διασπαστούν σε υποδιαστήματα και στη συνέχεια θα πρέπει να έρθουμε σε επαφή με κάθε $Tnode$ που διαχειρίζεται αυτά τα υποδιαστήματα. Οπότε όσο περισσότερα ιδιοχαρακτηριστικά με διαστήματα τιμών ορίζονται σε μία δημοσίευση, τόσο αυξάνει και ο αριθμός των $Tnodes$ που πρέπει να εντοπίσουμε.

Για να έχουμε μια σαφή εικόνα όσον αφορά τον αριθμό των υποδιαστημάτων που τεμαχίζεται ένα διάστημα (και άρα του αριθμού των $Tnodes$ που θα πρέπει να επικοινωνήσουμε), μεταβάλλαμε το μέγεθος του διαστήματος και μετρήσαμε τα αντίστοιχα υποδιαστήματα που προκύπτουν με την εφαρμογή του αλγορίθμου διάσπασης (Πίνακας 4.1). Συγκεκριμένα, για κάθε διάστημα, $|R|$ δημιουργήσαμε 100,000 διαστήματα

Σχήμα 5.9: Η επίδραση του μεγέθους του διαστήματος τιμών στο πλήθος των παραγόμενων υποδιαστημάτων με την εφαρμογή του αλγόριθμου διάσπασης.

με μέγεθος που κυμαίνεται από 0 ως το $|R|$ ακολουθώντας ομοιόμορφη κατανομή. Για κάθε ένα από αυτά τα διαστήματα, εφαρμόσαμε τον αλγόριθμο διάσπασης και μετρήσαμε τα παραγόμενα υποδιαστήματα. Στο Σχήμα 5.9 παρουσιάζουμε τη μέση και τη μέγιστη τιμή του πλήθους των υποδιαστημάτων σε κάθε περίπτωση. Μπορείτε να δείτε ότι όσο το μέγεθος του διαστήματος αυξάνει, το ίδιο παρατηρείται και στον αριθμό των υποδιαστημάτων. Όμως, όπως δείχνει και η θεωρητική μελέτη μας⁶, η αύξηση είναι λογαριθμική.

Στο ίδιο Σχήμα μπορείτε επίσης να παρατηρήσετε ότι ο μέγιστος αριθμός υποδιαστημάτων στα οποία τεμαχίζεται ένα διάστημα R με $|R| = 10,000$ είναι 42 που αποτελεί μία σαφή ένδειξη του αριθμού των υποδιαστημάτων (και άρα του αριθμού των αιτήσεων αποθήκευσης του *SubID*) που αντιστοιχεί σε ένα περιορισμό ανισότητας σε ιδιοχαρακτηριστικό a_i με μέγεθος πεδίου ορισμού $|D_{ai}| = 10,000$.

5.5.3 Εξισορρόπηση του Φόρτου Αποθήκευσης και Ανάκτησης των SubIDs

Σε αυτή την ενότητα μελετάμε το πως επηρεάζεται η κατανομή του φόρτου στο δίκτυο από την κλίση της κατανομής της δημοτικότητας των ιδιοχαρακτηριστικών και των τιμών τους.

Η μετρική απόδοσής που χρησιμοποιούμε είναι ο συντελεστής διακύμανσης (CV-coefficient of variation) του φόρτου πρόσβασης των *Tnodes* (access load CV) και αποθήκευσης των *SubIDs* (storage load CV). Ο συντελεστής διακύμανσης για τον φόρτο αποθήκευσης ορίζεται ως ο λόγος της τυπικής απόκλισης των

⁶Χρησιμοποιώντας τον αλγόριθμό διάσπασης του Πίνακα 4.1, μπορούμε να διασπάσουμε ένα διάστημα R το πολύ σε $2 \times \beta \times \lfloor \log_\beta |R| \rfloor - \beta$ υποδιαστήματα τα οποία δεν επικαλύπτονται και η ένωσή τους μας δίνει το διάστημα R .

Σχήμα 5.10: Εξισορρόπηση του φόρτου ανάκτησης και αποθήκευσης των *SubIDs* μεταβάλλοντας την κατανομή της δημοτικότητας των ιδιοχαρακτηριστικών.

αποθηκευμένων *SubIDs* ανά κόμβο, προς τη μέση τιμή αυτών. Ο συντελεστής διακύμανσης για τον φόρτο πρόσβασης ορίζεται αντίστοιχα και περιλαμβάνει τις αιτήσεις δρομολόγησης και τις αιτήσεις ανάκτησης των *SubIDs* που δέχεται κάθε κόμβος. Μας ενδιαφέρει η τιμή του συντελεστή διακύμανσης να είναι όσο το δυνατόν μικρότερη, μιας και όταν αυτή πλησιάζει το 0 το δίκτυο μας θα είναι πλήρως ισορροπημένο.

Η Επίδραση της Κατανομής της Δημοτικότητας των Ιδιοχαρακτηριστικών και των Τιμών τους

Όπως μπορείτε να δείτε στο Σχήμα 5.10 ο συντελεστής διακύμανσης του φόρτου αποθήκευσης (Storage Load) και πρόσβασης (Access Load) αυξάνεται ελαφρώς καθώς η κλίση της κατανομής της δημοτικότητας των ιδιοχαρακτηριστικών αυξάνεται (μεταβάλλοντας την τιμή της παραμέτρου θ της Zipf κατανομής), επειδή μειώνεται ο αριθμός των ιδιοχαρακτηριστικών που ορίζονται σε κάθε δημοσίευση ή συνδρομή, με αποτέλεσμα πλέον να υπάρχουν λίγα και αρκετά δημοφιλή ιδιοχαρακτηριστικά. Η ύπαρξη λιγότερων ιδιοχαρακτηριστικών οδηγεί σε αυξημένο φόρτο εργασίας για ένα μικρό σύνολο των κόμβων, κάτι που επηρεάζει αρνητικά την κατανομή του φόρτου στο δίκτυο. Εντούτοις, η διαφορά μεταξύ των κατανομών δημοτικότητας με $\theta = 0.0$ και 1.4 είναι σχετικά μικρή.

Στο Σχήμα 5.11 παρουσιάζεται η αύξηση του συντελεστή διακύμανσης του φόρτου αποθήκευσης και πρόσβασης με την αύξηση της τιμής της παραμέτρου θ . Η αύξηση αυτή οφείλεται στο γεγονός ότι όσο αυξάνει το θ τόσο λιγότερες και δημοφιλέστερες τιμές ορίζονται σε κάθε συνδρομή ή δημοσίευση. Αυτό οδηγεί στην υπερφόρτωση ενός μικρού συνόλου από *Tnodes* στην υποδομή *PastryStrings* ενώ οι περισσότεροι

Σχήμα 5.11: Εξισορρόπηση του φόρτου ανάκτησης και αποθήκευσης των *SubIDs* μεταβάλλοντας την κατανομή δημοτικότητας των τιμών των ιδιοχαρακτηριστικών.

Tnodes δεν προσπελάζονται σχεδόν καθόλου .

Για να μελετήσουμε πώς η κατανομή του φόρτου αποθήκευσης μπορεί να βελτιωθεί με την εφαρμογή της τεχνικής μετατόπισης του πεδίου ορισμού των ιδιοχαρακτηριστικών αριθμητικού τύπου (που παρουσιάστηκε στην ενότητα 5.2) εκτελέσαμε ένα σύνολο πειραμάτων μεταβάλλοντας την παράμετρο θ της κατανομής της δημοτικότητας των τιμών. Παρατηρήσαμε λοιπόν ότι ο συντελεστή διακύμανσης του φόρτου αποθήκευσης επηρεάζεται ελάχιστα από την αύξηση της κλίσης της κατανομής της δημοτικότητας (σε αντίθεση με τα αποτελέσματα του Σχήματος 5.11).

Αυτό οφείλεται στο γεγονός ότι η επεξεργασία των ιδιοχαρακτηριστικών με διαστήματα τιμών επισκιάζει την επεξεργασία των συμβολοσειρών (επειδή θα πρέπει να επισκεφθούμε πολύ περισσότερους *Tnodes*). Έτσι, η τεχνική της μετατόπισης του πεδίου ορισμού που αφορά αποκλειστικά τα ιδιοχαρακτηριστικά αριθμητικού τύπου, δείχνει να βοηθάει στην κατανομή των δημοφιλών τιμών σε περισσότερους κόμβους και άρα να μειώνει τις ανισοροπίες στο φόρτο αποθήκευσης που παρουσιάζονται διαφορετικά.

Η Επίδραση της Διατήρησης Πολλαπλών Αντιγράφων

Σε αυτή την ενότητα μετρήσουμε πώς η δημιουργία αντιγράφων (replicas) των δέντρων επηρεάζει την εξισορρόπηση του φόρτου την αποθήκευσης και πρόσβασης, μεταβάλλοντας τον συντελεστή *RF* (Replication Factor) που ορίζει τον αριθμό των αντιγράφων που αντιστοιχούν σε κάθε δέντρο στην υποδομή *Pastry-Strings*.

Σχήμα 5.12: Εξισορρόπηση του φόρτου ανάκτησης και αποθήκευσης των *SubIDs*.

Όπως μπορείτε να δείτε στο Σχήμα 5.12 ο συντελεστής ο συντελεστή διακύμανσης του φόρτου αποθήκευσης (Storage Load CV) και του φόρτου πρόσβασης (Access Load CV) μειώνεται και προσεγγίζει την τιμή 0 (πλήρως εξισορροπημένο δίκτυο), καθώς αυξάνεται ο αριθμός των αντιγράφων ανά δέντρο (*RF*). Μια άλλη σημαντική παρατήρηση είναι ότι το δίκτυο είναι περισσότερο ισορροπημένο όταν αναφερόμαστε στον φόρτο αποθήκευσης σε σχέση με τον φόρτο πρόσβασης.

Αυτό οφείλεται στο γεγονός ότι οι συνδρομές που περιλαμβάνουν ιδιοχαρακτηριστικά με περιορισμούς διαστημάτων τιμών μπορούν να παραγάγουν έναν μεγαλύτερο αριθμό αιτήσεων αποθήκευσης των *SubIDs* σε σχέση με τις αιτήσεις ανάκτησης αυτών κατά την επεξεργασία των δημοσιεύσεων. Ο αριθμός λοιπόν των *SubIDs* που αποθηκεύονται είναι μεγαλύτερος με τα *SubIDs* να είναι περισσότερο ομοιόμορφα κατανομημένα μεταξύ των κόμβων του δικτύου, σε σχέση με τις αιτήσεις ανάκτησης των *SubIDs*.

5.5.4 Επεξεργασία και Σύζευξη Δημοσιεύσεων με Συνδρομές

Πλήθος Μηνυμάτων

Πραγματοποιήσαμε επίσης μία σειρά πειραμάτων προκειμένου να μετρήσουμε τον αριθμό μηνυμάτων που απαιτούνται κατά την επεξεργασία των δημοσιεύσεων και των συνδρομών, μεταβάλλοντας την κλίση της κατανομής της δημοτικότητας των ιδιοχαρακτηριστικών και των τιμών τους.

Παρατηρήσαμε ότι καθώς η κλίση της κατανομής της δημοτικότητας των ιδιοχαρακτηριστικών αυξάνεται, λιγότερα ιδιοχαρακτηριστικά ορίζονται ανά δημοσίευση/συνδρομή και έτσι ο μέσος όρος των μηνυμάτων μειώνεται όπως μπορείτε να δείτε στο Σχήμα 5.13.

Σχήμα 5.13: Πλήθος μηνυμάτων που απαιτούνται κατά την επεξεργασία των συνδρομών και των δημοσιεύσεων.

Και εδώ, η βασική παρατήρησή μας είναι ότι η επεξεργασία των συνδρομών απαιτεί την εκτέλεση περισσότερων βημάτων (ή αποστολή μηνυμάτων), έναντι της επεξεργασίας των δημοσιεύσεων. Πιστεύουμε ότι είναι επιθυμητή μία τέτοια συμπεριφορά μιας και σε πραγματικά συστήματα δημοσιεύσεων/συνδρομών, οι δημοσιεύσεις αναμένεται ότι θα γεννιούνται με πολύ υψηλότερους ρυθμούς σε σχέση με τις συνδρομές, ενώ παράλληλα η επεξεργασία των συνδρομών θα πρέπει να γίνεται όσο το δυνατόν ταχύτερα.

Αυξήσαμε επίσης την κλίση της κατανομής της δημοτικότητας των τιμών και παρατηρήσαμε ότι ο αριθμός μηνυμάτων παραμένει σχετικά αμετάβλητος. Από την θεωρητική μελέτη προκύπτει ότι ο αριθμός των μηνυμάτων δεν επηρεάζεται από την δημοτικότητα των τιμών. Αντιθέτως, επηρεάζεται η κατανομή του φόρτου πρόσβασης και αποθήκευσης όπως άλλωστε παρατηρήσαμε σε προηγούμενη ενότητα.

Πλήθος Διακινουμένων SubIDs Πάνω στο Δίκτυο

Σε αυτή την ομάδα πειραμάτων η μετρική απόδοσης μας είναι ο συνολικός αριθμός των *SubIDs* που διακινούνται κατά την επεξεργασία κάθε δημοσίευσης.

Μεταβάλλουμε την κλίση της κατανομής της δημοτικότητας των τιμών των ιδιοχαρακτηριστικών ενώ η τιμή της παραμέτρου ϑ για τη δημοτικότητα των ιδιοχαρακτηριστικών είναι ίση με 0.8. Παρατηρήσαμε ότι καθώς αυξάνεται η τιμή του ϑ (μεταβάλλοντας την τιμή της από 0.0 έως 2.0) αυξάνεται δραματικά και των πλήθος των *SubIDs* που συλλέγονται ανά δημοσίευση.

Όταν η τιμή του ϑ είναι ίση με 2.0, ο αριθμός των *SubIDs* (που αποτελεί μια ένδειξη της κίνησης που θα πρέπει να περιμένουμε) αυξάνεται κατά 20 φορές σε σχέση με την περίπτωση όπου $\vartheta = 0.8$ (που αποτελεί

Σχήμα 5.14: Ποσοστό των ενεργών εγγραφών των *CRT* μεταβάλλοντας το β .

μία τυπική τιμή όπως έχουν δείξει σχετικές εργασίες [25]). Για την τιμή $\vartheta = 2.0$, ένα πολύ μικρό σύνολο κόμβων εμπλέκεται στην επεξεργασία των συνδρομών και των δημοσιεύσεων, αποθηκεύοντας σχεδόν όλα τα *SubIDs* και εξυπηρετώντας αντίστοιχα σχεδόν όλες τις αιτήσεις ανάκτησης αυτών. Έτσι, είναι αναμενόμενο οι λίστες με *SubIDs* που επιστρέφονται στον συντονιστή (το κόμβο όπου γεννήθηκε τη δημοσίευση και αναλαμβάνει την διαδικασία σύζευξης) να αποτελούνται από δεκάδες φορές περισσότερα *SubIDs* σε σχέση με την περίπτωση όπου η δημοτικότητα πλησιάζει την ομοιόμορφη κατανομή (ϑ πλησιάζει στο 0.0).

Αυξάνοντας την κλίση της κατανομής της δημοτικότητας των ιδιοχαρακτηριστικών (μεταβάλλοντας το ϑ από 0.0 έως 2.0) παρατηρήσαμε ότι το πλήθος των *SubIDs* μειώνεται κατά 2.5 φορές, επειδή λιγότερα ιδιοχαρακτηριστικά ορίζονται σε κάθε συνδρομή/δημοσίευση όταν το $\vartheta = 2.0$ σε σχέση με την περίπτωση όπου το $\vartheta = 0.8$.

5.5.5 Αυτόνομη Οργάνωση και Αντιμετώπιση Βλαβών

Εκτελέσαμε μία σειρά πειραμάτων για να εξακριβώσουμε την ορθότητα και να μετρήσουμε την απόδοση των προτεινόμενων αλγορίθμων που εγγυώνται τη αυτόνομη οργάνωση και την ανάνηψη υπό την παρουσία βλαβών στην υποδομή *PastryStrings*. Σε αυτή την ενότητα παρουσιάζουμε κυρίως το κόστος διατήρησης της πρόσθετης κατάστασης δρομολόγησης (πίνακες *CRT* και δείκτες *FP*) και την διαθεσιμότητας των αποθηκευμένων συνδρομών κατά την παρουσία βλαβών στο δίκτυο.

Μετρήσαμε επίσης την συμπεριφορά της υποδομής *PastryStrings* κατά την άφιξη νέων κόμβων και την αναχώρηση κόμβων με τη θέληση τους και καταλήξαμε στο συμπέρασμα ότι οι αλγόριθμοι μας λειτουργούν

Σχήμα 5.15: Ποσοστό των ενεργών εγγραφών των *CRT* μεταβάλλοντας το μέγιστο μήκος των συμβολοσειρών.

όπως αναμενόταν. Οπότε, θα επικεντρωθούμε σε αυτή την ενότητα στην απόδοση των αλγορίθμων μας κατά την ύπαρξη βλαβών στο δίκτυο.

Κόστος Διατήρησης της Σχετικής με τη Δρομολόγηση Πληροφορίας

Όπως αναφέρθηκε νωρίτερα στην ενότητα 5.3 οι πίνακες δρομολόγησης *CRT* αναμένεται ότι θα περιέχουν ένα μικρό ποσοστό ενεργών δεικτών. Μετρήσαμε λοιπόν το ποσοστό των καταχωρήσεων των *CRT* που περιέχουν δείκτες προς άλλους *Tnodes*, μεταβάλλοντας το πλήθος των διαθέσιμων χαρακτήρων του αλφαβήτου μας (που στην ουσία ορίζει τον βαθμό g των δέντρων και κατ' επέκταση και το μέγεθος των *CRT*) και το μέγιστο μήκος των συμβολοσειρών που ορίζονται στις δημοσιεύσεις και στις συνδρομές. Σε αυτό το σύνολο πειραμάτων ο συνολικός αριθμός των πινάκων *CRT* που υπάρχουν στο δίκτυο έφτασε τις 40,000 ενώ ο συνολικός αριθμός των *Tnodes* έφτασε τις 70,000.

Στο Σχήμα 5.14 μεταβάλλουμε το β και μετράμε το μέσο ποσοστό των ενεργών καταχωρήσεων στον πίνακα *CRT*, για δύο διαφορετικά μεγέθη συμβολοσειρών. Όπως μπορείτε να παρατηρήσετε το ποσοστό πέφτει κάτω από το 10% καθώς το β αυξάνεται. Στο Σχήμα 5.15 μεταβάλλουμε το μέγιστο μήκος των συμβολοσειρών για δύο διαφορετικές τιμές του β . Μπορείτε να παρατηρήσετε ότι το ποσοστό των ενεργών καταχωρήσεων μειώνεται στο 20% όταν αυξάνεται το μήκος των συμβολοσειρών (και το $\beta = 5$) ενώ για το μεγαλύτερο β εκείνο το ποσοστό παραμένει κοντά στο 10%.

Σε πραγματικά συστήματα το β μπορεί να φτάσει τη τιμή 36 και το μήκος των συμβολοσειρών, τους

Σχήμα 5.16: Πλήθος χαμένων *SubIDs* συναρτήσει του ποσοστού των κόμβων που αποσυνδέονται από το δίκτυο λόγω βλάβης.

10 χαρακτήρες (αυτές θα μπορούσε να είναι χαρακτηριστικές τιμές για μια εφαρμογή ενημέρωσης που εκμεταλλεύεται την υποδομή *PastryStrings*, και σχετίζεται με το χρηματιστήριο και τις τιμές των μετοχών). Έτσι, θα περιμένουμε να έχουμε πίνακες *CRT* με $\beta^2 = 1296$ καταχωρήσεις. Εντούτοις όπως μπορείτε να δείτε στο Σχήμα 5.14 μόνο το 4% των εγγράφων τους (≈ 50) θα αποτελούνται από ενεργούς δείκτες προς κόμβους *Tnodes*. Αυτή η παρατήρηση δείχνει ότι το κόστος διατήρησης των πινάκων *CRT* θα είναι σχετικά μικρό.

Επιπλέον, εκτελέσαμε ορισμένα πειράματα μεταβάλλοντας την κλίση της κατανομής της δημοτικότητας των τιμών, και παρατηρήσαμε ότι επηρεάζει ελάχιστα το ποσοστό των έγκυρων εγγράφων τους. Έτσι, πιστεύουμε ότι κόστος διατήρησης των πινάκων δρομολόγησης *CRT* είναι μικρό και ανεξάρτητο από τα χαρακτηριστικά των τιμών των συνδρομών και των δημοσιεύσεων

Υπαρξη Βλαβών και Διαθεσιμότητα Συνδρομών

Σε αυτή την ενότητα μετρήσαμε τη διαθεσιμότητα των αποθηκευμένων αναγνωριστικών των συνδρομών συναρτήσει του ποσοστού αποτυχίας των κόμβων και του βαθμού αντιγραφής (δηλ. του πλήθους των αντιγράφων που αντιστοιχούν σε κάθε δέντρο). Ένας κόμβος θεωρείται ότι έχει υποστεί βλάβη και έτσι αποτυγχάνει να επικοινωνήσει με τους υπόλοιπους με πιθανότητα ίση με το ποσοστό αποτυχίας.

Εφαρμόζουμε και συγκρίνουμε δύο διαφορετικά σενάρια αποτυχίας των κόμβων και αποκατάστασης των δενδρικών δομών:

Σχήμα 5.17: Επιρροή του ποσοστού αποτυχίας στην ακεραιότητα της δενδρικής δομής.

1. *βαθμιαίες αποτυχίες* (gradual failures), όπου η διαδικασία αναδημιουργίας των δέντρων ξεκινάει αμέσως μετά από την αποτυχία ενός συγκεκριμένου *Tnode* ⁷.
2. *μαζικές αποτυχίες* (instant failures), όπου οι κόμβοι αποτυγχάνουν ταυτόχρονα και αντιστοιχεί στο σενάριο όπου ολόκληρες περιοχές του δικτύου αποκόπτονται από το υπόλοιπο δίκτυο.

Τα δύο κύρια συμπεράσματα που προκύπτουν από το Σχήμα 5.16 είναι ότι σε έναν ιδανικό κόσμο όπου ο χειρισμός των βλαβών των κόμβων ενεργοποιείται αμέσως μόλις αυτή συμβεί, οι χαμένες συνδρομές (*SubIDs*) ανακτώνται από τα αντίγραφα δέντρα και έτσι τελικά σχεδόν κανένα από το αποθηκευμένο *SubIDs* δεν χάνεται. Στο Σχήμα 5.16 αυτό φαίνεται με την καμπύλη *gradual failures* που παραμένει κοντά στο 0% ανεξάρτητα από το ποσοστό αποτυχίας.

Η επόμενη ενδιαφέρουσα παρατήρηση έχει να κάνει με το γεγονός ότι η ύπαρξη αντιγράφων ανά δέντρο πράγματι βοηθάει την διαδικασία ανάνηψης. Όπως μπορείτε να δείτε στο Σχήμα 5.16 τις καμπύλες που αντιστοιχούν στην ύπαρξη 1, 2, 4, και 9 αντιγραφα ανά δέντρο, όσο αυξάνεται το πλήθος των αντιγράφων, τόσο μειώνεται το ποσοστό των χαμένων *SubIDs*.

Επιρροή των Βλαβών στην Διατήρηση της Δενδρικής Δομής

Στη συνέχεια εκτελέσαμε μία σειρά πειραμάτων ώστε να μετρήσουμε την ακεραιότητα των δενδρικών δομών υπό την παρουσία βλαβών και με την ενεργοποίηση του μηχανισμού αναδιοργάνωσης τους.

⁷ Προφανώς αυτό είναι ένα ιδανικό σενάριο δεδομένου ότι σε πραγματικά δίκτυα θα χρειαστεί να παρέλθει κάποιο χρονικό διάστημα ώστε να ανιχνεύσουν οι γειτονικοί κόμβοι την βλάβη.

Στο Σχήμα 5.17, στην περίπτωση που δεν αναδιοργανώνονται τα δέντρα (καμπύλες without reconstruction), ο αριθμός των ορφανών *Tnodes* αυξάνεται καθώς αυξάνουμε το ποσοστό αποτυχίας και φθάνει σε ποσοστό 80%. Όταν εφαρμόζουμε τους αλγορίθμους αναδημιουργίας των δέντρων μας, μπορείτε να δείτε ότι κανένα ορφανό *Tnode* δεν υπάρχει ανεξάρτητα από το ποσοστό αποτυχίας. Οι προτεινόμενοι αλγόριθμοι, θα επιτρέψουν στους *Tnodes* να ανακαλύψουν πιθανά ορφανά παιδιά και να επανασυνδεθούν με αυτά

Στο ίδιο σχήμα, μπορείτε να δείτε το ποσοστό των ορφανών δέντρων που προκύπτουν χωρίς την εφαρμογή του μηχανισμού αναδιοργάνωσης. Το γεγονός ότι το ποσοστό των ορφανών δέντρων είναι πολύ υψηλό, ακόμη και για μικρά ποσοστά αποτυχίας, υποδηλώνει ότι τα ορφανά δέντρα αποτελούνται από έναν σχετικά μικρό αριθμό κόμβων.

6 Σύζευξη Δημοσιεύσεων και Συνδρομών με Τελεστές Υποσυμβολοσειράς

Σε αυτό το Κεφάλαιο θα ασχοληθούμε αποκλειστικά με την επεξεργασία συνδρομών και δημοσιεύσεων με ιδιοχαρακτηριστικά τύπου συμβολοσειράς.

Συγκεκριμένα, θα προτείνουμε σχετικούς αλγορίθμους για την επεξεργασία συνδρομών με περιορισμό τύπου υποσυμβολοσειράς (substring - π.χ. `*101*`). Οι λύσεις που προτείνουμε είναι δυνατό να εφαρμοστούν σε ένα σύστημα δημοσιεύσεων/συνδρομών που λειτουργεί είτε πάνω από οποιοδήποτε DHT δίκτυο ομοτίμων εταίρων (στα πρότυπα της υποδομής *Pyracanthus*), είτε πάνω στην υποδομή *PastryStrings* που εκμεταλλεύεται συγκεκριμένου τύπου δίκτυα, και συγκεκριμένα αυτά που εκτελούν δρομολόγηση κατά πρόθεμα (όπως τα δίκτυα *Pastry* [85], *Tapestry* [103] και *Bamboo* [84]).

Λαμβάνοντας υπόψη αυτά τα περιβάλλοντα, θα προτείνουμε και θα μελετήσουμε τρεις διαφορετικές ομάδες αλγορίθμων (για την επεξεργασία των συνδρομών και των δημοσιεύσεων). Η πρώτη (η απλούστερη) βασίζεται στην παραγωγή και τον έλεγχο όλων των δυνατών υποσυμβολοσειρών μιας δεδομένης συμβολοσειράς και για αυτό από εδώ και στο εξής θα αποκαλούμε *complete*. Η δεύτερη είναι βασισμένη στην προσαρμογή της τεχνικής διάσπασης μιας συμβολοσειράς στα *ngrams* που την αποτελούν ([54]).

Η μελέτη αυτών των τεχνικών έδειξε ότι η καθεμία δείχνει να υπερτερεί εξετάζοντας διαφορετικές μετρικές. Έτσι, προτείνουμε την προσέγγιση *ngramsBloom* που εκμεταλλεύεται τα φίλτρα Bloom [22] και δείχνει να αποδίδει σχετικά καλά για όλες τις μετρικές απόδοσης που εξετάζουμε (απαιτήσεις σε εύρος ζώνης, πλήθος μηνυμάτων, πλήθος μεταφερόμενων *SubIDs*, εξισορρόπησης του φόρτου κ.α.).

Για λόγους παρουσίασης, θα υποθέσουμε ότι οι δημοσιεύσεις αποτελούνται από ένα ιδιοχαρακτηριστικό τύπου συμβολοσειράς και οι συνδρομές αποτελούνται επίσης από ένα ιδιοχαρακτηριστικό με περιορισμό υποσυμβολοσειράς (π.χ. `*10*`) και συμβολοσειρές δυαδικού αλφάβητου. Η γενίκευση του μοντέλου που παρουσιάζουμε εδώ, ώστε να συμπεριλαμβάνει περισσότερα του ενός ιδιοχαρακτηριστικά καθώς και πιο σύνθετους τελεστές όπως πρόθεμα, επίθεμα για συμβολοσειρές και διαστήματα τιμών για αριθμητικές τιμές, έχει ήδη παρουσιαστεί στα κεφάλαια 4 και 5.

6.1 Ορισμός του Προβλήματος και των Σχετικών Αλγορίθμων

Αρχικά, παρουσιάζουμε τις ανεξάρτητες από το DHT προσεγγίσεις ($*$ – IND) και έπειτα τις προσεγγίσεις που βασίζονται στην υποδομή *PastryStrings* ($*$ – PS). Για τον συμβολισμό των συμβολοσειρών θα χρησιμοποιήσουμε την έννοια του μονοδιάστατου πίνακα, όπου σε κάθε θέση του (ξεκινώντας από την θέση 1) βρίσκεται και ένας από τους χαρακτήρες της συμβολοσειράς. Θα χρησιμοποιήσουμε τον εξής συμβολισμό για την συμβολοσειρά s : $s[1...l_s]$, όπου l_s είναι το πλήθος των χαρακτήρων που την αποτελούν. Έτσι, για παράδειγμα, η συμβολοσειρά '1011' συμβολίζεται με $s[1...4] = 1011$, ενώ ο χαρακτήρας σε συγκεκριμένη θέση i συμβολίζεται με $s[i]$ (π.χ. $s[2] = 0$).

Το πρόβλημα μπορεί να οριστεί ως εξής: δεδομένου ότι το μέγιστο μήκος των συμβολοσειρών είναι l , και ότι μια συνδρομή ορίζει ένα περιορισμό υποσυμβολοσειράς με τιμή $s[1...l_s]$ όπου $l_s \leq l$, θα πρέπει να εντοπίσουμε και να συλλέξουμε το αναγνωριστικό της συνδρομής όταν καταφθάσει μια δημοσίευση, εάν η συμβολοσειρά $s[1...l_s]$ αποτελεί τμήμα της συμβολοσειράς της δημοσίευσης $e[1...l_e]$ (με $l_s \leq l_e \leq l$). Με άλλα λόγια, για όλα τα i με $0 \leq i \leq l_s$, θα πρέπει να υπάρχει σταθερά k ώστε $s[i] = e[k + i]$.

6.1.1 Η Προσέγγιση complete-IND

Η βασική ιδέα στην complete-IND προσέγγισης είναι η αποθήκευση του *SubID* της συνδρομής στον κόμβο που είναι αρμόδιος (με βάση τη συνάρτηση κατακερματισμού) για την συμβολοσειρά της συνδρομής. Κατόπιν, όταν φθάνει στο σύστημα μας μία δημοσίευση, παράγουμε όλες τις πιθανές υποσυμβολοσειρές της τιμής της και επικοινωνούμε με όλους τους αρμόδιους κόμβους του DHT ώστε να συλλέξουμε τα αποθηκευμένα *SubIDs*.

Υποθέστε ότι δημιουργείται μια συνδρομή με περιορισμό υποσυμβολοσειράς με τιμή $s[1...l_s]$. Χρησιμοποιώντας τη συνάρτηση κατακερματισμού του DHT και την συμβολοσειρά s σαν όρισμα, εντοπίζουμε τον κόμβο που θα αποθηκευθεί το αναγνωριστικό της (*SubID*). Σε κάθε κόμβο, μαζί με τον αποθηκευμένο κατάλογο των *SubIDs*, διατηρούμε και μια λίστα V στην οποία αποθηκεύουμε την συμβολοσειρά της συνδρομής που αντιστοιχεί σε κάθε αποθηκευμένο *SubID*. Όταν καταφθάσει μία δημοσίευση με τιμή $e[1...l_e]$ παράγουμε όλες τις δυνατές υποσυμβολοσειρές της τιμής e , και με την βοήθεια της συνάρτησης κατακερματισμού εντοπίζουμε τους κόμβους που ενδεχομένως να έχουν αποθηκευμένο κάποιο *SubID* που η τιμή του στη λίστα V του αντίστοιχου κόμβου, αποτελεί τμήμα της συμβολοσειράς e .

6.1.2 Η Προσέγγιση με Χρήση των *ngrams* πάνω σε Οποιοδήποτε DHT:

ngrams-IND

Η βασική ιδέα πίσω από την τεχνική *ngrams* είναι η διάσπαση κάθε συμβολοσειράς (είτε αυτή ορίζεται σε συνδρομές και είτε σε δημοσιεύσεις) σε υποσυμβολοσειρές μήκους n χαρακτήρων. Στη συνέχεια αυτές οι υποσυμβολοσειρές μπορούν να χρησιμοποιηθούν για την αποθήκευση των συνδρομών και τον εντοπισμό τους κατά την άφιξη των δημοσιεύσεων. Δεδομένης μιας συμβολοσειράς $v[1...l_v]$ και *ngrams* μήκους n χαρακτήρων, το i -οστό *ngram* της συμβολοσειράς v , δίνεται γενικά από την σχέση :

$$v_i = v[i...(n + i - 1)], i \leq l_v - n + 1$$

Ειδικότερα, εάν τα *ngrams* αποτελούνται από 2 χαρακτήρες ($n = 2$ οπότε αποκαλούνται *bigrams*), το i -οστό *bigram* της συμβολοσειράς $v[1...l_v]$, δίνεται γενικά από την σχέση :

$$v_i = v[i...(i + 1)], i \leq l_v - 1$$

Παραδείγματος χάριν τα 3 *bigrams* της συμβολοσειράς $v[1...4]=1011$ είναι: $v_1=10$, $v_2=01$, και $v_3=11$.

Όταν μια συνδρομή φθάνει στο σύστημα με τιμή $s[1...l_s]$, τεμαχίζουμε αρχικά το s σε $(l_s - n + 1)$ *ngrams*. Για κάθε *ngram* κατασκευάζουμε το extended *SubID* (*xSubID*) που περιέχει: (i) το *SubID* όπως το έχουμε ορίσει μέχρι τώρα, (ii) τη σειρά εμφάνισης του *ngram* στη συμβολοσειρά s , (iii) το πλήθος των *ngrams* που προκύπτουν από την συμβολοσειρά, και (iv) το ίδιο το *ngram*, όπως μπορείτε να δείτε παρακάτω :

<i>SubID</i>	order	number of ngrams	ngram value
--------------	-------	------------------	-------------

Extended SubID - xSubID

Στη συνέχεια, αποθηκεύουμε το *xSubID* στους κόμβους με αναγνωριστικά που προκύπτουν από την εφαρμογή της συνάρτησης κατακερματισμού χρησιμοποιώντας το κάθε *ngram* σαν όρισμα. Τα επιπλέον πεδία στο *xSubID* είναι απαραίτητα για την αποφυγή των *false positives*¹, και για την αναδόμηση της τιμής (από την οποία προκύπτουν τα *ngrams*) ώστε να είμαστε σε θέση να εντοπίσουμε τις συνδρομές που συζευγνούν με μια συγκεκριμένη δημοσίευση, στηριζόμενοι στα *xSubIDs* που έχουμε συλλέξει.

Όταν μία δημοσίευση καταφθάσει με τιμή $e[1...l_e]$, τεμαχίζουμε την e σε $(l_e - n + 1)$ *ngrams* και επικοινωνούμε με εκείνους τους κόμβους των οποίων τα αναγνωριστικά προκύπτουν από την εφαρμογή της συνάρτησης κατακερματισμού με τα *ngrams* σαν όρισμα, ώστε συλλέξουμε τα αποθηκευμένα *xSubIDs*.

¹ Παραδείγματος χάριν, οι συμβολοσειρές 101 και 010 έχουν το ίδιο σύνολο *ngrams* τα 10 και 01, εντούτοις είναι διαφορετικές.

Οι λίστες των $xSubIDs$ που έχουμε πλέον στη διάθεση μας, ομαδοποιούνται με βάση το $SubID$ (το πρώτο πεδίο του $xSubID$) και μετράμε πόσες φορές εμφανίζεται το κάθε $xSubID$. Θα πρέπει να σημειωθεί, ότι κάθε εμφάνιση ενός $xSubID$ σε αυτές τις λίστες, αντιστοιχεί και σε σύζευξη ενός από τα $ngrams$ της τιμής της συνδρομής με ένα από τα $ngrams$ της τιμής της δημοσίευσης.

Καταλήγουμε σε σύζευξη μεταξύ δημοσίευσης και συνδρομών εάν:

1. ο αριθμός των εμφανίσεων του $xSubID$ είναι ίσος με το τρίτο πεδίο του, που σημαίνει ότι όλα τα $ngrams$ της τιμής της συνδρομής έχουν συλλεχθεί.
2. η αναδημιουργία της τιμής της συνδρομής με βάση το δεύτερο πεδίο του $xSubID$ (που δείχνει την σειρά εμφάνισης του $ngram$) και του τρίτου (το ίδιο το $ngram$), είναι όντως τμήμα της τιμής της δημοσίευσης.

Στη συνέχεια μπορείτε να δείτε ένα παράδειγμα αυτής της διαδικασίας.

Παράδειγμα 10 Επεξεργασία συνδρομής με περιορισμό υποσυμβολοσειράς

Ας υποθέσουμε ότι ορίζονται τρεις συνδρομές με τελειστή υποσυμβολοσειράς και τιμές $*101*$, $*1001*$, και $*010*$. Δεδομένου ότι τα $ngrams$ σε αυτό το παράδειγμα αποτελούνται από 2 χαρακτήρες ($bigrams$, $n = 2$) τεμαχίζουμε τις συμβολοσειρές σε $bigrams$ και αποθηκεύουμε τα ακόλουθα $xSubIDs$ στους αντίστοιχους κόμβους.

$Sub1 : *101*$	$Sub2 : *1001*$	$Sub3 : *010*$
$SubID_1 1 2 10$	$SubID_2 1 3 10$	$SubID_3 1 2 01$
$SubID_1 2 2 01$	$SubID_2 2 3 00$	$SubID_3 2 2 10$
-	$SubID_2 3 3 01$	-

Στη συνέχεια, μία δημοσίευση καταφθάνει στο σύστημα με τιμή 11011. Κατ' αρχάς, η τιμή θα τεμαχιστεί σε 4 $bigrams$: 11, 10, 01, και 11, και οι κατάλληλοι κόμβοι θα ερωτηθούν για τα αποθηκευμένα $xSubIDs$ χρησιμοποιώντας τη συνάρτηση κατακερματισμού και τα $bigrams$ σαν όρισμα. Για κάθε $bigram$ συλλέγουμε τις ακόλουθες λίστες:

11	10	01	11
-	$SubID_1 1 2 10$	$SubID_1 2 2 01$	-
-	$SubID_2 1 3 10$	$SubID_2 3 3 01$	-
-	$SubID_3 2 2 10$	$SubID_3 1 2 01$	-

Μετά την ομαδοποίηση των $xSubIDs$ με βάση το $SubID$ που περικλείουν, προκύπτει ο ακόλουθος πίνακας:

$SubID_1$	$SubID_2$	$SubID_3$
$SubID_1 1 2 10$	$SubID_2 1 3 10$	$SubID_3 1 2 01$
$SubID_1 2 2 01$	$SubID_2 3 3 01$	$SubID_3 2 2 10$
Group Size 2	Group Size 2	Group Size 2

Η συνδρομή $Sub1$ συζευγνύει με τη δημοσίευση επειδή το μέγεθος του συνόλου των $SubID_1$ είναι ίσο με το πλήθος (2) των *bigrams* της τιμής της συνδρομής και η αναδημιουργία της συμβολοσειράς (101) με βάση τα πεδία των $xSubIDs$ που έχουν συγκεντρωθεί είναι όντως τμήμα της συμβολοσειράς της δημοσίευσης (11011). Η συνδρομή $Sub2$ δεν συζευγνύει με τη δημοσίευση επειδή το μέγεθος του συνόλου (2) είναι διαφορετικό από τον αριθμό του *bigrams* της τιμής της (το τρίτο πεδίο στο $xSubID$ είναι ίσο με 3). Τέλος, ούτε η συνδρομή $Sub3$ συζευγνύει με τη δημοσίευση, επειδή αν και το μέγεθος του συνόλου είναι ίσο με τον αριθμό του *bigrams* η αναδημιουργία της συμβολοσειράς (010) δεν αποτελεί τμήμα της τιμής της δημοσίευσης.

6.1.3 Η Προσέγγιση complete-PS

Είναι δυνατό να εκμεταλλευτούμε την υποδομή *PastryStrings* ώστε να βελτιώσει περαιτέρω την απόδοση της επεξεργασίας των δημοσιεύσεων και των συνδρομών. Τα $SubIDs$ των συνδρομών αποθηκεύονται στο δέντρο που είναι υπεύθυνο για την διαχείριση των συμβολοσειρών που ο πρώτος χαρακτήρας τους είναι ίδιος με τον πρώτο χαρακτήρα της τιμής της συνδρομής. Οι δημοσιεύσεις υποβάλλονται σε επεξεργασία με αντίστοιχο τρόπο, με βάση την συμβολοσειρά που ορίζεται σε αυτές.

Όταν μια συνδρομή ορίζει την τιμή $s[1...l_s]$, αρχικά εντοπίζουμε το κατάλληλο δέντρο, με βάση αυτή τη τιμή που είναι το $T_{s[1]}$. Στη συνέχεια, ακολουθούμε το μονοπάτι μέχρι τον κόμβο $Tnode_{s[1...l_s]}$ ελέγχοντας ένα χαρακτήρα κάθε φορά, όπου τελικά αποθηκεύουμε τον $SubID$.

Κατά την άφιξη μιας δημοσίευσης με τιμή $e[1...l_e]$, αφαιρούμε έναν χαρακτήρα τη φορά από την αρχή της συμβολοσειράς και μεταχειριζόμαστε το επίθεμα που παράγεται ως ξεχωριστή δημοσίευση. Για κάθε τέτοιο επίθεμα, $e_i[i...l_e]$ εντοπίζουμε το κατάλληλο δέντρο $T_{e[i]}$ και μετά ακολουθούμε το μονοπάτι μέχρι τον κόμβο $Tnode_{e_i[i...l_e]}$. Από κάθε κόμβο στα μονοπάτια αυτά, συλλέγουμε τα αποθηκευμένα $SubIDs$. Με αυτό τον τρόπο ελέγχουμε όλες τις δυνατές υποσυμβολοσειρές της τιμής της δημοσίευσης για πιθανή σύζευξη με τις αποθηκευμένες συνδρομές.

Θα παρατηρήσατε ότι με την τεχνική που μόλις περιγράψαμε για την επεξεργασία των δημοσιεύσεων, είναι πιθανό να χρειαστεί να επισκεφθούμε το ίδιο δέντρο για την επεξεργασία διαφορετικών υποσυμβολοσειρών της τιμής τους, που όμως ξεκινάνε με τον ίδιο χαρακτήρα. Παραδείγματος χάριν, εάν η τιμή της δημοσίευσης είναι $e[1...l_e]$ και οι χαρακτήρες στις θέσεις i και j είναι όμοιοι, τότε θα πρέπει να επισκεφθού-

Σχήμα 6.1: Ανάπτυξη των δέντρων κατά την επεξεργασία των συνδρομών και σύζευξη με τη δημοσίευση στη προσέγγιση complete-PS.

με το ίδιο δέντρο για την εξέταση των υποσυμβολοσειρών $e_i[i...l_e]$ και $e_j[j...l_e]$. Για να αποφύγουμε λοιπόν πολλαπλές επισκέψεις στο ίδιο δέντρο, μπορούμε να ομαδοποιήσουμε τις υποσυμβολοσειρές που προκύπτουν με βάση των πρώτο χαρακτήρα τους και να επεξεργαστούμε παράλληλα αυτές που ανήκουν στην ίδια ομάδα (και πρόκειται να σταλούν για επεξεργασία στο ίδιο δέντρο). Στο παράδειγμα που ακολουθεί παρουσιάζεται η τεχνική που μόλις περιγράψαμε.

Παράδειγμα 11 Επεξεργασία συνδρομής και δημοσίευσης με την προσέγγιση complete-PS.

Υποθέστε ότι δύο συνδρομές καταφθάνουν στο σύστημα με τιμές *101* και *010*. Η πρώτη συνδρομή θα αποθηκευτεί στο δέντρο T_1 . Αρχικά, δεν έχει σχηματιστεί κανένα δέντρο, οπότε κατασκευάζεται σταδιακά καθώς επεξεργαζόμαστε την πρώτη συνδρομή (Σχήμα 6.1). Κατά συνέπεια κατασκευάζουμε τη ρίζα του δέντρου T_1 , $Tnode_1$, μαζί με τον πίνακα δρομολόγησης της, και στη συνέχεια τους κόμβους $Tnode_{10}$ και $Tnode_{101}$ που τελικά αποθηκεύεται το $SubID_1$. Η δεύτερη συνδρομή υποβάλλεται σε επεξεργασία ομοίως, δημιουργώντας τους κόμβους $Tnode_0$, $Tnode_{01}$, και $Tnode_{010}$ όπου το $SubID_2$ αποθηκεύεται.

Όταν δημιουργηθεί μία δημοσίευση με τιμή 1101, αρχικά ομαδοποιούμε τα επιθέματα της συμβολοσειράς με βάση τον πρώτο χαρακτήρα τους. Με την αφαίρεση ενός χαρακτήρα κάθε φορά από την αρχή της συμβολοσειράς 1101 προκύπτουν τα επιθέματα 1101, 101, και 1 που θα αποσταλούν στο δέντρο T_1 και

Σχήμα 6.2: Σύζευξη δημοσίευσης με συνδρομές στην προσέγγιση ngrams-PS.

το 01 που θα αποσταλεί στο δέντρο T_0 . Για την πρώτη ομάδα εντοπίζουμε το δέντρο T_1 και ακολουθούμε μονοπάτι (εάν υπάρχει) προς τους $Tnode_{1101}$, $Tnode_{101}$, και $Tnode_{1}$, όπου εντοπίζουμε το $SubID_1$ στον κόμβο $Tnode_{101}$. Παράλληλα με αυτή τη διαδικασία εντοπίζουμε το δέντρο T_0 , και ακολουθούμε το μονοπάτι προς τον $Tnode_{01}$ όπου δεν βρίσκουμε κανένα αποθηκευμένο $SubID$. Εφόσον έχουμε συλλέξει μόνο το $SubID_1$ καταλήγουμε σε σύζευξη της δημοσίευσης με την πρώτης συνδρομή (με αναγνωριστικό $SubID_1$). $\diamond$

6.1.4 Η Προσέγγιση με Χρήση των ngrams πάνω στην Υποδομή PastryStrings: ngrams-PS

Σε αυτήν την προσέγγιση επιλέγουμε να τεμαχίσουμε τις συμβολοσειρές σε ngrams και να τα επεξεργαστούμε πάνω από την υποδομή PastryStrings. Η επεξεργασία κάθε συνδρομής περιλαμβάνει τον τεμαχισμό της συμβολοσειράς της $s[1...l_s]$ σε $(l_s - n + 1)$ ngrams και για καθέναν από αυτά, τον εντοπισμό του κατάλληλου δέντρου (βασισμένο στον πρώτο χαρακτήρα του ngram) και αποθήκευση του $xSubID$ στον κατάλληλο $Tnode$ μέσα στο δέντρο.

Η επεξεργασία των δημοσιεύσεων επιτυγχάνεται με την παρόμοιο τρόπο (όσο αναφορά τον εντοπισμό των κατάλληλων $Tnodes$). Όταν εντοπίσουμε σε αυτούς αποθηκευμένα $xSubIDs$ τα συλλέγουμε και τα επεξεργαζόμαστε περαιτέρω (όπως περιγράφηκε στην ενότητα 6.1.2) προκειμένου να καταλήξουμε σε πιθανή σύζευξη ή όχι.

Παράδειγμα 12 Επεξεργασία συνδρομής και δημοσίευσης με την προσέγγιση ngrams-PS.

Υποθέστε ότι έχουμε μια συνδρομή με περιορισμό υποσυμβολοσειράς και τιμή *101*. Αρχικά τεμαχίζουμε

την συμβολοσειρά στα *bigrams* 10 και 01 και τα αποθηκεύουμε στα κατάλληλα δέντρα (Σχήμα 6.2). Κατασκευάζουμε τα δέντρα T_1 και T_0 (εφόσον αυτά δεν υπάρχουν) και αποθηκεύουμε στον κόμβο $Tnode_{10}$ το $xSubID : SubID_1|1|2|10$ και στον $Tnode_{01}$ το $xSubID : SubID_1|2|2|01$.

Όταν καταφθάσει μία δημοσίευση με τιμή 1101, την τεμαχίζουμε στα *bigrams* 11, 10, και 01 και τα ομαδοποιούμε με βάση τον πρώτο χαρακτήρα τους, ώστε να σταλεί κάθε ομάδα στο κατάλληλο δέντρο της υποδομής *PastryStrings* και να εντοπιστούν τα αποθηκευμένα *SubIDs*. Συγκεκριμένα, τα 11 και 10 στέλνονται στους κόμβους $Tnode_{11}$ και $Tnode_{10}$ όπου είναι αποθηκευμένο το $SubID_1$. Παράλληλα, εντοπίζουμε τον $Tnode_{01}$ απ' όπου συλλέγουμε το $SubID_1$.

Στο τέλος αυτής της διαδικασίας, δύο *xSubIDs* έχουν συλλεχθεί: $SubID_1|1|2|10$ και $SubID_1|2|2|01$. Καταλήγουμε σε σύζευξη, μιας και έχουν συλλεχθεί *xSubIDs* και για τα δύο *bigrams* της συνδρομής ενώ παράλληλα η αναδημιουργημένη τιμή της συνδρομής αποτελεί τμήμα της τιμής της δημοσίευσης.

6.2 Η Επέκταση των *ngrams* Προσεγγίσεων με την Χρήση *Bloom* Φίλτρων

Το σημαντικότερο μειονέκτημα των *ngrams* προσεγγίσεων (*ngrams*-IND και *ngrams*-PS) είναι η αυξανόμενη κίνηση στο δίκτυο που παράγεται κατά τη διάρκεια της φάσης συλλογής των *SubIDs*. Συγκεκριμένα, όταν καταφθάνει μια δημοσίευση που συζευγνύει με μία συνδρομή με τιμή $s[1...l_s]$, θα πρέπει να μεταφερθούν $l_s - n - 1$ *xSubIDs* της συνδρομής (ένα για κάθε *ngram* της τιμής της).

Για να μειώσουμε την κυκλοφορία στο δίκτυο, προτείνουμε μία διαδικασία τριών φάσεων που εκμεταλλεύεται τα *Bloom* φίλτρα [22]. Για την εφαρμογή αυτής της διαδικασίας, θα πρέπει να τροποποιήσουμε την *ngrams* προσέγγιση ώστε να αποθηκεύουμε σε κάθε κόμβο μαζί με το *xSubID* και την συμβολοσειρά της συνδρομής στη λίστα V . Αυτή η προσθήκη θα μας επιτρέψει να ανακαλύψουμε ποια συνδρομή συζευγνύει με την τιμή της δημοσίευσης χωρίς να πρέπει να συλλέξει όλα τα αποθηκευμένα *SubIDs* που αντιστοιχούν σε κάθε *ngram* της συνδρομής.

Τα *Bloom* φίλτρα είναι μία σχετικά απλή δομή δεδομένων που μπορεί να χρησιμοποιηθεί για να εξετάσουμε εάν ένα αντικείμενο e δεν είναι μέλος ενός καθορισμένου συνόλου αντικειμένων S . Ένα φίλτρο *Bloom* αποτελείται από ένα πίνακα με $m - bit$ και k συναρτήσεις κατακερματισμού που ανάλογα με την είσοδο επιστρέφουν μία από τις m θέσεις του πίνακα. Κάθε θέση αυτού του πίνακα είναι αρχικοποιημένη στη τιμή 0. Αρχικά εισάγουμε το σύνολο S στο *Bloom* φίλτρο με το να χρησιμοποιούμε τις συναρτήσεις κατακερματισμού με κάθε αντικείμενο που ανήκει στο σύνολο S ως όρισμα, και να μεταβάλλουμε σε 1 την τιμή της θέσης στον πίνακα που μας υποδεικνύει η κάθε συνάρτηση κατακερματισμού. Για να εξακριβώσουμε

εάν ένα αντικείμενο e δεν αποτελεί μέλος του συνόλου S , χρησιμοποιούμε τις ίδιες συναρτήσεις κατακερματισμού, ελέγχοντας αυτή τη φορά το αντικείμενο e . Εάν τουλάχιστον ένα από τα $bits$ στις αντίστοιχες θέσεις είναι 0, καταλήγουμε στο ότι το e δεν αποτελεί μέλος του συνόλου S .

Εντούτοις μπορούν να εμφανιστούν false positives με πιθανότητα $(1 - e^{-\frac{ki}{m}})^k$ όπου i είναι ο αριθμός των αντικειμένων που έχουμε εισάγει στο φίλτρο. False positives μπορούν να προκύψουν όταν για παράδειγμα οι θέσεις που εξετάζουμε για ένα αντικείμενο e έχουν την τιμή 1 εξαιτίας της εισαγωγής άλλων αντικειμένων και όχι του e . Είναι προφανές ότι η πιθανότητα μειώνεται όσο αυξάνεται το μέγεθος του φίλτρου (m) ενώ αυξάνεται όσο αυξάνεται και αριθμός των εισαγωγών i .

Η βασική ιδέα πίσω από την προσέγγιση που παρουσιάζουμε εδώ είναι να χρησιμοποιήσουμε τα φίλτρα Bloom σε κάθε κόμβο που θα προσπελαστεί κατά την επεξεργασία μιας δημοσίευσης και στη συνέχεια να τα συλλέξουμε, να τα αναλύσουμε και να τα χρησιμοποιήσουμε κατάλληλα, ώστε να συλλέξουμε το *SubID* κάθε συνδρομής που συζευγνύει με τη δημοσίευση μόνο μία φορά (και όχι $l_s - n - 1$). Η διαδικασία που προτείνουμε περιλαμβάνει τα παρακάτω βήματα:

1. **Φάση 1: Κατασκευή των Φίλτρων Bloom.** Κατά την άφιξη μιας δημοσίευσης με συμβολοσειρά $e[1...l_e]$ ο κόμβος στον οποίο κατέφθασε γίνεται ο συντονιστής ολόκληρης της διαδικασίας. Ο συντονιστής τεμαχίζει τη συμβολοσειρά στα *ngrams* που την αποτελούν και την στέλνει στους κόμβους που είναι αρμόδιοι για την διαχείριση των *ngrams*².

Κατόπιν, κάθε ένας από αυτούς τους κόμβους, i , με βάση την συμβολοσειρά της δημοσίευσης και τις συμβολοσειρές των συνδρομών που αντιστοιχούν στα αποθηκευμένα *SubIDs* (μπορούν να βρεθούν στην λίστα V), προσδιορίζουν το μέγεθος του συνόλου RS_i (Result Set) των *SubIDs* που συζευγνύουν με την δημοσίευση και θα πρέπει να αποσταλούν στον συντονιστή. Η τιμή $|RS_i|$ αποστέλλεται πίσω στο συντονιστή, ο οποίος στη συνέχεια, με βάση τη μέγιστη τιμή αυτών $\max(|RS_i|)$, υπολογίζει το μέγεθος (m) των Bloom φίλτρων που θα πρέπει να κατασκευαστούν ώστε να προκύπτει μια αποδεκτή πιθανότητα false positive. Ο συντονιστής στέλνει το μέγεθος m των φίλτρων σε όλους τους εμπλεκόμενους κόμβους.

2. **Φάση 2: Συλλογή των Φίλτρων Bloom.** Κατά τη λήψη της τιμής m , κάθε εμπλεκόμενος κόμβος i κατασκευάζει ένα Bloom φίλτρο, BF_i με μέγεθος m και εισάγει όλα τα *SubIDs* που ανήκουν στο σύνολο των *SubIDs* που θα πρέπει να επιστραφούν στον συντονιστή, RS_i , χρησιμοποιώντας μόνο μία συνάρτηση κατακερματισμού. Κατόπιν τα BF_i αποστέλλονται πίσω στον συντονιστή.

3. **Φάση 3: Συλλογή των SubIDs.** Ο συντονιστής συλλέγει όλα τα BFs και κατασκευάζει τον πίνακα

²Τα αναγνωριστικά αυτών των κόμβων προκύπτουν είτε με την χρήση της συνάρτησης κατακερματισμού με τα *ngrams* σαν όρισμα, είτε αξιοποιώντας την υποδομή *PastryStrings* και επιλέγοντας τους κατάλληλους *Tnodes*.

BFM μεγέθους $(l_e - n - 1) \times m$, αποτελούμενος από τα φίλτρα Bloom που έχει συλλέξει από όλους τους εμπλεκόμενους κόμβους. Σε κάθε γραμμή i του πίνακα BFM βρίσκεται το φίλτρο BF_i του κόμβου i . Εξετάζοντας τις στήλες του BFM μπορούμε να εξαγάγουμε χρήσιμες πληροφορίες. Εάν εξετάσουμε την στήλη j και βρούμε ότι στη γραμμή i η τιμή είναι 1 και αν υποθέσουμε ότι η συνδρομή με $SubID_k$ θέτει την τιμή 1 στη θέση j , τότε το $SubID_k$ είναι αποθηκευμένο στον κόμβο i . Παραδείγματος χάριν, στο Σχήμα 6.3 εάν το bit στη θέση $(i=2, j=2)$ του πίνακα BFM είναι ίσο με 1, και το Sub1 θέτει το bit στη θέση $(j=2)$, τότε μπορούμε να συμπεράνουμε ότι η συνδρομή Sub1 είναι αποθηκευμένη στον κόμβο P_2 , τον κόμβο που έχει στείλει το BF_2 .

Για κάθε εμπλεκόμενο κόμβο i , ο συντονιστής διατηρεί ένα σύνολο (αρχικά κενό) δεικτών στις στήλες του πίνακα BFM (ή στις θέσεις των $bits$ στα φίλτρα BF), γνωστό και ως PS_i . Κατόπιν ελέγχει κάθε μη κενή στήλη του BFM και ανακαλύπτει ποιοι κόμβοι έχουν θέσει το αντίστοιχο bit σε 1. Στη συνέχεια, επιλέγεται τυχαία ³ ένας από αυτούς, για παράδειγμα ο i , και εισάγεται στο PS_i η θέση της τρέχουσας στήλης.

Κάθε σύνολο PS στέλνεται έπειτα πίσω στους εμπλεκόμενους κόμβους. Όταν ο κάθε κόμβος i λάβει το PS_i που του αντιστοιχεί, επανεξετάζει το σύνολο των $SubIDs$ που πρόκειται να στείλει στον συντονιστή RS_i , με το να ελέγχει κάθε $SubID$ που ανήκει σε αυτό το σύνολο σε συνδυασμό με το φίλτρο Bloom και το PS_i . Συγκεκριμένα, χρησιμοποιώντας τη συνάρτηση κατακερματισμού του Bloom φίλτρου ελέγχει τη θέση που ορίζει το κάθε $SubID$ στο φίλτρο BF_i και αναζητάει αυτή τη θέση στο σύνολο PS_i . Αν η θέση υπάρχει στο σύνολο PS_i , τότε το συγκεκριμένο $SubID$ θα πρέπει να επιστραφεί στον συντονιστή.

Παρακάτω μπορείτε να δείτε ένα παράδειγμα, που εκμεταλλεύεται την επέκταση των *ngrams* προσεγγίσεων με την βοήθεια των φίλτρων Bloom που μόλις παρουσιάσαμε.

Παράδειγμα 13 Επεξεργασία συνδρομής και δημοσίευσης με την εκμετάλλευση των Bloom φίλτρων.

Ας υποθέσουμε ότι είναι ήδη αποθηκευμένες τρεις συνδρομές με περιορισμούς υποσυμβολισειράς και τιμές για την κάθε μια: $Sub_1=*101*$, $Sub_2=*1010*$, και $Sub_3=*110*$. Εάν υποθέσουμε ότι τα *ngram* 11, 10, και 01 αντιστοιχίζονται (με την βοήθεια της συνάρτησης κατακερματισμού του DHT δικτύου) στους κόμβους P_1 , P_2 , και P_3 τότε το αναγνωριστικό της συνδρομής Sub_1 θα αποθηκευτεί στους κόμβους P_2 και P_3 μαζί με την συμβολισειρά της στον πίνακα V , το αναγνωριστικό της Sub_2 θα αποθηκευτεί στους κόμβους P_2 και P_3 , και τελικά το αναγνωριστικό της Sub_3 θα αποθηκευτεί στους P_1 και P_2 (Σχήμα 6.3).

³Μπορούμε να επεξεργαστούμε όλα τα PC_i (αφού ολοκληρώσουμε την διαδικασία συμπλήρωσης τους) ώστε να κατανείμουμε το φόρτο πρόσβασης σε όλους τους εμπλεκόμενους κόμβους ομοιόμορφα, αν αυτό είναι δυνατόν.

Σχήμα 6.3: Παράδειγμα της προσέγγισης με ngrams και την χρήση των Bloom φίλτρων.

Μια δημοσίευση καταφθάνει στο σύστημα με συμβολοσειρά 1101. Τεμαχίζοντας τη συμβολοσειρά σε ngrams οδηγούμαστε στους κόμβους P_1 , P_2 , και P_3 όπου και ζητάμε να μας επιστρέψουν τα μεγέθη των συνόλων των SubIDs που συζευγνύουν με την τιμή της δημοσίευσης. Στη συνέχεια υπολογίζουμε την κατάλληλη τιμή του Bloom φίλτρου, που σε αυτό το παράδειγμα ας θεωρήσουμε ότι είναι ίσο με 5 bits ($m = 5$). Ο συντονιστής στέλνει το m στους κόμβους και ζητάει από αυτούς να ετοιμάσουν τα Bloom φίλτρα και να του τα στείλουν. Ο συντονιστής συλλέγει έπειτα τα BF's και κατασκευάζει τον πίνακα BFM όπως μπορείτε να δείτε στο Σχήμα 6.3.

Όπως μπορείτε να δείτε μόνο δύο στήλες έχουν επηρεαστεί (οι στήλες 1 και 2). Η στήλη 1 εμπλέκει τους κόμβους P_1 και P_2 ενώ η στήλη 2 εμπλέκει τους P_2 και P_3 . Επιλέγουμε να ζητήσουμε από τον κόμβο P_1 τα SubIDs που είναι αποθηκευμένα εκεί και έχουν επηρεάσει τη στήλη 1, στέλνοντας του το σύνολο $PS_1 = \{1\}$.

Για τα SubIDs που έχουν επηρεάσει τη στήλη 2 επικοινωνούμε με τον κόμβο P_2 αποστέλλοντας του αντίστοιχα το σύνολο $PS_2 = \{2\}$. Όταν ο κόμβος P_1 λάβει το PS_1 , εξετάζει το σύνολο των αποθηκευμένων SubIDs που συζευγνύουν με τη δημοσίευση και εντοπίζει εκείνα τα SubIDs που επηρεάζουν τη στήλη 1 του φίλτρου BF_1 . Σε αυτό το παράδειγμα ο P_1 βρίσκει ότι σε αυτή τη κατηγορία ανήκει μόνο το Sub_3 οπότε και το στέλνει στον συντονιστή. Ακολουθώντας την ίδια διαδικασία στον κόμβο P_2 , αποστέλλεται στον συντονιστή το Sub_1 . Τα SubIDs λοιπόν που έχουν συγκεντρωθεί στον συντονιστή (και αντιστοιχούν στις συνδρομές Sub_1

και Sub_3) συζευγνύουν με δημοσίευση.

Βλέπετε λοιπόν ότι με την τεχνική αυτή, μειώνουμε τον αριθμό των μεταφερομένων $xSubIDs$ από 4 (όσα δηλαδή είναι τα $ngrams$ των Sub_1 και Sub_3) σε 2. ◇

6.2.1 Τεχνική Μείωσης της Κίνησης στο Δίκτυο

Ένα σημαντικό ζήτημα σχετικά με τη επέκταση των προσεγγίσεων με την χρήση των $ngrams$ και των Bloom φίλτρων είναι ότι κάτω από συγκεκριμένες συνθήκες η κίνηση που παράγεται με την εφαρμογή της διαδικασίας τριών φάσεων μπορεί να είναι μεγαλύτερη από την κίνηση που παράγεται από την απλή $ngrams$ προσέγγιση. Αυτό σημαίνει ότι περιστασιακά μπορεί να είναι προτιμητέο να διακινούμε πολλαπλά αντίγραφα των $SubIDs$ (με την εφαρμογή της απλής $ngrams$ τεχνικής) παρά να προσπαθούμε να τα μειώσουμε με την διαδικασία των τριών φάσεων.

Αυτή η παρατήρηση μας οδήγησε στην επέκταση της διαδικασίας τριών φάσεων ως εξής: κατά τη διάρκεια της πρώτης φάσης και μετά από την συλλογή των $|RS_i|$ από κάθε ένα από τους k εμπλεκόμενους κόμβους $1 \leq i \leq k$, ο συντονιστής υπολογίζει την κίνηση που αναμένεται να παραχθεί είτε με την εφαρμογή της απλής $ngrams$ προσέγγισης (δηλ. την συλλογή όλων των $SubIDs$ που βρίσκονται σε κάθε κόμβο i και συζευγνύουν με τη δημοσίευση και προκύπτει από το άθροισμα $\sum_{i=1}^k |RS_i|$) ή με τη συνέχιση και την εφαρμογή της τρίτης φάσης της διαδικασίας (που περιλαμβάνει την κίνηση που παράγεται με την αποστολή των Bloom φίλτρων, των PSs , των m , και τελικά των $SubIDs$). Αφού υπολογιστεί η αναμενόμενη κίνηση ο συντονιστής επιλέγει να χρησιμοποιήσει την προσέγγιση που παράγει τη μικρότερη κίνηση.

Κατά τη πειραματική αξιολόγηση των προσεγγίσεων που παρουσιάζουμε εδώ (ενότητα 6.4), παρατηρήσαμε ότι όταν οι λίστες των $SubIDs$ που πρόκειται να αποσταλούν στον συντονιστή είναι σχετικά μικρές, είναι προτιμότερο να εφαρμόσουμε την απλή $ngrams$ τεχνική, ενώ για τα μεγαλύτερα μεγέθη το συνολικό κόστος της διαδικασίας τριών φάσεων δείχνει να είναι μικρό (σε σχέση με την απλή $ngrams$ τεχνική).

6.2.2 Τεχνική Αύξησης της Πιθανότητας Σύζευξης

Ένα σημαντικό ζήτημα που προκύπτει από τη χρήση των Bloom φίλτρων είναι ότι πλέον η σύζευξη των δημοσιεύσεων με τις συνδρομές, επηρεάζεται από την false positive πιθανότητα που χαρακτηρίζει αυτά τα φίλτρα. Αυτό σημαίνει ότι υπάρχει πιθανότητα να μην μπορέσουμε να εντοπίσουμε και να συλλέξουμε μερικές συνδρομές που όμως συζευγνύουν με την δημοσίευση, σαν αποτέλεσμα των false positives που μπορούν να προκύψουν κατά την εισαγωγή των αντικειμένων ($SubIDs$) στα Bloom φίλτρα BF_i .

Παραδείγματος χάριν, υποθέστε ότι κατά την εξέταση ενός από ιδιοχαρακτηριστικά της δημοσίευσης, το $SubID_a$ στον κόμβο P_1 θέτει το ίδιο bit στη θέση j με το $SubID_b$ στον κόμβο P_2 . Εάν το σύνολο θέσεων

PS_1 περιλαμβάνει τη θέση j τότε μόνο το $SubID_a$ θα καταφέρει να φτάσει στον συντονιστή.

Αυτή η πιθανότητα μπορεί να ρυθμιστεί ώστε να είναι πολύ μικρή, αυξάνοντας απλά το μέγεθος των φίλτρων BF_i . Για να μειώσουμε όμως περαιτέρω αυτήν την πιθανότητα μπορούμε συνεχίσουμε σε μια τέταρτη φάση επεξεργασίας όπου θα εντοπίσουμε εκείνα τα $SubIDs$ που αν και συζευγνύουν με την δημοσίευση, εντούτοις δεν έχουν εντοπιστεί.

Κατά την επεξεργασία δημοσιεύσεων με περισσότερα από ένα ιδιοχαρακτηριστικά, ο συντονιστής θα έχει συλλέξει ένα σύνολο S_i , από $SubIDs$ για κάθε ένα ιδιοχαρακτηριστικό a_i . Από αυτά τα $SubIDs$ ($\in \bigcup S_i$) ένα υποσύνολο τους M , θα αποτελείται από $SubIDs$ συνδρομών που συζευγνύουν με τη δημοσίευση. Εξετάζουμε στη συνέχεια τα υπόλοιπα $SubIDs$ που ανήκουν στα υποσύνολα R_i με $R_i = S_i \setminus M$. Για κάθε ένα $SubID$ που ανήκει στο σύνολο R_i εξετάζουμε τα BFM_j με $j \neq i$ των ιδιοχαρακτηριστικών a_j , όπου το συγκεκριμένο $SubID$ δεν αποτελεί μέρος του συνόλου R_j (είτε επειδή η συνδρομή δεν συζευγνύει με τη δημοσίευση στην τιμή του ιδιοχαρακτηριστικού a_j , είτε επειδή παρουσιάστηκε false positive κατά την εισαγωγή των $SubIDs$ στα Bloom φίλτρα για το συγκεκριμένο ιδιοχαρακτηριστικό).

Στη συνέχεια, εξετάζουμε τη θέση του BFM_j που μας επιστρέφει η συνάρτηση κατακερματισμού του Bloom φίλτρου με το συγκεκριμένο $SubID$ σαν όρισμα. Εάν εντοπίσουμε ένα τουλάχιστον bit ίσο με 1, και δεδομένου ότι το συγκεκριμένο $SubID$ δεν είναι μέλος του R_j (δηλ. δεν συλλέχτηκε κατά την επεξεργασία του ιδιοχαρακτηριστικού a_j) συμπεραίνουμε ότι έχει παρουσιαστεί false positive κατά τον σχηματισμό των Bloom φίλτρων, και άρα η συγκεκριμένη συνδρομή συζευγνύει με την τιμή του a_j . Συνεχίζουμε με το ίδιο τρόπο και με τα υπόλοιπα ιδιοχαρακτηριστικά και τελικά καταλήγουμε για το ποια από αυτά τα νέα $SubIDs$ αντιστοιχούν σε συνδρομές που συζευγνύουν με τη δημοσίευση, οπότε και μεταφέρονται στο σύνολο M .

6.3 Θεωρητική Ανάλυση Απόδοσης

Δεδομένου ότι σε μία συνδρομή ορίζεται η συμβολοσειρά $s[1...l_s]$ και ότι αντίστοιχα σε μία δημοσίευση η συμβολοσειρά $e[1...l_e]$, ο αριθμός μηνυμάτων που απαιτούνται κατά την επεξεργασία τους σε ένα δίκτυο με N κόμβους παρουσιάζεται στον Πίνακα 6.1.

Όπως μπορείτε να δείτε η προτιμότερη προσέγγιση κατά την επεξεργασία των συνδρομών είναι η complete-IND. Εντούτοις, η επεξεργασία των δημοσιεύσεων στην complete-IND κοστίζει πολύ και δεδομένου ότι οι δημοσιεύσεις αναμένονται να καταφθάνουν με πολύ υψηλότερους ρυθμούς από τις συνδρομές, η επεξεργασία των δημοσιεύσεων θα πρέπει να είναι όσο το δυνατόν γρηγορότερη. Η δεύτερη προσέγγιση, ngrams-IND, αποδίδει ελαφρώς καλύτερα από την complete-IND κατά την επεξεργασία των δημοσιεύσεων.

	Επεξεργασία Συνδρομών	Επεξεργασία Δημοσιεύσεων
complete-IND	$O(\log(N))$	$l_e^2 \times O(\log(N))$
ngrams-IND	$l_s \times O(\log(N))$	$l_e \times O(\log(N))$
complete-PS	$l_s + O(\log(N))$	$l_e^2 + O(\log(N))$
ngrams-PS	$l_s \times n + O(\log(N))$	$l_e \times n + O(\log(N))$

Πίνακας 6.1: Πολυπλοκότητα επεξεργασία δημοσιεύσεων και συνδρομών με τελεστές υποσυμβολοσειράς.

Κατά την εφαρμογή αυτών των δύο λύσεων πάνω στην υποδομή *PastryStrings* βλέπουμε ότι η ngrams-PS είναι η αποδοτικότερη.

Το πλήθος μηνυμάτων που απαιτείται στις προσεγγίσεις ngramsBloom-IND και ngramsBloom-PS προκύπτει από το να προσθέσουμε μία σταθερά (που οφείλεται σε εφαρμογή της διαδικασίας τριών φάσεων) στο πλήθος μηνυμάτων που απαιτούνται στις προσεγγίσεις ngrams-IND και ngrams-PS αντίστοιχα.

6.4 Πειραματική Αξιολόγηση και Μελέτη Απόδοσης

6.4.1 Χαρακτηριστικά και Παράμετροι Πειραματικής Διάταξης

Εκτελέσαμε μια σειρά πειραμάτων σε ένα δίκτυο που προσομοιώνει την λειτουργικότητα ενός DHT δικτύου αποτελούμενο από 800 κόμβους. Παρόμοια αποτελέσματα προκύπτουν και για δίκτυα με πολύ περισσότερους κόμβους.

Χρησιμοποιήσαμε την κατανομή δημοτικότητας Zipf τόσο για τα ιδιοχαρακτηριστικά του μοντέλου δημοσιεύσεων/συνδρομών όσο και για τις τιμές τους. Οι τιμές της παραμέτρου ϑ κυμαίνονται από 0.0 έως 1.6. Το μοντέλο δημοσιεύσεων/συνδρομών που εφαρμόσαμε σ' αυτά τα πειράματα, αποτελείται από 5 ιδιοχαρακτηριστικά. Ο ακριβής αριθμός αυτών που εμπλέκονται στη δήλωση των δημοσιεύσεων και των συνδρομών καθορίζεται από την κατανομή Zipf της δημοτικότητας των ιδιοχαρακτηριστικών. Η δημοτικότητα των τιμών των ιδιοχαρακτηριστικών ακολουθεί και αυτή την Zipf κατανομή και μιας και παίζει σημαντικό ρόλο (για παράδειγμα στην κατανομή του φόρτου αποθήκευσης των *SubIDs*, ή του φόρτου προσπέλασης των κόμβων), μεταβάλλουμε την κλίση της δίνοντας τιμές στη παράμετρο ϑ που κυμαίνονται από 0.0 μέχρι 1.4.

Τα ιδιοχαρακτηριστικά τόσο στις συνδρομές όσο και στις δημοσιεύσεις είναι τύπου συμβολοσειράς με το πλήθος των διαθέσιμων χαρακτήρων που σχηματίζουν τις συμβολοσειρές, β να είναι ίσο με 20 και οι συμβολοσειρές να έχουν μέγιστο μέγεθος 15 χαρακτήρων.

Αν δεν αναφερθεί κάτι διαφορετικό, τα πειράματα μας που αφορούν τις τεχνικές που βασίζονται σε ngrams, εκτελέστηκαν με τις προσεγγίσεις *ngramsBloom* (χρησιμοποιώντας τον συμβολισμό ngramsBloom-IND και ngramsBloom-PS). Το μέγεθος των Bloom φίλτρων σε αυτές τις περιπτώσεις είναι τέτοιο ώστε το πολύ το 2% των συνδρομών που συζευγνύουν με μία δημοσίευση να μην εντοπίζονται (εξαιτίας των false

Σχήμα 6.4: Πλήθος μηνυμάτων κατά την επεξεργασία των δημοσιεύσεων και των συνδρομών συναρτήσει της δημοτικότητας των ιδιοχαρακτηριστικών.

positives).

6.4.2 Πλήθος Μηνυμάτων

Σε αυτό το σύνολο πειραμάτων μετρήσαμε τον αριθμό μηνυμάτων (ή πλήθος βημάτων πάνω στο δίκτυο) που απαιτούνται για τη συλλογή των *SubIDs* κατά την επεξεργασία των δημοσιεύσεων και την αποθήκευση των *SubIDs* κατά την επεξεργασία των συνδρομών, μεταβάλλοντας την κλίση της κατανομής της δημοτικότητας των ιδιοχαρακτηριστικών και των τιμών τους.

Γενικά παρατηρήσαμε την ίδια συμπεριφορά που παρουσιάσαμε κατά τη θεωρητική ανάλυση στην ενότητα 6.3 όπως μπορείτε να δείτε στο Σχήμα 6.4 όπου μεταβάλλουμε την κλίση της κατανομής της δημοτικότητας των ιδιοχαρακτηριστικών. Κατά την επεξεργασία των συνδρομών η προσέγγιση complete-IND αποδίδει καλύτερα σε σχέση με τις υπόλοιπες, ενώ κατά την επεξεργασία των δημοσιεύσεων οι αποδοτικότερες είναι

Σχήμα 6.5: Κίνηση στο δίκτυο κατά την επεξεργασία των δημοσιεύσεων. Σύγκριση των προσεγγίσεων με ngrams με την επέκταση και χρήση των Bloom φίλτρων.

οι ngramsBloom-IND και complete-PS.

Η προσέγγιση ngramsBloom-PS είναι ελαφρώς χειρότερη σε σχέση με την ngramsBloom-IND, επειδή θα πρέπει να προσπελαστούν λίγο περισσότεροι κόμβοι (εκείνοι που είναι κοντά στη ρίζα κάθε δέντρου) ώστε να φτάσουμε στον τελικό κόμβο όπου είναι αποθηκευμένο το κάθε *SubID*. Μπορείτε επίσης να παρατηρήσετε ότι ο αριθμός των μηνυμάτων μειώνεται καθώς η κλίση της δημοτικότητας των ιδιοχαρακτηριστικών γίνεται πιο απότομη, επειδή λιγότερα ιδιοχαρακτηριστικά ορίζονται σε κάθε δημοσίευση ή συνδρομή.

Αναφορικά με την συμπεριφορά των προσεγγίσεων μας σε σχέση με τη κατανομή της δημοτικότητας των τιμών, οι τάσεις μεταξύ των αλγορίθμων είναι οι ίδιες ενώ ο αριθμός των μηνυμάτων μένει σχεδόν αμετάβλητος όσο αυξάνεται η τιμή της μεταβλητής θ . Αυτό είναι αναμενόμενο μιας και η κατανομή της δημοτικότητας των τιμών επηρεάζει κυρίως την κίνηση στο δίκτυο και την κατανομή του φόρτου πρόσβασης και αποθήκευσης των *SubIDs*.

6.4.3 Κίνηση στο Δίκτυο

Το πλήθος των *SubIDs* που επιστρέφονται στον συντονιστή κατά την επεξεργασία των δημοσιεύσεων είναι σχεδόν το ίδιο μεταξύ όλων των προσεγγίσεων ανεξάρτητα από την κατανομή της δημοτικότητας των ιδιο-χαρακτηριστικών και των τιμών τους. Αυτό οφείλεται στο γεγονός ότι και οι *ngramsBloom* και οι *complete* προσεγγίσεις συλλέγουν τον ίδιο ακριβώς αριθμό *SubIDs*.

Εντούτοις, κατά εξέταση της παραγόμενης κίνησης στο δίκτυο (Σχήμα 6.5 (a) σε λογαριθμική κλίμακα), μπορείτε να παρατηρήσετε ότι για λιγότερο απότομες κατανομές δημοτικότητας των τιμών των ιδιο-χαρακτηριστικών, η προσέγγιση *ngramsBloom-IND* παράγει τη μικρότερη κίνηση. Αυτό συμβαίνει επειδή η *ngramsBloom-IND* επικοινωνεί με λιγότερους κόμβους ($l - n$) σε αντίθεση με τις *complete* προσεγγίσεις που επικοινωνούν με (l^2) και την *ngramsBloom-PS* (όπου οι *Tnodes* που βρίσκονται κοντά στη ρίζα θα πρέπει επίσης προσπελασθούν).

Σε κατανομές δημοτικότητας με υψηλότερη κλίση (μεγαλύτερες τιμές του ϑ οι προσεγγίσεις *ngramsBloom* παράγουν περισσότερη κίνηση (κατά περίπου 1.5 φορά, στο Σχήμα 6.5 (a) το μεγεθυμένο τμήμα με γραμμική κλίμακα στους άξονες) εξαιτίας της διαδικασίας τριών φάσεων που ακολουθούμε. Όσον αφορά τη δημοτικότητα των ιδιοχαρακτηριστικών, για μεγάλες τιμές του ϑ η κίνηση μειώνεται επειδή εμπλέκονται λιγότερα ιδιοχαρακτηριστικά σε κάθε συνδρομή ή δημοσίευση.

Εξετάσαμε επίσης και παρουσιάζουμε στο Σχήμα 6.5 (b) το ποσοστό της κίνησης που μπορούμε να κερδίσουμε με την εφαρμογή των *Bloom* φίλτρων σε σχέση με την απλή εφαρμογή της τεχνικής με την χρήση των *ngrams*. Μπορείτε να δείτε ότι με τη χρησιμοποίηση των *Bloom* φίλτρων μπορούμε να μειώσουμε την κίνηση στο δίκτυο κατά περίπου 3 φορές.

6.4.4 Εξισορρόπηση του Φόρτου Πρόσβασης και Αποθήκευσης των SubIDs

Πραγματοποιήσαμε μια σειρά πειραμάτων μετρώντας τον συντελεστή διακύμανσης (CV - Coefficient of Variation) του φόρτου πρόσβασης των κόμβων και αποθήκευσης των *SubIDs*.

Μεταβάλλοντας την παράμετρο ϑ της κατανομής της δημοτικότητας των τιμών (Σχήμα 6.6) παρατηρήσαμε ανισορροπίες του φόρτου πρόσβασης στην προσέγγιση *complete-PS*. Αυτό οφείλεται στο γεγονός ότι τα δέντρα που τελικά σχηματίζονται, έχουν μεγάλο βάθος και μικρό πλάτος (που μεταφράζεται σε μικρό βαθμό g), με αποτέλεσμα το πλήθος των κόμβων που βρίσκονται κοντά στη ρίζα να είναι σχετικά μικρό και άρα αυτοί οι κόμβοι να προσπελάζονται επανειλημμένα κατά την εξυπηρέτηση των αιτήσεων δρομολόγησης προς κόμβους που βρίσκονται σε μεγαλύτερα βάθη. Αντίθετα, στην προσέγγιση *ngramsBloom-PS* τα δέντρα έχουν μικρό βάθος και υψηλό βαθμό (g) με συνέπεια ο φόρτος πρόσβασης να εξισορροπείται και να κατανέμεται σε όσο το δυνατόν περισσότερους κόμβους.

Σχήμα 6.6: Κατανομή του φόρτου πρόσβασης και αποθήκευσης συναρτήσει της κατανομής της δημοτικότητας των τιμών των ιδιοχαρακτηριστικών.

Κατά την εξέταση το φόρτου αποθήκευσης οι ανεξάρτητες από το DHT προσεγγίσεις (* – *IND*) προκαλούν μεγαλύτερες ανισορροπίες σε σχέση με αυτές που εφαρμόζονται πάνω στην υποδομή *PastryStrings* όταν οι τιμές της παραμέτρου θ είναι μικρές. Για υψηλότερες τιμές παρατηρούμε ότι η προσέγγιση *complete-IND* οδηγεί σε ένα ιδιαίτερα ανισόρροπο, από πλευράς φόρτου αποθήκευσης, δίκτυο.

Μπορούμε να μειώσουμε οποιοδήποτε είδος ανισορροπιών, εάν εφαρμόσουμε την τεχνική της αντιγραφής (replication). Παρατηρήσαμε παραδείγματος χάριν, ότι ο συντελεστής διακύμανσης (CV) του φόρτου πρόσβασης στην προσέγγιση *ngramsBloom-PS* μειώνεται κατά 60% όταν διατηρούμε στο δίκτυο μας 6 αντίγραφα (replicas) για κάθε δέντρο στην υποδομή *PastryStrings*.

Επίσης παρατηρήσαμε ότι ο συντελεστής διακύμανσης μπορεί να μειωθεί περαιτέρω σε μία πλήρη *PastryStrings* υποδομή όπου όλοι οι δυνατοί περιορισμοί σε αριθμητικές τιμές και συμβολοσειρές μπορούν να οριστούν στις συνδρομές. Η ανισορροπία που μπορεί να εμφανιστεί κατά την εφαρμογή των προσεγγίσεων

	complete-IND	ngrams-IND	ngramsBloom-IND
DHT Selection Flexibility	YES	YES	YES
Extra Routing State	NO	NO	NO
Number of Messages (events)	Very Bad	Very Good	Good
Total Network Traffic	Good	Bad	Very Good
Result Set Sizes	Very Good	Bad	Very Good
	complete-PS	ngrams-PS	ngramsBloom-PS
DHT Selection Flexibility	NO	NO	NO
Extra Routing State	YES	YES	YES
Number of Messages (events)	Very Good	Very Good	Good
Total Network Traffic	Good	Very Bad	Good
Result Set Sizes	Very Good	Bad	Very Good

Πίνακας 6.2: Χαρακτηριστικά και απόδοση των προτεινόμενων προσεγγίσεων για την επεξεργασία τελεστών υποσυμβολοσειρών.

με *ngrams* και του γεγονότος ότι οι κόμβοι που βρίσκονται κοντά στη ρίζα προσπελάζονται επανειλημμένα, μπορεί να μειωθεί εάν όλοι οι άλλοι κόμβοι προσπελάζονται λόγω της επεξεργασίας των υπολοίπων τελεστών (σε αριθμητικές τιμές και συμβολοσειρές).

Εξετάσαμε πώς ο συντελεστής διακύμανσης επηρεάζεται με τη μεταβολή της παραμέτρου ϑ της κατανομής της δημοτικότητας των τιμών σε ένα περιβάλλον όπου υποστηρίζεται μόνο ο τελεστής υποσυμβολοσειράς και σε μία πλήρως λειτουργική *PastryStrings* υποδομή. Παρατηρήσαμε λοιπόν ότι ο συντελεστής διακύμανσης του φόρτου πρόσβασης μειώνεται κατά 20% ενώ ο φόρτος αποθήκευσης κατά 50%.

6.4.5 Σχολιασμός των Χαρακτηριστικών και της Απόδοσης των Προτεινόμενων Προσεγγίσεων

Χρησιμοποιώντας τα συμπεράσματα που αποκομίσαμε από την θεωρητική και πειραματική μελέτη των προσεγγίσεων που προτείνουμε για την επεξεργασία του τελεστή υποσυμβολοσειράς, παρουσιάζουμε τον πίνακα 6.2. Στο πίνακα αυτό μπορεί κανείς να δει συγκεντρωτικά την απόδοση όλων των προτεινόμενων προσεγγίσεων αναφορικά με: (i) το αν μπορούν να εφαρμοστούν πάνω από οποιοσδήποτε DHT δίκτυο - *DHT Selection Flexibility*, (ii) την επιπλέον κατάσταση δρομολόγησης που πιθανόν να χρειάζεται να διατηρούμε - *Extra Routing State*, (iii) τον αριθμός μηνυμάτων που απαιτούνται κατά τη διάρκεια της επεξεργασίας των δημοσιεύσεων - *Number of Messages*, (iv) τη συνολική κίνηση που παράγεται στο δίκτυο κατά την επεξεργασία των δημοσιεύσεων - *Total Network Traffic* και (v) το πλήθος των *SubIDs* που συλλέγονται - *Result Set Sizes*. Στον πίνακα παρουσιάζεται η απόδοση των προσεγγίσεων χρησιμοποιώντας την κλίμακα *Very Bad*, *Bad*, *Good*, και *Very Good* για κάθε μετρική.

Γενικά η προσέγγιση complete-IND έχει πολύ καλή απόδοση σχετικά με το πλήθος των *SubIDs* που

επιστρέφονται (μιας και μεταφέρεται το ίδιο *SubID* παραπάνω από μία φορά). Εντούτοις, έχει χαμηλή απόδοση όταν κανείς κοιτάει το πλήθος των μηνυμάτων που απαιτούνται κατά την επεξεργασία των δημοσιεύσεων μιας και θα πρέπει να ελεγχθούν όλες οι δυνατές υποσυμβολοσειρές της τιμής της δημοσίευσης. Στην άλλη πλευρά βρίσκεται η προσέγγιση ngrams-IND που έχει πολύ καλή απόδοση όσον αφορά τον αριθμό των μηνυμάτων αλλά πάσχει από τον αυξημένο αριθμό μεταφερόμενων *SubIDs* και κατ' επέκταση από την αύξηση της κίνησης στο δίκτυο.

Οι προσεγγίσεις complete-PS και ngrams-PS αντιμετωπίζουν τα προβλήματα απόδοσης των complete-IND και ngrams-IND εις βάρος της εισαγωγής και διατήρησης επιπρόσθετης κατάστασης δρομολόγησης στο δίκτυο (που σχετίζεται με το γενικό κόστος διατήρησης αυτής της πληροφορίας) θυσιάζοντας έτσι την ευελιξία του να μπορούμε να εφαρμόσουμε την λύση που προτείνουμε πάνω από οποιοδήποτε DHT.

Τέλος, οι προσεγγίσεις που βασίζονται στην χρήση των Bloom φίλτρων, αποδίδουν σχετικά καλά αναφορικά με όλες τις μετρικές που εξετάζουμε. Η προσέγγιση ngramsBloom-IND απολαμβάνει το όφελος της ευρείας εφαρμογής της πάνω από οποιοδήποτε DHT χωρίς την διατήρηση επιπρόσθετης κατάστασης δρομολόγησης. Από την άλλη η προσέγγιση ngramsBloom-PS μπορεί να εφαρμοστεί σε συγκεκριμένο DHT, και πάνω στην υποδομή *PastryStrings* που μπορεί να εξυπηρετήσει ένα μεγάλο σύνολο τελεστών πάνω σε αριθμητικές τιμές και συμβολοσειρές.

7 Επίλογος

Η ανάπτυξη συστημάτων δημοσιεύσεων/συνδρομών σε δίκτυα μεγάλης κλίμακας, αποτελεί έναν ενδιαφέροντα τομέα έρευνας και ανάπτυξης. Ειδικότερα, οι ανεξάρτητες από το δίκτυο λύσεις είναι επιθυμητές δεδομένου ότι είναι ευρέως εφαρμόσιμες πάνω από οποιοδήποτε δίκτυο ομοτίμων εταίρων που βασίζεται σε κατανεμημένους πίνακες κατακερματισμού. Μπορούν όμως να θυσιάζουν ορισμένα οφέλη που μπορούμε να αποκομίσουμε από την εκμετάλλευση των ιδιαίτερων χαρακτηριστικών συγκεκριμένων δικτύων.

Προς αυτή τη κατεύθυνση προτείνουμε την υποδομή *Pyracanthus*, που μπορεί να υποστηρίξει την επεξεργασία συνδρομών και δημοσιεύσεων πάνω από οποιοδήποτε DHT. Στον αντίποδα αυτής της σχεδιαστικής επιλογής βρίσκεται η δεύτερη υποδομή που προτείνουμε, η *PastryStrings*, που εκμεταλλεύεται για τον ίδιο σκοπό το δίκτυο ομοτίμων εταίρων Pastry ή οποιοδήποτε άλλο δίκτυο που εκτελεί δρομολόγηση κατά πρόθεμα στο πρότυπο του Pastry.

Η υποδομή *Pyracanthus* αποτελείται από ένα σύνολο νέων αλγορίθμων για την επεξεργασία των συνδρομών και των δημοσιεύσεων, με ιδιοχαρακτηριστικά αριθμητικού τύπου και τύπου συμβολοσειράς. Οι γενικευμένες δεντρικές δομές που έχουμε αναπτύξει για αυτό το σκοπό, δείχνουν να μπορούν να ρυθμιστούν κατάλληλα (ανάλογα με την εφαρμογή που τα αξιοποιεί) ώστε για παράδειγμα να επιτύχουμε αποδοτικότερη επεξεργασία των συνδρομών έναντι των δημοσιεύσεων ή το αντίθετο. Ομοίως, η υποδομή *Pyracanthus* μπορεί να ρυθμιστεί κατάλληλα ώστε να αξιοποιεί stateful ή stateless προσεγγίσεις για την εξυπηρέτηση αριθμητικών τιμών ή συμβολοσειρών.

Ειδικότερα, η υποδομή *Pyracanthus* περιλαμβάνει τρεις διαφορετικές προσεγγίσεις για την επεξεργασία των συνδρομών και των δημοσιεύσεων με ιδιοχαρακτηριστικά αριθμητικού τύπου. Δύο από αυτές (stateless) δεν απαιτούν την διατήρηση πληροφορίας σχετική με τη δρομολόγηση (πέρα από αυτή που είναι απαραίτητη στο ίδιο το DHT δίκτυο), ενώ η τρίτη είναι stateful. Επιπλέον, η υποδομή *Pyracanthus* εξοπλίζεται με δύο προσεγγίσεις, μια stateful και μια stateless, για την εξυπηρέτηση ιδιοχαρακτηριστικών τύπου συμβολοσειράς.

Όλοι οι προτεινόμενοι αλγόριθμοι (είτε για αριθμητικά ιδιοχαρακτηριστικά ή για τα ιδιοχαρακτηριστικά τύπου συμβολοσειράς) αξιολογήθηκαν αναλυτικά όσον αφορά το πλήθος των μηνυμάτων που απαιτούνται κατά την εκτέλεση τους, ενώ παράλληλα υλοποιήθηκαν και εξετάστηκαν με τεχνικές προσομοίωσης, ώστε

να επιβεβαιώσουμε τα θεωρητικά αποτελέσματα μας και παράλληλα να ελέγξουμε την ορθότητα τους αλλά και τα ιδιαίτερα χαρακτηριστικά τους που αρχικά ήταν δυσδιάκριτα. Συγκεκριμένα, η πειραματική μελέτη περιελάμβανε μετρικές απόδοσης όπως οι απαιτήσεις σε εύρος ζώνης, το πλήθος μηνυμάτων (ή βημάτων πάνω στο δίκτυο), το πλήθος των μεταφερόμενων *SubIDs*, η κίνηση που παράγεται στο δίκτυο, η εξισορρόπηση του φόρτου πρόσβασης και αποθήκευσης, η ποσότητα πληροφορίας σχετική με την δρομολόγηση που πρέπει να διατηρείται (όπου αυτό είναι απαραίτητο), απαιτήσεις σε χώρο αποθήκευσης κ.α.

Τα βασικά συμπεράσματα που προκύπτουν από την αξιολόγησή μας είναι ότι: (i) η εφαρμογή των προτεινόμενων αλγορίθμων κατανέμει τον φόρτο πρόσβασης και αποθήκευσης σχετικά ικανοποιητικά σε όλο το δίκτυο και (ii) το κόστος επεξεργασίας των συνδρομών και των δημοσιεύσεων (από πλευράς παραγόμενων μηνυμάτων και απαιτήσεων σε εύρος ζώνης) είναι σχετικά μικρό, κάτι που αντιστοιχεί σε μερικά μόνο lookups πάνω στο δίκτυο DHT.

Συγκεκριμένα, η προσέγγιση stateful *Range Tree Forest* απαιτεί την εκτέλεση λιγότερων από 40 βήματα για την επεξεργασία των συνδρομών και των δημοσιεύσεων, σε ένα δίκτυο αποτελούμενο από 1000 κόμβους. Αυτό το γεγονός δείχνει ότι οι ανεξάρτητες από το υποκείμενο δίκτυο προσεγγίσεις, όπως η υποδομή *Pyracanthus*, μπορούν να θεωρηθούν ως μία εναλλακτική λύση για την ανάπτυξη μεγάλης κλίμακας συστημάτων δημοσιεύσεων/συνδρομών που βασίζονται στο περιεχόμενο.

Δεδομένου των θετικών χαρακτηριστικών που παρατηρήσαμε στην υποδομή *Pyracanthus*, θελήσαμε να προχωρήσουμε ένα βήμα παραπέρα και να εκμεταλλευτούμε τα ιδιαίτερα εκείνα στοιχεία ενός συγκεκριμένου DHT (*Patsry*) ώστε να βελτιώσουμε ακόμα περισσότερο την απόδοση. Η υποδομή *PastryStrings* λοιπόν είναι εξαρτώμενη από το δίκτυο ομοτίμων εταίρων πάνω στο οποίο λειτουργεί, χωρίς να παρεμποδίζει τη λειτουργικότητα του αλλά απλά με το να εκμεταλλεύεται την πληροφορία (στους πίνακες δρομολόγησης του) που έτσι κι αλλιώς υπάρχει, ώστε να επιτύχει αποδοτική επεξεργασία των συνδρομών και των δημοσιεύσεων.

Στα πλαίσια του *PastryStrings*, αναπτύξαμε και εξετάσαμε εκτενώς διάφορους αλγορίθμους που εγγυώνται ότι θα λειτουργεί αποτελεσματικά κάτω από συχνές αφίξεις και αναχωρήσεις κόμβων (σεβόμενος την αυτονομία κόμβων που τους χαρακτηρίζει στα δίκτυα ομοτίμων εταίρων) και θα προσαρμόζεται στη δυναμική τους, αντιμετωπίζοντας επιτυχώς πιθανές βλάβες που μπορεί να υπάρξουν. Η πειραματική αξιολόγηση αυτών των αλγορίθμων επιβεβαίωσε την ορθή λειτουργία τους, ενώ παράλληλα έδειξε ότι το κόστος που πρέπει να πληρώσουμε για την εφαρμογή τους είναι σχετικά μικρό.

Τέλος, αναπτύξαμε και παρουσιάσαμε διάφορες προσεγγίσεις που στοχεύουν στην επεξεργασία δημοσιεύσεων και σύζευξη αυτών με συνδρομές που ορίζουν περιορισμούς υποσυμβολοσειράς. Συγκεκριμένα, παρουσιάσαμε τις προσεγγίσεις *complete* και *ngrams* που μπορούν να εφαρμοστούν είτε στην υποδομή *Pyracanthus* ή στην υποδομή *PastryStrings*.

Η θεωρητική και πειραματική μελέτη αυτών, έδειξαν ότι η *complete* και *ngrams* προσφέρουν πλεονεκτήματα και μειονεκτήματα ανάλογα με τις μετρικές που εξετάζουμε. Για αυτόν τον λόγο παρουσιάσαμε την επέκταση τους, με χρήση των *ngrams* που εκμεταλλεύονται τα φίλτρα Bloom, ώστε να προσφέρουν σχετικά καλή απόδοση για όλες τις μετρικές μας. Η *ngramsBloom* αποτελεί μια πιθανολογική προσέγγιση για τη σύζευξη των δημοσιεύσεων με τις συνδρομές, κάτι που πιστεύουμε ότι είναι πιο κατάλληλο για την ανάπτυξη μεγάλης κλίμακας εφαρμογών διαχείρισης της πληροφορίας σε δομημένα δίκτυα ομοτίμων εταιρών.

Ευρετήριο

PastryStrings

- Αντιστοίχιση διαστήματος σε *Tnode*, 98
- Αντιστοίχιση ετικέτας *Tnode* σε διάστημα, 99
- Δείκτης *FP*, 104
- Εντοπισμός ρίζας, 110
- Πίνακας Δρομολόγησης *CRT*, 104
- Attribute, 19, 35
- Bayeux, 24
- BitTorrent, 12
- Bloom Filter, 134
- Broker, 33
- CAN, 17
- Chord, 15
- Consistent Hashing, 16
- Distributed Hash Table, DHT, 12
- False positives, 135
- Flat topologies, 8
- Freenet, 8
- Gini Coefficient - G, 76
- Gnutella, 9
- Gryphon, 26
- Hermes, 27
- Hierarchical topologies, 8
- Kazaa, 11
- Kyra, 29
- Lorenz curve, 76
- MEDYM, 29
- Meghdoot, 28
- Multidimensional Processing, 69
 - Coordinated Matching, 69
 - Distributed Matching, 70
 - Hybrid Matching, 72
- Napster, 11
- PastryStrings, 91
- Pastry, 13
- Peer-to-Peer, P2P, 7
- Producer, 33
- Pyracanthus, 39
- REBECA, 25
- SIENA, 24
- Scribe, 23
- String Trees, 93
- Subscriber, 33
- Successor, 16

Zipf, 73

CRT - Children Routing Table, 104

FP - Father Pointer, 104

bigram, 129

ngram, 129

Pyracanthus

Αντιστοίχιση ετικέτας *Tnode* σε διάστημα, 46

Διάσπαση διαστημάτων σε υποδιαστήματα, 47

Υπολογισμός των ετικετών των κόμβων ενός μο-
νοπατιού, 50

RHolder, 110

SubID, 36

xSubID, 129

Αναγνωριστικό Συνδρομής *SubID*, 36

Αναγνωριστικό Συνδρομής *xSubID*, 129

Βαθμός δέντρου, 45

Δίκτυα Ομοτίμων Εταίρων, ΔΟΕ, 7

Δομημένα, 12

Μη Δομημένα, 8

Διαμεσολαβητής, 33

Δρομολόγηση κατά πρόθεμα, 13

Εντοπισμός κόμβου σε DHT, 37

Επίπεδες τοπολογίες, 8

Επεξεργασία δημοσίευσης, 34

Επεξεργασία διαστημάτων με δενδρικές δομές, 45,
49, 55, 93

Επεξεργασία συνδρομής, 34

Φίλτρο Bloom, 134

Ιδιοχαρακτηριστικό, 19, 35

Ακρίβεια, 35

Πεδίο ορισμού, 35

Ιεραρχικές τοπολογίες, 8

Καμπύλη Lorenz, 76

Κατανεμημένος Πίνακας Κατακερματισμού, ΚΠΚ, 12

Κατανομή δημοτικότητας Zipf, 73

Μοντέλο Δημοσιεύσεων/Συνδρομών, 35

Attribute-based, 35

Object-based, 35

- Tuple-based, 35
- Παραγωγός, 33
- Πολυδιάστατη επεξεργασία, 69
 - Κατανεμημένη Σύζευξη, 70
 - Κεντριοποιημένη Σύζευξη, 69
 - Υβριδική Σύζευξη, 72
- Προϋπόθεση Σύζευξης με ngrams, 130
- Προσεγγίσεις με κόμβο «ραντεβού», 34
- Προϋπόθεση Σύζευξης Δημοσίευσης με Συνδρομές, 35
- Σύζευξη Δημοσίευσης με Συνδρομές , 60
- Συμβολοσειρά bigram, 129
- Συμβολοσειρά ngram, 129
- Συνάρτηση Κατακερματισμού με Διατήρηση Τάξης, 40
- Συνδρομητής, 33
- Συντελεστής Gini - G, 76
- Συστήματα Δημοσιεύσεων Συνδρομών, 18
 - Βασισμένα σε θεματικές ενότητες, 19
 - Βασισμένα στο περιεχόμενο, 19
- Συστήματα Δημοσιεύσεων Συνδρομών Βασισμένα στο Περιεχόμενο
 - DHT-Dependent, 21
 - DHT-Independent, 21
 - Stateful, 22
 - Stateless, 23
- Τελεστής Ιδιοχαρακτηριστικού Συνδρομής, 35
 - Διάστημα τιμών, range, 35
 - Επίθεμα, suffix, 35
 - Ισότητα, equality, 35
 - Πρόθεμα, prefix, 35
 - Υποσυμβολοσειρά, substring, 127
- Υψος δέντρου, 45

Γλωσσάρι

Access	Προσπέλαση
Address space	Χώρος διευθύνσεων
Attribute	Ιδιοχαρακτηριστικό
Balancing	Εξισορόπηση
Bandwidth	Εύρος ζώνης
Broker	Διαμεσολαβητής
Client	Χρήστης
Connectivity	Συνεκτικότητα
Consistent hashing	Συνεπής κατακερματισμός
Constraint	Περιορισμός
Content-based	Βασισμένο στο Περιεχόμενο
Continuous query	Ερώτημα συνεχούς ροής
Coordinated matching	Κεντρικοποιημένη σύζευξη
Coordinator	Συντονιστής
Data streams	Συνεχής ροή δεδομένων
Distributed Hash Table - DHT	Κατανεμημένος Πίνακας Κατακερματισμού - Κ.Π.Κ.
Distributed matching	Κατανεμημένη σύζευξη
Domain	Εύρος τιμών
Failure	Βλάβη (αναφορικά με κόμβους)
Failure	Σφάλμα (αναφορικά με κόμβους)
False positives	Εσφαλμένη θετική απάντηση
Fault tolerance	Ανοχή σε σφάλματα, βλάβες
Flat topology	Επίπεδη τοπολογία
Hash function	Συνάρτηση κατακερματισμού
Hierarchical topology	Ιεραρχική τοπολογία

Hop count	Πλήθος των βημάτων
Hops	Βημάτων πάνω στο δίκτυο
Hybrid matching	Υβριδική σύζευξη
Identifier	Αναγνωριστικό
Keyword	Λέξη κλειδί
Label	Ετικέτα
Mapping	Αντιστοίχιση
Matching Event to Subscription	Σύζευξη Δημοσίευσης με Συνδρομή
Match	Σύζευξη
Message count	Πλήθος των μηνυμάτων
Network connection	Σύνδεση μεταξύ των κόμβων στο δίκτυο
Network hops	Βημάτων πάνω στο δίκτυο
Network traffic	Κίνηση στο δίκτυο
Node	Κόμβος
Order preserving hash function	Συνάρτηση κατακερματισμού με διατήρηση τάξης
Overlay network	Επικαλυπτικό δίκτυο
Peer-to-Peer - P2P	Δίκτυα ομότιμων εταίρων - Δ.Ο.Ε.
Pointer (in routing table)	Δείκτης (σε πίνακα δρομολόγησης)
Popularity distribution	Κατανομή δημοτικότητας
Prefix	Πρόθεμα
Publication	Δημοσίευση
Publisher	Παραγωγός πληροφορίας
Query	Ερώτημα
Range query	Ερώτημα με τελεστή διαστήματος τιμών
Range	Διάστημα
Replication Factor	Συντελεστή αντιγραφής
Replication	Ανάπτυξη και διατήρηση αντιγράφων
Replica	Αντίγραφο
Response time	Χρόνος απόκρισης
Result Set	Σύνολο Αποτελεσμάτων

Round-trip time	Χρόνος διάδοσης μετέπιστροφής
Routing State	Πληροφορία σχετική με δρομολόγησης
Routing table entry	Εγγραφή του Πίνακα δρομολόγησης
Routing table	Πίνακας δρομολόγησης
Routing	Δρομολόγηση
Scalability	Κλιμάκωση
Server	Εξυπηρετητής
Skewness	Κλίση κατανομής
Steady state	Κατάσταση ηρεμίας
Storage	Αποθήκευση
String	Συμβολοσειρά
Subrange	Υποδιάστημα
Subscriber	Συνδρομητής
Subscription	Συνδρομή
Substring	Υποσυμβολοσειρά
Subtree	Υποδέντρο
Suffix	Επίθεμα
Super-peer	Υπερκόμβος
Topic-based	Βασισμένο σε Θεματική Ενότητα
Topology	Τοπολογία
Website	Ιστότοπος
Weighted mean	Σταθμισμένη μέση τιμή
Workload	Φόρτος
Εύρος ζώνης	Bandwidth
Ανάπτυξη και διατήρηση αντιγράφων	Replication
Αναγνωριστικό	Identifier
Ανοχή σε σφάλματα, βλάβες	Fault tolerance
Αντίγραφο	Replica
Αντιστοίχιση	Mapping
Αποθήκευση	Storage
Βασισμένο σε Θεματική Ενότητα	Topic-based

Βασισμένο στο Περιεχόμενο	Content-based
Βημάτων πάνω στο δίκτυο	Network hops
Βλάβη (αναφορικά με κόμβους)	Failure
Δίκτυα ομότιμων εταίρων - Δ.Ο.Ε.	Peer-to-Peer - P2P
Δείκτης (σε πίνακα δρομολόγησης)	Pointer (in routing table)
Δημοσίευση	Publication
Διάστημα	Range
Διαμεσολαβητής	Broker
Δρομολόγηση	Routing
Εγγραφή του Πίνακα δρομολόγησης	Routing table entry
Εξισορόπηση	Balancing
Εξυπηρετητής	Server
Επίθεμα	Suffix
Επίπεδη τοπολογία	Flat topology
Επικαλυπτικό δίκτυο	Overlay network
Ερώτημα με τελεστή διαστήματος τιμών	Range query
Ερώτημα συνεχούς ροής	Continuous query
Ερώτημα	Query
Εσφαλμένη θετική απάντηση	False positives
Ετικέτα	Label
Εύρος τιμών	Domain
Ιδιοχαρακτηριστικό	Attribute
Ιεραρχική τοπολογία	Hierarchical topology
Ιστότοπος	Website
Κίνηση στο δίκτυο	Network traffic
Κατάσταση ηρεμίας	Steady state
Κατανεμημένη σύζευξη	Distributed matching
Κατανεμημένος Πίνακας Κατακερματισμού - Κ.Π.Κ.	Distributed Hash Table - DHT
Κατανομή δημοτικότητας	Popularity distribution
Κεντριοποιημένη σύζευξη	Coordinated matching
Κλιμάκωση	Scalability

Κλίση κατανομής	Skewness
Κόμβος	Node
Λέξη κλειδί	Keyword
Πίνακας δρομολόγησης	Routing table
Παραγωγός πληροφορίας	Publisher
Περιορισμός	Constraint
Πλήθος των βημάτων	Hop count
Πλήθος των μηνυμάτων	Message count
Πληροφορία σχετική με δρομολόγησης	Routing State
Προσπέλαση	Access
Πρόθεμα	Prefix
Σταθμισμένη μέση τιμή	Weighted mean
Συμβολοσειρά	String
Συνάρτηση κατακερματισμού με διατήρηση τάξης	Order preserving hash function
Συνάρτηση κατακερματισμού	Hash function
Συνδρομή	Subscription
Συνδρομητής	Subscriber
Συνεκτικότητα	Connectivity
Συνεπής κατακερματισμός	Consistent hashing
Συνεχής ροή δεδομένων	Data streams
Συντελεστή αντιγραφής	Replication Factor
Συντονιστής	Coordinator
Σφάλμα (αναφορικά με κόμβους)	Failure
Σύζευξη Δημοσίευσης με Συνδρομή	Matching Event to Subscription
Σύζευξη	Match
Σύνδεση μεταξύ των κόμβων στο δίκτυο	Network connection
Σύνολο Αποτελεσμάτων	Result Set
Τοπολογία	Topology
Υβριδική σύζευξη	Hybrid matching
Υπερκόμβος	Super-peer
Υποδέντρο	Subtree

Υποδιάστημα

Subrange

Υποσυμβολοσειρά

Substring

Φόρτος

Workload

Χρήστης

Client

Χρόνος απόκρισης

Response time

Χρόνος διάδοσης μετέπιστροφής

Round-trip time

Χώρος διευθύνσεων

Address space

Βιβλιογραφία

- [1] *CORBA services: Common Object Services Specification*, 1998.
- [2] Napster peer-to-peer network, [online] <http://www.napster.com/>. 1999.
- [3] Kazaa peer-to-peer network, [online] <http://www.kazaa.com/>. 2001.
- [4] Bittorent peer-to-peer network, [online] <http://www.bittorent.org/>. 2003.
- [5] I. Aekaterinidis and P. Triantafillou. Internet scale string attribute publish/subscribe data networks. In *14th ACM Conference on Information and Knowledge Management (CIKM05)*, 2005.
- [6] I. Aekaterinidis and P. Triantafillou. Pastrystings: A comprehensive content-based publish/subscribe DHT network. In *the 26th IEEE International Conference on Distributed Computing and Systems (ICDCS 06)*, 2006.
- [7] I. Aekaterinidis and P. Triantafillou. Publish/subscribe information delivery with substring predicates. *IEEE Internet Computing Special Issue on Dynamic Information Dissemination*, 2007.
- [8] I. Aekaterinidis and P. Triantafillou. Substring matching in P2P publish/subscribe data management networks. In *the IEEE 23rd International Conference on Data Engineering, (ICDE 2007), (short paper)*, 2007.
- [9] Aguilera, Strom, Sturman, Astley, and Chandra. Matching events in a content-based subscription system. In *Proceedings of the 18th ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing*, 1999.
- [10] M. Altinel and M. J. Franklin. Efficient filtering of XML documents for selective dissemination of information. In *Proceedings of 26th International Conference on Very Large Data Bases, VLDB*, 2000.
- [11] M. Astley, J. Auerbach, S. Bhola, G. Buttner, M. .Kaplan, K. Miller, R. Saccone, R. Strom, D. C. Sturman, M. J. Ward, and Y. Zhao. Achieving scalability and throughput in a publish/subscribe system. In *IBM Research Report RC23103*, 2004.
- [12] S. Babu and J. Widom. Continuous queries over data streams. *SIGMOD Record (ACM Special Interest Group on Management of Data)*, 2001.
- [13] S. Baehni, P. Th, and E. Guerraoui. Data-aware multicast. In *Proceedings of the 2004 International Conference on Dependable Systems and Networks (DSN 2004), Dependable Computing and Communications Symposium*, 2004.
- [14] R. Baldoni, C. Marchetti, A. Virgillito, and R. Vitenberg. Content-based publish-subscribe over structured overlay networks. In *25th IEEE International Conference on Distributed Computing Systems (25th ICDCS'2005)*, 2005.
- [15] G. Banavar, T. Chandra, B. Mukherjee, and J. Nagarajao. An efficient multicast protocol for content-based publish-subscribe systems. In *Proceedings of the 9th International Conference on Distributed Computing Systems (19th ICDCS'99)*, 1999.

- [16] P. A. Bernstein, F. Giunchiglia, A. Kementsietsidis, J. Mylopoulos, L. Serafini, and I. Zaihrayeu. Data management for peer-to-peer computing: A vision. In *Proceedings of the WebDB*, 2002.
- [17] A. R. Bharambe, M. Agrawal, and S. Seshan. Mercury: supporting scalable multi-attribute range queries. In *Proceedings of the ACM SIGCOMM 2004 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, 2004.
- [18] A. R. Bharambe, S. G. Rao, and S. Seshan. Mercury: a scalable publish-subscribe system for internet games. In *Proceedings of the 1st Workshop on Network and System Support for Games, NETGAMES*, 2002.
- [19] S. Bianchi, A. K. Datta, P. Felber, and M. Gradinariu. Stabilizing peer-to-peer spatial filters. In *Proceedings of the 27th International Conference on Distributed Computing Systems (ICDCS 2007)*, 2007.
- [20] S. Bianchi, P. Felber, and M. Gradinariu. Content-based publish/subscribe using distributed r-trees. In *Proceedings of the 13th International Euro-Par Conference*, 2007.
- [21] K. P. Birman. The process group approach to reliable distributed computing. *Communications of the ACM*, 36(12):36-53, 1993.
- [22] B. H. Bloom. Space/time trade-offs in hash coding with allowable errors. *Commun. ACM*, 13(7):422-426, 1970.
- [23] W. J. Bolosky, J. R. Douceur, D. Ely, and M. Theimer. Feasibility of a serverless distributed file system deployed on an existing set of desktop PCs. In *Proceedings of the International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS)*, 2000.
- [24] P. A. Boncz and C. Treijtel. AmbientDB: Relational query processing in a P2P network. In *Proceedings of the First International Workshop on Databases, Information Systems, and Peer-to-Peer Computing, DBISP2P*, 2003.
- [25] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker. Web caching and zipf-like distributions: evidence and implications. *Proceedings of the Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies, INFOCOM '99*, 1999.
- [26] F. Cao and J. Singh. Medym: Match-early with dynamic multicast for content-based publish-/subscribe networks. In *Proceedings of the 6th International Middleware Conference, Middleware 2005*, 2005.
- [27] F. Cao and J. P. Singh. Efficient event routing in content-based publish/subscribe service network. In *INFOCOM*, 2004.
- [28] F. Cao and J. P. Singh. Towards scalable publish-subscribe service networks. Technical report, Princeton University, 2005.
- [29] A. Carzaniga, E. D. Nitto, D. S. Rosenblum, and A. L. Wolf. Issues in supporting event-based architectural styles. In *Third International Software Architecture Workshop*, 1998.
- [30] A. Carzaniga, D. S. Rosenblum, and A. L. Wolf. Design and evaluation of a wide-area event notification service. *ACM Transactions on Computer Systems*, 2001.
- [31] A. Carzaniga, M. J. Rutherford, and A. L. Wolf. A routing scheme for content-based networking. In *Proceedings of the INFOCOM Conference on Computer Communications (INFOCOM)*, 2004.
- [32] A. Carzaniga and A. Wolf. Forwarding in a content-based network. In *Proceedings of the ACM SIGCOMM'03 Conference*, 2003, 2003.
- [33] M. Castro, P. Druschel, A. Kermarrec, and A. Rowstron. Scribe: A large-scale and decentralized application-level multicast infrastructure. *Journal on Selected Areas in Communication*, 2002.

- [34] C. Y. Chan, P. Felber, M. N. Garofalakis, and R. Rastogi. Efficient filtering of XML documents with XPath expressions. In *Proceedings of the 18th International Conference on Data Engineering (ICDE)*, 2002.
- [35] Y. Chawathe, S. Ratnasamy, L. Breslau, N. Lanham, and S. Shenker. Making gnutella-like P2P systems scalable. In *Proceedings of the ACM SIGCOMM 2003 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, 2003.
- [36] D. R. Cheriton and S. E. Deering. Multicast routing in datagram internetworks and extended LANs. *ACM Transactions on Computing Systems*, 8(2):85, 1990.
- [37] M. Cilia, M. Antollini, C. BornhG'Avd, and A. Buchmann. Dealing with heterogeneous data in pub/sub systems: The concept-based approach. In *Proceedings of the Third International Workshop on Distributed Event-Based Systems (DEBS)*, 2004.
- [38] I. Clarke, O. Sandberg, B. Wiley, and T. W. Hong. Freenet: a distributed anonymous information storage and retrieval system. In *International Workshop on Design Issues in Anonymity and Unobservability*, 2000.
- [39] A. Crespo and H. Garcia-Molina. Routing indices for peer-to-peer systems. In *Proceedings of the 22th International Conference on Distributed Computing Systems (ICDCS)*, 2002.
- [40] G. Cugola, E. D. Nitto, and A. Fuggetta. The jedi event-based infrastructure and its application to the development of the opss wfms. *IEEE Transactions on Software Engineering*, 2001.
- [41] A. Datta, M. Hauswirth, R. John, R. Schmidt, and K. Aberer. Range queries in trie-structured overlays. In *Proceedings of the Fifth IEEE International Conference on Peer-to-Peer Computing (ICP2PC)*, 2005.
- [42] M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf. *Computational Geometry: Algorithms and Applications*. Springer-Verlag, 1997.
- [43] Y. Diao, M. Altinel, M. J. Franklin, H. Zhang, and P. Fischer. Path sharing and predicate evaluation for high-performance XML filtering. *ACM Transactions on Database Systems*, 2003.
- [44] P. Druschel and A. Rowstron. PAST: A large-scale, persistent peer-to-peer storage utility. In *Proceedings of the Eighth IEEE Workshop on Hot Topics in Operating Systems (HotOS-VIII)*, 2001.
- [45] P. T. Eugster, P. Felber, R. Guerraoui, and A. M. Kermarrec. The many faces of publish/subscribe. *ACM Computing Surveys*, 2003.
- [46] F. Fabret, H. A. Jacobsen, F. Llirbat, J. Pereira, K. A. Ross, and D. Shasha. Filtering algorithms and implementation for very fast Publish/Subscribe systems. In *SIGMOD*, 2001.
- [47] L. Fiege and G. Muhl. Rebeca event-based electronic commerce architecture. In <http://www.gkec.informatik.tu-darmstadt.de/rebeca>, 2000.
- [48] Ganesan, Sun, and Garcia-Molina. YAPPERS: A peer-to-peer lookup service over arbitrary topology. In *Proceedings of the INFOCOM Conference on Computer Communications, joint conference of the IEEE Computer and Communications Societies*, 2003.
- [49] P. Ganesan, B. Yang, and H. G. Molina. One torus to rule them all: Multidimensional queries in P2P systems. In *Proceedings of the Seventh International Workshop on the Web and Databases (WebDB)*, 2004.
- [50] J. Gao and P. Steenkiste. An adaptive protocol for efficient support of range queries in DHT-based systems. In *Proceedings of the 12th IEEE International Conference on Network Protocols (ICNP'04)*, 2004.

- [51] B. Gedik and L. Liu. PeerCQ: a decentralized and self-configuring peer-to-peer information monitoring system. In *Proceedings of the 23th International Conference on Distributed Computing Systems (ICDCS)*, 2003.
- [52] A. Gupta, D. Agrawal, and A. E. Abbadi. Approximate range selection queries in peer-to-peer systems. In *Proceedings of the First Biennial Conference on Innovative Data Systems Research (CIDR)*, 2003.
- [53] A. Gupta, O. D. Sahin, D. Agrawal, and A. E. Abbadi. Meghdoot: Content-based publish subscribe over P2P networks. In *Proceedings of the 5th International Middleware Conference, Middleware*, 2004.
- [54] 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 *Proceedings of the First International Workshop on Peer-to-Peer Systems (IPTPS)*, 2002.
- [55] N. J. A. Harvey, M. B. Jones, S. Saroiu, M. Theimer, and A. Wolman. Skipnet: A scalable overlay network with practical locality properties. In *Proceedings of the 4th USENIX Symposium on Internet Technologies and Systems (USITS)*, 2003.
- [56] R. Huebsch, J. M. Hellerstein, N. Lanham, B. T. Loo, S. Shenker, and I. Stoica. Querying the internet with PIER. In *Proceedings of 29th International Conference on Very Large Data Bases, VLDB*, 2003.
- [57] H. V. Jagadish, B. C. Ooi, and Q. H. Vu. Baton: A balanced tree structure for peer-to-peer networks. In *Proceedings of 31st International Conference on Very Large Data Bases, VLDB*, 2005.
- [58] 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 *Proceedings of the ACM Symposium on Theory of Computing (STOC)*, 1997.
- [59] King and Saia. Choosing a random peer. In *Proceedings of the 23th ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing, PODC*, 2004.
- [60] G. Koloniari and E. Pitoura. Content-based routing of path queries in peer-to-peer systems. In *Proceedings of the 9th International Conference Advances in Database Technology, EDBT*, 2004.
- [61] G. Koloniari and E. Pitoura. Peer-to-peer management of XML data: issues and research challenges. *SIGMOD Record (ACM Special Interest Group on Management of Data)*, 2005.
- [62] Koubarakis, Skiadopoulos, and Tryfonopoulos. Logic and computational complexity for boolean information retrieval. *IEEE TKDE: IEEE Transactions on Knowledge and Data Engineering*, 18, 2006.
- [63] M. Koubarakis, T. Koutris, C. Tryfonopoulos, and P. Raftopoulou. Information alert in distributed digital libraries: The models, languages, and architecture of DIAS. In *Proceedings of the 6th European Conference on Research and Advanced Technology for Digital Libraries (ECDL)*, 2002.
- [64] B. Krishnamurthy, J. Wang, and Y. Xie. Early measurements of a cluster-based architecture for P2P systems. In *Proceedings of the First ACM SIGCOMM Internet Measurement Workshop (IMW-01)*, 2001.
- [65] J. Kubiawicz, D. Bindel, Y. Chen, P. Eaton, D. Geels, R. Gummadi, S. Rhea, H. Weatherspoon, W. Weimer, C. Wells, and B. Zhao. Oceanstore: An architecture for global-scale persistent storage. In *Proceedings of the ACM ASPLOS*, 2000.
- [66] Lv, Ratnasamy, and Shenker. Can heterogeneity make gnutella scalable? In *Proceedings of the First International Workshop on Peer-to-Peer Systems (IPTPS)*, 2002.

- [67] L. Massoulié, E. L. Merrer, A. Kermarrec, and A. J. Ganesh. Peer counting and sampling in overlay networks: random walk methods. In *Proceedings of the 25th Annual ACM Symposium on Principles of Distributed Computing PODC*, 2006.
- [68] G. Mühl, L. Fiege, F. C. Gärtner, and A. P. Buchmann. Evaluating advanced routing algorithms for content-based publish/subscribe systems. In *Proceedings of the 10th International Workshop on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems, MASCOOTS*. IEEE Computer Society, 2002.
- [69] S. W. Ng, B. C. Ooi, K. L. Tan, and A. Zhou. PeerDB: A P2P-based system for distributed data sharing. In *Proceedings of the 19th International Conference on Data Engineering (ICDE)*, 2003.
- [70] B. Nguyen, S. Abiteboul, G. Cobena, and M. Preda. Monitoring XML data on the web. In *SIGMOD*, 2001.
- [71] N. Ntarmos, T. Pitoura, and P. Triantafillou. Range query optimization leveraging peer heterogeneity. In *Proceedings of the 3rd International Workshop on Databases, Information Systems and Peer-to-Peer Computing (DBISP2P 2005)*, 2005.
- [72] N. Ntarmos, P. Triantafillou, and G. Weikum. Counting at large: Efficient cardinality estimation in internet-scale data networks. In *Proceedings of the 22nd International Conference on Data Engineering (ICDE06)*, 2006.
- [73] B. Oki, M. Pfluegl, A. Siegel, and D. Skeen. The information bus: An architecture for extensible distributed systems. *ACM Operating Systems Review*, 1993.
- [74] B. C. Ooi, Y. Shu, and K.-L. Tan. Relational data sharing in peer-based data management systems. *SIGMOD Record (ACM Special Interest Group on Management of Data)*, 2003.
- [75] V. Papadimos, D. Maier, and K. Tufte. Distributed query processing and catalogs for peer-to-peer systems. In *Proceedings of the First Biennial Conference on Innovative Data Systems Research (CIDR)*, 2003.
- [76] P. R. Pietzuch and J. Bacon. Hermes: a distributed event-based middleware architecture. In *DEBS02*, 2002.
- [77] P. R. Pietzuch and J. Bacon. Peer-to-peer overlay broker networks in an event-based middleware. In *Proceedings of the Second International Workshop on Distributed Event-Based Systems (DEBS)*, 2003.
- [78] T. Pitoura, N. Ntarmos, and P. Triantafillou. Replication, load balancing and efficient range query processing in DHTs. In *Proceedings of the 11th International Conference Advances in Database Technology, EDBT*, 2006.
- [79] T. Pitoura and P. Triantafillou. Load distribution fairness in P2P data management systems. In *Proceedings of the IEEE 23rd International Conference on Data Engineering, (ICDE 2007)*, 2007.
- [80] C. G. Plaxton, R. Rajaraman, and A. W. Richa. Accessing nearby copies of replicated objects in a distributed environment. In *ACM Symposium on Parallel Algorithms and Architectures*, 1997.
- [81] S. Ramabadran, S. Ratnasamy, J. M. Hellerstein, and S. Shenker. Brief announcement: Prefix hash tree. In *Proceedings of the 23rd Annual ACM Symposium on Principles of Distributed Computing PODC*, 2004.
- [82] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. A scalable content addressable network. In *Proceedings of the ACM SIGCOMM2001 Conference*, 2001.
- [83] S. Ratnasamy, M. Handley, R. Karp, and S. Shenker. Application-level multicast using content-addressable networks. In *3rd International Workshop of NGC*, 2001.

- [84] S. Rhea, D. Geels, T. Roscoe, and J. Kubiatowicz. Handling churn in a DHT. In *Proceedings of the 2004 USENIX Annual Technical Conference (USENIX '04)*, 2004.
- [85] A. Rowstron and P. Druschel. Pastry: Scalable and distributed object location and routing for large-scale peer-to-peer systems. In *Proceedings of the International Middleware Conference, Middleware 2001*, 2001.
- [86] B. Segall, D. Arnold, J. Boot, M. Henderson, and T. Phelps. Content based routing with elvin4. In *AUUG2K (Canberra, Australi)*, 2000.
- [87] S. Shi, G. Yang, D. Wang, J. Yu, S. Qu, and M. Chen. Making peer-to-peer keyword searching feasible using multi-level partitioning. In *Proceedings of the 3rd International Workshop on Peer-to-Peer Systems (IPTPS)*, 2004.
- [88] I. Stoica, R. Morris, D. Karger, F. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *Proceedings of the ACM SIGCOMM2001 Conference*, 2001.
- [89] D. Tam, R. Azimi, and H. Jacobsen. Building content-based publish/subscribe systems with distributed hash tables. In *Proceedings of the First International Workshop on Databases, Information Systems, and Peer-to-Peer Computing, DBISP2P*, 2003.
- [90] C. Tang and Z. Xu. pfilter: Global information filtering and dissemination using structured overlays. In *Proceedings of the FTDCS*, 2003.
- [91] W. W. Terpstra, S. Behnel, L. Fiege, A. Zeidler, and A. P. Buchmann. A peer-to-peer approach to content-based publish/subscribe. In *Proceedings of the Second International Workshop on Distributed Event-Based Systems (DEBS)*, 2003.
- [92] P. Triantafillou and I. Aekaterinidis. Publish-subscribe over structured P2P networks. In *Proceedings of the Third International Workshop on Distributed Event-Based Systems (DEBS)*, 2004.
- [93] P. Triantafillou and A. Economides. Subscription summaries for scalability and efficiency in publish/subscribe systems. In *Proceedings of the First International Workshop on Distributed Event-Based Systems (DEBS)*, 2002.
- [94] P. Triantafillou and A. Economides. Subscription summarization: A new paradigm for efficient publish/subscribe systems. In *Proceedings of the 14th International Conference on Distributed Computing Systems (ICDCS)*, 2004.
- [95] P. Triantafillou, C. Xiruhaki, M. Koubarakis, and N. Ntarmos. Towards high performance peer-to-peer content and resource sharing systems. In *Proceedings of the First Biennial Conference on Innovative Data Systems Research (CIDR)*, 2003.
- [96] C. Tryfonopoulos, S. Idreos, and M. Koubarakis. Publish/subscribe functionality in IR environments using structured overlay networks. In *Proceedings of the 26th Annual International ACM SIGIR Conference*, 2005.
- [97] C. Tryfonopoulos, M. Koubarakis, and Y. Drougas. Filtering algorithms for information retrieval models with named attributes and proximity operators. In *Proceedings of the 27th Annual International ACM SIGIR Conference*, 2004.
- [98] C. Tryfonopoulos, C. Zimmer, M. Koubarakis, and G. Weikum. Architectural alternatives for information filtering in structured overlay networks. *IEEE Internet Computing Special Issue on Dynamic Information Dissemination*, 2007.
- [99] M. Waldman, A. D. Rubin, and L. F. Cranor. Publius: A robust, tamper-evident, censorship-resistant Web publishing system. In *Proceedings of the Ninth USENIX Security Symposium*, 2000.

- [100] A. L. Wolf, A. Carzaniga, and D. S. Rosenblum. Achieving scalability and expressiveness in an internet-scale event notification service. In *Proceedings of the 19th Annual ACM Symposium on Principles of Distributed Computing PODC*, 2000.
- [101] B. Yang, P. Vinograd, and H. Garcia-Molina. Evaluating guess and non-forwarding peer-to-peer search. In *Proceedings of the 14th International Conference on Distributed Computing Systems (ICDCS)*, 2004.
- [102] G. I. Zachary, N. Khandelwal, A. Kapur, and M. Cakir. Orchestra: Rapid, collaborative sharing of dynamic data. In *Proceedings of the Second Biennial Conference on Innovative Data Systems Research (CIDR)*, 2005.
- [103] B. Y. Zhao, L. Huang, J. Stribling, S. C. Rhea, A. D. Joseph, and J. D. Kubiatowicz. Tapestry: A global-scale overlay for rapid service deployment. *IEEE Journal on Selected Areas in Communications*, 2003.
- [104] B. Y. Zhao, A. D. Joseph, and J. D. Kubiatowicz. Tapestry: An infrastructure for fault-tolerant wide-area location and routing. Technical Report UCB/CSD-01-1141, University of California at Berkley, 2001.
- [105] S. Q. Zhuang, B. Y. Zhao, A. D. Joseph, R. H. Katz, and J. D. Kubiatowicz. Bayeux: An architecture for scalable and fault-tolerant wide-area data dissemination. In *ACM NOSSDAV*, 2001.