ΣΧΕΔΙΑΣΗ & ΜΕΛΕΤΗ ΔΟΜΗΜΕΝΩΝ ΔΙΟΜΟΤΙΜΩΝ ΣΥΣΤΗΜΑΤΩΝ ΑΠΟΘΗΚΕΥΣΗΣ & ΔΙΑΧΕΙΡΙΣΗΣ ΠΟΛΥΔΙΑΣΤΑΤΩΝ ΔΕΔΟΜΕΝΩΝ

Η
ΜΕΤΑΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ ΕΞΕΙΔΙΚΕΥΣΗΣ

Υποβάλλεται στην

ορισθείσα από την Γενική Συνέλευση Ειδικής Σύνθεσης
του Τμήματος Πληροφορικής
Εξεταστική Επιτροπή

από την

Μαρία Χριστοδουλίδου

ως μέρος των Υποχρεώσεων

για τη λήψη

του

ΜΕΤΑΠΤΥΧΙΑΚΟΥ ΔΙΠΛΩΜΑΤΟΣ ΣΤΗΝ ΠΛΗΡΟΦΟΡΙΚΗ

ΜΕ ΕΞΕΙΔΙΚΕΥΣΗ ΣΤΟ ΛΟΓΙΣΜΙΚΟ

Φεβρουάριος 2008

ΑΦΙΕΡΩΣΗ

Στην οικογένειά μου

ΕΥΧΑΡΙΣΤΙΕΣ

Η διατριβή αυτή εκπονήθηκε στο Τμήμα Πληροφορικής του Πανεπιστημίου Ιωαννίνων με επιβλέπουσα την κ. Παναγιώτα Φατούρου.

Θα ήθελα, καταρχήν, να ευχαριστήσω θερμά την κ. Παναγιώτα Φατούρου, επιβλέπουσα της διατριβής μου και καθηγήτρια μου από το δεύτερο έτος εισαγωγής μου στο Τμήμα Πληροφορικής, για την υπομονή που επέδειξε και για τη συνεχή και άψογη καθοδήγησή της καθ' όλη τη διάρκεια της συνεργασίας μας.

Ακόμη, αισθάνομαι την ανάγκη να ευχαριστήσω τους γονείς μου και την αδελφή μου, που ήταν πάντα δίπλα μου με την αγάπη, την κατανόηση και την υποστήριξη που μου προσέφεραν, καθώς και με την υπομονή που επιδείκνυαν.

ΠΕΡΙΕΧΟΜΕΝΑ

ΕΥΡΕΤΗΡΙΟ ΣΧΗΜΑΤΩΝ

ΠΕΡΙΛΗΨΗ

Μαρία Χριστοδουλίδου του Θεόδωρου και της Χρυσούλας.
MSc, ΤμήμαΠληροφορικής, Πανεπιστήμιο Ιωαννίνων, Φεβρουάριος, 2008.
Σχεδίαση & Μελέτη Δομημένων Διομότιμων Συστημάτων Αποθήκευσης & Διαχείρισης Πολυδιάστατων Δεδομένων.
Επιβλέπουσα: Παναγιώτα Φατούρου

Η παρούσα διατριβή πραγματεύεται το πρόβλημα της διαχείρισης πολυδιάστατων δεδομένων σε δομημένα διομότιμα συστήματα. Αρχικά, περιγράφονται τα συστήματα του MAAN [4] και του Mercury [3], δύο από τα συστήματα που έχουν προταθεί και επιχειρούν να δώσουν λύση στο πρόβλημα. Κάθε ένα από αυτά χρησιμοποιεί διαφορετικούς τρόπους για να υποστηρίζεται η διαχείριση των δεδομένων και, κυρίως, η αποδοτική επίλυση ερωτήσεων διαστήματος. Επίσης, και τα δύο αυτά συστήματα βασίζονται στη βασική δομή του συστήματος Chord, που είναι ο δακτύλιος.

Στη συνέχεια, προτείνεται ένα νέο σύστημα, το TreeChord. Το σύστημα αυτό βασίζεται, επίσης, στη δομή του Chord [26]. Η είσοδος και η αποχώρηση κόμβων γίνεται όπως ακριβώς και στο Chord. Η διαχείριση, όμως, των δεδομένων γίνεται βάσει μίας δενδρικής δομής που χωρίζει το δακτύλιο σε ισομεγέθη τμήματα. Το σύστημα αυτό επιτρέπει την αποδοτική διαχείριση πολυδιάστατων δεδομένων, ενώ παράλληλα επιφέρει εξισορρόπηση φορτίου ανάμεσα στους συμμετέχοντες κόμβους, όποια κατανομή κι αν ακολουθούν τα δεδομένα εισόδου.

Τα συστήματα αυτά, τα οποία υλοποιήθηκαν σε έναν προσομοιωτή, μελετώνται και συγκρίνονται θεωρητικά και πειραματικά, ενώ εξάγονται χρήσιμα συμπεράσματα. Πιο συγκεκριμένα, αποδεικνύεται πειραματικά πως το σύστημα του TreeChord

υπερτερεί έναντι των άλλων δύο συστημάτων ως προς διάφορα ενδιαφέροντα μετρικά.

EXTENDED ABSTRACT IN ENGLISH

Christodoulidou Maria, T.
MSc, Computer Science Department, University of Ioannina, Greece. February, 2008.
Design & Study of Structured Peer-to-Peer Systems for Data Storage and Management
Thesis Supervisor: Panagiota Fatourou

In this dissertation, we study structured peer-to-peer systems which support multi-attribute data storage and management. Firstly, we describe the Chord system. Chord assigns a unique identifier to each node and data in the system using hashing techniques. Nodes are organized in a logical ring, called *identifier circle* or *Chord ring*, according to their identifiers, while data with identifier k is stored at the node with the higher identifier that succeeds k. Each node in Chord stores information about a few other nodes on the ring, besides its immediate predecessor and successor, in order to efficiently find data. In particular, each node stores information about other nodes at exponentially increasing distances from itself on the ring or about their successors if these nodes do not exist. This information is stored at a table called *finger table*. Chord supports one-dimensional data storage and management. However, hashing destroys the order of data. Consequently, Chord cannot support range queries.

The MAAN system, which is described in Chapter 5, extends Chord to support multi-attribute data storage and management, as well as range queries. MAAN organizes nodes the way Chord does. In addition, nodes store the same routing information that nodes in Chord maintain. However, MAAN uses locality preserving hashing in order to allocate data to nodes participating in the system and a multi-dimensional query resolution mechanism that employs the Chord routing algorithm.

In Chapter 6, we present the Mercury system, which supports multi-attribute data storage and management, as well as range queries, using the basic Chord topology. Mercury organizes nodes in groups, called hubs, by creating a routing hub for each data attribute. Each routing hub is a logical collection of nodes in the system, organized in a ring. A range is assigned randomly to each node participating in the system. Each node in Mercury maintains information about a few other randomly chosen nodes in the system. This information allows nodes to efficiently find data stored in the system. Mercury does not use hashing techniques for data allocation.

In Chapter 7, we propose a new system, called TreeChord, that supports multi-attribute data storage and management, as well as range queries. TreeChord also uses the Chord ring topology. Node management in Mercury is similar to node management in Chord. Additionally, nodes maintain the same routing information as Chord nodes do. However, data allocation is based on a tree structure, which divides the Chord identifier space in equal pieces. TreeChord supports multi-attribute data storage and management in an efficient and scalable manner, while providing load balancing, regardless of the distribution followed by the data inserted.

In Chapters 8 and 9, we provide information on the simulator used for the implementation of the systems described. Lastly, in Chapter 10, these systems are experimentally evaluated and compared. More precisely, it is experimentally shown that TreeChord outperforms MAAN and Mercury with respect to some interesting metrics.

ΚΕΦΑΛΑΙΟ 1. ΕΙΣΑΓΩΓΗ

1.1. Διομότιμα Συστήματα

Τα τελευταία χρόνια, η ανάπτυξη που γνώρισε το Διαδίκτυο ήταν μεγάλη. Πολλές είναι οι εφαρμογές που αναπτύχθηκαν. Η πιο πρόσφατη είναι τα Διομότιμα Συστήματα (*Peer-to-peer* ή *P2P*). Πρόκειται για κατανεμημένα συστήματα χωρίς ιεραρχική οργάνωση ή κεντρικοποιημένο έλεγχο στα οποία οι συμμετέχοντας κόμβοι (*peers*) δρουν και ως *εξυπηρέτες* (*servers*) και ως *εξυπηρετούμενοι* (*clients*) και επιτρέπουν το διαμοιρασμό δεδομένων.

Όπως είναι γνωστό, στο διαδίκτυο αρχικά χρησιμοποιήθηκε το αρχιτεκτονικό μοντέλο του *εξυπηρετούμενου – εξυπηρέτη (client – server)*. Στο μοντέλο αυτό, τα δεδομένα αποθηκεύονται στους εξυπηρέτες. Για την εύρεση κάποιας πληροφορίας ο εξυπηρετούμενος εντοπίζει τον εξυπηρέτη που κατέχει την επιθυμητή πληροφορία και κάνει μια αίτηση για να την αποκτήσει. Κατόπιν, ο εξυπηρέτης ανταποκρίνεται παρέχοντας, συνήθως, την ίδια την πληροφορία. Τις περισσότερες φορές υπάρχει μικρός αριθμός εξυπηρετών που παρέχουν υπηρεσίες στους υπόλοιπους χρήστες. Οι εξυπηρέτες αυτοί είναι συνήθως διαθέσιμοι για μεγάλο χρονικό διάστημα, ενώ οι εξυπηρετούμενοι παραμένουν συνδεδεμένοι για λίγο.

Στα Διομότιμα Συστήματα, η παραπάνω αρχιτεκτονική εγκαταλείπεται. Οι συμμετέχοντες σε αυτά δρουν και ως εξυπηρετούμενοι και ως εξυπηρέτες, δηλαδή αποθηκεύουν, ανακτούν και

συνεισφέρουν στην εύρεση δεδομένων που αναζητούνται σε ένα δυναμικό σύστημα. Επίσης, η οργάνωση και διαχείριση κόμβων και δεδομένων γίνεται με διαφορετικό τρόπο.
Το μοντέλο αυτό, όμως, γεννά δύο σημαντικά ζητήματα: πώς θα οργανωθούν οι συμμετέχοντες κόμβοι και ποιος μηχανισμός αναζήτησης θα εφαρμοστεί προκειμένου να είναι δυνατή η αποδοτική εύρεση των ζητούμενων δεδομένων. Στα Διομότιμα Συστήματα προτείνονται δύο λύσεις: η κεντρικοποιημένη και η αποκεντρικοποιημένη αναζήτηση.

Τα κεντρικοποιημένα συστήματα (Napster [13]) είναι τα πρώτα συστήματα που εμφανίζονται. Σε αυτά, ένας κεντρικός εξυπηρέτης χρησιμοποιείται για την καταγραφή και εύρεση των δεδομένων. Αποδεικνύονται, όμως, ιδιαίτερα ευάλωτα, καθώς υπάρχει μοναδικό σημείο αποτυχίας, του οποίου η κατάρρευση μπορεί να οδηγήσει σε κατάρρευση ολόκληρου του συστήματος. Επίσης, η απόδοσή τους είναι χαμηλή όταν το πλήθος των κόμβων αυξηθεί πολύ, δηλαδή δεν εμφανίζουν καλές ιδιότητες κλιμάκωσης.

Τα μειονεκτήματα αυτά έρχονται να καλύψουν τα αποκεντρικοποιημένα συστήματα, όπως είναι τα αδόμητα δίκτυα Gnutella [2, 12, 24], Freenet [6, 7, 11], καθώς και τα δομημένα Chord [26], CAN [23] και άλλα. Φυσικά, σε κάθε ένα από αυτά χρησιμοποιούνται διαφορετικές τεχνικές οργάνωσης και διαχείρισης των κόμβων και των δεδομένων.

Τα αποκεντρικοποιημένα συστήματα, λοιπόν, διακρίνονται σε δύο κατηγορίες, τα *αδόμητα* (*unstructured*) και τα *δομημένα* (*structured*).

Στα αδόμητα συστήματα, δεν υπάρχει καμία οργάνωση των κόμβων που συμμετέχουν. Οι κόμβοι δε γνωρίζουν ποια είναι η τοπολογία του συστήματος, ενώ ο εντοπισμός των δεδομένων συνήθως βασίζεται σε τυφλές μεθόδους αναζήτησης, όπως είναι η πλημμύρα (*flooding*), και όχι στην τοπολογία του συστήματος.

Στα δομημένα συστήματα, η οργάνωση των συμμετεχόντων κόμβων και η διαχείριση των δεδομένων γίνεται βάσει κάποιας τοπολογίας, έτσι ώστε να πραγματοποιούνται αποδοτικά οι αναζητήσεις. Πολλά από αυτά τα συστήματα χρησιμοποιούν τεχνικές κατακερματισμού για την οργάνωση κόμβων και δεδομένων (*DHT-based* συστήματα), όπως για παράδειγμα το Chord, το CAN, το Pastry [25], το Symphony [21], κ.ά. Στα συστήματα αυτά, ανατίθενται μοναδικά αναγνωριστικά σε κόμβους και δεδομένα μέσω τεχνικών κατακερματισμού. Βάσει

αυτών των αναγνωριστικών, τα δεδομένα ανατίθενται στους κόμβους οι οποίοι οργανώνονται σε κάποια τοπολογία (π.χ. έναν δακτύλιο). Κάθε κόμβος διατηρεί πληροφορίες για ένα λογικό πλήθος άλλων κόμβων του συστήματος (πληροφορίες δρομολόγησης). Η αναζήτηση δεδομένων γίνεται αποδοτικά μέσω μιας διαδικασίας δρομολόγησης που βασίζεται στην τοπολογία που ορίζουν οι κόμβοι. Επιπλέον, η δομή των συστημάτων επιτρέπει την εξαγωγή θεωρητικών συμπερασμάτων για την απόδοσή τους. Τα συστήματα αυτά, όμως, δεν έχουν την πρόσθετη λειτουργικότητα που απαιτείται από τις σύγχρονες εφαρμογές και, κυρίως, δεν επιτρέπουν τη διαχείριση πολυδιάστατων δεδομένων, καθώς, επίσης, και την επίλυση ερωτήσεων διαστήματος. Αυτό οφείλεται στις τεχνικές κατακερματισμού που εφαρμόζονται, οι οποίες καταστρέφουν τη διάταξη των δεδομένων. Για να επιλύσουν το πρόβλημα αυτό, τα συστήματα χρησιμοποιούν άλλες τεχνικές. Πιο συγκεκριμένα, αρκετές προσπάθειες έχουν γίνει για την προσαρμογή του συστήματος του Chord, έτσι ώστε να είναι δυνατή η επίλυση ερωτήσεων διαστήματος. Κάποια συστήματα χρησιμοποιούν συναρτήσεις κατακερματισμού που διατηρούν τη διάταξη (*locality preserving hash functions*) για την κατανομή των δεδομένων στους κόμβους (MAAN [4]), ενώ άλλα αναθέτουν διαστήματα τιμών στους κόμβους που συμμετέχουν και κατανέμουν τα δεδομένα σε αυτούς βάσει των πραγματικών τους κλειδιών (Mercury [3]).

1.2. Στόχοι της Διατριβής

Στη διατριβή αυτή παρουσιάζονται κάποια διομότιμα συστήματα που έχουν προταθεί, τα οποία επιτρέπουν τη διαχείριση πολυδιάστατων δεδομένων, καθώς, επίσης, και την αποδοτική επίλυση ερωτήσεων διαστήματος. Τα συστήματα αυτά χρησιμοποιούν τη βασική δομή του Chord, που είναι ο δακτύλιος, και την επεκτείνουν έτσι ώστε να αποκτήσουν πρόσθετη λειτουργικότητα. Επιπλέον, προτείνεται ένα ακόμη σύστημα, το TreeChord, το οποίο βασίζεται, επίσης, στο Chord και επιτρέπει τη διαχείριση πολυδιάστατων δεδομένων και την αποδοτική επίλυση ερωτήσεων διαστήματος. Όλα αυτά τα συστήματα μελετώνται και συγκρίνονται θεωρητικά και πειραματικά μεταξύ τους ως προς τα χαρακτηριστικά και την απόδοση. Στόχος της θεωρητικής μελέτης είναι να αναλυθεί η συμπεριφορά τους, ενώ μέσω της πειραματικής μελέτης επιβεβαιώνονται τα θεωρητικά αποτελέσματα που παρουσιάζονται.

1.3. Δομή της Διατριβής

Η διατριβή αυτή περιέχει 10 κεφάλαια. Στο Κεφάλαιο 2, παρουσιάζονται συνοπτικά προηγούμενα ερευνητικά αποτελέσματα και διομότιμα συστήματα που έχουν προταθεί στο παρελθόν και σχετίζονται με την παρούσα εργασία. Στο Κεφάλαιο 3, εισάγονται κάποιες βασικές έννοιες για τα διομότιμα συστήματα και τον τρόπο που λειτουργούν. Στο Κεφάλαιο 4, περιγράφεται το σύστημα του Chord, ένα διομότιμο σύστημα που βασίζεται στη χρήση κατακερματισμού και επιτρέπει τη διαχείριση μονοδιάστατων δεδομένων και την επίλυση απλών ερωτήσεων. Στη βασική δομή του Chord, που είναι ο δακτύλιος, βασίζονται όλα τα επόμενα συστήματα που περιγράφονται. Στο Κεφάλαιο 5, περιγράφεται το MAAN, ένα σύστημα βασισμένο στη δομή του Chord, που επιτρέπει τη διαχείριση πολυδιάστατων δεδομένων και την αποδοτική επίλυση ερωτήσεων διαστήματος χρησιμοποιώντας συναρτήσεις κατακερματισμού που διατηρούν τη διάταξη. Στο Κεφάλαιο 6, περιγράφεται το Mercury. Πρόκειται για ένα ακόμη σύστημα που επιτρέπει τη διαχείριση πολυδιάστατων δεδομένων και την αποδοτική επίλυση ερωτήσεων διαστήματος. Στο Mercury δε χρησιμοποιούνται τεχνικές κατακερματισμού για την οργάνωση κόμβων και δεδομένων, αλλά ανατίθενται διαστήματα στους συμμετέχοντες κόμβους και τα δεδομένα κατανέμονται σε αυτούς βάσει των πραγματικών τους τιμών. Στο Κεφάλαιο 7, παρουσιάζεται το TreeChord. Το σύστημα που προτείνεται χρησιμοποιεί τη δομή του Chord για την οργάνωση των κόμβων. Η κατανομή των δεδομένων, όμως, στους κόμβους του συστήματος γίνεται βάσει μίας δενδρικής δομής που χωρίζει το δακτύλιο σε ισομεγέθη τμήματα. Στη συνέχεια, στο Κεφάλαιο 8, περιγράφεται συνοπτικά ο προσομοιωτής που χρησιμοποιήθηκε για την πειραματική μελέτη των συστημάτων που παρουσιάστηκαν, ενώ το Κεφάλαιο 9 αναφέρεται στις ίδιες τις υλοποιήσεις. Το Κεφάλαιο 10 εστιάζει στη θεωρητική και πειραματική μελέτη και σύγκριση των συστημάτων που υλοποιήθηκαν. Τέλος, το Κεφάλαιο 11 περιλαμβάνει τον επίλογο της παρούσας διατριβής, όπου εξάγονται κάποια συμπεράσματα και συζητούνται τα ανοικτά προβλήματα.

ΚΕΦΑΛΑΙΟ 2. ΣΧΕΤΙΚΕΣ ΕΡΓΑΣΙΕΣ

2.1. Αδόμητα Διομότιμα Συστήματα

Όπως προαναφέρθηκε, τα πρώτα διομότιμα συστήματα που προτάθηκαν ήταν αδόμητα. Στα συστήματα αυτά, δεν υπάρχει κάποια οργάνωση των κόμβων που συμμετέχουν βάσει της οποίας να γίνεται η κατανομή των δεδομένων.

2.1.1. Κεντρικοποιημένα Συστήματα

Το πρώτο αδόμητο διομότιμο σύστημα που εμφανίστηκε ήταν το Napster, το οποίο δημιουργήθηκε αρχικά για το διαμοιρασμό αρχείων .mp3. Το Napster παρουσίαζε κεντρικοποιημένη αρχιτεκτονική. Η εύρεση δεδομένων γινόταν μέσω ενός κεντρικού καταλόγου που διατηρούνταν σε έναν κεντρικό εξυπηρέτη και στον οποίο καταγράφονταν όλα τα δεδομένα που υπήρχαν στο σύστημα. Το βασικότερο μειονέκτημα του Napster ήταν ο κεντρικοποιημένος χαρακτήρας του, καθώς οδηγούσε στην ύπαρξη ενός κεντρικού σημείου αποτυχίας, του οποίου η κατάρρευση θα μπορούσε να οδηγήσει σε κατάρρευση ολόκληρου του συστήματος. Για τους λόγους αυτούς, η αρχιτεκτονική του Napster γρήγορα εγκαταλείφθηκε.

2.1.2. Μη Κεντρικοποιημένα Συστήματα

Τα μειονεκτήματα του κεντρικοποιημένου Napster έρχονται να καλύψουν τα αποκεντρικοποιημένα αδόμητα συστήματα (Gnutella, FreeNet). Στα συστήματα αυτά δεν υπάρχει κεντρικός κατάλογος, επομένως ούτε μοναδικό σημείο αποτυχίας, αλλά ούτε και αυστηρή δομή. Οι κόμβοι εισέρχονται με αυθαίρετο τρόπο στο σύστημα, και δε γίνεται προσπάθεια να διατηρηθεί κάποια συγκεκριμένη τοπολογία. Η βασική μέθοδος αναζήτησης σε τέτοιου είδους συστήματα είναι η πλημμύρα (*flooding*). Κατά την αναζήτηση με τη μέθοδο της πλημμύρας, μία ερώτηση προωθείται από έναν κόμβο σε όλους τους κόμβους εντός κάποιας ακτίνας, η οποία ορίζεται στην ερώτηση. Φυσικά, δεν υπάρχουν εγγυήσεις για το απαιτούμενο πλήθος βημάτων, ούτε και είναι σίγουρη η εύρεση ενός ζητούμενου δεδομένου που έχει πράγματι εισαχθεί στο σύστημα.

2.2. Δομημένα Διομότιμα Συστήματα

Από την άλλη, στα δομημένα συστήματα, η οργάνωση των συμμετεχόντων κόμβων και η διαχείριση των δεδομένων γίνεται βάσει κάποιας τοπολογίας. Αυτό επιτρέπει όχι μόνο την αποδοτική αναζήτηση δεδομένων, αλλά και την εξαγωγή θεωρητικών συμπερασμάτων για την απόδοση των συστημάτων. Πολλά από τα συστήματα της κατηγορίας αυτής βασίζονται σε τεχνικές κατακερματισμού για την οργάνωση κόμβων και δεδομένων στην οριζόμενη τοπολογία (*Distributed Hash Table–based* ή *DHT-based* συστήματα). Στα DHT-based συστήματα (Chord, CAN, Pastry, Symphony), ανατίθενται μοναδικά αναγνωριστικά σε κόμβους και δεδομένα μέσω τεχνικών κατακερματισμού. Βάσει αυτών των αναγνωριστικών, τα δεδομένα ανατίθενται στους κόμβους οι οποίοι οργανώνονται σε κάποια τοπολογία (π.χ. έναν δακτύλιο ή ένα πλέγμα). Η αναζήτηση δεδομένων γίνεται αποδοτικά μέσω μιας διαδικασίας δρομολόγησης, η οποία βασίζεται στην τοπολογία που ορίζουν οι κόμβοι και εκμεταλλεύεται πληροφορίες δρομολόγησης που διατηρούν οι κόμβοι για ένα λογικό πλήθος άλλων κόμβων του συστήματος. Οι πληροφορίες αυτές μπορεί να είναι χορδές που κόβουν το δακτύλιο (Chord) ή τυχαίες ακμές πάνω σε αυτόν (Symphony). Για παράδειγμα, στο CAN, οι κόμβοι οργανώνονται σε ένα χώρο καρτεσιανών συντεταγμένων D διαστάσεων (*d-torus*). Ο χώρος αυτός διαχωρίζεται σε τμήματα, που ονομάζονται ζώνες. Κάθε κόμβος εισέρχεται σε τυχαίο σημείο του χώρου κατά την είσοδό του στο σύστημα και αναλαμβάνει μία ζώνη. Επιπλέον, κάθε κόμβος διατηρεί πληροφορίες για άλλους κόμβους με των οποίων τις ζώνες γειτονεύει η ζώνη του. Οι κόμβοι επιλύουν ερωτήσεις δρομολογώντας τις από ένα αρχικό σε

ένα τελικό σημείο στο χώρο συντεταγμένων. Από την άλλη, στο σύστημα του Chord, όλοι οι κόμβοι τοποθετούνται ομοιόμορφα σε ένα δακτύλιο, ενώ τα δεδομένα του συστήματος κατανέμονται ομοιόμορφα στους υπάρχοντες κόμβους. Οι κόμβοι διατάσσονται στο δακτύλιο βάσει του κλειδιού τους, ενώ ένα δεδομένο διατηρείται στον κόμβο με το κοντινότερο μεγαλύτερο κλειδί από το κλειδί του. Προκειμένου να πραγματοποιούνται αποδοτικά οι αναζητήσεις δεδομένων στο σύστημα, κάθε κόμβος στο Chord διατηρεί δείκτες και σε μερικούς ακόμη κόμβους, εκτός από τον προκάτοχο και το διάδοχό του στο δακτύλιο. Οι δείκτες αυτοί, που ονομάζονται fingers, μοιάζουν με χορδές που κόβουν το δακτύλιο και επιτρέπουν την αποδοτική επίλυση απλών ερωτήσεων.

2.3. Διομότιμα Συστήματα που Υποστηρίζουν Ερωτήσεις Διαστήματος

Οι σύγχρονες εφαρμογές, όμως, απαιτούν τις περισσότερες φορές να υποστηρίζεται η διαχείριση πολυδιάστατων δεδομένων και η επεξεργασία πιο πολύπλοκων ερωτήσεων, όπως είναι οι ερωτήσεις διαστήματος. Μεγάλη έρευνα έχει γίνει για το σχεδιασμό δομημένων συστημάτων που επιτρέπουν τη διαχείριση πολυδιάστατων δεδομένων και την επίλυση ερωτήσεων διαστήματος. Κάποια από αυτά χρησιμοποιούν την τοπολογία του Chord [π.χ. 3, 4, 22], άλλα χρησιμοποιούν την τοπολογία του CAN [π.χ. 8, 21], ενώ άλλα βασίζονται σε δενδρικές δομές [π.χ. 1, 14, 15, 16, 27].

Για παράδειγμα, το σύστημα του MAAN οργανώνει τους συμμετέχοντες κόμβους όπως ακριβώς και το Chord. Η κατανομή των δεδομένων, όμως, στους κόμβους που συμμετέχουν στο σύστημα γίνεται μέσω τεχνικών κατακερματισμού που διατηρούν τη διάταξη. Από την άλλη, το σύστημα του Mercury χρησιμοποιεί, επίσης, τη βασική δομή του Chord, που είναι ο δακτύλιος. Οι κόμβοι του συστήματος, όμως, οργανώνονται σε περισσότερους του ενός δακτυλίους χωρίς να εφαρμόζεται κάποια τεχνική κατακερματισμού. Κατά την είσοδό τους στο σύστημα, οι κόμβοι αναλαμβάνουν την ευθύνη ενός εύρους τιμών και τα δεδομένα κατανέμονται σε αυτούς βάσει των πραγματικών τους τιμών. Επιπλέον, οι πληροφορίες δρομολόγησης που διατηρούν οι κόμβοι κατασκευάζονται με διαφορετικό τρόπο από αυτόν που χρησιμοποιείται στο Chord.

Από την άλλη, κάποια συστήματα που επιτρέπουν την επίλυση ερωτήσεων διαστήματος βασίζονται στη δομή του CAN. Για παράδειγμα, το σύστημα του MURK [8] διαχειρίζεται

πολυδιάστατα δεδομένα και ερωτήσεις διαστήματος διαμερίζοντας κατάλληλα το χώρο δεδομένων, όπως ακριβώς διαμερίζουν και οι διαστάσεις το χώρο συντεταγμένων στο CAN. Από την άλλη, το σύστημα που προτείνεται στην εργασία [21] χρησιμοποιεί ένα χώρο 2D διαστάσεων (όπως το CAN χρησιμοποιεί χώρο D διαστάσεων). Το σύστημα αυτό εφαρμόζει τεχνικές κατακερματισμού σε διαστήματα τιμών, όχι σε απλές τιμές. Καθώς, όμως η κατανομή διαστημάτων τιμών στους κόμβους δεν είναι απαραίτητα ομοιόμορφη, απαιτείται κάποια τεχνική εξισορρόπησης φορτίου. Οι λειτουργίες που επιτελούνται στο σύστημα πραγματοποιούνται με τρόπο ανάλογο με αυτόν που εφαρμόζεται στο CAN.

Σε άλλα συστήματα, οι κόμβοι οργανώνονται βάσει κάποιας δενδρικής δομής αναλαμβάνοντας την ευθύνη ενός εύρους τιμών του συστήματος. Με τον τρόπο αυτό, επιτρέπεται η επίλυση ερωτήσεων διαστήματος. Για παράδειγμα, στο BATON [14] οι κόμβοι οργανώνονται σε ένα εξισορροπημένο δυαδικό δένδρο αναζήτησης. Κάθε κόμβος γνωρίζει ποια είναι η ακριβής του θέση στο δένδρο, δηλαδή σε ποιο επίπεδο του δένδρου ανήκει και σε ποιο σημείο αυτού. Γνωρίζει, επίσης, ποιος είναι ο πατρικός του κόμβος στο δένδρο, ποιοι είναι οι θυγατρικοί του, καθώς, και κάποιους ακόμη κόμβους στο ίδιο επίπεδο. Κάθε κόμβος του συστήματος είναι υπεύθυνος για ένα συγκεκριμένο εύρος τιμών. Μάλιστα, το εύρος αυτό βρίσκεται δεξιά του εύρους για το οποίο είναι υπεύθυνο το αριστερό του παιδί και αριστερά του εύρους για το οποίο είναι υπεύθυνο το δεξί του παιδί, όπως συμβαίνει και σε ένα απλό δυαδικό δένδρο αναζήτησης. Το γεγονός πως κατά την είσοδο νέων κόμβων στο σύστημα και την αποχώρηση κόμβων από αυτό το δένδρο παραμένει εξισορροπημένο επιτρέπει να επιλύονται αποδοτικά απλές ερωτήσεις, αλλά και ερωτήσεις διαστήματος. Το BATON* [15] αποτελεί γενίκευση του BATON.

Το σύνολο των συστημάτων που επιτρέπουν τη διαχείριση πολυδιάστατων δεδομένων είναι πολύ μεγάλο. Τα πλεονεκτήματα και μειονεκτήματα που εμφανίζει κάθε ένα από αυτά καθορίζονται από τον τρόπο σχεδίασης και υλοποίησης που ακολουθείται. Τα συστήματα που μελετήθηκαν στην παρούσα διατριβή συμπεριλαμβάνονται στα συστήματα που επιτρέπουν τη διαχείριση πολυδιάστατων δεδομένων και την επίλυση ερωτήσεων διαστήματος και η πειραματική τους μελέτη που θα ακολουθήσει θα αναδείξει τα πλεονεκτήματα και τα μειονεκτήματά τους.

ΚΕΦΑΛΑΙΟ 3. ΜΟΝΤΕΛΟ

Στο Κεφάλαιο αυτό περιγράφονται κάποιες βασικές έννοιες που χρησιμοποιούνται στη συνέχεια για την περιγραφή των συστημάτων που υλοποιήθηκαν, καθώς, επίσης, και για την πειραματική τους μελέτη.

3.1. Βασικές Έννοιες

Τα συστήματα τα οποία μελετάμε είναι δυναμικά και έτσι κόμβοι μπορούν να εισέρχονται και να αποχωρούν από το σύστημα οποιαδήποτε χρονική στιγμή. Θεωρούμε ότι σε κάθε κόμβο που συμμετέχει στο σύστημα, αντιστοιχίζεται ένα μοναδικό αναγνωριστικό (*identifier*). Το αναγνωριστικό αυτό, που ονομάζεται και κλειδί (*key*), ανατίθεται σε έναν κόμβο κατά την είσοδό του στο σύστημα και αποτελείται από m bits. Επομένως, τα κλειδιά των κόμβων μπορούν να ανήκουν στο διάστημα τιμών $\left[0, 2^m\right)$ ή $\left[0, 2^m - 1\right]$. Το διάστημα αυτό ονομάζεται χώρος αναγνωριστικών (*identifier space*) του συστήματος. Για τη θεωρητική μελέτη ενός συστήματος, θεωρούμε ότι N είναι το πλήθος των κόμβων που συμμετέχουν κάποια χρονική στιγμή σε αυτό.

Ένα *χαρακτηριστικό* (*attribute*) ή μία *διάσταση* (*dimension*) είναι μία από τις παραμέτρους που χαρακτηρίζουν κάθε δεδομένο του συστήματος. Για παράδειγμα, αν θέλαμε να διατηρήσουμε πληροφορίες για τοποθεσίες, όπως είναι το γεωγραφικό μήκος και πλάτος των τοποθεσιών, θα χρειαζόμασταν να αποθηκεύσουμε δεδομένα που χαρακτηρίζονται από δύο παραμέτρους, το γεωγραφικό μήκος και το γεωγραφικό πλάτος. Θεωρούμε, λοιπόν, ότι κάθε

δεδομένο d που εισάγεται σε ένα σύστημα είναι της μορφής $v=(v_1,v_2,...,v_{M-1},v_M)$, όπου v_i η τιμή του χαρακτηριστικού i του δεδομένου, $1 \le i \le M$. Το σύνολο $A=\{a_1,a_2,...,a_M\}$ των χαρακτηριστικών των δεδομένων του συστήματος λέμε ότι έχει μέγεθος M, δηλαδή το M αποτελεί το πλήθος των διαστάσεων των δεδομένων. Επίσης, m_{a_i} και M_{a_i} η ελάχιστη και η μέγιστη τιμή που μπορεί να λάβει το χαρακτηριστικό a_i. Αν ένα δεδομένο αποτελείται από ένα και μόνο χαρακτηριστικό, τότε ονομάζεται *μονοδιάστατο*. Στην αντίθετη περίπτωση, το δεδομένο θεωρείται *πολυδιάστατο* (*multi-attribute* ή *multi-dimesional*). Όπως και στους κόμβους κάθε συστήματος, σε κάθε χαρακτηριστικό ενός δεδομένου ανατίθεται ένα μοναδικό κλειδί που αποτελείται από m bits. Επομένως, τα κλειδιά των δεδομένων μπορούν να ανήκουν, επίσης, στο διάστημα τιμών $[0,2^m)$ ή $[0,2^m-1]$. Τα αναγνωριστικά που ανατίθενται τόσο στους κόμβους, όσο και στα δεδομένα ενός συστήματος είναι αυτά που επιτρέπουν σε κάποια συστήματα τη διαχείρισή τους. Σε κάποια άλλα, χρησιμοποιούνται οι ίδιες οι τιμές των δεδομένων και όχι τα κλειδιά που τους ανατίθενται για τη διαχείρισή τους. Επίσης, θεωρούμε ότι σε κάθε διάσταση των δεδομένων ενός συστήματος ορίζεται μία διάταξη (για να είναι δυνατός ο ορισμός ερωτήσεων διαστήματος). Τη διάταξη αυτή τη συμβολίζουμε με $\le$ (ή $\ge$).

Όπως προαναφέρθηκε, κάθε σύστημα που μελετάται επιτρέπει τη διαχείριση δεδομένων, δηλαδή την εισαγωγή και την αναζήτηση τους στο σύστημα. Προκειμένου, όμως, τα δεδομένα να διατηρούν τη μεταξύ τους διάταξη, η εισαγωγή δεδομένων στο σύστημα γίνεται είτε βάσει των κλειδιών που τους ανατίθενται μέσω τεχνικών κατακερματισμού, είτε βάσει των πραγματικών τους τιμών. Αυτό εξαρτάται από την ίδια την οργάνωση του εκάστοτε συστήματος.

Σε ένα σύστημα μπορούν να πραγματοποιούνται και αναζητήσεις δεδομένων υπό τη μορφή ερωτήσεων. Μία ερώτηση μπορεί να είναι απλή ή πολύπλοκη. Μία απλή ερώτηση είναι της μορφής $q=(d_1,d_2,...,d_{M-1},d_M)$, όπου d_i η τιμή του χαρακτηριστικού i του δεδομένου, $1 \le i \le M$. Με άλλα λόγια, σε μία απλή ερώτηση, προσδιορίζεται ακριβώς η τιμή κάθε ενός από τα χαρακτηριστικά του ζητούμενου δεδομένου. Για παράδειγμα, μπορούμε να αναζητήσουμε αν υπάρχει κάποια πόλη με συγκεκριμένο γεωγραφικό μήκος και πλάτος. Από την άλλη, μία ερώτηση διαστήματος είναι της μορφής

$q=\left[(d_{1l},d_{1u}),(d_{2l},d_{2u}),...,(d_{M-1l},d_{M-1u}),(d_{Ml},d_{Mu})\right]$, όπου (d_{il},d_{iu}) είναι το διάστημα τιμών στο οποίο επιθυμούμε να ανήκουν οι τιμές του χαρακτηριστικού i των δεδομένων, $1\le i\le M$. Με άλλα λόγια, η ερώτηση q αναζητά τα δεδομένα $v=(v_1,v_2,...,v_M)$ των οποίων οι τιμές κάθε χαρακτηριστικού i ικανοποιούν τη συνθήκη $v_{il}\le v_i\le v_{iu}$, $1\le i\le M$. Για παράδειγμα, μπορούμε να αναζητήσουμε όλες τις πόλεις των οποίων οι γεωγραφικές συντεταγμένες τις τοποθετούν σε συγκεκριμένη περιοχή στον κόσμο. Είναι δυνατόν να μην ορίζεται το κάτω ή το άνω όριο της συνθήκης για κάποιο χαρακτηριστικό ή ακόμη και ολόκληρη η συνθήκη για κάποιο χαρακτηριστικό. Έστω A_q το σύνολο των χαρακτηριστικών για τα οποία ορίζεται μία συνθήκη στην ερώτηση διαστήματος q.

Έστω, λοιπόν, μία ερώτηση διαστήματος της μορφής $q=\left[(d_{1l},d_{1u}),...,(d_{Ml},d_{Mu})\right]$. Η *επιλεκτικότητα* (*selectivity*) s_i της ερώτησης για τη διάσταση i ορίζεται ως ο λόγος του εύρους που καλύπτει η ερώτηση προς το συνολικό εύρος τιμών του χώρου αναγνωριστικών, που είναι 2^m, δηλαδή $s_i=\frac{d_{il}-d_{iu}}{2^m}$, $1\le i\le M$.

3.2. Παράμετροι Απόδοσης

Όταν μελετούμε ένα σύστημα, εξετάζουμε κάποια χαρακτηριστικά του, που μας επιτρέπουν να διαπιστώσουμε κατά πόσο είναι αποδοτικό ή όχι. Οι κόμβοι που συμμετέχουν σε ένα σύστημα διατηρούν μέρος από τα εισαγόμενα δεδομένα και συμμετέχουν στην επίλυση των ερωτήσεων διαστήματος που πραγματοποιούνται. Το πλήθος των δεδομένων που διατηρεί ένας κόμβος, το πλήθος των ερωτήσεων διαστήματος στων οποίων την επίλυση συμμετέχει εξετάζοντας τοπικά τα δεδομένα που διατηρεί, καθώς, επίσης, και το μέσο πλήθος δεδομένων με τα οποία απαντά σε μία ερώτηση διαστήματος καθορίζουν το φορτίο που φέρει ο κόμβος στο σύστημα και αποτελούν παραμέτρους που εξετάζονται κατά την πειραματική μελέτη ενός συστήματος.

Επιπλέον, θεωρούμε ως μέσο φορτίο αποθηκευμένων δεδομένων $avgLoad_d$ του συστήματος το λόγο του συνολικού πλήθους δεδομένων που έχουν εισαχθεί στο σύστημα προς το συνολικό πλήθος κόμβων που συμμετέχουν σε αυτό. Αν n είναι ένας κόμβος του

συστήματος, n_d είναι το πλήθος των δεδομένων που διατηρεί και $k \geq 0$ μία σταθερά, τότε αν $n_d > (1+k) \times avgLoad_d$, θεωρούμε ότι ο κόμβος είναι υπερφορτωμένος (*overloaded*), ενώ αν $n_d < (1-k) \times avgLoad_d$, θεωρούμε ότι ο κόμβος είναι υποφορτωμένος (*underloaded*). Στα συστήματα που μελετούμε εξετάζουμε το ποσοστό των υπερφορτωμένων ή υποφορτωμένων κόμβων του συστήματος, δηλαδή το ποσοστό των μη εξισορροπημένων κόμβων του συστήματος, για διαφορετικές τιμές του k.

Ακόμη, θεωρούμε ως μέσο φορτίο διαχειριζόμενων ερωτήσεων διαστήματος $avgLoad_q$ του συστήματος το λόγο του συνολικού πλήθους ερωτήσεων διαστήματος στων οποίων την επίλυση συμμετέχουν οι κόμβοι του συστήματος προς το συνολικό πλήθος κόμβων που συμμετέχουν σε αυτό. Αν n είναι ένας κόμβος του συστήματος, n_q είναι το πλήθος των ερωτήσεων διαστήματος στων οποίων την επίλυση συμμετέχει και $k \geq 0$ μία σταθερά, τότε αν $n_d > (1+k) \times avgLoad_q$, θεωρούμε ότι ο κόμβος είναι υπερφορτωμένος (*overloaded*), ενώ αν $n_d < (1-k) \times avgLoad_q$, θεωρούμε ότι ο κόμβος είναι υποφορτωμένος (*underloaded*). Στα συστήματα που μελετούμε εξετάζουμε το ποσοστό των υπερφορτωμένων ή υποφορτωμένων κόμβων του συστήματος, δηλαδή το ποσοστό των μη εξισορροπημένων κόμβων του συστήματος, για διαφορετικές τιμές του k.

ΚΕΦΑΛΑΙΟ 4. ΤΟ ΣΥΣΤΗΜΑ ΤΟΥ CHORD

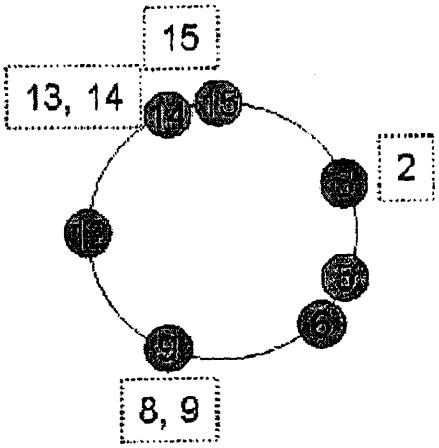
4.1. Εισαγωγή

Στο κεφάλαιο αυτό περιγράφεται το σύστημα του Chord [26], ένα σύστημα που επιτρέπει τη διαχείριση μονοδιάστατων δεδομένων και στου οποίου τη δομή βασίζονται πολλά άλλα συστήματα (π.χ. το MAAN [4], επίσης, το Mercury [3] καθώς και το TreeChord που προτείνεται σε αυτή την εργασία). Στο Chord, όλοι οι κόμβοι τοποθετούνται ομοιόμορφα σε ένα δακτύλιο (*Chord ring*), ενώ τα δεδομένα του συστήματος κατανέμονται ομοιόμορφα στους υπάρχοντες κόμβους. Ο δακτύλιος αυτός αποτελεί το χώρο αναγνωριστικών του συστήματος. Πιο συγκεκριμένα, το Chord αντιστοιχίζει ένα κλειδί (που αποτελείται από m bits, όπως προαναφέρθηκε) τόσο στους κόμβους, όσο και στα δεδομένα του συστήματος χρησιμοποιώντας συνεπή κατακερματισμό (*consistent hashing* [5, 17]). Οι κόμβοι διατάσσονται στο δακτύλιο βάσει του κλειδιού τους, ενώ ένα δεδομένο διατηρείται στον κόμβο με το κοντινότερο μεγαλύτερο κλειδί από το κλειδί του, ο οποίος ονομάζεται διάδοχος κόμβος (*successor node*). Συμβολίζουμε με $succ(k)$ το διάδοχο κόμβο του κλειδιού k. Επομένως, κάθε κόμβος είναι υπεύθυνος για όλα τα δεδομένα με κλειδιά μεταξύ του κλειδιού του προηγούμενου κόμβου στο δακτύλιο (*προκάτοχος κόμβος* ή *predecessor node*) και του δικού του κλειδιού. Στο Σχήμα 4.1 βλέπουμε ένα παράδειγμα δακτυλίου με $m = 4$. Οι κόμβοι 3, 5, 6, 9, 12, 14 και 15 συμμετέχουν στο σύστημα. Επιπλέον, ο κόμβος 2 διατηρεί το δεδομένο με κλειδί 2, ο κόμβος 9 διατηρεί τα δεδομένα με κλειδιά 8 και 9, ο κόμβος 14 διατηρεί τα δεδομένα με κλειδιά 13 και 14 και ο κόμβος 15 διατηρεί το δεδομένο με κλειδί 15.

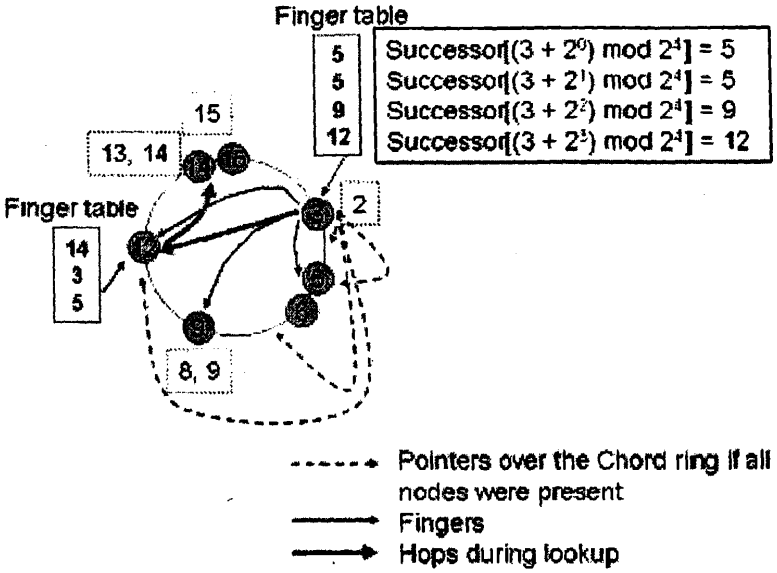
Προκειμένου να πραγματοποιούνται αποδοτικά οι αναζητήσεις δεδομένων στο σύστημα, κάθε κόμβος στο Chord γνωρίζει, εκτός από τον προκάτοχο και το διάδοχό του στο δακτύλιο, και μερικούς ακόμη κόμβους. Πιο συγκεκριμένα, κάθε κόμβος γνωρίζει όλους τους άλλους κόμβους που έχουν το κοντινότερο μεγαλύτερο κλειδί από κλειδιά τα οποία απέχουν από το δικό του κλειδί κατά αυξανόμενες δυνάμεις του 2. Οι πληροφορίες αυτές, οι οποίες χρησιμοποιούνται κατά τη δρομολόγηση, διατηρούνται σε έναν πίνακα δρομολόγησης, που ονομάζεται πίνακας δεικτών (*finger table*). Επομένως, ο πίνακας δρομολόγησης κάθε κόμβου n περιέχει το πολύ $\log 2^m = m$ καταχωρήσεις. Η $i-$οστή καταχώρηση του πίνακα αυτού περιέχει τις πληροφορίες για τον πρώτο κόμβο n' που έπεται του n και απέχει κατά τουλάχιστον 2^{i-1} από αυτόν στο δακτύλιο, δηλαδή $n' = succ\left(n+2^{i-1}\right)$, $1 \le i \le m$. Πρόκειται για τον $i-$οστό δείκτη ($i-th\ finger$). Οι δείκτες αυτοί μοιάζουν με χορδές στο δακτύλιο του Chord. Στο Σχήμα 4.2 βλέπουμε για παράδειγμα πώς υπολογίζονται οι δείκτες που διατηρούνται στον κόμβο 3. Αν συμμετείχαν στο σύστημα οι κόμβοι με όλα τα δυνατά κλειδιά, τότε ο κόμβος 3 θα έπρεπε να διατηρεί δείκτες στους κόμβους 4, 5, 7 και 11. Καθώς, όμως, ο κόμβος 4 δεν υπάρχει, ο πρώτος δείκτης δείχνει στο διάδοχο του 4, δηλαδή το 5. Ο δεύτερος δείκτης δείχνει, επίσης, στον ίδιο κόμβο, δηλαδή τον 5, ο τρίτος δείκτης δείχνει στον κόμβο 9, που είναι ο διάδοχος του 7 και ο τέταρτος δείκτης δείχνει στο διάδοχο του 11, που είναι ο κόμβος 12.

Βάσει των παραπάνω, αποδεικνύεται ότι σε ένα σύστημα N κόμβων κάθε κόμβος γνωρίζει $O(\log N)$ κόμβους και επιλύει τις απλές ερωτήσεις κλειδιών μέσω $O(\log N)$ μηνυμάτων σε άλλους κόμβους.

Το Chord υποστηρίζει αποδοτικά τη διαχείριση μονοδιάστατων δεδομένων στο σύστημα βασιζόμενο στις πληροφορίες δρομολόγησης που διατηρούνται στους κόμβους του συστήματος. Στη συνέχεια περιγράφεται αναλυτικότερα ο τρόπος με τον οποίο επιλύονται οι απλές ερωτήσεις κλειδιών στο Chord.



Σχήμα 4.1 Ο Δακτύλιος του Chord. Οι Κόμβοι και τα Δεδομένα που Διατηρούν.



Σχήμα 4.2 Οι δείκτες Δρομολόγησης που Διατηρούνται στο Chord και Ένα Παράδειγμα Επίλυσης Απλής Ερώτησης.

4.2. Επίλυση Απλών Ερωτήσεων

Προκειμένου να επιλυθούν οι απλές ερωτήσεις, οι κόμβοι του συστήματος χρησιμοποιούν τους πίνακες δρομολόγησης που περιγράφηκαν παραπάνω. Έστω, λοιπόν, ένας κόμβος n του συστήματος που αναζητά το κλειδί k. Ο n εξετάζει αρχικά αν ο ίδιος είναι ο διάδοχος κόμβος του κλειδιού k, δηλαδή αν το κλειδί k ανήκει στο διάστημα μεταξύ του κλειδιού του προκατόχου του και του δικού του. Αν όντως ισχύει κάτι τέτοιο, ο κόμβος n πρέπει να διατηρεί το ζητούμενο δεδομένο, οπότε εξετάζονται τα δεδομένα που διατηρούνται τοπικά.

Στην αντίθετη περίπτωση, ο n αναζητά στον πίνακα δεικτών του τον κόμβο που έχει το μεγαλύτερο κλειδί το οποίο είναι μικρότερο από το κλειδί που αναζητείται και προωθεί στον κόμβο αυτό την ερώτηση. Η διαδικασία συνεχίζεται μέχρι να βρεθεί ο διάδοχος κόμβος του ζητούμενου κλειδιού k, που πιθανόν διατηρεί και το ζητούμενο δεδομένο. Αποδεικνύεται πως η επίλυση τέτοιου είδους απλών ερωτήσεων απαιτεί $O(\log N)$ βήματα και $O(\log N)$ μηνύματα. Στο Σχήμα 4.2 βλέπουμε ένα παράδειγμα επίλυσης μίας απλής ερώτησης. Ο κόμβος 3 αναζητά το κλειδί 13. Προωθεί, λοιπόν, την ερώτηση στον κόμβο 12 από τον πίνακα δρομολόγησης που διατηρεί, που είναι ο κόμβος με το μεγαλύτερο κλειδί του οποίου η τιμή δεν υπερβαίνει την τιμή του κλειδιού που αναζητείται, δηλαδή την τιμή 13. Στη συνέχεια, ο κόμβος 12 προωθεί την ερώτηση στο διάδοχό του, όπου επιλύεται, αφού ο κόμβος 14, που είναι ο διάδοχος κόμβος του ζητούμενου κλειδιού 13, διαθέτει το δεδομένο που ζητείται.

Το σύστημα του Chord, επομένως, κατορθώνει να επιλύσει αποδοτικά απλές ερωτήσεις χρησιμοποιώντας τις λίγες πληροφορίες που διατηρούν οι συμμετέχοντες κόμβοι για άλλους κόμβους του συστήματος. Η επίλυση ερωτήσεων διαστήματος, όμως, δεν είναι δυνατή, αφού οι τεχνικές κατακερματισμού που εφαρμόζονται για την κατανομή των δεδομένων στους κόμβους του συστήματος καταστρέφουν τη διάταξη των δεδομένων.

4.3. Εισαγωγή Νέων Κόμβων στο Σύστημα

Η επίλυση των απλών ερωτήσεων στο σύστημα πρέπει να επιτρέπεται ακόμη κι αν το σύστημα αλλάζει δυναμικά, καθώς νέοι κόμβοι εισέρχονται και άλλοι αποχωρούν οποιαδήποτε στιγμή. Για το λόγο αυτό, πρέπει να διατηρείται η συνέχεια του δακτυλίου, δηλαδή κάθε κόμβος πρέπει να γνωρίζει το σωστό προκάτοχο και διάδοχό του στο δακτύλιο. Επιπλέον, οι πίνακες δεικτών πρέπει να διατηρούνται ενημερωμένοι και τα κλειδιά να διατηρούνται στους σωστούς κόμβους, έτσι ώστε οι αναζητήσεις να πραγματοποιούνται σωστά και αποδοτικά. Επομένως, κάθε φορά που ένας νέος κόμβος εισέρχεται στο σύστημα πρέπει να αρχικοποιεί κατάλληλα τον πίνακα δεικτών και να ενημερώνει το διάδοχό του στο δακτύλιο, έτσι ώστε να μεταφέρονται στον ίδιο τα κατάλληλα δεδομένα.

Όταν, λοιπόν, εισάγεται ένας νέος κόμβος n στο σύστημα, υπολογίζεται αρχικά το κλειδί k που του αντιστοιχεί. Το κλειδί αυτό προκύπτει εφαρμόζοντας μία συνάρτηση

κατακερματισμού, όπως είναι η SHA-1 συνάρτηση, στην IP διεύθυνση του κόμβου. Στη συνέχεια, ευρίσκεται ο διάδοχος κόμβος του k, $succ(k)$, μέσω του αλγορίθμου δρομολόγησης του συστήματος και ο νέος κόμβος εισάγεται ως προκάτοχος του $succ(k)$. Κατόπιν, ενημερώνονται κατάλληλα οι predecessor και successor δείκτες των κόμβων n και $succ(k)$. Επίσης, ο κόμβος n αρχικοποιεί τον πίνακα δρομολόγησής του χρησιμοποιώντας τον πίνακα δρομολόγησης του $succ(k)$. Οι καταχωρήσεις του πίνακα αυτού ενημερώνονται με την πάροδο του χρόνου μέσω μίας ρουτίνας που εκτελείται περιοδικά από κάθε κόμβο του συστήματος. Τέλος, μεταφέρονται στον κόμβο n όσα από τα δεδομένα του $succ(k)$ αντιστοιχούν σε κλειδιά μεγαλύτερα από το κλειδί k του νέου κόμβου, αφού τα κλειδιά αυτά πρέπει να αποθηκευτούν στο νέο τους διάδοχο, που είναι πλέον ο n. Και οι υπόλοιποι κόμβοι, όμως, πρέπει να ενημερωθούν για την είσοδο του νέου κόμβου. Η ενημέρωσή τους γίνεται, επίσης, μέσω της ρουτίνας που εκτελείται περιοδικά από όλους τους κόμβους του συστήματος. Σε περίπτωση που πραγματοποιηθεί μια ερώτηση πριν να ολοκληρωθούν οι ρουτίνες σταθεροποίησης, μπορεί να καθυστερήσει ή και να αποτύχει. Αποδεικνύεται, όμως, ότι όσο ο χρόνος που χρειάζεται για την ενημέρωση των διαδόχων κόμβων είναι μικρότερος από το χρόνο που περνάει για να διπλασιαστεί το σύστημα σε μέγεθος, η αναζήτηση εξακολουθεί να κοστίζει $O(\log N)$ βήματα και $O(\log N)$ μηνύματα.

Σημειώνεται ότι για να είναι πιο αξιόπιστο το σύστημα, κάθε κόμβος διατηρεί μια λίστα μεγέθους ls με τους επόμενους ls διαδόχους του (*successor list*). Έτσι, αν ο διάδοχος ενός κόμβου δεν ανταποκρίνεται κατά την προώθηση μίας ερώτησης, δοκιμάζεται ο επόμενος από τη λίστα. Για να διασπαστεί ο δακτύλιος πλέον, πρέπει να αποτύχουν ταυτόχρονα όλοι οι κόμβοι της λίστας, γεγονός με πολύ μικρή πιθανότητα να συμβεί.

Τέλος, όταν αποτύχει ένας κόμβος, πρέπει να ενημερωθούν οι δείκτες άλλων κόμβων προς αυτών. Η ενημέρωση αυτή επιτυγχάνεται μέσω της ρουτίνας που εκτελείται περιοδικά από κάθε κόμβο του συστήματος. Ωστόσο, στην περίπτωση που ένας κόμβος αποχωρήσει οικιοθελώς από το σύστημα , μπορεί, πριν την αποχώρησή του, να μεταφέρει τα κλειδιά για τα οποία είναι υπεύθυνος στο διάδοχό του και να ενημερώσει τους δύο κόμβους μεταξύ των οποίων βρίσκεται για την αποχώρησή του, έτσι ώστε να ενημερώσουν κι αυτοί με τη σειρά τους τους πίνακες δεικτών που διατηρούν.

ΚΕΦΑΛΑΙΟ 5. ΤΟ ΣΥΣΤΗΜΑ ΤΟΥ MAAN

5.1. Εισαγωγή

Στο κεφάλαιο αυτό περιγράφεται το σύστημα του MAAN [4] (*Multi-Attribute Addressable Network*), ένα δομημένο σύστημα ομότιμων κόμβων που βασίζεται στην αρχιτεκτονική του Chord, αλλά επιτρέπει τη διαχείριση πολυδιάστατων δεδομένων και την πραγματοποίηση ερωτήσεων διαστήματος. Στο MAAN χρησιμοποιούνται συναρτήσεις κατακερματισμού που διατηρούν τη διάταξη (*locality preserving hash functions*) για την ανάθεση των δεδομένων στους κόμβους του συστήματος και εφαρμόζονται δύο αλγόριθμοι επίλυσης ερωτήσεων διαστήματος. Κάθε κόμβος του MAAN διατηρεί τις ίδιες πληροφορίες που διατηρεί το Chord για τους υπόλοιπους κόμβους του συστήματος, ενώ η επίλυση των ερωτήσεων διαστήματος πραγματοποιείται γρήγορα.

5.2. Βασικά Στοιχεία του Συστήματος

Όπως και στο σύστημα του Chord, σε κάθε κόμβο του MAAN ανατίθεται ένα μοναδικό αναγνωριστικό (*node ID*), που αποτελείται από m δυαδικά ψηφία. Το αναγνωριστικό αυτό προκύπτει όπως και στο σύστημα του Chord, με εφαρμογή μίας συνάρτησης κατακερματισμού, όπως είναι η SHA1, στην IP διεύθυνση του κόμβου. Όλοι οι κόμβοι του

συστήματος οργανώνονται σε ένα λογικό δακτύλιο μεγέθους 2^m βάσει του αναγνωριστικού τους. Με άλλα λόγια, ο χώρος των αναγνωριστικών του συστήματος είναι ο $[0, 2^m)$. Έτσι, οι κόμβοι κατανέμονται ομοιόμορφα στο λογικό δακτύλιο.

Στα δεδομένα του συστήματος, τα οποία μπορούν να είναι πολυδιάστατα, ανατίθενται, επίσης, αναγνωριστικά, ένα για κάθε μία από τις διαστάσεις τους, μέσω συναρτήσεων κατακερματισμού και, στη συνέχεια, τοποθετούνται προς αποθήκευση στους κατάλληλους κόμβους του συστήματος. Ενώ, όμως, στο σύστημα του Chord χρησιμοποιείται η ίδια συνάρτηση κατακερματισμού για την ανάθεση αναγνωριστικών τόσο στα δεδομένα, όσο και στους κόμβους του συστήματος, η οποία καταστρέφει τη διάταξη των δεδομένων, με αποτέλεσμα να μην καθίσταται δυνατή η πραγματοποίηση ερωτήσεων διαστήματος, στο σύστημα του MAAN εφαρμόζονται συναρτήσεις κατακερματισμού που διατηρούν τη διάταξη για την ανάθεση αναγνωριστικών στα δεδομένα. Τα αναγνωριστικά που προκύπτουν ανήκουν, επίσης, στο χώρο αναγνωριστικών $[0, 2^m)$. Ωστόσο, οι συναρτήσεις κατακερματισμού που διατηρούν τη διάταξη δεν καταφέρνουν πάντα να κατανείμουν ομοιόμορφα τα κλειδιά στους κόμβους. Έτσι, μπορούν να εφαρμοστούν διάφορες συναρτήσεις κατακερματισμού, ανάλογα με την κατανομή που ακολουθείται από τα δεδομένα εισόδου.

Ορισμός. Μία συνάρτηση κατακερματισμού H διατηρεί τη διάταξη αν έχει την ακόλουθη ιδιότητα: $\forall v_i, v_j$ (1) $H(v_i) < H(v_j)$ αν και μόνο αν $v_i < v_j$ και, επίσης, (2) αν το διάστημα $[v_i, v_j]$ διαμερίζεται στα διαστήματα $[v_i, v_k]$ και $[v_k, v_j]$, τότε το αντίστοιχο διάστημα $[H(v_i), H(v_j)]$ πρέπει να διαμερίζεται στα διαστήματα $[H(v_i), H(v_k)]$ και $[H(v_k), H(v_j)]$.

Έστω, λοιπόν, ένα δεδομένο και ένα χαρακτηριστικό α αυτού, με εύρος τιμών $[v_{\min}, v_{\max}]$. Μία απλή συνάρτηση κατακερματισμού που διατηρεί τη διάταξη είναι η

$H(v) = (v - v_{\min}) \times (2^m - 1) / (v_{\max} - v_{\min})$	Εξ. 4.1

όπου $v \in [v_{min}, v_{max}]$ είναι η τιμή του δεδομένου για το χαρακτηριστικό α. Είναι προφανές ότι για κάθε τιμή v του χαρακτηριστικού το αναγνωριστικό $H(v)$ που προκύπτει ανήκει στο χώρο αναγνωριστικών $[0, 2^m)$.

Οι τιμές των χαρακτηριστικών των δεδομένων που εισάγονται στο σύστημα, όμως, μπορεί να μην είναι ομοιόμορφα κατανεμημένες. Στην περίπτωση αυτή, η απλή συνάρτηση κατακερματισμού που διατηρεί τη διάταξη των δεδομένων και παρουσιάστηκε παραπάνω δεν παράγει τιμές ομοιόμορφα κατανεμημένες στο χώρο των αναγνωριστικών, αφού απλά μετατοπίζει τις τιμές εισόδου. Αυτό μπορεί να δημιουργήσει προβλήματα με την κατανομή φορτίου μεταξύ των κόμβων του συστήματος, με αποτέλεσμα άλλοι κόμβοι να υπερφορτώνονται με πολλά δεδομένα, ενώ άλλοι να διαθέτουν ελάχιστο πλήθος δεδομένων. Δεδομένου, όμως, ότι οι τιμές των χαρακτηριστικών των δεδομένων εισόδου ακολουθούν μία συνεχή και μονότονα αύξουσα συνάρτηση κατανομής, που είναι εκ των προτέρων γνωστή, λύση στο παραπάνω πρόβλημα αποτελεί η χρήση ομοιόμορφων συναρτήσεων κατακερματισμού που διατηρούν τη διάταξη (*uniform locality preserving hash functions*). Τέτοιου είδους συναρτήσεις παράγουν ομοιόμορφα κατανεμημένες τιμές στο χώρο αναγνωριστικών. Διάφορες γνωστές κατανομές (π.χ. η κανονική ή Gaussian, η Pareto και η εκθετική) έχουν αυτές τις ιδιότητες. Αν, λοιπόν, οι τιμές $v \in [v_{min}, v_{max}]$ των χαρακτηριστικών των δεδομένων ακολουθούν ή συγκλίνουν σε κάποια κατανομή με συνεχή και μονότονα αύξουσα συνάρτηση κατανομής $D(v)$ και $P(v) = dD(v)/dv$ είναι η συνάρτηση πιθανότητας, τότε αποδεικνύεται ότι η συνάρτηση

$H(v) = D(v) \times (2^m - 1)$	Εξ. 4.2

είναι μία ομοιόμορφη συνάρτηση κατακερματισμού που διατηρεί τη διάταξη.

Αν, λοιπόν, τα δεδομένα ακολουθούν συγκεκριμένες κατανομές με τις εν λόγω ιδιότητες και χρησιμοποιηθεί μία ομοιόμορφη συνάρτηση κατακερματισμού που διατηρεί τη διάταξη των δεδομένων, τότε τα δεδομένα κατανέμονται ομοιόμορφα στους κόμβους του δακτυλίου, εφόσον, φυσικά, και οι κόμβοι έχουν κατανεμηθεί ομοιόμορφα στο δακτύλιο.

```
n.insert(data)
  for each dimension i
    n' = find_successor(H_i(i.data));
    insert data at node n';
```

Σχήμα 5.1 Ψευδοκώδικας για την Εισαγωγή Δεδομένων στο Σύστημα του MAAN.

5.3. Εισαγωγή Δεδομένων

Όλα τα δεδομένα που εισάγονται στο σύστημα ανατίθενται στους κόμβους που συμμετέχουν σε αυτό. Ένα δεδομένο με τιμή v για κάποιο χαρακτηριστικό του ανατίθεται στον κόμβο εκείνο στον οποίο έχει ανατεθεί το ίδιο ή το κοντινότερο μεγαλύτερο αναγνωριστικό από το αναγνωριστικό του δεδομένου $H(v)$ για το συγκεκριμένο χαρακτηριστικό. Ο κόμβος αυτός είναι ο *διάδοχος* (successor) κόμβος, δηλαδή ο $succ(H(v))$.

Έστω ότι τα δεδομένα του συστήματος δεν έχουν μόνο ένα χαρακτηριστικό (μονοδιάστατα), αλλά έχουν M χαρακτηριστικά $\alpha_1, \alpha_2, ..., \alpha_M$, δηλαδή είναι πολυδιάστατα, και έστω $\langle a_i, v_i \rangle$, $1 \le i \le M$, το ζεύγος που προσδιορίζει σε ποιο χαρακτηριστικό αναφερόμαστε (a_i) και ποια η τιμή του εν λόγω χαρακτηριστικού (v_i). Για κάθε χαρακτηριστικό a_i των δεδομένων του συστήματος χρησιμοποιούμε μία συνάρτηση κατακερματισμού που διατηρεί τη διάταξη και αντιστοιχίζουμε τις τιμές των χαρακτηριστικών των δεδομένων στο χώρο αναγνωριστικών του συστήματος.

Κάθε πολυδιάστατο δεδομένο, λοιπόν, που εισάγεται στο σύστημα διατηρείται στον κόμβο $n_i = succ(H(v_i))$ για κάθε τιμή v_i, $1 \le i \le M$. Με άλλα λόγια, κάθε δεδομένο εισάγεται ως προς όλα τα χαρακτηριστικά του στο σύστημα και κάθε κόμβος διατηρεί ένα δεδομένο ως ένα διάνυσμα με τιμές για κάθε διάσταση. Έτσι, για κάθε δεδομένο κρατούνται στο σύστημα τόσα αντίγραφα, όσες και οι διαστάσεις του. Ο κόμβος στον οποίο διατηρείται ένα δεδομένο μετά την εισαγωγή του στο σύστημα ευρίσκεται μέσω του αλγορίθμου δρομολόγησης του Chord, που εκτελείται που εκτελείται για το κλειδί οποιασδήποτε διάστασης.

Στο Σχήμα 5.1 βλέπουμε τον ψευδοκώδικα για την εισαγωγή δεδομένων στο σύστημα του MAAN.

5.4. Επίλυση Απλών Ερωτήσεων

Κάθε κόμβος του συστήματος μπορεί να εκτελεί απλές ερωτήσεις στις οποίες προσδιορίζει την τιμή που πρέπει να έχει κάθε ένα από τα χαρακτηριστικά του επιθυμητού δεδομένου. Μία απλή ερώτηση επιλύεται με τον ίδιο τρόπο που επιλύονται και οι ερωτήσεις στο σύστημα του Chord. Όπως είδαμε, όμως, στο MAAN γίνεται εισαγωγή των δεδομένων ως προς κάθε ένα από τα χαρακτηριστικά τους, με αποτέλεσμα να υπάρχουν τόσα αντίγραφα ενός δεδομένου, όσα και τα χαρακτηριστικά του. Μπορούμε, επομένως, να επιλέξουμε μεταξύ των αντιγράφων ενός δεδομένου εκείνο που θα αναζητήσουμε. Η επιλογή μπορεί να γίνει είτε τυχαία, είτε βάσει υπολογισμού ενός μετρικού (π.χ. βάσει της απόστασης στο δακτύλιο του κλειδιού ενός αντιγράφου από το κλειδί του τρέχοντα κόμβου).

Αν επιλέγουμε τυχαία και με ομοιόμορφο τρόπο ποιο από τα αντίγραφα του δεδομένου θα αναζητήσουμε, οι ερωτήσεις κατανέμονται ομοιόμορφα στους κόμβους του συστήματος με μεγάλη πιθανότητα και δεν παρατηρείται ανομοιομορφία ως προς το πλήθος των ερωτήσεων που καλούνται να απαντήσουν οι κόμβοι στο σύστημα.

Από την άλλη πλευρά, μπορούμε να υπολογίσουμε την 'απόσταση' του κόμβου που εκκινεί την ερώτηση από κάθε ένα από τα αντίγραφα του δεδομένου που αναζητείται και, στη συνέχεια, να επιλέξουμε προς αναζήτηση εκείνο το αντίγραφο που απέχει λιγότερο. Η απόσταση ενός κόμβου από ένα δεδομένο μπορεί να οριστεί ως η απόλυτη τιμή της λογαριθμικής διαφοράς του αναγνωριστικού του κόμβου από το αναγνωριστικό του δεδομένου. Η επιλογή του δεδομένου προς αναζήτηση με τον τρόπο αυτό οδηγεί σε μείωση του πλήθους των βημάτων που απαιτούνται για την επίλυση μίας απλής ερώτησης, δηλαδή για την εύρεση του κόμβου που διατηρεί το ζητούμενο δεδομένο.

Στο Σχήμα 5.2 βλέπουμε τον ψευδοκώδικα για την επίλυση απλών ερωτήσεων στο σύστημα του MAAN.

```
n.simple_find(data)
   choose i^th dimension;
   n.find_successor(H_i(data.i));

n.single_dimension_range_query(i, l, u)
   n_l = find_successor(H_i(l));
   n_l.check_upper_bound(i, u);

n.check_upper_bound(i, u)
   n checks data stored locally;
   if n is the successor of H_i(u)
      stop forwarding the query;
   else
      n' = n.successor;
      n'.check_upper_bound(i, u);
```

Σχήμα 5.2 Ψευδοκώδικας για την Επίλυση Απλών Ερωτήσεων και Ερωτήσεων Διαστήματος στο Σύστημα του MAAN.

5.5. Επίλυση Ερωτήσεων Διαστήματος

Ένας κόμβος, όμως, μπορεί να εκτελεί και ερωτήσεις διαστήματος. Στην περίπτωση αυτή, υπενθυμίζουμε πως ο κόμβος που εκκινεί μία ερώτηση διαστήματος ορίζει το ζητούμενο εύρος τιμών για κάθε χαρακτηριστικό των δεδομένων που αναζητά. Υπενθυμίζουμε ότι η ερώτηση διαστήματος, λοιπόν, ορίζεται για κάθε χαρακτηριστικό ως εξής: $v_{il} \le v_i \le v_{iu}$, $1 \le i \le M$, όπου v_{il} και v_{iu} το κάτω και το άνω όριο που προσδιορίζουν το διάστημα στο οποίο πρέπει να ανήκουν οι τιμές του χαρακτηριστικού i των δεδομένων.

Το θεώρημα που ακολουθεί μας επιτρέπει να σχεδιάσουμε αλγόριθμο επίλυσης ερωτήσεων διαστήματος στο σύστημα.

Θεώρημα 5.1. Έστω μία συνάρτηση κατακερματισμού H που διατηρεί τη διάταξη και αντιστοιχίζει τις τιμές v ενός χαρακτηριστικού στο χώρο αναγνωριστικών $\left[0, 2^m\right)$. Έστω,

επίσης, μία ερώτηση διαστήματος $[l,u]$, όπου l και u το κάτω και το άνω όριο της ερώτησης, αντίστοιχα. Τότε τα αναγνωριστικά των κόμβων που διαθέτουν δεδομένα με τιμή $v \in [l,u]$ για το συγκεκριμένο χαρακτηριστικό πρέπει να ανήκουν στο διάστημα $\left[successor\left(H(l)\right), successor\left(H(u)\right)\right]$ [4].

Απόδειξη. Σύμφωνα με όσα προαναφέρθηκαν, ένα δεδομένο με τιμή v για το εν λόγω χαρακτηριστικό ανατίθεται στον κόμβο $successor\left(H(v)\right)$. Ο κόμβος αυτός είναι ο πρώτος κόμβος του οποίου το αναγνωριστικό ισούται ή ακολουθεί το αναγνωριστικό $H(v)$ στο χώρο αναγνωριστικών. Λόγω, όμως, του παραπάνω ορισμού και αφού $l \le v \le u$, δεδομένα με τιμή v για το εν λόγω χαρακτηριστικό ανατίθενται σε κόμβο με αναγνωριστικό n για τον οποίο ισχύει $successor\left(H(l)\right) \le n \le successor\left(H(u)\right)$.

5.5.1. Γενικός Αλγόριθμος Επίλυσης Ερωτήσεων Διαστήματος

Όσα προαναφέρθηκαν μας οδηγούν στον αλγόριθμο επίλυσης ερωτήσεων διαστήματος. Έστω κάποιος κόμβος με αναγνωριστικό n, που αναζητά όλα τα δεδομένα με οποιαδήποτε τιμή $v \in [l,u]$ για κάποιο χαρακτηριστικό α. Ο κόμβος n χρησιμοποιεί τον αλγόριθμο δρομολόγησης του συστήματος του Chord και δρομολογεί την ερώτηση στον κόμβο με αναγνωριστικό n_l, που είναι ο διάδοχος του $H(l)$ $\left(succ\left(H(l)\right)\right)$. Στην ερώτηση που προωθείται στο δακτύλιο ορίζονται η διάσταση α για την οποία εκτελείται η ερώτηση διαστήματος, το κλειδί $k = H(l)$ που αναζητείται με τον αλγόριθμο δρομολόγησης του Chord και το ζητούμενο εύρος $[l,u]$. Όταν ο κόμβος n_l λάβει την ερώτηση, εξετάζει αν κάποιο από τα δεδομένα που διατηρεί τοπικά ανήκουν στο ζητούμενο εύρος ελέγχοντας την αντίστοιχη διάσταση, η οποία ορίζεται στην ερώτηση. Όσα από τα τοπικά δεδομένα ανήκουν όντως στο εύρος που ορίζεται στην ερώτηση επιστρέφονται. Στη συνέχεια, ο κόμβος n_l ελέγχει αν ο ίδιος είναι και διάδοχος του $H(u)$, δηλαδή της τιμής που έχει αντιστοιχιστεί στο άνω όριο του ζητούμενου εύρους. Αν κάτι τέτοιο ισχύει, η προώθηση της ερώτησης σταματά, ενώ όλα τα δεδομένα που βρέθηκαν και ανήκουν στο ζητούμενο εύρος έχουν επιστραφεί στον

κόμβο n. Στην αντίθετη περίπτωση, ο n_l προωθεί την ερώτηση στον επόμενό του στο δακτύλιο, δηλαδή στο διάδοχό του, έστω n_i. Ο n_i με τη σειρά του εξετάζει τα δεδομένα που διατηρεί επιστρέφοντας όσα από αυτά ανήκουν στο ζητούμενο εύρος και προωθεί την ερώτηση στον επόμενό του, έως ότου η ερώτηση καταλήξει στον κόμβο n_u, που είναι ο διάδοχος του $H(u)$ $\left(succ\left(H(u)\right)\right)$.

Στο Σχήμα 5.2 βλέπουμε τον ψευδοκώδικα του γενικού αλγορίθμου για την επίλυση ερωτήσεων διαστήματος στο σύστημα του MAAN.

Σύμφωνα με το Θεώρημα 5.1, όσοι από τους κόμβους του συστήματος διαθέτουν δεδομένα με χαρακτηριστικό των οποίων η τιμή ανήκει στο ζητούμενο εύρος $[l,u]$, πρέπει να έχουν τοποθετηθεί μεταξύ των κόμβων n_l και n_u στο δακτύλιο. Ο αλγόριθμος που μόλις περιγράφηκε αναζητά δεδομένα με τιμές χαρακτηριστικού που ανήκουν στο ζητούμενο εύρος στους κόμβους που πιθανώς θα τα διατηρούσαν, αν είχαν εισαχθεί στο σύστημα.

Καθώς χρησιμοποιείται ο αλγόριθμος δρομολόγησης του Chord, η δρομολόγηση της ερώτησης διαστήματος στον κόμβο n_l απαιτεί $O(\log N)$ βήματα, όπου N είναι το πλήθος των κόμβων που συμμετέχουν στο σύστημα. Κατόπιν, η ερώτηση πρέπει να προωθηθεί από τον κόμβο n_l στον κόμβο n_u. Επειδή, όμως, κάθε κόμβος προωθεί την ερώτηση στον επόμενό του στο δακτύλιο, αν υπάρχουν K κόμβοι μεταξύ των n_l και n_u, τότε απαιτούνται επιπλέον K βήματα. Συνεπώς, μία ερώτηση διαστήματος που αναζητά δεδομένα των οποίων οι τιμές για ένα μόνο χαρακτηριστικό ανήκουν σε κάποιο εύρος τιμών απαιτεί $O(\log N)+K$ βήματα.

5.5.2. Αλγόριθμοι Επίλυσης Ερωτήσεων Διαστήματος ως προς Πολλά Χαρακτηριστικά

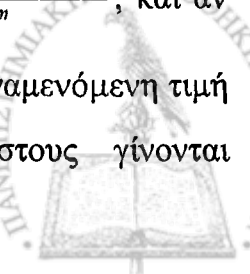
Ο γενικός αλγόριθμος για την επίλυση ερωτήσεων διαστήματος ως προς ένα χαρακτηριστικό, που παρουσιάστηκε παραπάνω, μπορεί να γενικευτεί. Για την πραγματοποίηση ερωτήσεων διαστήματος για περισσότερα του ενός χαρακτηριστικά υπάρχουν διάφορες προσεγγίσεις. Το MAAN εστιάζει σε τρεις από αυτές, οι οποίες περιγράφονται στη συνέχεια.

5.5.2.1. Επαναληπτική Επεξεργασία Ερωτήσεων

Κάθε κόμβος n που επιθυμεί να εκτελέσει μία ερώτηση διαστήματος q ως προς M χαρακτηριστικά, εκτελεί M ερωτήσεις διαστήματος, μία για κάθε χαρακτηριστικό, χρησιμοποιώντας το γενικό αλγόριθμο που παρουσιάστηκε παραπάνω. Σε κάθε μία από αυτές τις ερωτήσεις ελέγχεται μόνο αν ικανοποιείται η συνθήκη για το αντίστοιχο χαρακτηριστικό και επιστρέφονται τα κατάλληλα δεδομένα. Το αποτέλεσμα των ερωτήσεων αυτών είναι να λάβει ο κόμβος n δεδομένα που δεν ικανοποιούν όλες τις συνθήκες. Προκειμένου, λοιπόν, ο κόμβος n να απαντήσει στην αρχική ερώτηση q πρέπει να υπολογίσει την τομή των δεδομένων που θα λάβει.

Στη συνέχεια εξετάζουμε την πολυπλοκότητα του αλγορίθμου υπολογίζοντας το πλήθος των βημάτων ή των μηνυμάτων που απαιτούνται μέχρι να καταλήξει η ερώτηση σε όλους τους κόμβους που πιθανώς διαθέτουν δεδομένα, αλλά όχι και το πλήθος των βημάτων ή των μηνυμάτων που απαιτούνται για την επιστροφή των δεδομένων στον κόμβο που εκκίνησε την ερώτηση, παρότι το δεύτερο κόστος αναμένεται να είναι ιδιαίτερα υψηλό.

Παρατηρούμε, λοιπόν, πως όταν εφαρμόζεται ο παραπάνω αλγόριθμος επίλυσης ερωτήσεων διαστήματος, εκτελείται μία ερώτηση για κάθε μία από τις διαστάσεις των δεδομένων του συστήματος ξεχωριστά. Επομένως, σύμφωνα με όσα προαναφέρθηκαν, κάθε ερώτηση διαστήματος η οποία αναζητά δεδομένα των οποίων οι τιμές για ένα μόνο χαρακτηριστικό, έστω i, ανήκουν σε κάποιο εύρος τιμών απαιτεί $O(\log N)+K_i$ μηνύματα και $O(\log N)+K_i$ βήματα, όπου K_i είναι το πλήθος των κόμβων που ανήκουν στο εύρος που ορίζεται στην ερώτηση για τη διάσταση i. Η ερώτηση διαστήματος q, λοιπόν, που πραγματοποιείται στο σύστημα απαιτεί $O\left(\sum_{i=1}^{M}(\log N+K_i)\right)$ ή $O\left(M\log N+\sum_{i=1}^{M}K_i\right)$ μηνύματα, όπου M το πλήθος των διαστάσεων των δεδομένων του συστήματος. Αν χρησιμοποιήσουμε τον ορισμό της επιλεκτικότητας s_i μιας ερώτησης διαστήματος ως προς τη διάσταση i που παρουσιάστηκε σε προηγούμενη ενότητα, δηλαδή το λόγο του εύρους που καλύπτει η ερώτηση προς το συνολικό εύρος του χώρου αναγνωριστικών ή, με άλλα λόγια, $s_i=\frac{H(v_{iu})-H(v_{il})}{2^m}$, και αν υποθέσουμε ότι οι κόμβοι κατανέμονται ομοιόμορφα στο δακτύλιο, τότε η αναμενόμενη τιμή του K_i ισούται με $s_i\times N$ και οι αναμενόμενες τιμές του κόστους γίνονται



$O\left(\sum_{i=1}^{M}(\log N + N \times s_i)\right)$ και $O\left(M \log N + \sum_{i=1}^{M} N \times s_i\right)$, αντίστοιχα. Σημειώνουμε, πως στους παραπάνω υπολογισμούς δεν συμπεριλαμβάνεται το πλήθος βημάτων ή μηνυμάτων που απαιτείται για την επιστροφή των δεδομένων στον κόμβο που εκκινεί την ερώτηση διαστήματος.

Όταν μετρούμε το κόστος μιας ερώτησης διαστήματος βάσει του πλήθους των μηνυμάτων που απαιτούνται για την ικανοποίησή της, υπολογίζουμε το άθροισμα των μηνυμάτων που αποστέλλονται για την ικανοποίηση της ερώτησης σε κάθε διάσταση. Όταν, όμως, μετρούμε το κόστος μιας ερώτησης βάσει του πλήθους των βημάτων που απαιτούνται για την ικανοποίησή της, υπολογίζουμε το πλήθος των μηνυμάτων που απαιτούνται για την ικανοποίηση της ερώτησης σε κάθε διάσταση και, έπειτα, βρίσκουμε το μέγιστο αυτών των τιμών. Συνεπώς, μία ερώτηση διαστήματος q που επιλύεται με εφαρμογή του παραπάνω αλγορίθμου απαιτεί $O(\log N + K_{a_i})$ ή $O(\log N + N \times s_{a_i})$ βήματα, όπου a_i είναι η διάσταση για την οποία η ποσότητα $O(\log N + N \times s_{a_i})$ είναι μέγιστη.

Συμπεραίνουμε, λοιπόν, πως όταν εφαρμόζεται ο παραπάνω αλγόριθμος, το πλήθος των βημάτων που απαιτούνται για να καταλήξει μία ερώτηση διαστήματος σε όλους τους κόμβους με τα κατάλληλα δεδομένα είναι κατά κάποιο τρόπο μία γραμμική συνάρτηση του m, αφού $N = O(2^m)$, ενώ το πλήθος των μηνυμάτων που απαιτούνται για να καταλήξει μία ερώτηση διαστήματος στους κόμβους με τα κατάλληλα δεδομένα αυξάνεται γραμμικά με το πλήθος M των διαστάσεων των δεδομένων του συστήματος και, επομένως, δεν μπορεί να εφαρμοστεί αποδοτικά σε συστήματα στα οποία τα εισαγόμενα δεδομένα είναι πολλών διαστάσεων. Επιπρόσθετα, είναι αναμενόμενο πως δε θα είναι χρήσιμα όλα τα δεδομένα που θα κληθεί να επιστρέψει ο αλγόριθμος. Όλα αυτά οδηγούν στη σχεδίαση ενός γρηγορότερου αλγορίθμου, που προτείνεται στην ακόλουθη ενότητα.

5.5.2.2. Επεξεργασία Ερωτήσεων σε μία μόνο Διάσταση

Το τελικό αποτέλεσμα μίας ερώτησης διαστήματος ως προς M χαρακτηριστικά αποτελείται από δεδομένα που ικανοποιούν τις συνθήκες που ορίζονται για κάθε ένα από τα M χαρακτηριστικά. Αν, λοιπόν, X το σύνολο των δεδομένων που ικανοποιούν όλες τις

συνθήκες της ερώτησης και X_i το σύνολο των δεδομένων που ικανοποιούν μόνο τη συνθήκη για το χαρακτηριστικό a_i, όπου $1 \le i \le M$, τότε εύκολα αντιλαμβανόμαστε ότι $X = \bigcap_{i=1}^{M} X_i$ και κάθε X_i αποτελεί υπερσύνολο του X. Αντί, λοιπόν, να εκτελούμε M ερωτήσεις υπολογίζοντας ένα υπερσύνολο του X για κάθε μία από τις διαστάσεις των δεδομένων του συστήματος, μπορούμε να εκτελέσουμε μία από αυτές τις ερωτήσεις επιλέγοντας μία από τις διαστάσεις, π.χ. τη διάσταση a_k, και να ελέγξουμε ποια από τα δεδομένα που επιστρέφει ικανοποιούν όλες τις συνθήκες της αρχικής ερώτησης q.

Τα παραπάνω μας οδηγούν στη σχεδίαση ενός νέου αλγορίθμου επίλυσης ερωτήσεων διαστήματος που περιγράφεται στη συνέχεια. Έστω n ο κόμβος που εκκινεί μία ερώτηση διαστήματος q για M χαρακτηριστικά. Υπενθυμίζουμε ότι η ερώτηση διαστήματος ορίζεται για κάθε χαρακτηριστικό ως εξής: $v_{il} \le \alpha_i \le v_{iu}$, $1 \le i \le M$, όπου v_{il} και v_{iu} το κάτω και το άνω όριο που προσδιορίζουν το διάστημα στο οποίο πρέπει να ανήκουν οι τιμές του χαρακτηριστικού i των δεδομένων. Έστω, όπως είπαμε, ότι επιλέγουμε να εκτελέσουμε την ερώτηση για το χαρακτηριστικό a_k. Το χαρακτηριστικό αυτό ονομάζεται κυρίαρχο χαρακτηριστικό (*dominant attribute*). Ο κόμβος n εφαρμόζει το γενικό αλγόριθμο επίλυσης ερωτήσεων διαστήματος που περιγράφηκε παραπάνω, με κάποια τροποποίηση. Χρησιμοποιεί, λοιπόν, τον αλγόριθμο δρομολόγησης του συστήματος του Chord και δρομολογεί την ερώτηση στον κόμβο με αναγνωριστικό n_l, που είναι ο διάδοχος του $H(v_{kl})$ $\left(succ\left(H(v_{kl})\right)\right)$, δηλαδή ο διάδοχος της τιμής του κάτω ορίου που ορίζεται για το χαρακτηριστικό a_k της ερώτησης διαστήματος q. Στην ερώτηση που προωθείται στο δακτύλιο περιέχεται το κλειδί $k = H(v_{kl})$ που αναζητείται με τον αλγόριθμο δρομολόγησης του Chord και το ζητούμενο εύρος $[v_{il}, v_{iu}]$ για κάθε διάσταση $1 \le i \le M$. Όταν ο κόμβος n_l λάβει την ερώτηση, εξετάζει τα δεδομένα που διατηρεί τοπικά και αποφασίζει ποια θα επιστραφούν στον κόμβο n. Στη συνέχεια, ο κόμβος n_l ελέγχει αν ο ίδιος είναι και διάδοχος του $H(v_{ku})$, δηλαδή της τιμής που έχει αντιστοιχιστεί στο άνω όριο του ζητούμενου εύρους για το κυρίαρχο χαρακτηριστικό. Αν κάτι τέτοιο ισχύει, η προώθηση της ερώτησης σταματά. Στην αντίθετη περίπτωση, ο n_l προωθεί την ερώτηση στον επόμενό του στο δακτύλιο,

δηλαδή το διάδοχό του, έστω n_j. Ο n_j με τη σειρά του εξετάζει τα δεδομένα που διατηρεί αποφασίζοντας ποια από αυτά πρέπει να επιστραφούν στον κόμβο n και προωθεί την ερώτηση στον επόμενό του, έως ότου η ερώτηση καταλήξει στον κόμβο n_u, που είναι ο διάδοχος του $H(v_{ku})$ $\left(successor\left(H(v_{ku})\right)\right)$.

Κάθε κόμβος που εξετάζει τα δεδομένα που διατηρεί τοπικά έτσι ώστε να αποφασίσει ποια από αυτά πρέπει να επιστραφούν στον κόμβο που εκκίνησε την ερώτηση διαστήματος λαμβάνει την απόφαση ακολουθώντας δύο προσεγγίσεις.

Στην πρώτη προσέγγιση, ο κόμβος ελέγχει μόνο το κυρίαρχο χαρακτηριστικό a_k των δεδομένων. Για το λόγο αυτό, στην ερώτηση διαστήματος περιέχεται μόνο η συνθήκη για τη διάσταση a_k. Επομένως, ένα δεδομένο επιστρέφεται στον κόμβο n αν ικανοποιείται η αντίστοιχη συνθήκη. Ο κόμβος n, που λαμβάνει τα δεδομένα, είναι εκείνος που θα ελέγξει τις συνθήκες για τα υπόλοιπα χαρακτηριστικά (*Single Attribute Dominated Query Resolution with Redundant Data*). Στην περίπτωση αυτή, ο n πιθανόν να λάβει και δεδομένα που δε θα ανήκουν στην τελική απάντηση της ερώτησης διαστήματος q. Παρατηρούμε, λοιπόν, πως το πλήθος των μηνυμάτων που απαιτούνται για την επιστροφή των κατάλληλων δεδομένων στον κόμβο n, επομένως και το πλήθος των βημάτων που απαιτούνται για τον ίδιο λόγο είναι αυξημένο.

Στη δεύτερη προσέγγιση, κάθε κόμβος που καλείται να αποφασίσει ποια δεδομένα πρέπει να επιστραφούν στον κόμβο n ελέγχει τοπικά όλα τα χαρακτηριστικά των δεδομένων (*Single Attribute Dominated Query Resolution*). Επομένως, στην ερώτηση διαστήματος περιέχονται όλες οι συνθήκες που ορίζονται στην ερώτηση διαστήματος, για κάθε μία από τις διαστάσεις που ορίζονται. Στην περίπτωση αυτή, τα μηνύματα που μεταφέρουν τις απαντήσεις περιέχουν μόνο δεδομένα που ικανοποιούν όλες τις συνθήκες της αρχικής ερώτησης q και, επομένως, ανήκουν στην τελική απάντηση.

Αναμένουμε, λοιπόν, πως το πλήθος των βημάτων και των μηνυμάτων που απαιτούνται για να βρεθούν τα κατάλληλα δεδομένα, αλλά όχι και για να επιστραφούν στον κόμβο που εκκίνησε την ερώτηση διαστήματος, παραμένει ίδιο, όποια προσέγγιση κι αν ακολουθηθεί.

Όταν, λοιπόν, εφαρμόζεται ο παραπάνω αλγόριθμος, οι αναμενόμενες τιμές του κόστους γίνονται $O(\log N + K_k)$ ή $O(\log N + N \times s_k)$ και $O(\log N + K_k)$ ή $O(\log N + N \times s_k)$, αντίστοιχα, για να καταλήξει μία ερώτηση διαστήματος σε όλους τους κόμβους που πρέπει να επιστρέψουν δεδομένα στον κόμβο n, όπου K_k το πλήθος των κόμβων που ανήκουν στο εύρος που ορίζεται για το κυρίαρχο χαρακτηριστικό a_k και s_k η επιλεκτικότητα της ερώτησης διαστήματος για αυτό. Αν, επιπλέον, επιλέξουμε το χαρακτηριστικό a_k να είναι εκείνο για το οποίο η επιλεκτικότητα που ορίζεται στην ερώτηση είναι ελάχιστη και αν $s_{\min}$ η επιλεκτικότητα αυτή, η επίλυση μίας ερώτησης διαστήματος απαιτεί $O(\log N + N \times s_{\min})$ μηνύματα και $O(\log N + N \times s_{\min})$ βήματα. Αν, επίσης, $s_{\min} = \varepsilon$, όπου ε μία πάρα πολύ μικρή σταθερά, τότε ο όρος $N \times s_{\min}$ στον προηγούμενο τύπο παύει να είναι ο κυρίαρχος και οι αναμενόμενες τιμές κόστους γίνονται $O(\log N)$ και $O(\log N)$, αντίστοιχα.

Ενώ, όμως, το πλήθος των βημάτων και των μηνυμάτων που απαιτούνται για να βρεθούν τα κατάλληλα δεδομένα, αλλά όχι και να επιστραφούν στον κόμβο που εκκίνησε την ερώτηση διαστήματος, παραμένει ίδιο όποια προσέγγιση κι αν ακολουθηθεί, το πλήθος των δεδομένων που επιστρέφονται από κάθε αλγόριθμο, επομένως, και το πλήθος των μηνυμάτων που απαιτούνται για την επιστροφή των δεδομένων διαφέρει. Πιο συγκεκριμένα, όταν ακολουθείται η πρώτη προσέγγιση, αναμένουμε ότι στον κόμβο n επιστρέφεται πληθώρα άχρηστων δεδομένων, δηλαδή δεδομένα που δεν ικανοποιούν όλες τις συνθήκες της ερώτησης διαστήματος q, με αποτέλεσμα να αυξάνεται το κόστος επικοινωνίας μεταξύ των κόμβων για την επιστροφή των δεδομένων. Όταν, όμως, ακολουθείται η δεύτερη προσέγγιση, στον κόμβο n επιστρέφονται μόνο σωστά δεδομένα, δηλαδή δεδομένα που ικανοποιούν κάθε μία από τις συνθήκες της ερώτησης διαστήματος q. Το πλήθος αυτών των δεδομένων είναι πολύ μικρότερο του αντίστοιχου πλήθους των δεδομένων που επιστρέφονται όταν ακολουθείται η πρώτη προσέγγιση.

Παρατηρούμε, λοιπόν, πως όταν εφαρμόζεται ο δεύτερος αλγόριθμος, το πλήθος των μηνυμάτων ή βημάτων που απαιτούνται για να προωθηθεί η ερώτηση σε όλους τους κόμβους που πιθανώς διαθέτουν δεδομένα που πρέπει να επιστραφούν στον κόμβο n δεν εξαρτάται από το πλήθος των διαστάσεων των δεδομένων του συστήματος. Επομένως, ο αλγόριθμος μπορεί να εφαρμοστεί και σε συστήματα στα οποία τα δεδομένα έχουν μεγάλο πλήθος

διαστάσεων. Φυσικά, όμως, οι ερωτήσεις που προωθούνται στη σύστημα για την επίλυση μιας ερώτησης διαστήματος πρέπει να εμπεριέχουν τις καθορισμένες συνθήκες για όλες τις διαστάσεις.

ΚΕΦΑΛΑΙΟ 6. ΤΟ ΣΥΣΤΗΜΑ ΤΟΥ MERCURY

6.1. Εισαγωγή

Στο κεφάλαιο αυτό παρουσιάζεται το Mercury [3], ένα ακόμη σύστημα που επιτρέπει την επίλυση ερωτήσεων διαστήματος για πολυδιάστατα δεδομένα. Στο σύστημα του Mercury οι κόμβοι οργανώνονται σε λογικές ομάδες. Για κάθε μία από τις διαστάσεις των δεδομένων που εισάγονται στο σύστημα δημιουργείται και μία ομάδα, που ονομάζεται δακτύλιος δρομολόγησης (*routing hub*). Η εισαγωγή κάποιου δεδομένου στο σύστημα πραγματοποιείται σε κάθε έναν από τους δακτυλίους που υπάρχουν, ενώ η επίλυση μίας ερώτησης λαμβάνει χώρα μέσα σε έναν μόνο δακτύλιο, κατάλληλα επιλεγμένο.

Έστω, λοιπόν, a ένα χαρακτηριστικό του συστήματος και H_a ο δακτύλιος που αντιστοιχεί στο χαρακτηριστικό αυτό. Όπως προαναφέρθηκε, οι κόμβοι που ανήκουν στο δακτύλιο H_a οργανώνονται σε ένα δακτύλιο. Κάθε ένας από αυτούς είναι υπεύθυνος για ένα εύρος τιμών r_a. Έστω, επίσης, $\pi_a(d)$ η τιμή του χαρακτηριστικού a που ορίζεται σε ένα δεδομένο d και $\pi_a(q)$ το εύρος του χαρακτηριστικού a μίας ερώτησης διαστήματος q. Ο κόμβος που

διαχειρίζεται το εύρος r_a διατηρεί όλα τα δεδομένα d για τα οποία $\pi_a(d) \in r_a$ και επιλύει όλες τις ερωτήσεις q για τις οποίες $\pi_a(q) \cap r_a \neq \varnothing$, δηλαδή όλες τις ερωτήσεις στις οποίες το οριζόμενο εύρος καλύπτεται εν μέρει ή πλήρως από το εύρος που διαχειρίζεται ο κόμβος. Τα διαστήματα που διαχειρίζονται οι κόμβοι ανατίθενται σε αυτούς κατά την είσοδό τους στο σύστημα με τρόπο που θα περιγραφεί στη συνέχεια. Η ανάθεση των δεδομένων στους συμμετέχοντες κόμβους ενός δακτυλίου δε γίνεται μέσω συναρτήσεων κατακερματισμού, αλλά βάσει των τιμών των αντίστοιχων χαρακτηριστικών των δεδομένων και των διαστημάτων τιμών για τα οποία είναι υπεύθυνοι οι κόμβοι. Ωστόσο, θα μπορούσαν να χρησιμοποιηθούν και συναρτήσεις κατακερματισμού που διατηρούν τη διάταξη, όπως αυτές που παρουσιάστηκαν στο σύστημα του MAAN. Η οργάνωση αυτή επιτρέπει την επίλυση ερωτήσεων διαστήματος στο σύστημα. Το γεγονός, όμως, πως δε χρησιμοποιείται κάποια συνάρτηση κατακερματισμού για την κατανομή των δεδομένων στο σύστημα μπορεί να οδηγήσει σε ανομοιόμορφη κατανομή των δεδομένων στους κόμβους του συστήματος. Έτσι, στο Mercury απαιτούνται μηχανισμοί εξισορρόπησης φορτίου. Για την επίτευξη των παραπάνω κάθε κόμβος δε διατηρεί παρά λογικό πλήθος γειτονικών του στο σύστημα.

6.2. Είσοδος Κόμβων στο Σύστημα

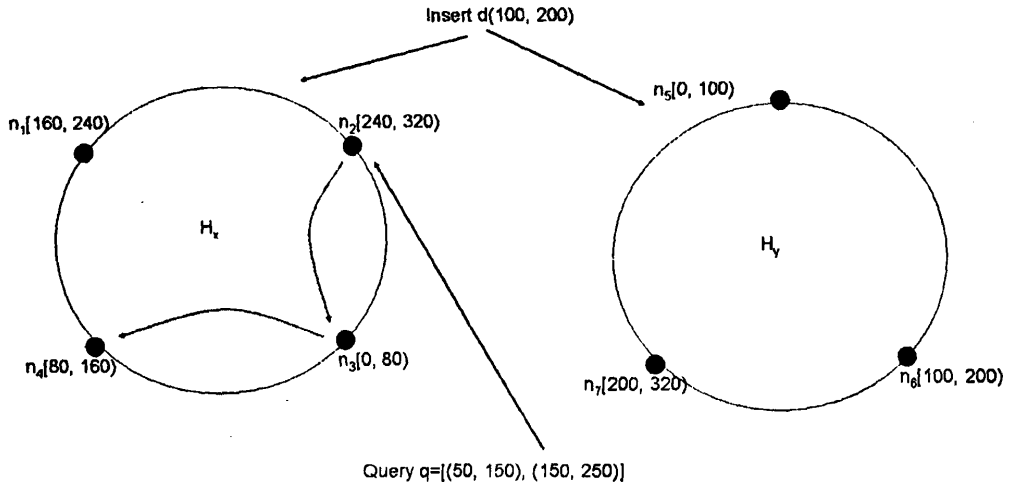
Κάθε κόμβος στο Mercury διατηρεί δείκτες στον προκάτοχο και το διάδοχό του στο δακτύλιο τον οποίο συμμετέχει, ένα δείκτη σε έναν κόμβο κάθε άλλου δακτυλίου του συστήματος (*cross-hub links*) και, επιπλέον, e μακρινές ακμές προς κόμβους του ίδιου δακτυλίου. Οι δείκτες του κόμβου στον προκάτοχο και το διάδοχό του, καθώς, επίσης, και οι μακρινές ακμές που διατηρεί αποτελούν το σύνολο των γειτονικών του κόμβων. Κατά την είσοδο, λοιπόν, ενός κόμβου στο σύστημα ή την αποχώρησή του από αυτό, οι παραπάνω δείκτες των κόμβων που συμμετέχουν στο σύστημα πρέπει να διατηρούνται συνεπείς. Στη συνέχεια περιγράφεται αναλυτικά πώς επιτυγχάνεται αυτό.

Έστω ένας νέος κόμβος n που επιθυμεί να εισέλθει στο σύστημα του Mercury, το οποίο οργανώνονται σε D δακτυλίους, δηλαδή τόσους δακτυλίους, όσες και οι διαστάσεις των δεδομένων. Ο κόμβος n επιλέγει έναν τυχαίο αριθμό $1 \leq i \leq D$, που καθορίζει σε ποιον από τους δακτυλίους του συστήματος θα εισέλθει. Ο n πρέπει να γνωρίζει κάποιον κόμβο που ήδη συμμετέχει στο Mercury. Έστω n' ο κόμβος αυτός. Ο n χρησιμοποιεί τους δείκτες προς κόμβους των άλλων δακτυλίων που διατηρεί ο n' και δρομολογεί αρχικά το αίτημα για

εισαγωγή σε τυχαίο κόμβο του δακτυλίου που επιλέχθηκε. Επιλέγεται, λοιπόν, ένα τυχαίο σημείο x στο δακτύλιο και το αίτημα για εισαγωγή του νέου κόμβου δρομολογείται στο διάδοχο κόμβο του x. Έστω $s = succ(x)$ ο κόμβος αυτός. Ο νέος κόμβος n εισέρχεται στο δακτύλιο ως προκάτοχος του s. Το εύρος τιμών για το οποίο είναι υπεύθυνος ο κόμβος s διαχωρίζεται στα δύο. Ο νεοεισερχόμενος κόμβος n γίνεται υπεύθυνος για το πρώτο μισό του διαστήματος, ενώ ο κόμβος s αποκτά πλέον την ευθύνη του δεύτερου μισού. Στη συνέχεια, ο n αντιγράφει τις δομές δρομολόγησης του s, δηλαδή τον πίνακα των μακρινών ακμών και τους δείκτες προς κόμβους άλλων δακτυλίων. Με την πάροδο του χρόνου και για να πραγματοποιείται πιο αποδοτικά η δρομολόγηση, ο n ενημερώνει τον πίνακα των μακρινών του ακμών, και ανακαλύπτει νέους κόμβους σε άλλους δακτυλίους για να συνδεθεί μαζί τους, εκκινώντας τυχαίους περιπάτους σε αυτά. Σημειώνεται πως για να μπορεί να ανακάμψει το σύστημα μετά από αποτυχίες, κάθε κόμβος διατηρεί περισσότερους από έναν δείκτες σε κόμβους κάθε άλλου δακτυλίου του συστήματος, καθώς, επίσης, και επιπλέον δείκτες προς τον προκάτοχο και το διάδοχο κόμβο. Με άλλα λόγια, ένας κόμβος γνωρίζει εκτός από τον προηγούμενο και τον επόμενό του στο δακτύλιο και κάποιους άλλους κόμβους που ακολουθούν (εφεδρικές συνδέσεις).

6.2.1. Κατασκευή των Μακρινών Ακμών

Έστω ένας κόμβος n που είναι υπεύθυνος για το διάστημα $[l,r)$ σε κάποιο δακτύλιο H_a του συστήματος και $I = [0,1]$ το μοναδιαίο διάστημα. Έστω, επίσης, m_a και M_a η ελάχιστη και η μέγιστη τιμή που μπορεί να λάβει το χαρακτηριστικό a, που αντιστοιχεί στο H_a, δηλαδή οι τιμές που καθορίζουν το χώρο αναγνωριστικών στον οποίο αντιστοιχίζονται τα δεδομένα που εισάγονται στο σύστημα. Για κάθε μακρινή ακμή που επιθυμεί να δημιουργήσει ο κόμβος n στο δακτύλιο H_a, επιλέγει έναν αριθμό $x \in I$ χρησιμοποιώντας την αρμονική κατανομή $\left(p_n(x) = \frac{1}{n \log x} if \; x \in \left[\frac{1}{n}, 1\right] \right)$. Κατόπιν, αναζητά τον κόμβο n' που είναι υπεύθυνος για την τιμή $r + (M_a - m_a)x$ στο δακτύλιο και του ζητά να γίνει γειτονικός του. Σημειώνεται ότι κάθε κόμβος δέχεται να συμμετέχει σε το πολύ $2e$ μακρινές συνδέσεις και η τιμή του e μπορεί να ορίζεται διαφορετικά για κάθε κόμβο.



Σχήμα 6.1 Σύστημα Mercury με δύο δακτυλίους.

6.3. Εισαγωγή Δεδομένων

Κάθε δεδομένο d που εισάγεται στο σύστημα αποστέλλεται σε όλους τους δακτυλίους H_α, όπου $\alpha \in A_d$. Υπενθυμίζουμε πως το A_d είναι το σύνολο των χαρακτηριστικών που ορίζονται στο δεδομένο d. Εντός κάθε δακτυλίου H_α, το δεδομένο d αποθηκεύεται στον κόμβο στου οποίου το εύρος ανήκει η τιμή του χαρακτηριστικού α. Εφόσον, λοιπόν, κάθε δεδομένο που εισάγεται στο σύστημα αποστέλλεται σε όλους τους δακτυλίους, κάθε ερώτηση διαστήματος μπορεί να επιλυθεί σε έναν μόνο δακτύλιο του συστήματος και, μάλιστα, σε οποιονδήποτε από αυτά.

Ένας άλλος σχεδιασμός θα απαιτούσε κάθε δεδομένο d να εισάγεται σε έναν μόνο δακτύλιο H_α, όπου $\alpha \in A_d$. Τότε, κάθε ερώτηση διαστήματος θα έπρεπε να προωθείται σε όλους τους δακτυλίους που αντιστοιχούν σε χαρακτηριστικά που ορίζονται στην ερώτηση. Ωστόσο, ερωτήσεις διαστήματος με μεγάλη επιλεκτικότητα ως προς κάποιο χαρακτηριστικό θα μπορούσαν να προκαλέσουν αποστολή μεγάλου πλήθους μηνυμάτων στον αντίστοιχο δακτύλιο και να υπερφορτωθεί. Για το λόγο αυτό προτιμάται ο σχεδιασμός που προτάθηκε στην προηγούμενη παράγραφο.

Όταν εισάγουμε ένα δεδομένο d σε έναν δακτύλιο H_a, αναζητούμε τον κόμβο στο εύρος του οποίου ανήκει η τιμή $\pi_a(d)$ για να αποθηκεύσουμε το d. Για την αναζήτηση χρησιμοποιούνται οι δομές δρομολόγησης που διατηρούνται από τους κόμβους του

συστήματος, δηλαδή οι δείκτες στον προκάτοχο και το διάδοχο ενός κόμβου στο δακτύλιο τον οποίο συμμετέχει, οι δείκτες σε κόμβους άλλων δακτυλίων του συστήματος, καθώς, επίσης, και ο πίνακας των μακρινών ακμών, που είναι ανάλογες με αυτές που διατηρεί το σύστημα του Chord, ενώ η διαδικασία δρομολόγησης είναι διαφορετική από αυτή που εφαρμόζεται στο Chord και περιγράφεται στη συνέχεια. Στο Σχήμα 6.1 βλέπουμε ένα σύστημα Mercury με δύο δακτυλίους, τους H_x και H_y. Το εύρος και των δύο δακτυλίων είναι το $[0,320]$. Δίπλα σε κάθε κόμβο φαίνεται το εύρος τιμών για το οποίο είναι υπεύθυνος. Η εισαγωγή του δεδομένου $d=(100,200)$ γίνεται και στους δύο δακτυλίους. Μάλιστα, στο δακτύλιο H_x το δεδομένο εισάγεται στον κόμβο n_4 αφού $100 \in [80,160)$, ενώ στο δακτύλιο H_y το δεδομένο εισάγεται στον κόμβο n_7 αφού $200 \in [200,320)$.

6.4. Επίλυση Απλών Ερωτήσεων

Το γεγονός πως κάθε δεδομένο d που εισάγεται στο σύστημα αποστέλλεται σε όλους τους δακτυλίους H_α, όπου $\alpha \in A_d$, καθιστά δυνατή την επίλυση κάθε απλής ερώτησης με την εκτέλεσή της σε έναν μόνο δακτύλιο του συστήματος. Επομένως, όταν ένας κόμβος αναζητεί κάποιο δεδομένο, προωθεί μία απλή ερώτηση μόνο σε έναν από τους δακτυλίους του συστήματος. Ο δακτύλιος αυτός επιλέγεται ανάμεσα στους δακτυλίους που αντιστοιχούν σε χαρακτηριστικά για τα οποία ορίζεται κάποια τιμή στην ερώτηση. Με άλλα λόγια, μία απλή ερώτηση προωθείται σε έναν δακτύλιο H_a, όπου a οποιοδήποτε από τα χαρακτηριστικά του συνόλου A_q. Υπενθυμίζουμε πως A_q είναι το σύνολο των χαρακτηριστικών που ορίζονται σε μία ερώτηση q. Η επιλογή του δακτυλίου γίνεται τυχαία. Είναι, βέβαια, πιθανόν ένας κόμβος να προωθήσει την ερώτηση σε διαφορετικό δακτύλιο από αυτό στο οποίο ο ίδιος ανήκει. Αυτό γίνεται μόνο κατά το πρώτο βήμα επίλυσης της ερώτησης. Στη συνέχεια, η αναζήτηση εξελίσσεται εντός του ιδίου δακτυλίου με τον τρόπο που περιγράφεται στη συνέχεια.

Για μία απλή ερώτηση q που εκτελείται στο δακτύλιο H_a, η $\pi_a(q)$ είναι μία απλή τιμή. Κατά την επίλυσή της, λοιπόν, αναζητούμε τον κόμβο που είναι υπεύθυνος για την τιμή $\pi_a(q)$. Για να πραγματοποιείται αποδοτικά η δρομολόγηση, κάθε κόμβος στο Mercury χρησιμοποιεί τις e μακρινές ακμές (*long-distance links*) προς άλλους κόμβους του ίδιου δακτυλίου που προαναφέρθηκαν.

Έστω, λοιπόν, ένας κόμβος n που είναι υπεύθυνος για το διάστημα $[l,r)$ και $dist$ η απόσταση δύο κόμβων κατά τη φορά των δεικτών του ρολογιού. Αν m_a και M_a η ελάχιστη και η μέγιστη τιμή που μπορεί να λάβει ένα χαρακτηριστικό a, όπως προαναφέραμε, τότε η απόσταση μεταξύ δύο κόμβων n_1 και n_2 ορίζεται ως εξής:

$dist(n_1,n_2)=n_2-n_1$ αν $n_1 \le n_2$ & $dist(n_1,n_2)=(M_a-m_a)+(n_2-n_1)$ αν $n_1 > n_2$.	Εξ. 5.1

Όταν ένας κόμβος n αναζητά μία τιμή v, εξετάζει όλους τους γειτονικούς του κόμβους n_i, κάθε ένας από τους οποίους είναι υπεύθυνος για το διάστημα $[l_i,r_i)$, στους οποίους διατηρεί δείκτες. Ο κόμβος n υπολογίζει την απόσταση $dist(l_i,v)$ για κάθε έναν από τους κόμβους αυτούς και προωθεί την ερώτηση σε εκείνον το γειτονικό του κόμβο που ελαχιστοποιεί την απόσταση αυτή. Η διαδικασία συνεχίζεται μέχρι η απλή ερώτηση να καταλήξει στον κόμβο στου οποίου το εύρος ανήκει η τιμή που αναζητείται. Η διαδικασία αυτή είναι αντίστοιχη της διαδικασίας δρομολόγησης που ακολουθείται στο σύστημα του Chord, στο οποίο ένας κόμβος n που αναζητά μία τιμή v εξετάζει τους γειτονικούς του κόμβους στον πίνακα δρομολόγησης που διατηρεί και προωθεί την ερώτηση σε εκείνον το γειτονικό του κόμβο του οποίου το κλειδί απέχει κατά την ελάχιστη απόσταση από το ζητούμενο κλειδί.

Αν υποθέσουμε ότι τα διαστήματα για τα οποία είναι υπεύθυνοι οι κόμβοι στο σύστημα του Mercury είναι ομοιόμορφα, τότε αποδεικνύεται ότι η δρομολόγηση μίας απλής ερώτησης εντός ενός δακτυλίου απαιτεί $O\left(\frac{1}{e}\log^2 N\right)$ βήματα με μεγάλη πιθανότητα. Επειδή, επιπλέον, για να οδηγηθεί η ερώτηση σε έναν δακτύλιο απαιτείται το πολύ ένα ακόμη βήμα, η δρομολόγηση μίας απλής ερώτησης απαιτεί $O\left(\frac{1}{e}\log^2 N\right)$ βήματα συνολικά [3, 18, 20].

6.5. Επίλυση Ερωτήσεων Διαστήματος

Δυνατή, όμως, είναι και η επίλυση κάθε ερώτησης διαστήματος με την εκτέλεσή της σε έναν μόνο δακτύλιο του συστήματος, αφού κάθε δεδομένο d που εισάγεται στο σύστημα αποστέλλεται σε όλους τους δακτυλίους H_α, όπου $\alpha \in A_d$. Και στην περίπτωση αυτή, ο δακτύλιος στον οποίο θα εκτελεστεί η ερώτηση διαστήματος επιλέγεται ανάμεσα στους δακτυλίους που αντιστοιχούν σε χαρακτηριστικά για τα οποία ορίζεται κάποιο εύρος ή κάποια τιμή στην ερώτηση, δηλαδή σε έναν δακτύλιο H_a, όπου a οποιοδήποτε από τα χαρακτηριστικά του συνόλου A_q. Η επιλογή του δακτυλίου μπορεί να γίνει είτε τυχαία, όπως και στην περίπτωση μίας απλής ερώτησης, είτε εξετάζοντας την επιλεκτικότητα της ερώτησης διαστήματος και επιλέγοντας το δακτύλιο που αντιστοιχεί στη διάσταση της οποίας το οριζόμενο εύρος στην ερώτηση διαστήματος είναι ελάχιστο. Αυτό γίνεται για να ελαχιστοποιηθεί το πλήθος των κόμβων στους οποίους πρέπει να προωθηθεί η ερώτηση προκειμένου να επιλυθεί.

Γεννάται, όμως, το ζήτημα της επιλογής του δακτυλίου που αντιστοιχεί στο χαρακτηριστικό με την ελάχιστη επιλεκτικότητα. Μια δυνατή λύση παρέχεται από τη χρήση ιστογραμμάτων, που κατασκευάζουν οι κόμβοι του συστήματος χρησιμοποιώντας τη μέθοδο της τυχαίας δειγματοληψίας, με τρόπο που περιγράφεται αναλυτικά στη συνέχεια. Πιο συγκεκριμένα, οι κόμβοι συγκεντρώνουν πληροφορίες που τους επιτρέπουν να γνωρίζουν ποιοι δακτύλιοι είναι μεγαλύτεροι από άλλους και, μάλιστα, πού εμφανίζουν μεγάλη πυκνότητα κόμβων. Βάσει αυτών των πληροφοριών, λοιπόν, ένας κόμβος μπορεί να αποφασίζει σε ποιο δακτύλιο μία ερώτηση πιθανώς θα κοστίσει λιγότερο και εκεί να την αποστείλει.

Και στην περίπτωση μίας ερώτησης διαστήματος, όμως, είναι πιθανόν ένας κόμβος να προωθήσει την ερώτηση σε διαφορετικό δακτύλιο από αυτό στο οποίο ο ίδιος ανήκει, κατά το πρώτο βήμα επίλυσης της ερώτησης. Στη συνέχεια, η αναζήτηση εξελίσσεται εντός του ιδίου δακτυλίου, καθώς η ερώτηση διαστήματος προωθείται σε όσους κόμβους θα μπορούσαν να διαθέτουν δεδομένα που ικανοποιούν την ερώτηση.

Για μία ερώτηση διαστήματος q που εκτελείται στο δακτύλιο H_a, η $\pi_a(q)$ είναι ένα ολόκληρο διάστημα. Κατά την επίλυσή της, λοιπόν, αναζητούμε τον κόμβο που είναι υπεύθυνος για την πρώτη τιμή αυτού του διαστήματος και, στη συνέχεια, προωθούμε σε

διαδοχικούς κόμβους του δακτυλίου, που είναι υπεύθυνοι για διαδοχικά διαστήματα τιμών, μέχρι να καλυφθεί το ζητούμενο εύρος. Η εύρεση του κόμβου που είναι υπεύθυνος για την πρώτη τιμή αυτού του διαστήματος γίνεται όπως ακριβώς και η εύρεση ενός κόμβου που είναι υπεύθυνος για την τιμή που αναζητείται κατά την επίλυση μίας απλής ερώτησης, με τον τρόπο που περιγράφηκε παραπάνω. Στη συνέχεια, η ερώτηση διαστήματος προωθείται σε όσους κόμβους θα μπορούσαν να διαθέτουν δεδομένα που ικανοποιούν την ερώτηση και ανήκουν στον ίδιο δακτύλιο. Η προώθηση γίνεται μέσω των δεικτών σε διαδόχους κόμβους που διατηρούνται από τους κόμβους του συστήματος.

Αν υποθέσουμε ότι τα διαστήματα για τα οποία είναι υπεύθυνοι οι κόμβοι στο σύστημα του Mercury είναι ομοιόμορφα, τότε αποδεικνύεται ότι η επίλυση μίας ερώτησης διαστήματος εντός ενός δακτυλίου απαιτεί $O\left(\frac{1}{e}\log^2 N\right)$ βήματα για την εύρεση του κόμβου που είναι υπεύθυνος για την πρώτη τιμή που εμφανίζεται στο ζητούμενο εύρος και K επιπλέον βήματα για την εύρεση των υπολοίπων κόμβων που μπορούν να διαθέτουν δεδομένα που ικανοποιούν την ερώτηση διαστήματος, όπου K είναι το πλήθος των κόμβων που είναι υπεύθυνοι για διαστήματα τιμών που καλύπτουν εν μέρει ή πλήρως το ζητούμενο εύρος [3, 18, 20].

Στο Σχήμα 6.1 βλέπουμε ένα παράδειγμα επίλυσης μίας ερώτησης διαστήματος. Η ερώτηση $q=\left[(50,150),(150,250)\right]$ αποστέλλεται στο δακτύλιο H_x. Η προώθησή της ξεκινά από τον κόμβο n_2, ο οποίος την προωθεί στον κόμβο n_3 του οποίου το εύρος καλύπτει ένα μέρος του ζητούμενου εύρους τιμών και, επομένως, εξετάζει τα δεδομένα που διαθέτει τοπικά. Στη συνέχεια, ο n_3 προωθεί την ερώτηση στον κόμβο n_4, ο οποίος, επίσης, εξετάζει τα δεδομένα που διατηρεί.

6.6. Αποχώρηση Κόμβου από το Σύστημα

Κατά την αποχώρηση ενός κόμβου από το σύστημα του Mercury, οι κόμβοι που διατηρούν κάθε είδους δείκτες προς αυτόν πρέπει να ενημερωθούν.

Όταν κάποιος κόμβος ανακαλύψει ότι ο διάδοχός του στο δακτύλιο έχει αποτύχει ή αποχωρήσει από το σύστημα, βρίσκει ποιος είναι ο κόμβος που τον ακολουθεί στο δακτύλιο

χρησιμοποιώντας τις εφεδρικές πληροφορίες που διατηρεί και αντικαθιστά το δείκτη προς το διάδοχο.

Επίσης, όταν ένας κόμβος ανακαλύψει ότι κάποια από τις μακρινές του ακμές τον συνδέει με κόμβο που δε συμμετέχει πλέον στο σύστημα, απλώς δημιουργεί μία νέα μακρινή ακμή με τον τρόπο που περιγράφηκε παραπάνω. Αξίζει να σημειωθεί πως μαζικές αποτυχίες ή αποχωρήσεις κόμβων μπορούν να επηρεάσουν την κατανομή των μακρινών ακμών των κόμβων του συστήματος. Για να παραμένει, λοιπόν, καλή η κατανομή των μακρινών ακμών και να πραγματοποιείται αποδοτικά η δρομολόγηση, οι κόμβοι του συστήματος περιοδικά ανακατασκευάζουν τις μακρινές ακμές τους. Μία τέτοια διαδικασία, βέβαια, μπορεί να εκκινείται μόνο όταν το πλήθος των κόμβων που συμμετέχουν στο σύστημα αλλάξει δραματικά. Με άλλα λόγια, κάθε κόμβος προσπαθεί περιοδικά να εκτιμήσει το πλήθος των κόμβων που συμμετέχουν στο σύστημα. Αν γίνει αντιληπτό ότι το πλήθος αυτό διπλασιάστηκε ή υποδιπλασιάστηκε, οι κόμβοι εκκινούν τη διαδικασία ανακατασκευής των μακρινών τους ακμών.

Τέλος, όταν κάποιος κόμβος αντιληφθεί ότι μία από τις συνδέσεις που διατηρεί με κόμβο κάποιου άλλου δακτυλίου έχει αποτύχει, τότε μπορεί να χρησιμοποιήσει τις εφεδρικές συνδέσεις που διατηρεί προς κάποιον άλλο κόμβο του ίδιου δακτυλίου και εκκινώντας έναν τυχαίο περίπατο στο δακτύλιο να ανακαλύψει ένα νέο κόμβο για να συνδεθεί μαζί του. Αν τέτοιου είδους εφεδρικές πληροφορίες δεν είναι διαθέσιμες, ζητά τις αντίστοιχες πληροφορίες των προκατόχου ή του διαδόχου του στο δακτύλιο. Αν και στην περίπτωση αυτή δεν είναι δυνατή η δημιουργία μίας νέας σύνδεσης, ο κόμβος μπορεί να επικοινωνήσει με τον κόμβο μέσω του οποίου εισήλθε στο σύστημα και να λάβει από αυτόν τις πληροφορίες που θα του επιτρέψουν να αντικαταστήσει τη σύνδεση που έχει αποτύχει.

6.7. Εξισορρόπηση φορτίου

Όταν ένας κόμβος εισέρχεται στο σύστημα, αναλαμβάνει την ευθύνη ενός εύρους τιμών. Καθώς, όμως, κάποια διαστήματα τιμών μπορούν να είναι πιο δημοφιλή από άλλα, οι κόμβοι που είναι υπεύθυνοι για αυτά καλούνται να απαντήσουν σε περισσότερες ερωτήσεις, με αποτέλεσμα να υπερφορτώνονται. Για το λόγο αυτό, το Mercury χρησιμοποιεί μία πρωτότυπη τεχνική τυχαίας δειγματοληψίας, που επιτρέπει στους κόμβους να συγκεντρώνουν στατιστικά

στοιχεία για το σύστημα. Η ιδέα είναι να συγκεντρώνουν οι κόμβοι πληροφορίες για κάποιο μέρος του συστήματος και να τις ανταλλάσσουν μεταξύ τους. Οι πληροφορίες που συγκεντρώνονται χρησιμοποιούνται από τους κόμβους για την κατασκευή ιστογραμμάτων (π.χ. για την κατανομή του φορτίου των κόμβων ή την κατανομή των κόμβων). Τα ιστογράμματα αυτά βοηθούν στην εξισορρόπηση του φορτίου μεταξύ των κόμβων του συστήματος.

Κάθε κόμβος στο Mercury μπορεί να υπολογίσει το μέσο φορτίο $\overline{L}$ του συστήματος χρησιμοποιώντας τα ιστογράμματα που έχει κατασκευάσει. Επίσης, μπορεί να υπολογίσει και το τοπικό φορτίο ως το μέσο φορτίο του ίδιου, του προκατόχου και του διαδόχου του. Μπορεί, επομένως, βάσει αυτών των πληροφοριών να προσδιορίσει αν είναι υπερφορτωμένος ή όχι. Ένας κόμβος θεωρεί ότι είναι υπερφορτωμένος αν ο λόγος του τοπικού φορτίου προς το μέσο φορτίο του συστήματος είναι μεγαλύτερος από μία παράμετρο λ και ελαφρά φορτωμένος αν ο λόγος αυτός είναι μικρότερος από $1/\lambda$. Με μεγάλη πιθανότητα, αν ένας κόμβος είναι ελαφρά φορτωμένος, τότε και οι άμεσοι γείτονές του θα είναι ελαφρά φορτωμένοι. Αν κάτι τέτοιο, όμως, δεν ισχύει, τότε ο υπερφορτωμένος κόμβος παραδίδει μέρος του εύρους που διαχειρίζεται στον ελαφρά φορτωμένο κόμβο. Αν και αυτό δεν είναι δυνατό να γίνει καθώς και οι τρεις κόμβοι είναι υπερφορτωμένοι, ο κόμβος χρησιμοποιεί τα ιστογράμματα που διαθέτει και του επιτρέπουν να γνωρίζει ποια σημεία του συστήματος είναι υπερφορτωμένα και ποια όχι. Χρησιμοποιώντας, λοιπόν, τις πληροφορίες αυτές ένας υπερφορτωμένος κόμβος επικοινωνεί με έναν ελαφρά φορτωμένο κόμβο του συστήματος και του ζητά να αποχωρήσει από το σημείο του συστήματος στο οποίο έχει εισέλθει και να εισέλθει εκ νέου στο σημείο όπου έχει παρατηρηθεί υπερφόρτωση. Με αυτόν τον τρόπο το πρόβλημα επιλύεται. Στη συνέχεια, περιγράφεται αναλυτικότερα η τεχνική τυχαίας δειγματοληψίας που εφαρμόζεται στο σύστημα.

6.7.1. Τυχαία Δειγματοληψία

Όπως προαναφέρθηκε, το σύστημα του Mercury χρειάζεται μία τεχνική τυχαίας δειγματοληψίας, που θα επιτρέπει στους κόμβους να συγκεντρώνουν στατιστικά στοιχεία για το σύστημα.

Η τυχαία δειγματοληψία πραγματοποιείται με κάποιου είδους τυχαίο περίπατο στο σύστημα και συλλέγοντας πληροφορίες από όπου ο περίπατος αυτός περνάει. Ξεκινά, λοιπόν, η διαδικασία από κάποιον κόμβο του συστήματος ο οποίος αποστέλλει ένα μήνυμα με μικρό (π.χ. $\log N$) TTL. Κάθε κόμβος που λαμβάνει το μήνυμα επιλέγει έναν τυχαίο γειτονικό του και το προωθεί, μειώνοντας το μετρητή TTL. Ο κόμβος στον οποίο ο μετρητής μηδενίζεται αποστέλλει πίσω ένα δείγμα. Το δείγμα αυτό καθορίζεται από το μετρικό του συστήματος για το οποίο ο κόμβος που εκκίνησε τη δειγματοληψία συγκεντρώνει πληροφορίες. Αξίζει να σημειώσουμε πως οι κόμβοι που συμμετέχουν στη διαδικασία της δειγματοληψίας χρησιμοποιούν μόνο τοπικές πληροφορίες. Τα μηνύματα αυτά μπορούν να μεταφέρονται με τα μηνύματα που ανταλλάσσουν περιοδικά οι κόμβοι για τον έλεγχο των συνδέσεων.

6.7.1.1. Διατήρηση Προσεγγιστικών Ιστογραμμάτων

Ο αλγόριθμος τυχαίας δειγματοληψίας που μόλις περιγράφηκε μπορεί να χρησιμοποιηθεί από τους κόμβους του Mercury για τη συγκέντρωση στατιστικών στοιχείων του συστήματος και τη δημιουργία ιστογραμμάτων (π.χ. για την κατανομή του φορτίου των κόμβων ή την κατανομή των κόμβων). Υπενθυμίζουμε ότι η ιδέα είναι να συγκεντρώνουν οι κόμβοι πληροφορίες για κάποιο μέρος του συστήματος και να τις ανταλλάσσουν μεταξύ τους.

Έστω, λοιπόν, N_{ds} η τοπική ds – γειτονιά ενός κόμβου, δηλαδή το σύνολο όλων των κόμβων που βρίσκονται σε απόσταση το πολύ ds από τον κόμβο, χωρίς να συμπεριλάβουμε τις μακρινές ακμές που διατηρούνται. Κάθε κόμβος ζητά περιοδικά πληροφορίες από τους κόμβους του συνόλου N_{ds} και παράγει μία τοπική εκτίμηση του στατιστικού στοιχείου που τον ενδιαφέρει.

Επιπλέον, ο κόμβος μπορεί περιοδικά να ζητάει δείγματα από k_1 κόμβους με τη μέθοδο της τυχαίας δειγματοληψίας που περιγράφηκε παραπάνω. Κάθε ένας από αυτούς τους k_1 κόμβους αποστέλλει πίσω όχι μόνο την τοπική του εκτίμηση, αλλά και τις k_2 πιο πρόσφατες εκτιμήσεις που ο ίδιος έχει λάβει. Το σύνολο όλων των εκτιμήσεων που έχει λάβει ο κόμβος χρησιμοποιείται για τη δημιουργία του ιστογράμματος. Ο καθορισμός των παραμέτρων k_1 και k_2 γίνεται λαμβάνοντας υπόψη την ακρίβεια που επιθυμούμε να έχει το ιστόγραμμα και το κόστος με το οποίο μπορούμε να επιβαρύνουμε το σύστημα. Πειραματικές μετρήσεις,

πάντως, έχουν δείξει πως αν θέσουμε τις παραμέτρους k_1 και k_2 ίσες με $\log N$, τα ιστογράμματα που προκύπτουν είναι αρκετά ακριβή, ενώ το σύστημα δεν επιβαρύνεται υπερβολικά.

Τα ιστογράμματα που κατασκευάζονται με τον παραπάνω τρόπο μπορούν να χρησιμοποιηθούν από τους κόμβους για τη δημιουργία αποδοτικότερων μακρινών ακμών. Πιο συγκεκριμένα, αρχικά, υπολογίζεται προσεγγιστικά το πλήθος N των κόμβων που συμμετέχουν στο σύστημα. Κατόπιν, για κάθε μία μακρινή ακμή που επιθυμούμε να κατασκευάσουμε, δημιουργούμε μία τιμή n_i στο διάστημα $[1, N]$ μέσω της αρμονικής κατανομής. Η τιμή αυτή μας δείχνει πόσους κόμβους στο δακτύλιο πρέπει να προσπεράσουμε έτσι ώστε να οδηγηθούμε στον κόμβο με τον οποίο πρέπει να συνδεθούμε. Το ιστόγραμμα χρησιμοποιείται για τον υπολογισμό της τιμής v_i για την οποία θα είναι υπεύθυνος ο νέος γειτονικός κόμβος. Συνεπώς, δρομολογείται μία ερώτηση για την τιμή v_i βάσει του πρωτοκόλλου του συστήματος. Με τον τρόπο αυτό, οι κόμβοι καταφέρνουν να δημιουργούν αποδοτικότερα μακρινές ακμές.

ΚΕΦΑΛΑΙΟ 7. ΤΟ ΣΥΣΤΗΜΑ ΤΟΥ TREE-CHORD

7.1. Εισάγωγή

Στο κεφάλαιο αυτό περιγράφεται το σύστημα του TreeChord. Το TreeChord είναι ένα ακόμη σύστημα που επιτρέπει τη διαχείριση πολυδιάστατων δεδομένων, καθώς, επίσης, και την επίλυση ερωτήσεων διαστήματος. Οι κόμβοι που συμμετέχουν στο σύστημα διατηρούν τις ίδιες πληροφορίες δρομολόγησης που διατηρούν και οι κόμβοι στο Chord, δηλαδή οι πίνακες δρομολόγησης (finger tables) παραμένουν ίδιοι. Ακόμη, η εισαγωγή κόμβων στο σύστημα, καθώς και η αποχώρησή τους από αυτό γίνεται όπως ακριβώς και στο Chord. Εκείνο που αλλάζει είναι η διαχείριση των δεδομένων, δηλαδή ο τρόπος αποθήκευσης και αναζήτησης των δεδομένων αυτών στο σύστημα. Η κατανομή των δεδομένων γίνεται χρησιμοποιώντας μία δενδρική δομή που χωρίζει το δακτύλιο σε ίσα μέρη. Επιπρόσθετα, το TreeChord επιλύει ερωτήσεις διαστήματος βασιζόμενο σε αυτή ακριβώς τη δενδρική δομή. Το σύστημα μελετάται πειραματικά στο Κεφάλαιο 10. Η πειραματική μελέτη δείχνει πως το σύστημα του TreeChord επιτυγχάνει να κατανείμει ομοιόμορφα ή σχεδόν ομοιόμορφα τα εισαγόμενα δεδομένα στους κόμβους, ανεξάρτητα από την κατανομή που ενδεχομένως ακολουθούν. Αξίζει να σημειωθεί πως στο TreeChord μπορούν να εφαρμοστούν και οι συναρτήσεις κατακερματισμού που εφαρμόζονται στο σύστημα του MAAN, καθώς και οι τεχνικές εξισορρόπησης φορτίου που εφαρμόζονται στο σύστημα του Mercury.

7.2. Εισαγωγή Δεδομένων

Όπως προαναφέραμε, το TreeChord χρησιμοποιεί τη δομή ενός δένδρου για να κατανείμει τα δεδομένα ανάμεσα στους κόμβους του συστήματος. Το δένδρο αυτό είναι ένα τέλειο δένδρο βάθους d στο οποίο κάθε κόμβος συσχετίζεται με μία τιμή στο εύρος τιμών του δακτυλίου του συστήματος. Δεδομένου ότι το δένδρο είναι τέλειο, έχει 2^d φύλλα. Σε κάθε διδιάστατο δεδομένο που εισάγεται στο σύστημα, ο κόμβος που εκκινεί τη διαδικασία εισαγωγής του αντιστοιχεί ένα τέτοιο δένδρο βάθους d. Έστω (k_x, k_y) ένα διδιάστατο δεδομένο. Η ρίζα του δένδρου έχει την τιμή k_x, ενώ κάθε ένα από τα 2^d φύλλα του (από αριστερά προς τα δεξιά) αντιστοιχεί στην τιμή $\left(k_x + \frac{i}{2^d} \times 2^m\right) \% 2^m$, $i = 1, \ldots, 2^d$. Θεωρούμε ότι κάθε ένα από τα φύλλα αυτά αντιστοιχεί σε ένα διάστημα τιμών $(lb, ub]$, όπου ub η τιμή που αντιστοιχεί στο συγκεκριμένο φύλλο και lb η τιμή που αντιστοιχεί στο προηγούμενο κατά σειρά φύλλο. Σημειώνεται ότι ως σειρά των φύλλων θεωρείται η ακολουθία που δημιουργείται όταν τα εξετάζουμε κυκλικά από τα αριστερά προς τα δεξιά. Είναι προφανές πως δε χρειάζεται να υπολογίσουμε τους εσωτερικούς κόμβους του δένδρου για να βρούμε ποιες τιμές αντιστοιχούν στα φύλλα του. Στο Σχήμα 7.1 βλέπουμε το δένδρο που κατασκευάζεται για το δεδομένο (k_x, k_y), ενώ στο Σχήμα 7.2 βλέπουμε πώς χωρίζεται ο δακτύλιος βάσει των τιμών που αντιστοιχίζονται στους κόμβους του δένδρου.

Παρατηρούμε πως τα φύλλα του δένδρου χωρίζουν το δακτύλιο εύρους 2^m σε 2^d ισομεγέθη τμήματα. Ένα από τα φύλλα του δένδρου θα μας οδηγήσει στον κόμβο ο οποίος θα κληθεί να αποθηκεύσει το δεδομένο (k_x, k_y). Το φύλλο αυτό είναι εκείνο στου οποίου το εύρος τιμών ανήκει η τιμή k_y. Έστω, λοιπόν, lv η τιμή του φύλλου αυτού. Ο κόμβος στον οποίο θα αποθηκευτεί το δεδομένο (k_x, k_y) είναι ο διάδοχος κόμβος του lv στο δακτύλιο, έστω $l = succ(lv)$. Ο κόμβος που εκκινεί τη διαδικασία εισαγωγής του νέου δεδομένου στο σύστημα αναζητά τον κόμβο l ακολουθώντας τον ίδιο αλγόριθμο δρομολόγησης που εφαρμόζεται και στο σύστημα του Chord. Η εύρεση του κόμβου l στον οποίο θα αποθηκευτεί το νέο δεδομένο απαιτεί, επομένως, $O(\log N)$ βήματα.

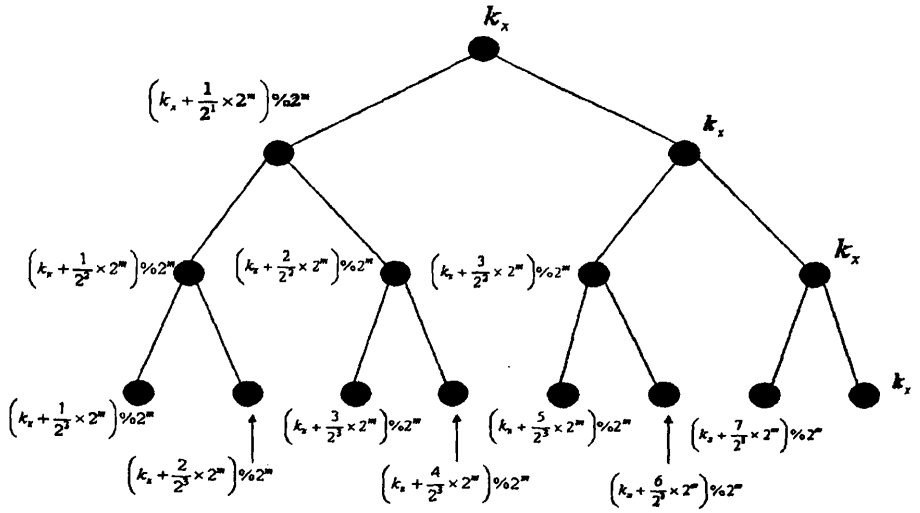
Το δένδρο, λοιπόν, που ορίζει ένα διδιάστατο δεδομένο (k_x, k_y) έχει τις ακόλουθες ιδιότητες:

α) η ρίζα του αναπαριστά την τιμή στην πρώτη διάσταση του αρχικού δεδομένου που επιχειρούμε να αποθηκεύσουμε στο σύστημα, δηλαδή την τιμή (k_x),

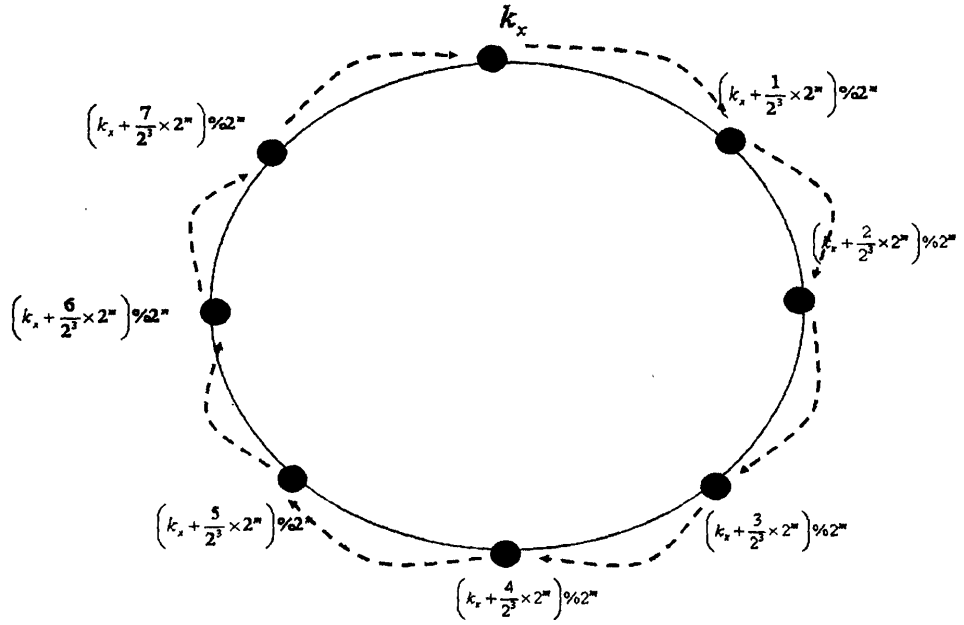
β) κάθε δεξί παιδί ενός κόμβου αναπαριστά την ίδια τιμή με τον πατέρα του στο δένδρο, αφού, αν l το επίπεδο του θυγατρικού κόμβου, τότε η τιμή που αναπαριστά ο πατέρας του είναι $k_x + \frac{\frac{i}{2}}{2^{l-1}} \times 2^m = k_x + \frac{i}{2 \times 2^{l-1}} \times 2^m = k_x + \frac{i}{2^l} \times 2^m$,

γ) κάθε αριστερό παιδί ενός κόμβου αναπαριστά μία τιμή που απέχει από την τιμή που αναπαριστά ο πατέρας του κατά $\frac{1}{2^l} \times 2^m$, όπου l είναι το βάθος του κόμβου στον οποίο αναφερόμαστε, αφού $\left(k_x + \frac{\frac{i+1}{2}}{2^{l-1}} \times 2^m \right) - \left(k_x + \frac{i}{2^l} \times 2^m \right) = -\frac{1}{2^l} \times 2^m$

δ) κάθε κόμβος σε επίπεδο βάθους l αναπαριστά μία τιμή που απέχει από την τιμή που αναπαριστά ο επόμενός του κόμβος στο ίδιο επίπεδο κατά $\frac{1}{2^l} \times 2^m$, αφού $\left(k_x + \frac{i+1}{2^l} \times 2^m \right) - \left(k_x + \frac{i}{2^l} \times 2^m \right) = -\frac{1}{2^l} \times 2^m$.



Σχήμα 7.1 Το Δένδρο που Κατασκευάζεται για το Δεδομένο (k_x, k_y) στο TreeChord.



Σχήμα 7.2 Χωρισμός του Δακτυλίου Βάσει των Τιμών των Κόμβων Του Δένδρου που Κατασκευάζεται για το Δεδομένο $\left(k_x, k_y\right)$.

Συνοψίζοντας, μπορούμε να πούμε πως κάθε διδιάστατο δεδομένο $\left(k_x, k_y\right)$ που εισάγεται στο σύστημα καθορίζει μοναδικά τις τιμές των κόμβων ενός τέλειου δένδρου βάθους d, όπως περιγράφηκε παραπάνω, βάσει του οποίου γίνεται η εισαγωγή του δεδομένου στον κατάλληλο κόμβο του συστήματος. Η τιμή του δεδομένου στην πρώτη διάσταση $\left(k_x\right)$ καθορίζει τις τιμές των κόμβων του δένδρου, ενώ η τιμή του δεδομένου στη δεύτερη διάσταση $\left(k_y\right)$ καθορίζει το φύλλο του δένδρου του οποίου η τιμή θα αποτελέσει το μονοδιάστατο κλειδί σύμφωνα με το οποίο το δεδομένο θα αποθηκευτεί στο δακτύλιο. Επιπλέον, η διαφορά μεταξύ της τιμής που αναπαριστά ένας κόμβος του δένδρου και της τιμής που αναπαριστά ο επόμενός του στο ίδιο επίπεδο του δένδρου είναι μία δύναμη του 2. Κάθε κόμβος του συστήματος διατηρεί δείκτες προς τους διαδόχους κόμβους κλειδιών που απέχουν από το δικό του κατά διαδοχικές δυνάμεις του 2. Επομένως, ο διάδοχος κόμβος ενός φύλλου του δένδρου διατηρεί δείκτη προς το διάδοχο κόμβο του επόμενου φύλλου στο ίδιο επίπεδο του δένδρου.

Τέλος, είναι αξιοσημείωτο το γεγονός πως δεν υλοποιείται κάποιο δένδρο για το εκάστοτε δεδομένο, αλλά απλά υπολογίζονται οι τιμές των φύλλων του μέσω απλών μαθηματικών

υπολογισμών. Ωστόσο, μέσω των δεικτών των πινάκων δρομολόγησης που διατηρούνται σε κάθε κόμβο, τα δένδρα αυτά σχηματίζονται στο σύστημα.

7.3. Επίλυση Απλών Ερωτήσεων

Στο σύστημα του TreeChord η επίλυση απλών ερωτήσεων πραγματοποιείται με εξίσου απλό τρόπο με αυτόν που ακολουθείται στο Chord. Η διαδικασία βασίζεται και πάλι στη δενδρική δομή που περιγράφηκε παραπάνω. Αρχικά, υπολογίζεται η τιμή του φύλλου στο οποίο αποθηκεύεται το δεδομένο. Έστω, λοιπόν, ότι κάποιος κόμβος n του συστήματος αναζητά ένα δεδομένο (k_x, k_y). Ο n υπολογίζει τις τιμές που αντιστοιχούν στα φύλλα του δένδρου, καθώς, επίσης, και την τιμή του φύλλου στο εύρος τιμών του οποίου αντιστοιχεί η τιμή k_y. Έστω l το φύλλο αυτό. Ο κόμβος n αναζητά το διάδοχο κόμβο του l $(succ(l))$ χρησιμοποιώντας τον ίδιο αλγόριθμο δρομολόγησης που εφαρμόζεται στο Chord. Η εύρεση του $succ(l)$ απαιτεί $O(\log N)$ βήματα και $O(\log N)$ μηνύματα. Όταν, τελικά, ο $succ(l)$ λάβει την ερώτηση, ελέγχει τα δεδομένα που διαθέτει τοπικά και απαντά κατάλληλα στην ερώτηση παρέχοντας το δεδομένο, αν το διαθέτει.

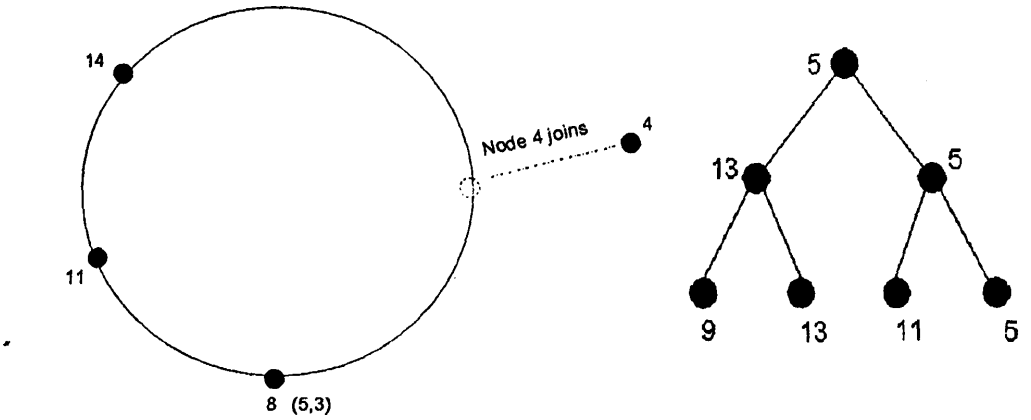
7.4. Ενημέρωση των Δεδομένων των Κόμβων Μετά την Εισαγωγή Ενός Νέου Κόμβου στο Σύστημα

Η εισαγωγή νέων κόμβων στο σύστημα του TreeChord γίνεται όπως ακριβώς και στο Chord. Το μόνο που αλλάζει είναι ο τρόπος με τον οποίο αποφασίζεται ποια δεδομένα θα λάβει προς αποθήκευση ο νέος κόμβος.

Έστω ένας νέος κόμβος n που εισέρχεται στο σύστημα του TreeChord και έστω ότι pn και sn είναι ο προκάτοχος και ο διάδοχος του n στο δακτύλιο, αντίστοιχα. Μετά την είσοδο του n στο σύστημα, κάποια από τα δεδομένα που διατηρεί ο sn πιθανόν να πρέπει πλέον να μεταφερθούν στον κόμβο n. Όταν ο sn πληροφορηθεί πως ο n είναι ο νέος του προκάτοχος στο δακτύλιο, εξετάζει ένα προς ένα τα δεδομένα που διατηρεί. Έτσι, για κάθε δεδομένο (k_x, k_y) που διατηρεί, ο sn υπολογίζει τις τιμές των φύλλων του δένδρου που περιγράφηκε παραπάνω και αποφασίζει σε ποιο από αυτά αντιστοιχεί το δεδομένο (k_x, k_y). Έστω l' το

φύλλο αυτό. Αν το l' έπεται του n στο δακτύλιο, τότε ο sn εξακολουθεί να είναι ο διάδοχος του l'. Επομένως, το δεδομένο (k_x, k_y) πρέπει να παραμείνει στον κόμβο sn. Αν, όμως το l' προηγείται του n στο δακτύλιο, τότε διάδοχος του l' στο δακτύλιο είναι πλέον ο κόμβος n. Επομένως, το δεδομένο (k_x, k_y) πρέπει να μεταφερθεί προς αποθήκευση στον νεοεισερχόμενο κόμβο n. Η μεταφορά γίνεται με κατάλληλο μήνυμα από τον sn στον n.

Δεδομένου του τρόπου αποθήκευσης των δεδομένων, παρατηρούμε πως η απλή διαδικασία που ακολουθείται στο Chord κατά την εισαγωγή ενός κόμβου στο σύστημα δεν είναι επαρκής. Αυτό εξηγείται στη συνέχεια και με ένα παράδειγμα. Με τη διαδικασία, λοιπόν, που μόλις περιγράφηκε όλα τα δεδομένα ανακατανέμονται και μεταφέρονται προς αποθήκευση στους σωστούς κόμβους του συστήματος.



Σχήμα 7.3 Ενημέρωση των Δεδομένων των Κόμβων Μετά την Εισαγωγή Ενός Νέου Κόμβου στο Σύστημα.

Στο Σχήμα 7.3 βλέπουμε ένα παράδειγμα εισαγωγής κόμβου σε σύστημα με $m=4$ και πώς αυτό επηρεάζει τα υπάρχοντα δεδομένα. Αρχικά, στο σύστημα υπάρχουν οι κόμβοι 8, 11 και 14. Το δεδομένο $(5,3)$ που έχει εισαχθεί στο σύστημα διατηρείται στον κόμβο 8. Το δένδρο που αντιστοιχεί στο δεδομένο και φαίνεται στο Σχήμα μας υποδεικνύει ότι το δεδομένο πρέπει να αποθηκευτεί στον κόμβο $succ(5)=8$. Στη συνέχεια, εισέρχεται ο κόμβος 4 στο σύστημα ως προκάτοχος του κόμβου 8. Ο κόμβος 8 υπολογίζει τις τιμές των κόμβων του δένδρου και βρίσκει ότι το δεδομένο $(5,3)$ πρέπει να αποθηκευτεί στον κόμβο $succ(5)=8$. Με άλλα λόγια, το δεδομένο παραμένει στον κόμβο 8. Αν, όμως, η ενημέρωση των δεδομένων των κόμβων γινόταν όπως στο Chord, βάσει της τιμής των δεδομένων στη

δεύτερη διάσταση, τότε ο κόμβος 8 θα έπρεπε να αποστείλει το δεδομένο $(5,3)$ προς αποθήκευση στον κόμβο $succ(3)=4$. Παρατηρούμε, λοιπόν, ότι ο απλός έλεγχος που εφαρμόζεται στο σύστημα του Chord για την ενημέρωση των δεδομένων των κόμβων μετά την είσοδο νέων κόμβων στο σύστημα δεν επαρκεί.

7.5. Επίλυση Ερωτήσεων Διαστήματος

Το σύστημα του TreeChord επιτρέπει την επίλυση ερωτήσεων διαστήματος. Οι ερωτήσεις αυτές μπορούν να αφορούν οποιαδήποτε από τις δύο διαστάσεις ή και τις δύο.

7.5.1. Επίλυση Ερωτήσεων Διαστήματος ως Προς την y Διάσταση

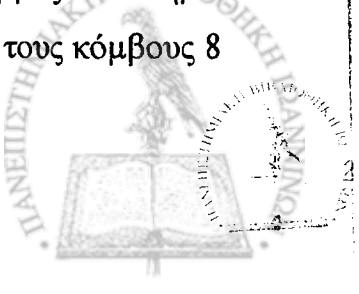
Έστω ένας κόμβος n του συστήματος που αναζητά όλα τα δεδομένα του συστήματος που ικανοποιούν τις συνθήκες $[(x,-),(y_{lb},y_{ub})]$, δηλαδή επιθυμεί την επίλυση μίας ερώτησης διαστήματος μόνο ως προς τη δεύτερη διάσταση. Όλα τα δεδομένα που θα μπορούσαν να ικανοποιήσουν την ερώτηση έχουν τη μορφή (x,y_1), (x,y_2), ..., κλπ. Παρατηρούμε πως όλα αυτά τα δεδομένα έχουν την ίδια τιμή για την πρώτη διάσταση. Αυτό σημαίνει πως τα δένδρα που υπολογίζονται για την εισαγωγή τους στο σύστημα ταυτίζονται. Ο κόμβος n υπολογίζει το μοναδικό αυτό δένδρο, όπως περιγράφηκε παραπάνω, δηλαδή υπολογίζει τις τιμές που αντιστοιχούν στα φύλλα του. Αρκεί πλέον να βρει σε ποια από τα φύλλα του δένδρου πρέπει να αναζητήσει τα δεδομένα. Για το λόγο αυτό, ο n βρίσκει τα φύλλα στων οποίων το εύρος τιμών ανήκουν οι τιμές y_{lb} και y_{ub}. Έστω l_{lb} και l_{ub} τα φύλλα αυτά. Είναι προφανές πως κάθε δεδομένο με τιμή $y_{lb} \le y \le y_{ub}$ για τη δεύτερη διάσταση ανήκει σε διαστήματα τιμών για τα οποία είναι υπεύθυνα όλα τα φύλλα ξεκινώντας από το l_{lb} και καταλήγοντας στο l_{ub} στη διάταξη που ορίζεται. Επομένως, προκειμένου ο n να επιλύσει την ερώτηση διαστήματος, πρέπει να βρει κάθε έναν από τους διαδόχους κόμβους όλων των φύλλων μεταξύ των l_{lb} και l_{ub}.

Έστω $l_1, l_2, ..., l_q$, κλπ. η ακολουθία των φύλλων που πρέπει να βρεθούν για να ικανοποιηθεί η ερώτηση διαστήματος που πραγματοποιείται στο σύστημα, όπου $l_1 = l_{lb}$ και $l_q = l_{ub}$, και l_i και

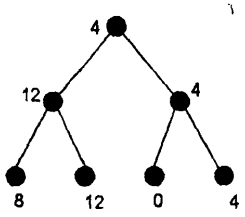
l_{i+1} δύο διαδοχικά φύλλα της ακολουθίας αυτής. Όπως προαναφέρθηκε, η απόσταση μεταξύ δύο διαδοχικών φύλλων του δένδρου είναι μία δύναμη του 2. Υπενθυμίζουμε, όμως, ότι κάθε κόμβος του συστήματος διατηρεί δείκτες προς τους διαδόχους κόμβους κλειδιών που απέχουν από το δικό του κατά διαδοχικές δυνάμεις του 2. Επομένως, ο διάδοχος κόμβος του φύλλου l_i διατηρεί δείκτη προς το διάδοχο κόμβο του φύλλου l_{i+1} και, έτσι, αρκεί ένα μόνο βήμα για τη μετάβαση από έναν τέτοιο κόμβο στον επόμενό του. Συνεπώς, αν χρησιμοποιήσουμε τον αλγόριθμο δρομολόγησης του Chord για να μεταβούμε από τον κόμβο n στον κόμβο $n_1 = succ(l_1)$ σε $O(\log N)$ βήματα, στη συνέχεια, μπορούμε να μεταβούμε από τον κόμβο n_1 στον κόμβο $n_2 = succ(l_2)$ σε ένα μόνο βήμα, χρησιμοποιώντας τον κατάλληλο δείκτη από τον πίνακα δρομολόγησης του n_1. Κατόπιν, μπορούμε να κάνουμε το ίδιο για να μεταβούμε από τον κόμβο n_2 στον κόμβο $n_3 = succ(l_3)$, επίσης, σε ένα μόνο βήμα, χρησιμοποιώντας τον κατάλληλο δείκτη από τον πίνακα δρομολόγησης του n_2, κοκ. Επομένως, η μετάβαση σε κάθε ένα από τα φύλλα του δένδρου, εκτός από το πρώτο, απαιτεί ένα μόνο βήμα.

Η επίλυση, λοιπόν, μίας ερώτησης διαστήματος μόνο ως προς τη δεύτερη διάσταση απαιτεί τελικά $O(2^d + \log N)$ βήματα και $O(2^d + \log N)$ μηνύματα. Επειδή, συνήθως το βάθος d του δένδρου είναι μικρό, η τιμή 2^d θεωρείται σταθερά. Στα πειράματα που πραγματοποιούμε και περιγράφονται στο Κεφάλαιο 10, θεωρούμε δένδρο βάθους $d = 2$, δηλαδή δένδρο με $2^d = 4$ φύλλα. Παρατηρούμε πως το κόστος τέτοιου είδους ερωτήσεων είναι συγκρίσιμο με το αντίστοιχο κόστος στο MAAN και στο Mercury, αφού η επιπλέον επιβάρυνση είναι μία σχετικά μικρή προσθετική σταθερά. Σημειώνεται πως θεωρητική και πειραματική σύγκριση των συστημάτων που περιγράφηκαν γίνεται στο Κεφάλαιο 10.

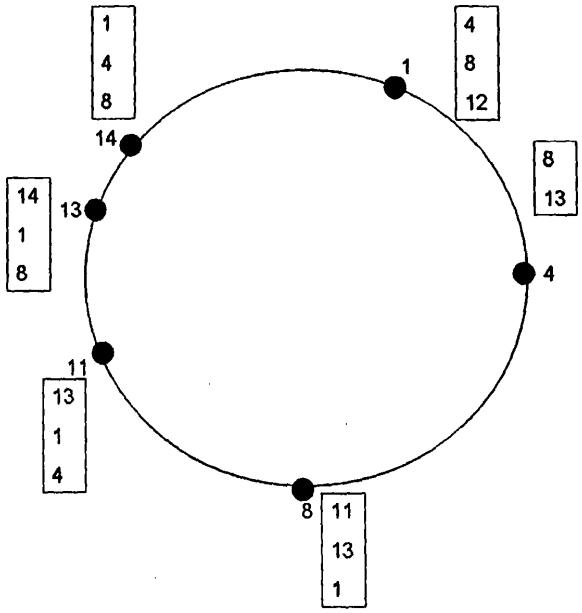
Στο Σχήμα 7.4 και το Σχήμα 7.5 μπορούμε να δούμε πώς επιλύεται μία ερώτηση διαστήματος ως προς την y-διάσταση. Έχουμε υποθέσει ότι το σύστημα έχει $m = 4$, δηλαδή το μέγιστο πλήθος κόμβων που μπορούν να συμμετέχουν σε αυτό είναι $2^4 = 16$ κόμβοι, και συμμετέχουν σε αυτό οι κόμβοι 1, 4, 8, 11, 13 και 14. Δίπλα σε κάθε κόμβο βλέπουμε τους δείκτες που διατηρεί ο κόμβος (fingers). Για παράδειγμα, βλέπουμε ότι ο κόμβος 1 διατηρεί δείκτες προς τους κόμβους 4, 8 και 12, ενώ ο 4 κόμβος διατηρεί δείκτες προς τους κόμβους 8



και 13. Έστω ότι πραγματοποιείται η ερώτηση $q=[(4,-),(4,11)]$ ξεκινώντας από τον κόμβο



Σχήμα 7.4 Το Δένδρο που Ορίζεται για την Ερώτηση Διαστήματος $q=[(4,-),(4,11)]$.



Σχήμα 7.5 Ένα Σύστημα Κόμβων, στο Οποίο Αναπαρίστανται οι Κόμβοι με τα Κλειδιά τους, καθώς, επίσης, και οι Πίνακες Δεικτών που Διατηρούν.

14. Αρχικά, υπολογίζονται οι τιμές που αντιστοιχούν στους κόμβους του δένδρου με ρίζα την τιμή 4 (Σχήμα 7.4). Στη συνέχεια, ο κόμβος βρίσκει ποια φύλλα του δένδρου πρέπει να εξετάσει. Η ακολουθία των φύλλων αυτών περιλαμβάνει τα φύλλα 4, 8 και 12. Αναζητείται, λοιπόν, ο κόμβος $succ(4)=4$. Η δρομολόγηση σε αυτόν γίνεται μέσω του αλγορίθμου δρομολόγηση που εφαρμόζεται στο Chord. Ο κόμβος 14 προωθεί την ερώτηση απευθείας στον 4 μέσω των δεικτών που διατηρεί. Ο κόμβος αυτό ελέγχει τα δεδομένα που διατηρεί τοπικά. Όμως, η ερώτηση πρέπει να προωθηθεί και στον κόμβο $succ(8)=8$. Η προώθηση

γίνεται σε ένα μόνο βήμα από τον κόμβο 4, που διατηρεί δείκτη στον κόμβο 8. Και ο κόμβος 8 ελέγχει με τη σειρά του τα δεδομένα που διατηρεί τοπικά. Τέλος, η ερώτηση πρέπει να προωθηθεί στον κόμβο $succ(12)=13$. Η προώθηση γίνεται και πάλι σε ένα μόνο βήμα από τον κόμβο 8, που διατηρεί δείκτη στον κόμβο 13. Ο κόμβος 12 ελέγχει με τη σειρά του τα δεδομένα που διατηρεί τοπικά και η επίλυση της ερώτησης ολοκληρώνεται. Σημειώνεται ότι η εξέταση των μονοπατιών γίνεται παράλληλα.

7.5.2. Επίλυση Ερωτήσεων Διαστήματος ως προς και τις Δύο Διαστάσεις

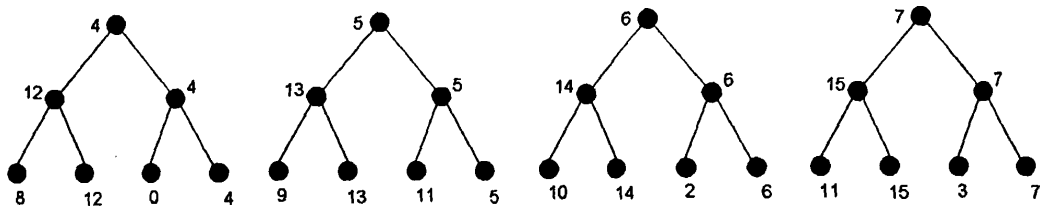
Έστω ένας κόμβος του συστήματος που πραγματοποιεί μία ερώτηση διαστήματος ως προς τις δύο διαστάσεις. Όλα τα δεδομένα που θα μπορούσαν να ικανοποιήσουν την ερώτηση σχηματίζουν περισσότερα του ενός δένδρα και, μάλιστα, τόσα όσες και οι τιμές που ορίζονται στο ζητούμενο διάστημα για την πρώτη διάσταση. Ο κόμβος που εκκινεί την ερώτηση θα έπρεπε να υπολογίσει κάθε ένα από αυτά τα δένδρα, όπως περιγράφηκε παραπάνω. Ωστόσο, αφού οι ρίζες διαδοχικών δένδρων αντιστοιχούν σε διαδοχικές τιμές, τα αντίστοιχα φύλλα των δένδρων αντιστοιχούν σε, επίσης, διαδοχικές τιμές. Έστω, λοιπόν, δύο οποιαδήποτε διαδοχικά δένδρα. Σύμφωνα με τα παραπάνω, ο διάδοχος κόμβος του $i-$οστού φύλλου του δεύτερου δένδρου είναι είτε ο διάδοχος κόμβος του $i-$οστού φύλλου του πρώτου δένδρου, είτε ο επόμενός του. Για το λόγο αυτό, ο κόμβος που πραγματοποιεί μία ερώτηση διαστήματος ως προς τις δύο διαστάσεις υπολογίζει μόνο τα δένδρα που αντιστοιχούν στο κάτω και το άνω άκρο του διαστήματος που ορίζεται για την πρώτη διάσταση στην ερώτηση. Κατόπιν, βρίσκει από ποια από τα φύλλα του πρώτου δένδρου ξεκινούν μονοπάτια με κόμβους που πιθανόν διαθέτουν δεδομένα που ικανοποιούν την ερώτηση διαστήματος και αρχίζει την εξερεύνηση των μονοπατιών αυτών. Η μετάβαση στην αρχή του πρώτου μονοπατιού απαιτεί την εύρεση του διαδόχου κόμβου του αντίστοιχου φύλλου μέσω του αλγορίθμου δρομολόγησης που εφαρμόζεται και στο Chord. Η μετάβαση σε κάθε επόμενο μονοπάτι (με τη σειρά που ορίζει η διάταξη των αντίστοιχων φύλλων που αποτελούν την αρχή τους) γίνεται μέσω των πληροφοριών δρομολόγησης που διατηρούνται στο διάδοχο κόμβο του φύλλου του προηγούμενου μονοπατιού, σε ένα μόνο βήμα. Το τέλος κάθε μονοπατιού που ξεκινά από ένα φύλλο του πρώτου δένδρου σηματοδοτείται από το αντίστοιχο φύλλο του τελευταίου δένδρου.

Πιο συγκεκριμένα, έστω ένας κόμβος n του συστήματος που πραγματοποιεί μία ερώτηση διαστήματος ως προς τις δύο διαστάσεις, δηλαδή μία ερώτηση της μορφής $\left[(x_{lb}, x_{ub}),(y_{lb}, y_{ub})\right]$. Όλα τα δεδομένα που θα μπορούσαν να ικανοποιήσουν την ερώτηση έχουν τη μορφή (x_i, y_i), όπου $x_{lb} \le x_i \le x_{ub}$ και $y_{lb} \le y_{ij} \le y_{ub}$. Με άλλα λόγια, όλα αυτά τα δεδομένα έχουν αποθηκευτεί στο σύστημα βάσει περισσοτέρων του ενός δένδρων και, μάλιστα, τόσων όσες και οι τιμές που ορίζονται στο ζητούμενο διάστημα (x_{lb}, x_{ub}). Ο κόμβος n υπολογίζει το δένδρο με ρίζα την τιμή x_{lb} (πρώτο δένδρο), καθώς, επίσης, και το δένδρο με ρίζα την τιμή x_{ub} (δεύτερο δένδρο). Στη συνέχεια, βρίσκει το σύνολο PL όλων των φύλλων των δένδρων αυτών που επικαλύπτουν το ζητούμενο διάστημα στη δεύτερη διάσταση, δηλαδή το διάστημα (y_{lb}, y_{ub}). Κατόπιν, βρίσκει το φύλλο $v \in PL$ που καλύπτει το χαμηλότερο δυνατό μέρος του ζητούμενου εύρους και το φύλλο $u \in PL$ που καλύπτει το υψηλότερο δυνατό μέρος του ζητούμενου εύρους. Προφανώς, τα φύλλα αυτά μπορούν να ανήκουν στο ίδιο ή σε διαφορετικό δένδρο. Έστω i και j η θέση των φύλλων v και u, αντίστοιχα, στη διάταξη που ορίζουν τα δένδρα για τα φύλλα τους. Το σύνολο $L = \{l_i, l_{i+1}, ..., l_j\}$ των φύλλων του πρώτου δένδρου, από το $i-$οστό έως και το $j-$οστό, κατά σειρά, ορίζουν τα μονοπάτια που πρέπει να εξερευνηθούν για να βρεθούν τα δεδομένα που ικανοποιούν την ερώτηση διαστήματος. Πιο συγκεκριμένα, αν $S = \{s_i, s_{i+1}, ..., s_j\}$ το αντίστοιχο σύνολο των διαδόχων κόμβων των μελών του συνόλου L, όπου $s_k = succ(l_k)$, $i \le k \le j$, τότε οι κόμβοι του συνόλου S αποτελούν την αρχή των μονοπατιών που πρέπει να εξερευνηθούν προκειμένου να επιλυθεί η ερώτηση διαστήματος που πραγματοποιείται στο σύστημα. Η προώθηση της ερώτησης σε κάθε επόμενο κόμβο του μονοπατιού γίνεται μέσω των δεικτών σε διαδόχους κόμβους. Τελικός αποδέκτης της ερώτησης σε ένα μονοπάτι που ξεκινά από το διάδοχο κόμβο s_k ενός φύλλου l_k του πρώτου δένδρου είναι ο διάδοχος κόμβος του αντίστοιχου φύλλου του τελευταίου δένδρου, δηλαδή του δένδρου με ρίζα την τιμή x_{ub}. Η προώθηση της ερώτησης από τον κόμβο n στην αρχή του πρώτου μονοπατιού γίνεται μέσω του αλγορίθμου δρομολόγησης που εφαρμόζεται και στο Chord και απαιτεί $O(\log N)$ βήματα και $O(\log N)$ μηνύματα. Η προώθησή, όμως, στην αρχή κάθε επόμενου μονοπατιού (με τη σειρά που ορίζει η διάταξη των αντίστοιχων φύλλων που αποτελούν την

αρχή τους) γίνεται μέσω των πληροφοριών δρομολόγησης που διατηρούνται στο διάδοχο κόμβο του φύλλου του προηγούμενου μονοπατιού, σε ένα μόνο βήμα.

Στο Σχήμα 7.5 μπορούμε να δούμε το σύστημα στο οποίο επιλύεται μία ερώτηση διαστήματος ως προς τις δύο διαστάσεις. Έστω, λοιπόν, η ερώτηση $q=\left[(4,7),(6,12)\right]$ που εκκινείται από τον κόμβο 1 του συστήματος. Στο Σχήμα 7.6 βλέπουμε όλα τα δένδρα που προκύπτουν βάσει του ζητούμενου εύρους στην πρώτη διάσταση. Παρατηρούμε πως τα αντίστοιχα φύλλα διαδοχικών δένδρων αντιστοιχίζονται σε διαδοχικές τιμές. Για παράδειγμα, τα πρώτα φύλλα των δένδρων με ρίζα την τιμή 4 και 5 έχουν αντιστοιχιστεί στις τιμές 8 και 9, ενώ τα δεύτερα φύλλα έχουν αντιστοιχιστεί στις τιμές 12 και 13. Με άλλα λόγια, τα δένδρα 'μοιάζουν' μεταξύ τους. Επίσης, εξετάζοντας δύο διαδοχικά δένδρα και παρατηρώντας και το δακτύλιο του συστήματος, βλέπουμε πως ο διάδοχος κόμβος ενός φύλλου του δεύτερου κατά σειρά δένδρου είναι είτε ο διάδοχος κόμβος του αντίστοιχου φύλλου του πρώτου δένδρου, είτε ο επόμενός του. Για παράδειγμα, για τα δύο πρώτα φύλλα των δύο πρώτων δένδρων, παρατηρούμε ότι $succ(8)=8$, $succ(9)=11$, δηλαδή οι διάδοχοι των αντίστοιχων φύλλων δεν ταυτίζονται. Επίσης, για τα δύο επόμενα φύλλα των δύο πρώτων δένδρων, παρατηρούμε ότι $succ(12)=13$, $succ(13)=13$, δηλαδή οι διάδοχοι των αντίστοιχων φύλλων ταυτίζονται.

Ακόμη, παρατηρώντας όλα τα δένδρα που υπολογίζονται στο Σχήμα 7.6, βλέπουμε ότι για τη σωστή εύρεση των φύλλων από τα οποία ξεκινούν μονοπάτια που πρέπει να εξερευνηθούν είμαστε αναγκασμένοι να υπολογίσουμε τις τιμές που αντιστοιχούν στα φύλλα όχι μόνο του πρώτου, αλλά και του τελευταίου δένδρου. Για παράδειγμα, αν κατά την επίλυση της ερώτησης $q=\left[(4,7),(6,12)\right]$ βασιστούμε μόνο στις τιμές των κόμβων του πρώτου δένδρου, πρέπει να εξερευνήσουμε μόνο τα μονοπάτια που ξεκινούν από τους διαδόχους κόμβους των φύλλων 8 και 12. Με αυτόν τον τρόπο, όμως, δεν εξετάζονται οι κόμβοι 6 και 7 του τρίτου και του τέταρτου δένδρου, αντίστοιχα, οι οποίοι μπορεί να διαθέτουν κατάλληλα δεδομένα, καθώς δεν ανήκουν στα μονοπάτια που εξερευνούμε. Αν, όμως, υπολογίσουμε τις τιμές που αντιστοιχούν στους κόμβους και του πρώτου, αλλά και του τελευταίου δένδρου και, στη συνέχεια, τα φύλλα από τα οποία ξεκινούν τα προς εξερεύνηση μονοπάτια με τον τρόπο που περιγράφηκε παραπάνω, τότε βρίσκουμε ότι μονοπάτια ξεκινούν από τους κόμβους 4, 8 και 12 του πρώτου δένδρου. Το φύλλο 4 είναι το αντίστοιχο φύλλο του φύλλου 7 του τελευταίου δένδρου με του οποίου το εύρος επικαλύπτεται το ζητούμενο διάστημα τιμών.



Σχήμα 7.6 Τα Δένδρα που Ορίζονται για την Ερώτηση Διαστήματος $q=\left[(4,7),(6,12)\right]$.

Επομένως, κατά την επίλυση της ερώτησης $q=\left[(4,7),(6,12)\right]$, ο κόμβος 1 του συστήματος που την εκκινεί υπολογίζει τις τιμές των φύλλων του πρώτου και του τελευταίου δένδρου και βρίσκει ότι μονοπάτια προς εξερεύνηση ξεκινούν από τους διαδόχους κόμβους των φύλλων 4, 8 και 12 του πρώτου δένδρου. Τα υπόλοιπα δένδρα δεν υπολογίζονται. Έτσι, ο κόμβος 1 προωθεί την ερώτηση στον κόμβο 4 μέσω του αλγορίθμου δρομολόγησης που εφαρμόζεται στο Chord. Στην προκειμένη περίπτωση, αυτό απαιτεί ένα βήμα, αφού ο κόμβος 1 διατηρεί δείκτη στον κόμβο 4. Το πρώτο μονοπάτι προς εξερεύνηση, λοιπόν, ξεκινά από τον $succ(4)=4$ και προωθείται στον επόμενό του κόμβο, που είναι ο $succ(7)=8$. Στον κόμβο αυτό τερματίζεται το μονοπάτι, αφού η ερώτηση έχει καταλήξει στο διάδοχο κόμβο του αντίστοιχου φύλλου του τελευταίου δένδρου. Για την εξερεύνηση του δεύτερου μονοπατιού, η ερώτηση προωθείται από τον κόμβο $succ(4)=4$ (τον πρώτο κόμβο του πρώτου μονοπατιού) στον κόμβο $succ(8)=8$ (τον πρώτο κόμβο του δεύτερου μονοπατιού) μέσω του κατάλληλου δείκτη, δηλαδή σε ένα μόνο βήμα, και, έπειτα, στον κόμβο 11, που τον ακολουθεί. Στον κόμβο αυτό τερματίζεται το δεύτερο μονοπάτι, αφού $succ(11)=11$, δηλαδή η ερώτηση έχει καταλήξει στο διάδοχο κόμβο του αντίστοιχου φύλλου του τελευταίου μονοπατιού. Τέλος, για την εξερεύνηση του τρίτου και τελευταίου μονοπατιού, η ερώτηση προωθείται από τον κόμβο $succ(8)=8$ (τον πρώτο κόμβο του δεύτερου μονοπατιού) στον κόμβο $succ(12)=13$ (τον πρώτο κόμβο του τρίτου μονοπατιού) μέσω του κατάλληλου δείκτη, δηλαδή σε ένα μόνο βήμα. Κατόπιν, προωθείται στον επόμενο κόμβο του 13, που είναι ο κόμβος 14. Από τον κόμβο αυτό, προωθείται και πάλι στον επόμενό του, που είναι ο κόμβος 1. Στο σημείο αυτό, τερματίζεται και το τρίτο μονοπάτι, αφού $succ(15)=1$, δηλαδή η

ερώτηση έχει καταλήξει στο διάδοχο κόμβο του αντίστοιχου φύλλου του τελευταίου μονοπατιών. Σημειώνεται και πάλι ότι η εξερεύνηση των μονοπατιών γίνεται παράλληλα.

Αντίστοιχα επιλύονται και ερωτήσεις διαστήματος που αναφέρονται μόνο στη x διάσταση. Στην περίπτωση αυτή, όμως, ξεκινούν μονοπάτια από όλα τα φύλλα του πρώτου δένδρου.

Καθώς οι κόμβοι κατανέμονται ομοιόμορφα στο δακτύλιο του συστήματος, όπως ακριβώς και στο Chord, κάθε ένα από τα μονοπάτια που πρέπει να εξερευνηθούν αναμένεται πως αποτελείται από $N \times s_y$ κόμβους, όπου υπενθυμίζουμε πως s_y είναι η επιλεκτικότητα στην y διάσταση. Το συνολικό πλήθος των φύλλων από όπου ξεκινά η επεξεργασία μίας ερώτησης διαστήματος είναι μία σταθερά $\sigma < 2^d$, όπου d το βάθος του δένδρου, η οποία εξαρτάται από την επιλεκτικότητα της ερώτησης στη δεύτερη διάσταση και προσδιορίζεται πειραματικά. Οι πειραματικές μετρήσεις μας επιτρέπουν να υπολογίσουμε προσεγγιστικά το ποσοστό των ερωτήσεων διαστήματος που οδηγούν στην εξερεύνηση ενός μόνο μονοπατιού, δύο μονοπατιών, ..., 2^d μονοπατιών. Τα ποσοστά αυτά εξαρτώνται από την επιλεκτικότητα της δεύτερης διάστασης των ερωτήσεων διαστήματος που πραγματοποιούνται στο σύστημα. Ο υπολογισμός, λοιπόν, της σταθεράς γίνεται ως εξής:

$$\sigma = p_1 \times 1 + p_2 \times 2 + p_3 \times 3 + ... + p_{2^d} \times 2^d,$$

όπου p_i το ποσοστό των ερωτήσεων διαστήματος που οδηγούν στην εξερεύνηση i μονοπατιών, $i = 1, 2, ..., 2^d$.

7.6. Γενίκευση του Συστήματος για Περισσότερες Διαστάσεις

Το σύστημα μπορεί να γενικευτεί για περισσότερες από δύο διαστάσεις. Και στην περίπτωση αυτή, το TreeChord χρησιμοποιεί μία δενδρική δομή για να κατανείμει τα πολυδιάστατα δεδομένα ανάμεσα στους κόμβους που συμμετέχουν στο σύστημα. Η δομή αυτή αποτελείται από πολλά επίπεδα δένδρων, καθένα από τα οποία σχετίζεται με κάποια από τις διαστάσεις. Έστω, λοιπόν, $v = (v_1, v_2, ..., v_M)$ ένα δεδομένο M διαστάσεων. Η ρίζα της δενδρικής δομής έχει την τιμή v_1, ενώ κάθε ένα από τα 2^d φύλλα της (από αριστερά προς τα δεξιά) αντιστοιχεί στην τιμή $\left(v_1 + \frac{i}{2^d} \times 2^m\right) \% 2^m$, $i = 1, ..., 2^d$. Θεωρούμε ότι κάθε ένα από τα φύλλα αυτά

αντιστοιχεί σε ένα διάστημα τιμών $(lb, ub]$, όπου ub η τιμή που αντιστοιχεί στο συγκεκριμένο φύλλο και lb η τιμή που αντιστοιχεί στο προηγούμενο κατά σειρά φύλλο. Και πάλι, σειρά των φύλλων θεωρείται η ακολουθία που δημιουργείται όταν τα εξετάζουμε κυκλικά από τα αριστερά προς τα δεξιά. Αυτό είναι το δένδρο που αντιστοιχεί στην πρώτη διάσταση, καθώς ορίζεται από την τιμή του δεδομένου στην πρώτη διάσταση. Στη συνέχεια, κάθε ένα από τα φύλλα αυτού του δένδρου σχηματίζει ένα νέο τέλειο δένδρο βάθους d, με τον ίδιο τρόπο που περιγράφηκε παραπάνω. Οι τιμές των φύλλων του εκάστοτε δένδρου καθορίζονται από την τιμή που έχει αντιστοιχιστεί στη ρίζα του, δηλαδή, ουσιαστικά, στα φύλλα του δένδρου του προηγούμενου επιπέδου. Αυτά είναι τα δένδρα που αντιστοιχούν στη δεύτερη διάσταση, καθώς ορίζονται από την τιμή του δεδομένου στη δεύτερη διάσταση. Με τον ίδιο τρόπο, σχηματίζονται και τα επόμενα επίπεδα. Το τελευταίο επίπεδο αποτελείται από τα δένδρα που αντιστοιχούν στη $M-1$ διάσταση. Όπως παρατηρούμε, τα φύλλα όλων των δένδρων που σχηματίζονται στα διάφορα επίπεδα, ανεξαρτήτως διάταξης, αντιστοιχίζονται στις ίδιες τιμές, αφού οι διαφορετικές τους ρίζες περιστρέφουν κατά κάποιο τρόπο ένα σύνολο τιμών.

Ένα από τα φύλλα του δένδρου του τελευταίου επιπέδου, όπως έχει σχηματιστεί, θα μας οδηγήσει στον κόμβο που θα κληθεί να αποθηκεύσει το πολυδιάστατο δεδομένο. Το φύλλο αυτό είναι εκείνο στου οποίου το εύρος τιμών ανήκει η τιμή v_M, δηλαδή η τιμή του δεδομένου στην τελευταία διάσταση. Για τον υπολογισμό της τιμής που αντιστοιχεί στο φύλλο αυτό, αρκεί απλώς ο υπολογισμός των τιμών που αντιστοιχούν στα 2^d φύλλα, που είναι ίδιες για όλα τα δέντρα των διαφόρων επιπέδων, αφού δεν έχει σημασία η διάταξή τους. Αν lv η τιμή του φύλλου αυτού, το πολυδιάστατο δεδομένο καλείται να αποθηκεύσει ο διάδοχος κόμβος του lv στο δακτύλιο, $l = succ(lv)$. Ο κόμβος που εκκινεί τη διαδικασία εισαγωγής του νέου δεδομένου στο σύστημα αναζητά τον κόμβο l ακολουθώντας τον ίδιο αλγόριθμο δρομολόγησης που εφαρμόζεται και στο σύστημα του Chord. Και στην περίπτωση αυτή, η εύρεση του κόμβου l στον οποίο θα αποθηκευτεί το νέο δεδομένο απαιτεί $O(\log N)$ βήματα.

7.6.1. Επίλυση Απλών Ερωτήσεων

Η επίλυση απλών ερωτήσεων πραγματοποιείται με τρόπο ανάλογο με αυτόν που ακολουθείται στην περίπτωση των δύο διαστάσεων. Η διαδικασία βασίζεται και πάλι στη δενδρική δομή που περιγράφηκε παραπάνω. Αρχικά, υπολογίζεται η τιμή του φύλλου στο οποίο αποθηκεύεται το δεδομένο, με τον τρόπο που μόλις περιγράφηκε. Έστω l η τιμή του φύλλου αυτού. Ο κόμβος n που πραγματοποιεί την ερώτηση αναζητά το διάδοχο κόμβο του l $(succ(l))$ χρησιμοποιώντας τον ίδιο αλγόριθμο δρομολόγησης που εφαρμόζεται στο Chord. Η εύρεση του $succ(l)$ απαιτεί $O(\log N)$ βήματα και $O(\log N)$ μηνύματα. Όταν, τελικά, ο $succ(l)$ λάβει την ερώτηση, ελέγχει τα δεδομένα που διαθέτει τοπικά και απαντά κατάλληλα στην ερώτηση παρέχοντας το δεδομένο, αν το διαθέτει.

7.6.2. Επίλυση Ερωτήσεων Διαστήματος

Η επίλυση ερωτήσεων διαστήματος γίνεται, επίσης, με τρόπο αντίστοιχο με αυτόν που ακολουθείται και στην περίπτωση των δύο διαστάσεων. Όλα τα δεδομένα που θα μπορούσαν να ικανοποιήσουν την ερώτηση σχηματίζουν περισσότερες της μιας δενδρικές δομές και, μάλιστα, τόσες όσες και οι τιμές που ορίζονται στο ζητούμενο διάστημα για την πρώτη διάσταση. Ωστόσο, αφού οι ρίζες διαδοχικών δενδρικών δομών αντιστοιχούν σε διαδοχικές τιμές, τα αντίστοιχα φύλλα των δένδρων σε αντίστοιχα επίπεδα αντιστοιχούν σε, επίσης, διαδοχικές τιμές. Έστω, λοιπόν, δύο οποιεσδήποτε διαδοχικές δενδρικές δομές. Σύμφωνα με τα παραπάνω, ο διάδοχος κόμβος του $i-$οστού φύλλου του δεύτερου δένδρου κάποιου επιπέδου είναι είτε ο διάδοχος κόμβος του $i-$οστού φύλλου του πρώτου δένδρου στο ίδιο επίπεδο, είτε ο επόμενός του. Για το λόγο αυτό, ο κόμβος που πραγματοποιεί μία ερώτηση διαστήματος προς όλες τις διαστάσεις υπολογίζει μόνο τις δενδρικές δομές που αντιστοιχούν στο κάτω και το άνω άκρο του διαστήματος που ορίζεται για την πρώτη διάσταση στην ερώτηση. Κατόπιν, βρίσκει από ποια από τα φύλλα του πρώτης δενδρικής δομής ξεκινούν μονοπάτια με κόμβους που πιθανόν διαθέτουν δεδομένα που ικανοποιούν την ερώτηση διαστήματος και αρχίζει την εξερεύνηση των μονοπατιών αυτών. Η μετάβαση στην αρχή του πρώτου μονοπατιού απαιτεί την εύρεση του διαδόχου κόμβου του αντίστοιχου φύλλου μέσω του αλγορίθμου δρομολόγησης που εφαρμόζεται και στο Chord. Η μετάβαση σε κάθε επόμενο μονοπάτι (με τη σειρά που ορίζει η διάταξη των αντίστοιχων φύλλων που αποτελούν την αρχή τους) γίνεται μέσω των πληροφοριών δρομολόγησης που διατηρούνται στο διάδοχο

κόμβο του φύλλου του προηγούμενου μονοπατιού, σε ένα μόνο βήμα. Το τέλος κάθε μονοπατιού που ξεκινά από ένα φύλλο του πρώτου δένδρου σηματοδοτείται από το αντίστοιχο φύλλο του τελευταίου δένδρου.

Πιο συγκεκριμένα, έστω ένας κόμβος n του συστήματος που πραγματοποιεί μία ερώτηση διαστήματος προς όλες τις διαστάσεις, δηλαδή μία ερώτηση της μορφής $\left[(d_{1l},d_{1u}),...,(d_{Ml},d_{Mu})\right]$. Όλα τα δεδομένα που θα μπορούσαν να ικανοποιήσουν την ερώτηση έχουν τη μορφή $v=(v_1,...,v_M)$, όπου $d_{il}\le v_i\le d_{iu}$. Ο κόμβος n υπολογίζει τη δενδρική δομή με ρίζα την τιμή v_1 (πρώτο δένδρο), καθώς, επίσης, και το δένδρο με ρίζα την τιμή v_M (δεύτερο δένδρο). Στη συνέχεια, βρίσκει το σύνολο PL όλων των φύλλων των δένδρων των τελικών επιπέδων που επικαλύπτουν το ζητούμενο διάστημα στη M διάσταση, δηλαδή το διάστημα (d_{Ml},d_{Mu}). Κατόπιν, βρίσκει το φύλλο $vt\in PL$ που καλύπτει το χαμηλότερο δυνατό μέρος του ζητούμενου εύρους και το φύλλο $u\in PL$ που καλύπτει το υψηλότερο δυνατό μέρος του ζητούμενου εύρους. Προφανώς, τα φύλλα αυτά, που ανήκουν σε δένδρα του τελικού επιπέδου, μπορούν να ανήκουν στην ίδια ή σε διαφορετική δενδρική δομή. Έστω i και j η θέση των φύλλων vt και u, αντίστοιχα, στη διάταξη που ορίζεται για αυτά. Το σύνολο $L=\{l_i,l_{i+1},...,l_j\}$ των φύλλων της πρώτης δενδρικής δομής, από το $i-$οστό έως και το $j-$οστό, κατά σειρά, ορίζουν τα μονοπάτια που πρέπει να εξερευνηθούν για να βρεθούν τα δεδομένα που ικανοποιούν την ερώτηση διαστήματος. Πιο συγκεκριμένα, αν $S=\{s_i,s_{i+1},...,s_j\}$ το αντίστοιχο σύνολο των διαδόχων κόμβων των μελών του συνόλου L, όπου $s_k=succ(l_k)$, $i\le k\le j$, τότε οι κόμβοι του συνόλου S αποτελούν την αρχή των μονοπατιών που πρέπει να εξερευνηθούν προκειμένου να επιλυθεί η ερώτηση διαστήματος που πραγματοποιείται στο σύστημα. Η προώθηση της ερώτησης σε κάθε επόμενο κόμβο του μονοπατιού γίνεται μέσω των δεικτών σε διαδόχους κόμβους. Τελικός αποδέκτης της ερώτησης σε ένα μονοπάτι που ξεκινά από το διάδοχο κόμβο s_k ενός φύλλου l_k του πρώτου δένδρου είναι ο διάδοχος κόμβος του αντίστοιχου φύλλου της τελευταίας δενδρικής δομής, δηλαδή της δενδρικής δομής με ρίζα την τιμή v_{Mu}.

Η προώθηση της ερώτησης από τον κόμβο n στην αρχή του πρώτου μονοπατιού γίνεται μέσω του αλγορίθμου δρομολόγησης που εφαρμόζεται και στο Chord και απαιτεί $O(\log N)$

βήματα και $O(\log N)$ μηνύματα. Η προώθησή, όμως, στην αρχή κάθε επόμενου μονοπατιού (με τη σειρά που ορίζει η διάταξη των αντίστοιχων φύλλων που αποτελούν την αρχή τους) γίνεται μέσω των πληροφοριών δρομολόγησης που διατηρούνται στο διάδοχο κόμβο του φύλλου του προηγούμενου μονοπατιού, σε ένα μόνο βήμα. Επίσης, καθώς οι κόμβοι κατανέμονται ομοιόμορφα στο δακτύλιο του συστήματος, όπως ακριβώς και στο Chord, κάθε ένα από τα μονοπάτια που πρέπει να εξερευνηθούν αναμένεται πως αποτελείται από $N \times s_M$ κόμβους, όπου υπενθυμίζουμε πως s_M είναι η επιλεκτικότητα στη M διάσταση. Το συνολικό πλήθος των μονοπατιών καθορίζει και πάλι η σταθερά $\sigma < 2^d$, η οποία εξαρτάται από την επιλεκτικότητα της ερώτησης στη M διάσταση και προσδιορίζεται, όπως είπαμε, πειραματικά.

Συμπερασματικά, μπορούμε να πούμε πως η δενδρική δομή που χρησιμοποιείται στο TreeChord για την κατανομή των δεδομένων στους κόμβους του συστήματος δεν επηρεάζει αρνητικά μία ομοιόμορφη κατανομή των αναγνωριστικών των δεδομένων εισόδου που πιθανόν ακολουθείται, επιτρέποντας στα δεδομένα να εξακολουθούν να κατανέμονται ομοιόμορφα στο σύστημα. Από την άλλη, το TreeChord επιτυγχάνει να κατανείμει καλά και δεδομένα εισόδου των οποίων τα αναγνωριστικά ακολουθούν μη ομοιόμορφες κατανομές, όπως είναι η εκθετική. Επίσης, υπενθυμίζουμε ότι στο TreeChord δεν εφαρμόζεται κατακερματισμός για την κατανομή των δεδομένων. Θα μπορούσαν, όμως, να εφαρμοστούν συναρτήσεις κατακερματισμού που διατηρούν τη διάταξη, όπως αυτές που περιγράφηκαν στο σύστημα του MAAN. Επίσης, αν και το πλήθος των μηνυμάτων που αποστέλλονται κατά την επίλυση μίας ερώτησης διαστήματος είναι μεγαλύτερο από το αντίστοιχο πλήθος στα άλλα συστήματα, η καθυστέρηση που επιφέρεται στο σύστημα δεν είναι μεγάλη, καθώς τα μονοπάτια εξετάζονται ταυτόχρονα. Επιπρόσθετα, αν και οι κόμβοι συμμετέχουν στην επίλυση πολλών ερωτήσεων διαστήματος εξετάζοντας τοπικά τα δεδομένα που διατηρούν, απαντούν σε μία ερώτηση διαστήματος με πολύ λιγότερα δεδομένα κατά μέσο όρο από ότι στα άλλα συστήματα. Επιπλέον, ο φόρτος που αποκτούν οι κόμβοι εξαιτίας των μηνυμάτων που αποστέλλονται για την επίλυση ερωτήσεων διαστήματος δεν δημιουργεί συμφόρηση σε κάποιο τμήμα του δακτυλίου, αφού τα μηνύματα διαφορετικών ερωτήσεων διαστήματος απασχολούν διαφορετικά τμήματα του δακτυλίου. Στο Κεφάλαιο 10, συζητούνται πιο αναλυτικά και επαληθεύονται όλες οι παραπάνω παρατηρήσεις.

ΚΕΦΑΛΑΙΟ 8. Ο ΠΡΟΣΟΜΟΙΩΤΗΣ ΤΩΝ ΣΥΣΤΗΜΑΤΩΝ

8.1. Εισαγωγή

Στο κεφάλαιο αυτό περιγράφεται ο προσομοιωτής που χρησιμοποιήθηκε για τις υλοποιήσεις των συστημάτων που μελετώνται πειραματικά. Ο προσομοιωτής αποτελείται από δύο τμήματα: α) το πρόγραμμα *traffic_gen*, το οποίο παράγει τα στοιχεία που καθορίζουν την πορεία της προσομοίωσης και β) το πρόγραμμα *sim*, το οποίο υλοποιεί ουσιαστικά την προσομοίωση. Ο προσομοιωτής είναι γραμμένος στη γλώσσα προγραμματισμού C.

8.2. Το Πρόγραμμα traffic_gen

Το πρόγραμμα αυτό χρησιμοποιείται για τη δημιουργία του αρχείου που περιγράφει την πορεία των γεγονότων που θα λάβουν χώρα κατά την προσομοίωση. Δέχεται δύο παραμέτρους.

Η πρώτη παράμετρος αποτελεί το αρχείο που δέχεται το πρόγραμμα ως είσοδο, το οποίο καθορίζει τα είδη των γεγονότων που θα συμβούν κατά τη διάρκεια της προσομοίωσης, καθώς, επίσης, και το πλήθος τους. Πρόκειται για το *σενάριο* της προσομοίωσης. Το αρχείο αυτό περιέχει τριών ειδών εντολές, καθεμία με διαφορετική σημασία για το πρόγραμμα.

Η εντολή 'events num avg wjoin wleave wfail winsert wfind' περιγράφει το είδος των γεγονότων που θα λάβουν χώρα κατά τη διάρκεια της προσομοίωσης, καθώς, επίσης, και το πλήθος τους. Πιο συγκεκριμένα, υποδεικνύει στο πρόγραμμα να δημιουργήσει πλήθος *num* γεγονότων σύμφωνα με μια κατανομή Poisson, με ρυθμό άφιξης $1/avg$. Οι παράμετροι *wjoin*, *wleave*, *wfail*, *winsert* και *wfind* αναπαριστούν το ποσοστό σύμφωνα με το οποίο α) ένας κόμβος εισέρχεται στο σύστημα (*wjoin*), β) ένας κόμβος εγκαταλείπει το σύστημα (*wleave*), γ) ένας κόμβος αποτυγχάνει (*wfail*), δ) ένα δεδομένο εισάγεται στο σύστημα (*winsert*), ε) ένα δεδομένο αναζητείται στο σύστημα (*wfind*) (απλή ερώτηση).

Η εντολή 'wait time' εισάγει μία χρονική παύση μεταξύ του συνόλου των γεγονότων που προηγούνται αυτής και του συνόλου των γεγονότων που έπονται.

Τέλος, η εντολή 'exit' σηματοδοτεί τον τερματισμό της προσομοίωσης.

Η δεύτερη παράμετρος αποτελεί το seed που θα χρησιμοποιήσει η γεννήτρια τυχαίων αριθμών που θα χρησιμοποιηθεί από τη διαδικασία της προσομοίωσης.

Το πρόγραμμα traffic_gen παράγει ως έξοδο ένα αρχείο που περιγράφει ποια γεγονότα θα συμβούν στο σύστημα κατά τη διάρκεια της προσομοίωσης. Το αρχείο αυτό περιγράφεται αναλυτικότερα στη συνέχεια.

8.3. Το πρόγραμμα sim

Το πρόγραμμα αυτό υλοποιεί ουσιαστικά την προσομοίωση, δηλαδή εκτελεί τα γεγονότα που παρήγαγε το πρόγραμμα traffic_gen υλοποιώντας το πρωτόκολλο του συστήματος. Δέχεται δύο παραμέτρους.

Η πρώτη παράμετρος είναι το αρχείο εισόδου του προγράμματος. Το αρχείο αυτό, το οποίο αποτελεί την έξοδο του προγράμματος traffic_gen, περιέχει αναλυτική περιγραφή των γεγονότων που θα συμβούν κατά τη διάρκεια της προσομοίωσης.

Τα γεγονότα μπορεί να έχουν τη μορφή:

'join n t': ο κόμβος n εισέρχεται στο σύστημα τη χρονική στιγμή t.

'leave n t': ο κόμβος n αποχωρεί από το σύστημα τη χρονική στιγμή t.

'fail n t': ο κόμβος n αποτυγχάνει τη χρονική στιγμή t.

'insert n id t': το δεδομένο id εισάγεται στο σύστημα τη χρονική στιγμή t και η διαδικασία εκκινείται από τον κόμβο n.

'find n id t': το δεδομένο id αναζητείται τη χρονική στιγμή t και η διαδικασία εκκινείται από τον κόμβο n.

'exit t': η προσομοίωση τερματίζεται τη χρονική στιγμή t.

Η δεύτερη παράμετρος είναι ο seed που θα χρησιμοποιηθεί από τη γεννήτρια τυχαίων αριθμών του προγράμματος.

Η έξοδος που παράγεται από το πρόγραμμα sim είναι μια καταγραφή των γεγονότων που εκτελέστηκαν κατά την προσομοίωση. Για τις εισαγωγές, τις αποχωρήσεις και τις αποτυχίες κόμβων τυπώνεται ένα μήνυμα που δείχνει αν πραγματοποιήθηκαν κατά τη διάρκεια της προσομοίωσης και αν όχι, για ποιο λόγο απέτυχαν. Για τις εισαγωγές και τις αναζητήσεις δεδομένων τυπώνεται ένα μήνυμα που δείχνει αν πραγματοποιήθηκαν ή όχι, καθώς, επίσης, και το πλήθος των βημάτων που απαιτήθηκαν για να εκτελεστούν οι λειτουργίες.

Ακόμη, τυπώνονται και μηνύματα που δείχνουν ποια ήταν η κατάσταση των κόμβων μετά το τέλος της προσομοίωσης. Για κάθε κόμβο τυπώνεται η λίστα με τους δείκτες δρομολόγησης, η λίστα με τα δεδομένα που διατηρούνται, η λίστα των αιτήσεων αναζήτησης που δεν ικανοποιήθηκαν κατά την προσομοίωση, το πλήθος των δεδομένων μέσω των οποίων ο κόμβος ικανοποίησε απλές ερωτήσεις και το πλήθος των απλών ερωτήσεων στων οποίων την επίλυση συμμετείχε.

8.3.1. Βασικές Μεταβλητές, Δομές Δεδομένων και Συναρτήσεις του Προγράμματος

Στη συνέχεια περιγράφονται συνοπτικά κάποιες από τις βασικές μεταβλητές, δομές δεδομένων και συναρτήσεις που υλοποιούνται στον προσομοιωτή.

8.3.1.1. Η Μεταβλητή NUM_BITS

Η μεταβλητή αυτή αποτελεί το πλήθος των bits που απαιτούνται για την αναπαράσταση των αναγνωριστικών που ανατίθενται σε κόμβους και δεδομένα του συστήματος και καθορίζει τι μέγιστο πλήθος κόμβων που μπορούν να συμμετέχουν στο σύστημα.

8.3.1.2. Η Δομή Node

Η δομή αυτή περιγράφει κάθε κόμβο που συμμετέχει στο σύστημα. Τα πεδία που περιλαμβάνει είναι τα εξής: α) το πεδίο id, που αποτελεί το αναγνωριστικό που ανατίθεται στον κόμβο κατά την είσοδό του στο σύστημα, β) το πεδίο status, που δείχνει αν ο κόμβος συμμετέχει στο σύστημα, αν έχει αποχωρήσει ή αν βρίσκεται στη διαδικασία εισαγωγής του σε αυτό, γ) το πεδίο docList, που αποτελεί τη λίστα των δεδομένων που διατηρεί ο κόμβος, δ) το πεδίο fingerList, που αποτελεί τη λίστα με τους δείκτες δρομολόγησης που διατηρεί ο κόμβος, ε) το πεδίο reqList, που αποτελεί τη λίστα των ερωτημάτων που πρέπει να επιλύσει ο κόμβος.

8.3.1.3. Η Δομή Document

Η δομή αυτή περιγράφει κάθε δεδομένο που εισάγεται στο σύστημα. Το πεδίο που περιλαμβάνει είναι το πεδίο id, που αποτελεί το αναγνωριστικό που ανατίθεται στο δεδομένο.

8.3.1.4. Η Δομή Request

Η δομή αυτή περιγράφει κάθε λειτουργία που πραγματοποιείται στο σύστημα. Τα πεδία που περιλαμβάνει είναι τα ακόλουθα: α) το πεδίο type, που δείχνει αν πρόκειται για λειτουργία εισαγωγής ή αναζήτησης δεδομένου, εισαγωγής κόμβου ή λειτουργίας για τη διατήρηση της συνέπειας του συστήματος, β) το πεδίο initiator, που δείχνει ποιος κόμβος εκκινεί τη λειτουργία, γ) το πεδίο sender, που δείχνει ποιος είναι ο τρέχων αποστολέας του μηνύματος που περιγράφει τη λειτουργία που εκτελείται, δ) το πεδίο pred, που δείχνει ποιος είναι ο μέχρις στιγμής γνωστός προκάτοχος της λειτουργίας, ε) το πεδίο succ, που δείχνει ποιος είναι ο μέχρις στιγμής γνωστός διάδοχος της λειτουργίας, στ) το πεδίο done, που αποκτά τιμή 1 όταν το μήνυμα της λειτουργίας καταλήξει στον τελικό της αποδέκτη, δηλαδή στο διάδοχό της, ζ) το πεδίο hops, που δείχνει το πλήθος βημάτων που απαιτήθηκαν για την προώθηση του μηνύματος της λειτουργίας.

8.3.1.5. Η Δομή Event

Η δομή αυτή περιγράφει κάθε γεγονός που πρόκειται να συμβεί στο σύστημα. Περιλαμβάνει τα πεδία: α) το πεδίο nodeId, που δείχνει ποιον κόμβο αφορά το γεγονός, β) το πεδίο fun(), που δείχνει ποια συνάρτηση θα εκτελεστεί κατά την εκτέλεση του γεγονότος, γ) το πεδίο params, που δείχνει ποιες είναι οι παράμετροι της συνάρτησης που θα κληθεί κατά την εκτέλεση του γεγονότος στο σύστημα.

8.3.1.6. Η Δομή Heap

Η δομή αυτή περιγράφει τη δομή του σωρού στον οποίο αποθηκεύονται όλα τα γεγονότα που πρόκειται να συμβούν στο σύστημα. Η εισαγωγή των γεγονότων στη δομή αυτή γίνεται βάσει μιας χρονικής τιμής που τους αποδίδεται από το πρόγραμμα traffic_gen και καθορίζει τη χρονική στιγμή (ως προς το χρόνο προσομοίωσης) κατά την οποία θα λάβει χώρα το γεγονός.

8.3.1.7. Η Συνάρτηση void genEvent(int nodeId, void (*fun)(), void *params, double time)

Η συνάρτηση αυτή καλείται από τον προσομοιωτή κάθε φορά που διαβάζεται ένα γεγονός από το αρχείο εισόδου, το οποίο πρέπει να λάβει χώρα στο σύστημα. Το γεγονός εισάγεται σε ένα σωρό βάσει της χρονικής στιγμής (ως προς το χρόνο προσομοίωσης) που πρέπει να συμβεί.

8.3.1.8. Η Συνάρτηση Event *getEvent(CalQueue *evCal, double time, double* event_time)

Η συνάρτηση αυτή εξάγει το επόμενο γεγονός που πρέπει να λάβει χώρα στο σύστημα από το σωρό στον οποίο έχει εισαχθεί σύμφωνα με τη χρονική στιγμή που προσδιορίζεται για αυτό.

8.3.1.9. Η Συνάρτηση Request *newRequest(ID x, int type, int style, ID initiator)

Η συνάρτηση αυτή δημιουργεί ένα νέο αίτημα που περιγράφει μία λειτουργία που πρέπει να εκτελεστεί στο σύστημα

8.3.1.10. Η Συνάρτηση void insertRequest(Node *n, Request *r)

Η συνάρτηση αυτή εισάγει ένα αίτημα προς εξυπηρέτηση στον κατάλληλο κόμβο, δηλαδή στον κόμβο που καλείται να επιτελέσει μία λειτουργία στο σύστημα.

8.3.1.11. Η Συνάρτηση Request *getRequest(Node *n)

Η συνάρτηση αυτή εξάγει ένα αίτημα από τη λίστα αιτημάτων ενός κόμβου. Στη συνέχεια, ο κόμβος επεξεργάζεται το αίτημα αυτό.

8.3.1.12. Η Συνάρτηση void processRequest(Node *n)

Η συνάρτηση αυτή υλοποιεί ουσιαστικά τον αλγόριθμο δρομολόγησης του συστήματος, προσομοιώνοντας την προώθηση ενός αιτήματος στον κατάλληλο κόμβο.

8.3.1.13. Η Συνάρτηση int processRequest1(Node *n, Request *r)

Η συνάρτηση αυτή καλείται όταν ένας κόμβος επιθυμεί να ελέγξει αν το αίτημα προς εξυπηρέτηση που επεξεργάζεται πρέπει να σταματήσει να προωθείται. Αυτό ισχύει αν ο κόμβος είναι ο διάδοχος κόμβος του αναγνωριστικού που έχει ανατεθεί στο αίτημα.

8.3.1.14. Η Συνάρτηση void join(Node *n, int *nodeId)

Η συνάρτηση αυτή καλείται όταν εκκινείται ένα γεγονός εισαγωγής κόμβου στο σύστημα. Δημιουργεί τα κατάλληλα αιτήματα προς εξυπηρέτηση, τα εισάγει στους κατάλληλους κόμβους και εκκινεί τη διαδικασία επεξεργασίας τους.

8.3.1.15. Η Συνάρτηση void join1(ID id, ID succ)

Η συνάρτηση αυτή καλείται όταν έχει ολοκληρωθεί η διαδικασία εισαγωγής ενός νέου κόμβου στο σύστημα. Στην πραγματικότητα, εκτελείται όταν βρεθεί ο διάδοχος του νέου κόμβου στο σύστημα. Η συνάρτηση εκκινεί τη διαδικασία σταθεροποίησης του συστήματος.

8.3.1.16. Η Συνάρτηση void leave(Node *n, void *dummy)

Η συνάρτηση αυτή καλείται όταν εκκινείται ένα γεγονός αποχώρησης κόμβου από το σύστημα. Δημιουργεί τα κατάλληλα αιτήματα προς εξυπηρέτηση, τα εισάγει στους κατάλληλους κόμβους και εκκινεί τη διαδικασία επεξεργασίας τους.

8.3.1.17. Η Συνάρτηση void nodeFailure(Node *n, void *dummy)

Η συνάρτηση αυτή καλείται όταν εκκινείται ένα γεγονός αποτυχίας κόμβου στο σύστημα. Δημιουργεί τα κατάλληλα αιτήματα προς εξυπηρέτηση, τα εισάγει στους κατάλληλους κόμβους και εκκινεί τη διαδικασία επεξεργασίας τους.

8.3.1.18. Η Συνάρτηση void insertDocument(Node *n, ID *docId)

Η συνάρτηση αυτή καλείται όταν εκκινείται ένα γεγονός εισαγωγής δεδομένων στο σύστημα. Δημιουργεί τα κατάλληλα αιτήματα προς εξυπηρέτηση, τα εισάγει στους κατάλληλους κόμβους και εκκινεί τη διαδικασία επεξεργασίας τους.

8.3.1.19. Η Συνάρτηση void insertDocumentLocal(Node *n, ID *docId)

Η συνάρτηση αυτή εκτελείται από τον κόμβο του συστήματος που πρέπει να αποθηκεύσει το νέο δεδομένο που εισάγεται στο σύστημα και επιτελεί τη διαδικασία εισαγωγής του δεδομένου.

8.3.1.20. Η Συνάρτηση void findDocument(Node *n, ID *docId)

Η συνάρτηση αυτή καλείται όταν εκκινείται ένα γεγονός αναζήτησης δεδομένων στο σύστημα. Δημιουργεί τα κατάλληλα αιτήματα προς εξυπηρέτηση, τα εισάγει στους κατάλληλους κόμβους και εκκινεί τη διαδικασία επεξεργασίας τους.

8.3.1.21. Η Συνάρτηση void findDocumentLocal(Node *n, ID *docId)

Η συνάρτηση αυτή εκτελείται από τον κόμβο του συστήματος που πρέπει να ελέγξει αν διαθέτει το δεδομένο που αναζητείται στο σύστημα και επιτελεί τη διαδικασία ελέγχου των δεδομένων που διατηρούνται τοπικά από τον κόμβο.

8.3.1.22. Η Συνάρτηση void updateDocList(Node *n, Node *s)

Η συνάρτηση αυτή ελέγχει αν ο νέος κόμβος που εισάγεται στο σύστημα πρέπει να λάβει κάποιο από τα δεδομένα για τα οποία ήταν υπεύθυνος πριν ο διάδοχός του στο σύστημα.

8.3.1.23. Η Συνάρτηση void stabilize(Node *n)

Η συνάρτηση αυτή καλείται περιοδικά από τους κόμβους του συστήματος. Αποτελεί τη ρουτίνα που ενημερώνει τις δομές δρομολόγησης που διατηρούνται στους κόμβους.

Σημειώνεται ότι κατά τη διάρκεια εκτέλεσης του προγράμματος προσομοίωσης, γεγονότα εισάγονται στη δομή του σωρού και εξάγονται από αυτή κάθε φορά που ένα μήνυμα προωθείται από έναν κόμβο του συστήματος σε κάποιον άλλο, προσομοιώνοντας κατ' αυτόν τον τρόπο τον κατανεμημένο χαρακτήρα των υλοποιούμενων συστημάτων.

ΚΕΦΑΛΑΙΟ 9. ΥΛΟΠΟΙΗΣΕΙΣ

Στο κεφάλαιο αυτό περιγράφονται συνοπτικά οι τροποποιήσεις που πραγματοποιήθηκαν στον κώδικα του προσομοιωτή για την υλοποίηση των συστημάτων που μελετώνται.

9.1. Τροποποιήσεις για Όλες τις Υλοποιήσεις

Αρχικά, τροποποιήθηκε το πρόγραμμα του traffic_gen έτσι ώστε τα δεδομένα που διαχειρίζεται το σύστημα να είναι πολυδιάστατα, αλλά και να μπορούν να ακολουθούν κάποια κατανομή κατά την εισαγωγή τους στο σύστημα (π.χ. την ομοιόμορφη ή την εκθετική). Κατόπιν, τροποποιήθηκε η δομή του συστήματος που περιγράφει τα δεδομένα, έτσι ώστε να περιγράφονται πολυδιάστατα δεδομένα. Φυσικά, τροποποιήθηκαν κατάλληλα και όσες συναρτήσεις του συστήματος προσπελαύνουν τη δομή αυτή, έτσι ώστε να διαχειρίζονται τις τιμές όλων των διαστάσεων των δεδομένων.

Ακόμη, τροποποιήθηκαν οι συναρτήσεις findDocument() και findDocumentLocal(), έτσι ώστε να είναι δυνατή η επίλυση απλών ερωτήσεων, αλλά και ερωτήσεων διαστήματος.

9.2. Επιπλέον Τροποποιήσεις για την Υλοποίηση του Συστήματος του MAAN

Στη συνέχεια περιγράφονται οι τροποποιήσεις που έγιναν στον προσομοιωτή για την υλοποίηση του συστήματος του MAAN.

Εκτός από τις γενικές τροποποιήσεις που πραγματοποιήθηκαν, τροποποιήθηκε, επιπλέον, η συνάρτηση processRequest(), η οποία υλοποιεί τον αλγόριθμο δρομολόγησης που εφαρμόζεται στο σύστημα. Τέλος, τροποποιήθηκε η συνάρτηση findDocumentLocal(), στην οποία γίνεται επεξεργασία ενός αιτήματος απλής ερώτησης ή ερώτησης διαστήματος που πραγματοποιείται στο σύστημα.

9.3. Επιπλέον Τροποποιήσεις για την Υλοποίηση του Συστήματος του Mercury

Στη συνέχεια περιγράφονται οι τροποποιήσεις που έγιναν στον προσομοιωτή για την υλοποίηση του συστήματος του Mercury.

Εκτός από τις γενικές τροποποιήσεις που πραγματοποιήθηκαν, έγιναν πολλές αλλαγές προκειμένου να προσομοιωθούν οι πολλαπλοί δακτύλιοι που υλοποιούνται στο σύστημα. Τροποποιήθηκε, επιπλέον, η δομή που περιγράφει κάθε κόμβο του συστήματος, έτσι ώστε να περιγράφει τις διαφορετικές πληροφορίες δρομολόγησης που διατηρούν οι κόμβοι σε κάθε έναν από τους δακτυλίους που σχηματίζονται στο σύστημα. Υπενθυμίζουμε ότι στο σύστημα του Mercury, κάθε κόμβος μπορεί να συμμετέχει σε οποιοδήποτε δακτύλιο και γνωρίζει όχι μόνο τον προκάτοχο και το διάδοχό του σε κάθε ένα από αυτά, αλλά διατηρεί και *e* μακρινές ακμές προς κόμβους του ιδίου δακτυλίου. Επίσης, κάθε κόμβος γνωρίζει και έναν ακόμη κόμβο για κάθε άλλο δακτύλιο του συστήματος. Για την υλοποίηση, λοιπόν, του Mercury τροποποιήθηκαν οι συναρτήσεις που κατασκευάζουν και ενημερώνουν τις πληροφορίες δρομολόγησης που διατηρούν οι κόμβοι. Τροποποιήθηκε, ακόμη, η συνάρτηση processRequest(), η οποία υλοποιεί τον αλγόριθμο δρομολόγησης που εφαρμόζεται στο Mercury. Τέλος, τροποποιήθηκε η συνάρτηση findDocumentLocal(), στην οποία γίνεται επεξεργασία ενός αιτήματος απλής ερώτησης ή ερώτησης διαστήματος που πραγματοποιείται στο σύστημα.

9.4. Επιπλέον Τροποποιήσεις για την Υλοποίηση του Συστήματος του TreeChord

Στη συνέχεια περιγράφονται οι τροποποιήσεις που έγιναν στον προσομοιωτή για την υλοποίηση του συστήματος του TreeChord.

Εκτός από τις γενικές τροποποιήσεις που πραγματοποιήθηκαν, τροποποιήθηκαν, επιπλέον, οι συναρτήσεις που καλούνται όταν εκκινείται ένα γεγονός εισαγωγής ή αναζήτησης δεδομένου στο σύστημα, δηλαδή οι συναρτήσεις insertDocument() και findDocument(). Στις συναρτήσεις αυτές γίνεται χρήση της δενδρικής δομής βάσει της οποίας γίνεται η διαχείριση των δεδομένων του συστήματος. Επίσης, τροποποιήθηκε η συνάρτηση updateDocList(), η οποία καθορίζει ποια δεδομένα μεταφέρονται σε έναν νέο κόμβο που εισέρχεται στο σύστημα και ποια παραμένουν στο διάδοχό του. Και η συνάρτηση αυτή χρησιμοποιεί τη δενδρική δομή βάσει της οποίας γίνεται διαχείριση των δεδομένων.

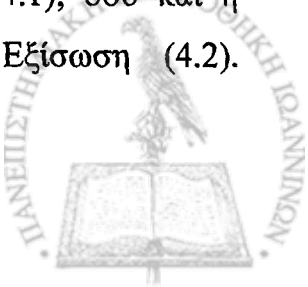
ΚΕΦΑΛΑΙΟ 10. ΠΕΙΡΑΜΑΤΙΚΗ ΜΕΛΕΤΗ

Στις ενότητες που ακολουθούν μελετώνται πειραματικά και συγκρίνονται τα συστήματα του TreeChord, του MAAN και του Mercury. Αρχικά, εξετάζουμε την κατανομή του φορτίου στους κόμβους κάθε συστήματος. Προκειμένου να γίνει αυτό, εξετάζουμε πώς κατανέμονται τα δεδομένα και οι ερωτήσεις στους κόμβους (Ενότητα 10.1 και Ενότητα 10.2, αντίστοιχα). Στη συνέχεια, συγκρίνουμε τους αλγορίθμους επίλυσης ερωτήσεων διαστήματος που εφαρμόζονται σε κάθε σύστημα (Ενότητα 10.3).

10.1. Κατανομή Δεδομένων στους Κόμβους

Προκειμένου να εξετάσουμε την κατανομή των δεδομένων στους κόμβους κάθε συστήματος, μετρούμε το πλήθος των δεδομένων που διατηρεί κάθε κόμβος, καθώς, επίσης, και το ποσοστό των υπερφορτωμένων ή υποφορτωμένων κόμβων κάθε συστήματος (κόμβοι με μη εξισορροπημένο φορτίο).

Σε κάθε ένα από τα συστήματα που εξετάζονται, τα αναγνωριστικά των δεδομένων εισόδου ακολουθούν την ομοιόμορφη ή την εκθετική κατανομή (ως κατανομή που δεν κατανέμει τα κλειδιά ομοιόμορφα). Στο σύστημα του MAAN, και στις δύο περιπτώσεις, εφαρμόζεται τόσο η απλή συνάρτηση κατακερματισμού που περιγράφεται στην Εξίσωση (4.1), όσο και η ομοιόμορφη συνάρτηση κατακερματισμού που περιγράφεται στην Εξίσωση (4.2).



Σημειώνεται ότι οι τιμές που δίνουν και οι δύο συναρτήσεις κατακερματισμού ανήκουν στο διάστημα $\left[0, 2^m - 1\right]$, που είναι το εύρος του δακτυλίου του συστήματος.

10.1.1. Απεικόνιση Πλήθους Κόμβων που Διατηρούν Διαφορετικό Πλήθος Δεδομένων

Στα πειράματα που ακολουθούν εξετάζουμε πώς κατανέμονται τα δεδομένα στους κόμβους που συμμετέχουν σε κάθε σύστημα μετρώντας το πλήθος των δεδομένων που διατηρεί κάθε κόμβος. Όπως θα δούμε και στη συνέχεια, όταν τα αναγνωριστικά των δεδομένων εισόδου ακολουθούν την ομοιόμορφη κατανομή, σε όλα τα συστήματα τα δεδομένα κατανέμονται σχεδόν ομοιόμορφα στους κόμβους. Όταν, όμως, ακολουθείται η εκθετική κατανομή, το σύστημα του TreeChord επιτυγχάνει καλύτερη κατανομή των δεδομένων.

Σε κάθε πείραμα, αρχικά εισέρχονται οι κόμβοι στο σύστημα και, έπειτα, γίνεται εισαγωγή των δεδομένων. Στο πρώτο πείραμα εισάγονται 500 κόμβοι και 5000 δεδομένα, στο δεύτερο πείραμα εισάγονται 1000 κόμβοι και 10000 δεδομένα, στο τρίτο πείραμα εισάγονται 4000 κόμβοι και 40000 δεδομένα, ενώ στο τέταρτο πείραμα εισάγονται 8000 κόμβοι και 80000 δεδομένα.

Στις γραφικές παραστάσεις που ακολουθούν (Σχήμα 10.1 – Σχήμα 10.4) απεικονίζεται ο αριθμός των κόμβων που διατηρούν ακριβώς i δεδομένα, για κάθε $i \geq 0$.

Εξετάζοντας το σύστημα του TreeChord στο οποίο το βάθος του δένδρου είναι 2, δηλαδή το πλήθος των φύλλων του δένδρου είναι $2^2 = 4$, παρατηρούμε πως όταν τα δεδομένα εισόδου ακολουθούν την ομοιόμορφη κατανομή, δηλαδή τα αναγνωριστικά των δεδομένων εισόδου κατανέμονται ομοιόμορφα στο εύρος του δακτυλίου του συστήματος, τότε δεν παρατηρείται σημαντική ανομοιομορφία στο φορτίο που φέρουν οι κόμβοι του συστήματος. Αν όλα τα δεδομένα κατανέμονταν τέλεια ανάμεσα στους κόμβους του συστήματος, κάθε κόμβος θα έπρεπε να διατηρεί τόσα δεδομένα όσα και ο λόγος του πλήθους των εισαγόμενων δεδομένων στο σύστημα προς το πλήθος των συμμετεχόντων κόμβων. Στην προκειμένη περίπτωση, κάθε κόμβος θα έπρεπε να διατηρεί 10 δεδομένα. Παρατηρούμε, λοιπόν, πως οι περισσότεροι κόμβοι διατηρούν 10 ή σχεδόν 10 δεδομένα. Υπάρχουν, βέβαια, και κάποιοι κόμβοι που διατηρούν έως και τριπλάσιο πλήθος δεδομένων (30), ενώ ελάχιστοι είναι οι κόμβοι που διατηρούν ακόμη περισσότερα δεδομένα. Οι περισσότεροι κόμβοι, λοιπόν, διατηρούν τόσα

δεδομένα όσα και η αναμενόμενη τιμή που υπολογίζεται αν τα εισαγόμενα δεδομένα κατανεμηθούν τέλεια ανάμεσα στους κόμβους του συστήματος. Μπορούμε, επομένως, να πούμε ότι ο τρόπος που χρησιμοποιεί το σύστημα του TreeChord για να κατανείμει τα δεδομένα στους κόμβους δεν 'καταστρέφει' τον καλό διαμοιρασμό των εισαγόμενων δεδομένων που δημιουργεί η ομοιόμορφη κατανομή των τιμών των αναγνωριστικών τους.

Εξετάζοντας το σύστημα του MAAN, παρατηρούμε πως όταν χρησιμοποιείται η απλή συνάρτηση κατακερματισμού που περιγράφεται στην Εξίσωση (4.1) και τα αναγνωριστικά των δεδομένων εισόδου κατανέμονται ομοιόμορφα στο εύρος του δακτυλίου του συστήματος ή όταν χρησιμοποιείται η δεύτερη συνάρτηση κατακερματισμού που περιγράφεται στην Εξίσωση (4.2) και τα αναγνωριστικά των δεδομένων εισόδου ακολουθούν μία συνεχή και μονότονα αύξουσα κατανομή, όπως είναι η εκθετική, τότε, επίσης, δεν παρατηρείται σημαντική ανομοιομορφία στο φορτίο που φέρουν οι κόμβοι του συστήματος. Πιο συγκεκριμένα, οι περισσότεροι κόμβοι διατηρούν περίπου 20 δεδομένα, όσα και η αναμενόμενη τιμή, αφού για κάθε δισδιάστατο δεδομένο στο σύστημα του MAAN γίνεται εισαγωγή του ως προς κάθε μία από τις διαστάσεις του. Υπάρχουν, επίσης, λίγοι κόμβοι που διατηρούν έως και τετραπλάσιο πλήθος δεδομένων, δηλαδή 80 δεδομένα, ενώ ελάχιστοι κόμβοι διατηρούν περισσότερα από 80 δεδομένα. Επομένως, μπορούμε να πούμε πως δεν υπάρχουν σημαντικές διακυμάνσεις στο φορτίο που φέρουν οι κόμβοι, ενώ δεν υπάρχουν και πολλοί υπερφορτωμένοι κόμβοι ή πολλοί κόμβοι με πολύ μικρό φορτίο. Στην πρώτη περίπτωση, αυτό συμβαίνει καθώς η συνάρτηση κατακερματισμού που χρησιμοποιείται παράγει ομοιόμορφα κατανεμημένες τιμές στο εύρος τιμών όταν τα αναγνωριστικά των δεδομένων είναι ομοιόμορφα κατανεμημένα στο δακτύλιο. Στη δεύτερη περίπτωση, αυτό οφείλεται στη συνεχή και μονότονα αύξουσα κατανομή, που είναι η εκθετική, η οποία ακολουθείται από τα αναγνωριστικά των δεδομένων εισόδου και επιτρέπει στη συνάρτηση κατακερματισμού που χρησιμοποιείται να παράγει τιμές ομοιόμορφα κατανεμημένες στο εύρος τιμών του συστήματος.

Εξετάζοντας το σύστημα του Mercury, και πάλι παρατηρούμε πως όταν τα αναγνωριστικά των δεδομένων εισόδου ακολουθούν την ομοιόμορφη κατανομή, τότε δεν παρατηρείται σημαντική ανομοιομορφία στο φορτίο που φέρουν οι κόμβοι του συστήματος. Πιο συγκεκριμένα, οι περισσότεροι κόμβοι διατηρούν περίπου 20 δεδομένα, όσα και η αναμενόμενη τιμή, αφού για κάθε δισδιάστατο δεδομένο στο σύστημα του Mercury γίνεται

εισαγωγή του σε κάθε ένα από τους δύο δακτυλίους που αντιστοιχούν σε κάθε μία από τις διαστάσεις του. Υπάρχουν, επίσης, λίγοι κόμβοι που διατηρούν έως 40 δεδομένα, πολλοί λιγότεροι κόμβοι που διατηρούν περίπου 80 δεδομένα και ελάχιστοι κόμβοι που διατηρούν περισσότερα από 80 δεδομένα. Επομένως, τα δεδομένα που εισάγονται στο σύστημα κατανέμονται σχεδόν ομοιόμορφα στους κόμβους.

Σημειώνεται ότι τα παραπάνω αποτελέσματα είναι παρόμοια για τα συστήματα των 500, 1000, 4000 ή 8000 κόμβων.

Από την άλλη, όταν εξετάζουμε το σύστημα του TreeChord στο οποίο τα αναγνωριστικά των δεδομένων δεν είναι ομοιόμορφα κατανεμημένα στο δακτύλιο, αλλά ακολουθούν κάποια άλλη κατανομή, όπως π.χ. είναι η εκθετική, οι διακυμάνσεις που παρατηρούνται στο φορτίο που φέρουν οι κόμβοι του συστήματος δεν είναι, επίσης, έντονες. Πιο συγκεκριμένα, οι περισσότεροι κόμβοι διατηρούν περίπου 10 δεδομένα, υπάρχουν λίγοι κόμβοι που διατηρούν έως και διπλάσιο πλήθος δεδομένων, δηλαδή 20 δεδομένα, ενώ υπάρχουν και πολλοί λίγοι κόμβοι που διατηρούν περίπου 40 δεδομένα και ελάχιστοι με πολλά περισσότερα δεδομένα. Αυτό οφείλεται ακριβώς στη δενδρική δομή που χρησιμοποιεί το TreeChord, η οποία κατανέμει τα δεδομένα σχεδόν ομοιόμορφα στο σύστημα κόβοντας το δακτύλιο κατάλληλα, χωρίς να επηρεάζεται ιδιαίτερα από τη μη ομοιόμορφη κατανομή των τιμών των συνιστωσών των δεδομένων εισόδου.

Όταν εξετάζουμε το σύστημα του MAAN στο οποίο εφαρμόζεται η απλή συνάρτηση κατακερματισμού που περιγράφεται στην Εξίσωση (4.1) και τα αναγνωριστικά των δεδομένων εισόδου δεν είναι ομοιόμορφα κατανεμημένα στο δακτύλιο, αλλά ακολουθούν κάποια άλλη κατανομή, όπως π.χ. είναι η εκθετική, τότε παρατηρούνται σημαντικές διακυμάνσεις στο φορτίο που φέρουν οι κόμβοι του συστήματος. Πιο συγκεκριμένα, σχεδόν οι μισοί κόμβοι του συστήματος διατηρούν καθόλου ή ελάχιστα δεδομένα, ενώ οι περισσότεροι από τους υπόλοιπους φέρουν τεράστιο φορτίο, έως και 500 δεδομένα, τιμή που απέχει κατά πολύ από το αναμενόμενο πλήθος δεδομένων που θα έπρεπε να διατηρούν οι κόμβοι αν όλα τα δεδομένα κατανέμονταν τέλεια ανάμεσα στους κόμβους του συστήματος. Αυτό οφείλεται στη συνάρτηση κατακερματισμού που εφαρμόζεται στο σύστημα, η οποία παρότι διατηρεί τη διάταξη, δεν παράγει ομοιόμορφες τιμές αν οι τιμές των αναγνωριστικών δεν είναι ομοιόμορφα κατανεμημένες.

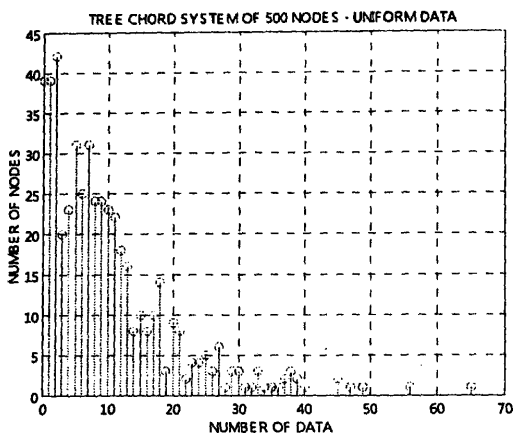
Όταν τα αναγνωριστικά των δεδομένων ακολουθούν την ομοιόμορφη κατανομή και εφαρμόζεται η ομοιόμορφη συνάρτηση κατακερματισμού που περιγράφεται στην Εξίσωση (4.2), τότε παρατηρούνται και πάλι σημαντικές διακυμάνσεις στο φορτίο που φέρουν οι κόμβοι. Και στην περίπτωση αυτή υπάρχουν λίγοι κόμβοι με τεράστιο πλήθος δεδομένων, που μπορεί να ξεπεράσει και τα 500, ενώ οι περισσότεροι κόμβοι (σχεδόν οι μισοί) διατηρούν ελάχιστο πλήθος δεδομένων, έως και μηδενικό. Αυτό οφείλεται στο γεγονός πως η ομοιόμορφη κατανομή δεν έχει τις ιδιότητες που απαιτεί η ομοιόμορφη συνάρτηση κατακερματισμού προκειμένου να παράγει τιμές ομοιόμορφα κατανεμημένες στο εύρος του δακτυλίου του συστήματος.

Επομένως, προκειμένου να γίνει καλός διαμοιρασμός των δεδομένων στο σύστημα του MAAN στο οποίο εφαρμόζεται η απλή συνάρτηση κατακερματισμού που περιγράφεται στην Εξίσωση (4.1), οι συνιστώσες των αναγνωριστικών των δεδομένων σε κάθε διάσταση πρέπει να είναι ομοιόμορφα κατανεμημένες στο δακτύλιο, ενώ όταν εφαρμόζεται η ομοιόμορφη συνάρτηση κατακερματισμού που περιγράφεται στην Εξίσωση (4.2), οι συνιστώσες των αναγνωριστικών των δεδομένων σε κάθε διάσταση πρέπει να ακολουθούν μία συνεχή και μονότονα αύξουσα κατανομή, όπως είναι η εκθετική.

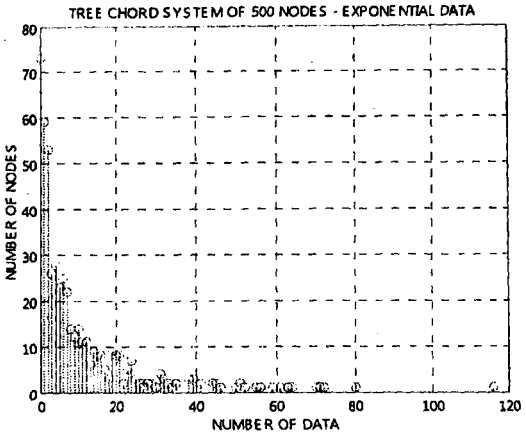
Ακόμη, όταν εξετάζουμε το σύστημα του Mercury στο οποίο τα αναγνωριστικά των δεδομένων δεν είναι ομοιόμορφα κατανεμημένα στο δακτύλιο, αλλά ακολουθούν κάποια άλλη κατανομή, όπως π.χ. είναι η εκθετική, οι διακυμάνσεις που παρατηρούνται στο φορτίο που φέρουν οι κόμβοι του συστήματος είναι, επίσης, έντονες. Πιο συγκεκριμένα, σχεδόν οι μισοί κόμβοι του συστήματος διατηρούν από καθόλου έως ελάχιστα δεδομένα, ενώ οι περισσότεροι από τους υπόλοιπους κόμβους φέρουν φορτίο τεράστιο, που ξεπερνά ακόμη και τα 400 δεδομένα. Παρατηρούμε, λοιπόν, πως το φορτίο των περισσότερων κόμβων του συστήματος είναι κατά πολύ μεγαλύτερο ή μικρότερο από αυτό που θα έφεραν οι κόμβοι αν η κατανομή των δεδομένων ανάμεσα σε όλους τους κόμβους του συστήματος ήταν τέλεια. Αυτό οφείλεται ακριβώς στο γεγονός πως το Mercury δεν καταφέρνει για κάποιες κατανομές να κατανείμει καλά τα δεδομένα. Επομένως, προκειμένου να γίνει όσο το δυνατόν καλύτερος διαμοιρασμός των δεδομένων εισόδου στο σύστημα του Mercury, οι συνιστώσες των αναγνωριστικών των δεδομένων σε κάθε διάσταση πρέπει να είναι ομοιόμορφα κατανεμημένες στο δακτύλιο.

Παρατηρούμε, λοιπόν, πως είτε εφαρμόζεται η απλή συνάρτηση κατακερματισμού που περιγράφεται στην Εξίσωση (4.1) και οι συνιστώσες των αναγνωριστικών των δεδομένων σε κάθε διάσταση ακολουθούν την ομοιόμορφη κατανομή, είτε εφαρμόζεται η ομοιόμορφη συνάρτηση κατακερματισμού που περιγράφεται στην Εξίσωση (4.2) και οι συνιστώσες των αναγνωριστικών των δεδομένων σε κάθε διάσταση ακολουθούν μία συνεχή και μονότονα αύξουσα κατανομή, όπως είναι η εκθετική, τότε το σύστημα του MAAN εγγυάται ότι τα δεδομένα κατανέμονται ομοιόμορφα. Και στο σύστημα του Mercury, όμως, όταν τα αναγνωριστικά των δεδομένων ακολουθούν την ομοιόμορφη κατανομή, τότε τα δεδομένα κατανέμονται, επίσης, σχεδόν ομοιόμορφα στους κόμβους του συστήματος. Το ίδιο συμβαίνει και στο σύστημα του TreeChord, το οποίο δεν προκαλεί αλλοιώσεις στην ομοιόμορφη κατανομή που ακολουθούν τα αναγνωριστικά των δεδομένων, αλλά επιτρέπει στα δεδομένα να κατανέμονται σχεδόν ομοιόμορφα στους κόμβους του συστήματος.

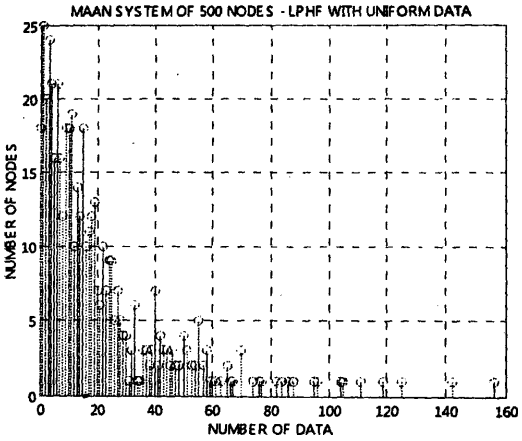
Είναι αξιοσημείωτο, όμως, ότι το σύστημα του TreeChord υπερέχει σημαντικά των άλλων συστημάτων όταν τα δεδομένα ακολουθούν την εκθετική κατανομή, καθώς, επίσης, και του συστήματος του MAAN στο οποίο εφαρμόζεται η απλή συνάρτηση κατακερματισμού που περιγράφεται στην Εξίσωση (4.1) και οι συνιστώσες των αναγνωριστικών των δεδομένων σε κάθε διάσταση ακολουθούν την εκθετική κατανομή ή εφαρμόζεται η ομοιόμορφη συνάρτηση κατακερματισμού που περιγράφεται στην Εξίσωση (4.2) και οι συνιστώσες των αναγνωριστικών των δεδομένων σε κάθε διάσταση ακολουθούν την ομοιόμορφη κατανομή. Στην περίπτωση αυτή, το σύστημα του TreeChord υπερέχει, καθώς είναι το μόνο σύστημα που επιτυγχάνει να κατανείμει τα δεδομένα που εισάγονται στο σύστημα σχεδόν ομοιόμορφα ανάμεσα στους συμμετέχοντες κόμβους. Σε όλα τα άλλα συστήματα, η κατανομή των δεδομένων στους κόμβους είναι κάθε άλλο παρά ομοιόμορφη, αφού σχεδόν οι μισοί κόμβοι που συμμετέχουν σε αυτά διατηρούν ελάχιστα ή καθόλου δεδομένα, ενώ και η πλειοψηφία των εισαγόμενων δεδομένων συγκεντρώνεται σε ένα μικρό αριθμό κόμβων των συστημάτων. Σημειώνεται ότι τα αποτελέσματα είναι παρόμοια για τα συστήματα των 500, 1000, 4000 ή 8000 κόμβων, ενώ το σύστημα του TreeChord υπερέχει σε κάθε περίπτωση. Επομένως, προκειμένου να γίνει όσο το δυνατόν καλύτερος διαμοιρασμός των δεδομένων εισόδου σε ένα σύστημα, στο TreeChord οι συνιστώσες των αναγνωριστικών των δεδομένων σε κάθε διάσταση δεν πρέπει να ακολουθούν απαραίτητα κάποια συγκεκριμένη κατανομή, σε αντίθεση με τα άλλα συστήματα όπου πρέπει να τηρούνται κάποιες προϋποθέσεις.



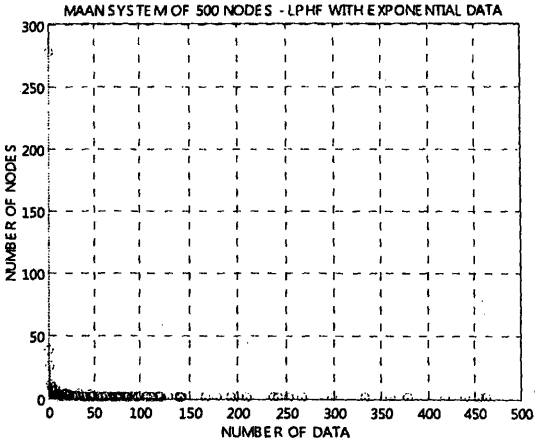
(α)



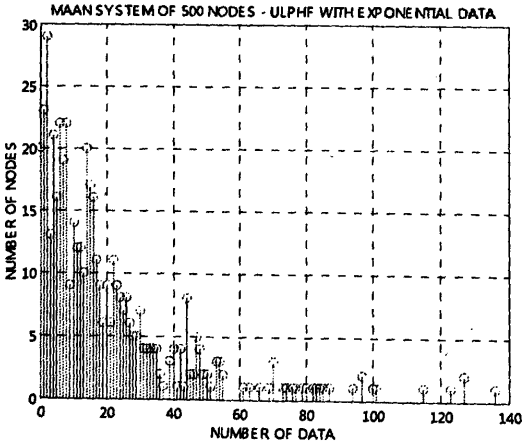
(ε)



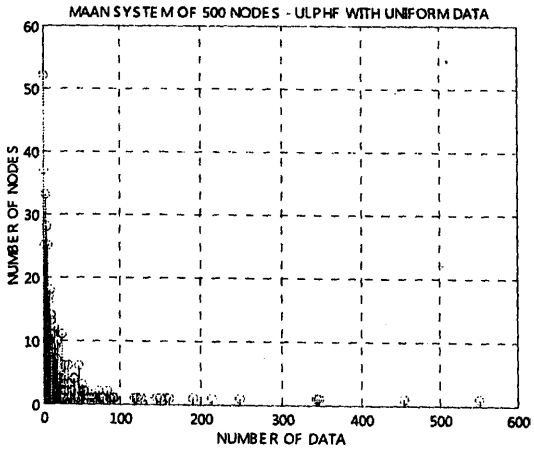
(β)



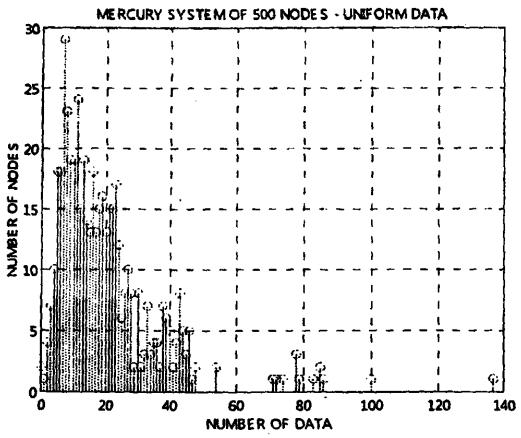
(ζ)



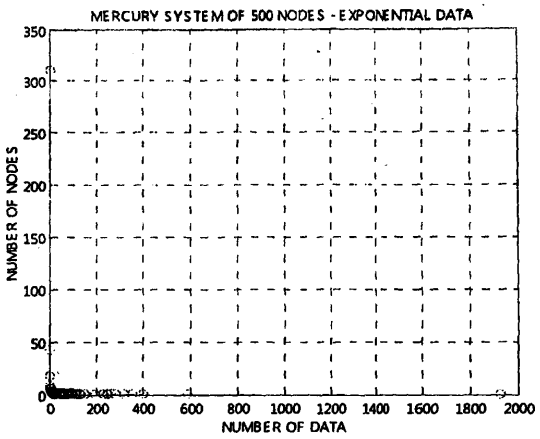
(γ)



(η)

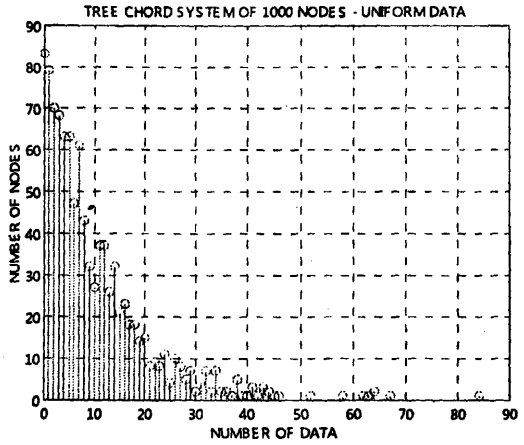


(δ)

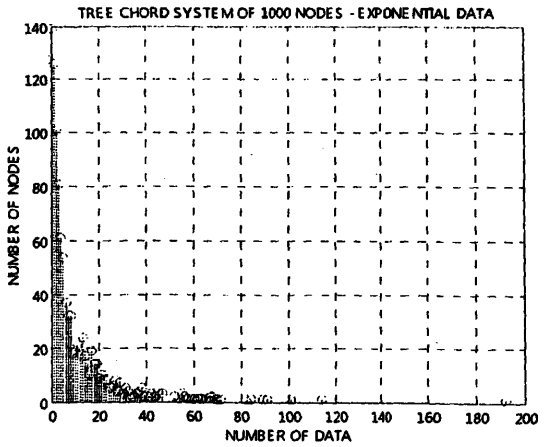


(θ)

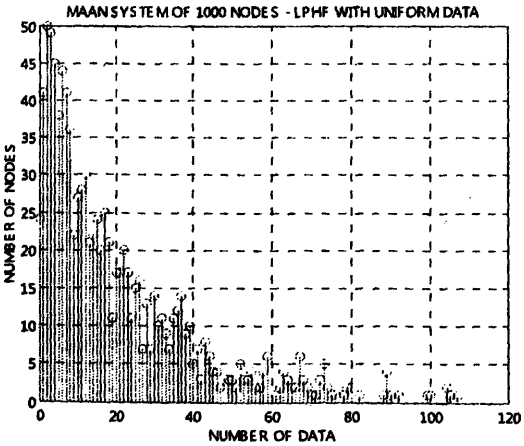
Σχήμα 10.1 Πλήθος Κόμβων που Διατηρούν ακριβώς i Δεδομένα, $i \geq 0$, στα Συστήματα του TreeChord, του MAAN και του Mercury με 500 Κόμβους. Τα Δεδομένα Ακολουθούν την Ομοιόμορφη ή την Εκθετική Κατανομή.



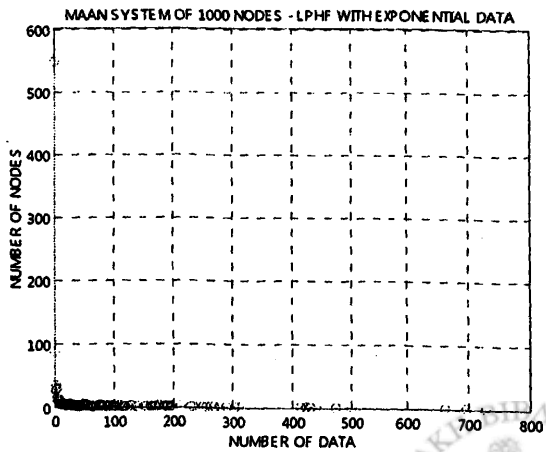
(α)



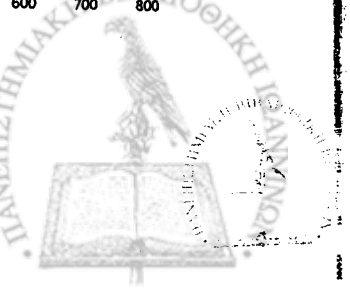
(ε)

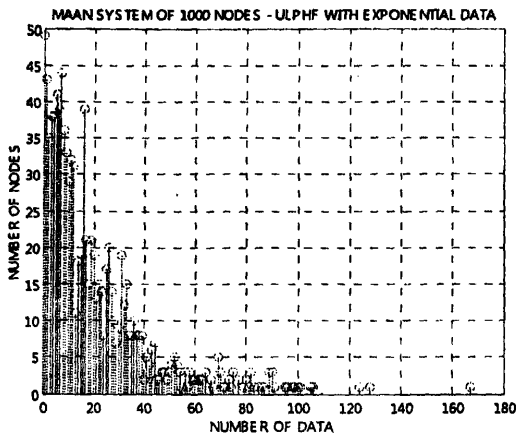


(β)

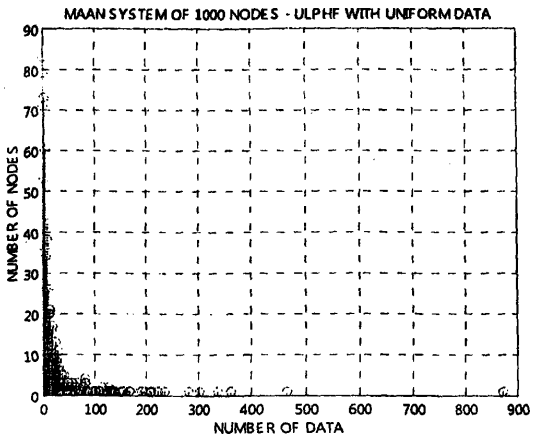


(ζ)

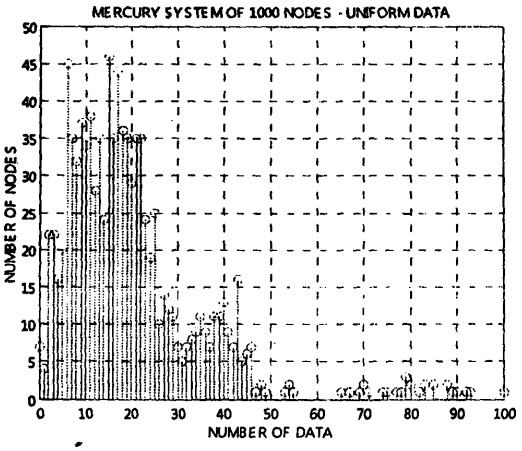




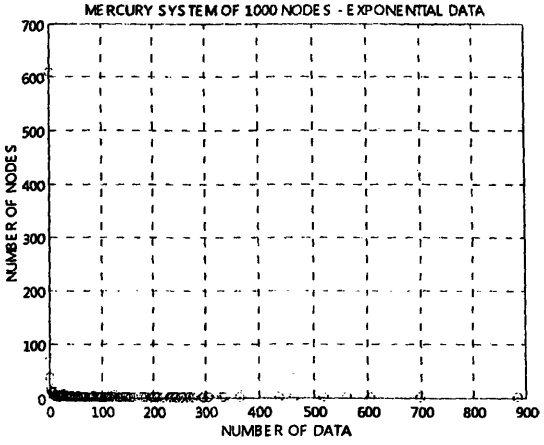
(γ)



(η)

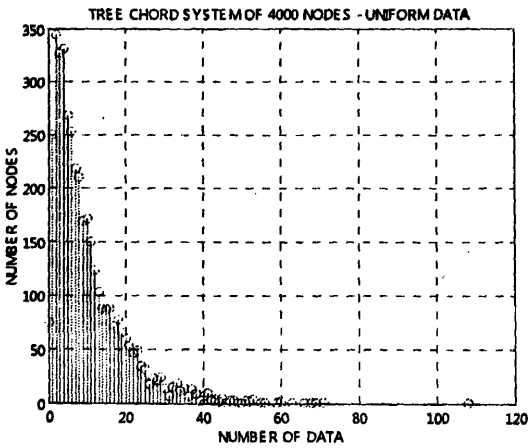


(δ)

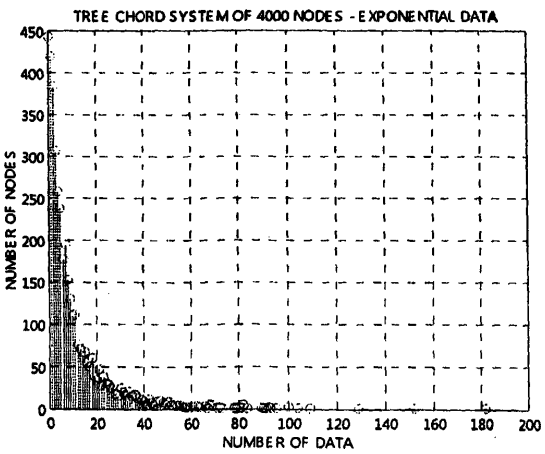


(θ)

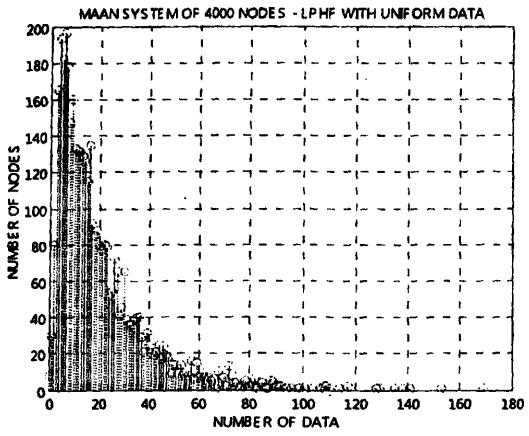
Σχήμα 10.2 Πλήθος Κόμβων που Διατηρούν ακριβώς i Δεδομένα, $i \geq 0$, στα Συστήματα του TreeChord, του MAAN και του Mercury με 1000 Κόμβους. Τα Δεδομένα Ακολουθούν την Ομοιόμορφη ή την Εκθετική Κατανομή.



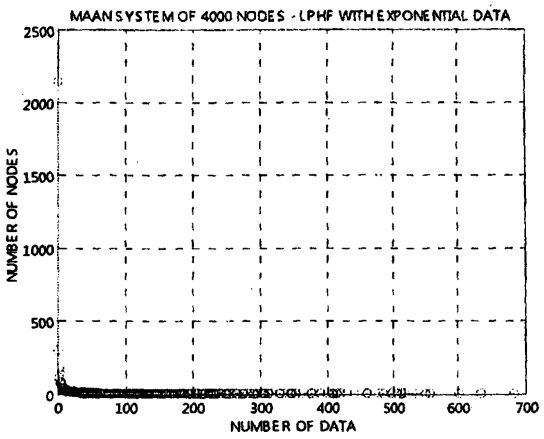
(α)



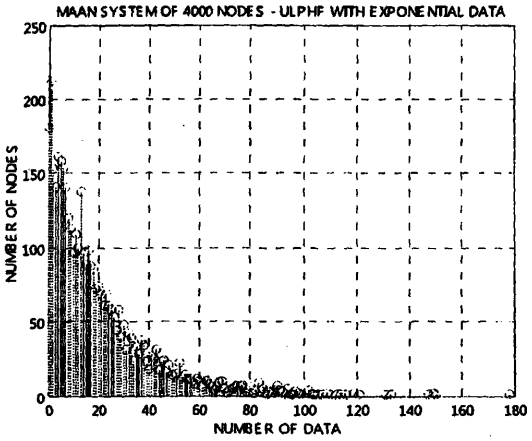
(ε)



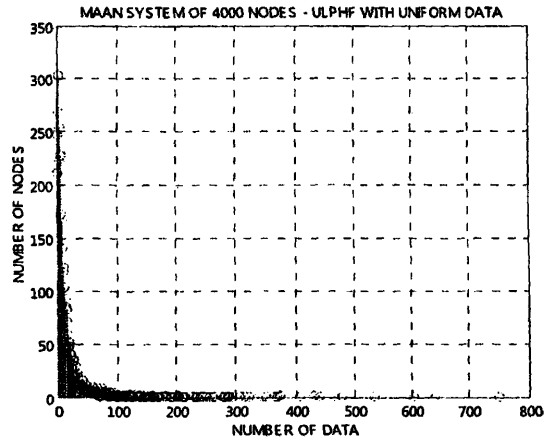
(β)



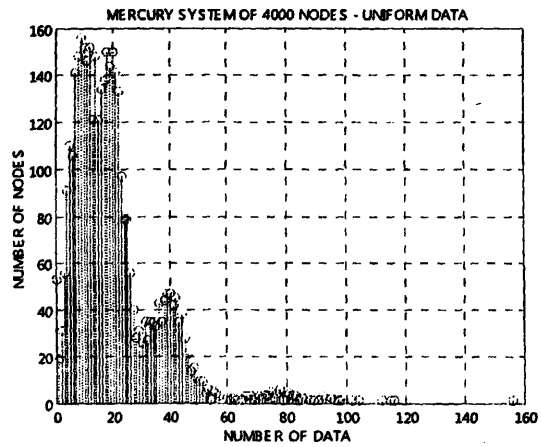
(ζ)



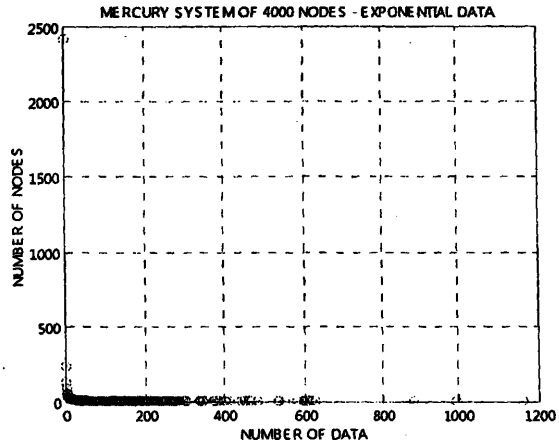
(γ)



(η)

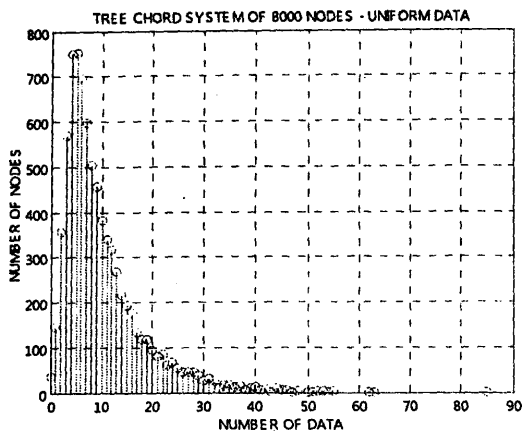


(δ)

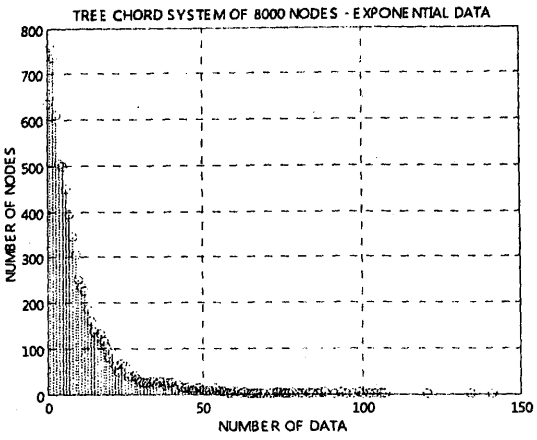


(θ)

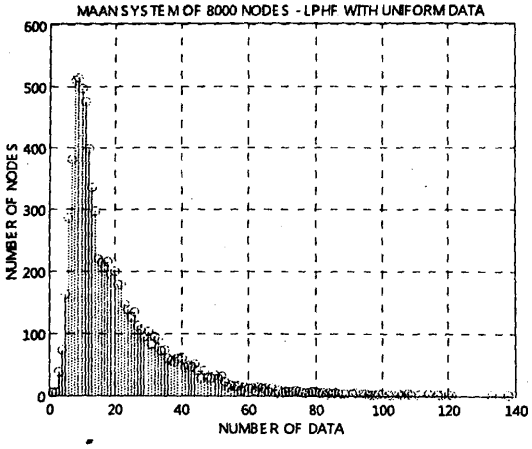
Σχήμα 10.3 Πλήθος Κόμβων που Διατηρούν ακριβώς i Δεδομένα, $i \geq 0$, στα Συστήματα του TreeChord, του MAAN και του Mercury με 4000 Κόμβους. Τα Δεδομένα Ακολουθούν την Ομοιόμορφη ή την Εκθετική Κατανομή.



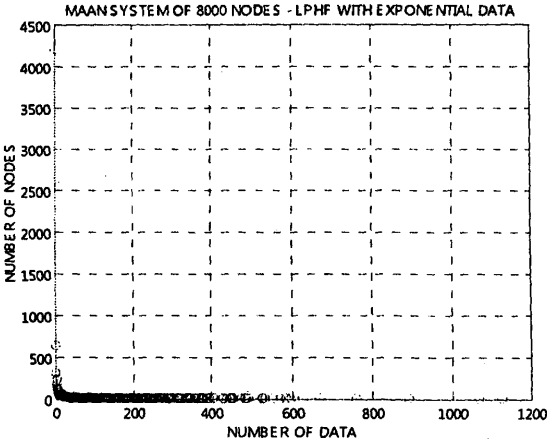
(α)



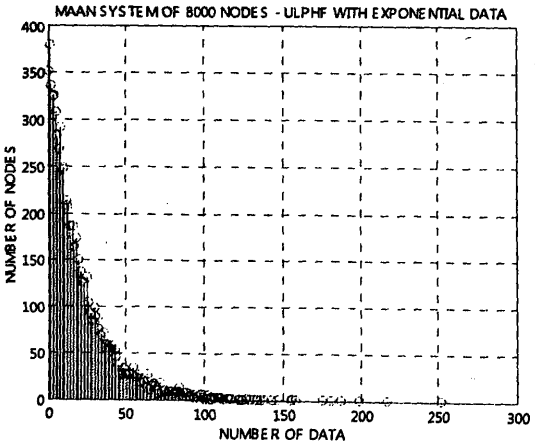
(ε)



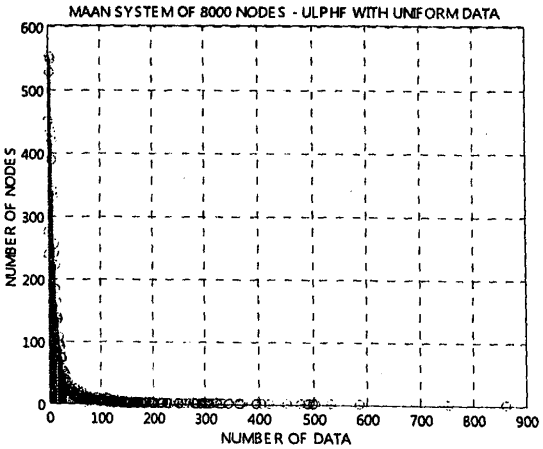
(β)



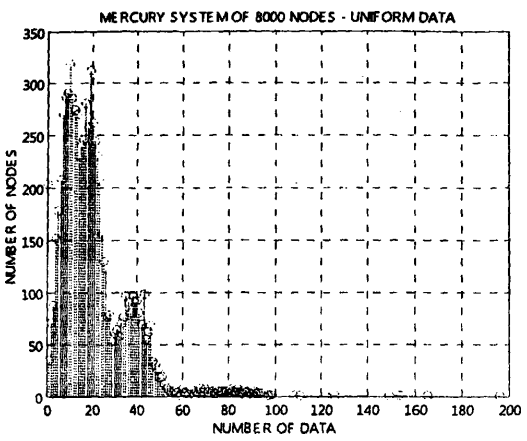
(ζ)



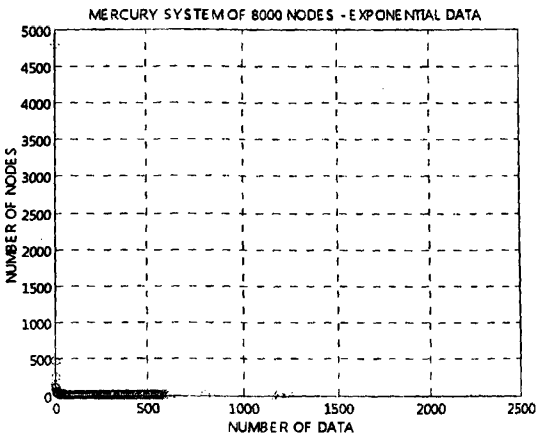
(γ)



(η)



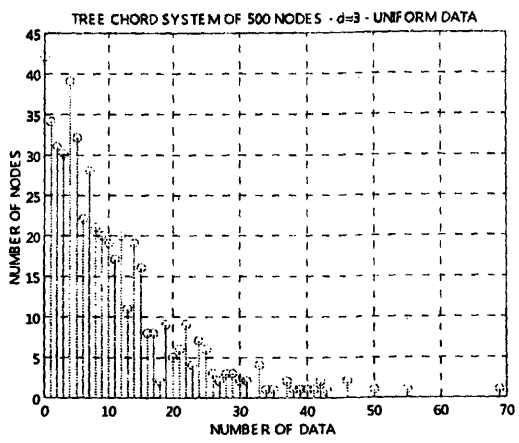
(δ)



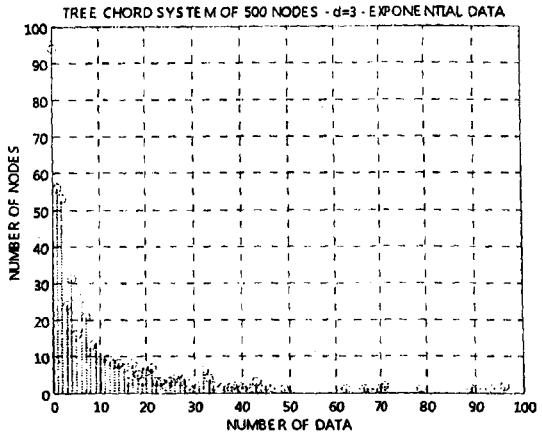
(θ)

Σχήμα 10.4 Πλήθος Κόμβων που Διατηρούν ακριβώς i Δεδομένα, $i \geq 0$, στα Συστήματα του TreeChord, του MAAN και του Mercury με 8000 Κόμβους. Τα Δεδομένα Ακολουθούν την Ομοιόμορφη ή την Εκθετική Κατανομή.

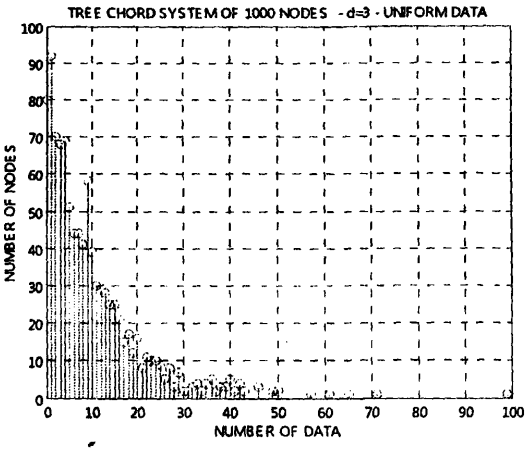
Τέλος, αξίζει να σημειωθεί πως αν αυξήσουμε το βάθος του δένδρου δεδομένων από 2 σε 3, δηλαδή το πλήθος των φύλλων βάσει των οποίων γίνεται εισαγωγή των δεδομένων στο σύστημα από $2^2 = 4$ σε $2^3 = 8$, τότε παρατηρούμε ότι στην περίπτωση που τα αναγνωριστικά των δεδομένων κατανέμονται ομοιόμορφα στο δακτύλιο, μικρές είναι οι διακυμάνσεις που παρατηρούνται στο φορτίο που φέρουν οι κόμβοι του συστήματος, όπως είναι αναμενόμενο. Στην περίπτωση, όμως, που τα αναγνωριστικά των δεδομένων ακολουθούν την εκθετική κατανομή, οι διακυμάνσεις που παρατηρούνται στο φορτίο που φέρουν οι κόμβοι του συστήματος είναι λιγότερο έντονες, συγκρινόμενες με τα αντίστοιχα συστήματα του TreeChord, όπου το δένδρο έχει βάθος ίσο με 2. Καθώς αυξάνεται το πλήθος των φύλλων του δένδρου δεδομένων, ο δακτύλιος χωρίζεται σε περισσότερα τμήματα προκειμένου να διαμοιραστούν τα δεδομένα και, συνεπώς, τα δεδομένα κατανέμονται ακόμη καλύτερα στο δακτύλιο του συστήματος (Σχήμα 10.5).



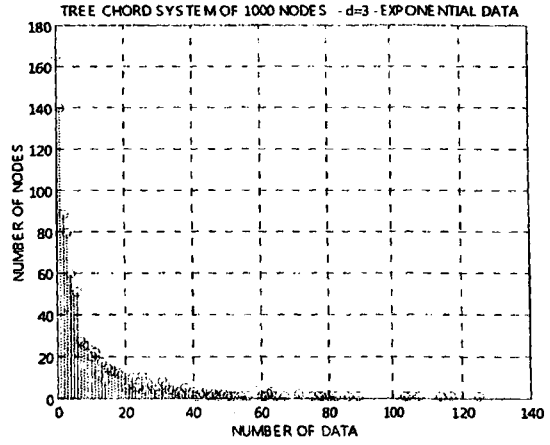
(α)



(β)



(γ)



(δ)

Σχήμα 10.5 Πλήθος Κόμβων που Διατηρούν ακριβώς i Δεδομένα, $i \geq 0$, στο Σύστημα του TreeChord με βάθος δένδρου $d = 3$. Τα Δεδομένα Ακολουθούν την Ομοιόμορφη ή την Εκθετική Κατανομή.

Στο Παράρτημα Α μπορούμε να παρατηρήσουμε τα ίδια αποτελέσματα με μεγαλύτερη λεπτομέρεια, αφού στις γραφικές παραστάσεις που παρουσιάζονται απεικονίζεται το ακριβές πλήθος δεδομένων που διατηρεί κάθε κόμβος ενός συστήματος.

10.1.2. Ποσοστά Κόμβων με Μη Εξισορροπημένο Φορτίο

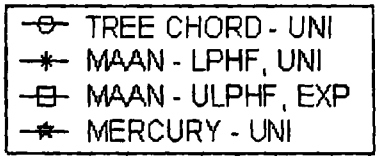
Στα πειράματα αυτά εξετάζουμε το ποσοστό των υπερφορτωμένων ή υποφορτωμένων κόμβων (δηλαδή των κόμβων με μη εξισορροπημένο φορτίο) για κάθε από τα συστήματα που μελετάμε. Υπενθυμίζουμε ότι θεωρούμε ως μέσο φορτίο *avgLoad* του συστήματος το λόγο του συνολικού πλήθους δεδομένων που έχουν εισαχθεί στο σύστημα προς το συνολικό πλήθος κόμβων που συμμετέχουν στο σύστημα. Υπενθυμίζουμε, επίσης, ότι αν n είναι ένας

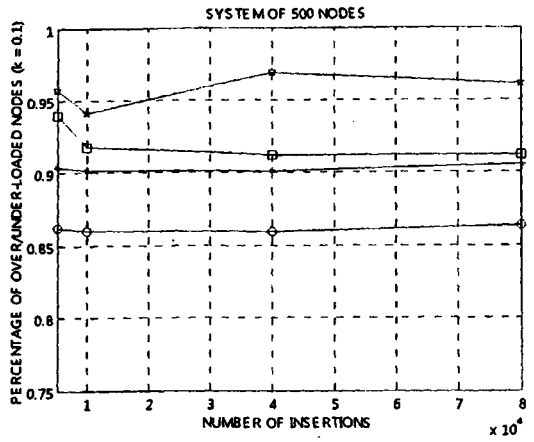
κόμβος του συστήματος και n_d είναι το πλήθος των δεδομένων που διατηρεί, τότε για κάποια σταθερά k της οποίας η τιμή καθορίζεται στη συνέχεια, αν $n_d > (1+k) \times avgLoad$, θεωρούμε ότι ο κόμβος είναι υπερφορτωμένος (*overloaded*), ενώ αν $n_d < (1-k) \times avgLoad$, θεωρούμε ότι ο κόμβος είναι υποφορτωμένος (*underloaded*).

Μελετούμε και πάλι συστήματα των 500, 1000, 4000 και 8000 κόμβων. Σε κάθε ένα από αυτά τα συστήματα εισάγουμε 5000, 10000, 40000 και 80000 δεδομένα και υπολογίζουμε το ποσοστό των κόμβων με μη εξισορροπημένο φορτίο για $k = 0.1$, $k = 0.5$ και $k = 0.9$, δηλαδή το ποσοστό των κόμβων που φέρουν φορτίο κατά 10%, 50% και 90% μεγαλύτερο ή μικρότερο από το μέσο φορτίο του συστήματος. Υπενθυμίζουμε ότι για την κατανομή των δεδομένων στους κόμβους του συστήματος ακολουθούνται όσα περιγράφηκαν στα πειράματα της πρώτης κατηγορίας.

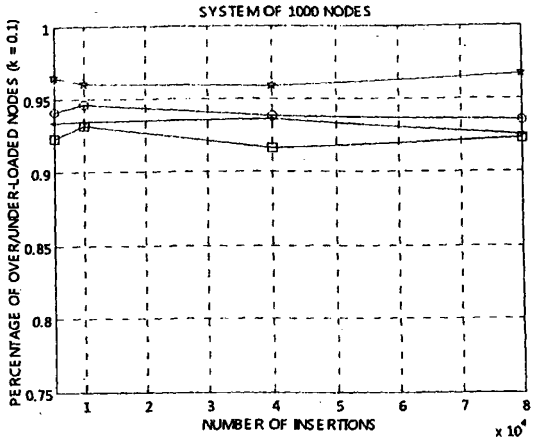
Στη συνέχεια εξετάζουμε τα συστήματα TreeChord και Mercury στα οποία τα αναγνωριστικά των δεδομένων εισόδου ακολουθούν την ομοιόμορφη κατανομή μαζί με το σύστημα του MAAN στο οποίο οι δύο συναρτήσεις κατακερματισμού συνδυάζονται με τις κατάλληλες κατανομές δεδομένων εισόδου που επιτρέπουν στα δεδομένα να κατανεμηθούν ομοιόμορφα ανάμεσα στους κόμβους.

Στις πρώτες γραφικές παραστάσεις (Σχήμα 10.6) απεικονίζεται το ποσοστό των κόμβων που φέρουν φορτίο κατά 10% μεγαλύτερο ή μικρότερο από το μέσο φορτίο του συστήματος. Παρατηρούμε, λοιπόν, πως σε όλα τα συστήματα μεγάλο ποσοστό των κόμβων (σχεδόν πάντα μεγαλύτερο από 90%) είναι κόμβοι με μη εξισορροπημένο φορτίο. Σε κάποιες περιπτώσεις το σύστημα του TreeChord φαίνεται να υπερτερεί των άλλων, αφού το ποσοστό των κόμβων με μη εξισορροπημένο φορτίο είναι κάτω από το 90%.

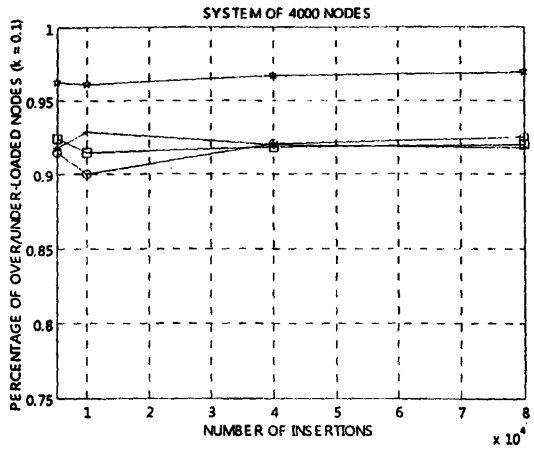




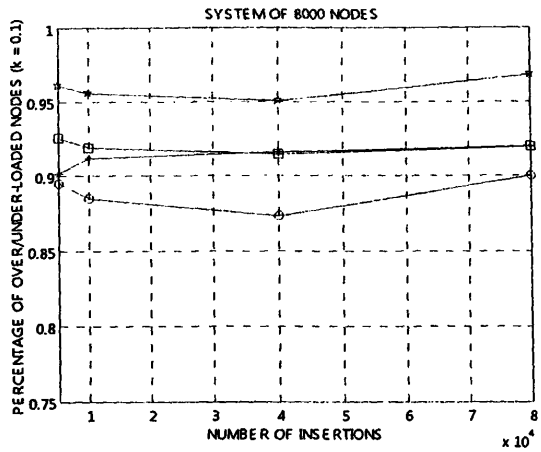
(α)



(β)



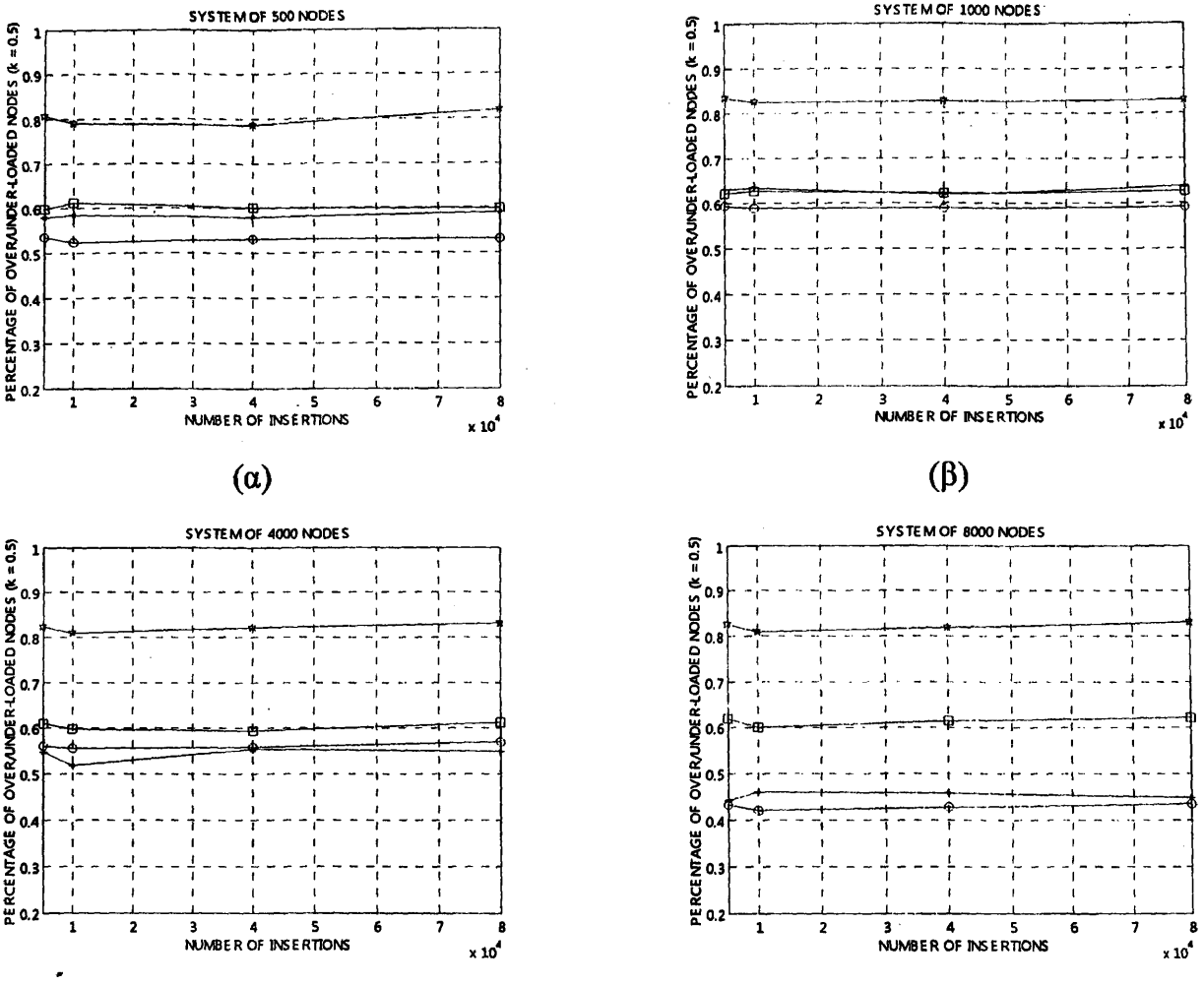
(γ)



(δ)

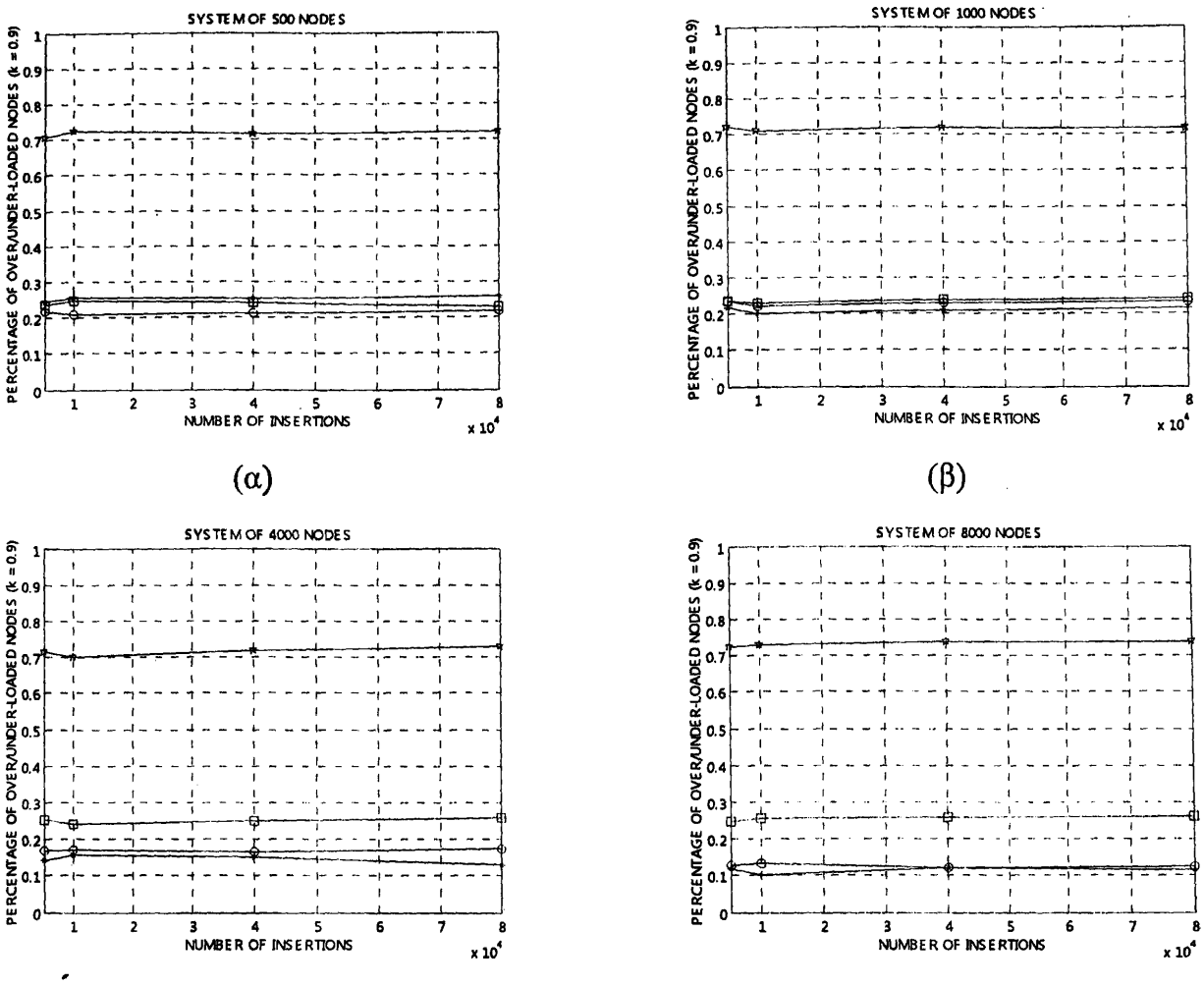
Σχήμα 10.6 Ποσοστό Υπερφορτωμένων ή Υποφορτωμένων Κόμβων για $k = 0.1$.

Στις επόμενες γραφικές παραστάσεις (Σχήμα 10.7) απεικονίζεται το ποσοστό των κόμβων που φέρουν φορτίο κατά 50% μεγαλύτερο ή μικρότερο από το μέσο φορτίο του συστήματος. Παρατηρούμε, λοιπόν, πως σε όλες τις περιπτώσεις το ποσοστό των κόμβων με μη εξισορροπημένο φορτίο για το σύστημα του Mercury είναι περίπου 80%. Στο σύστημα του MAAN το ποσοστό αυτό κυμαίνεται από 50 έως 60%, ενώ στο σύστημα του TreeChord το ποσοστό αυτό είναι τις περισσότερες φορές 5 – 10% μικρότερο. Το πόσο σημαντική είναι αυτή η διαφορά γίνεται αντιληπτό αν αναλογιστούμε πως η διαφορά του 5 – 10% μεταξύ των συστημάτων του TreeChord και του MAAN και η διαφορά του 30% μεταξύ των συστημάτων του TreeChord και του Mercury για το σύστημα των 8000 κόμβων ισοδυναμεί με 400 και 2400 λιγότερους κόμβους με μη εξισορροπημένο φορτίο, αντίστοιχα.



Σχήμα 10.7 Ποσοστό Υπερφορτωμένων ή Υποφορτωμένων Κόμβων για $k = 0.5$.

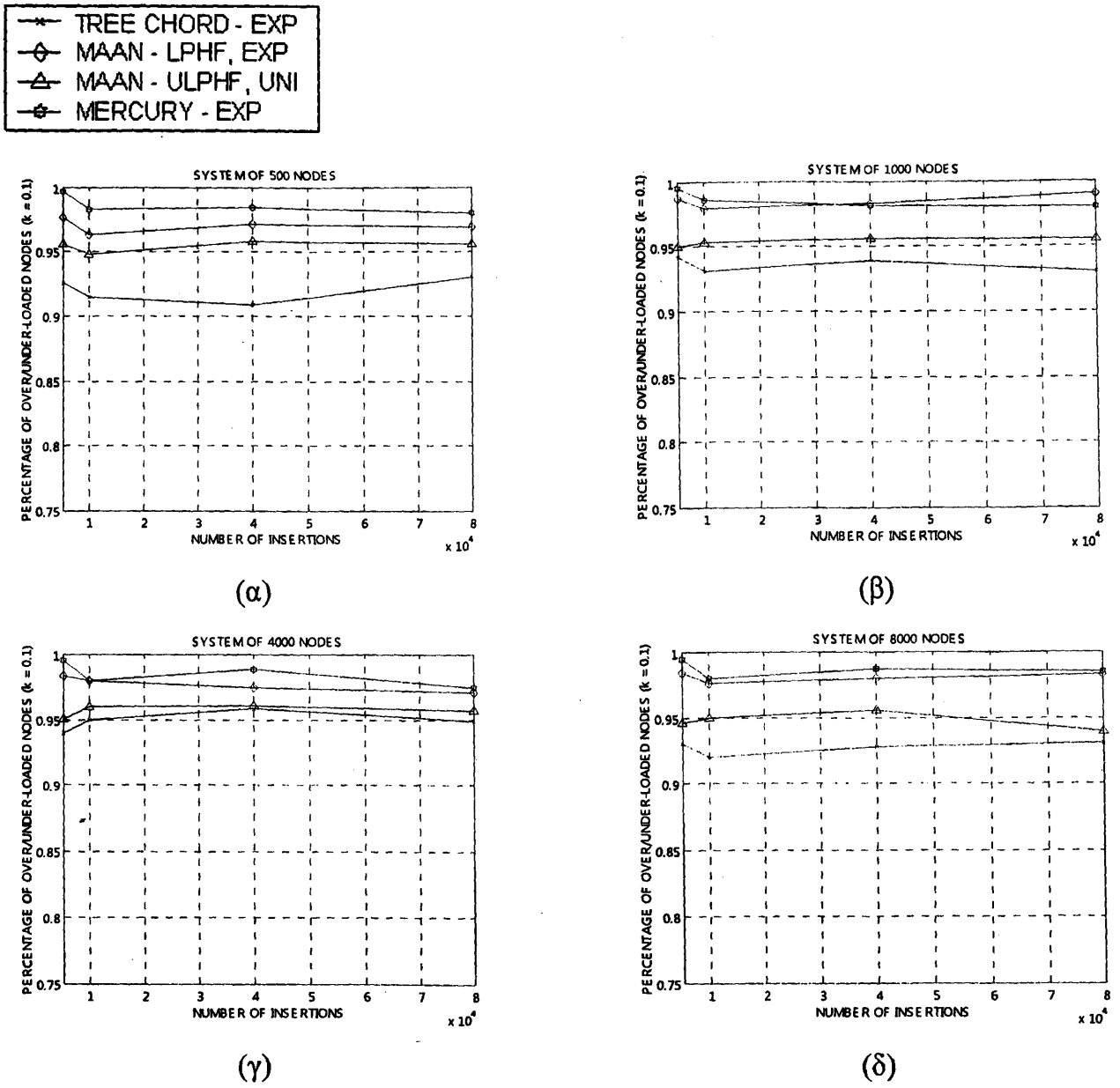
Στις ακόλουθες γραφικές παραστάσεις (Σχήμα 10.8) απεικονίζεται το ποσοστό των κόμβων που φέρουν φορτίο κατά 90% μεγαλύτερο ή μικρότερο από το μέσο φορτίο του συστήματος. Παρατηρούμε, λοιπόν, πως σε όλες τις περιπτώσεις το ποσοστό των κόμβων με μη εξισορροπημένο φορτίο για το σύστημα του Mercury είναι περίπου 70%. Στο σύστημα του MAAN το ποσοστό αυτό κυμαίνεται από 15 έως 25%, ενώ στο σύστημα του TreeChord το ποσοστό αυτό είναι τις περισσότερες φορές μικρότερο.



Σχήμα 10.8 Ποσοστό Υπερφορτωμένων ή Υποφορτωμένων Κόμβων για $k = 0.9$.

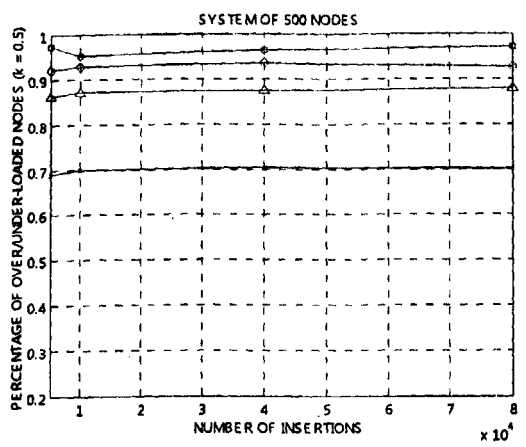
Στις επόμενες γραφικές παραστάσεις εξετάζουμε τα συστήματα TreeChord και Mercury στα οποία τα αναγνωριστικά των δεδομένων εισόδου ακολουθούν την εκθετική κατανομή μαζί με το σύστημα του MAAN στο οποίο οι δύο συναρτήσεις κατακερματισμού συνδυάζονται με ακατάλληλες κατανομές δεδομένων εισόδου, που δεν επιτρέπουν στα δεδομένα να κατανεμηθούν ομοιόμορφα ανάμεσα στους κόμβους.

Στις πρώτες γραφικές παραστάσεις (Σχήμα 10.9) απεικονίζεται το ποσοστό των κόμβων που φέρουν φορτίο κατά 10% μεγαλύτερο ή μικρότερο από το μέσο φορτίο του συστήματος. Παρατηρούμε, λοιπόν, πως σε όλα τα συστήματα μεγάλο ποσοστό των κόμβων (σχεδόν πάντα μεγαλύτερο από 90%) είναι κόμβοι με μη εξισορροπημένο φορτίο. Σε όλες τις περιπτώσεις, όμως, το σύστημα του TreeChord φαίνεται να υπερτερεί των άλλων, αφού το ποσοστό των κόμβων με μη εξισορροπημένο φορτίο μειώνεται λίγο.

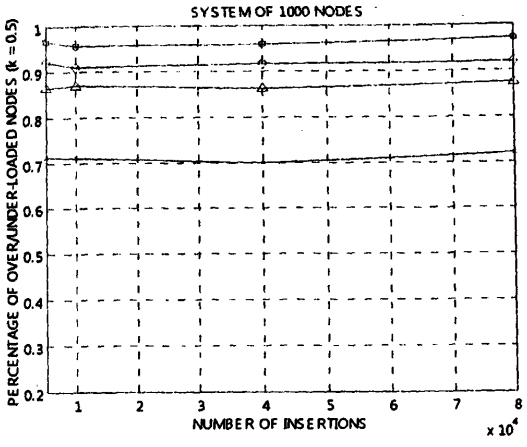


Σχήμα 10.9 Ποσοστό Υπερφορτωμένων ή Υποφορτωμένων Κόμβων για $k = 0.1$.

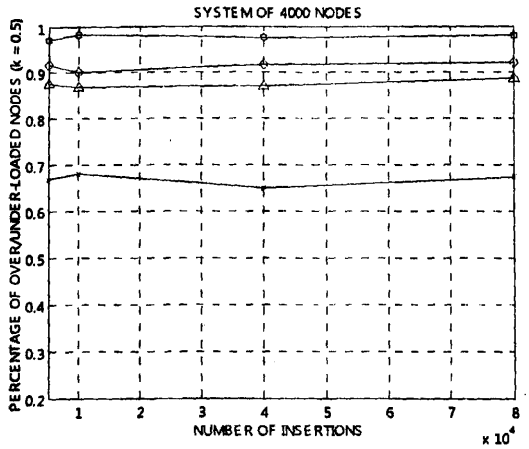
Στις επόμενες γραφικές παραστάσεις (Σχήμα 10.10) απεικονίζεται το ποσοστό των κόμβων που φέρουν φορτίο κατά 50% μεγαλύτερο ή μικρότερο από το μέσο φορτίο του συστήματος. Παρατηρούμε πως ενώ στο σύστημα του TreeChord το ποσοστό των κόμβων με μη εξισορροπημένο φορτίο μειώνεται στο 70% περίπου ή ακόμη περισσότερο, σε όλα τα άλλα συστήματα το ποσοστό παραμένει υψηλό (περίπου 90%). Η υπεροχή του TreeChord, λοιπόν, σε όλες τις περιπτώσεις είναι μεγάλη και περίπου ίση με 20%.



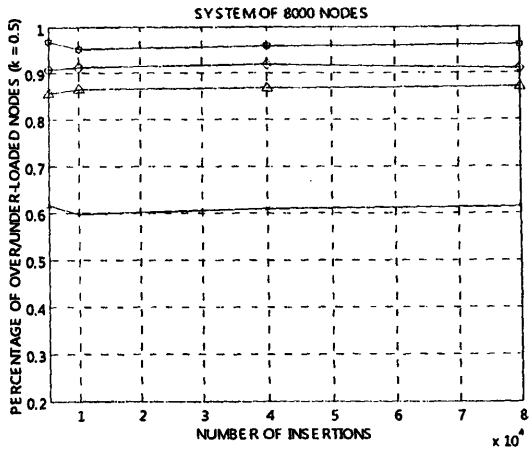
(α)



(β)



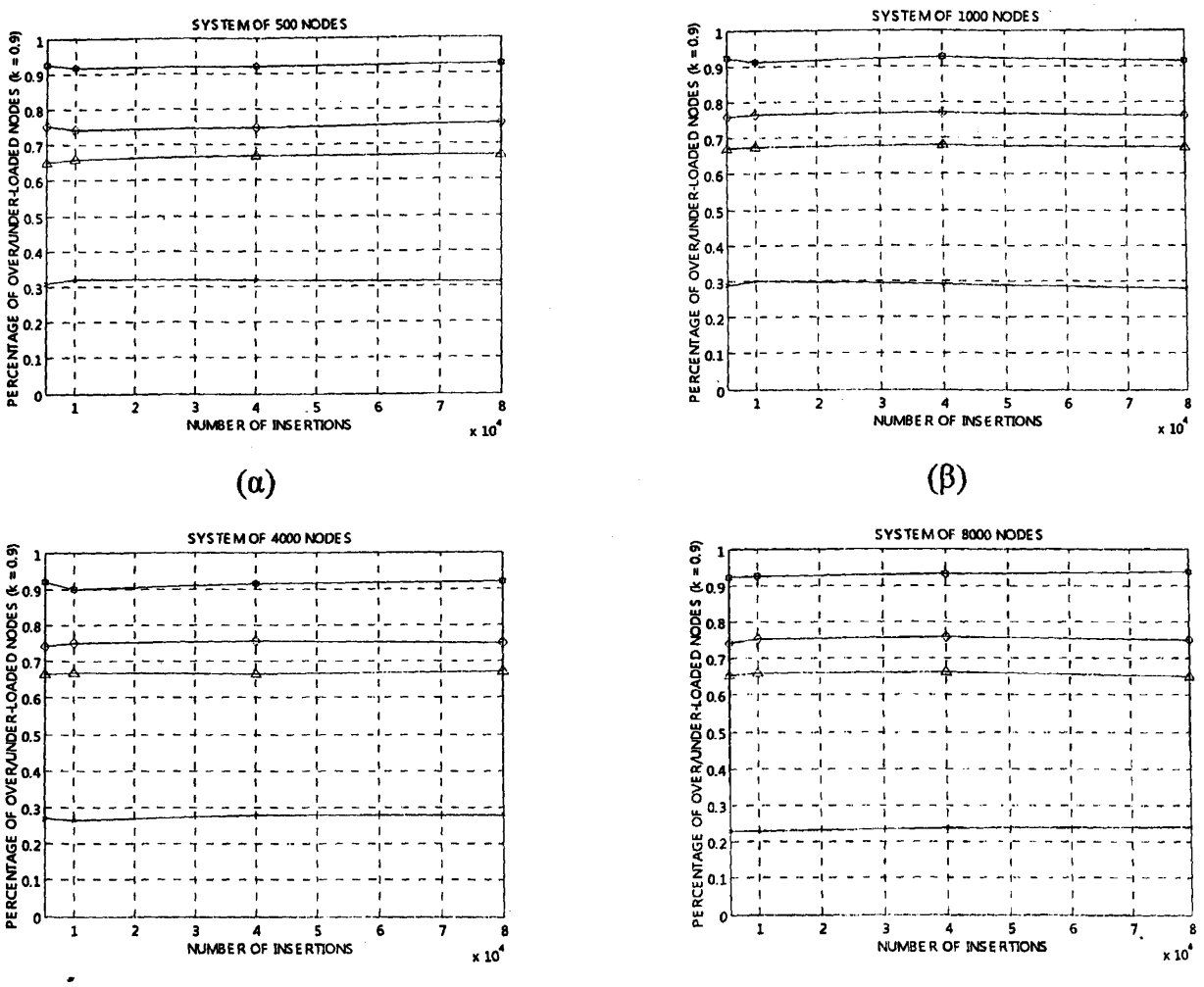
(γ)



(δ)

Σχήμα 10.10 Ποσοστό Υπερφορτωμένων ή Υποφορτωμένων Κόμβων για $k = 0.5$.

Τέλος, στις επόμενες γραφικές παραστάσεις (Σχήμα 10.11) απεικονίζεται το ποσοστό των κόμβων που φέρουν φορτίο κατά 90% μεγαλύτερο ή μικρότερο από το μέσο φορτίο του συστήματος. Στην περίπτωση αυτή παρατηρούμε πως ενώ στο σύστημα του TreeChord το ποσοστό των κόμβων με μη εξισορροπημένο φορτίο μειώνεται στο 30% περίπου ή ακόμη περισσότερο, σε όλα τα άλλα συστήματα το ποσοστό παραμένει υψηλό (περίπου 90% και 70% για το σύστημα του Mercury και του MAAN, αντίστοιχα). Το σύστημα του TreeChord, λοιπόν, σε όλες τις περιπτώσεις βελτιώνει το ποσοστό των κόμβων με μη εξισορροπημένο φορτίο κατά 40 – 60%, τιμή πολύ μεγάλη.



Σχήμα 10.11 Ποσοστό Υπερφορτωμένων ή Υποφορτωμένων Κόμβων για $k = 0.9$.

Επομένως, μπορούμε να πούμε πως το σύστημα του TreeChord καταφέρνει να κατανείμει πολύ καλύτερα τα δεδομένα στους κόμβους του συστήματος, ακόμα κι αν η κατανομή των αναγνωριστικών εισόδου δεν είναι ομοιόμορφη.

Τέλος, αξίζει να σημειωθεί πως το ποσοστό των κόμβων με μη εξισορροπημένο φορτίο δεν επηρεάζεται από το πλήθος των εισαγόμενων δεδομένων στο σύστημα, αφού όπως παρατηρούμε το ποσοστό αυτό παραμένει σχεδόν αμετάβλητο για κάθε σύστημα, όσο κι αν αυξάνεται το πλήθος των δεδομένων που εισάγονται σε κάθε σύστημα.

10.2. Κατανομή Ερωτήσεων Διαστήματος στους Κόμβους

Στα πειράματα που περιγράφονται στη συνέχεια εξετάζουμε πώς κατανέμονται οι ερωτήσεις διαστήματος στους κόμβους που συμμετέχουν στο σύστημα.

10.2.1. Φόρτος Κόμβων Λόγω Συμμετοχής τους στην Επίλυση Ερωτήσεων Διαστήματος

Εξετάζουμε και πάλι τα συστήματα που μελετήθηκαν στα προηγούμενα πειράματα και σε κάθε ένα από αυτά εκτελούμε ένα ενδεικτικό πλήθος ερωτήσεων διαστήματος ($q = 5000$). Σημειώνεται πως η επίλυση μίας ερώτησης διαστήματος είναι μία ιδιαίτερα πολύπλοκη διαδικασία, για αυτό και το πλήθος των 5000 ερωτήσεων διαστήματος θεωρείται ικανοποιητικό. Για κάθε ερώτηση διαστήματος που πραγματοποιείται στο σύστημα δημιουργείται μία ομοιόμορφα τυχαία τιμή για κάθε διάσταση των δεδομένων του συστήματος. Η τιμή αυτή ανήκει στο εύρος τιμών του αντίστοιχου χαρακτηριστικού των δεδομένων που μπορούν να εισαχθούν στο σύστημα, το οποίο είναι σε όλα τα συστήματα και για όλες τις διαστάσεις ίδιο και ίσο με $\left[0, 2^m\right)$, όπου m το απαραίτητο πλήθος bits για την αναπαράσταση του μεγαλύτερου δυνατού αναγνωριστικού στο σύστημα. Στη συνέχεια, το κάτω και το άνω όριο της συγκεκριμένης διάστασης της ερώτησης διαστήματος προκύπτουν αν αφαιρέσουμε ή προσθέσουμε, αντίστοιχα, ένα κατάλληλο offset στην τυχαία τιμή που λάβαμε. Το offset καθορίζεται βάσει της επιλεκτικότητας (selectivity) της ερώτησης διαστήματος. Πιο συγκεκριμένα, σε αυτή την κατηγορία πειραμάτων η επιλεκτικότητα των ερωτήσεων είναι 15% για κάθε διάσταση. Σημειώνεται ότι στην κατηγορία πειραμάτων στο σύστημα του MAAN εφαρμόζεται μόνο ο δεύτερος αλγόριθμος επίλυσης ερωτήσεων διαστήματος που ακολουθεί τη δεύτερη προσέγγιση για να αποφασίζουν οι κόμβοι ποια δεδομένα πρέπει να επιστραφούν, που έχει το μικρότερο κόστος όσον αφορά το πλήθος των κόμβων που συμμετέχουν στην επίλυση μίας ερώτησης διαστήματος.

Στις γραφικές παραστάσεις (Σχήμα 10.12 – Σχήμα 10.14) απεικονίζεται το πλήθος των κόμβων που επεξεργάζονται ακριβώς 1, 2, 3, κλπ. ερωτήσεις διαστήματος ελέγχοντας τα δεδομένα που διαθέτουν τοπικά.

Στο σύστημα του TreeChord μία ερώτηση διαστήματος προωθείται σε $\log N$ κόμβους μέχρι να βρεθεί το πρώτο φύλλο του δένδρου που πιθανόν διαθέτει δεδομένα που ικανοποιούν την ερώτηση. Οι κόμβοι αυτοί δεν ελέγχουν τα δεδομένα που διαθέτουν τοπικά, απλώς προωθούν την ερώτηση στο δακτύλιο και άρα δεν προσμετρώνται. Στη συνέχεια, η ερώτηση προωθείται στα υπόλοιπα κατάλληλα φύλλα από όπου ξεκινούν μονοπάτια κόμβων που πιθανόν διαθέτουν τα κατάλληλα δεδομένα. Όσοι κόμβοι ανήκουν στα μονοπάτια αυτά ελέγχουν τα δεδομένα που διαθέτουν τοπικά και επιστρέφουν όσα ικανοποιούν όλες τις συνθήκες που ορίζονται στην ερώτηση διαστήματος. Καθώς οι κόμβοι κατανέμονται ομοιόμορφα στο δακτύλιο του συστήματος, κάθε ένα από αυτά τα μονοπάτια αναμένεται να αποτελείται από $N \times s$ κόμβους. Παρατηρούμε ότι η κατανομή των δεδομένων προφανώς δεν παίζει κάποιο ρόλο στο πλήθος των κόμβων που θα συμμετέχουν στην απάντηση μίας ερώτησης διαστήματος. Αντίθετα, όπως θα δούμε αργότερα, η κατανομή αυτή επηρεάζει το πλήθος των δεδομένων που επιστρέφει κάθε κόμβος. Το συνολικό πλήθος των φύλλων που ξεκινούν την επεξεργασία μίας ερώτησης διαστήματος είναι μία σταθερά $\sigma \leq 2^d$, όπου d το βάθος του δένδρου, η οποία εξαρτάται από την επιλεκτικότητα της ερώτησης στη δεύτερη διάσταση και προσδιορίζεται πειραματικά.

Στο σύστημα πραγματοποιούνται q ερωτήσεις διαστήματος. Επομένως, αν η κατανομή των κόμβων είναι καλή, τότε κάθε ένας από τους κόμβους του συστήματος θα συμμετάσχει κατά μέσο όρο σε $\sigma \times \frac{q \times n \times s}{n} = \sigma \times q \times s$ ερωτήσεις διαστήματος εξετάζοντας τα δεδομένα που διαθέτει τοπικά. Είναι αξιοσημείωτο πως το πλήθος είναι ανεξάρτητο του πλήθους των κόμβων που έχουν εισαχθεί στο σύστημα. Από πειραματικές μετρήσεις προκύπτει ότι όταν η επιλεκτικότητα των ερωτήσεων διαστήματος που πραγματοποιούνται στο σύστημα είναι 15%, όπως στα πειράματα που πραγματοποιούμε, περίπου 40% των ερωτήσεων διαστήματος οδηγούν στην εξερεύνηση ενός μόνο μονοπατιού, ενώ περίπου 60% των ερωτήσεων διαστήματος οδηγούν στην εξερεύνηση δύο μονοπατιών, με αποτέλεσμα η σταθερά να υπολογίζεται ως εξής:

$$\sigma = \frac{40}{100} \times 1 + \frac{60}{100} \times 2 = 1.6.$$

Αυτό επαληθεύεται στα πειράματα που ακολουθούν.

Στο σύστημα του MAAN, μία ερώτηση διαστήματος προωθείται σε $\log N$ κόμβους μέχρι να βρεθεί το κάτω όριο του ζητούμενου διαστήματος. Οι κόμβοι αυτοί δε συμμετέχουν στον υπολογισμό των απαντήσεων, απλώς προωθούν την ερώτηση στο δακτύλιο. Στη συνέχεια, η ερώτηση προωθείται σε ακόμη $N \times s_k$ κόμβους, όπου k οποιαδήποτε διάσταση. Οι κόμβοι αυτοί συμμετέχουν στον υπολογισμό της απάντησης της ερώτησης εξετάζοντας τα δεδομένα που διαθέτουν τοπικά, έτσι ώστε να αποφασίσουν αν πρέπει να επιστρέψουν κάποια από αυτά στον κόμβο που εκκίνησε την ερώτηση διαστήματος. Στο σύστημα πραγματοποιούνται q ερωτήσεις διαστήματος. Επομένως, κάθε ένας από τους κόμβους του συστήματος συμμετέχει κατά μέσο όρο σε $\frac{q \times n \times s_k}{n} = q \times s_k$ ερωτήσεις διαστήματος εξετάζοντας τα δεδομένα που διαθέτει τοπικά.

Στο σύστημα του Mercury, μία ερώτηση διαστήματος προωθείται σε $O\left(\frac{1}{e}\log^2 n\right)$ κόμβους μέχρι να βρεθεί το κάτω όριο του ζητούμενου διαστήματος. Οι κόμβοι αυτοί δεν απαντούν στην ερώτηση παρέχοντας δεδομένα, απλώς την προωθούν στο δακτύλιο. Στη συνέχεια, η ερώτηση προωθείται σε ακόμη $N \times s$ κόμβους. Αν στο σύστημα του Mercury πραγματοποιούνται q ερωτήσεις διαστήματος, τότε εκτελούνται κατά μέσο όρο περίπου q/D ερωτήσεις διαστήματος σε κάθε έναν από τους δύο δακτυλίους του συστήματος, αφού τα δεδομένα που εισάγονται είναι D διαστάσεων. Επομένως, κάθε ένας από τους κόμβους του συστήματος συμμετέχει κατά μέσο όρο σε $\frac{q \times n \times s}{D \times n} = \frac{q \times s}{D}$ ερωτήσεις διαστήματος εξετάζοντας τα δεδομένα που διαθέτει τοπικά, το οποίο εξαρτάται από το πλήθος των δακτυλίων του συστήματος, άρα και του πλήθους των διαστάσεων των δεδομένων.

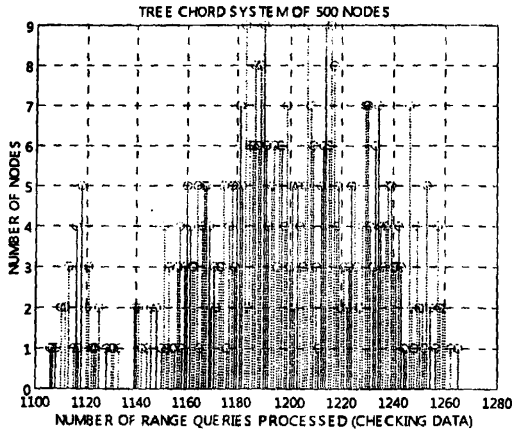
Η κατανομή των ερωτήσεων διαστήματος στους κόμβους κάθε συστήματος παρουσιάζεται στις επόμενες γραφικές παραστάσεις. Οι τιμές για κάθε σύστημα επαληθεύουν όσα παρουσιάστηκαν παραπάνω, αφού το μεγαλύτερο πλήθος κόμβων συμμετέχει σε όσες σχεδόν ερωτήσεις διαστήματος πρέπει.

Παρατηρούμε, λοιπόν, ότι κάθε κόμβος στο σύστημα του TreeChord ελέγχει περισσότερες φορές τα δεδομένα που διαθέτει τοπικά, συγκριτικά με τα άλλα συστήματα. Αυτό οφείλεται στη σταθερά σ, που αποτελεί τον αριθμό των εξεταζόμενων μονοπατιών στο σύστημα και

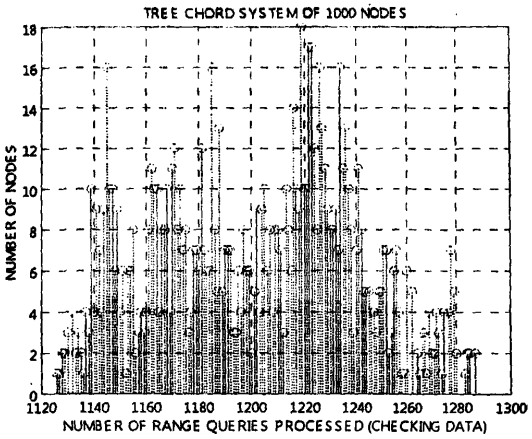
οδηγεί σε αύξηση του πλήθους των κόμβων που πρέπει να εξεταστούν για να επαληθευτεί αν διαθέτουν ή όχι δεδομένα που πρέπει να επιστραφούν στον κόμβο που ξεκινά μία ερώτηση διαστήματος. Η διαφορά από τα άλλα συστήματα καθορίζεται από την επιλεκτικότητα της ερώτησης, η οποία καθορίζει την τιμή της σταθεράς. Από την άλλη, παρατηρούμε ότι κάθε κόμβος στο σύστημα του Mercury ελέγχει πολύ λιγότερες φορές τα δεδομένα που διαθέτει τοπικά και, μάλιστα, σχεδόν μισές από την αντίστοιχη τιμή στο MAAN. Αυτό οφείλεται στο γεγονός πως οι κόμβοι του συστήματος του MAAN οργανώνονται σε έναν και μόνο δακτύλιο, ενώ οι κόμβοι του συστήματος του Mercury οργανώνονται σε τόσους ισομεγέθεις δακτυλίους, όσες είναι και οι διαστάσεις των εισαγόμενων δεδομένων του συστήματος, με αποτέλεσμα να μειώνεται το πλήθος των κόμβων που πρέπει να εξετάσουν τοπικά τα δεδομένα τους προκειμένου να διαπιστωθεί αν πρέπει να επιστραφεί κάποιο δεδομένο για την ικανοποίηση μίας ερώτησης διαστήματος.

Πιο συγκεκριμένα, στα πειράματα που πραγματοποιούμε, κάθε κόμβος στο σύστημα του TreeChord συμμετέχει στην επίλυση περίπου 60% και 110% περισσότερων ερωτήσεων διαστήματος από το σύστημα του MAAN και του Mercury, αντίστοιχα, και κάθε κόμβος στο σύστημα του MAAN συμμετέχει στην επίλυση περίπου 50% περισσότερων ερωτήσεων διαστήματος από το σύστημα του Mercury.

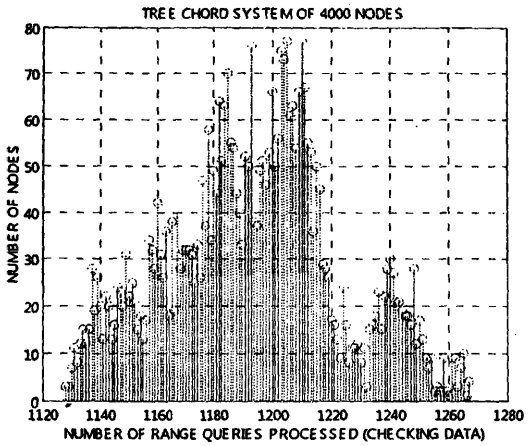
Μπορούμε, επομένως, να πούμε πως οι κόμβοι που συμμετέχουν στην επίλυση μίας ερώτησης διαστήματος είναι περισσότεροι στο σύστημα του TreeChord από ότι στα άλλα συστήματα. Ωστόσο, θα πρέπει να τονιστεί πως το πλήθος των ερωτήσεων διαστήματος στων οποίων την επίλυση συμμετέχει ένας κόμβος δεν είναι ενδεικτικό του πόσο γρήγορα μπορεί να επιλυθεί η ερώτηση, καθώς η επίλυση μίας ερώτησης γίνεται ταυτόχρονα από κόμβους που συμμετέχουν σε διαφορετικά μονοπάτια στο σύστημα του TreeChord. Επίσης, το πλήθος των ερωτήσεων διαστήματος στων οποίων την επίλυση συμμετέχει ένας κόμβος δεν είναι ο μόνος σημαντικός παράγοντας που καθορίζει το φόρτου του κόμβου λόγω της συμμετοχής του στην επίλυση ερωτήσεων διαστήματος. Εξίσου σημαντικό είναι και το πλήθος των δεδομένων με τα οποία απαντά κατά μέσο όρο ένας κόμβος σε μία ερώτηση διαστήματος. Η παράμετρος αυτή μελετάται στη συνέχεια και, όπως θα δούμε, το σύστημα του TreeChord υπερέχει σημαντικά έναντι των άλλων δύο συστημάτων.



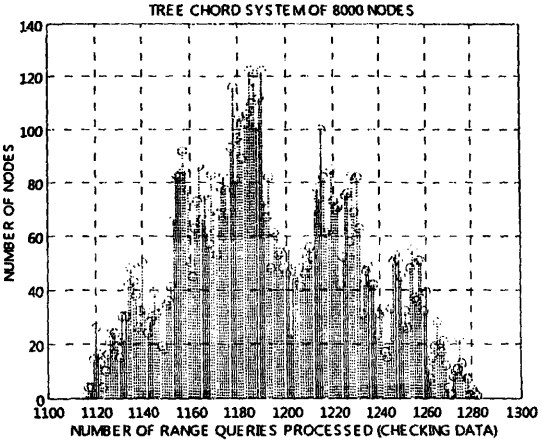
(α)



(β)

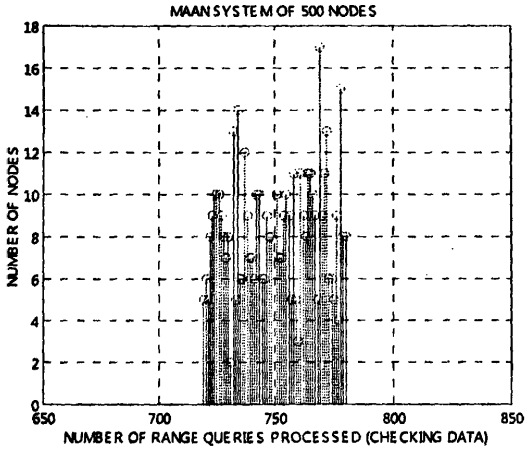


(γ)

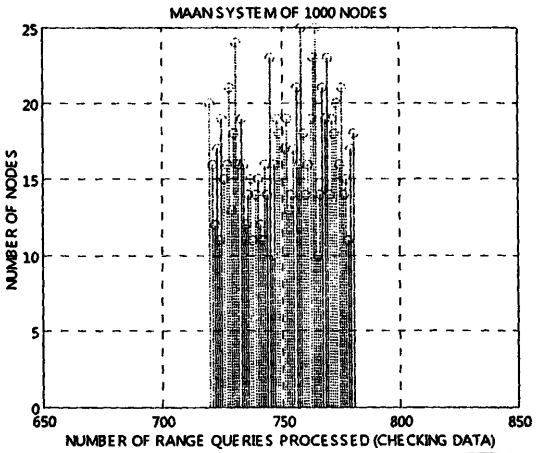


(δ)

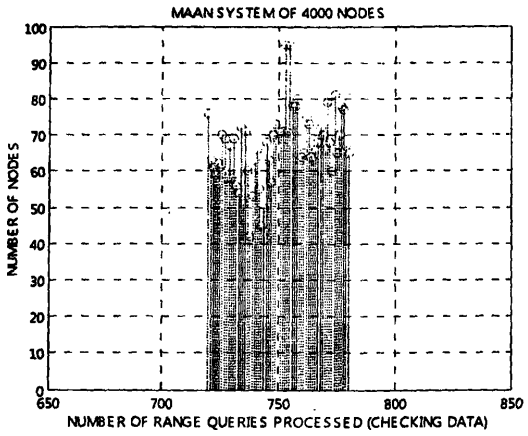
Σχήμα 10.12 Πλήθος Κόμβων που Συμμετέχουν στην Επίλυση ακριβώς i Ερωτήσεων Διαστήματος Ελέγχοντας Δεδομένα, $i \geq 0$, για το σύστημα του TreeChord.



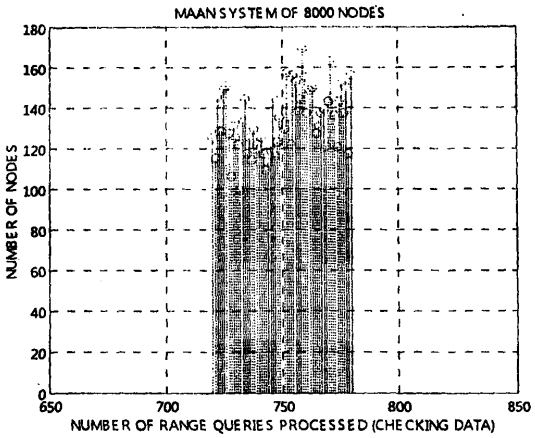
(α)



(β)

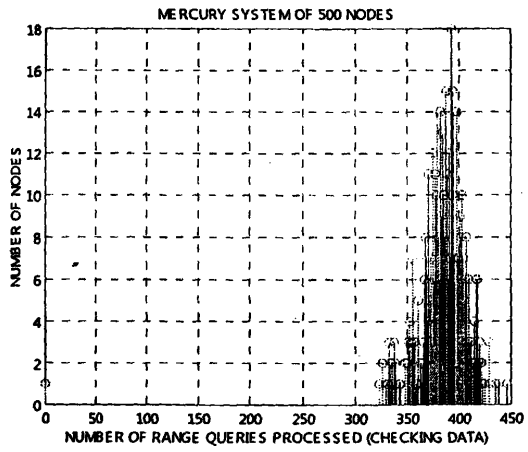


(γ)

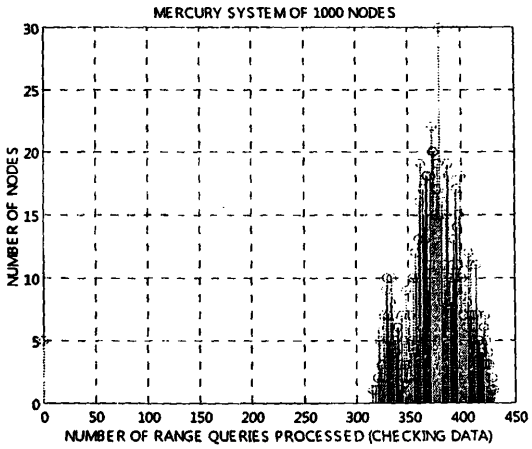


(δ)

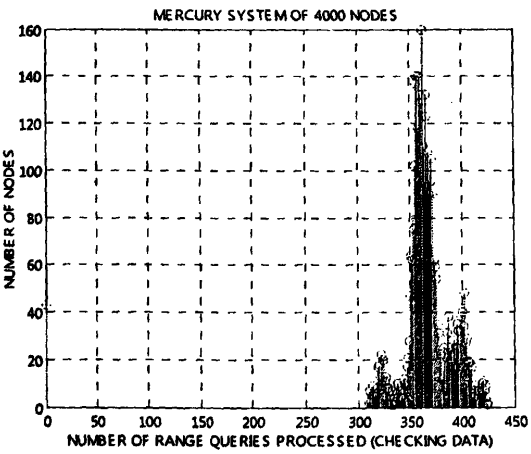
Σχήμα 10.13 Πλήθος Κόμβων που Συμμετέχουν στην Επίλυση ακριβώς i Ερωτήσεων Διαστήματος Ελέγχοντας Δεδομένα, $i \geq 0$, για το σύστημα του MAAN.



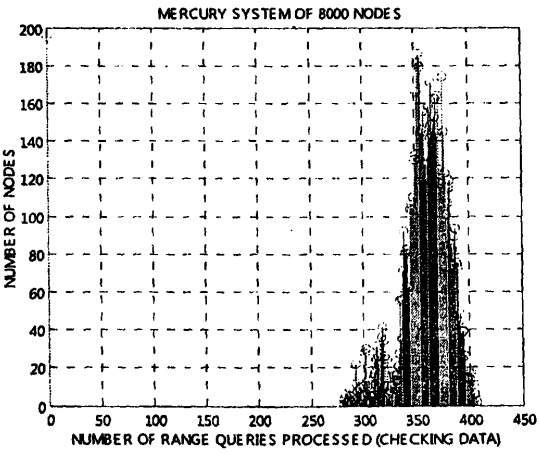
(α)



(β)



(γ)



(δ)

Σχήμα 10.14 Πλήθος Κόμβων που Συμμετέχουν στην Επίλυση ακριβώς i Ερωτήσεων Διαστήματος Ελέγχοντας Δεδομένα, $i \geq 0$, για το σύστημα του Mercury.

Στο Παράρτημα Β μπορούμε να παρατηρήσουμε τα ίδια αποτελέσματα με μεγαλύτερη λεπτομέρεια, αφού στις γραφικές παραστάσεις που παρουσιάζονται απεικονίζεται το ακριβές πλήθος ερωτήσεων διαστήματος στις οποίες συμμετέχει κάθε κόμβος ενός συστήματος.

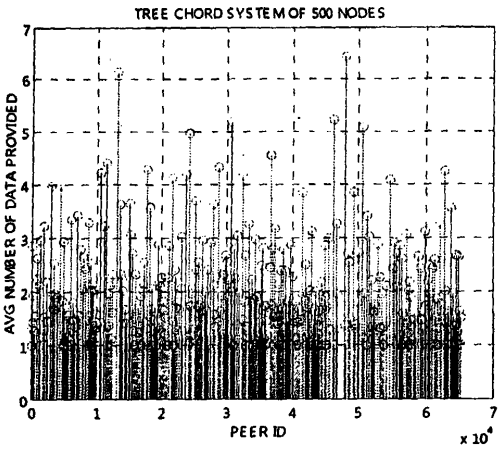
10.2.2. Απεικόνιση Πλήθους Δεδομένων με τα Οποία Απαντά Κάθε Κόμβος σε Μία Ερώτηση Διαστήματος Κατά Μέσο Όρο

Επαναλαμβάνουμε τα πειράματα της προηγούμενης ενότητας. Στις πρώτες γραφικές παραστάσεις απεικονίζεται το πλήθος των δεδομένων με τα οποία απαντά ένας κόμβος σε μία ερώτηση διαστήματος, κατά μέσο όρο (Σχήμα 10.15 – Σχήμα 10.18). Ακολουθούν γραφικές παραστάσεις στις οποίες απεικονίζεται το ποσοστό των κόμβων κάθε συστήματος που επιστρέφουν λιγότερα από 3 δεδομένα κατά μέσο όρο, το ποσοστό των κόμβων κάθε συστήματος που επιστρέφουν 3 ή 4 δεδομένα κατά μέσο όρο καθώς, επίσης, και το ποσοστό των κόμβων κάθε συστήματος που επιστρέφουν περισσότερα από 4 δεδομένα κατά μέσο όρο, στα συστήματα που μελετήθηκαν (Σχήμα 10.19).

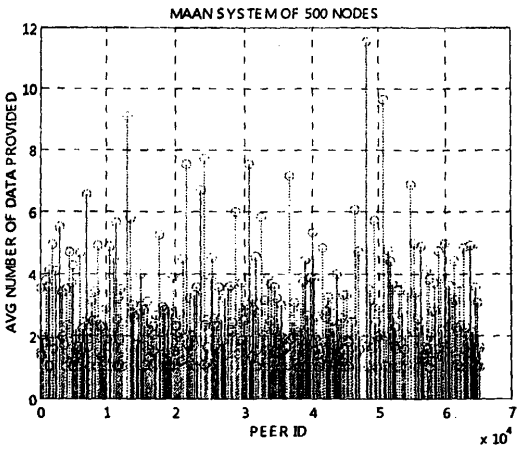
Παρατηρούμε, λοιπόν, ότι στο σύστημα του TreeChord με 500 κόμβους, οι περισσότεροι κόμβοι επιστρέφουν κατά μέσο όρο το πολύ 2 δεδομένα, αρκετοί 3 δεδομένα, πολύ λίγοι κόμβοι περισσότερα από 3 δεδομένα, ενώ μόλις 2 κόμβοι επιστρέφουν περισσότερα από 6 δεδομένα κατά μέσο όρο. Στα αντίστοιχα συστήματα του MAAN και του Mercury, όμως, το πλήθος των κόμβων που επιστρέφουν περισσότερα από 3 δεδομένα κατά μέσο όρο είναι περισσότεροι, ενώ υπάρχουν και αρκετοί κόμβοι που επιστρέφουν πολλά περισσότερα από 3 δεδομένα. Μάλιστα, στα συστήματα του MAAN και του Mercury υπάρχουν κόμβοι που επιστρέφουν ακόμη και 12 δεδομένα ή 18 δεδομένα, αντίστοιχα. Παρόμοια είναι τα αποτελέσματα στα συστήματα των 1000, 4000 και 8000 κόμβων.

Παρατηρούμε, λοιπόν, ότι παρόλο που στο σύστημα του TreeChord εξετάζονται αρκετοί περισσότεροι κόμβοι για να βρεθούν τα δεδομένα που ικανοποιούν μία ερώτηση διαστήματος, οι περισσότεροι κόμβοι απαντούν με λιγότερα δεδομένα κατά μέσο όρο σε κάθε ερώτηση διαστήματος. Ενώ, λοιπόν, στο σύστημα του TreeChord απαιτούνται περισσότερα μηνύματα για την εύρεση των δεδομένων που πρέπει να επιστραφούν, αποστέλλονται λιγότερα μηνύματα από κάθε κόμβο για να επιστραφούν τα κατάλληλα δεδομένα στον κόμβο που εκκινεί μία ερώτηση διαστήματος. Με άλλα λόγια, λοιπόν, μπορούμε να πούμε ότι στο

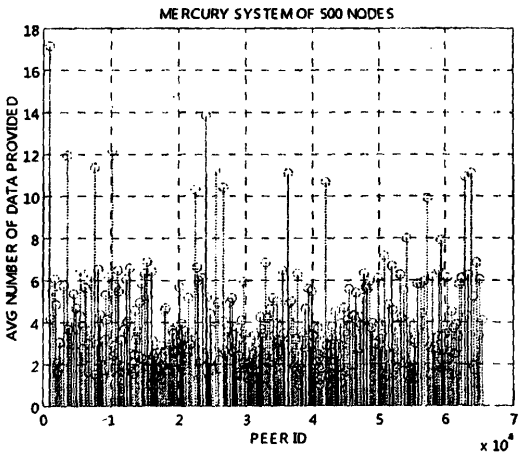
σύστημα του TreeChord κοστίζει περισσότερο η εύρεση των δεδομένων που ικανοποιούν τις συνθήκες μίας ερώτησης διαστήματος από ότι στα άλλα συστήματα, αλλά κοστίζει λιγότερο σε κάθε κόμβο η επιστροφή των δεδομένων.



(α)

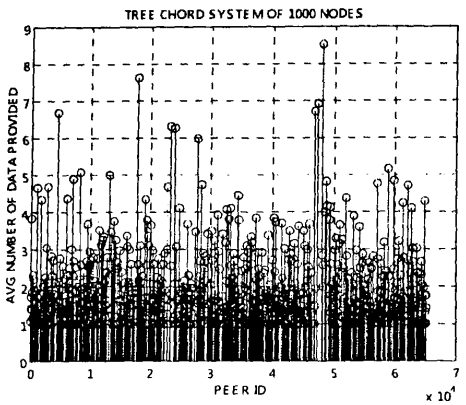


(β)

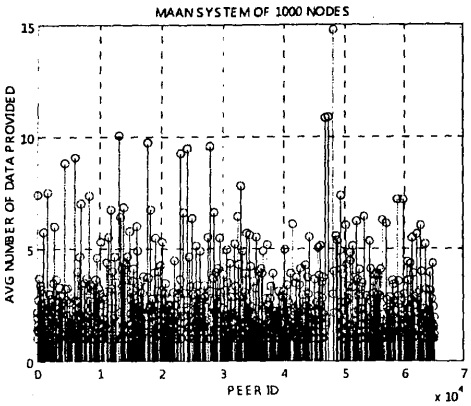


(γ)

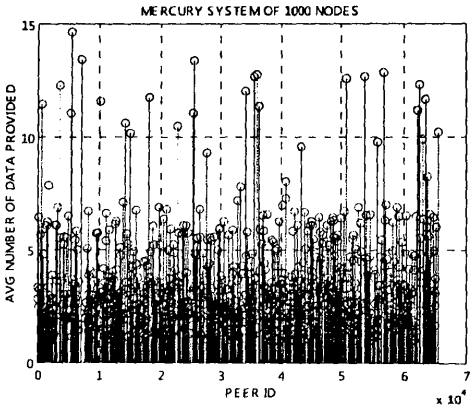
Σχήμα 10.15 Πλήθος Δεδομένων με τα Οποία Απαντά Κάθε Κόμβος σε Μία Ερώτηση Διαστήματος Κατά Μέσο Όρο σε Συστήματα των 500 Κόμβων.



(α)

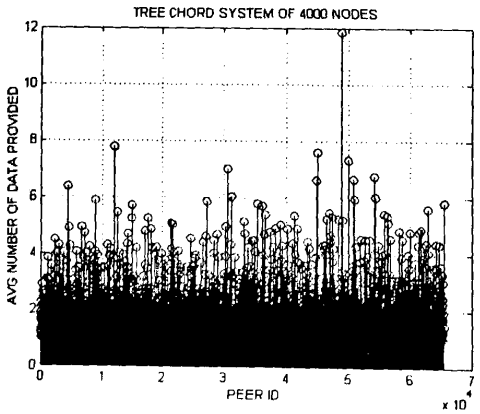


(β)

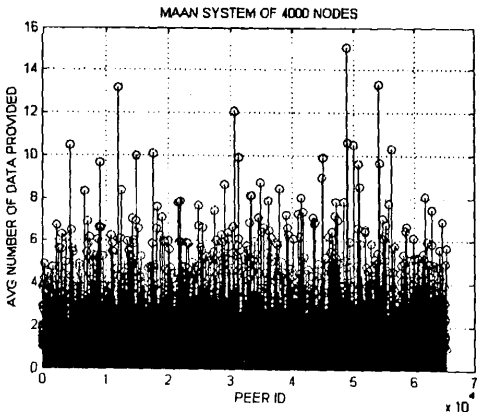


(γ)

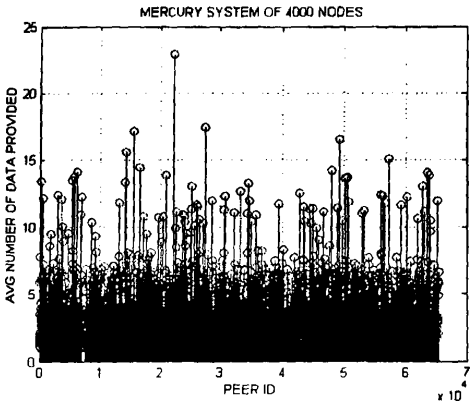
Σχήμα 10.16 Πλήθος Δεδομένων με τα Οποία Απαντά Κάθε Κόμβος σε Μία Ερώτηση Διαστήματος Κατά Μέσο Όρο σε Συστήματα των 1000 Κόμβων.



(α)

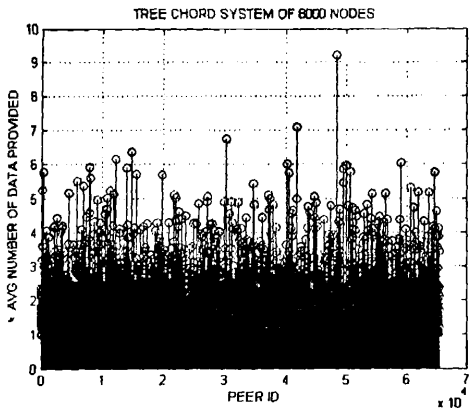


(β)

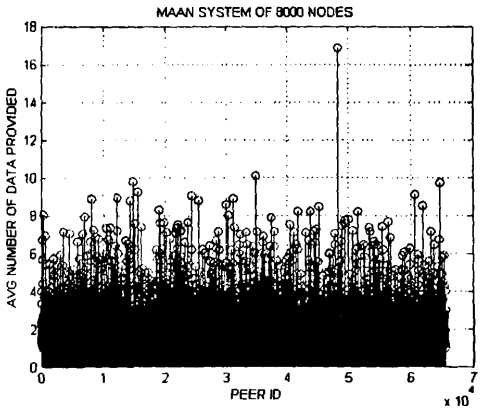


(γ)

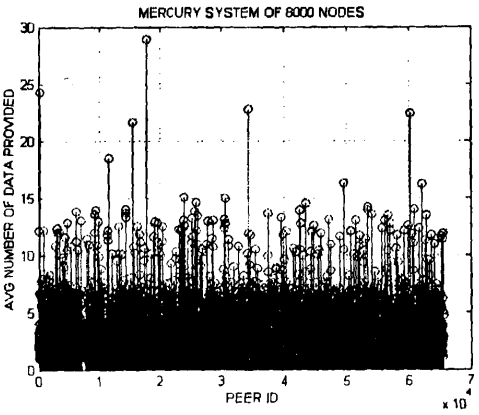
Σχήμα 10.17 Πλήθος Δεδομένων με τα Οποία Απαντά Κάθε Κόμβος σε Μία Ερώτηση Διαστήματος Κατά Μέσο Όρο σε Συστήματα των 4000 Κόμβων.



(α)



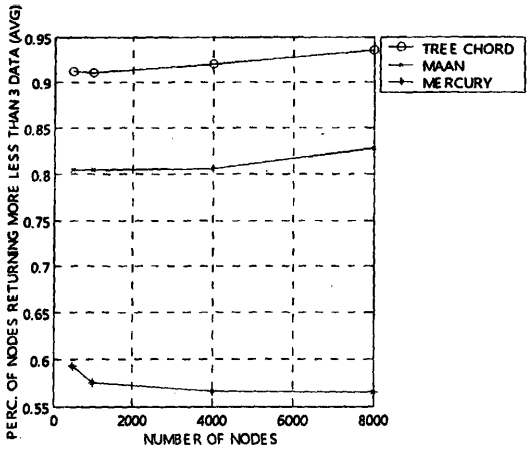
(β)



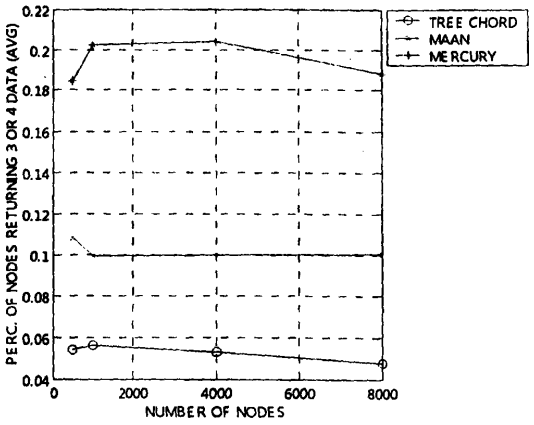
(γ)

Σχήμα 10.18 Πλήθος Δεδομένων με τα Οποία Απαντά Κάθε Κόμβος σε Μία Ερώτηση Διαστήματος Κατά Μέσο Όρο σε Συστήματα των 8000 Κόμβων.

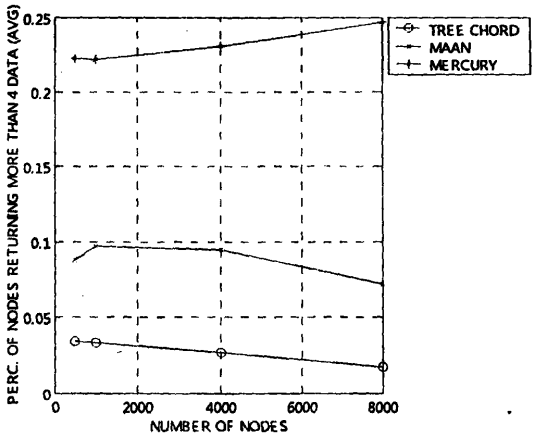
Στις ακόλουθες γραφικές παραστάσεις (Σχήμα 10.19) απεικονίζεται το ποσοστό των κόμβων κάθε συστήματος που επιστρέφουν λιγότερα από 3 δεδομένα κατά μέσο όρο, 3 ή 4 δεδομένα κατά μέσο όρο και περισσότερα από 4 δεδομένα κατά μέσο όρο, στα συστήματα που μελετήθηκαν.



(α)



(β)



(γ)

Σχήμα 10.19 Ποσοστό Κόμβων που Επιστρέφουν Λιγότερα από 3 Δεδομένα Κατά Μέσο Όρο, 3 ή 4 Δεδομένα Κατά Μέσο Όρο και Περισσότερα από 4 Δεδομένα Κατά Μέσο Όρο.

Στην πρώτη περίπτωση, παρατηρούμε ότι ενώ στο σύστημα του TreeChord το ποσοστό των κόμβων που επιστρέφουν λιγότερα από 3 δεδομένα κατά μέσο όρο είναι ιδιαίτερα υψηλό και, μάλιστα, μεγαλύτερο από 90%, στο σύστημα του MAAN μειώνεται περίπου στο 80% και στο σύστημα του Mercury είναι χαμηλότερο από 60%.

Από την άλλη, όταν εξετάζουμε το ποσοστό των κόμβων που επιστρέφουν 3 ή 4 δεδομένα κατά μέσο όρο, παρατηρούμε ότι στο σύστημα του TreeChord είναι ιδιαίτερα χαμηλό και, μάλιστα, περίπου 5%. Αντίστοιχα, στο MAAN το ποσοστό αυξάνεται σε περίπου 10%, ενώ στο Mercury αγγίζει το 20%.

Τέλος, όταν εξετάζουμε το ποσοστό των κόμβων που επιστρέφουν περισσότερα από 4 δεδομένα κατά μέσο όρο, παρατηρούμε ότι στο σύστημα του TreeChord το ποσοστό μειώνεται ακόμη περισσότερο και γίνεται μικρότερο του 4% ή και ακόμη χαμηλότερο. Αντίστοιχα, στο MAAN το ποσοστό κυμαίνεται από 7% έως περίπου 10%, ενώ στο Mercury είναι κατά πολύ μεγαλύτερο, ξεπερνώντας το 20%.

Μπορούμε, λοιπόν, να πούμε ότι αν και οι κόμβοι στο σύστημα του TreeChord επιβαρύνονται περισσότερο κατά την αναζήτηση των δεδομένων προωθώντας μηνύματα για να βρεθούν τα δεδομένα που πρέπει να επιστραφούν, ωστόσο επιβαρύνονται πολύ λιγότερο με το κόστος της επιστροφής των δεδομένων που ικανοποιούν μία ερώτηση διαστήματος, αφού κάθε κόμβος αποστέλλει πολύ λιγότερα δεδομένα.

10.3. Σύγκριση Αλγορίθμων Επίλυσης Ερωτήσεων Διαστήματος

Στα πειράματα που ακολουθούν, εξετάζουμε τους αλγορίθμους επίλυσης ερωτήσεων διαστήματος που έχουν υλοποιηθεί στα συστήματα του TreeChord, του MAAN και του Mercury.

Στη συνέχεια, υπενθυμίζουμε πώς λειτουργούν οι αλγόριθμοι επίλυσης ερωτήσεων διαστήματος στα τρία συστήματα, που περιγράφηκαν παραπάνω, και, κατόπιν, παρουσιάζουμε τις πειραματικές μετρήσεις που πραγματοποιήσαμε.

Κατά την επίλυση μίας ερώτησης διαστήματος q στο σύστημα του TreeChord δύο διαστάσεων, ο κόμβος που εκκινεί την ερώτηση υπολογίζει αρχικά το δένδρο δεδομένων βάσει της τιμής του κάτω ορίου της ερώτησης q στη διάσταση η οποία έχει χρησιμοποιηθεί στο σύστημα για την εισαγωγή των δεδομένων στο σύστημα, δηλαδή τη x-διάσταση. Στη συνέχεια, βρίσκει ποιο από τα φύλλα του δένδρου είναι υπεύθυνο για την τιμή του κάτω ορίου της ερώτησης q στην άλλη διάσταση, δηλαδή την y-διάσταση, και δρομολογεί την

ερώτηση στο διάδοχο κόμβο του φύλλου αυτού, έστω sn, χρησιμοποιώντας τον αλγόριθμο δρομολόγησης του Chord.

Όταν ο κόμβος sn λάβει την ερώτηση q, υπολογίζει εκ νέου το δένδρο δεδομένων βάσει της τιμής του κάτω ορίου της ερώτησης q στη x-διάσταση, έστω πρώτο, αλλά και το δένδρο δεδομένων βάσει της τιμής του άνω ορίου της ερώτησης q στη x-διάσταση, έστω τελευταίο. Κατόπιν, βρίσκει ποια από τα φύλλα του πρώτου δένδρου επικαλύπτουν το ζητούμενο εύρος στην y-διάσταση. Κάθε ένα από τα φύλλα αυτά αποτελεί την αρχή ενός μονοπατιού που αποτελείται από κόμβους που πρέπει να εξεταστούν για να βρεθούν δεδομένα που ανήκουν στο ζητούμενο εύρος. Τα μονοπάτια εξετάζονται βάσει των φύλλων, από το πρώτο έως το τελευταίο. Η προώθηση της ερώτησης πραγματοποιείται μέσω διαδόχων κόμβων και σταματά όταν η ερώτηση αποσταλεί στο διάδοχο κόμβο του αντίστοιχου φύλλου του τελευταίου δένδρου. Η μετάβαση στην αρχή του επόμενου μονοπατιού προς εξέταση γίνεται μέσω του κατάλληλου δείκτη από τις διατηρούμενες δομές δρομολόγησης του Chord στον κόμβο sn, δηλαδή σε ένα μόνο βήμα. Ο τρέχων κόμβος sn, στον οποίο έχει καταλήξει η ερώτηση q αποτελεί την αρχή του πρώτου μονοπατιού. Σημειώνεται ότι τα μονοπάτια εξετάζονται παράλληλα.

Κάθε κόμβος που είναι υπεύθυνος για εύρος τιμών που επικαλύπτει το εύρος τιμών που ορίζεται στην ερώτηση διαστήματος και λαμβάνει την ερώτηση ελέγχει τα δεδομένα που διατηρεί τοπικά. Ένα δεδομένο επιστρέφεται αν ικανοποιούνται όλες οι συνθήκες, για κάθε μία από τις διαστάσεις που ορίζονται στην ερώτηση διαστήματος. Επομένως, ένας κόμβος που εκκινεί μία ερώτηση διαστήματος λαμβάνει μόνο σωστά δεδομένα, δηλαδή δεδομένα που ικανοποιούν όλες τις συνθήκες.

Για τη δρομολόγηση, λοιπόν, μίας ερώτησης διαστήματος, στον πρώτο κόμβο που πιθανώς διαθέτει κατάλληλα δεδομένα για την ικανοποίησή της, απαιτούνται $O(\log N)$ μηνύματα (και βήματα), όπως και στο Chord. Επίσης, η μετάβαση στην αρχή κάθε μονοπατιού (εκτός από το πρώτο, στο οποίο καταλήγουμε μέσω του αλγορίθμου δρομολόγησης του Chord) απαιτεί ένα επιπλέον μήνυμα (και βήμα). Τέλος, η εξέταση κάθε κόμβου ενός μονοπατιού απαιτεί $N \times s$ μηνύματα (και βήματα), καθώς η κατανομή των κόμβων στο δακτύλιο γίνεται όπως και στο σύστημα του Chord και, επομένως, είναι ομοιόμορφη, ενώ το πλήθος των μονοπατιών που πρέπει να εξεταστούν είναι μία σταθερά, έστω σ, η οποία, όπως θα δούμε

και στη συνέχεια, εξαρτάται από την επιλεκτικότητα της ερώτησης στη δεύτερη διάσταση. Επομένως, η εύρεση όλων των δεδομένων που ικανοποιούν κάθε μία από τις συνθήκες που ορίζονται στην ερώτηση διαστήματος q απαιτεί $\sigma \times N \times s + \log N$ μηνύματα (και μια σταθερά για τα βήματα στην αρχή κάθε μονοπατιού, που μπορεί να είναι το πολύ όσα και τα φύλλα του δένδρου μείον 1), καθώς όταν μετρούμε το κόστος μιας ερώτησης διαστήματος βάσει του απαιτούμενου πλήθους μηνυμάτων, υπολογίζουμε το άθροισμα των μηνυμάτων που αποστέλλονται για την εύρεση των δεδομένων που θα επιστραφούν από κάθε μονοπάτι. Όταν, όμως, μετρούμε το κόστος μιας ερώτησης βάσει του πλήθους των βημάτων που απαιτούνται για την εύρεση των δεδομένων, υπολογίζουμε το πλήθος των μηνυμάτων που απαιτούνται για την εύρεση των δεδομένων σε κάθε μονοπάτι και, έπειτα, βρίσκουμε το μέγιστο αυτών των τιμών, αφού τα μονοπάτια εξετάζονται παράλληλα.

Κατά την επίλυση μίας ερώτησης διαστήματος q στο σύστημα του MAAN, μπορούν να εφαρμοστούν δύο αλγόριθμοι. Όπως προαναφέρθηκε, όταν εφαρμόζεται ο πρώτος αλγόριθμος (*Iterative Query Resolution*) για την επίλυση μίας ερώτησης διαστήματος q, πραγματοποιούνται M ερωτήσεις, μία για κάθε διάσταση των δεδομένων. Σε κάθε μία από αυτές τις ερωτήσεις ελέγχεται μόνο η συνθήκη για την αντίστοιχη διάσταση, έτσι ώστε να επιστραφούν τα κατάλληλα δεδομένα. Όταν τα δεδομένα καταλήξουν στον τελικό αποδέκτη, ελέγχεται αν ικανοποιούνται και οι συνθήκες για τις υπόλοιπες διαστάσεις.

Όταν μετρούμε το κόστος μιας ερώτησης διαστήματος βάσει του απαιτούμενου πλήθους μηνυμάτων, υπολογίζουμε το άθροισμα των μηνυμάτων που αποστέλλονται για την εύρεση των δεδομένων που θα επιστραφούν σε κάθε διάσταση. Όταν, όμως, μετρούμε το κόστος μιας ερώτησης βάσει του πλήθους των βημάτων που απαιτούνται για την εύρεση των δεδομένων, υπολογίζουμε το πλήθος των μηνυμάτων που απαιτούνται για την εύρεση των δεδομένων σε κάθε διάσταση και, έπειτα, βρίσκουμε το μέγιστο αυτών των τιμών, αφού οι ερωτήσεις πραγματοποιούνται παράλληλα σε κάθε διάσταση. Κάθε ερώτηση διαστήματος, λοιπόν, που πραγματοποιείται στο σύστημα απαιτεί $O(\log N + K_d)$ ή $O(\log N + N \times s_d)$ βήματα, όπου $d = \arg\max_{1 \le i \le M}(O(\log N + N \times s_i))$, και $O\left(\sum_{i=1}^{M}(\log N + K_i)\right)$ μηνύματα, όπου M το πλήθος των διαστάσεων των δεδομένων του συστήματος και K_i το πλήθος των κόμβων του συστήματος που ανήκουν στο εύρος που ορίζεται στην ερώτηση για τη διάσταση i. Υπενθυμίζουμε ότι

στο πλήθος αυτό δεν έχει συμπεριληφθεί και το κόστος που απαιτείται για την επιστροφή των ίδιων των δεδομένων στον κόμβο που εκκίνησε την ερώτηση.

Ακόμη, όταν εφαρμόζεται ο δεύτερος αλγόριθμος, πραγματοποιείται μία μόνο ερώτηση σε κάποια από τις διαστάσεις των δεδομένων του συστήματος, έστω a_k. Επιλέγεται η διάσταση για την οποία η επιλεκτικότητα είναι ελάχιστη. Το πλήθος, όμως, των δεδομένων που επιστρέφονται εξαρτάται από την προσέγγιση που ακολουθείται από τους κόμβους που λαμβάνουν την ερώτηση για να αποφασίσουν ποια από τα δεδομένα που διαθέτουν τοπικά πρέπει να επιστρέψουν. Για την επιλογή των δεδομένων που πρέπει να επιστραφούν ακολουθούνται δύο προσεγγίσεις. Όταν ακολουθείται η πρώτη προσέγγιση, ένα δεδομένο επιστρέφεται αν ικανοποιείται η συνθήκη για τη συγκεκριμένη διάσταση (*Single Attribute Dominated Query Resolution Redundant Data*). Στον τελικό αποδέκτη ελέγχεται αν ισχύουν και οι συνθήκες για τις υπόλοιπες διαστάσεις και, επομένως, ο κόμβος μπορεί να λάβει και άχρηστα δεδομένα, δηλαδή δεδομένα που δεν ικανοποιούν όλες τις συνθήκες που ορίζονται στην ερώτηση διαστήματος και, άρα, δεν αποτελούν μέρος της τελικής απάντησης. Όταν, όμως, ακολουθείται η δεύτερη προσέγγιση, ένα δεδομένο επιστρέφεται αν ικανοποιεί όλες τις συνθήκες για κάθε διάσταση (*Single Attribute Dominated Query Resolution*), με αποτέλεσμα να επιστρέφονται στον κόμβο που εκκίνησε την ερώτηση μόνο σωστά δεδομένα, δηλαδή δεδομένα που ικανοποιούν όλες τις συνθήκες.

Όταν, λοιπόν, εφαρμόζεται ο δεύτερος αλγόριθμος, απαιτούνται $O(\log N + K_k)$ ή $O(\log N + N \times s_k)$ μηνύματα και $O(\log N + K_k)$ ή $O(\log N + N \times s_k)$ βήματα για να καταλήξει μία ερώτηση διαστήματος σε όλους τους κόμβους που πρέπει να επιστρέψουν δεδομένα στον κόμβο n, όπου K_k το πλήθος των κόμβων που ανήκουν στο εύρος που ορίζεται για το κυρίαρχο χαρακτηριστικό a_k και s_k η επιλεκτικότητα της ερώτησης διαστήματος για αυτό, όποια προσέγγιση κι αν ακολουθηθεί. Αν, επιπλέον, επιλέξουμε το χαρακτηριστικό a_k να είναι εκείνο για το οποίο η επιλεκτικότητα που ορίζεται στην ερώτηση είναι ελάχιστη και αν $s_{\min}$ η επιλεκτικότητα αυτή, απαιτούνται $O(\log N + N \times s_{\min})$ μηνύματα και $O(\log N + N \times s_{\min})$ βήματα.

Ενώ, όμως, το πλήθος των βημάτων και των μηνυμάτων που απαιτούνται για να βρεθούν τα κατάλληλα δεδομένα, αλλά όχι και να επιστραφούν στον κόμβο που εκκίνησε την ερώτηση διαστήματος, παραμένει ίδιο όποια προσέγγιση κι αν ακολουθηθεί, το πλήθος των δεδομένων που επιστρέφονται από κάθε αλγόριθμο, επομένως, και το πλήθος των μηνυμάτων που απαιτούνται για την επιστροφή των δεδομένων διαφέρει. Πιο συγκεκριμένα, όταν ακολουθείται η πρώτη προσέγγιση, στον κόμβο n επιστρέφονται πιθανόν και άχρηστα δεδομένα, δηλαδή δεδομένα που δεν ικανοποιούν όλες τις συνθήκες της ερώτησης διαστήματος q, με αποτέλεσμα να αυξάνεται το κόστος επικοινωνίας μεταξύ των κόμβων για την επιστροφή των δεδομένων. Όταν, όμως, ακολουθείται η δεύτερη προσέγγιση, στον κόμβο n επιστρέφονται μόνο σωστά δεδομένα, δηλαδή δεδομένα που ικανοποιούν κάθε μία από τις συνθήκες της ερώτησης διαστήματος q.

Κατά την επίλυση μίας ερώτησης διαστήματος q στο σύστημα του Mercury, ο κόμβος που εκκινεί την ερώτηση την αποστέλλει σε έναν μόνο από τους δακτυλίους του συστήματος μέσω των δεικτών που διατηρεί προς κάθε δακτύλιο. Επιλέγεται ο δακτύλιος που αντιστοιχεί στη διάσταση για την οποία η επιλεκτικότητα της ερώτησης διαστήματος είναι ελάχιστη. Το βήμα αυτό απαιτεί ένα μόλις μήνυμα (και βήμα).

Στη συνέχεια, η ερώτηση προωθείται εντός του δακτυλίου μέσω διαδόχων κόμβων ή μακρινών ακμών, μέχρι να βρεθεί ο κόμβος στου οποίου το εύρος τιμών ανήκει το κάτω όριο της ερώτησης. Αποδεικνύεται ότι το βήμα αυτό απαιτεί $O\left(\frac{1}{e}\log^2 n\right)$ μηνύματα (και βήματα).

Κατόπιν, από τον κόμβο αυτό, η ερώτηση προωθείται σε διαδοχικούς κόμβους του δακτυλίου, μέχρι να βρεθεί ο κόμβος στου οποίου το εύρος τιμών ανήκει το άνω όριο της ερώτησης.

Κάθε κόμβος που είναι υπεύθυνος για εύρος τιμών που καλύπτει μέρος της ερώτησης διαστήματος ελέγχει τα δεδομένα που διατηρεί τοπικά. Ένα δεδομένο επιστρέφεται αν ικανοποιούνται όλες οι συνθήκες για κάθε μία από τις διαστάσεις που ορίζονται στην ερώτηση διαστήματος. Επομένως, ένας κόμβος που εκκινεί μία ερώτηση διαστήματος λαμβάνει μόνο σωστά δεδομένα, δηλαδή δεδομένα που ικανοποιούν όλες τις συνθήκες.

Για τη δρομολόγηση, λοιπόν, μίας ερώτησης διαστήματος, στον πρώτο κόμβο που πιθανώς κατέχει κατάλληλα δεδομένα για την ικανοποίησή της, απαιτούνται $O\left(\frac{1}{e}\log^2 N\right)$ μηνύματα (και βήματα) συνολικά, όπου e το πλήθος των μακριών ακμών που διατηρούν οι κόμβοι του συστήματος. Αν, επίσης, a_k η διάσταση που αντιστοιχεί στο δακτύλιο όπου δρομολογείται η ερώτηση διαστήματος, N_k το πλήθος των κόμβων του αντίστοιχου δακτυλίου και s_k η επιλεκτικότητα της ερώτησης διαστήματος για αυτή τη διάσταση, απαιτούνται επιπλέον $N_k \times s_k$ μηνύματα (και βήματα) για να βρεθούν και όλοι οι υπόλοιποι κόμβοι που πιθανώς κατέχουν δεδομένα που ικανοποιούν την ερώτηση διαστήματος. Σημειώνεται ότι ο δακτύλιος στον οποίο αποστέλλεται η ερώτηση επιλέγεται έτσι ώστε το αντίστοιχο χαρακτηριστικό a_k να είναι εκείνο για το οποίο η επιλεκτικότητα που ορίζεται στην ερώτηση είναι ελάχιστη.

Στη συνέχεια, παρουσιάζονται οι πειραματικές μετρήσεις που πραγματοποιήσαμε.

10.3.1. Κόστος Αλγορίθμων Επίλυσης Ερωτήσεων Διαστήματος σε Συστήματα με Συγκεκριμένο Πλήθος Κόμβων

Στα πειράματα της κατηγορίας αυτής εξετάζουμε ποιο είναι το κόστος κάθε αλγορίθμου επίλυσης ερωτήσεων διαστήματος υπολογίζοντας το μέσο πλήθος βημάτων ή μηνυμάτων που απαιτείται για την εύρεση όλων των δεδομένων που πρέπει να επιστραφούν στον κόμβο που εκκινεί μία ερώτηση διαστήματος, όχι, όμως, και για την επιστροφή τους σε αυτόν.

Μελετούμε τα συστήματα που περιγράφηκαν στις προηγούμενες ενότητες. Σε κάθε ένα από αυτά πραγματοποιούμε 5000, 10000, 40000 και 80000 ερωτήσεις διαστήματος και υπολογίζουμε το απαιτούμενο μέσο κόστος. Οι ερωτήσεις δημιουργούνται με τον τρόπο που περιγράφηκε. Σημειώνεται ότι η επιλεκτικότητα των ερωτήσεων είναι 15% για κάθε διάσταση. Πειράματα όπου μεταβάλλεται το πλήθος των διαστάσεων των δεδομένων ή η επιλεκτικότητα των ερωτήσεων παρουσιάζονται σε επόμενες ενότητες. Τα αποτελέσματα φαίνονται στις επόμενες γραφικές παραστάσεις.

Στις επόμενες γραφικές παραστάσεις (Σχήμα 10.20) συγκρίνουμε το κόστος των αλγορίθμων επίλυσης ερωτήσεων διαστήματος μετρώντας τον αριθμό βημάτων που απαιτούνται για την

εύρεση των δεδομένων που πρέπει να επιστραφούν, όχι, όμως, και για την επιστροφή τους στον κόμβο που εκκινεί μία ερώτηση διαστήματος.

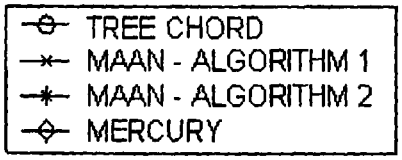
Παρατηρούμε ότι σε κάθε περίπτωση το κόστος του αλγορίθμου που εφαρμόζεται στο TreeChord είναι σχεδόν ίδιο με το κόστος και των δύο αλγορίθμων που εφαρμόζονται στο MAAN. Αυτό εξηγείται από το γεγονός ότι στο σύστημα του TreeChord για κάθε ερώτηση διαστήματος υπολογίζουμε το πλήθος των βημάτων που απαιτούνται για να βρεθούν τα κατάλληλα δεδομένα στο μακρύτερο μονοπάτι, αλλά μόνο σε ένα από όσα εξερευνηθούν (αφού όλα εξερευνώνται παράλληλα), ενώ και οι κόμβοι του συστήματος κατανέμονται ομοιόμορφα στο δακτύλιο και, επιπλέον, η εξερεύνηση κάθε μονοπατιού απαιτεί $N \times s + 1$ βήματα (το ένα βήμα αφορά στη μετάβαση με ένα βήμα στην αρχή του μονοπατιού). Επίσης, στο σύστημα του MAAN παρατηρούμε ότι σε κάθε περίπτωση το μέσο κόστος και των δύο αλγορίθμων που εφαρμόζονται είναι σχεδόν ίδιο και, μάλιστα, ίσο με $O(\log N + N \times s_k)$. Αυτό εξηγείται από το γεγονός πως για κάθε ερώτηση διαστήματος υπολογίζουμε το πλήθος των βημάτων που απαιτούνται για να βρεθούν τα κατάλληλα δεδομένα σε μία μόνο διάσταση και, επιπλέον, η επιλεκτικότητα των ερωτήσεων είναι ίδια για κάθε διάσταση, ενώ οι κόμβοι του συστήματος κατανέμονται ομοιόμορφα στο δακτύλιο.

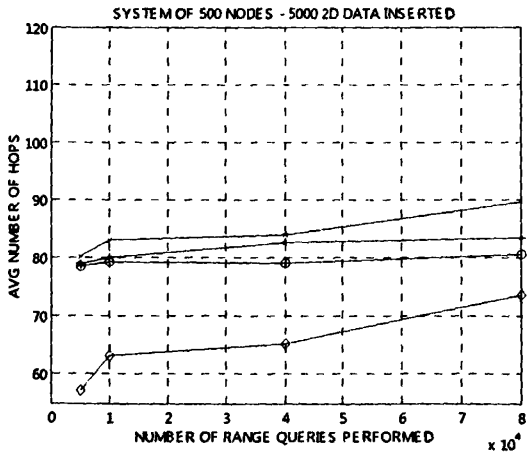
Αντιλαμβανόμαστε, λοιπόν, γιατί το μέσο κόστος του αλγορίθμου επίλυσης ερωτήσεων διαστήματος που εφαρμόζεται στο TreeChord και το αντίστοιχο μέσο κόστος των δύο αλγορίθμων που εφαρμόζονται στο MAAN είναι σχεδόν ίδιο. Επίσης, παρατηρούμε ότι το κόστος είναι ανεξάρτητο από το πλήθος των διαστάσεων των δεδομένων του συστήματος, καθώς σε κάθε περίπτωση μετριέται το πλήθος των βημάτων που απαιτούνται σε ένα μόνο μονοπάτι ή μία μόνο διάσταση και δεν αθροίζονται τα αντίστοιχα πλήθη βημάτων που απαιτούνται για να βρεθούν τα δεδομένα που πρέπει να επιστραφούν για κάθε ένα από τα μονοπάτια ή κάθε μία από τις διαστάσεις στην περίπτωση που εφαρμόζεται ο πρώτος αλγόριθμος. Αν η επιλεκτικότητα των ερωτήσεων ήταν διαφορετική για κάθε διάσταση, τότε πιθανότατα το κόστος στο TreeChord, αλλά και του δεύτερου αλγορίθμου στο MAAN θα ήταν ακόμη μικρότερο. Σημειώνεται ότι στο MAAN το κόστος και των δύο αλγορίθμων ταυτίζεται αν το πλήθος των βημάτων που απαιτούνται για την ικανοποίηση της ερώτησης στη διάσταση με τη μικρότερη επιλεκτικότητα είναι μεγαλύτερο από τα αντίστοιχα πλήθη βημάτων που απαιτούνται για την ικανοποίηση της ερώτησης σε κάθε άλλη διάσταση, αλλιώς το κόστος του πρώτου αλγορίθμου είναι μεγαλύτερο. Το πόσο μεγαλύτερο είναι καθορίζεται

από την επιλεκτικότητα της ερώτησης σε κάθε διάσταση. Γίνεται φανερό, λοιπόν, ότι το κόστος των δύο αλγορίθμων που εφαρμόζονται στο MAAN μπορεί να ταυτίζεται. Αν, όμως, υπερέχει κάποιος από τους δύο αλγορίθμους, δηλαδή εμφανίζει μικρότερο κόστος σε σχέση με τον άλλο, τότε αυτός είναι ο δεύτερος, στον οποίο μία ερώτηση διαστήματος δρομολογείται με τον ίδιο ακριβώς τρόπο, αλλά επιστρέφει πιθανότατα διαφορετικό πλήθος δεδομένων.

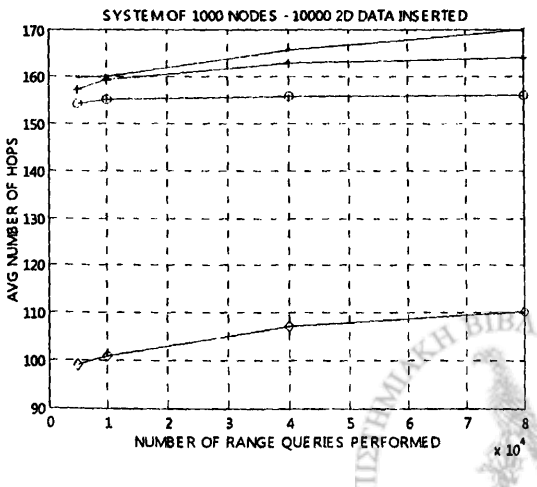
Επίσης, παρατηρούμε ότι το κόστος του αλγορίθμου επίλυσης που εφαρμόζεται στο Mercury είναι σε κάθε περίπτωση μικρότερο όλων των άλλων. Αυτό οφείλεται στο γεγονός πως οι κόμβοι του συστήματος διαμοιράζονται σε τόσους δακτυλίους όσες και οι διαστάσεις των δεδομένων που εισάγονται στο σύστημα, με αποτέλεσμα να μειώνεται το πλήθος των κόμβων που πρέπει να εξεταστούν για να βρεθούν τα δεδομένα που πρέπει να επιστραφούν για την ικανοποίηση μίας ερώτησης διαστήματος. Σημειώνεται, ωστόσο, πως κάθε κόμβος καλείται να ικανοποιήσει μία ερώτηση διαστήματος παρέχοντας πολύ περισσότερα δεδομένα.

Τέλος, αξίζει να σημειωθεί πως σε κάθε σταθερό σύστημα κόμβων, το μέσο κόστος μίας ερώτησης διαστήματος είναι περίπου ίδιο, όσες ερωτήσεις κι αν πραγματοποιούνται στο σύστημα. Αυτό οφείλεται στο γεγονός πως το κόστος των ερωτήσεων επηρεάζεται από το μέγεθος κάθε συστήματος και την επιλεκτικότητα των ερωτήσεων που πραγματοποιούνται σε αυτό, αλλά όχι και από το πλήθος των πραγματοποιούμενων ερωτήσεων.

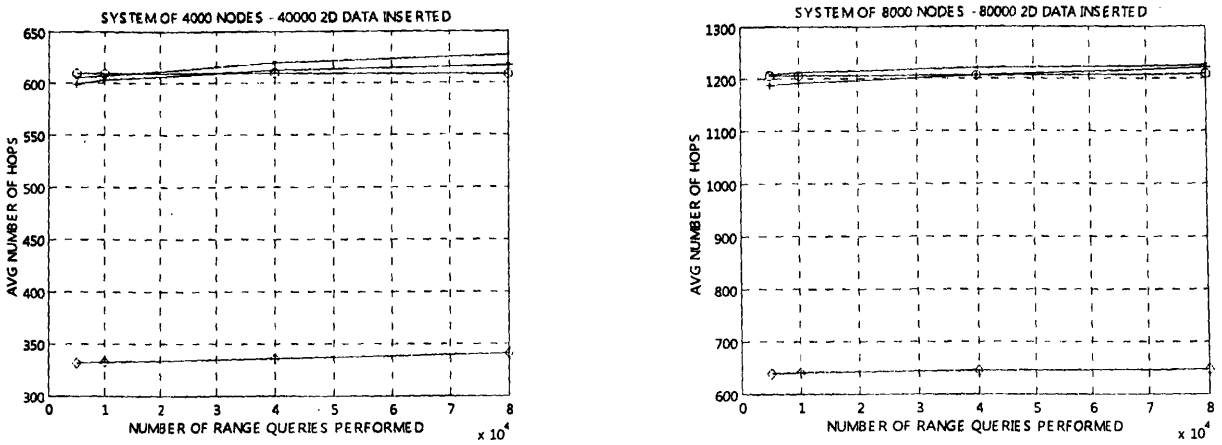




(α)



(β)



(γ) (δ)

Σχήμα 10.20 Κόστος Αλγορίθμων Επίλυσης Ερωτήσεων Διαστήματος ως προς Πλήθος Απαιτούμενων Βημάτων καθώς Μεταβάλλεται το Πλήθος των Ερωτήσεων Διαστήματος που Πραγματοποιούνται.

Στις επόμενες γραφικές παραστάσεις (Σχήμα 10.21) παρουσιάζουμε το μέσο κόστος των αλγορίθμων επίλυσης ερωτήσεων διαστήματος όταν μετρούμε τον αριθμό μηνυμάτων που απαιτούνται για την εύρεση των δεδομένων που πρέπει να επιστραφούν, όχι, όμως, και για την επιστροφή τους.

Παρατηρούμε, λοιπόν, πως το μέσο κόστος μίας ερώτησης διαστήματος στο TreeChord είναι αυξημένο συγκρινόμενο με το αντίστοιχο μέσο κόστος στο Mercury ή στο MAAN όπου εφαρμόζεται ο δεύτερος αλγόριθμος επίλυσης ερωτήσεων διαστήματος, αλλά και αρκετά μικρότερο συγκρινόμενο με το αντίστοιχο μέσο κόστος στο MAAN όπου εφαρμόζεται ο πρώτος αλγόριθμος. Αυτό οφείλεται στο γεγονός πως στο σύστημα του TreeChord αρκετές φορές, αλλά όχι όλες, πρέπει να εξεταστούν περισσότερα από ένα μονοπάτια στο σύστημα, άρα και περισσότεροι κόμβοι. Το πλήθος των ερωτήσεων που απαιτούν την εξέταση περισσότερων του ενός μονοπατιού υπολογίστηκε πειραματικά. Μετά από μία σειρά πειραμάτων, λοιπόν, παρατηρήσαμε ότι όταν η επιλεκτικότητα μίας ερώτησης διαστήματος (σε κάθε διάσταση) είναι 0.15, 40% των ερωτήσεων που πραγματοποιούνται στο σύστημα καταλήγουν στην εξέταση ενός μόνο μονοπατιού, ενώ 60% των ερωτήσεων καταλήγουν στην εξέταση δύο μονοπατιών. Αυτή η παρατήρηση μας οδηγεί στον ακόλουθο υπολογισμό της σταθεράς σ και του κόστους:

$$avgMsgs = \frac{40}{100} \times N \times s + \frac{60}{100} \times N \times s \times 2 + \log N = 1.6 \times N \times s + \log N,$$

που δικαιολογεί το αυξημένο κόστος του αλγορίθμου επίλυσης ερωτήσεων διαστήματος στο σύστημα του TreeChord. Ωστόσο, αξίζει να σημειωθεί πως αν και μία ερώτηση διαστήματος προωθείται σε πολύ περισσότερους κόμβους, οι ίδιοι αυτοί κόμβοι που λαμβάνουν μία ερώτηση καλούνται να την ικανοποιήσουν παρέχοντας πολύ λίγα δεδομένα. Επιπλέον, τα μηνύματα που προωθούνται για την επίλυση μίας ερώτησης διαστήματος δεν επιβαρύνουν κάποια περιοχή του συστήματος δημιουργώντας υπερφόρτωση, αφού διασπείρονται σε διαφορετικά μονοπάτια που αποτελούνται από διαφορετικούς κόμβους σε διαφορετικές περιοχές του συστήματος.

Από την άλλη, όταν εξετάζουμε το σύστημα του MAAN, παρατηρούμε ότι όταν εφαρμόζεται ο πρώτος αλγόριθμος, το κόστος μίας ερώτησης διαστήματος, κατά μέσο όρο, είναι σχεδόν διπλάσιο από ότι όταν εφαρμόζεται ο δεύτερος αλγόριθμος, όποια προσέγγιση κι αν ακολουθείται για να αποφασίσουν οι κόμβοι που λαμβάνουν την ερώτηση ποια δεδομένα πρέπει να επιστρέψουν. Αυτό συμβαίνει καθώς τα δεδομένα που εισάγονται στο σύστημα είναι δύο διαστάσεων και κατά την εφαρμογή του πρώτου αλγορίθμου εκτελείται μία ερώτηση σε κάθε μία από τις διαστάσεις, ενώ κατά την εφαρμογή του δεύτερου αλγορίθμου εκτελείται μία ερώτηση σε μία μόνο διάσταση. Αντιλαμβανόμαστε, λοιπόν, ότι το κόστος του πρώτου αλγορίθμου εξαρτάται από το πλήθος των διαστάσεων των δεδομένων του συστήματος, ενώ του δεύτερου αλγορίθμου είναι ανεξάρτητο. Μάλιστα, όσο αυξάνεται το πλήθος των διαστάσεων των δεδομένων, τόσο αυξάνεται το κόστος του πρώτου αλγορίθμου, σε αντίθεση με το αντίστοιχο κόστος του δεύτερου αλγορίθμου που παραμένει αμετάβλητο. Επομένως, αν τα δεδομένα του συστήματος ήταν M διαστάσεων και εφαρμοζόταν ο πρώτος αλγόριθμος, το κόστος μίας ερώτησης διαστήματος θα ήταν σχεδόν M φορές το κόστος του δεύτερου αλγορίθμου.

Σημειώνεται, επίσης, ότι στην προκειμένη περίπτωση, το κόστος του δεύτερου αλγορίθμου είναι σχεδόν το μισό του κόστους του πρώτου και όχι πολύ μικρότερο, αφού η επιλεκτικότητα των ερωτήσεων διαστήματος είναι ίδια για κάθε διάσταση. Αν η επιλεκτικότητα των ερωτήσεων ήταν διαφορετική για κάθε διάσταση, πιθανώς το κόστος του δεύτερου αλγορίθμου θα ήταν ακόμη μικρότερο, αφού μία ερώτηση εκτελείται πάντα στη διάσταση για την οποία η επιλεκτικότητα είναι ελάχιστη και, επιπλέον, η κατανομή των κόμβων στο δακτύλιο γίνεται όπως και στο σύστημα του Chord και, επομένως, είναι ομοιόμορφη.

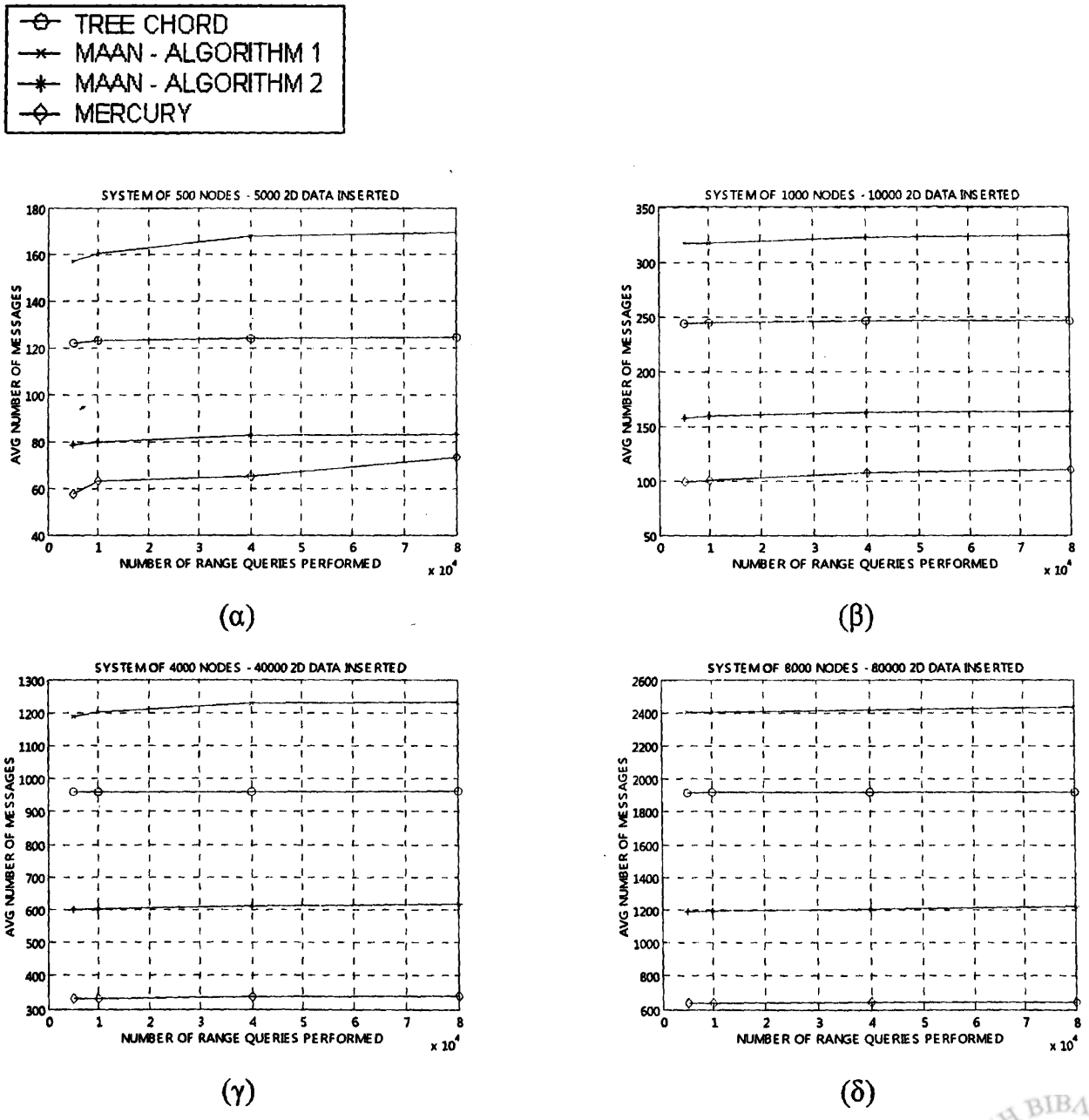
Γίνεται φανερό, λοιπόν, ότι η διαφορά μεταξύ του κόστους των δύο αλγορίθμων που εφαρμόζονται στο MAAN είναι πολύ σημαντική και δείχνει πόσο λιγότερο επιβαρύνεται το σύστημα από τα μηνύματα των κόμβων που αναζητούν δεδομένα όταν εφαρμόζεται ο δεύτερος αλγόριθμος, ιδιαίτερα όταν αυξάνεται το πλήθος των διαστάσεων των δεδομένων του συστήματος ή οι ερωτήσεις παρουσιάζουν διαφορετική επιλεκτικότητα σε κάθε διάσταση.

Ακόμη, παρατηρούμε ότι το κόστος μίας ερώτησης διαστήματος στο Mercury είναι αρκετά μικρότερο από το αντίστοιχο κόστος που υπολογίσαμε για τα άλλα συστήματα. Αυτό οφείλεται στο γεγονός πως οι κόμβοι του συστήματος οργανώνονται σε τόσους δακτυλίους, όσα και το πλήθος των διαστάσεων των εισαγόμενων στο σύστημα δεδομένων. Μάλιστα, οι δακτύλιοι που δημιουργούνται έχουν περίπου το ίδιο μέγεθος, δηλαδή σε κάθε ένα από αυτά συμμετέχει περίπου ίδιο πλήθος κόμβων, αφού κάθε κόμβος αποφασίζει με ομοιόμορφα τυχαίο τρόπο σε ποιον από τους δακτυλίους του συστήματος θα εισέλθει. Επιπλέον, μία ερώτηση διαστήματος πραγματοποιείται σε έναν μόνο από τους δακτυλίους του συστήματος, όχι σε περισσότερα, γεγονός που μειώνει σημαντικά το κόστος της, αφού περιορίζει το πλήθος των κόμβων στους οποίους θα πρέπει να προωθηθεί προκειμένου να ικανοποιηθεί. Αντιλαμβανόμαστε, λοιπόν, ότι το κόστος του αλγορίθμου επίλυσης ερωτήσεων διαστήματος στο Mercury εξαρτάται από το πλήθος των διαστάσεων των δεδομένων του συστήματος και, μάλιστα, όπως θα δούμε και στη συνέχεια, όσο το πλήθος των διαστάσεων των δεδομένων του συστήματος αυξάνεται, τόσο το κόστος των ερωτήσεων διαστήματος μειώνεται. Σημειώνεται, επίσης, ότι το κόστος του αλγορίθμου στο Mercury θα μειωνόταν αν η επιλεκτικότητα των ερωτήσεων ήταν διαφορετική για κάθε διάσταση, καθώς για την επίλυση κάθε ερώτησης διαστήματος επιλέγεται και πάλι η διάσταση για την οποία η επιλεκτικότητα που ορίζεται είναι ελάχιστη. Ωστόσο, αξίζει να σημειώσουμε πως αν και μία ερώτηση διαστήματος στο Mercury προωθείται σε λιγότερους κόμβους, οι ίδιοι αυτοί κόμβοι καλούνται να την επιλύσουν παρέχοντας πολύ περισσότερα δεδομένα.

Έτσι, παρατηρούμε πως όσο αυξάνεται το μέγεθος των συστημάτων, τόσο αυξάνεται και η διαφορά του κόστους του αλγορίθμου επίλυσης που εφαρμόζεται στο Mercury και του πρώτου (αντ. του δεύτερου) αλγορίθμου που εφαρμόζεται στο MAAN. Πιο συγκεκριμένα, το κόστος στο MAAN είναι περίπου 50% μεγαλύτερο (αντ. διπλάσιο) στα συστήματα των 500

και 1000 κόμβων και σχεδόν διπλάσιο (αντ. τετραπλάσιο) στα συστήματα των 4000 και 8000 κόμβων.

Επίσης, παρατηρούμε πως όσο αυξάνεται το μέγεθος των συστημάτων, τόσο αυξάνεται και η διαφορά του κόστους του αλγορίθμου επίλυσης που εφαρμόζεται στο Mercury και του αλγορίθμου που εφαρμόζεται στο TreeChord. Πιο συγκεκριμένα, το κόστος στο Mercury είναι περίπου διπλάσιο στο σύστημα των 500 κόμβων, σχεδόν δυόμισι φορές μεγαλύτερο στο σύστημα των 1000 κόμβων και σχεδόν τριπλάσιο και ακόμη μεγαλύτερο στα συστήματα των 4000 και 8000 κόμβων, αντίστοιχα.



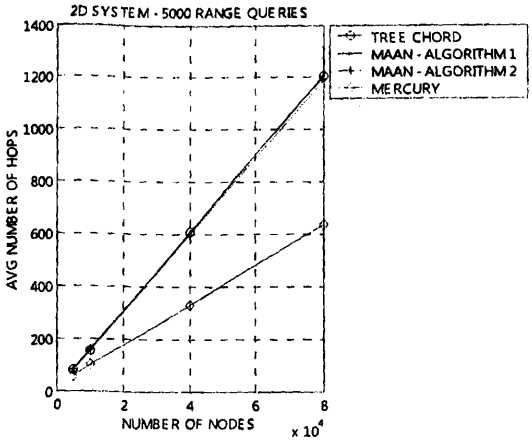
Σχήμα 10.21 Κόστος Αλγορίθμων Επίλυσης Ερωτήσεων Διαστήματος ως προς Πλήθος Απαιτούμενων Μηνυμάτων καθώς Μεταβάλλεται το Πλήθος των Ερωτήσεων Διαστήματος που Πραγματοποιούνται.

10.3.2. Κόστος Αλγορίθμων Επίλυσης Ερωτήσεων Διαστήματος όταν Μεταβάλλεται το Μέγεθος του Συστήματος

Στα πειράματα της κατηγορίας αυτής εξετάζουμε πώς μεταβάλλεται το κόστος των αλγορίθμων επίλυσης ερωτήσεων διαστήματος όταν μεταβάλλεται το μέγεθος του συστήματος, δηλαδή το πλήθος των κόμβων που συμμετέχουν σε αυτό. Υπενθυμίζουμε ότι υπολογίζεται το πλήθος των μηνυμάτων ή βημάτων που απαιτούνται για την εύρεση των δεδομένων που πρέπει να επιστραφούν, όχι, όμως, και για την επιστροφή τους στον κόμβο που εκκινεί την ερώτηση διαστήματος. Μελετούμε και πάλι τα συστήματα που περιγράφηκαν στις προηγούμενες ενότητες.

Η επόμενη γραφική παράσταση (Σχήμα 10.22) προκύπτει όταν υπολογίζουμε το μέσο κόστος μετρώντας το πλήθος των βημάτων που αποστέλλονται. Οι διαφορές μεταξύ των αλγορίθμων δικαιολογούνται από όσα ειπώθηκαν παραπάνω. Αξίζει, όμως, να σημειωθεί ότι το κόστος όλων των αλγορίθμων επίλυσης ερωτήσεων διαστήματος εξαρτάται από το πλήθος των κόμβων που συμμετέχουν στο σύστημα και, μάλιστα, όσο αυξάνεται ο αριθμός των κόμβων του συστήματος, τόσο αυξάνεται και το κόστος των αλγορίθμων επίλυσης ερωτήσεων διαστήματος. Αυτό δικαιολογείται από το γεγονός πως σε κάθε τύπο κόστους που αναφέραμε παραπάνω υπάρχει ένας παράγοντας $N \times s$, που δείχνει τη γραμμική αύξηση που έχει το κόστος όταν αυξάνεται γραμμικά το πλήθος των κόμβων που συμμετέχουν σε ένα σύστημα (υπάρχει και ένας λογαριθμικός παράγοντας, αλλά είναι ασήμαντος).

Αντίστοιχα, γραμμική είναι η αύξηση του κόστους ως προς το πλήθος των βημάτων που απαιτούνται για την εύρεση των δεδομένων που πρέπει να επιστραφούν όταν αυξάνεται γραμμικά το πλήθος των κόμβων που συμμετέχουν σε ένα σύστημα.



Σχήμα 10.22 Κόστος Αλγορίθμων Επίλυσης Ερωτήσεων Διαστήματος ως προς Πλήθος Απαιτούμενων Βημάτων καθώς Μεταβάλλεται το Πλήθος των Κόμβων που Συμμετέχουν στο Σύστημα.

10.3.3. Κόστος Αλγορίθμων Επίλυσης Ερωτήσεων Διαστήματος όταν Μεταβάλλεται το Πλήθος των Διαστάσεων των Δεδομένων

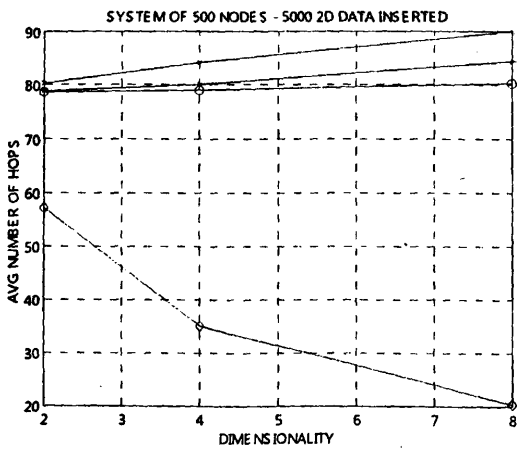
Στα πειράματα της κατηγορίας αυτής εξετάζουμε πώς μεταβάλλεται το μέσο κόστος των αλγορίθμων επίλυσης ερωτήσεων διαστήματος όταν μεταβάλλεται το πλήθος των διαστάσεων των δεδομένων του συστήματος. Υπενθυμίζουμε ότι υπολογίζεται το πλήθος των μηνυμάτων ή βημάτων που απαιτούνται για την εύρεση των δεδομένων που πρέπει να επιστραφούν, όχι, όμως, και για την επιστροφή τους στον κόμβο που εκκινεί την ερώτηση διαστήματος. Και πάλι μελετούμε τα ίδια συστήματα με τις προηγούμενες ενότητες. Χρησιμοποιούμε δεδομένα 2, 4 και 8 διαστάσεων.

Οι επόμενες γραφικές παραστάσεις (Σχήμα 10.23) προκύπτουν όταν υπολογίζουμε το μέσο κόστος ενός αλγορίθμου μετρώντας το πλήθος των βημάτων που απαιτούνται για την εύρεση των δεδομένων που πρέπει να επιστραφούν. Η μείωση που παρουσιάζει το κόστος του αλγορίθμου στο Mercury οφείλεται, όπως προαναφέρθηκε, στο γεγονός πως οι κόμβοι οργανώνονται σε τόσους σχεδόν ισομεγέθεις δακτυλίους, όσες και οι διαστάσεις των δεδομένων του συστήματος. Σημειώνουμε, όμως, πως παρότι μία ερώτηση διαστήματος προωθείται σε πολύ λιγότερους κόμβους, οι κόμβοι που τελικά απαντούν παρέχουν περισσότερα δεδομένα.

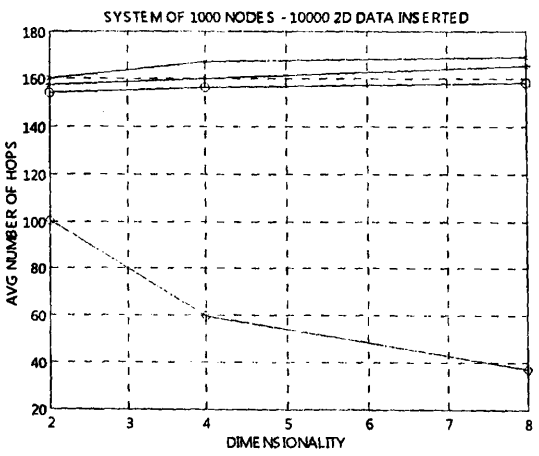
Από την άλλη, για το σύστημα του MAAN, παρατηρούμε ότι το κόστος και των δύο αλγορίθμων είναι σχεδόν ίδιο και δε μεταβάλλεται καθώς αυξάνεται το πλήθος των

διαστάσεων των δεδομένων του συστήματος. Στην περίπτωση αυτή, κατά την εφαρμογή του πρώτου αλγορίθμου στην οποία εκτελείται μία ερώτηση σε κάθε μία από τις διαστάσεις του συστήματος, υπολογίζουμε το πλήθος των βημάτων που απαιτούνται για την εκτέλεσή της σε κάθε μία από τις διαστάσεις και, έπειτα, το μέγιστο των τιμών αυτών, το οποίο αποτελεί και το τελικό κόστος της ερώτησης. Σε όλες τις περιπτώσεις, λοιπόν, το κόστος μιας ερώτησης διαστήματος υπολογίζεται ως το πλήθος των βημάτων που απαιτούνται για την εκτέλεση μίας ερώτησης σε κάποια από τις διαστάσεις των δεδομένων. Καθώς, λοιπόν, η επιλεκτικότητα των ερωτήσεων σε κάθε διάσταση είναι ίδια και οι κόμβοι του συστήματος κατανέμονται ομοιόμορφα στο δακτύλιο, το κόστος του πρώτου αλγορίθμου δεν είναι πολύ μεγαλύτερο από το αντίστοιχο κόστος του δεύτερου αλγορίθμου. Στην καλύτερη περίπτωση, μάλιστα, για τον πρώτο αλγόριθμο το κόστος του θα ταυτίζεται με το κόστος του δευτέρου αλγορίθμου.

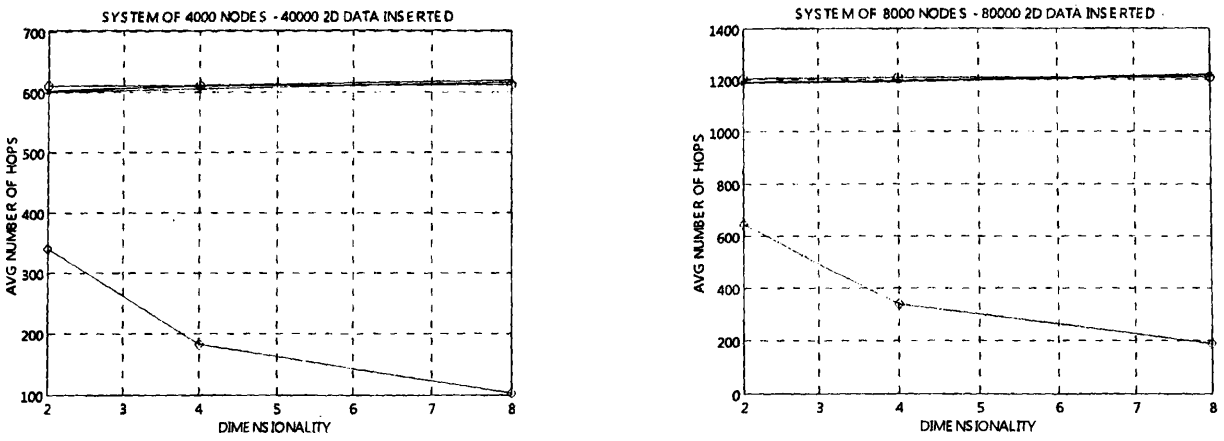
Για το σύστημα του TreeChord, παρατηρούμε ότι το κόστος, επίσης, δε μεταβάλλεται καθώς αυξάνεται το πλήθος των διαστάσεων των δεδομένων του συστήματος. Στην περίπτωση αυτή, κατά την επίλυση μίας ερώτησης διαστήματος, υπολογίζουμε το πλήθος των βημάτων που απαιτούνται για να βρεθούν τα κατάλληλα δεδομένα στο μακρύτερο μονοπάτι από αυτά που εξερευνώνται, αλλά μόνο σε ένα από αυτά.



(α)



(β)



(γ) (δ)

Σχήμα 10.23 Κόστος Αλγορίθμων Επίλυσης Ερωτήσεων Διαστήματος ως προς Πλήθος Απαιτούμενων Βημάτων σε Συστήματα με Διαφορετικό Πλήθος Διαστάσεων.

Οι επόμενες γραφικές παραστάσεις (Σχήμα 10.24) προκύπτουν όταν υπολογίζουμε το κόστος ενός αλγορίθμου μετρώντας το πλήθος των μηνυμάτων που απαιτούνται για την εύρεση των δεδομένων που πρέπει να επιστραφούν.

Όταν εξετάζουμε το σύστημα του TreeChord, παρατηρούμε ότι σε ένα σταθερό σύστημα κόμβων, το μέσο κόστος του αλγορίθμου επίλυσης ερωτήσεων διαστήματος παραμένει σχεδόν ίδιο, ανεξάρτητα από το πλήθος των διαστάσεων που έχουν τα δεδομένα. Όπως αναφέρθηκε σε προηγούμενο κεφάλαιο, το πλήθος των διαστάσεων των δεδομένων του συστήματος καθορίζει το μέγεθος της δενδρικής δομής που υπολογίζεται, δηλαδή το πλήθος των επιπέδων από τα οποία αποτελείται. Όσες κι αν είναι, όμως, οι διαστάσεις, μία ερώτηση διαστήματος οδηγεί στην εξερεύνηση το πολύ 2^d μονοπατιών, που ξεκινούν από φύλλα τα οποία καθορίζονται από το τελικό επίπεδο της δενδρικής δομής που υπολογίζεται. Το ακριβές πλήθος των μονοπατιών καθορίζεται από την επιλεκτικότητα της ερώτησης διαστήματος στην τελευταία διάσταση και προσδιορίζεται πειραματικά. Στα πειράματα που πραγματοποιήσαμε, η επιλεκτικότητα μίας ερώτησης διαστήματος (σε κάθε διάσταση) είναι 15%. Στην περίπτωση αυτή, παρατηρήσαμε ότι 40% των ερωτήσεων που πραγματοποιούνται στο σύστημα καταλήγουν στην εξέταση ενός μόνο μονοπατιού, ενώ 60% των ερωτήσεων καταλήγουν στην εξέταση δύο μονοπατιών. Επομένως, ο υπολογισμός της σταθεράς σ και του κόστους έχει ως εξής:

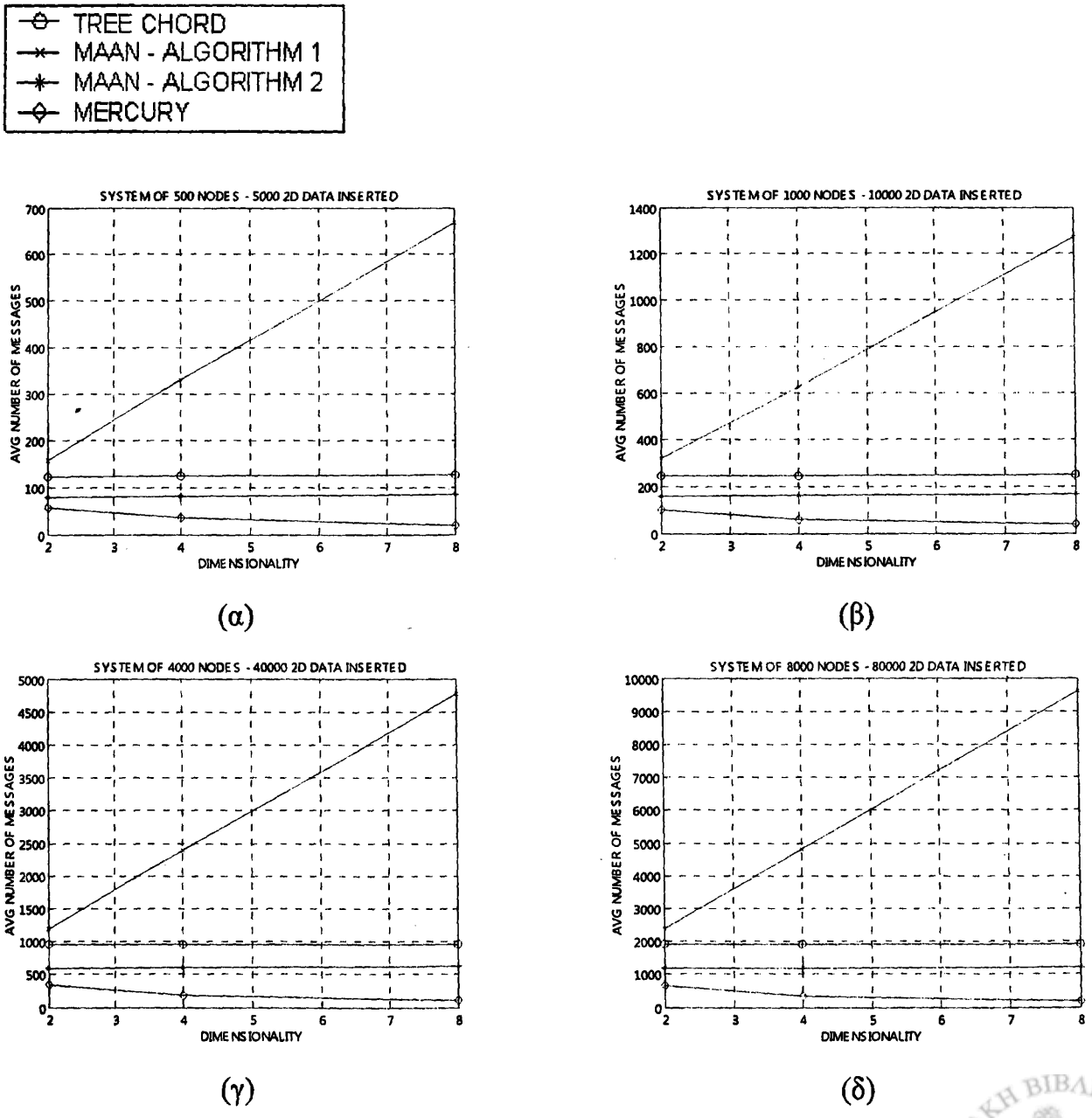
$$avgMsgs = \frac{40}{100} \times N \times s + \frac{60}{100} \times N \times s \times 2 + \log N = 1.6 \times N \times s + \log N,$$

που δικαιολογεί το αυξημένο κόστος του αλγορίθμου επίλυσης ερωτήσεων διαστήματος στο σύστημα του TreeChord, σε σχέση με το αντίστοιχο μέσο κόστος στο Mercury ή στο MAAN όπου εφαρμόζεται ο δεύτερος αλγόριθμος επίλυσης ερωτήσεων διαστήματος.

Όταν εξετάζουμε το σύστημα του MAAN, παρατηρούμε ότι σε ένα σταθερό σύστημα κόμβων το μέσο κόστος του δεύτερου αλγορίθμου παραμένει σχεδόν ίδιο, ανεξάρτητα από το πλήθος των διαστάσεων που έχουν τα δεδομένα. Αυτό συμβαίνει καθώς όσες κι αν είναι οι διαστάσεις των δεδομένων του συστήματος, κατά την εκτέλεση μίας ερώτησης διαστήματος με εφαρμογή του δευτέρου αλγορίθμου εκτελείται μία ερώτηση σε μία μόνο διάσταση και, μάλιστα, εκείνη για την οποία η επιλεκτικότητα είναι ελάχιστη. Αντίθετα, όταν εφαρμόζεται ο πρώτος αλγόριθμος, το μέσο κόστος αυξάνεται καθώς αυξάνεται το πλήθος των διαστάσεων των δεδομένων του συστήματος. Πιο συγκεκριμένα, κατά την εφαρμογή του πρώτου αλγορίθμου απαιτείται σχεδόν διπλάσιο πλήθος μηνυμάτων όταν τα δεδομένα είναι δύο διαστάσεων, σχεδόν τετραπλάσιο πλήθος μηνυμάτων όταν τα δεδομένα είναι τεσσάρων διαστάσεων και σχεδόν οκταπλάσιο πλήθος μηνυμάτων όταν τα δεδομένα είναι οκτώ διαστάσεων. Το γεγονός πως για την εύρεση όλων των δεδομένων που πρέπει να επιστραφούν όταν εφαρμόζεται ο πρώτος αλγόριθμος επίλυσης ερωτήσεων διαστήματος πρέπει μα εκτελεστεί μία ερώτηση σε κάθε μία από τις διαστάσεις που ορίζονται μας οδηγεί στο συμπέρασμα πως αν τα δεδομένα του συστήματος είναι M διαστάσεων, ο πρώτος αλγόριθμος θα απαιτεί σχεδόν M φορές το κόστος του άλλου αλγορίθμου, δηλαδή το πλήθος των απαιτούμενων μηνυμάτων θα είναι σχεδόν M φορές το πλήθος των μηνυμάτων που απαιτεί ο δεύτερος αλγόριθμος.

Από την άλλη, όταν εξετάζουμε το σύστημα του Mercury, παρατηρούμε ότι σε κάθε σταθερό σύστημα κόμβων το κόστος του αλγορίθμου μειώνεται όσο αυξάνεται το πλήθος των διαστάσεων που έχουν τα δεδομένα. Αυτό συμβαίνει καθώς οι κόμβοι του συστήματος οργανώνονται σε τόσους σχεδόν ίσου μεγέθους δακτυλίων, όσες είναι και οι διαστάσεις των δεδομένων του συστήματος. Με άλλα λόγια, όσο αυξάνεται το πλήθος των διαστάσεων των δεδομένων του συστήματος και αφού η επιλεκτικότητα για κάθε διάσταση της ερώτησης είναι ίδια, τόσο μειώνεται το μέσο κόστος του αλγορίθμου, καθώς το μέγεθος του δακτυλίου στον οποίο εκτελείται μία ερώτηση διαστήματος μειώνεται. Πιο συγκεκριμένα, το κόστος του αλγορίθμου μειώνεται σχεδόν στο μισό όταν το πλήθος των διαστάσεων διπλασιάζεται και ακόμη περισσότερο όσο το πλήθος των διαστάσεων αυξάνεται.

Επίσης, το γεγονός πως το μέσο κόστος του αλγορίθμου επίλυσης στο Mercury είναι μικρότερο από το αντίστοιχο μέσο κόστος των αλγορίθμων που εφαρμόζονται στο MAAN ή στο TreeChord οφείλεται ακριβώς στο γεγονός πως στο Mercury οι κόμβοι διαμοιράζονται σε οργανώνονται σε τόσους δακτυλίους, όσες και οι διαστάσεις των αντίστοιχων συστημάτων, ενώ στο MAAN, αλλά και στο TreeChord όλοι οι κόμβοι οργανώνονται σε έναν και μόνο δακτύλιο. Υπενθυμίζουμε, όμως, πως οι κόμβοι καλούνται να ικανοποιήσουν μία ερώτηση διαστήματος παρέχοντας περισσότερα δεδομένα.



Σχήμα 10.24 Κόστος Αλγορίθμων Επίλυσης Ερωτήσεων Διαστήματος ως προς Πλήθος Απαιτούμενων Μηνυμάτων σε Συστήματα με Διαφορετικό Πλήθος Διαστάσεων.

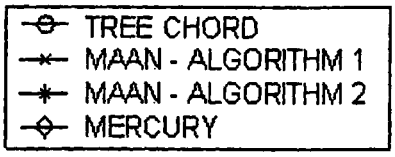
10.3.4. Κόστος Αλγορίθμων Επίλυσης Ερωτήσεων Διαστήματος όταν Μεταβάλλεται η Επιλεκτικότητα των Ερωτήσεων Διαστήματος

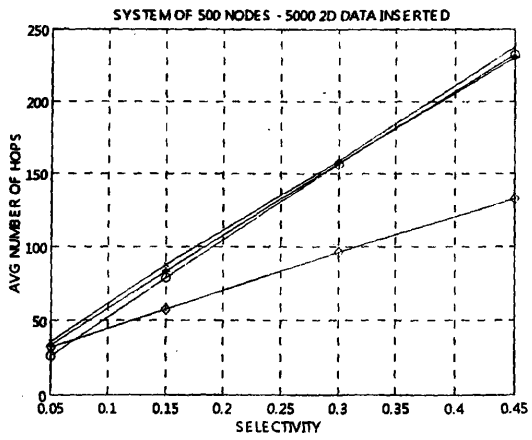
Στα πειράματα της κατηγορίας αυτής εξετάζουμε πώς μεταβάλλεται το μέσο κόστος των αλγορίθμων επίλυσης ερωτήσεων διαστήματος όταν μεταβάλλεται η επιλεκτικότητα των ερωτήσεων διαστήματος. Υπενθυμίζουμε πως υπολογίζουμε το πλήθος των μηνυμάτων ή βημάτων που απαιτούνται για την εύρεση των δεδομένων που πρέπει να επιστραφούν. Εξετάζουμε και πάλι τα ίδια συστήματα. Εκτελούμε ερωτήσεις διαστήματος με επιλεκτικότητα 5%, 15%, 30% και 45% για κάθε μία από τις διαστάσεις των δεδομένων.

Στις επόμενες γραφικές παραστάσεις (Σχήμα 10.25) υπολογίζουμε το κόστος των αλγορίθμων επίλυσης ερωτήσεων διαστήματος μετρώντας τον απαιτούμενο αριθμό βημάτων. Σε κάθε περίπτωση, παρατηρούμε ότι το κόστος όλων των αλγορίθμων αυξάνεται καθώς αυξάνεται η επιλεκτικότητα των ερωτήσεων διαστήματος που πραγματοποιούνται. Καθώς οι κόμβοι κάθε συστήματος κατανέμονται ομοιόμορφα στο δακτύλιο, όσο αυξάνεται η επιλεκτικότητα μιας ερώτησης, δηλαδή το εύρος του δακτυλίου που καλύπτει, τόσοι περισσότεροι κόμβοι θα ανήκουν στο ζητούμενο εύρος, επομένως, και θα πρέπει να λάβουν μηνύματα κατά την εκτέλεσή της. Για το λόγο αυτό το κόστος των αλγορίθμων αυξάνεται καθώς αυξάνεται η επιλεκτικότητα των ερωτήσεων διαστήματος που πραγματοποιούνται.

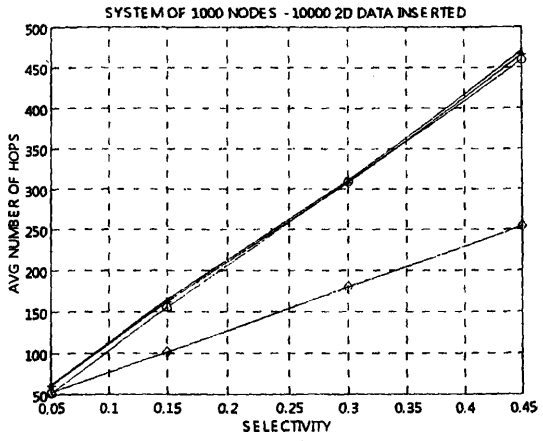
Στην περίπτωση του MAAN, βέβαια, παρατηρούμε ότι το κόστος του πρώτου αλγορίθμου είναι σχεδόν ίδιο με το αντίστοιχο κόστος του δεύτερου αλγορίθμου. Στην καλύτερη περίπτωση, μάλιστα, και σύμφωνα με όσα προαναφέρθηκαν, το κόστος των δύο αλγορίθμων θα ταυτιζόταν. Σε κάθε περίπτωση, πάντως, το πόσο μεγάλη θα ήταν η διαφορά μεταξύ του κόστους του δεύτερου αλγορίθμου από τη μία και του αντίστοιχου κόστους του πρώτου αλγορίθμου από την άλλη, εξαρτάται μόνο από την επιλεκτικότητα των ερωτήσεων διαστήματος σε κάθε διάσταση, αφού, φυσικά, οι κόμβοι που εισέρχονται στο σύστημα κατανέμονται ομοιόμορφα στο δακτύλιο.

Και στο σύστημα του TreeChord, όμως, το κόστος αυξάνεται γραμμικά καθώς αυξάνεται η επιλεκτικότητα των ερωτήσεων διαστήματος που πραγματοποιούνται στο σύστημα, αφού το κόστος υπολογίζεται ως το μήκος του μακρύτερου μονοπατιού από αυτά που ακολουθούνται κατά την επίλυση μίας ερώτησης διαστήματος, το οποίο ισούται με $N \times s$ και δεν εξαρτάται από τη σταθερά σ που προαναφέρθηκε.

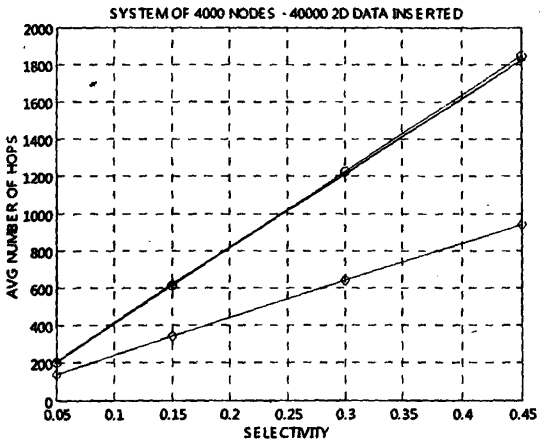




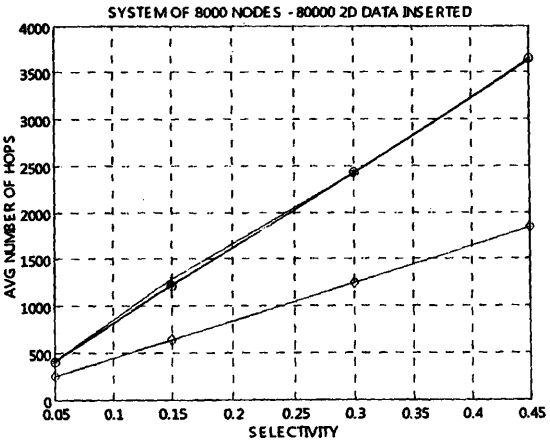
(α)



(β)



(γ)



(δ)

Σχήμα 10.25 Κόστος Αλγορίθμων Επίλυσης Ερωτήσεων Διαστήματος ως προς Πλήθος Απαιτούμενων Βημάτων για Ερωτήσεις Διαστήματος με Διαφορετική Επιλεκτικότητα.

ΚΕΦΑΛΑΙΟ 11. ΕΠΙΛΟΓΟΣ

Στη διατριβή αυτή, μελετήθηκε το πρόβλημα της αποθήκευσης και διαχείρισης πολυδιάστατων δεδομένων σε δομημένα συστήματα διομότιμων κόμβων. Αρχικά, παρουσιάστηκαν κάποια από τα συστήματα που έχουν προταθεί. Το πρώτο από αυτά, το MAAN, βασίζεται στο σύστημα του Chord. Η είσοδος και η αποχώρηση των κόμβων γίνονται όπως ακριβώς και στο Chord. Η κατανομή των δεδομένων, όμως, γίνεται μέσω συναρτήσεων κατακερματισμού που διατηρούν τη διάταξη. Το δεύτερο σύστημα, το Mercury, οργανώνει τους κόμβους σε λογικούς δακτυλίους αναθέτοντάς τους κάποιο εύρος τιμών, ενώ η κατανομή των δεδομένων γίνεται βάσει των πραγματικών τους τιμών κι όχι μέσω τεχνικών κατακερματισμού. Προτάθηκε, επίσης, ένα νέο σύστημα, το TreeChord. Το σύστημα αυτό χρησιμοποιεί το δακτύλιο του Chord για την οργάνωση των κόμβων και των δεδομένων. Η είσοδος και η αποχώρηση κόμβων από το σύστημα γίνεται όπως ακριβώς και στο Chord. Η διαχείριση των δεδομένων, όμως, βασίζεται σε μία δενδρική δομή που χωρίζει το δακτύλιο σε ισομεγέθη τμήματα. Τα παραπάνω συστήματα μελετήθηκαν και συγκρίθηκαν θεωρητικά και πειραματικά και αποδείχθηκε πως το σύστημα του TreeChord υπερέχει ως προς διάφορα ενδιαφέροντα μετρικά.

Ένα ενδιαφέρον ζήτημα θα ήταν να εξεταστεί πώς μπορεί να εφαρμοστεί στα συστήματα ο μηχανισμός εξισορρόπησης φορτίου που προτείνεται στο Mercury. Η τεχνική αυτή θα μπορούσε να αξιολογηθεί πειραματικά με μετρήσεις της απόδοσης των συστημάτων ως προς διάφορα μετρικά. Επίσης, ιδιαίτερα ενδιαφέρον θα ήταν να εξεταστεί πώς το μέγεθος της δενδρική δομής του TreeChord θα μπορούσε να αλλάζει δυναμικά στο σύστημα και πώς αυτό θα επηρέαζε την απόδοση.

ΑΝΑΦΟΡΕΣ

[1] K. Aberer. "P-Grid: A Self-Organizing Access Structure for P2P Information Systems". Sixth International Conference on Cooperative Information Systems (CoopIS 2001), Lecture Notes in Computer Science, 2172: 179 – 194, 2001.

[2] E. Adar and B. Huberman. "Free Riding on Gnutella". First Monday, October 2000.

[3] A. R.Bharambe, M. Agrawal, and S. Seshan. "Mercury: Supporting Scalable Multi-Attribute Range Queries". SIGCOMM '04, 2004.

[4] M. Cai, M. Frank, J. Chen and P. Szekely. "MAAN: a multi-attribute addressable network for grid information services". Proceedings of the 4th International Workshop on Grid Computing, 2003.

[5] J. L. Carter and M. N. Wegman. "Universal classes of hash functions". Journal of Computer and System Sciences 18, 2, 1979.

[6] I. Clarke, S. G. Miller, T. W. Hong, O.Sandberg and B. Wiley. "Protecting Free Expression Online with Freenet". IEEE Internet Computing, 2002.

[7] I. Clarke, O. Sandberg, B. Wiley and T. W. Hong. "Freenet: A Distributed Anonymous Information Storage and Retrieval System". Proceedings of the ICSI Workshop on Design Issues in Anonymity and Unobservability, July 2000.

[8] P. Ganesan, B. Yang and H. Garcia Molina. "One Torus to Rule Them All: Multi-dimensional Queriesin P@P Systems". In WebDB, pages 19 – 24, 2004.

[9] N. Harvey, M. B. Jones, S. Saroiu, M. Theimer and A. Wolman. "SkipNet: A Scalable Overlay Network with Practical Locality Properties". Technical Report MSR-TR-2002-92, Microsoft Research, 2002.

[10] N. Harvey, M. B. Jones, S. Saroiu, M. Theimer and A. Wolman. "SkipNet: A Scalable Overlay Network with Practical Locality Properties". Fourth USENIX Symposium on Internet Technologies and Systems (USITS '03), 2003.

[11] HTTP://FREENETPROJECT.ORG/

[12] HTTP://GNUTELLA.WEGO.COM/

[13] HTTP://WWW.NAPSTER.COM/

[14] H. V. Jagadish, B. C. Ooi and Q. H. Vu. "BATON: a balanced tree structure for peer-to-peer networks". VLDB 2005, pages 661 – 672, 2005.

[15] H. V. Jagadish, B. C. Ooi, Q. H. Vu, K. L. Tan, Q. H. Vu and R. Zhang. Speeding up Search in Peer-to-Peer Networks with a Multi-way Tree Structure". In SIGMOD 2006.

[16] H. V. Jagadish, B. C. Ooi, Q. H. Vu, R. Zhang and A. Zhou. "VBI-Tree: A Peer-to-Peer Framework for Supporting Multi-Dimensional Indexing Schemes". In ICDE 2006.

[17] D. Karger, E. Lehman, F. Leighton, M. Levine, D. Lewin and R. Panigrahy. "Consistent hashing and random trees: Distributed hashing protocols for relieving hot spots on the World Wide Web". Proceedings of the 29th Annual ACM Symposium on Theory of Computing (El Paso, TX, May 1997).

[18] J. Kleinberg. "The Small-World Phenomenon: An Algorithmic Perspective". Proceedings 32nd ACM Symposium on Theory of Computing, 2000.

[19] H. R. Lewis and L. Denenberg. "Data Structures & Their Algorithms". Harper Collins Publishers, 1991.

[20] G. S. Manku, M. Bawa and P. Raghanan. "Symphony: Distributed Hashing In A Small World". Proceedings of the. 4th USENIX Symposium on Internet Technologies and Systems, 2003.

[21] O. D. Sahin, A. Gupta, D. Agrawal and A. El Abbadi. "A Peer-to-peer Framework for Caching Range Queries". 20th International Conference on Data Engineering (ICDE '04), 2004.

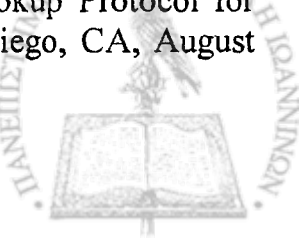
[22] T. Pitoura, N. Ntarmos and P. Triantafillou. "Replication, Load Balancing and Efficient Range Query Processing in DHTs". In EDBT, pages 131 – 148, 2006.

[23] S. Ratnasamy, P. Francis, M. Handley, R, Karp and S. Shenker. "A Scalable Content-Addressable Network". ICSI Technical Report, January 2001.

[24] M. Ripeanu. "Peer-to-Peer Architecture Case Study: Gnutella Network". Proceedings of the 2001 International conference on P2P computing.

[25] A. Rowstron and P. Druschel. "Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems". IFIP/ACM Middleware, November 2001.

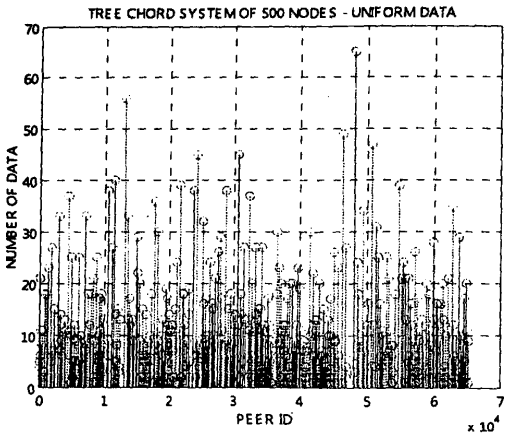
[26] I. Stoica, R. Morris, D. Liben-Nowell, D. R. Karger, M. Frans Kaashoek, F. Dabek and H. Balakrishnan. "Chord: A Scalable Peer-to-Peer Lookup Protocol for Internet Applications". Proceedings of the ACM Sigcomm, San Diego, CA, August 2001.



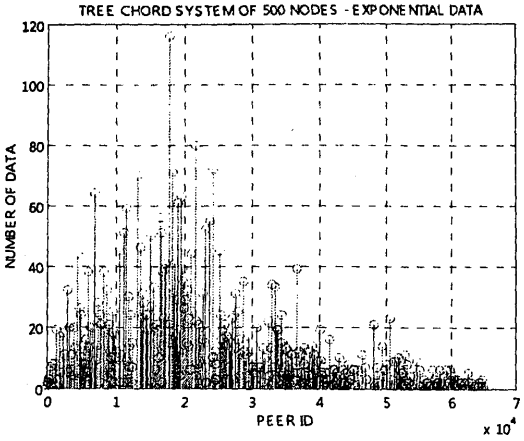
[27] C. Zheng, G. Shen, S. Lind and S. Shenker. "Distributed Segment Tree: Support of Range Query and Cover Query over DHT". In IPTPS, 2006.

ΠΑΡΑΡΤΗΜΑ Α

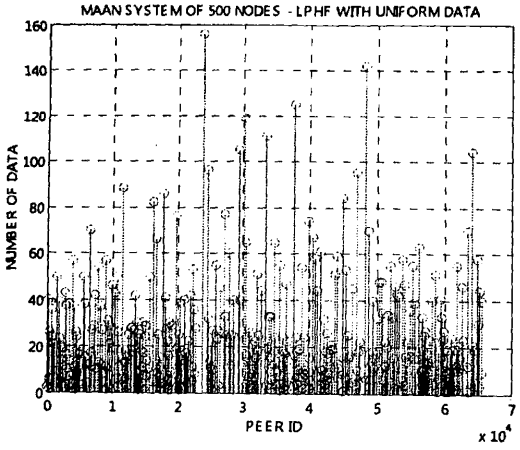
Εκτελούμε τα ίδια πειράματα με εκείνα της Ενότητας 10.1.1, αλλά στις γραφικές παραστάσεις (Σχήμα Α.1 – Σχήμα Α.4) που παρουσιάζουμε απεικονίζουμε πόσα δεδομένα διατηρεί κάθε κόμβος. Μάλιστα, η διάταξη των κόμβων στη γραφική παράσταση είναι η διάταξή τους στο σύστημα στο οποίο ανήκουν. Τα αποτελέσματα δικαιολογούνται από όσα αναφέρθηκαν στην Ενότητα 10.1.1. Συνοπτικά, και πάλι παρατηρούμε πως όταν τα αναγνωριστικά των δεδομένων εισόδου ακολουθούν την ομοιόμορφη κατανομή για το σύστημα του Mercury, τα δεδομένα κατανέμονται ομοιόμορφα ή σχεδόν ομοιόμορφα στους κόμβους. Στην αντίθετη περίπτωση, δηλαδή όταν τα αναγνωριστικά των δεδομένων εισόδου ακολουθούν την εκθετική κατανομή, τα δεδομένα δεν κατανέμονται ομοιόμορφα στους κόμβους του συστήματος. Επίσης, παρατηρούμε πως όταν η κατανομή των αναγνωριστικών των δεδομένων εισόδου στο σύστημα του MAAN συνδυάζεται με την κατάλληλη συνάρτηση κατακερματισμού, τα δεδομένα κατανέμονται, επίσης, ομοιόμορφα ή σχεδόν ομοιόμορφα στους κόμβους. Στην αντίθετη περίπτωση, δηλαδή όταν η κατανομή των αναγνωριστικών των δεδομένων εισόδου δε συνδυάζεται με την κατάλληλη συνάρτηση κατακερματισμού, τα δεδομένα δεν κατανέμονται ομοιόμορφα στους κόμβους του συστήματος. Τέλος, παρατηρούμε πως είτε τα αναγνωριστικά των δεδομένων εισόδου ακολουθούν την ομοιόμορφη, είτε την εκθετική κατανομή, το σύστημα του TreeChord καταφέρνει να κατανείμει τα δεδομένα ομοιόμορφα ή σχεδόν ομοιόμορφα στους κόμβους.



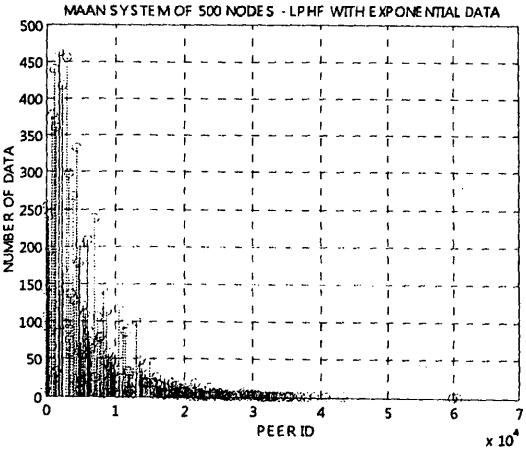
(α)



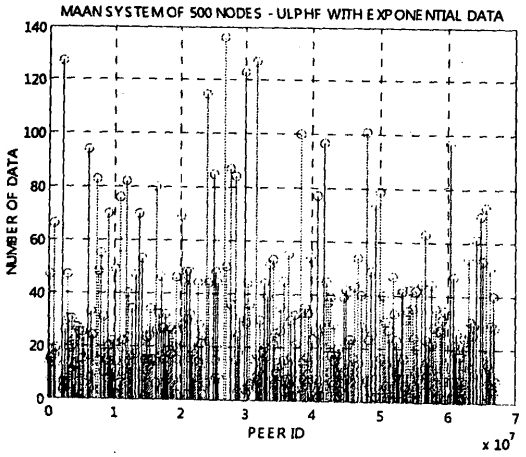
(ε)



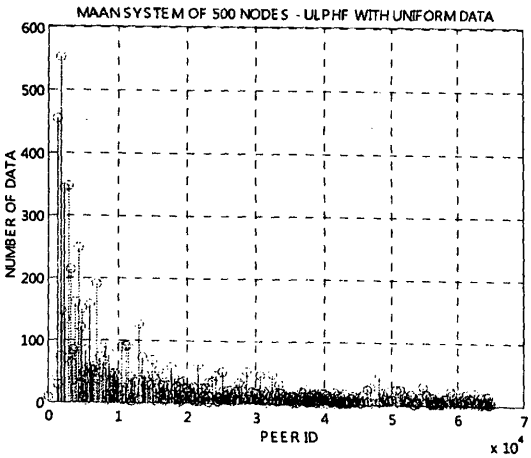
(β)



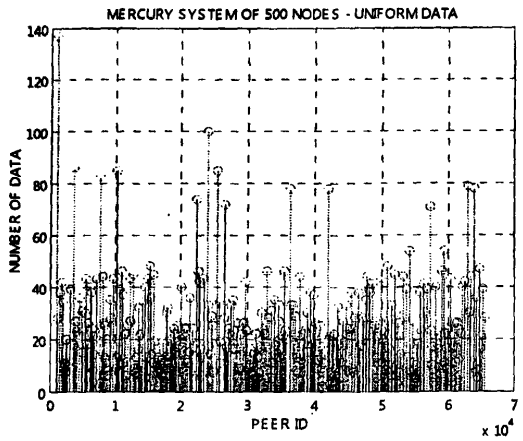
(ζ)



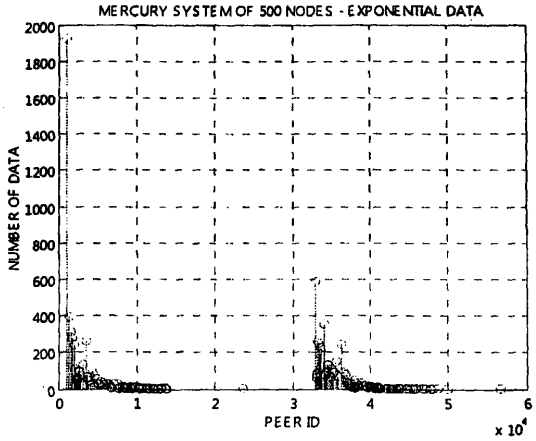
(γ)



(η)

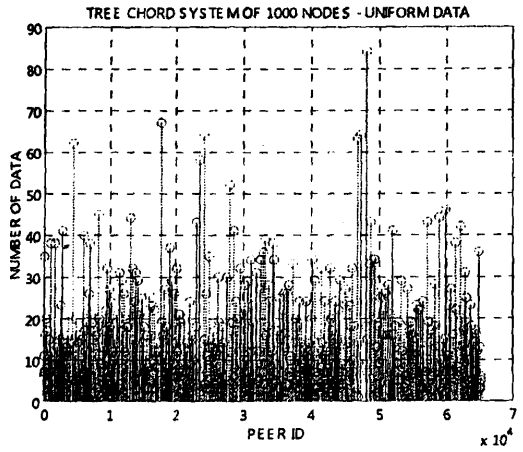


(δ)

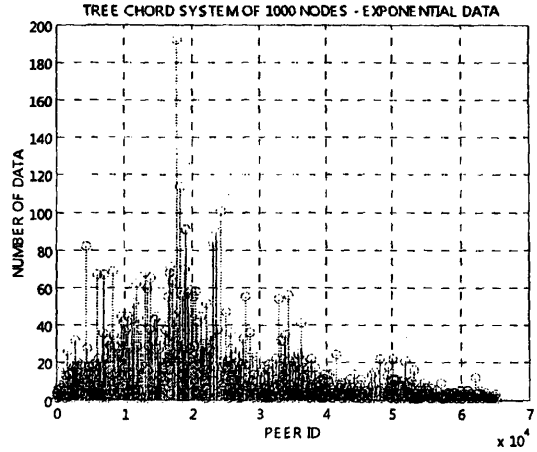


(θ)

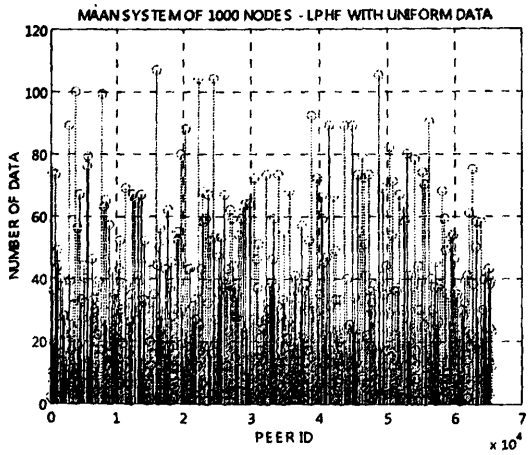
Σχήμα Α.1 Πλήθος Δεδομένων που Διατηρεί Κάθε Κόμβος σε Συστήματα με 500 Κόμβους.



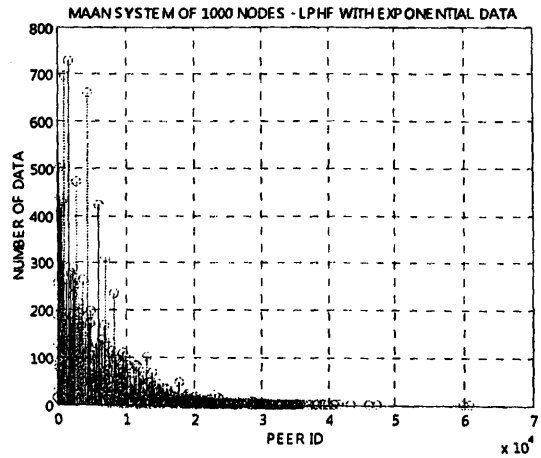
(α)



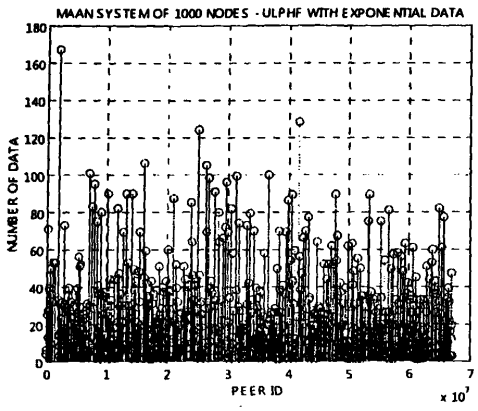
(ε)



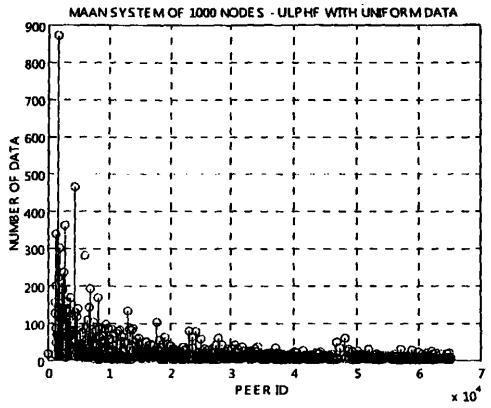
(β)



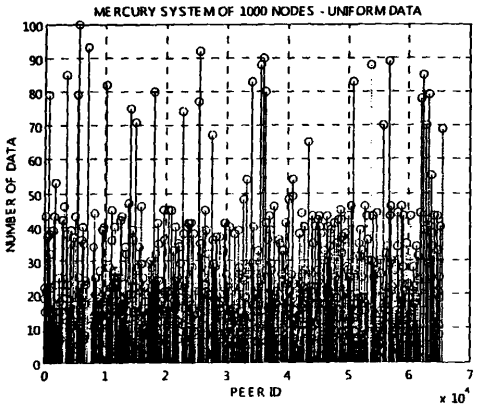
(ζ)



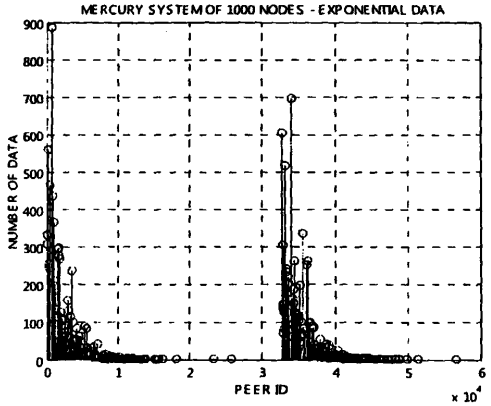
(γ)



(η)

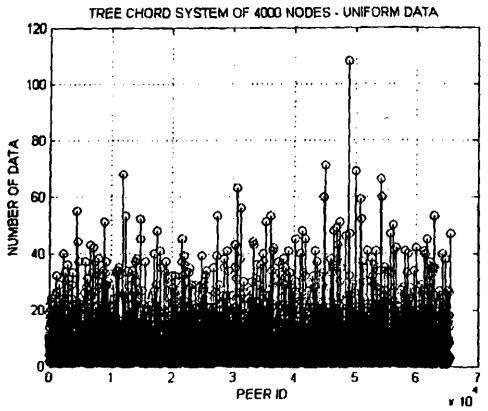


(δ)

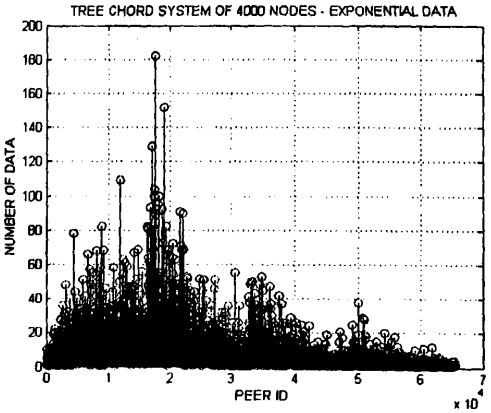


(θ)

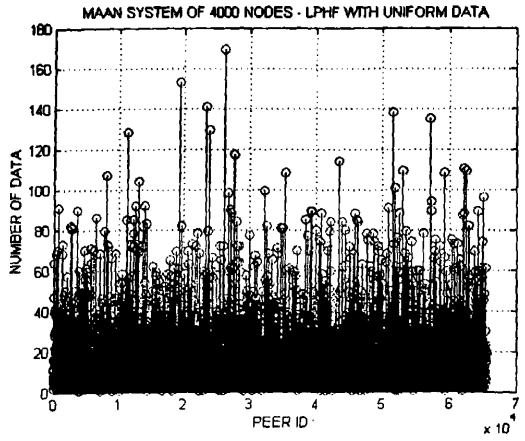
Σχήμα Α.2 Πλήθος Δεδομένων που Διατηρεί Κάθε Κόμβος σε Συστήματα με 1000 Κόμβους.



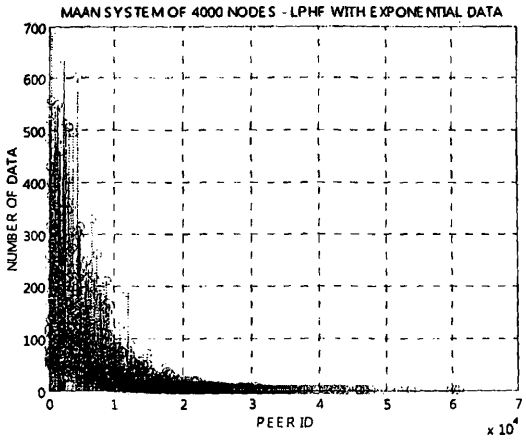
(α)



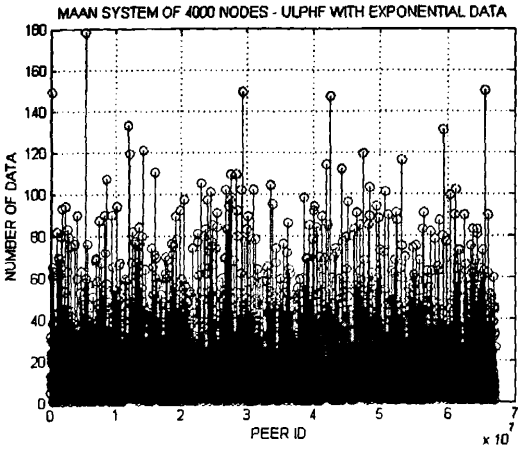
(ε)



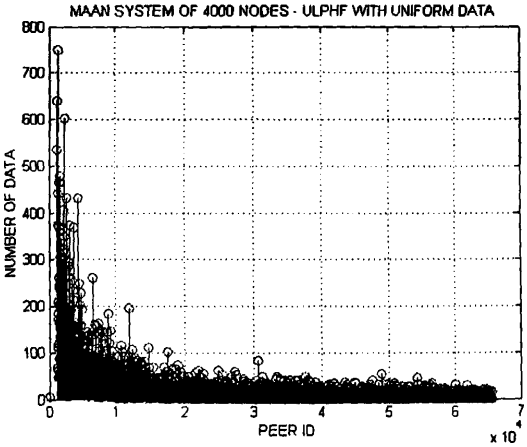
(β)



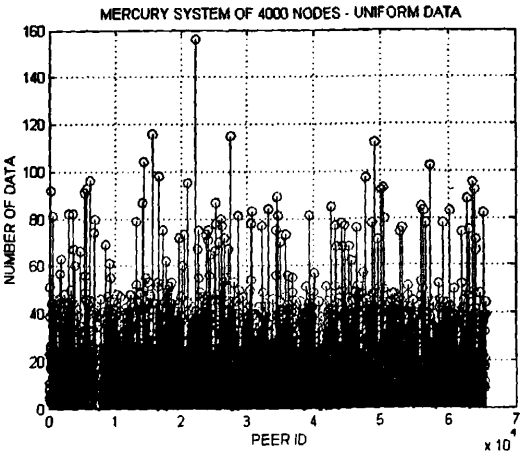
(ζ)



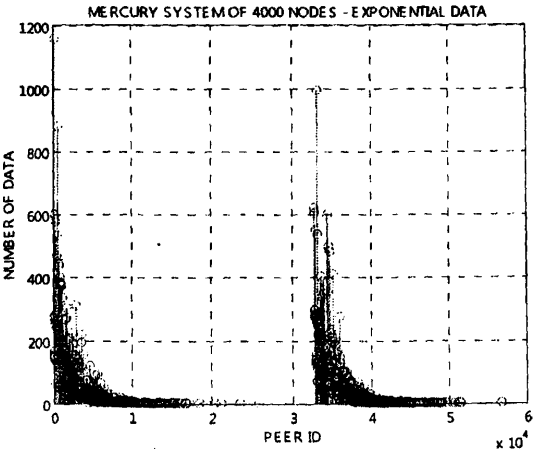
(γ)



(η)

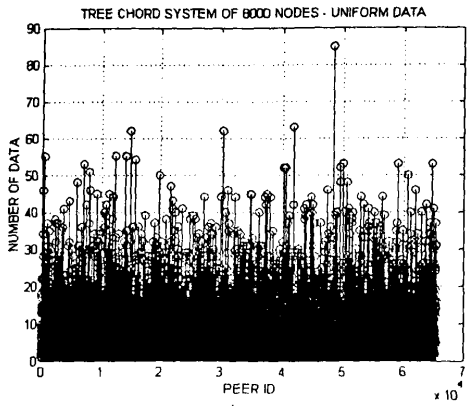


(δ)

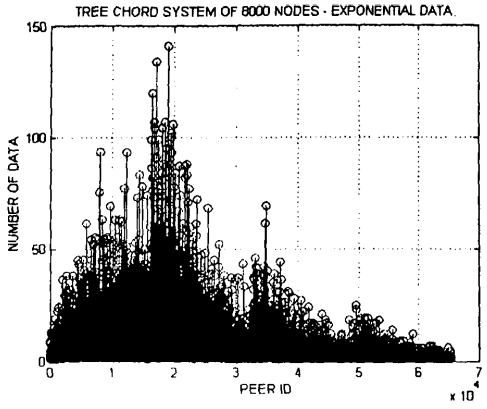


(θ)

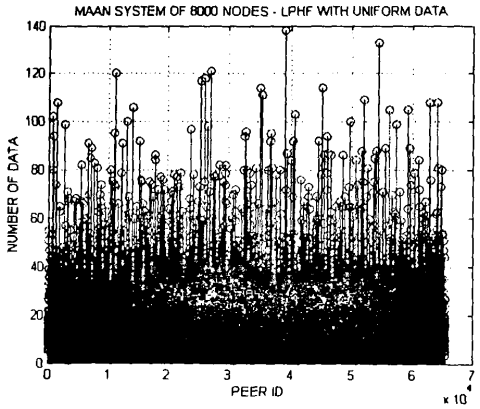
Σχήμα Α.3 Πλήθος Δεδομένων που Διατηρεί Κάθε Κόμβος σε Συστήματα με 4000 Κόμβους.



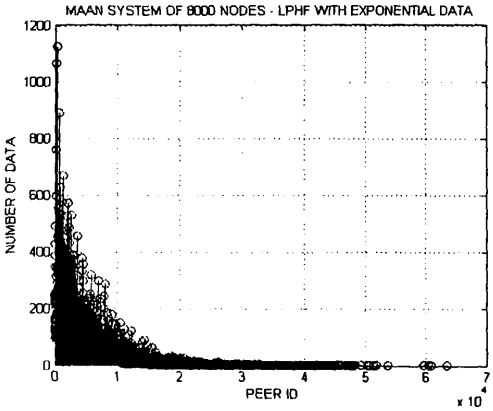
(α)



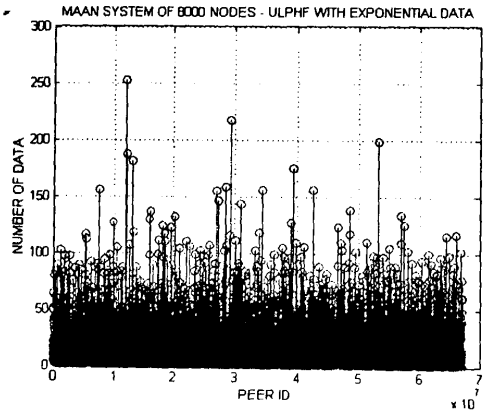
(ε)



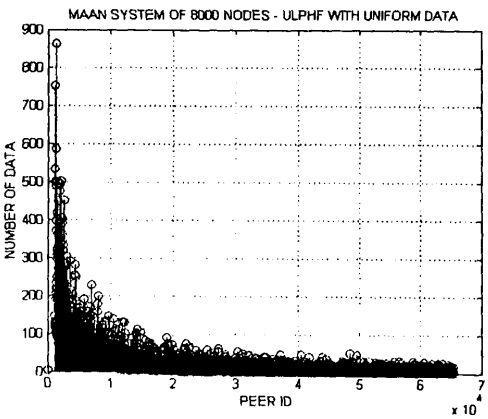
(β)



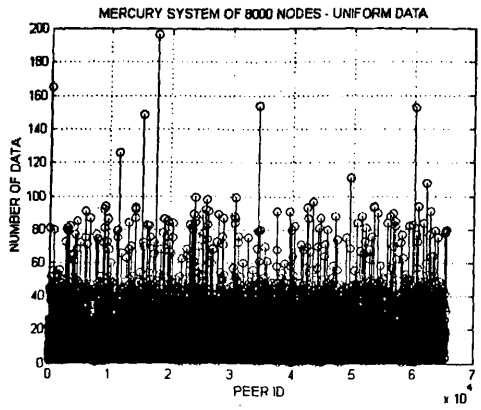
(ζ)



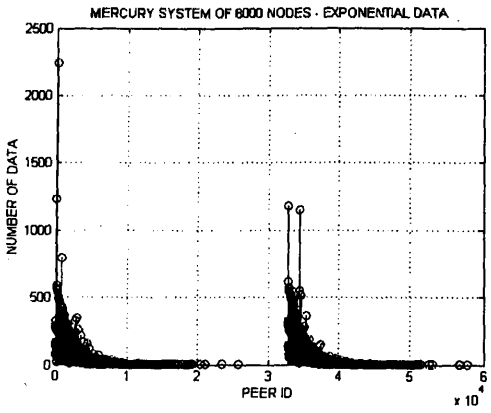
(γ)



(η)



(δ)

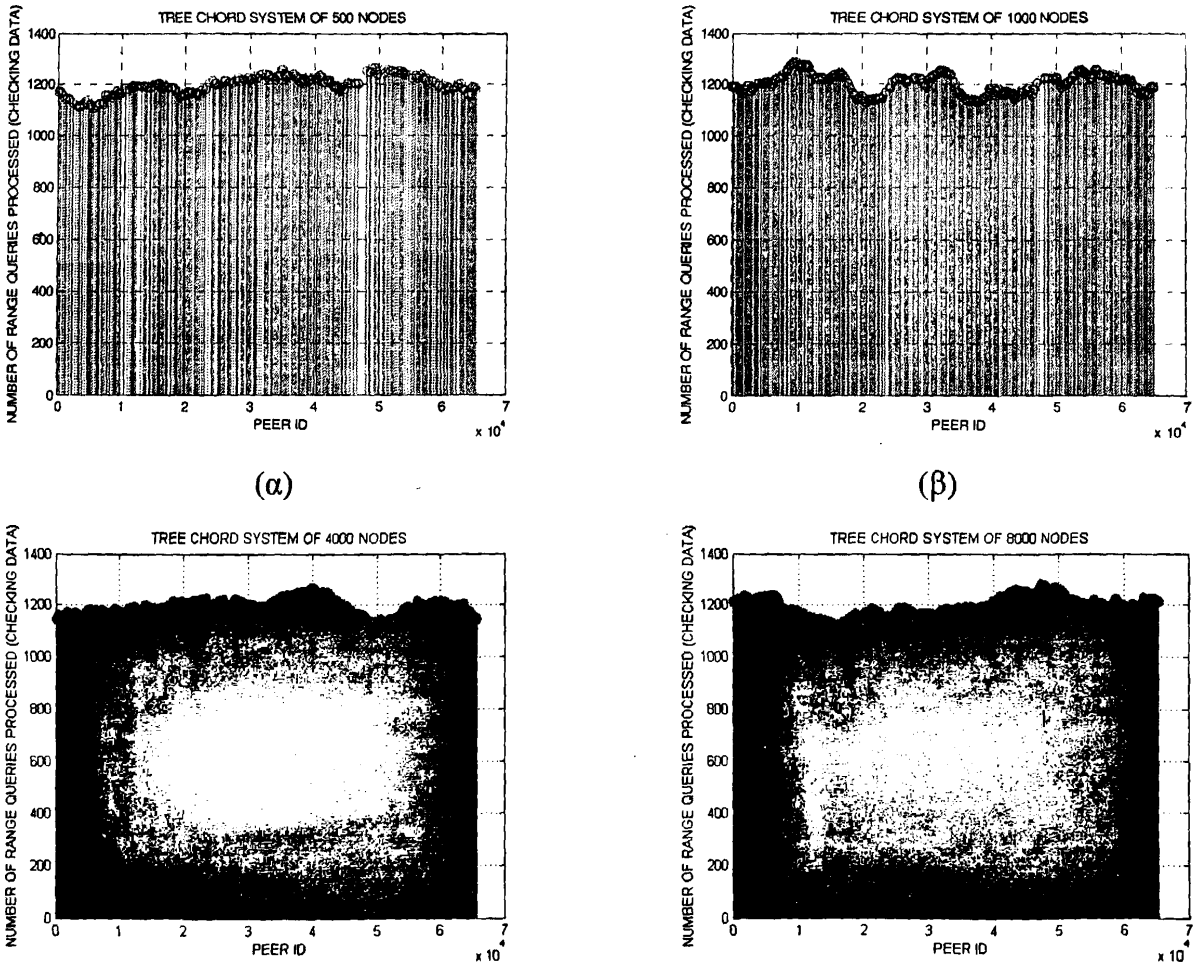


(θ)

Σχήμα Α.4 Πλήθος Δεδομένων που Διατηρεί Κάθε Κόμβος σε Συστήματα με 8000 Κόμβους.

ΠΑΡΑΡΤΗΜΑ Β

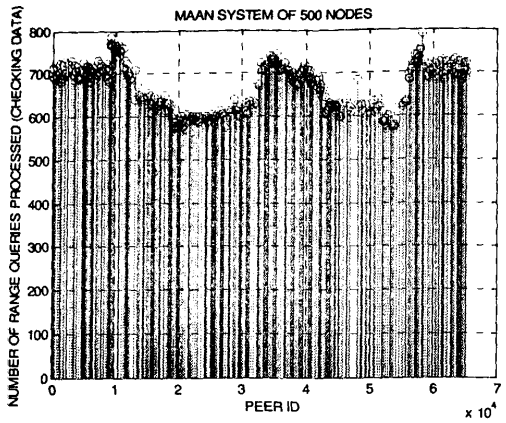
Επαναλαμβάνουμε τα πειράματα της Ενότητας 10.2.1. Στην περίπτωση αυτή, όμως, γραφικές παραστάσεις (Σχήμα Β1 – Σχήμα Β3) απεικονίζεται πόσες ερωτήσεις διαστήμ επεξεργάζεται κάθε κόμβος του συστήματος ελέγχοντας τα δεδομένα που διαθέτει το Και στην περίπτωση αυτή, οι διαφορές μεταξύ των συστημάτων δικαιολογούνται βάσει ά αναφέρθηκαν στην Ενότητα 10.2.1.



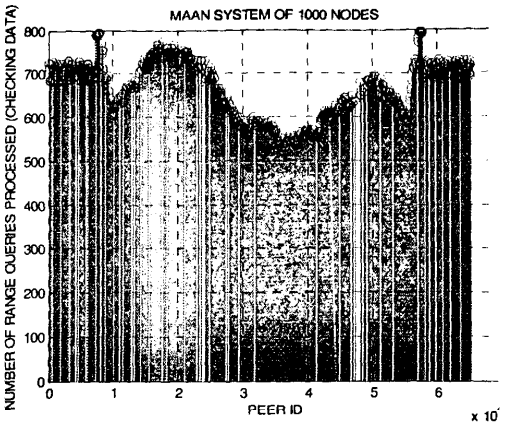
(α) (β)

(γ) (δ)

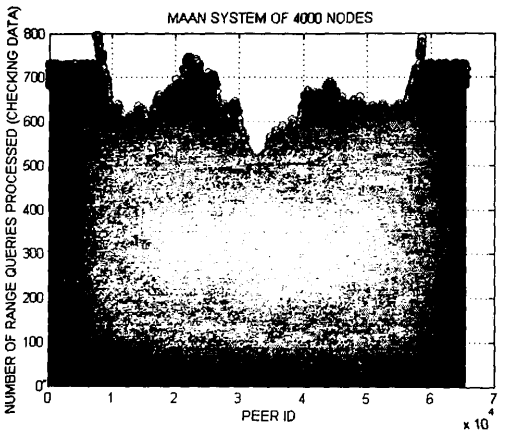
Σχήμα Β.1 Πλήθος Ερωτήσεων Διαστήματος στων Οποίων την Επίλυση Συμμετέχει Κάθε Κόμβος στο Σύστημα του TreeChord.



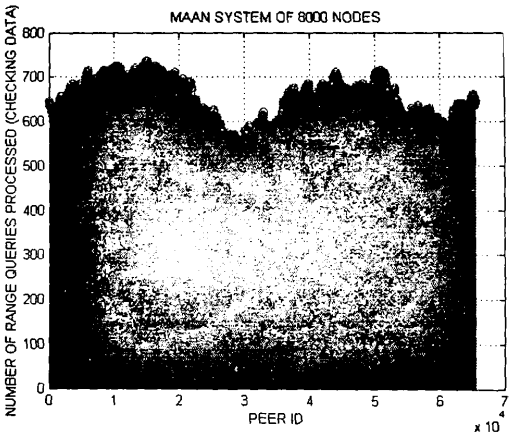
(α)



(β)

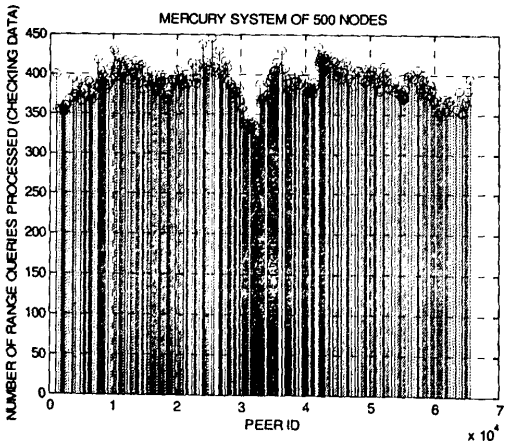


(γ)

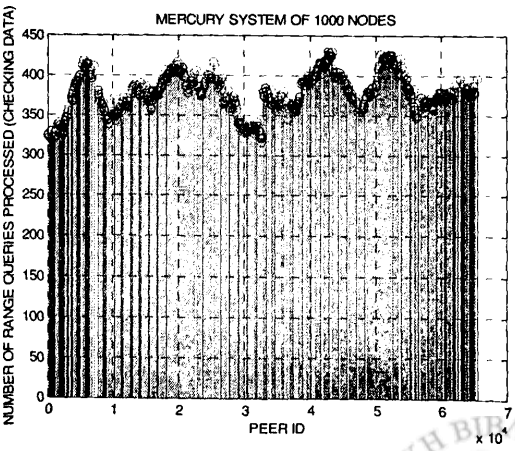


(δ)

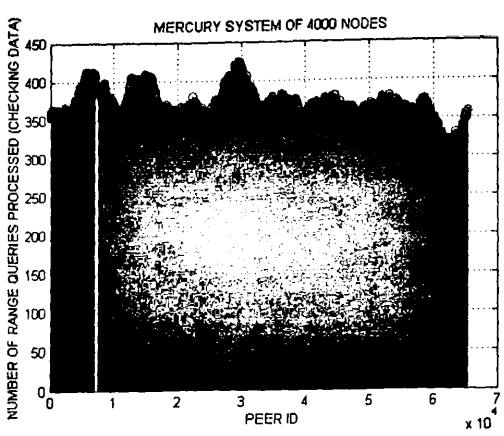
Σχήμα Β.2 Πλήθος Ερωτήσεων Διαστήματος στων Οποίων την Επίλυση Συμμετέχει Κάθ Κόμβος στο Σύστημα του MAAN.



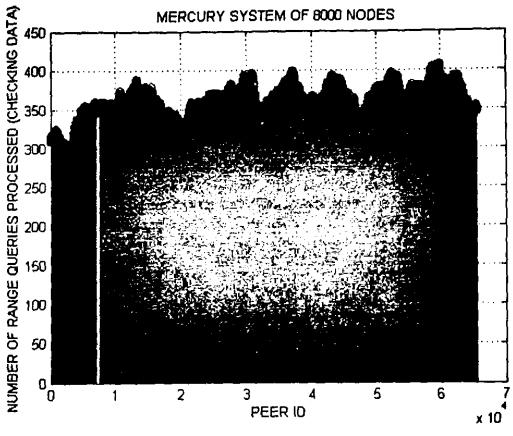
(α)



(β)



(γ)



(δ)

Σχήμα Β.3 Πλήθος Ερωτήσεων Διαστήματος στων Οποίων την Επίλυση Συμμετέχει Κάθε Κόμβος στο Σύστημα του Mercury.

ΠΑΡΑΡΤΗΜΑ Γ

Στα πειράματα αυτά εξετάζουμε μόνο τους αλγορίθμους επίλυσης ερωτήσεων διαστήματος που εφαρμόζονται στο σύστημα του MAAN. Πιο συγκεκριμένα, εξετάζουμε πόσα δεδομένα επιστρέφονται από κάθε ερώτηση διαστήματος (Σχήμα Γ1), πόσα από αυτά που επιστρέφονται είναι σωστά (Σχήμα Γ2), δηλαδή ικανοποιούν όλες τις συνθήκες της ερώτησης διαστήματος, και πόσα 'χάνονται' όταν εφαρμόζεται ο πρώτος και ο δεύτερος αλγόριθμος (Σχήμα Γ3). Κατά την εφαρμογή του δευτέρου αλγορίθμου, οι κόμβοι μπορούν να ακολουθούν τις δύο διαφορετικές προσεγγίσεις που προαναφέρθηκαν για να αποφασίζουν ποια δεδομένα πρέπει να επιστραφούν στον κόμβο που εκκινεί μία ερώτηση διαστήματος.

Πιο συγκεκριμένα, έστω K το πλήθος των ερωτήσεων που πραγματοποιούνται στο σύστημα. Ταξινομούμε τις ερωτήσεις βάσει του χρόνου κατά τον οποίο εκτελέστηκαν στο σύστημα και αντιστοιχίζουμε τη διάταξη που προέκυψε στις τιμές 1, 2, 3, ..., K. Αυτές οι τιμές θεωρούνται τα αναγνωριστικά των ερωτήσεων (*query IDs*). Στη συνέχεια, παρουσιάζουμε στις γραφικές παραστάσεις πόσα δεδομένα επιστρέφονται από κάθε ερώτηση διαστήματος, πόσα από αυτά που επιστρέφονται είναι σωστά, δηλαδή ικανοποιούν όλες τις συνθήκες της ερώτησης διαστήματος και πόσα 'χάνονται'.

Μελετώνται τα συστήματα που περιγράφηκαν στις προηγούμενες ενότητες. Στα συστήματα αυτά πραγματοποιούνται 1000 ερωτήσεις διαστήματος με επιλεκτικότητα 15% σε κάθε διάσταση. Σημειώνεται ότι τα αναγνωριστικά των δεδομένων εισόδου ακολουθούν την εκθετική κατανομή και, επομένως, εφαρμόζεται η ομοιόμορφη συνάρτηση κατακερματισμού που διατηρεί τη διάταξη και περιγράφεται στην Εξίσωση (4.2), προκειμένου να γίνει καλός διαμοιρασμός των δεδομένων ανάμεσα στους κόμβους κάθε συστήματος.

Όπως προαναφέρθηκε, οι κόμβοι του συστήματος έχουν τοποθετηθεί ομοιόμορφα στο δακτύλιο. Επίσης, η συνάρτηση κατακερματισμού που χρησιμοποιείται κατανέμει ομοιόμορφα τα δεδομένα σε κάθε διάσταση στο δακτύλιο. Όταν, λοιπόν, εφαρμόζεται ο πρώτος αλγόριθμος επίλυσης ερωτήσεων διαστήματος, εκτελείται μία ερώτηση σε κάθε μία

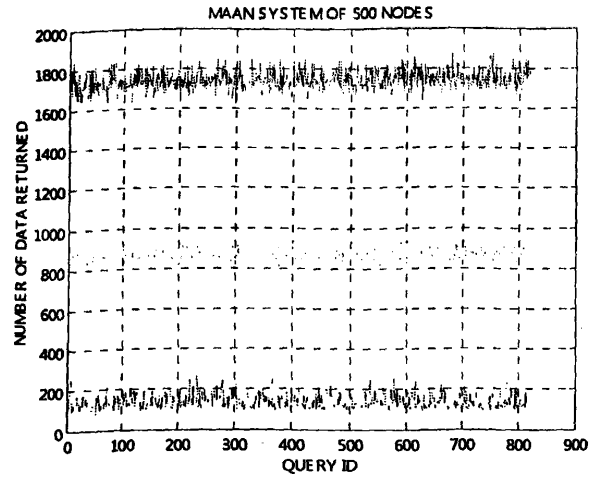
από τις διαστάσεις του συστήματος και ελέγχεται μία μόνο συνθήκη. Επομένως, λόγω της ομοιόμορφης κατανομής των δεδομένων επιστρέφονται $\sum_{i=1}^{M}(|data|\times|s_i|)$ δεδομένα. Όταν εφαρμόζεται ο δεύτερος αλγόριθμος επίλυσης ερωτήσεων διαστήματος και ακολουθείται η πρώτη προσέγγιση για να αποφασίσει κάθε κόμβος ποια δεδομένα πρέπει να επιστρέψει, εκτελείται μία ερώτηση σε μία μόνο διάσταση, έστω k, και ελέγχεται μόνο η συνθήκη που ορίζεται στην ερώτηση για τη συγκεκριμένη διάσταση. Επομένως, λόγω της ομοιόμορφης κατανομής των δεδομένων επιστρέφονται $|data|\times|s_k|$ δεδομένα. Όταν, όμως, εφαρμόζεται ο δεύτερος αλγόριθμος επίλυσης ερωτήσεων διαστήματος και ακολουθείται η δεύτερη προσέγγιση για να αποφασίσει κάθε κόμβος ποια δεδομένα πρέπει να επιστρέψει, εκτελείται και πάλι μία ερώτηση σε μία μόνο διάσταση, έστω k, αλλά ελέγχονται όλες οι συνθήκες που ορίζονται στην ερώτηση, δηλαδή μία συνθήκη για κάθε διάσταση των δεδομένων του συστήματος. Επομένως, λόγω της ομοιόμορφης κατανομής των δεδομένων, αλλά και του πλήθους των διαστάσεων των δεδομένων, επιστρέφονται $|data|\times\prod_{i=1}^{M}|s_i|$ δεδομένα. Αυτό οφείλεται στο γεγονός πως ενώ $|data|\times|s_1|$ δεδομένα του δακτυλίου ικανοποιούν μόνο τη συνθήκη της $1^{ης}$ διάστασης, από αυτά μόνο $|data|\times|s_1|\times|s_2|$ ικανοποιούν και τη συνθήκη της $2^{ης}$ διάστασης, από αυτά μόνο $|data|\times|s_1|\times|s_2|\times|s_3|$ ικανοποιούν και τη συνθήκη της $3^{ης}$ διάστασης, …, και από αυτά μόνο $|data|\times|s_1|\times|s_2|\times|s_3|\times\ldots\times|s_M|$ ικανοποιούν και τη συνθήκη της $M^{ης}$ διάστασης, δηλαδή όλες τις συνθήκες που ορίζονται σε κάθε ερώτηση διαστήματος, τα οποία και επιστρέφονται τελικά από τον αλγόριθμο.

Παρατηρούμε, λοιπόν, πως σε κάθε περίπτωση, ο πρώτος αλγόριθμος επιστρέφει σχεδόν διπλάσιο πλήθος δεδομένων από αυτό που επιστρέφει ο δεύτερος αλγόριθμος που ακολουθεί την πρώτη προσέγγιση, αφού τα δεδομένα που εισάγονται στο σύστημα είναι δύο διαστάσεων και η επιλεκτικότητα για τις δύο διαστάσεις είναι ίδια. Επίσης, ο δεύτερος αλγόριθμος που ακολουθεί τη δεύτερη προσέγγιση επιστρέφει πολύ λιγότερα δεδομένα και από τον πρώτο αλγόριθμο, αλλά και από το δεύτερο αλγόριθμο που ακολουθεί την πρώτη προσέγγιση, αφού κατά την εκτέλεση μίας ερώτησης σε μία διάσταση εξετάζονται οι συνθήκες σε όλες τις διαστάσεις.

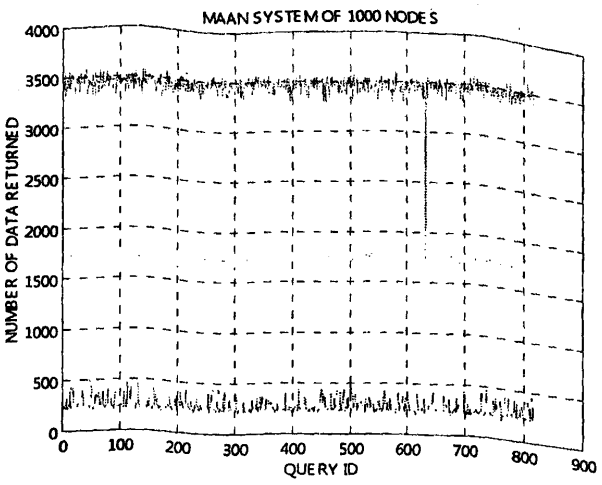
Από την άλλη, παρατηρούμε πως το πλήθος των δεδομένων που τελικά ικανοποιούν όλες τις συνθήκες μίας ερώτησης διαστήματος δεν εμφανίζει διαφορά είτε ακολουθείται η πρώτη, είτε η δεύτερη προσέγγιση του δεύτερου αλγορίθμου επίλυσης ερωτήσεων διαστήματος. Αυτό συμβαίνει καθώς μία ερώτηση εκτελείται με τον ίδιο τρόπο και στην ίδια διάσταση. Απλώς στην πρώτη περίπτωση μπορούν να θεωρηθούν κάποια δεδομένα έγκυρα, ενώ δεν ικανοποιούν όλες τις συνθήκες της ερώτησης. Επίσης, παρατηρούμε πως όταν εφαρμόζεται ο πρώτος αλγόριθμος, το πλήθος των έγκυρων δεδομένων που επιστρέφονται είναι κατά πολύ μεγαλύτερο από το πλήθος των δεδομένων που επιστρέφεται στις άλλες δύο περιπτώσεις. Αυτό οφείλεται στο γεγονός πως ενώ κατά την εφαρμογή του πρώτου αλγορίθμου επίλυσης ερωτήσεων διαστήματος πραγματοποιείται μία ερώτηση σε κάθε μία από τις διαστάσεις των δεδομένων του συστήματος, κατά την εφαρμογή του δευτέρου αλγορίθμου πραγματοποιείται μία μόνο ερώτηση σε κάποια από τις διαστάσεις, ενώ στο σύστημα υπάρχουν και αντίγραφα των δεδομένων (ένα για κάθε διάσταση) τα οποία βρίσκει ο πρώτος αλγόριθμος και επιστρέφει.

Τέλος, παρατηρούμε ότι ενώ ο πρώτος αλγόριθμος δεν αποτυγχάνει να βρει κάποιο έγκυρο δεδομένο που υπάρχει στο σύστημα και, επομένως, πρέπει να επιστραφεί, όταν εφαρμόζονται οι άλλες δύο προσεγγίσεις, αρκετά έγκυρα δεδομένα δεν ευρίσκονται για να επιστραφούν. Το μεγάλο πλήθος των έγκυρων δεδομένων που δεν επιστρέφει ο δεύτερος αλγόριθμος οφείλεται στο γεγονός πως στο σύστημα υπάρχει ένα αντίγραφο κάθε δεδομένου για κάθε μία διάσταση, τα οποία ο πρώτος αλγόριθμος επιτυγχάνει να βρει. Η ικανότητα του πρώτου αλγορίθμου να βρίσκει όλα τα δεδομένα που ικανοποιούν μία ερώτηση διαστήματος που πραγματοποιείται στο σύστημα οφείλεται στο γεγονός πως εκτελείται μία ερώτηση σε κάθε μία από τις διαστάσεις των δεδομένων και, επομένως, αν υπάρχει κάποιο έγκυρο δεδομένο στο σύστημα, αυτό σίγουρα θα επιστραφεί.

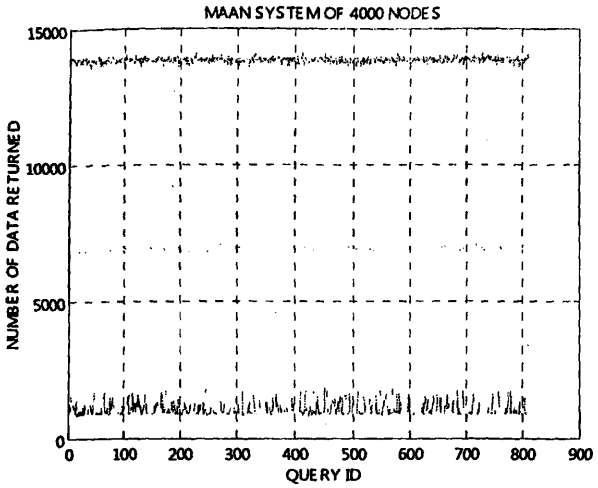
ALGORITHM 1
ALGORITHM 2 (APPROACH 1)
ALGORITHM 2 (APPROACH 2)



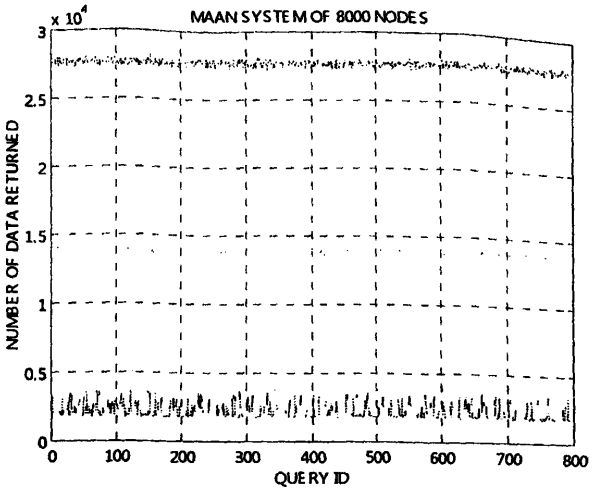
(α)



(γ)

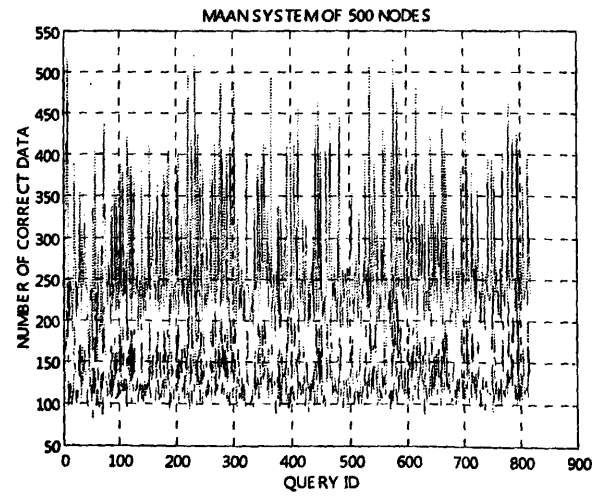


(γ)

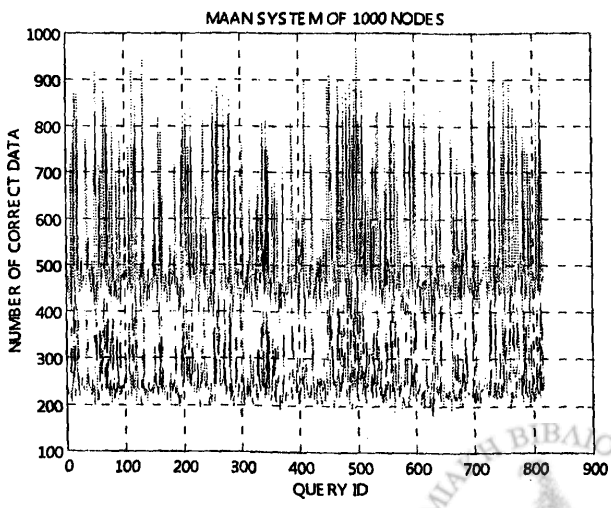


(δ)

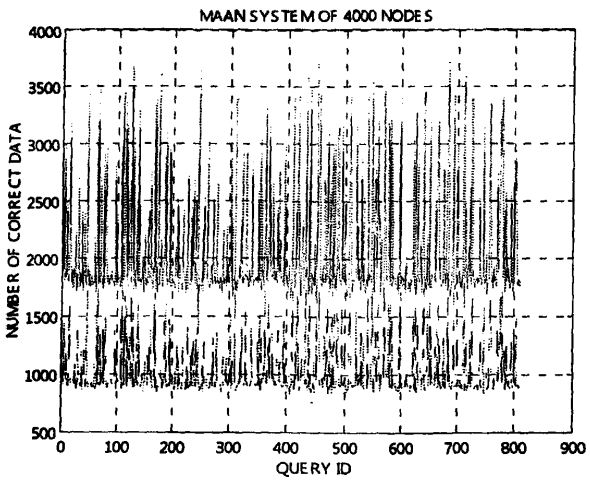
Σχήμα Γ.1 Πλήθος Δεδομένων που Επιστρέφονται από Κάθε Ερώτηση Διαστήματος που Εφαρμόζεται στο Σύστημα του MAAN.



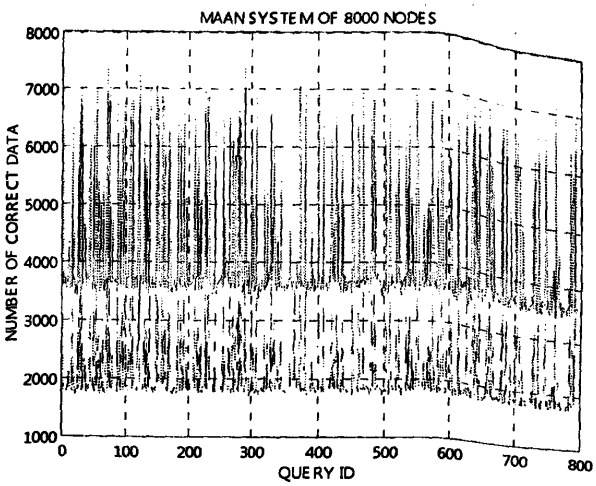
(α)



(β)

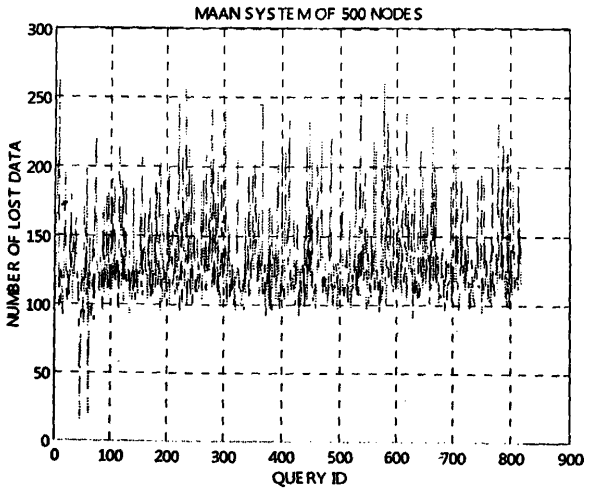


(γ)

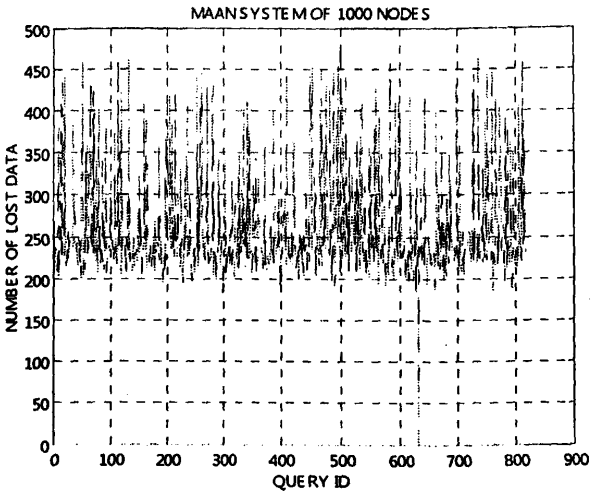


(δ)

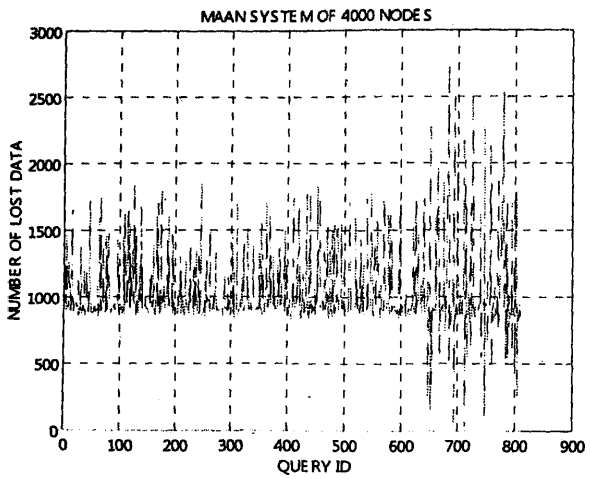
Σχήμα Γ.2 Πλήθος Σωστών Δεδομένων που Επιστρέφονται από Κάθε Ερώτηση Διαστήματος που Εφαρμόζεται στο Σύστημα του MAAN.



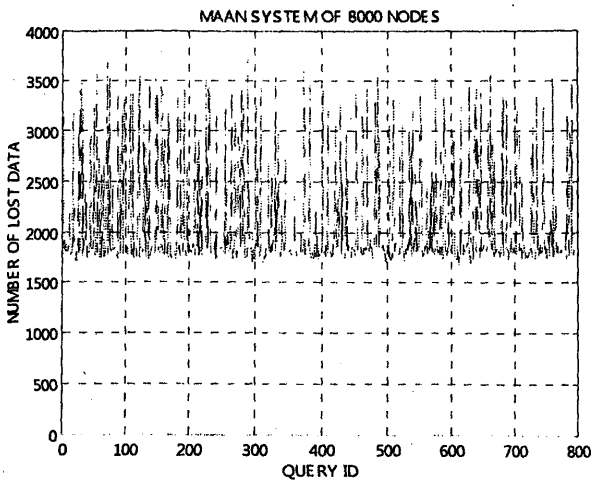
(α)



(β)



(γ)



(δ)

Σχήμα Γ.3 Πλήθος Δεδομένων που Χάνονται από Κάθε Ερώτηση Διαστήματος που Εφαρμόζεται στο Σύστημα του MAAN.

ΣΥΝΤΟΜΟ ΒΙΟΓΡΑΦΙΚΟ

Η Χριστοδουλίδου Μαρία εισήχθη στο Τμήμα Πληροφορικής του Πανεπιστημίου Ιωαννίνων το 2000 και έλαβε το Πτυχίο Πληροφορικής το 2005. Από το Φεβρουάριο του 2005 είναι Μεταπτυχιακή φοιτήτρια στο Τμήμα Πληροφορικής του Πανεπιστημίου Ιωαννίνων.

Στα επιστημονικά της ενδιαφέροντα συμπεριλαμβάνονται τα Κατανεμημένα Συστήματα, τα Λειτουργικά Συστήματα και τα Δίκτυα Υπολογιστών.