Τμημα Μηχανικων Ηλεκτρονικων Υπολογιστων
και Πληροφορικης

Πανεπιστημιο Πατρων

Διπλωματικη Εργασια

Συλλογή Δεδομένων σε Κινητά Δίκτυα και Ασύρματα Δίκτυα Αισθητήρων

Συγγραφέας

Μιχαήλ Π. Ρεσβάνης

Επιβλέπων

Καθ. Παύλος Σπυράκης

Συνεπιβλέπων

Δρ. Ιωάννης Χατζηγιαννάκης

Πατρα, 2009

Περιεχόμενα

1 Εισαγωγή **11**

1.1 Διαδίκτυο και Ευρυζωνικότητα 11

1.2 Παγκόσμιος Ιστός 13

1.3 Ψηφιακή Εποχή και Γιγαντιαίος Όγκος Πληροφορίας 14

1.4 Ασύρματα Δίκτυα Αισθητήρων 16

1.5 Κινητά Δίκτυα 17

1.6 Δίκτυα Ομοτίμων 18

1.7 Περιρρέουσα Νοημοσύνη 19

1.8 Περίγραμμα Διπλωματικής 20

2 Συνάθροιση Δεδομένων **21**

2.1 Ορισμός 21

2.2 Ενδοδικτυακή Συνάθροιση Δεδομένων και Δίκτυα Αισθητήρων 21

2.3 Πρόβλημα Μέτρησης Διαφορετικών Τιμών 23

2.3.1 Αλγόριθμοι Ροής 23

2.3.2 Συχνοτικές Ροπές 24

2.3.3 Ορισμός Προβλήματος Μέτρησης Διαφορετικών Τιμών 25

2.4 Αλγόριθμοι για Μέτρηση Διαφορετικών Τιμών 26

2.4.1 Αλγόριθμοι Δειγματοληψίας 26

2.4.2 Αλγόριθμοι Ροής 27

2.5 Probabilistic Counting with Stochastic Averaging (PCSA) 27

2.5.1 Πιθανοτική Μέθοδος Μέτρησης και η Ανάλυσή της 28

2.5.2 Probabilistic Counting Αλγόριθμος 36

2.6 Loglog Counting 39

2.6.1 Ο Βασικός Loglog Counting Αλγόριθμος 39

2.6.2 Απαιτήσεις Χώρου 42

2.7 Time-Decaying Sketches 43

2.7.1 Διατύπωση Προβλήματος 44

2.7.2 Συναθροιστικές Συναρτήσεις 46

2.7.3 Συναθροιστικές Συναρτήσεις με Ακέραια Συνάρτηση Φθοράς 49

2.7.4 Φθορά Κυλιόμενου Παραθύρου 58

3 Υλοποίηση και Προετοιμασία Πειραμάτων 61

3.1 Ανάπτυξη και Επαλήθευση Πηγαίου Κώδικα 61

3.2 Ειδικά Ζητήματα Υλοποίησης 65

3.3 Πειραματική Διαδικασία 70

3.3.1 Συσκευές και Περιβάλλοντα Εκτέλεσης 70

3.3.2 Περιγραφή της Εισόδου 77

3.3.3 Επιλογή Παραμέτρων των Αλγορίθμων 78

4 Πειραματική Αξιολόγηση Αλγορίθμων **81**

4.1 Πλήρης Παρουσίαση Γραφικών Παραστάσεων 81

4.1.1 iSense Core Module 81

4.1.2 Stargate NetBridge Gateway 83

4.1.3 OpenMoko Neo Freerunner 92

4.1.4 PC Engine Alix 101

4.2 Μέγεθος Εισόδου και Μετρήσεις Χρόνου 110

4.3 Σχετικό Σφάλμα και Μέγεθος Εισόδου 111

4.4 Εισάγοντας Διπλότυπα 112

4.4.1 Χρονική Αξιολόγηση 112

4.4.2 Αξιολόγηση Ακρίβειας 112

5 Συμπεράσματα και Μελλοντικές Κατευθύνσεις **115**

5.1 Συμπεράσματα 115

5.2 Μελλοντικές Κατευθύνσεις 116

Κατάλογος Σχημάτων

1.1 Παρατηρείστε ότι ο παγκόσμιος μέσος όρος αγγίζει το 24% 13

1.2 Το πλήθος των ιστιοσελίδων που έχουν καταγραφεί και δεικτοδοτούνται από την Google 14

1.3 Ψηφιακές πωλήσεις μουσικής και ποσοστό αύξησης αυτών για το χρονικό διάστημα $2004 - 2008$ 15

2.1 PCSA αλγόριθμος 38

2.2 Παράδειγμα με 8 στοιχεία εισόδου και το sketch της ροής S_0, S_1, S_2, S_3 για το decayed sum. Η παρούσα χρονική στιγμή είναι 15 και τα decayed weights του στοιχείου e_i τη χρονική στιγμή t συμβολίζονται με ω_i^t. Το στοιχείο e_4 στο διακεκομμένο κουτί υποδηλώνει ότι απορρίφθηκε από το S_0 εξαιτίας υπερχείλισης. 54

2.3 Γενικό sketch του αλγορίθμου με ακέραια συνάρτηση μεταφοράς. 55

2.4 Decayed sum για συνάρτηση φθοράς κυλιόμενου παραθύρου. 57

2.5 Selectivity estimation για ακέραια συνάρτηση φθοράς. 57

3.1 OpenMoko Neo Freerunner 71

3.2 Crossbow Stargate Netbridge Gateway 73

3.3 PC Engine Alix, alix1d 74

3.4 iSense Core Module 75

4.1 Μετρήσεις χρόνου σε είσοδο χωρίς διπλότυπα στο **Stargate Netbridge** 83

4.2 Μετρήσεις χρόνου σε είσοδο χωρίς διπλότυπα στο **Stargate Netbridge** 84

4.3 Μετρήσεις χρόνου σε είσοδο με 40% διπλότυπα στο **Stargate Netbridge** . . . 84

4.4 Μετρήσεις χρόνου σε είσοδο με 40% διπλότυπα στο **Stargate Netbridge** . . . 85

4.5 Χρονική απόδοση σε σχέση με ποσοστό διπλότυπων στο **Stargate Netbridge** . 86

4.6 Χρονική απόδοση σε σχέση με ποσοστό διπλότυπων στο **Stargate Netbridge** . 87

4.7 Χρονική απόδοση σε σχέση με ποσοστό διπλότυπων στο **Stargate Netbridge** . 87

4.8 Υπολογισμός σχετικού σφάλματος σε είσοδο χωρίς διπλότυπα στο **Stargate Netbridge** 88

4.9 Υπολογισμός σχετικού σφάλματος σε είσοδο χωρίς διπλότυπα στο **Stargate Netbridge** 89

4.10 Υπολογισμός σχετικού σφάλματος σε είσοδο χωρίς διπλότυπα στο **Stargate Netbridge** 89

4.11 Υπολογισμός σχετικού σφάλματος σε είσοδο με 40% διπλότυπα στο **Stargate Netbridge** 90

4.12 Υπολογισμός σχετικού σφάλματος σε είσοδο με 40% διπλότυπα στο **Stargate Netbridge** 90

4.13 Υπολογισμός σχετικού σφάλματος σε είσοδο με 40% διπλότυπα στο **Stargate Netbridge** 91

4.14 Μετρήσεις χρόνου σε είσοδο χωρίς διπλότυπα στο **OpenMoko Neo Freerunner** 92

4.15 Μετρήσεις χρόνου σε είσοδο χωρίς διπλότυπα στο **OpenMoko Neo Freerunner** 93

4.16 Μετρήσεις χρόνου σε είσοδο με 40% διπλότυπα στο **OpenMoko Neo Freerunner** 93

4.17 Μετρήσεις χρόνου σε είσοδο με 40% διπλότυπα στο **OpenMoko Neo Freerunner** 94

4.18 Συμπεριφορά αλγορίθμου σε σχέση με ποσοστό διπλότυπων στο **OpenMoko Neo Freerunner** 95

4.19 Συμπεριφορά αλγορίθμου σε σχέση με ποσοστό διπλότυπων στο **OpenMoko Neo Freerunner** 96

4.20 Συμπεριφορά αλγορίθμου σε σχέση με ποσοστό διπλότυπων στο **OpenMoko Neo Freerunner** 96

4.21 Υπολογισμός σχετικού σφάλματος σε είσοδο χωρίς διπλότυπα στο **OpenMoko Neo Freerunner** 97

4.22 Υπολογισμός σχετικού σφάλματος σε είσοδο χωρίς διπλότυπα στο **OpenMoko Neo Freerunner** 98

4.23 Υπολογισμός σχετικού σφάλματος σε είσοδο χωρίς διπλότυπα στο **OpenMoko Neo Freerunner** 98

4.24 Υπολογισμός σχετικού σφάλματος σε είσοδο με 40% διπλότυπα στο **OpenMoko Neo Freerunner** 99

4.25 Υπολογισμός σχετικού σφάλματος σε είσοδο με 40% διπλότυπα στο **OpenMoko Neo Freerunner** 99

4.26 Υπολογισμός σχετικού σφάλματος σε είσοδο με 40% διπλότυπα στο **OpenMoko Neo Freerunner** 100

4.27 Μετρήσεις χρόνου σε είσοδο χωρίς διπλότυπα στο **PC Engine Alix** 101

4.28 Μετρήσεις χρόνου σε είσοδο χωρίς διπλότυπα στο **PC Engine Alix** 102

4.29 Μετρήσεις χρόνου σε είσοδο με 40% διπλότυπα στο **PC Engine Alix** 102

4.30 Μετρήσεις χρόνου σε είσοδο με 40% διπλότυπα στο **PC Engine Alix** 103

4.31 Συμπεριφορά αλγορίθμου σε σχέση με ποσοστό διπλότυπων στο **PC Engine Alix** 104

4.32 Συμπεριφορά αλγορίθμου σε σχέση με ποσοστό διπλότυπων στο **PC Engine Alix** 105

4.33 Συμπεριφορά αλγορίθμου σε σχέση με ποσοστό διπλότυπων στο **PC Engine Alix** 105

4.34 Υπολογισμός σχετικού σφάλματος σε είσοδο χωρίς διπλότυπα στο **PC Engine Alix** 106

4.35 Υπολογισμός σχετικού σφάλματος σε είσοδο χωρίς διπλότυπα στο **PC Engine Alix** 107

4.36 Υπολογισμός σχετικού σφάλματος σε είσοδο χωρίς διπλότυπα στο **PC Engine Alix** 107

4.37 Υπολογισμός σχετικού σφάλματος σε είσοδο με 40% διπλότυπα στο **PC Engine Alix** 108

4.38 Υπολογισμός σχετικού σφάλματος σε είσοδο με 40% διπλότυπα στο **PC Engine Alix** 108

4.39 Υπολογισμός σχετικού σφάλματος σε είσοδο με 40% διπλότυπα στο **PC Engine Alix** 109

Κεφάλαιο 1

Εισαγωγή

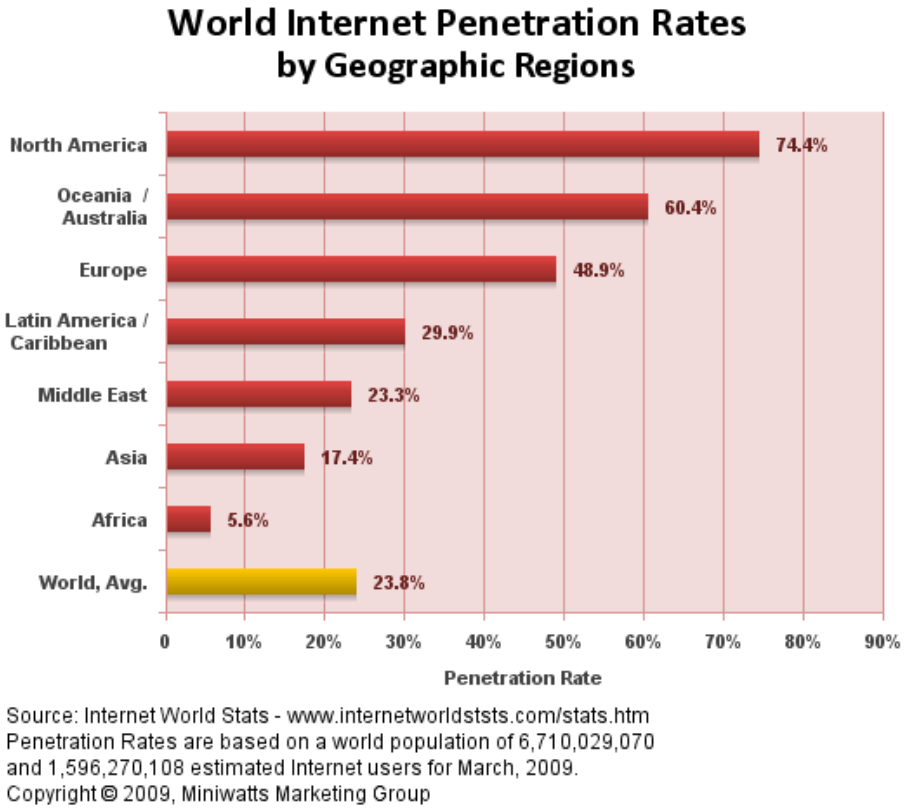
Βρισκόμαστε στην παρυφή μια νέας εποχής, η τεχνολογία της πληροφορίας και των επικοινωνιών αποτελεί την αιχμή του δόρατος για την παγκόσμια οικονομία και ανάπτυξη, τον πολιτισμό, τις μορφές κοινωνικής αλληλεπίδρασης και επαγωγικά της καθημερινής ζωής. Η σημερινή κοινωνία χαρακτηρίζεται ως η κοινωνία της πληροφορίας και όχι άδικα, τα θεμέλιά της εξαρτώνται από τη δημιουργία, κατανομή, διάχυση και χρησιμοποίηση του γιγαντιαίου διαθέσιμου όγκου πληροφορίας. Η τεχνολογία της πληροφορίας και των επικοινωνιών γνωρίζει αντίστοιχα ταχύτατη ανάπτυξη και συνεχώς "ψάχνει" καινούριους τρόπους διαχείρισης της πληροφορίας με σκοπό την δημιουργία αποδοτικότερων εφαρμογών και εργαλείων αποθήκευσης, διαμοιρασμού και επεξεργασίας της.

1.1 Διαδίκτυο και Ευρυζωνικότητα

Καθημερινά, ως χρήστες διάφορων συσκευών και υπηρεσιών, λαμβάνουμε μέρος στη διακίνηση μεγάλου όγκου πληροφοριών, χωρίς καν πολλές φορές να το γνωρίζουμε. Η απαρχή του φαινομένου αυτού προσδιορίζεται στην ευρύτατη χρήση *κινητών τηλεφώνων*, ο αριθμός των οποίων εκτιμάται ότι έφτασε τα *4.1 δισεκατομμύρια* στο τέλος του *2008*. Οι υπολογιστικές δυνατότητες των μικρών αυτών συσκευών έχουν γνωρίσει ταχύτατη ανάπτυξη τα τελευταία χρόνια, συνδυάζοντας πολλαπλές λειτουργίες, εκτός από αυτές τις κινητής τηλεφωνίας. Αρκεί

να αναλογιστούμε ότι πλέον έχουμε στη διάθεσή μας λειτουργίες όπως πρόσβαση στο διαδίκτυο, λήψη φωτογραφιών, λήψη και αναπαραγωγή video υψηλής ανάλυσης, αναπαραγωγή και εγγραφή μουσικής, χρήση κινητών ασύρματων πρωτοκόλλων (πχ. Bluetooth), και όλα αυτά με σχετικά μικρό κόστος για τις υπηρεσίες που προσφέρουν λόγω της τεράστιας ανάπτυξης της αντίστοιχης βιομηχανίας και αγοράς. Το εύρος των ασύρματων υπολογιστικών και επικοινωνιακών συσκευών είναι πλέον κάτι περισσότερο από τεράστιο, φορητοί προσωπικοί υπολογιστές (laptops, notebooks), προσωπικοί ψηφιακοί βοηθοί (PDAs), βομβητές, smartphones, carputers, Ultra-Mobile PCs και wearable computers είναι μόνο λίγες από τις συσκεύες που είναι διαθέσιμες σε ανθρώπους κάθε κοινωνικού επιπέδου. Η ασύρματη δικτύωση, είτε αναφερόμαστε σε εξωτερικούς είτε σε εσωτερικούς χώρους, παρέχει αμέτρητες δυνατότητες επέκτασης και εκμετάλλευσης των ήδη υπαρχουσών εφαρμογών από χρήστες οπουδήποτε και αν βρίσκονται, ακόμα και εν κινήσει. Δεν υπάρχει πλέον ο περιορισμός των ενσύρματων υποδομών, δεν χρειάζεται κάποιος να βρίσκεται στο χώρο εργασίας ή στο σπίτι του για να ελέγξει το ηλεκτρονικό του ταχυδρομείο, μπορεί κάλλιστα να ταξιδεύει οδικώς και μέσω του κινητού τηλεφώνου που υποστηρίζει υπηρεσία *Mobile Internet* να έχει πρόσβαση από οπουδήποτε οποτεδήποτε.

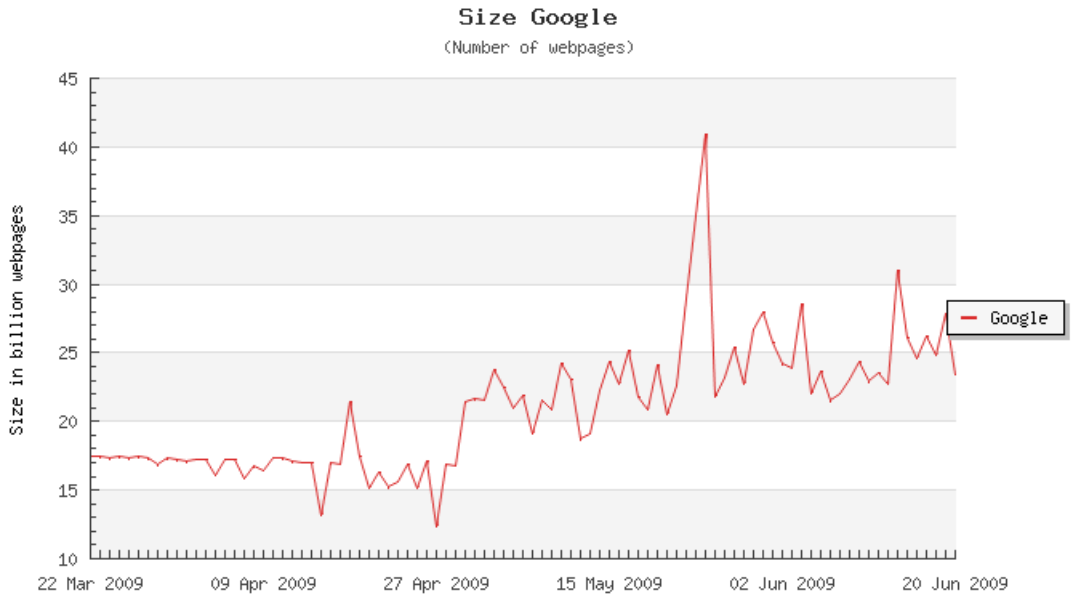
Από την άλλη μεριά, η *ευρυζωνικότητα* έχει καταστεί η αιχμή του δόρατος για την *τεχνολογία της πληροφορίας*. Η ευρυζωνική δικτύωση στο σπίτι είναι πλέον αυτονόητη και προσιτή σε όλους. Μαζί με την τελευταία, το διαδίκτυο, αυτό το παγκόσμιο φαινόμενο, μέσω και της ανάπτυξης των ασύρματων δικτύων, έχει διαδοθεί σε κάθε μέρος της γης, με εκατομμύρια χρηστών να εκμεταλλεύονται τους αμέτρητους διαθέσιμους πόρους του σε καθημερινή βάση. Η *Advanced Micro Devices* (AMD) εταιρία υποστηρίζει ότι τον Ιανουάριο του 2009 οι χρήστες του διαδικτύου ξεπέρασαν το 1.5 δισεκατομμύριο. Ένα κατατοπιστικό γράφημα είναι το παρακάτω, σχήμα 1.1, όπου δίνεται το ποσοστό της εισχώρησης της διαδικτυακής πρόσβασης ανά γεωγραφικές περιοχές. Μπορούμε να δούμε ότι ο παγκόσμιος μέσος όρος, ο οποίος αγγίζει το 24%, είναι μεγάλο νούμερο αναλογιζόμενοι ότι αντιστοιχεί σε περίπου 1.44 διεκατομμύριο χρήστες.



Σχήμα 1.1: Παρατηρείστε ότι ο παγκόσμιος μέσος όρος αγγίζει το 24%

1.2 Παγκόσμιος Ιστός

Αποτέλεσμα της επέκτασης της ευρυζωνικότητας και της ασύρματης δικτύωσης είναι και η ανάπτυξη του *Παγκόσμιου Ιστού* (World Wide Web). Ο αριθμός των ιστιοσελίδων (websites) έχει αυξηθεί σημαντικά τα τελευταία χρόνια, παρέχοντας πληροφορίες και υπηρεσίες για οτιδήποτε μπορεί να φανταστεί ο ανθρώπινος νους. Αυτό φαίνεται και από τη δημοσίευση της πιο ελκυστικής σε όρους περιβάλλοντος εργασίας αμερικανικής εταιρίας πληροφορικής, της *Google* στο σχήμα 1.2. Παρατηρούμε ότι *το πλήθος των ιστιοσελίδων* που η συγκεκριμένη εταιρία δεικτοδοτεί μέσω της μηχανής αναζήτησης παγκοσμίου ιστού, είναι τεράστιο, φτάνει στα μέσα του 2009 τα **29.62 δισεκατομμύρια**. Προσθέστε σε αυτόν τον αριθμό και τις σελίδες που δεν δεικτοδοτούνται και προσπαθήστε να φανταστείτε την ποσότητα πληροφορίας που είναι διαθέσιμη ανά πάσα στιγμή, σε οποιοδήποτε μέρος, σε οποιονδήποτε χρήστη. Ο όγκος πληροφοριών που διακινούνται μέσω του διαδικτύου είναι απλά ασύλληπτος.



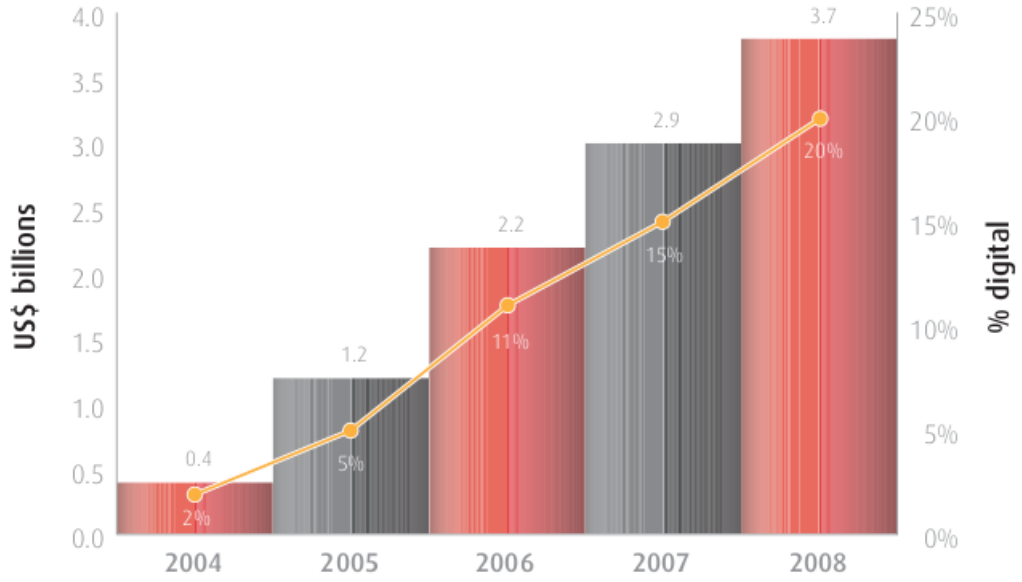
Σχήμα 1.2: Το πλήθος των ιστιοσελίδων που έχουν καταγραφεί και δεικτοδοτούνται από την Google

Την ανάπτυξη του παγκόσμιου ιστού με άξονα μέτρησης τον αριθμό των ενεργών ιστιοσελίδων, μπορούμε να επιδείξουμε και με την εγκαθίδρυση του *IPv6* (Internet Protocol version 6), αφού η προηγούμενη έκδοση του συγκεκριμένου πρωτοκόλλου διαδικτυακής διευθυνσιοδότησης *IPv4* έχει φτάσει στα όρια εξάντλησης διαθέσιμων διευθύνσεων. Το πιο επιθυμητό χαρακτηριστικό του IPv6 είναι ότι υποστηρίζει διευθύνσεις των 128 δυαδικών ψηφίων, που σημαίνει $2^{128} \approx 3.4 \times 10^{38}$ διευθύνσεις, νούμερο που φυσικά είναι τεράστιο για τις σημερινές διαστάσεις, όμως υποδηλώνει αυτά που θα ακολουθήσουν στο μέλλον.

1.3 Ψηφιακή Εποχή και Γιγαντιαίος Όγκος Πληροφορίας

Αφού λοιπόν, αναλογιστούμε τα προηγούμενα ας περάσουμε στις ποσότητες ψηφιακών δεδομένων τις οποίες καθημερινά προσπαθούμε να δαμάσουμε. Η *Ψηφιακή Επανάσταση* έχει ξεκινήσει ήδη από το 1980 και συνεχίζεται μέχρι σήμερα με τρομερούς ρυθμούς. Τα δεδομένα παντός είδους τείνουν να μετατρέπονται από αναλογικά σε ψηφιακά, φωτογραφίες, μουσική, ταινίες, έγγραφα, βιβλία, είναι λίγα παραδείγματα από τα πιο χρησιμοποιούμενα ψηφιακά

αντικείμενα, με τα οποία ερχόμαστε καθημερινά σε επαφή. Ο σκοπός της ψηφιοποίησης είναι συνήθως, η εύκολη διαχείριση, αποθήκευση, μεταφορά και διαμοίραση όλων αυτών των δεδομένων. Σήμερα, κάποιος που διαθέτει έναν εξωτερικό σκληρό δίσκο μεγέθους 500 gigabytes μπορεί να έχει ανά πάσα στιγμή μαζί του περίπου 8500 ώρες μουσικής ή 700 ταινίες ή 6000 ηλεκτρονικά βιβλία. Και σκεφτείτε πόσο εύκολο είναι σήμερα να έχεις πρόσβαση σε ψηφιακά δεδομένα μέσω του διαδικτύου, πχ. μέσω των εφαρμογών διαμοίρασης αρχείων των *δικτύων ομοτίμων* (Peer-to-Peer Networks), ώστε να έχεις στην κατοχή σου τεράστιο όγκο ψηφιακών δεδομένων σε ένα μόνο μικρό συρτάρι ή ράφι βιβλιοθήκης. Στο γράφημα 1.3 της *Διεθνούς Ομοσπουδίας Φωνογραφικής Βιομηχανίας* (IFPI), μπορούμε να δούμε τα έσοδα από τις ψηφιακές πωλήσεις μουσικής, αλλά και την αύξηση των τελευταίων στο διάστημα $2004 - 2008$. Αν συμπεριλάβουμε και την παράνομη διαμοίραση αρχείων που είναι και το μεγαλύτερο ποσοστό διακίνησης ψηφιακών δεδομένων, μπορούμε να υποθέσουμε τις τάξεις μεγέθους της ποσότητας των ψηφιακών αρχείων που διακινούνται μέσω του διαδικτύου.



Σχήμα 1.3: Ψηφιακές πωλήσεις μουσικής και ποσοστό αύξησης αυτών για το χρονικό διάστημα $2004 - 2008$

Οι απαιτήσεις της σημερινής εποχής για αποδοτικότερη αποθήκευση, χρήση και ε-

κμετάλλευση του γιγαντιαίου όγκου πληροφοριών οδηγούν την επιστήμη των υπολογιστών και της πληροφορικής σε εναλλακτικά μονοπάτια συλλογής δεδομένων. Τα παρόντα μέσα συγκέντρωσης δεδομένων, αν και αποδοτικά για το σταθερής υποδομής υπολογιστικό και επικοινωνιακό υλικό, κρίνονται ως ακατάλληλα για την χρησιμοποίηση τους στις νέες τεχνολογίες, οι οποίες κάνουν τα πρώτα τους βήματα στη σημερινή κοινωνία της πληροφορίας. Μερικές από τις τελευταίες είναι οι παρακάτω:

- *Ασύρματα Δίκτυα Αισθητήρων* (Wireless Sensor Networks)
- *Κινητά Δίκτυα Επικοινωνιών* (Mobile Communication Networks)
- *Δίκτυα Ομοτίμων* (Peer-to-Peer Networks)
- *Περιρρέουσα Νοημοσύνη* (Ubiquitous Computing)

1.4 Ασύρματα Δίκτυα Αισθητήρων

Η ερευνητική κοινότητα θεωρεί τα Ασύρματα Δίκτυα Αισθητήρων ως ένα *''συναρπαστικό αναδυόμενο τομέα δικτυακών συστημάτων αποτελούμενα από πολύ μικρής ισχύος ασύρματους κόμβους με μικροσκοπική ποσότητα επεξεργαστικής δύναμης (CPU) και μνήμης, που έχουν ώς σκοπό την υψηλής - ανάλυσης αίσθηση του περιβάλλοντος ''*. Αρκεί να αναφέρουμε ότι τα δίκτυα αισθητήρων είναι μια πρώτη μορφή της περιρρέουσας νοημοσύνης, αφού σε αυτά χρησιμοποιούνται μικρές, "έξυπνες" και φθηνές συσκευές που μπορούν να επιτηρούν και να ελέγχουν με αδιαμφισβήτητη αποδοτικότητα και ακρίβεια. Οι αισθητήρες εντός ασύρματων δικτύων αισθητήρων έχουν ποικίλους στόχους, λειτουργίες και δυνατότητες, γεγονός που επιβεβαιώνεται ολοένα και περισσότερο τα τελευταία χρόνια από τον ερευνητικό αλλά και βιομηχανικό/εμπορικό κόσμο.

Αυτά τα δίκτυα χρίζουν πολύ διαφορετικής προσέγγισης από ότι ένα γενικού σκοπού κατανεμημένο ή κινητό δίκτυο, εξαιτίας του μεγάλου αριθμού των κόμβων του δικτύου που μπορούν να φτάσουν την τάξη αρκετών χιλιάδων κόμβων και των φυσικών περιορισμών σε υπολογιστική ισχύ και μνήμη των συσκευών που τα απαρτίζουν. Η ποσότητα των δεδομένων

που πρέπει να είναι σε θέση να διαχειριστεί ένα ασύρματο δίκτυο αισθητήρων είναι τεράστια, και οι υπάρχουσες μέθοδοι συγκέντρωσης δεδομένων δανεισμένες από άλλα ερευνητικά πεδία, όπως αυτό των βάσεων δεδομένων, δεν μπορούν να προσαρμοστούν αποδοτικά για χρήση τους σε τέτοια δίκτυα. Αλλά ακόμα και δίκτυα αισθητήρων με πολύ λίγους κόμβους, πχ. των 100 κόμβων, μπορεί ανάλογα με την εφαρμογή να συγκεντρώνουν τεράστιο αριθμό μετρήσεων και ενδείξεων, αφού μπορεί για παράδειγμα να αναφέρουν ενδείξεις κάθε λεπτό ή κάθε 10 δευτερόλεπτα ή και σε πολύ μικρότερα χρονικά διαστήματα κατά τη διάρκεια μιας ολόκληρης ημέρας. Οι μέθοδοι συλλογής δεδομένων στα δίκτυα αισθητήρων είναι σημαντικό ερευνητικό θέμα και ο προσδιορισμός καινούριων και ευπροσάρμοστων επιλύσεων σε αυτό το πρόβλημα μπορεί να αλλάξει το γενικότερο σκηνικό στον τομέα των δικτύων αυτών.

1.5 Κινητά Δίκτυα

Επιβεβαιώνοντας τις προβλέψεις πολλών ειδικών, τα *Κινητά Δίκτυα* (Mobile Networks), μια σημαντική συνέπεια της ανάπτυξης της ασύρματης δικτύωσης, χρησιμοποιούνται σήμερα κατά κόρον από ανθρώπους διαφόρων ηλικιών, οικονομικής κατάστασης και τρόπου ζωής. Κάποιες από τις σημαντικότερες κατηγορίες κινητών δικτύων είναι τα τηλεφωνικά και δορυφορικά δίκτυα, τα δίκτυα μετάδοσης ραδιοπακέτων ή ad-hoc δίκτυα και τα δίκτυα σελιδοποίησης (paging networks), τα οποία χρησιμοποιούνται για τη διαχείρηση βομβητών. Γενικότερα, η έννοια της *κινητικότητας* (mobility) ενός δικτύου μπορεί να συνδυαστεί με πολλά είδη δικτύων, όπως για παράδειγμα τα κινητά δίκτυα αισθητήρων, στα οποία κάποιοι ή όλοι οι κόμβοι του δικτύου έχουν τη δυνατότητα της κίνησης, είτε αυτοματοποιημένα είτε με τον έλεγχο του χρήστη.

Η κίνηση των κόμβων, όπως είναι λογικό επιφέρει δυσκολίες και προβλήματα, τα οποία δεν παρουσιάζονται σε άλλα δίκτυα, όπως για παράδειγμα την αποστολή περισσότερων μηνυμάτων στον ίδιο κόμβο λόγω σύγχυσης του αποστολέα θεωρώντας ότι διαφορετικές συντεταγμένες συνεπάγονται και διαφορετικούς κόμβους. Το φαινόμενο αυτό οδηγεί σε διπλότυπα πακέτα δεδομένων, τα οποία μπορεί να αυξάνουν την ανεκτικότητα σε σφάλματα επικοινωνίας, ιδιότητα πολύ χρήσιμη στα κινητά δίκτυα λόγω της φύσης τους (παρεμπόδιση μονοπατιών,

αυξημένος θόρυβος, αλλαγές περιοχής κάλυψης, κτλ), αλλά μειώνουν την απόδοση λόγω του αυξημένου πλήθους μηνυμάτων και δυσκολεύουν την παραγωγή έγκυρων συμπερασμάτων, δεδομένου των διπλότυπων πληροφοριών. Επιπλέον, σκεφτείτε το πλήθος και τις πληροφορίες που μεταφέρονται μέσω κινητών δικτύων. Ήδη από το 2003, έχει εισαχθεί στον τομέα των κινητών τηλεπικοινωνιών το *τρίτης γενιάς* (3G) πρότυπο, το οποίο υποστηρίζει μετάδοση δεδομένων πολυμέσων σε πολύ υψηλές ταχύτητες (μέχρι **14.4 Mbits/sec** downlink και **5.8 Mbits/sec** uplink με υποστήριξη *HSPA+*), δίνοντας έτσι τη δυνατότητα δημιουργίας πιο προχωρημένων εφαρμογών εκμεταλλεύοντας τη μεγαλύτερη *δικτυακή χωρητικότητα* και τη βελτιωμένη *φασματική απόδοση* (spectral efficiency). Επομένως, αντιλαμβανόμαστε ότι οι τεχνικές συλλογής και συσσώρευσης δεδομένων είναι κεντρικό σημείο έρευνας και ανάπτυξης σε δίκτυα που εμπεριέχουν την υποστήριξη οποιουδήποτε τρόπου κινητικότητας.

1.6 Δίκτυα Ομοτίμων

Τα δίκτυα ομοτίμων εδώ και κάποια χρόνια βρίσκονται στο προσκήνιο της διαμοίρασης υπολογιστικών πόρων και δεδομένων. Αυτό που πολλοί χρησιμοποιούμε και όλοι γνωρίζουμε είναι η ανταλλαγή ή διαμοίραση δεδομένων, όπως μουσική, ταινίες, ηλεκτρονικά βιβλία σε ψηφιακή μορφή, μέσω διάφορων πρωτοκόλλων, *BitTorrent*, *DC++*, *Limewire* είναι μόνο λίγα από τα πιο δημοφιλή. Όμως, χρησιμοποιούνται και για την εξισορρόπηση υπολογιστικού φόρτου σε περιπτώσεις εφαρμογών που χρειάζονται τεράστια υπολογιστική δύναμη, όπως εφαρμογές μοριακής βιολογίας, μοντελοποίησης ανθρώπινων οργάνων και οικονομικών σεναρίων.

Σήμερα, τα δίκτυα ομοτίμων χρησιμοποιούνται εκτός από γενικότερη διαμοίραση ψηφιακών δεδομένων και για πραγματικού-χρόνου εφαρμογές, όπως για παράδειγμα η τηλεφωνία (πχ. Voice over IP). Άλλες σημαντικές εφαρμογές περιλαμβάνουν :

- Διαχείριση αποθήκευσης δεδομένων
- Instant messaging, πχ. *AOL Instant Message* (AIM)

- Collaborative content hosting
- Publish-subscribe systems

Κάποιος μπορεί πολύ εύκολα να αναλογιστεί το μέγεθος του ποσού των πληροφοριών που βρίσκονται εντός ενός δικτύου ομοτίμων καθημερινά. Η κλιμακωσιμότητα σε πλήθος κόμβων είναι ένα από τα θεμελιώδη χαρακτηριστικά των δικτύων αυτών, και για αυτόν ακριβώς το λόγο η χρησιμοποίηση τους στο έπακρο αποτελεί επιτακτική ανάγκη. Όσο όμως αυξάνεται ο αριθμός των χρηστών ενός τέτοιου δικτύου τόσο αυξάνεται και η ανάγκη για αποδοτική και αξιόπιστη συγκέντρωση δεδομένων εξαιτίας της *εκθετικής* αύξησης του όγκου των τελευταίων.

1.7 Περιρρέουσα Νοημοσύνη

Η περιρρέουσα νοημοσύνη βρίσκεται με τη μία ή την άλλη μορφή (Internet of Things, Διάχυτα Συστήματα, Everyware) στο επίκεντρο των μελλοντικών υπολογιστικών, πληροφοριακών και επικοινωνιακών τεχνολογιών. Σκεφτείτε τους παραπάνω τεχνολογικούς τομείς να συνδυάζονται και θα καταλάβετε τη διαίσθηση πίσω από την περιρρέουσα νοημοσύνη. Αποτελεί την τεχνολογική επανάσταση που αντιπροσωπεύει το μέλλον των υπολογιστών και των επικοινωνιών, και η ανάπτυξή του εξαρτάται από καινοτόμες τεχνικές σε ένα πλήθος σημαντικών πεδίων, από τη *επιστήμη των υλικών* μέχρι την *νανοτεχνολογία*. Το πιο σημαντικό χαρακτηριστικό της είναι ότι σε αυτό το πολύπλευρο, πολύπλοκο σχήμα υποστηρίζεται διαφάνεια, ώστε ο χρήστης να μην αναλαμβάνει καμία ευθύνη, να μην γνωρίζει πληροφορίες που δεν χρειάζεται, να επικεντρώνεται στις ανθρώπινες ανάγκες και να είναι σίγουρος ότι το υλικό και λογισμικό είναι ικανά να πραγματοποιούν τις απαραίτητες ρυθμίσεις για την καλύτερη προσαρμογή τους ανάλογα με το περιβάλλον και επαγωγικά για την αποδοτικότερη κάλυψη των αναγκών του.

Στην περίπτωση της περιρρέουσας νοημοσύνης το βάρος του τεράστιου ποσού των πληροφοριών που είμαστε αναγκασμένοι να κουβαλάμε και να επεξεργαζόμαστε ασταμάτητα εξαλείφεται στα μάτια μας, ενώ ουσιαστικά έχει περάσει στις μηχανές που όμως δεν μας έχουν

ανάγκη για την πραγματοποίηση των κατάλληλων ενεργειών. Ο **Mark Weiser**, πρωτοπόρος στον τομέα της περιρρέουσας νοημοσύνης χαρακτηριστικά αναφέρει: *"Τα μηχανήματα θα προσαρμοστούν στο ανθρώπινο περιβάλλον, αυτί να πιέζονται οι άνθρωποι να μπουν σε αυτόν των μηχανών..."*. Αναλογιστείτε τον όγκο των δεδομένων που αναθέτουμε στις υπολογιστικές μονάδες να διαχειριστούν, για να το κάνετε αυτό συνδυάστε ότι έχουμε αναφέρει ως τώρα για το διαδίκτυο, τα δίκτυα αισθητήρων, τα κινητά δίκτυα και τα δίκτυα ομοτίμων να συμβαίνουν ταυτόχρονα στο ίδιο σύστημα. Επομένως, κεντρικό ζήτημα του πυρήνα της περιρρέουσας νοημοσύνης είναι οι τεχνικές συλλογής δεδομένων. Χρειάζεται να πραγματοποιείται αποδοτική εκτίμηση των πληροφοριών, ώστε να υπάρχει η δυνατότητα, μέσω του τεράστιου όγκου δεδομένων, να λαμβάνονται οι κατάλληλες αποφάσεις με διαχωρισμό των έγκυρων και χρήσιμων δεδομένων από τα υπόλοιπα.

1.8 Περίγραμμα Διπλωματικής

Η παρούσα διπλωματική πραγματεύεται τη **συνάθροιση δεδομένων** και το **πρόβλημα μέτρησης των διαφορετικών τιμών** (Distict Value Counting) σε διάφορους τομείς των *κινητών* και *ασύρματων δικτύων*. Πιο συγκεκριμένα δίνεται η αναλυτική περιγραφή των αλγορίθμων επίλυσης του προβλήματος μέτρησης των διαφορετικών τιμών, **Probabilistic Counting with Stochastic Averaging** (PCSA) [1], και **Loglog Counting** [2], και του αλγορίθμου **Time-decaying Sketches** (TDS) [3] που απαντά στο γενικότερο πρόβλημα της συνάθροισης δεδομένων σε δίκτυα αισθητήρων. Έπειτα προχωρούμε στην περιγραφή της υλοποίησης των αλγορίθμων αυτών και της πειραματικής διαδικασίας που ακολουθήθηκε για την αξιολόγηση των τελευταίων, που από όσο γνωρίζουμε λαμβάνει χώρα για πρώτη φορά, στα συγκεκριμένα περιβάλλοντα. Στη συνέχεια, ακολουθεί η αξιολόγηση των πειραμάτων και η γραφική επίδειξη των αποτελεσμάτων με τα κατάλληλα σχόλια και παρατηρήσεις. Τέλος, συνοψίζουμε και καταλήγουμε στα συμπεράσματα και τις μελλοντικές κατευθύνσεις του συγκεκριμένου τομέα.

Κεφάλαιο 2

Συνάθροιση Δεδομένων

2.1 Ορισμός

Συνάθροιση δεδομένων (Data Aggregation) είναι οποιαδήποτε διαδικασία κατά την οποία συλλέγεται πληροφορία και εκφράζεται σε μορφή σύνοψης. Χρησιμοποιείται σε διάφορους τομείς της επιστήμης των υπολογιστών και κυρίως στα κινητά δίκτυα, τα δίκτυα αισθητήρων αλλά και στην βελτιστοποίηση ερωτημάτων βάσεων δεδομένων, όπου έχει και τις ρίζες της. Η βασική ιδέα είναι να συνδυαστούν τα δεδομένα, προερχόμενα από διαφορετικές πηγές καθοδόν προς τον αποδέκτη των δεδομένων, εξαλείφοντας τον πλεονασμό και ελαχιστοποιώντας το πλήθος των μεταδόσεων, σώζοντας έτσι πολύτιμη ενέργεια.

2.2 Ενδοδικτυακή Συνάθροιση Δεδομένων και Δίκτυα Αισθητήρων

Όταν αναφερόμαστε σε ασύρματα δίκτυα αισθητήρων, οι αλγόριθμοι συνάθροισης δεδομένων αποτελούν πρωταρχικό μέσο στη διατήρηση ενέργειας μέσω μείωσης του κόστους επικοινωνίας, το οποίο είναι συνήθως τάξεις μεγέθους μεγαλύτερο από τις τοπικές υπολογιστικές πράξεις σε μια συσκευή αισθητήρα. Η κύρια ιδέα πίσω από την *ενδοδικτυακή συνάθροιση*

δεδομένων (in-network data aggregation) είναι ο συνδυασμός δεδομένων προερχόμενων από διαφορετικές πηγές κατά τη διάρκεια της διαδρομής της απάντησης προς το κέντρο ελέγχου (sink), εξαλείφοντας τον πλεονασμό, ελαχιστοποιώντας το πλήθος των εκπομπών και τελικά αποφεύγοντας την παραπανίσια επικοινωνία.

Λαμβάνοντας υπόψη τους φυσικούς περιορισμούς ενός δικτύου αισθητήρων, δηλαδή την περιορισμένη ενέργεια, τη μικρή μνήμη και υπολογιστική ισχύ, δεν είναι παράξενο το ότι παρατηρούμε την ανάπτυξη των δομών δεδομένων και των μοντέλων δρομολόγησης πακέτων ως θεμελιώδη μονοπάτια στην έρευνα του συγκεκριμένου θέματος. Έχουν προταθεί πολυάριθμες προσεγγίσεις στη συνάθροιση δεδομένων, κυρίως για εφαρμογές σύνοψης δεδομένων και event-based εφαρμογές. Οι προσεγγίσεις αυτές χρησιμοποιούν διάφορες δομές, όπως *δέντρα* (tree-based), *συστάδες* (cluster-based) ή *εμπειρικές* (heuristic) δομές, η κάθε μία από τις οποίες εναρμονίζεται με ένα μέρος της ισορροπίας μεταξύ αποδοτικότητας σε ενέργεια, χρήσης μνήμης, καθυστέρησης και λάθους εκτίμησης του επιθυμητού αποτελέσματος. Παρόλα αυτά, η κατασκευή μιας βέλτιστης δομής για συνάθροιση δεδομένων υποστηρίζοντας ποικίλες *συναθροιστικές συναρτήσεις* (aggregate functions) έχει αποδειχθεί ότι αποτελεί **NP-hard πρόβλημα**.

Από την άλλη μεριά, κάποιες ερευνητικές δραστηριότητες επικεντρώνονται στο πως πραγματοποιείται η συνάθροιση από διαφορετικούς κόμβους, άλλες στο πως κατασκευάζεται και διατηρείται μια δομή που διευκολύνει τη συνάθροιση δεδομένων και τέλος, μερικές δίνουν βάρος στο πόσο αποδοτικά μπορεί γίνει η συμπίεση και η συνάθροιση των δεδομένων λαμβάνοντας υπόψη τη συσχέτιση των τελευταίων. Επιπρόσθετα, προσπάθειες γίνονται προς την κατεύθυνση δημιουργίας *γενικών πλαισίων εργασίας* (general frameworks) για συνάθροιση σε δίκτυα αισθητήρων, τα οποία όμως παρουσιάζουν απογοητευτικά αποτελέσματα όταν εξετάζονται σε διαφορετικές από τις προτεινόμενες τοπολογίες δρομολόγησης.

2.3 Πρόβλημα Μέτρησης Διαφορετικών Τιμών

Το πρόβλημα της μέτρησης των διαφορετικών τιμών σε ένα σύνολο ήταν από τα πρώτα που μελετήθηκαν στον τομέων των *ροών δεδομένων* (data streams). Στην βιβλιογραφία της *στατιστικής* επιστήμης το συγκεκριμένο θέμα αναφέρεται ως η εκτίμηση του πλήθους των ειδών ή των κλάσεων σε ένα πληθυσμό. Πριν δώσουμε τον τυπικό ορισμό του προβλήματος αναφέρουμε τους ορισμούς των *Αλγορίθμων Ροής* (Streaming Algorithms) και των *Συχνοτικών Ροπών* (Frequency Moments).

2.3.1 Αλγόριθμοι Ροής

Μια ροή εισόδου σε μια συνάρτηση $f : A^n \rightarrow B$ είναι μια ακολουθία από ζεύγη $((\pi(1), x_{\pi(1)}), \ldots, (\pi(n), x_{\pi(n)})$, όπου $x \in A^n$ είναι μια είσοδος για την f και $\pi \in S_n$ είναι συνδυασμός της εισόδου αυτής. Υποδηλώνουμε την είσοδο αυτή ως $\pi(x)$. Ένας αλγόριθμος ροής δεδομένων για μια συνάρτηση f είναι ένας τυχαίος αλγόριθμος ο οποίος δέχεται σαν είσοδο μία παράμετρο σφάλματος $\epsilon > 0$ και μία παράμετρο εμπιστοσύνης $0 \leq \delta < 1$, και του δίνεται πρόσβαση ενός περάσματος της ροής εισόδου $\pi(x)$. Η έξοδος του αλγορίθμου είναι μια ϵ-approximation της $f(x)$ με πιθανότητα τουλάχιστον $1 - \delta$, για οποιαδήποτε είσοδο x και οποιοδήποτε συνδυασμό π.

Τα δύο κύρια μεγέθη υπολογισμού της πολυπλοκότητας ενός αλγορίθμου ροής είναι ο *χώρος* και ο *χρόνος επεξεργασίας ανά αντικείμενο*.

- Ο **χώρος** για δεδομένα ϵ και δ είναι το μέγιστο ποσό που καταλαμβάνει ο αλγόριθμος για κάθε δυνατή ροή εισόδου, κάθε συνδυασμό εισόδου και κάθε τυχαία επιλογή του αλγορίθμου.

- Ο **χρόνος επεξεργασίας ανά αντικείμενο** για δεδομένα ϵ και δ είναι ο μέγιστος αριθμός βημάτων που ξοδεύει ο αλγόριθμος σε ένα μόνο ζευγάρι $(\pi(i), x_{\pi(i)})$ της ροής δεδομένων, όπου το μέγιστο υπολογίζεται από όλα τα $i \in [n]$, όλες τις πιθανές εισόδους, όλους τους συνδυασμούς των εισόδων αυτών και όλες τις τυχαίες επιλογές του

αλγορίθμου.

Υπάρχουν διάφορα είδη *approximation* όπως *relative approximation* και *ratio approximation*. Αυτό που θα απασχολήσει εμάς είναι η relative approximation, η οποία ορίζεται ως εξής :

Ορισμός 2.3.1 *Μια* (ϵ, δ)-***approximation*** *για μια συνάρτηση* f*, για* $0 \leq \epsilon \leq 1$*, δίνει για οποιαδήποτε είσοδο* x *μια τιμή στο διάστημα* $((1-\epsilon)f(x), (1+\epsilon)f(x))$ *με πιθανότητα τουλάχιστον* $1 - \delta$.

2.3.2 Συχνοτικές Ροπές

Οι συχνοτικές ροπές σαν έννοια, αναφέρθηκαν για πρώτη φορά από τους *Alon, Matias* και *Szegedy* στο [10]. Ουσιαστικά, οι συχνοτικές ροπές ενός συνόλου δεδομένων μας δίνουν σημαντικές δημογραφικές πληροφορίες για το σύνολο και είναι πολύ χρήσιμα εργαλεία για το πεδίο των βάσεων δεδομένων, της δρομολόγησης στα δίκτυα και γενικότερα όπου είναι απαραίτητη η διαχείριση μεγάλων συνόλων δεδομένων. Δείχνουν το βαθμό ετερογένειας (textitskew) των δεδομένων, που παίζει πρωτεύοντα ρόλο στις προαναφερθείσες περιπτώσεις. Ο τυπικός ορισμός δίνεται αμέσως παρακάτω :

Ορισμός 2.3.2 *Η k-ιοστή συχνοτική ροπή,* $F_k : A^n \rightarrow f$*, παίρνει ως είσοδο μια ακολουθία* n *αντικειμένων* $\sigma_1, \ldots, \sigma_n$ *από ένα σύνολο* $A = \alpha_1, \ldots, \alpha_m$ *μεγέθους* m *και δίνει ως έξοδο το άθροισμα* $\sum_{j=1}^{m} f_j^k$*, όπου* $f_j \equiv |i \in [n] | \sigma_i = \alpha_j|$ *είναι η συχνότητα του* α_j *στην ακολουθία εισόδου.*

F_0, για παράδειγμα, είναι ο αριθμός των διαφορετικών αντικειμένων της ακολουθίας εισόδου και F_1 είναι απλώς ίσο με n.

2.3.3 Ορισμός Προβλήματος Μέτρησης Διαφορετικών Τιμών

Το *πρόβλημα υπολογισμού του πλήθους των διαφορετικών τιμών* ενός πολυ-συνόλου S, μεγέθους N, ή αλλιώς ο υπολογισμός της μηδενικής συχνοτικής ροπής (F_0), ορίζεται ως η μέτρηση του αριθμού των στοιχείων του σύμπαντος που συναντώνται τουλάχιστον μία φορά στο σύνολο S πραγματοποιώντας ένα μόνο πέρασμα στο σύνολο και χρησιμοποιώντας την ελάχιστη δυνατή μνήμη. Η δομή δεδομένων η οποία διατηρείται από τον αλγόριθμο στην μνήμη καλείται *σύνοψη* (synopsis). Το ζητούμενο είναι να βρεθεί αλγόριθμος που δίνει ως έξοδο μια εκτίμηση $\hat{F}_0$, η οποία είναι εγγυημένα κοντά στην πραγματική τιμή της F_0 για την συγκεκριμένη ροή δεδομένων. Οι μετρικές στις οποίες βασίζονται οι αλγόριθμοι που θα αναλύσουμε είναι οι εξής :

relative error metric το σχετικό σφάλμα μιας εκτίμησης $\hat{F}_0$ είναι $|\hat{F}_0 - F_0|/F_0$.

ratio error metric το σφάλμα λόγου μιας εκτίμησης $\hat{F}_0$ είναι $max(F_0/\hat{F}_0, \hat{F}_0/F_0)$.

standard error metric το τυπικό σφάλμα ενός εκτιμητή Υ με τυπική απόκλιση $\sigma_\Upsilon(F_0)$ είναι $\sigma_\Upsilon(F_0)/F_0$.

(ϵ, δ)**-approximation scheme** ένα (ϵ, δ)-approximation scheme για την F_0 είναι μια τυχαία διαδικασία, κατά την οποία δεδομένων $0 \leq \epsilon, \delta < 1$, η παραγόμενη έξοδος είναι μια εκτίμηση $\hat{F}_0$ η οποία βρίσκεται ανάμεσα σε ένα σχετικό σφάλμα με πιθανότητα τουλάχιστον $1 - \delta$.

Σε αυτό το σημείο πρέπει να σημειωθεί ότι το τυπικό σφάλμα σ παρέχει ένα τρόπο για την ανάκτηση ισορροπίας (trade-off) μεταξύ (ϵ, δ) : Υπό συγκεκριμένες υποθέσεις κατανομής (πχ. Γαυσσιαν κατανομή), η εκτίμηση βρίσκεται ανάμεσα σε $\epsilon = \sigma, 2\sigma, 3\sigma$ σχετικό σφάλμα με πιθανότητα $1 - \delta = 65\%, 95\%, 99\%$, αντίστοιχα. Εν αντιθέσει με τα (ϵ, δ)-approximation schemes τα οποία δεν στηρίζονται σε συγκεκριμένες υποθέσεις κατανομής.

2.4 Αλγόριθμοι για Μέτρηση Διαφορετικών Τιμών

Εδώ θα παρουσιάσουμε συνοπτικά κάποιες από τις μεθόδους προσέγγισης της εκτίμησης του πλήθους των διαφορετικών τιμών και τις δυσκολίες που αντιμετώπισε η κάθε προσέγγιση. Τα δύο κύρια είδη είναι οι *Αλγόριθμοι Δειγματοληψίας* (Sampling-based Algorithms) και οι *Αλγόριθμοι Ροής*. Θα αναλύσουμε τους αλγορίθμους των *Flajolet* και *Martin*, *Probabilistic Counting with Stochastic Averaging*, των *Durand* και *Flajolet*, *Loglog Counting Algorithm* και των *Cormode*, *Tirthapura* και *Xu*, *Time-Decaying Sketches Algorithm*. Οι δύο πρώτοι στοχεύουν στην επίλυση του προβλήματος των διαφορετικών τιμών, ενώ ο τελευταίος αναφέρεται σε *συνάθροιση δεδομένων για δίκτυα αισθητήρων* και ενώ έχει περισσότερες δυνατότητες εμείς τον χρησιμοποιούμε για το πρόβλημα των διαφορετικών τιμών λόγω της τελείως διαφορετικής προσέγγισης, μέσω της χρησιμοποίησης *range-sampling τεχνικών*.

2.4.1 Αλγόριθμοι Δειγματοληψίας

Η πιο κοινότυπη προσέγγιση στο πρόβλημα μέτρησης διαφορετικών τιμών, προερχόμενη από τον τομέα της στατιστικής επιστήμης, αλλά και των βάσεων δεδομένων μέχρι τα μέσα της δεκαετίας του 1990, είναι η συλλογή ενός *δείγματος* (sample) των δεδομένων και έπειτα η εφαρμογή εξελιγμένων εκτιμητών στην κατανομή των τιμών εντός του δείγματος. Το γεγονός ότι αυτή η εκτενής έρευνα επικεντρώνεται σε εκτιμητές *δειγματοληψίας* οφείλεται κατά μέρος σε τρείς παράγοντες. Πρώτον, στην εφαρμοσμένη στατιστική, η επιλογή της συλλογής δεδομένων σε περισσότερα του ενός δείγματα του συνολικού πληθυσμού δεν λαμβάνεται γενικώς υπόψη λόγω του απαγορευτικού κόστους. Για παράδειγμα, η συλλογή δεδομένων για κάθε άτομο στον κόσμο ή για κάθε ζώο συγκεκριμένου είδους δεν είναι εφικτή. Δεύτερον, στον τομέα των βάσεων δεδομένων, όπου η σάρωση ολόκληρου ένος μεγάλου συνόλου δεδομένων είναι εφικτή, η χρησιμοποίηση δειγμάτων για την προσεγγιστική συλλογή στατιστικών έχει αποδειχθεί ταχύτατη και αποδοτική μέθοδος για ένα ευρύ φάσμα στατιστικών μεγεθών. Τρίτον, και θεμελιώδες για το πλαίσιο των ροών δεδομένων, η αποτυχία των υπαρχόντων αλγορίθμων δειγματοληψίας να εκτιμήσουν με ακρίβεια την F_0 έχει επικεντρώσει το ενδιαφέρον στην σύνθεση πιο εξεζητημένων αλγορίθμων μεγαλύτερης ακρίβειας.

2.4.2 Αλγόριθμοι Ροής

Προφανώς, η F_0 μπορεί να υπολογιστεί ακριβώς σε μία μόνο σάρωση ολόκληρου του συνόλου των δεδομένων, κρατώντας όλες τις μοναδικές τιμές που παρατηρούνται στη ροή δεδομένων. Αν το σύμπαν είναι $[1 \dots n]$, ένας *πίνακας ψηφίων* (bitmap) μεγέθους n μπορεί να χρησιμοποιηθεί ως *σύνοψη*, αρχικοποιημένος με μηδενικές τιμές, όπου το ψηφίο i τίθεται στο 1 όταν παρατηρείται ένα αντικείμενο με τιμή i. Όμως, στις περισσότερες περιπτώσεις το σύμπαν είναι πολύ μεγάλο (πχ. το σύμπαν των IP διευθύνσεων έχει μέγεθος $n = 2^{32}$), κάνοντας τη σύνοψη πολύ μεγαλύτερη από το μέγεθος της ροής N. Εναλλακτικά, μπορούμε να διατηρούμε ένα σύνολο από όλες τις μοναδικές τιμές που παρατηρούνται, το οποίο χρησιμοποιεί $F_0 \log_2 n$ ψηφία λόγω του ότι κάθε τιμή αποτελείται από $\log_2 n$ ψηφία. Παρόλα αυτά, η F_0 μπορεί να είναι τόσο μεγάλη όσο και το N, οπότε και πάλι η σύνοψη είναι αρκετά μεγάλη. Μία ενδιαφέρουσα και κοινή προσέγγιση για τη μείωση του μεγέθους της σύνοψης κατά ένα παράγοντα $\log_2 n$ είναι να χρησιμοποιηθεί μια *συνάρτηση κατακερματισμού* (hash function) $h()$ αντιστοιχίζοντας τις τιμές σε ένα διάστημα μεγέθους $\Theta(n)$, και προσαρμόζοντας την προαναφερθείσα μέθοδο του πίνακα ψηφίων. Πιο συγκεκριμένα, το ψηφίο $h(i)$ τίθεται 1 όταν παρατηρείται αντικείμενο με τιμή i. Στην περίπτωση της τελευταίας προσέγγισης πρέπει να σημειωθεί ότι το σφάλμα προσέγγισης προέρχεται από "συγκρούσεις" στον *πίνακα κατακερματισμένων τιμών* (hash table), δηλαδή από διαφορετικές τιμές της ροής που αντιστοιχίζονται στο *ίδιο* ψηφίο του πίνακα ψηφίων. Για την αποφυγή αυτού του φαινομένου, χρησιμοποιήθηκαν *Bloom Filters*, με τις πολλαπλές συναρτήσεις κατακερματισμού τους, αποδοτικά, αλλά ο χώρος παραμένει στο $\Theta(n)$ στην χειρότερη περίπτωση.

2.5 Probabilistic Counting with Stochastic Averaging (PCSA)

Ο συγκεκριμένος αλγόριθμος είναι μια καινοτόμα δουλειά των *Flajolet* και *Martin* που έλαβε χώρα το 1985, [1]. Από τότε παραμένει ο πιο σημαντικός και χρησιμοποιούμενος αλγόριθμος, ίσως με μικρές κατά μέρους αλλαγές σε ορισμένες περιπτώσεις, για την επίλυση του προβλήματος των διαφορετίκών τιμών. Ξεκίνησε από το ενδιαφέρον για βελτιστοποίηση

ερωτημάτων βάσεων δεδομένων, που ήταν εκείνη την εποχή δημοφιλές πεδίο έρευνας.

Η μέθοδος που προτείνουν είναι *πιθανοτικής φύσης* αφού τα αποτελέσματα εξαρτώνται από τη συγκεκριμένη συνάρτηση κατακερματισμού που χρησιμοποιείται στα συγκεκριμένα δεδομένα στα οποία εφαρμόζεται. Χρησιμοποιεί ελάχιστο επιπλέον χώρο και προσφέρει πρακτικά χρήσιμες εκτιμήσεις σε μεγάλου μεγέθους σύνολα δεδομένων. Η ακρίβεια της προσέγγισης είναι αντιστρόφως ανάλογη του αποθηκευτικού χώρου : χρησιμοποιώντας *λέξεις 64-ψηφίων* (64-bit words) αποτελούμενες από *32 ψηφία*, πετυχαίνει τυπική ακρίβεια *10%*. Επιπλέον, χρησιμοποιώντας *256 λέξεις*, η ακρίβεια βελτιώνεται στο ποσοστό του *5%* περίπου. Το πιο ενδιαφέρον επίτευγμα όμως, είναι ότι όσο το μέγεθος του συνόλου αυξάνεται η απόδοση του αλγορίθμου παραμένει ίδια. Για παράδειγμα, εφαρμόζοντας *λέξεις 32 ψηφίων*, μπορεί κάποιος να μετρήσει ασφαλώς μεγέθη *πολύ μεγαλύτερα από 100 εκατομμύρια*. Η μόνη υπόθεση που γίνεται είναι ότι οι εγγραφές μπορούν να κατακερματιστούν καταλλήλως σε ψευδο-ομοιόμορφη (pseudo-uniform) κατανομή. Αυτό όμως, δεν φαίνεται να είναι σημαντικός περιορισμός, αφού εμπειρικές μελέτες σε μεγάλες συλλογές δεδομένων έχουν δείξει ότι *προσεκτικές* υλοποιήσεις των τυπικών τεχνικών κατακερματισμού μπορούν πρακτικά να πετύχουν *ομοιομορφία* των κατακερματισμένων τιμών. Επιπλέον, ο συγκεκριμένος αλγόριθμος δεν περιέχει, από το σχεδιασμό του, δομές συλλογών δεδομένων που περιέχουν διπλότυπες εγγραφές, αντίθετα με τις τεχνικές *δειγματοληψίας* στις οποίες τα αποτελέσματα θα είναι τα ίδια είτε το στοιχείο εμφανίζεται ένα εκατομμύριο φορές είτε μόνο λίγες φορές.

Παρακάτω, ακολουθεί η επεξήγηση μιας πιθανοτικής μεθόδου μέτρησης και η ανάλυσή της για να γίνει πιο κατανοητή η περιγραφή του αλγορίθμου PCSA στη συνέχεια.

2.5.1 Πιθανοτική Μέθοδος Μέτρησης και η Ανάλυσή της

Η Βασική Μέθοδος Μέτρησης

Υποθέτουμε ότι έχουμε στην κατοχή μας μια συνάρτηση κατακερματισμού του τύπου :

$$\textbf{function } hash(x : records) : \ scalar\ range[0 \cdots 2^L - 1],$$

η οποία μετατρέπει τις εγγραφές σε ακέραιους, ικανοποιητικά ομοιόμορφα κατανεμημένους στο μονοδιάστατο διάστημα ή όμοια στο *δυαδικό αλφαριθμητικό* (βιναρψ στρινγ) μεγέθους L. Για y οποιονδήποτε μη αρνητικό αριθμό, ορίζουμε $bit(y,k)$ το k-ιοστό ψηφίο της δυαδικής αναπαράστασης του y, έτσι ώστε

$$y = \sum_{k \geq 0} bit(y,k) 2^k$$

Εισάγουμε, επίσης τη συνάρτηση $\rho(y)$ η οποία αναπαριστά τη θέση του λιγότερου σημαντικού *1-ψηφίου* στη δυαδική αναπαράσταση του y, με μια κατάλληλη σύμβαση για το $\rho(0)$:

$$\begin{aligned} \rho(y) = min_{k \geq 0}\, bit(y,k) \neq 0 \qquad & \text{αν} \quad y > 0 \\ = L \qquad & \text{αν} \quad y = 0 \end{aligned}$$

Παρατηρούμε ότι αν οι τιμές $hash(x)$ είναι ομοιόμορφα κατανεμημένες, το πρότυπο $0^k 1 \cdots$ εμφανίζεται με πιθανότητα 2^{-k-1}. Η ιδέα έχει σαν στόχο την καταγραφή των παρατηρήσεων τέτοιων προτύπων σε ένα πίνακα ψηφίων $BITMAP[0 \ldots L-1]$. Αν M είναι το πολυσύνολο στο οποίο μετράμε τις διαφορετικές τιμές, εφαρμόζουμε τις ακόλουθες πράξεις :

```
for i:=0 to L-1 do BITMAP[i]:=0;
for all x in M do
        begin
        index:=r(hash(x));
        if BITMAP[index]=0 then BITMAP[index]:=1;
end;
```

Επομένως, το στοιχείο του πίνακα $BITMAP[i]$ ισούται με 1 αν και μόνον αν μετά την εκτέλεση το πρότυπο της μορφής $0^i 1$ έχει εμφανιστεί ανάμεσα στις κατακερματισμένες τιμές των εγγραφών του πολυσυνόλου M. Παρατηρείστε ότι από κατασκευής, ο πίνακας $BITMAP$ εξαρτάται μόνο από το σύνολο των κατακερματισμένων τιμών και όχι από τη συγκεκριμένη συχνότητα με την οποία επαναλαμβάνονται οι τιμές καθαυτές.

Από τις προηγούμενες παρατηρήσεις που αφορούν τις πιθανότητες προτύπων, πρέπει να αναμένουμε, αν n είναι ο αριθμός των διαφορετικών στοιχείων στο M, το $BITMAP[0]$ να αντιστοιχίζεται προσεγγιστικά $n/2$ φορές, το $BITMAP[1]$ περίπου $n/4$ φορές και ούτω καθεξής. Επομένως, στο τέλος της εκτέλεσης, το στοιχείο $BITMAP[i]$ θα είναι σχεδόν σίγουρα *μηδέν* αν $i \gg \log_2 n$ και *ένα* αν $i \ll \log_2 n$ με το ένα άκρο αποτελούμενο από μηδενικά και άσσους για $i \approx \log_2 n$. Για παράδειγμα, πήραμε ως πολυσύνολο M τον πρώτο τόμο του εγχειριδίου του ΥΝΙΞ συστήματος. Το M αποτελείται από 26692 γραμμές από τις οποίες 16405 είναι διακριτές. Λαμβάνοντας υπόψη αυτές τις γραμμές ως εγγραφές και κατακερματίζοντάς τες σε τιμές μέσω πολλαπλασιαστικής συνάρτησης κατακερματισμού 24-ψηφίων ($L = 24$), βρήκαμε τον ακόλουθο $BITMAP$ πίνακα ψηφίων :

$$111111111111001100000000$$

Η αριστερότερη τιμή που ισούται με *μηδέν* βρίσκεται στη θέση 12 και η δεξιότερη τιμή που είναι ίση με *ένα* βρίσκεται στη θέση 15, ενώ $2^{14} = 16384$.

Προτείνουν, λοιπόν, οι συγγραφείς την χρησιμοποίηση της θέσης του αριστερότερου ψηφίου που ισούται με μηδέν (προσοχή, οι θέσεις ξεκινούν από το 0!) ως προσέγγιση του $\log_2 n$. Ας υποθέσουμε ότι R καλούμε αυτή την ποσότητα, θα δούμε ότι κάτω από την υπόθεση ότι οι κατακερματισμένες τιμές είναι ομοιόμορφα κατανεμημένες, η μέση τιμή (expected value) της R είναι κοντά στο :

$$E(R) \approx \log_2 \phi n, \quad \phi = 0.77351 \tag{2.1}$$

έτσι η διαισθητική μας επιλογή είναι δικαιολογημένη. Συγκεκριμένα, ο παράγοντας συσχέτισης ϕ παίζει σημαντικό ρόλο στο σχεδιασμό του τελικού αλγορίθμου που παρουσιάζεται εδώ. Θα αποδείξουμε ακόμα ότι κάτω από λογικές πιθανοτικές υποθέσεις, η τυπική απόκλιση του R είναι κοντά στο :

$$\sigma(R) \approx 1.12 \tag{2.2}$$

έτσι η εκτίμηση βασισμένη στην (2.1) θα είναι συνήθως μια δυαδικής τάξης μεγέθους εκτός του ακριβούς αποτελέσματος, γεγονός το οποίο αναδεικνύει την ανάγκη για πιο εξεζητημένους αλγορίθμους.

Πιθανοτικές Κατανομές

Στην παράγραφο αυτή θα οριστεί το πιθανοτικό μοντέλο με το οποίο οι συγγραφείς αποδεικνύουν τους ισχυρισμούς που έθεσαν στις εξισώσεις (2.1) και (2.2) όσον αφορά την κατανομή της τιμής της παραμέτρου R στη βασική διαδικασία μέτρησης. Θυμίζουμε ότι η παράμετρος R αναπαριστά την ποσότητα $log_2 n$, όπου n είναι ο αριθμός των διακριτών στοιχείων του πολυσυνόλου M. Παρακάτω ακολουθούν τα θεωρήματα και λήμματα που οδήγησαν τους συγγραφείς στους παραπάνω ισχυρισμούς και επιλογές.

Πιθανοτικό Μοντέλο

Έστω B το σύνολο των άπειρων δυαδικών αλφαριθμητικών. Το μοντέλο υποθέτει ότι τα ψηφία των στοιχείων του B είναι ομοιόμορφα και ανεξάρτητα κατανεμημένα. Ομοίως, τα αλφαριθμητικά μπορούν να θεωρηθούν πραγματικοί ακέραιοι αριθμοί στο διάστημα $[0, 1]$, και επίσης υποθέτουμε ότι οι αριθμοί είναι ομοιόμορφα κατανεμημένοι στο διάστημα αυτό. οι συναρτήσεις bit και ρ εκτείνονται ως το B χωρίς πρόβλημα. Υποδηλώνουμε με R_n την τυχαία μεταβλητή που ορίζεται στο B^n (υποθέτοντας ανεξαρτησία), αυτή είναι, το ανάλογο της παραμέτρου R :

Ορισμός 2.5.1 $R_n(x_1, x_2, \ldots, x_n) = r$ *αν και μόνον αν*

1. *για όλα τα* $0 \leq j < r$ *υπάρχει* i *τέτοιο ώστε* $r(x_i) = j$ *και*
2. *για όλα τα* i $\rho(x_i) \neq r$.

Επίσης, εισάγουμε την ακόλουθη σημειογραφία όσον αφορά την κατανομή πιθανότητας της R_n υπό το ομοιόμορφο μοντέλο :

$$p_{n,k} = Pr(R_n = k), \qquad q_{n,k} = PR(R \geq k)$$

$$\bar{R_n} = E(R_n) = \sum_{k \geq 0} k p_{n,k}$$

$$s_n^2 = E((R_n - \bar{R_n})^2) = \sum_{k \geq 0} k^2 p_{n,k} - \bar{R_n^2}$$

και έστω ότι $n(n)$ είναι ο αριθμός των άσσων στην δυαδική αναπαράσταση του n, πχ. $v(13) = v((1101)_2) = 3$.

Θεώρημα 2.5.2 *Η κατανομή πιθανότητας R_n χαρακτηρίζεται από :*

$$q_{n,k} = \sum_{j=0}^{2^k} (-1)^{n(j)} (1 - \frac{j}{2^k})^n$$

Περνώντας τώρα στην *απόκλιση* των ασυμπτωτικών μορφών των πιθανοτήτων που ορίστηκαν παραπάνω έχουμε το ακόλουθο θεώρημα και λήμματα για την ασυμπτωτική ανάλυση :

Θεώρημα 2.5.3 *Η κατανομή της R_n ικανοποιεί τις ακόλουθες εκτιμήσεις :*

1. *Αν $k < \log_2 n - 2\log_2 \log n$, τότε*

$$q_{n,k} = 1 - O(ne^{-\log^2 n})$$

2. *Αν $k \leq \frac{3}{2}\log_2 n$, τότε*

$$\begin{aligned} q_{n,k} &= \prod_{j=0}^{\infty} (1 - e^{-2^j (n/2^k)}) + O(\frac{\log^6 n}{\sqrt{n}}) \\ &= \sum_{j \geq 0} [(-1)^{n(j)} e^{-jn/2^k}] + O(\frac{\log^6 n}{\sqrt{n}}) \end{aligned}$$

3. *Αν* $k \geq \frac{3}{2}\log_2 n + \delta$, *με* $\delta \geq 0$, *το κάτω άκρο της κατανομής είναι εκθετικό :*

$$q_{n,k} = O(\frac{2^{-\delta}}{\sqrt{n}})$$

Το θεώρημα 2.4.2 εκφράζει ουσιαστικά την ύπαρξη ενός είδους *περιοριστικής κατανομής* για την κατανομή πιθανότητας R_n, όσο το n μεγαλώνει :

$$q_{n,k} \approx \psi(\frac{n}{2^k}), \qquad p_{n,k} \approx \psi(\frac{n}{2^k}) - \psi(\frac{n}{2^{k+1}})$$

Το άκρο της κατανομής πιθανότητας μειώνεται κατακόρυφα και στην πραγματικότητα, πειραματικά αποτελέσματα έχουν δείξει ότι η μείωση συνήθως είναι πολύ γρηγορότερη από ότι επιδεικνύει το θεώρημα 2.4.2.

Από το θεώρημα 2.4.2 προκύπτει το παρακάτω λήμμα :

Λήμμα 2.5.4 *Η μέση τιμή* $\bar{R}_n$ *της* R_n *ικανοποιεί τη σχέση :*

$$\bar{R}_n = \sum_{k\geq 1} k[\psi(\frac{n}{2^k}) - \psi(\frac{n}{2^k+1})] + O(\frac{1}{n^{0.49}})$$

Επομένως, το πρόβλημα εκτίμησης της μέσης τιμής $\bar{R}_n$ ασυμπτωτικά περιορίζεται σε αυτό της εκτίμησης του αθροίσματος της παραπάνω σχέσης. Έχουμε δηλαδή, τον υπολογισμό της :

$$F(x) = \sum_{k\geq 1} k[\psi(\frac{x}{2^k}) - \psi(\frac{x}{2^{k+1}})]$$

Για αυτό το λόγο, θα χρησιμοποιηθεί ο μετασχηματισμός *Mellin*, ο οποίος ορίζει ότι για μια συνάρτηση $f(x)$ ορισμένη για $x \geq 0$, με x πραγματικό, ο μετασχηματισμός είναι η μιγαδική συνάρτηση $f^*(s)$, η οποία δίνεται από τον τύπο :

$$f^*(s) \equiv M[f(x); s] = \int_0^\infty f(x)x^{s-1}dx \tag{2.3}$$

Ο μετασχηματισμός αυτός είναι ουσιαστικά η πολλαπλασιαστική έκδοση του *δίπλευρου μετασχηματισμού Laplace* και χρησιμοποιείται εκτενώς στη θεωρία ασυμπτωτικών σειρών. Θα τον χρησιμοποιήσουμε υπολογίζοντας πρώτα τον ευθύ μετασχηματισμό του αθροίσματος και έπειτα αντιστρέφοντας παίρνουμε την ασυμπτωτική επέκταση του αθροίσματος που ζητούμε. Μια σημαντική και χρήσιμη για μας ιδιότητα του μετασχηματισμού *Mellin* είναι η παρακάτω :

$$M[f(ax); s] = \alpha^{-s} f^*(s)$$

μέσω αυτής παίρνουμε τον παρακάτω τύπο του αντίστροφου μετασχηματισμού :

$$f(x) = \frac{1}{2i\pi} \int_{c-i\infty}^{c+i\infty} f^*(s) x^{-s} ds \tag{2.4}$$

όπου το c επιλέγεται μέσα στη *λωρίδα* (strap) όπου το ολοκλήρωμα της σχέσης (2.3) συγκλίνει απόλυτα. Το ενδιαφέρον για τον αντίστροφο μετασχηματισμό είναι ότι σε πολλές περιπτώσεις, μπορεί να υπολογιστεί μέσω του *θεωρήματος υπολοίπου του Cauchy*, αντιστοιχίζοντας κάθε υπόλοιπο σε έναν όρο της ασυμπτωτικής επέκτασης της f.

Λήμμα 2.5.5 *Ο μετασχηματισμός Mellin της $F(x)$ είναι για $-1 < Re(s) < 0$:*

$$F^*(s) = \frac{2^s}{1-2^s} N(s)\Gamma(s),$$

όπου $\Gamma(s)$ είναι η Γάμμα συνάρτηση του Euler και $N(s)$ είναι μια συνάρτηση αναλυτικής συνέχειας της συνάρτησης ορισμένης για $Re(s) > 1$ από τον τύπο :

$$N(s) = \sum_{j \geq 1} \frac{(-1)^{n(j)}}{j^s}$$

Πρέπει τώρα να κατασκευάσουμε πιο αποδοτικές ιδιότητες της συνάρτησης $N(s)$ για $Re(s) < 0$, επιβεβαιώνοντας ταυτόχρονα και την ιδιότητα της συνέχειας της συνάρτησης που χρησιμοποιήσαμε στο παραπάνω λήμμα.

Λήμμα 2.5.6 *Η συνάρτηση* $N(s)$ *ικανοποιεί τη σχέση* $N(0) = -1$. *Συγκεκριμένα, για* $s = \sigma + it$ *και* $\sigma > -0.99$, *ικανοποιεί τη σχέση :*

$$N(s) = O(1 + |s^3|)$$

Μπορούμε τώρα να γυρίσουμε στον ασυμπτωτικό υπολογισμό της $F(x)$, και επομένως της $\bar{R_n}$, χρησιμοποιώντας τον τύπο του αντίστροφου μετασχηματισμού Mellin (2.4).

Θεώρημα 2.5.7 *Η μέση τιμή της παραμέτρου* R_n *ικανοποιεί τη σχέση :*

$$\bar{R_n} = \log_2 \phi n + P(\log_2 n) + o(1),$$

όπου η σταθερά διόρθωσης $\phi = 0.077351\ldots$ *και δίνεται από του τύπο :*

$$\phi = 2^{-1/2} e^g \frac{2}{3} \prod_{p=1}^{\infty} [\frac{(4p+1)(4p+2)}{(4p)(4p+3)}]^{(-1)^{n(r)}}$$

και $P(u)$ *είναι μια περιοδική και συνεχής συνάρτηση του* u *με περίοδο ίση με 1 και πλάτος φραγμένο στο* 10_{-5}.

Με τα παραπάνω δεδομένα μπορούμε τώρα να εκτιμήσουμε την τυπική απόκλιση της R_n. Έστω S_n η *δεύτερη ροπή* (second moment) της R_n : $S_n = E(R_n^2)$. Η S_n προσεγγίζεται από τη συνάρτηση $G(n)$ όπως και πριν, όπου

$$G(x) = \sum_{k^2} [\psi(\frac{x}{2^k}) - \psi(\frac{x}{2^{k+1}})],$$

της οποίας ο μετασχηματισμός έχει βρεθεί για $Re(s) < 0$:

$$G^*(s) = \frac{2^s(1+2^s)}{(1-2^s)^2} \Gamma(s) N(s)$$

ο οποίος έχει τριπλό πόλο στο $s = 0$. Υπολογίζουμε την $G(z)$ από την $G^*(s)$ με αντίστροφο μετασχηματισμό ακολουθώντας τους υπολογισμούς των υπολοίπων και βρίσκουμε :

Θεώρημα 2.5.8 *Η τυπική απόκλιση της R_n ικανοποιεί τη σχέση :*

$$\sigma_n^2 = \sigma_\infty^2 + Q(\log_2 n) + o(1),$$

όπου $\sigma_\infty = 1.12127\ldots$ και $Q(u)$ είναι περιοδική συνάρτηση με μέση τιμή 0 και περίοδο 1.

Επομένως, έχουμε τώρα στη διάθεσή μας τους τύπους που χρειαζόμαστε για να πραγματοποιήσουμε τις εκτιμήσεις των ποσοτήτων που επιθυμούμε, δηλαδή τη μέση τιμή της R_n η οποία μας οδηγεί στον υπολογισμό της $\log_2 n$ που είναι και ο στόχος μας.

2.5.2 Probabilistic Counting Αλγόριθμος

Στην προηγούμενη υποενότητα είδαμε ότι η ποσότητα $\log_2 n$ που ουσιαστικά είναι το αποτέλεσμα της διαδικασίας μέτρησης που παρουσιάστηκε αρχικά, οι συγγραφείς την ονομάζουν *COUNT*, έχει μέση τιμή κοντά στο $\log_2 \phi n$ και τυπική απόκλιση περίπου 1.12. Στην πραγματικότητα, οι τιμές της $\lambda(n) = (1/\phi)2^{R_n}$ είναι εκπληκτικά κοντά στο n όπως στα παρακάτω παραδείγματα :

$$\lambda(10) = 10.502 \qquad \lambda(100) = 10.4997 \qquad \lambda(1000) = 1000.502$$

Επομένως, μπορούμε δικαιολογημένα να ελπίζουμε για πολύ καλές εκτιμήσεις του n από την παρατήρηση της R, χρησιμοποιώντας τον παράγοντα διόρθωσης ϕ. Παρόλα αυτά, η διασπορά των αποτελεσμάτων αντιστοιχεί σε τυπικό σφάλμα μιας δυαδικής τάξης μεγέθους, που είναι φυσικά πολύ μεγάλο για πολλές εφαρμογές.

Για να αποφευχθεί αυτή η κατάσταση, μπορούμε να εφαρμόσουμε την απλούστατη ιδέα της χρησιμοποίησης ενός συνόλου H από m συναρτήσεις κατακερματισμού, όπου m είναι σχεδιαστική παράμετρος και έτσι υπολογίζουμε m διαφορετικούς $BITMAP$ πίνακες ψηφίων. Με αυτόν τον τρόπο, παίρνουμε m εκτιμήσεις $R^{<1>}, R^{<2>}, \ldots, R^{<m>}$. Τότε μπορεί να υπολογιστεί ο μέσος όρος αυτών ως :

$$A = \frac{R^{<1>} + R^{<2>} + \cdots + R^{<m>}}{m}$$

Όταν n διαφορετικά στοιχεία ανήκουν στο σύνολο, η τυχαία μεταβλητή A έχει μέση τιμή και τυπική απόκλιση ως εξής :

$$E(A) \approx \log_2 \phi n, \qquad \sigma(A) \approx \sigma_\infty / \sqrt{m}$$

Έτσι, περιμένουμε ότι η τιμή 2^A θα παρέχει μια εκτίμηση του n με τυπικό σφάλμα της σχετικής τιμής $\approx K/\sqrt{m}$.

Αυτός ο αλγόριθμος, χρησιμοποιώντας *direct averaging* έχει όντως πολύ καλή απόδοση, το σχετικό σφάλμα του είναι περίπου 10% όταν $m = 64$, αλλά έχει ένα σημαντικό μειονέκτημα, χρειάζεται τον υπολογισμό m διαφορετικών συναρτήσεων κατακερματισμού και για αυτό το CPU κόστος ανά στοιχείο πολλαπλασιάζεται με ένα παράγοντα της τάξης του m.

Το γεγονός αυτό οδήγησε τους συγγραφείς από το *direct averaging* στο αντίστοιχο *stochastic*. Η βασική ιδέα είναι στη χρησιμοποίηση της συνάρτησης κατακερματισμού με σκοπό την κατανομή κάθε στοιχείου σε ένα από τα m κενά, υπολογίζοντας το $\alpha = h(x) \mod m$. Έπειτα, ενημερώνουμε τον αντίστοιχο $BITMAP$ πίνακα ψηφίων της διεύθυνσης α με την υπόλοιπη πληροφορία να περιέχεται στο $h(x)$, και συγκεκριμένα στο $h(x) \div m \equiv \lfloor h(x)/m \rfloor$. Στο τέλος, καθορίζουμε όπως παραπάνω τα $R^{<j>}$ και υπολογίζουμε τον μέσο όρο τους A από τη σχέση (2.5.2).

Ο αλγόριθμος που περιγράφηκε ακριβώς παραπάνω ονομάζεται *Probabilistic Counting with Stochastic Averaging*. Ο ψευδοκώδικάς του δίνεται λίγο παρακάτω, σχήμα 2.1. Ο αλγόριθμος αποδεικνύεται ότι παρουσιάζει *τυπικό σφάλμα* και *πόλωση* (bias) :

πόλωση : $1 + 0.3/m$

τυπικό σφάλμα : $0.78/\sqrt{m}$

```
program PCSA;

const nmap = 64;  {with nmap = 64, accuracy is typically 10%}
   {nmap corresponds to variable m in the analysis}
   φ = 0.77351 {the magic constant}; maxlength = 32;
   {with maxlength =32 (==L), one can count up to 10^8.}

var M: multiset of data of type records;
      x: records; hashedx, index, α, R, S, Ξ: integer;
      BITMAPS: array [0..nmap-1;0..maxlength-1] of integer;

function getelement(var x:records);
{reads an element x of type records from file M}
function hash(x:records):integer;
{hashes a record x into an integer over scalar range [0..2^maxlength-1]}
function ρ(y:integer):integer;
{returns the position of the first 1-bit in y; ranks start at 0.}

begin
while not eof(M) do
     begin
     getelement(x); hashedx:=hash(x);
     α:=hashedx mod nmap; index:=ρ(hashedx div nmap);
     if BITMAP[α,index]=0 then BITMAP[α,index]:=1;
     end;
S = 0;
for i:=0 to nmap-1 do
     begin
     R:=0; while (BITMAP[i,R]=1) and (R<maxlength) do R:=R+1; S:=S+R;
     end;
Ξ:=trunc(nmap/φ*2**(S/nmap));
{Result Ξ of the PCSA programme that estimates n}
end.
```

Σχήμα 2.1: PCSA αλγόριθμος

Όπου ως *τυπικό σφάλμα* καλούμε τον λόγο της τυπικής απόκλισης της εκτίμησης του n προς την τιμή του n, και *πόλωση* το λόγο μεταξύ της εκτίμησης του n και της ακριβής τιμή n.

2.6 Loglog Counting

Ο αλγόριθμος *Loglog Counting* είναι ουσιαστικά μια διαφορετική έκδοση του *PCSA* αλγορίθμου που παρουσιάστηκε στην προηγούμενη ενότητα. Η βασική διαφορά τους είναι ότι ο Loglog Counting αλγόριθμος μειώνει το μέγεθος της σύνοψης δεδομένων από $\log n$ σε $\log \log n$, ενώ όμως το τυπικό σφάλμα αυξάνεται από $0.78/\sqrt{m}$ σε $1.30/\sqrt{m}$, όπου m είναι το πλήθος των ΒΙΤΜΑΠ πινάκων ψηφίων που έχουν χρησιμοποιηθεί. Επομένως, αυτός ο αλγόριθμος βελτιώνει την πολυπλοκότητα χώρου κατά μία λογαριθμική τάξη μεγέθους, όμως το σφάλμα αυξάνεται αρκετά, αυτό βέβαια σε πολλές εφαρμογές είναι επιθυμητό επειδή η μείωση του χρησιμοποιούμενου χώρου είναι σημαντική και επιτρέπει τη μέτρηση των διαφορετικών στοιχείων ενός συνόλου με μέγεθος δισεκατομμυρίων στοιχείων.

Όπως και ο PCSA αλγόριθμος έτσι και ο Loglog Counting ανήκει στην κατηγορία των πιθανοτικών αλγορίθμων. Η πρώτη ιδέα της πιθανοτικής προσέγγισης σε ένα τέτοιου είδους πρόβλημα είναι η χρησιμοποίηση μιας συνάρτησης κατακερματισμού επιδιώκοντας την απεικόνιση των δεδομένων από την μορφή στην οποία βρίσκονται σε μία δυαδική που "μοιάζει" τυχαία και ανεξάρτητη. Η νέα αυτή μορφή των δεδομένων χρησιμοποιείται από τον αλγόριθμο για την παραγωγή εκτιμήσεων του πλήθους των διαφορετικών αντικειμένων.

2.6.1 Ο Βασικός Loglog Counting Αλγόριθμος

Υποθέτουμε ότι έχουμε στην κατοχή μας μια συνάρτηση κατακερματισμού h, η οποία μετατρέπει τα δεδομένα σε ικανοποιητικά μεγάλα δυαδικά αλφαριθμητικά με τέτοιο τρόπο ώστε οι κατακερματισμένες (ηασηεδ) τιμές να είναι *πολύ κοντά* σε ομοιόμορφη ανεξάρτητη κατανομή. Αναφερόμαστε με τον όρο *πολύ κοντά* εξαιτίας του ισχυρισμού του *D. Knuth* ότι είναι αδύνατο να οριστεί συνάρτηση κατακερματισμού η οποία δημιουργεί τυχαία δεδομένα από μη-τυχαία πραγματικά δεδομένα, αλλά στην πράξη δεν είναι δύσκολο να παράγουμε πολύ καλές "απομιμήσεις" τυχαίων δεδομένων. Αμέσως παρακάτω δίνεται το βασικό κομμάτι του Loglog Counting αλγορίθμου :

Αλγόριθμος **LOGLOG**(M: Πολυσύνολο από κατακερματισμένες τιμές, $m \equiv 2^k$)
Αρχικοποίησε τα $M^{(1)}, \ldots, M^{(m)}$ σε 0;
έστω $\rho(y)$ η θέση του πρώτου 1-ψηφίου από τα αριστερά στο y;
για $x = b_1 b_2 \cdots \in M$ **κάνε**
θέσε $j := \langle b_1 \cdots b_k \rangle_2$; (τιμή των πρώτων k ψηφίων με βάση το 2)
θέσε $M^{(x)} := \max(M^{(j)}, \rho(b_{k+1} b_{k+2} \cdots))$;

επέστρεψε $E := \alpha_m m 2^{\frac{1}{m}\sum_j M^{(j)}}$ ως την εκτίμηση για το n.

Ο θεμέλιος λίθος του αλγορίθμου βρίσκεται στη σάρωση του πολυσυνόλου και στην παρατήρηση συγκεκριμένων προτύπων της μορφής $0 * 1$ τα οποίο εμφανίζονται στην αρχή των κατακερματισμένων εγγραφών. Για ένα αλφαριθμητικό $x \in 0, 1^\infty$, έστω $\rho(x)$ όπως και πριν η συνάρτηση που μας δίνει τη θέση του πρώτου 1-ψηφίου. Επομένως, $\rho(1 \cdots) = 1$, $\rho(001 \cdots) = 3$, κτλ. Περιμένουμε περίπου τα $n/2^k$ από τα στοιχεία του M να έχουν $rho-$τιμή ίση με k. Δηλαδή, η ποσότητα :

$$R(M) := \max_{x \in M} \rho(x),$$

μπορεί να παρέχει μια γενική ένδειξη της τιμής του $\log_2 n$. Και πάλι όμως, ερχόμαστε στο πρόβλημα που αντιμετώπισαν και οι Flajolet και Martin στον βασικό τους αλγόριθμο *COUNT*. Η ποσότητα R προσεγγίζει το $\log_2 n$ με προστιθέμενη πόλωση (ο λόγος που ορίζεται ως η εκτίμηση του n προς την πραγματική τιμή του n) 1.33 και τυπική απόκλιση ίση με 1.87. Επομένως, οι εκτιμήσεις του αλγορίθμου με βάση την ποσότητα R βρίσκονται μέσα στο διάστημα ± 1.87 δυαδικών τάξεων μεγέθους.

Για αυτόν ακριβώς το λόγο, διαμοιράζουμε τα στοιχεία σε m υποσύνολα, με το $m = 2^k$ γιατί διευκολύνει το διαχωρισμό των στοιχείων, τα πρώτα k ψηφία ενός στοιχείου αναπαριστούν σε ποιό από τα m υποσύνολα θα εισαχθεί. Τότε θα υπολογιστούν τα m σε πλήθος R και ο μέσος όρος τους $\frac{1}{m}\sum_{j=1}^{m} M^{(j)}$, θα αποτελεί την προσέγγιση της ποσότητας $\log_2 n/m$ με μία προστιθέμενη πόλωση. Για να φτάσουμε στην εκτίμηση του n παίρνουμε τον τύπο :

$$E := \alpha_m m 2^{\frac{1}{m}\sum_j M^{(j)}}$$

Η σταθερά α_m προέρχεται από την ανάλυση που πραγματοποίησαν οι συγγραφείς προσπαθώντας να καταλήξουν στον υπολογισμό της μέσης τιμής της ποσότητας R. Η επίλυση του υπολογισμού αυτού όμως δεν είναι δυνατή με την παρούσα του μορφή, για αυτό και χρησιμοποιήθηκε η διαδικασία **Poissonization**. Μέσω της τελευταίας εκμεταλλεύτηκαν τη μετατροπή του πιθανοτικού μοντέλου από σταθερού-n σε **Poisson** με σκοπό την χρησιμοποίηση των πολύ χρήσιμων ιδιοτήτων του μετασχηματισμού *Mellin* (2.3). Αφού λοιπόν μετατραπεί το μοντέλο και γίνουν οι κατάλληλοι μετασχηματισμοί, παίρνουμε τις ασυμπτωτικές μορφές και με αντίστροφη διαδικασία, η οποία καλείται **depoissonization**, τις φέρνουμε πάλι στο αρχικό μας μοντέλο. Έτσι οι συγγραφείς κατέληξαν στον τύπο :

$$\alpha_m = (\Gamma(-1/m)\frac{1-2^{1/m}}{\log 2})^{-m},$$

όπου $\Gamma(s) := \frac{1}{s}\int_0^\infty e^{-t}t^s dt$. Η α_m διορθώνει τη συστηματική πόλωση της μέσης τιμής στο ασυμπτωτικό όριο αυτής.

Τα κύρια χαρακτηριστικά του αλγορίθμου Loglog Counting συνοψίζονται στο παρακάτω θεώρημα. Τα σύμβολα E και V συμβολίζουν την μέση τιμή και την απόκλιση αντίστοιχα.

Θεώρημα 2.6.1 *Θεωρείστε το βασικό αλγόριθμο Loglog Counting εφαρμοζόμενο σε ένα ιδανικό πολυσύνολο αγνώστου πλήθους διαφορετικών στοιχείων n και έστω E είναι η εκτιμώμενη τιμή του n που επιστρέφει ο αλγόριθμος.*

1. *Η εκτίμηση της E είναι μη-πολωμένη ασυμπτωτικά με την έννοια ότι όσο $n \to \infty$,*

$$\frac{1}{n}E_n(E) = 1 + \theta_{1,n} + o(1), \text{ όπου } |\theta_{1,n}| < 10^{-6}$$

2. *Το τυπικό σφάλμα οριζόμενο ως $\frac{1}{n}\sqrt{V_n(E)}$ όσο το $n \to \infty$ ικανοποιεί τη σχέση :*

$$\frac{1}{n}\sqrt{V_N(E)} = \frac{\beta_m}{\sqrt{m}} + \theta_{2,n} + o(1), \text{ όπου } |\theta_{2,n}| < 10^{-6}$$

Έτσι μπορεί κάποιος για παράδειγμα να έχει : $\beta_{128} = 1.305440$, $\beta_\infty = \sqrt{\frac{1}{12}\log^2 2 + \frac{1}{6}\pi^2} = 1.29806$.

Για να συνοψίσουμε, εκτός από πλήρως αμελητέες διακυμάνσεις των οποίων το πλάτος είναι μικρότερο από 10^{-6}, ο αλγόριθμος παρέχει ασυμπτωτικά έγκυρες εκτιμήσεις για το n. Το τυπικό σφάλμα, το οποίο μετράει με μια μέση-τετραγωνική έννοια και σε αναλογία με το n τις αναμενόμενες αποκλίσεις, είναι πολύ κοντά προσεγγιστικά με τον τύπο :

$$\text{Τυπικό Σφάλμα } \approx \frac{1.30}{\sqrt{m}}.$$

Για παράδειγμα, οι τιμές $m = 256$ και $m = 1024$ δίνουν τυπικό σφάλμα του 8% και 4% αντίστοιχα. Παρατηρήστε επίσης, ότι $\alpha_m \sim a_\infty - (2\pi^2 + \log^2 2)\,(48m)$, όπου $\alpha_\infty = e^{-\gamma}\sqrt{2}/2 = 0.39701$ (γ είναι η σταθερά του Euler), έτσι ώστε πρακτικά σε υλοποιήσεις το α_m μπορεί να αντικατασταθεί με το α_∞ χωρίς να παρατηρηθεί επιπλέον πόλωση όσο το $m \geq 64$.

2.6.2 Απαιτήσεις Χώρου

Ο αλγόριθμος Loglog Counting εφαρμόζεται με ουσιαστικά μη φραγμένο ακέραιο αριθμό από καταχωρητές και καταναλώνει μόνο m από αυτούς. Ένας αλγόριθμος καλείται l-ρεστριστεδ όταν κάθε ένας από τους $M^{(j)}$ καταχωρητές αποτελείται από l ψηφία, δηλαδή μπορεί να αποθηκεύσει οποιονδήποτε ακέραιο μεταξύ 0 και $2^l - 1$. Στο επόμενο θεώρημα φαίνεται με μαθηματικό τρόπο η $\log$-$\log$ ιδιότητα της βασικής πολυπλοκότητας χώρου του αλγορίθμου :

Θεώρημα 2.6.2 *Έστω $\omega(n)$ είναι μια συνάρτηση που τείνει στο άπειρο αυθαίρετα αργά και θεωρείστε τη συνάρτηση $l(n) = \log_2 \log_2 (\frac{n}{m}) + \omega(n)$. Τότε, ο l-restricted αλγόριθμος παρέχει την ίδια έξοδο με πιθανότητα που τείνει στο 1 όταν το n τείνει στο άπειρο.*

Ουσιαστικά, οι βοηθητικοί πίνακες ψηφίων που διατηρούνται από τον αλγόριθμο αποτελούνται από m "μικρά βψτες", που το καθένα έχουν μέγεθος $l(n)$. Με άλλα λόγια, ο συνολικός χώρος που απαιτεί ο αλγόριθμος για να μετρήσει μέχρι το n είναι $m \log_2 \log_2 (\frac{n}{m})(1 + o(1))$. Η

συνάρτηση κατακερματισμού χρειάζεται ακριβώς $2^{l(n)} + \log_2 m$ ψηφία για να αντιστοιχίσει τα στοιχεία του σύμπαντος σε τιμές. Παρατηρείστε ακόμα ότι όποτε ασυμφωνία παρουσιάζεται με την τιμή n καθεαυτή, ο ρεστριστεδ αλγόριθμος αυτόματα παρέχει τη σωστή απάντηση για όλες τις τιμές $\acute{n} \leq n$.

Θεωρείστε για παράδειγμα ότι θέλουμε να μετρήσουμε πλήθος διαφορετικών τιμών που δεν ξεπερνά το 2^{27}, δηλαδή εκατό εκατομμύρια, με ακρίβεια περίπου 4%. Από το θεώρημα 2.4.8, βλέπουμε ότι πρέπει να χρησιμοποιήσουμε $m = 1024 = 2^{10}$. Έπειτα, κάθε υποσύνολο επισκέπτεται χονδρικά $n/m = 2^{17}$ φορές. Οπότε έχουμε $\log_2 \log_2 2^{17} = 4.09$, αυτό ακριβώς που επιθυμούμε δηλαδή. Μετά παίρνουμε $\omega = 0.91$, έτσι ώστε κάθε καταχωρητής να έχει μέγεθος $l = 5$ ψηφία, δηλαδή τιμή πάντα μικρότερη από 32. Εφαρμόζοντας το πάνω όριο για τη συνολική πιθανότητα αποτυχίας βλέπουμε ότι μια l-ρεστριστιον θα επιφέρει πολύ μικρή επιρροή στο αποτέλεσμα : η πιθανότητα ασυμφωνίας είναι μικρότερη από 12%. Επομένως : *Ο βασικός Loglog Counting αλγόριθμος κάνει δυνατή την εκτίμηση του πλήθους διαφορετικών στοιχείων μεγέθους μέχρι και* 10^8 *με τυπικό σφάλμα* 4% *χρησιμοποιώντας* 1024 *καταχωρητές αποτελούμενους από 5 ψηφία ο καθένας, δηλαδή ένα πίνακα 640 βψτες.*

2.7 Time-Decaying Sketches

Μια πολύ διαφορετική προσέγγιση από αυτή των αλγορίθμων που είδαμε σε προηγούμενες ενότητες θα εξετάσουμε σε αυτή την παράγραφο. Η ερευνητική μελέτη που θα παρουσιαστεί εδώ είναι απόληξη της δουλειάς που λαμβάνει χώρα στον τομέα των δικτύων αισθητήρων. Πιο συγκεκριμένα είναι ένα νέο sketch για τη σύνοψη δικτυακών δεδομένων, το οποίο όμως έχει τις ακόλουθες ιδιότητες οι οποίες το χρίζουν πολύ χρήσιμο για αποδοτικής επικοινωνίας συνάθροιση δεδομένων σε *κατανεμημένα σενάρια ροής*, όπως αυτό των δικτύων αισθητήρων :

- *Ανοχή σε διπλότυπα* (duplicate insensitivity), εισαγωγές των ίδιων δεδομένων δεν επηρεάζουν το sketch, και φυσικά τις εκτιμήσεις των συναθροιστικών μεγεθών.
- *Φθορά στο χρόνο* (time decaying), το βάρος ενός αντικειμένου δεδομένων στο sketch μπορεί να μειωθεί με το χρόνο σύμφωνα με μια συνάρτηση φθοράς που ορίζει ο ίδιος

ο χρήστης.

- *Ασύγχρονες αφίξεις δεδομένων* (asynchronous arrivals), τα δεδομένα μπορούν να φτάνουν με οποιαδήποτε σειρά, το sketch δεν επηρεάζεται από τη σειρά με την οποία καταφτάνουν τα δεδομένα προς συνάθροιση.

Τα μεγέθη συνάθροισης για τα οποία μπορεί ο αλγόριθμος να παρέχει αποδεδειγμένα προσεγγιστικές εγγυήσεις είναι διάφορα, στα οποία περιέχονται το *άθροισμα* (sum), ο *μέσος όρος* (mean), *quantiles* (διακριτά ισαπέχοντα σημεία μιας αθροιστικής συνάρτησης κατανομής μιας τυχαίας μεταβλητής), *συχνά στοιχεία* (frequent elements). Αυτό που είχαν ως σκοπό οι συγγραφείς ήταν η παρουσίαση μιας γενικού σκοπού σύνοψης δεδομένων, στην οποία μπορούν να εφαρμοστούν αυθαίρετες *συναρτήσεις φθοράς χρόνου* (time decaying functions), ενώ παραμένει αναίσθητη στα διπλότυπα και μπορεί να χειριστεί ασύγχρονες αφίξεις στοιχείων. Επιπλέον, θα πρέπει να είναι εύκολο να ενημερωθεί με νέες παρατηρήσεις - στοιχεία, να ενωθούν πολλαπλές συνόψεις μαζί και τα ερωτήματα στη σύνοψη να δίνουν εγγυημένης ποιότητας αποκρίσεις σε μια ποικιλία ανάλυσης. Έτσι, αποδεικνύουν ότι αυτό το sketch σύνοψης είναι εύρωστο και το παρουσιάζουν. Οι **πολυπλοκότητες χώρου** και **χρόνου** που χρειάζεται για την ενημέρωση του σκετση είναι ***πολυλογαριθμικές*** στο μέγεθος των δεδομένων. Επιπλέον, πολλαπλά sketches μπορούν να υπολογιστούν σε κατανεμημένα δεδομένα και να συνδυαστούν χωρίς να χαθεί η εγγύηση της ακρίβειας.

Στην επόμενη υποενότητα θα εισάγουμε θεμελιώδη θεωρητικά ζητήματα στα οποία στηρίζεται ο αλγόριθμος και βοηθούν στην ανάλυση και καλύτερη κατανόησή του.

2.7.1 Διατύπωση Προβλήματος

Θεωρείστε μια ροή δεδομένων από παρατηρήσεις ενός μόνο αισθητήρα $R = \langle e_1, e_2, \ldots, e_n \rangle$. Κάθε παρατήρηση $e_i, 1 \leq i \leq n$ είναι μια τετράδα $(v_i, \omega_i, t_i, id_i)$, όπου οι είσοδοι ορίζονται ως εξής :

- v_i, θετική ακέραιη τιμή, ίσως μια παρατήρηση θερμοκρασίας από τον αισθητήρα.

- ω_i, βάρος (weight) συσχετιζόμενο με την παρατήρηση, ίσως ένας αριθμός που αναπαριστά την εμπιστοσύνη στην τελευταία.

- t_i, ακέραια χρονοσφραγίδα (timestamp), τίθεται τη στιγμή που λαμβάνεται η παρατήρηση e_i.

- id_i, μοναδική ταυτότητα της παρατήρησης e_i.

Αυτή η αφαιρετική μορφή του προβλήματος μπορεί να συμπεριληφθεί σε μια ποικιλία κλάσεων οι οποίες μπορούν να κωδικοποιηθούν με αυτόν τον τρόπο. Είναι σκόπιμα τόσο γενική, οι χρήστες έχουν τη δυνατότητα να αναθέσουν τιμές σε αυτά τα πεδία που καλύπτουν τις ανάγκες τους. Για παράδειγμα, αν η επιθυμητή συναθροιστική συνάρτηση είναι ο μέσος όρος θερμοκρασίας όλων των παρατηρήσεων, αυτό μπορεί να επιτευχθεί θέτοντας όλα τα βάρη $\omega_i = 1$ και τις τιμές v_i ως τις πραγματικές μετρήσεις θερμοκρασίας που παρατηρήθηκαν. Η μοναδική ταυτότητα id_i μιας παρατήρησης μπορεί να οριστεί ως ο συνδυασμός της μοναδικής ταυτότητας του αισθητήρα και της χρονοσφραγίδας της παρατήρησης.

Είναι δυνατόν η ίδια παρατήρηση να εμφανιστεί πολλές φορές στη ροή δεδομένων, με την ίδια ταυτότητα, τιμή και χρονοσφραγίδα - οι επαναλαμβανόμενες εμφανίσεις δεν πρέπει να συμπεριληφθούν όταν υπολογίζονται συναθροιστικές συναρτήσεις για τη ροή δεδομένων. Πρέπει να τονιστεί ότι το μοντέλο αυτό επιτρέπει στοιχεία της ροής να έχουν διαφορετικές ταυτότητες, αλλά με τις ίδιες τιμές ή χρονοσφραγίδες - σε αυτήν την περίπτωση θα θεωρηθούν ως διαφορετικές παρατηρήσεις και θα υπολογιστούν στις συναθροιστικές συναρτήσεις.

Θεωρούμε *ασύγχρονες* ροές δεδομένων, όπου τα στοιχεία δεν καταφθάνουν απαραίτητα με τη σειρά των χρονοσφραγίδων τους. Ο χειρισμός των μη συγχρονισμένων αφίξεων είναι ουσιαστικός και σημαντικός λόγω της multi-path δρομολόγησης, όπως και της ανάγκης χειρισμού των ενώσεων πολλαπλών sketches. Σημειώστε ότι η e_{i+1} λαμβάνεται μετά την e_i, και η e_n είναι η πιο πρόσφατη παρατήρηση. Στην περίπτωση υποστήριξης ασύγχρονων αφίξεων είναι δυνατόν $i > j$, ώστε η e_i να ληφθεί μετά την e_j, αλλά $t_i < t_j$.

Ορισμός 2.7.1 *Μια συνάρτηση $f(x)$ είναι **συνάρτηση φθοράς** (decay function) αν :*

1. *για κάθε* x, $f(x) \geq 0$ *και*

2. *η* $f(x)$ *είναι φθίνουσα, δηλαδή* $x_1 \geq x_2 \Rightarrow f(x_1) \leq f(x_2)$.

Το αποτέλεσμα της συνάρτησης φθοράς εφαρμόζεται στο *βάρος* ενός στοιχείου. Ακριβέστερα, το *decayed weight* του στοιχείου (v, ω, t, id) τη χρονική στιγμή $c \geq t$ είναι $\omega \cdot f(c-t)$. Ένα παράδειγμα συνάρτησης φθοράς είναι το μοντέλο του *κυλιόμενου παραθύρου* (sliding window model), όπου η $f(x)$ ορίζεται ως ακολούθως :

Ορισμός 2.7.2 *Για κάποιο μέγεθος παραθύρου (window size)* W *:*

$$f(x) = \begin{cases} 1, \text{ αν } x \leq W \\ 0, \text{ αλλιώς} \end{cases}$$

Άλλες αρκετά χρησιμοποιημένες συναρτήσεις φθοράς περιλαμβάνουν τη συνάρτησης εκθετική φθοράς $f(x) = e^{-\alpha x}$ για σταθερά α, και τη συνάρτηση πολυωνυμικής φθοράς $f(x) = x^{-\alpha}$.

Ορισμός 2.7.3 *Μια* ***ακέραια*** *συνάρτηση φθοράς είναι αυτή της οποίας το decayed weight* $\omega \cdot f(c-t)$ *είναι πάντα ακέραιος αριθμός.*

Παρατηρείστε ότι μια συνάρτηση κυλιόμενου παραθύρου είναι εξορισμού ακέραια συνάρτηση φθοράς, αλλά και συναρτήσεις που δεν είναι εξορισμού ακέραιες μπορούν με κατάλληλες πράξεις στρογγυλοποίησης και κλιμάκωσης να γίνουν.

2.7.2 Συναθροιστικές Συναρτήσεις

Συμβολίζουμε με $f(\cdot)$ μια συνάρτηση φθοράς, και με c τη χρονική στιγμή την οποία τίθεται ένα ερώτημα (query). Έστω ότι το σύνολο των διαφορετικών παρατηρήσεων που ανήκουν στο R (το πολυσύνολο των παρατηρήσεων) συμβολίζεται με D. Παρακάτω δίνονται οι συναρτήσεις που θεωρούμε :

1. Το ***decayed sum*** την χρονική στιγμή c ορίζεται ως :

$$V = \sum_{(v,\omega,t,id)\in D} \omega \cdot f(c-t)$$

Η συνάρτηση αυτή αναπαριστά το άθροισμα των decayed weights όλων των στοιχείων της ροής δεδομένων. Για παράδειγμα, υποθέστε ότι κάθε αισθητήρας δημοσιεύει μία ένδειξη θερμοκρασίας κάθε λεπτό, και ενδιαφερόμαστε να εκτιμήσουμε τη μέση θερμοκρασία όλων των ενδείξεων που δημοσιεύτηκαν τα τελευταία 90 λεπτά. Αυτό μπορεί να υπολογιστεί ως ο λόγος του αθροίσματος των τιμών των παρατηρήσεων των τελευταίων 90 λεπτών, προς το συνολικό αριθμό των παρατηρήσεων στα τελευταία 90 λεπτά. Για να υπολογιστεί το άθροισμα του αριθμητή, θεωρούμε μια ροή δεδομένων όπου τα decayed weights ω_i ισούνται με την τιμή της θερμοκρασίας, και το άθροισμα υπολογίζεται χρησιμοποιώντας για συνάρτηση φθοράς τη συνάρτηση κυλιόμενου παραθύρου μεγέθους (W) 90 λεπτών. Για τον υπολογισμό του πλήθους των παρατηρήσεων, θεωρούμε μια ροή δεδομένων όπου τα decayed weights ω_i ισούνται με 1, και το decayed sum υπολογίζεται και πάλι μέσω συνάρτησης κυλιόμενου παραθύρου μεγέθους 90 λεπτών.

2. Ανεπίσημα, το ***decayed*** ϕ***-quantile*** τη χρονική στιγμή c είναι μια τιμή n τέτοια ώστε το συνολικό decayed weight όλων των στοιχείων στο D των οποίων η τιμή είναι μικρότερη ή ίση από n, είναι ϕ-κλάσμα του συνολικού decayed weight. Για παράδειγμα, σε μια διάταξη όπου οι αισθητήρες δημοσιεύουν μετρήσεις θερμοκρασιών, κάθε μέτρηση μπορεί να έχει ένα *επίπεδο εμπιστοσύνης* (confidence level) συσχετισμένο με αυτή, το οποίο ανατίθεται από τον αισθητήρα. Ο χρήστης μπορεί να ενδιαφέρεται για το decayed mean θερμοκρασίας, όπου το decayed weight είναι αρχικά το επίπεδο εμπιστοσύνης και φθείρεται με το πέρασμα του χρόνου. Αυτό μπορεί να επιτευχθεί θέτοντας την τιμή v ίση με την τιμή της θερμοκρασίας, το αρχικό decayed weight ω ίσο με το επίπεδο εμπιστοσύνης, $\phi = 0.5$ και χρησιμοποιώντας μια κατάλληλη συνάρτηση φθοράς χρόνου.

Αφού όμως, ο υπολογισμός ακριβών ϕ-quantiles (ακόμη και στην περίπτωση χωρίς decayed weights) σε ένα μόνο πέρασμα των δεδομένων αποδεδειγμένα παίρνει γραμμικό

χώρο του μεγέθους των δεδομένων, θεωρούμε προσεγγιστικά ϕ-quantiles. Ο ορισμός μας είναι κατάλληλος στην περίπτωση που οι τιμές είναι ακέραιοι αριθμοί και που θα μπορούσαν να υπάρχουν πολλαπλά στοιχεία με την ίδια τιμή στο D.

Ορισμός 2.7.4 *Η **σχετική τάξη** (relative rank) μιας τιμής v στο D τη χρονική στιγμή c ορίζεται ως :*

$$\frac{\sum_{(v,\omega,t,id)\in D|v\leq u}\omega\cdot f(c-t)}{\sum_{(v,\omega,t,id)\in D}\omega\cdot f(c-t)}$$

Ορισμός 2.7.5 *Για μια μεταβλητή που ορίζεται από τον χρήστη $0<\epsilon<\phi$, το ϵ-**approximate decayed** ϕ-**quantile** είναι μια τιμή n τέτοια ώστε η σχετική τάξη της n να είναι τουλάχιστον $\phi-\epsilon$ και η σχετική τάξη της $n-1$ να είναι μικρότερη από $\phi+\epsilon$.*

3. ***Συχνά στοιχεία***. Έστω ότι η (decayed) σχετική συχνότητα της εμφάνισης μιας τιμής v τη συγκεκριμένη χρονική στιγμή c ορίζεται ως :

$$\psi(v)=\frac{\sum_{(v,\omega,t,id)\in D|v=u}\omega\cdot f(c-t)}{\sum_{(v,\omega,t,id)\in D}\omega\cdot f(c-t)}$$

Τα συχνά στοιχεία είναι αυτά των οποίων οι τιμές n είναι τέτοιες ώστε $\psi(n)>\phi$, για κάποια σταθερά ϕ, πχ. $\phi=2\%$. Η ακριβής εκδοχή των συχνών στοιχείων απαιτεί τη συχνότητα όλων των αντικειμένων να βρεθεί ακριβώς, το οποίο είναι αποδεδειγμένα ακριβό σε πολύ μικρό χώρο. Έτσι, θεωρούμε το ϵ-approximate συχνών στοιχείων πρόβλημα, το οποίο απαιτεί την επιστροφή των όλων των τιμών n τέτοιων ώστε $\psi(n)>\phi$ και καμία τιμή $\acute{n}$ τέτοια ώστε $\psi<\phi-\epsilon$.

4. ***Selectivity estimation*** ερώτημα. Είναι το ερώτημα που δεδομένου ενός κατηγορήματος (predicate) $P(v,\omega)$ το οποίο επιστρέφει 0 ή 1, υπολογίζει το Q που ορίζεται ως εξής :

$$Q=\frac{\sum_{(v,\omega,t,id)\in D}\omega P(v,\omega)f(c-t)}{\sum_{(v,\omega,t,id)\in D}\omega\cdot f(c-t)}$$

Ανεπίσημα, η εκλεκτικότητα ενός κατηγορήματος $P(\upsilon, \omega)$ είναι ο λόγος του συνολικού decayed weight όλων των δεδομένων της ροής που ικανοποιούν το κατηγόρημα P, προς το συνολικό decayed weight όλων των στοιχείων. Σημειώστε ότι, $0 \leq Q \leq 1$. Το ϵ-*approximate selectivity estimation* πρόβλημα επίκειται στην επιστροφή μιας τιμής $\hat{Q}$ τέτοιας ώστε $|\hat{Q} - Q| \leq \epsilon$.

Ο ακριβής υπολογισμός του duplicate insensitive decayed sum με τη χρήση μιας γενικής ακέραιας συνάρτησης φθοράς είναι αδύνατος σε μικρό χώρο, ακόμη και σε μη-κατανεμημένη διάταξη. Αν μπορούμε να υπολογίσουμε ακριβώς ένα duplicate insensitive άθροισμα, μπορούμε να εισάγουμε ένα στοιχείο e και να δούμε αν το άθροισμα αλλάζει. Η απάντηση καθορίζει αν το e έχει ήδη παρατηρηθεί ή όχι. Αφού είναι δυνατό να ανακατασκευάσουμε όλα τα ήδη παρατηρημένα στοιχεία στη ροή δεδομένων μέχρι αυτή τη χρονική στιγμή, τότε ένα ανάλογο sketch χρειάζεται γραμμικό χώρο στο μέγεθος των δεδομένων στη χειρότερη περίπτωση. Αυτό το όριο του γραμμικού χώρου ισχύει ακόμα και για ένα sketch που μπορεί να δώσει ακριβείς απαντήσεις με μια δ πιθανότητα σφάλματος, για $\delta < 1/2$, και για ένα sketch που μπορεί να δώσει ντετερμινιστικές προσεγγίσεις. Τέτοια κάτω όρια για ντετερμινιστικές προσεγγίσεις ισχύουν και για quantiles και συχνά στοιχεία στο duplicate insensitive μοντέλο. Επομένως, ψάχνουμε για τυχαία κατασκευασμένες προσεγγίσεις των συναθροιστικών συναρτήσεων που παρουσιάστηκαν παραπάνω. Ως αποτέλεσμα, όλες οι εγγυήσεις είναι της μορφής :

"Με πιθανότητα τουλάχιστον $1 - \delta$, η εκτίμηση είναι μια ϵ-approximation στην επιθυμητή συναθροιστική συνάρτηση."

2.7.3 Συναθροιστικές Συναρτήσεις με Ακέραια Συνάρτηση Φθοράς

Σε αυτή την υποενότητα, θα δούμε ένα sketch για duplicate insensitive decayed συνάθροιση με τη χρήση μιας συνάρτησης φθοράς χρόνου $f()$, τέτοιας ώστε τα decayed weights $\omega \cdot f(c-t)$ να είναι πάντα ακέραιοι αριθμοί. Θα δώσουμε μια γενική διαισθητική περιγραφή και έπειτα θα προχωρήσουμε στην τυπική περιγραφή του σκετση και τον υπολογισμό συναθροιστικών

συναρτήσεων δεδομένου του sketch.

Υψηλού Επιπέδου Περιγραφή

Θυμίζουμε ότι με R συμβολίζουμε το σύνολο των στοιχείων της ροής δεδομένων και με D το σύνολο των διαφορετικών στοιχείων στο R. Μέσω του sketch παρέχονται εκτιμήσεις για διάφορες συναθροιστικές συναρτήσεις, αλλά για την υψηλού επιπέδου περιγραφή, υποθέτουμε ότι η εργασία που θέλουμε να πραγματοποιήσουμε είναι να πάρουμε την απάντηση σε ένα ερώτημα για το decayed sum των στοιχείων του D τη χρονική στιγμή κ, δηλαδή

$$V = \sum_{(v,\omega,t,id)\in D} \omega \cdot f(\kappa - t).$$

Έστω ω_{max} είναι το γινόμενο της $f(0)$ με το μέγιστο βάρος των στοιχείων και id_{max} η μέγιστη τιμή ταυτότητας. Θεωρείστε την ακόλουθη υποθετική διαδικασία, η οποία λαμβάνει χώρα τη στιγμή του ερωτήματος κ. Τονίζουμε ότι αυτή η διαδικασία δεν εκτελείται από τον αλγόριθμο αλλά θα βοηθήσει στην κατανόηση του αλγορίθμου. Για κάθε διαφορετικό στοιχείου της ροής δεδομένων $e = (v, \omega, t, id)$, ένα διάστημα ακεραίων ορίζεται ως εξής :

$$r_e^\kappa = [\omega_{max} \cdot id, \omega_{max} \cdot id + \omega \cdot f(\kappa - t) - 1]$$

Παρατηρείστε ότι το μέγεθος των διαστημάτων είναι ακριβώς $\omega \cdot f(\kappa - t)$. Επιπλέον, αν το ίδιο στοιχείο e εμφανιστεί ξανά στη ροή, ένα πανομοιότυπο διάστημα ορίζεται, και για στοιχεία με διαφορετικές ταυτότητες τα διαστήματα είναι ανεξάρτητα μεταξύ τους. Επομένως, καταλήγουμε στην ακόλουθη παρατήρηση :

Παρατήρηση 2.7.6

$$\sum_{e=(v,\omega,t,id)\in D} \omega \cdot f(\kappa - t) = \left| \bigcup_{e\in R} r_e^\kappa \right|$$

Οι ακέραιοι στο r_e^κ τοποθετούνται σε τυχαία δείγματα $T_0, T_1, \ldots, T_M$ με διαδικασία που περιγράφεται αμέσως παρακάτω, όπου M είναι της τάξης

$\log(\omega_{max} \cdot id_{max})$ το οποίο θα οριστεί ακριβέστερα αργότερα. Κάθε ακέραιος στο r_e^κ τοποθετείται στο δείγμα T_0. Για $i = 0 \dots M-1$, κάθε ακέραιος στο δείγμα T_i τοποθετείται στο T_{i+1} με πιθανότητα περίπου $1/2$ (η πιθανότητα δεν είναι ακριβώς $1/2$ εξαιτίας της φύσης των συναρτήσεων δειγματοληψίας, οι οποίες θα οριστούν στη συνέχεια). Η πιθανότητα ένας ακέραιος να βρίσκεται στο δείγμα T_i είναι $p_i \approx 1/2^i$. Έπειτα, το decayed sum, V μπορεί να εκτιμηθεί χρησιμοποιώντας το T_i ως τον αριθμό των ακεραίων που έχουν επιλεχθεί στο T_i, πολλαπλασιασμένο με $1/p_i$. Αξίζει να σημειωθεί ότι η μέση τιμή της εκτίμησης χρησιμοποιώντας το T_i είναι V για κάθε i, και επιλέγοντας ένα "αρκετά μικρό" i μπορούμε να πάρουμε μια εκτίμηση για τη V η οποία είναι πολύ κοντά στην πραγματική της τιμή με μεγάλη πιθανότητα.

Ας εξηγήσουμε τώρα πως ο αλγόριθμος προσομοιώνει τη διαδικασία αυτή κάτω από δριμύτατο περιορισμό χώρου και αφίξεις των στοιχείων της ροής δεδομένων σε πραγματικό χρόνο. Η λανθασμένη μέτρηση λόγω διπλότυπων αποφεύγεται μέσω δειγματοληψίας βασισμένη σε μια συνάρτηση κατακερματισμού h, η οποία ορίζεται στη συνέχεια. Αν ένα στοιχείο e εμφανιστεί ξανά στη ροή, τότε το ίδιο διάστημα ακεραίων r_e^κ θα οριστεί και η συνάρτηση κατακερματισμού οδηγεί ακριβώς στην ίδια απόφαση με πριν όσον αφορά το αν θα τοποθετηθεί κάθε ακέραιος στο δείγμα T_i. Επομένως, αν ένα στοιχείο εμφανιστεί παραπάνω από μία φορές, είτε επιλέγεται κάθε φορά για προσθήκη στο δείγμα είτε δεν επιλέγεται ποτέ.

Ένα άλλο ζήτημα είναι ότι για ένα στοιχείο $e = (v, \omega, t, id)$, το μήκος του οριζόμενου διαστήματος r_e^κ είναι $\omega \cdot f(\kappa - t)$, το οποίο μπορεί να είναι πολύ μεγάλο. Αν δειγματοληπτήσουμε κάθε έναν από τους ακεραίους στο r_e^κ ξεχωριστά, αυτή η διαδικασία απαιτεί τον υπολογισμό της συνάρτησης κατακερματισμού $\omega \cdot f(\kappa - t)$ φορές για κάθε δείγμα, και αυτός είναι πολύ ακριβός συναρτήσει του χρόνου υπολογισμός, εκθετικός στο μέγεθος της εισόδου. Για αυτό το λόγο, χρησιμοποιούμε μια πολύ αποδοτική σε σχέση με το χρόνο τεχνική, η οποία καλείται *Range-Sampling* και κάνει δυνατή την δειγματοληψία ολόκληρων των διαστημάτων r_e^κ γρήγορα σε χρόνο $O(\log |r_e^\kappa|)$. Για να επιτευχθεί αυτό, η τεχνική αυτή εκμεταλλεύεται την παρούσα δομή μιας *pairwise idependent συνάρτηση κατακερματισμού* h. Πιο συγκεκριμένα, όλοι οι ακέραιοι στο r_e^κ οι οποίοι έχουν επιλεχθεί στο T_i αποθηκεύονται μαζί απλά αποθηκεύοντας το στοιχείο e στο T_i όταν τουλάχιστον ένας ακέραιος του r_e^κ επιλέγεται για εισαγωγή

στο T_i.

Ακόμα όμως και με την τεχνική *Range-Sampling* είναι αδύνατο για τον αλγόριθμο να υπολογίσει και να αποθηκεύσει τα δείγματα T_i με τον τρόπο που περιγράφηκε προηγουμένως. Καταρχήν, τη στιγμή του ερωτήματος κ, όταν τα βάρη, $\omega \cdot f(\kappa - t)$, δεν είναι γνωστά τη στιγμή που καταφθάνει το στοιχείο στη ροή δεδομένων. Για την αντιμετώπιση του προβλήματος αυτού, τονίζουμε ότι το βάρος τη χρονική στιγμή c είναι $\omega \cdot f(c-t)$, η οποία είναι γνησίως φθίνουσα όσον αφορά το c. Είναι δυνατόν να ορίσουμε ένα *χρόνο λήξης* (expiry time) για το στοιχείο e στο επίπεδο i, $expiry(e, i)$ τέτοιο ώστε εφόσον $c < expiry(e, i)$ το διάστημα r_e^c έχει τουλάχιστον έναν ακέραιο επιλεγμένο στο T_i, και για $c \geq expiry(e, i)$, το r_e^c δεν έχει κανέναν ακέραιο στο T_i. Με αυτόν τον τρόπο, μπορούμε να προσκολλήσουμε στο e το χρόνο λήξης του τη στιγμή που αυτό φτάνει στη ροή δεδομένων, και να το διατηρήσουμε στο T_i εφόσον η παρούσα χρονική στιγμή είναι μικρότερη από το $expiry(e, i)$. Για οποιοδήποτε μελλοντικό ερώτημα που τίθεται σε χρόνο $k \geq expiry(e, i)$, σε μια εκτίμηση για το άθροισμα χρησιμοποιώντας το δείγμα T_i δεν θα ληφθεί υπόψη το στοιχείο e.

Επόμενο θέμα είναι ότι για μικρές τιμές του i, το T_i μπορεί να είναι πολύ μεγάλο και επομένως να απαιτεί πολύ χώρο. Για αυτόν ακριβώς το λόγο ο αλγόριθμος αποθηκεύει μόνο τα υποσύνολα S_i, τα οποία περιέχουν το πολύ τ στοιχεία από το T_i με τους μεγαλύτερους χρόνους λήξης, ενώ τα υπόλοιπα στοιχεία απορρίπτονται (το τ είναι σχεδιαστική παράγραφος που εξαρτάται από την επιθυμητή ακρίβεια των αποτελεσμάτων). Αξίζει να αναφερθεί ότι τα τ στοιχεία με τους μεγαλύτερους χρόνους λήξης μιας ροής δεδομένων μπορούν εύκολα να διατηρηθούν σε ένα μόνο πέρασμα της ροής σε χώρο $O(\tau)$. Άρα, *τα δείγματα που αποθηκεύονται από τον αλγόριθμο είναι τα* $S_0, S_1, \ldots, S_M$.

Παράδειγμα Υπολογισμού Decayed Sum

Θα απλοποιήσουμε τον ορισμό του στοιχείου e σε μια τριάδα παραμέτρων ως εξής, $e = (\omega, t, id)$, αφού δεν θα χρειαστεί η τιμή v. Η ροή εισόδου $e_1, e_2, \ldots, e_8$ φαίνεται στο πάνω μέρος του σχήματος 2.2. Υποθέτουμε μια συνάρτηση φθοράς χρόνου κατά την οποία το decayed weight του στοιχείου (ω_i, t_i, id_i) τη χρονική στιγμή t είναι $\omega_i^t = \lfloor \frac{\omega}{t-t_i} \rfloor$. Το σχήμα 2.2

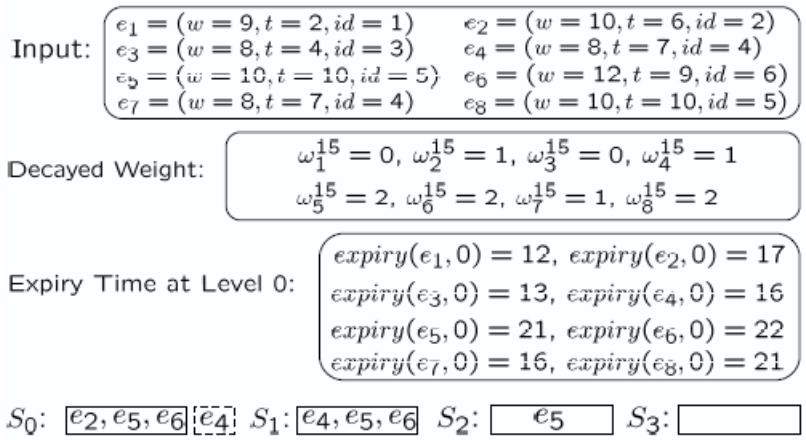
απεικονίζει μόνο τους χρόνους λήξης για το επίπεδο 0, υποθέτουμε ότι η παρούσα χρονική στιγμή είναι $c = 15$.

Τη χρονική στιγμή αυτή, τα στοιχεία e_1 και e_3 έχουν λήξει στο επίπεδο 0, γεγονός που υπονοεί ότι έχουν λήξει και στα υπόλοιπα επίπεδα. Τα e_7 και e_8 δεν εμφανίζονται στο sketch γιατί είναι διπλότυπα των e_4 και e_5 αντίστοιχα. Ανάμεσα στα υπόλοιπα στοιχεία e_2, e_4, e_5, e_6 μόνο τα $\tau = 3$ με τους μεγαλύτερους χρόνους λήξεις διατηρούνται στο S_0, επομένως το e_4 απορρίπτεται. Από το σύνολο e_2, e_4, e_5, e_6, το υποσύνολο e_4, e_5, e_6 περνά (με τυχαίο τρόπο) στο επόμενο επίπεδο S_1 βάσει των κατακερματισμένων τιμών των ακεραίων στο $r_{15}^{e_i}$ (δηλαδή, $expiry(e_4, 1) > 15$, $expiry(e_5, 1) > 15$, $expiry(e_6, 1) > 15$ και $expiry(e_2, 1) \leq 15$) και εφόσον υπάρχει χώρος αποθηκεύονται και τα τρία στοιχεία στο S_1. Έπειτα, μόνο το e_5 επιλέγεται στο S_2 και κανένα στοιχείο για το S_3.

Όταν ένα ερώτημα τίθεται για το decayed sum τη χρονική στιγμή 15, ο αλγόριθμος βρίσκει το μικρότερο επίπεδο l στο οποίο το δείγμα S_l δεν έχει απορρίψει κανένα στοιχείο με χρόνο λήξης μεγαλύτερο από 15. Για παράδειγμα, στο σχήμα 2.2, το επίπεδο αυτό είναι το $l = 1$. Παρατηρείστε ότι σε αυτό το επίπεδο ισχύει $S_l = T_l$, και έτσι το S_l μπορεί να χρησιμοποιηθεί για να δόση έγκυρη απάντηση στο ερώτημα για το V. Παίρνοντας διαισθητικά την απόφαση επιλογής του μικρότερου επιπέδου l, βλέπουμε ότι όσο μεγαλύτερο είναι το δείγμα τόσο καλύτερη θα είναι και η εκτίμηση.

Τυπική Περιγραφή

Σε αυτή την υποενότητα δείχνουμε πως θα κατασκευαστούν και θα διατηρηθούν τα δείγματα $S_0, S_1, \ldots, S_M$. Έστω h μια pairwise independent συνάρτηση κατακερματισμού επιλεγμένη από μια 2-universal οικογένεια συναρτήσεων κατακερματισμού που ορίζεται ως ακολούθως. Έστω $Y = \omega_{max}(id_{max} + 1)$. Το πεδίο ορισμού της h είναι το $[1 \ldots Y]$. Επιλέγεται ένας πρώτος αριθμός p, τέτοιος ώστε $10Y < p < 20Y$, και δύο αριθμοί α και b επιλεγμένοι ομοιόμορφα στην τύχη από το διάστημα $0, \ldots, p-1$. Τότε η συνάρτηση κατακερματισμού $h : 1 \ldots Y \to 0, \ldots, p-1$ ορίζεται ως $h(x) = (\alpha \cdot x + b) \mod p$. Η συνάρτηση $RangeSample(r, i)$, ορισμένη ακριβώς στο [], παίρνει σαν ορίσματα ένα διάστημα ακεραίων $r \subseteq [1, Y]$ και ένα



Σχήμα 2.2: Παράδειγμα με 8 στοιχεία εισόδου και το sketch της ροής S_0, S_1, S_2, S_3 για το decayed sum. Η παρούσα χρονική στιγμή είναι 15 και τα decayed weights του στοιχείου e_i τη χρονική στιγμή t συμβολίζονται με ω_i^t. Το στοιχείο e_4 στο διακεκομμένο κουτί υποδηλώνει ότι απορρίφθηκε από το S_0 εξαιτίας υπερχείλισης.

επίπεδο δειγματοληψίας $i \in [0, M]$, και επιστρέφει τον αριθμό των σημείων $x \in r$ έτσι ώστε $h(x) \in 0, \ldots, \lfloor 2^{-i}p \rfloor - 1$

Υπολογισμός της $expiry(e, i)$ Για οποιοδήποτε στοιχείο $e = (v, \omega, t, id)$ και επίπεδο $0 \leq i \leq M$, η συνάρτηση $expiry(e, i)$ επιστρέφει τη χρονική στιγμή $\bar{t}$ όταν για $c < \bar{t}$, $RangeSample(r_e^c, i) > 0$ και για $c \geq \bar{t}$, $RangeSample(r_e^c, i) = 0$. Θυμίζουμε ότι :

$$r_e^c = [\omega_{max} \cdot id, \omega_{max} \cdot id + \omega \cdot f(c - t) - 1]$$

Αφού η $f(c - t)$ είναι φθίνουσα συνάρτηση για το c, το διάστημα r_e^c συρρικνώνεται όσο το c αυξάνεται. Εφόσον η $RangeSample(r_e^c, i)$ είναι ο ακριβής αριθμός των ακεραίων του r_e^c που επιλέγονται στο T_i από τη συνάρτηση κατακερματισμού, είναι και αυτή φθίνουσα για το c. Αν $\lim_{t\to\infty} f(t) = f_{min} > 0$, και $RangeSample(r_e^\infty, i) > 0$, τότε ορίζουμε $expiry(e, i) = \infty$, δηλαδή το e δεν μπορεί να απορριφθεί από το επίπεδο i ποτέ. Ομοίως, αν $RangeSample(r_e^0, i) = 0$, τότε $expiry(e, i) = 0$, και το στοιχείο e δεν εισχωρεί στο επίπεδο i

ποτέ. Διαφορετικά, μπορούμε να υπολογίσουμε το $\bar{t}$ χρησιμοποιώντας δυαδικό ψάξιμο στο διάστημα των πιθανών χρόνων $[t, t + tmax]$, όπου t_{max} είναι το μικρότερο $\acute{t}$ ώστε $f(\acute{t}) = f_{min}$.

Για να ελέγξουμε αν το e δεν λήγει ποτέ, πραγματοποιούμε μία μόνο κλήση στη συνάρτηση $RangeSample$, αλλιώς κάνουμε $O(\log \bar{t}) = O(\log t_{max})$ υπολογισμούς της $RangeSample$. Η πολυπλοκότητα χρόνου υπολογισμού της $expiry(e, i)$ είναι λοιπόν $O(\log t_{max} \log \omega_{max})$, αφού το $O(\log \omega_{max})$ φράζει το κόστος της $RangeSample$. Το σκετση S για μια ακεραία συνάρτηση φθοράς είναι το σύνολο των ζευγών (S_i, t_i), για $i = 0 \ldots M$, όπου S_i είναι το δείγμα και t_i είναι ο μεγαλύτερος χρόνος λήξης από στοιχείο που έχει απορριφθεί από το S_i μέχρι εκείνη τη στιγμή. Στο σχήμα 2.3 φαίνονται σε μορφή ψευδοκώδικα τα βήματα αρχικοποίησης, ενημέρωσης του sketch και συνδυασμού δύο sketches.

Initialization:

1. Randomly choose a hash function h as described in Section 3.2
2. For $0 \leq i \leq M$, $S_i \leftarrow \emptyset$, $t_i \leftarrow -1$.
   // t_i is the maximum expiry time among all the elements discarded so far at level i

When a new item $e = (v, w, t, id)$ arrives:

For $0 \leq i \leq M$

1. If $(e \in S_i)$, then return; *// e is a duplicate.*
2. If $(\text{expiry}(e, i) > \max\{c, t_i\})$
   (a) $S_i \leftarrow S_i \cup \{e\}$
   (b) If $|S_i| > \tau$ then *// overflow*
      i. $t_i \leftarrow \min_{e \in S_i} \text{expiry}(e, i)$
      ii. $S_i \leftarrow S_i \backslash \{e | \text{expiry}(e, i) = t_i\}$

To merge two sketches S, S':

For $0 \leq i \leq M$

1. $S_i \leftarrow S_i \cup S'_i$
2. $t_i \leftarrow \max\{t_i, t'_i\}$
3. While $|S_i| > \tau$ do
   (a) $t_i \leftarrow \min_{e \in S_i} \text{expiry}(e, i)$
   (b) $S_i \leftarrow S_i \backslash \{e | \text{expiry}(e, i) = t_i\}$

Σχήμα 2.3: Γενικό sketch του αλγορίθμου με ακέραια συνάρτηση μεταφοράς.

Τα παρακάτω λήμματα δείχνουν τις ιδιότητες του sketch τις οποίες ισχυρίζονται οι συγγραφείς.

Λήμμα 2.7.7 *Το δείγμα S_i είναι ανεκτικό στη διάταξη των στοιχείων (order insensitive). Δεν επηρεάζεται από παραλλαγές στη σειρά άφιξης των στοιχείων της ροής. Είναι επίσης ανεκτικό σε διπλότυπα, αν ένα στοιχείο παρατηρηθεί πολλαπλές φορές το δείγμα είναι το ίδιο σαν να είχε παρατηρηθεί μία μόνο φορά.*

Λήμμα 2.7.8 *Υποθέστε ότι δύο δείγματα S_i και $\acute{S}_i$ κατασκευάστηκαν μέσω της ίδιας συνάρτησης κατακερματισμού h σε δύο διαφορετικές ροές δεδομένων R και $\acute{R}$ αντίστοιχα. Τότε τα S_i και $\acute{S}_i$ μπορούν να συνενωθούν για να δημιουργήσουν ένα δείγμα της ροής $R \cup \acute{R}$.*

Λήμμα 2.7.9 *(Πολυπλοκότητες χώρου και χρόνου). Η πολυπλοκότητα χώρου του sketch για ακέραια συνάρτηση φθοράς είναι $O(M\tau)$ μονάδες, όπου κάθε μονάδα είναι μια παρατήρησης (v, ω, t, id). Ο αναμενόμενος χρόνος για κάθε ενημέρωση είναι $O(\log \omega(\log \tau + \log t_{max} \log \omega_{max}))$. Η συνένωση δύο sketches παίρνει $O(M\tau)$ χρόνο.*

Υπολογίζοντας Decayed Μεγέθη Χρησιμοποιώντας το Sketch

Θα περιγράψουμε τώρα πως υπολογίζονται διάφορες συναθροιστικές συναρτήσεις χρησιμοποιώντας το sketch S. Για $i = 0 \dots M$, έστω $p_i = \frac{\lfloor p 2^{-i} \rfloor}{p}$ η πιθανότητα δειγματοληψίας στο επίπεδο i.

Decayed Sum Για τον υπολογισμό του decayed sum :

$$V = \sum_{(v,\omega,t,id) \in D} \omega \cdot f(c - t)$$

ορίζουμε το μέγιστο μέγεθος ενός δείγματος ως $\tau = 60/\epsilon^2$, και τον μέγιστο αριθμό επιπέδων ως $M = \lceil \log \omega_{max} + \log id_{max} \rceil$.

Θεώρημα 2.7.10 *Ο αλγόριθμος του σχήματος 2.4 δίνει εκτιμητή $\hat{V}$ του V τέτοιο ώστε $Pr[|\hat{V} - V| < \epsilon V] > \frac{2}{3}$. Ο χρόνος που παίρνει η απάντηση ενός ερωτήματος για το συγκεκριμένο άθροισμα είναι $O(\log M + \frac{1}{\epsilon^2} \log \omega_{max})$. Ο αναμενόμενος χρόνος για κάθε ενημέρωση είναι $O(\log \omega(\log \frac{1}{\epsilon} + \log t_{max} \log \omega_{max}))$. Η πολυπλοκότητα χώρου είναι $O(\frac{1}{\epsilon^2}(\log \omega_{max} \log id_{max}))$.*

When queried for the decayed sum at time c:

1. $\ell = \min\{i | 0 \leq i \leq M, t_i \leq c\}$
2. If ℓ does not exist, then the algorithm fails, return.
3. If ℓ exists, then return $\frac{1}{p_\ell} \sum_{e \in S_\ell} \text{RangeSample}(r_e^c, \ell)$

Σχήμα 2.4: Decayed sum για συνάρτηση φθοράς κυλιόμενου παραθύρου.

Σημειώνουμε ότι κανονικά θέλουμε μια εγγύηση ότι η πιθανότητα σφάλματος είναι $\delta \ll \frac{1}{3}$. Για να δοθεί τέτοια εγγύηση μπορούμε να κρατάμε $\Theta(\log 1/\delta)$ ανεξάρτητες αντιγραφές του sketch βασισμένες σε διαφορετικές συναρτήσεις κατακερματισμού, και να πάρουμε το μέσο όρο των εκτιμήσεων. Ένα τυπικό *Chernoff* φράγμα δείχνει ότι ο μέσος όρος εκτίμησης είναι έγκυρος στο ϵV με πιθανότητα τουλάχιστον $1 - \delta$.

Selectivity Estimation Για τον υπολογισμό του selectivity estimation, χρησιμοποιούμε το sketch για να βρούμε ένα δείγμα S_l τέτοιο ώστε το δείγμα του μικρότερου επιπέδου που δεν έχει απορρίψει κανένα στοιχείο του οποίου ο χρόνος λήξης ξεπερνά το c, και υπολογίζεται το P στα στοιχεία του S_l. Ο συγκεκριμένος αλγόριθμος δίνεται στο σχήμα 2.5. Εδώ, επιλέγουμε $\tau = 492/\epsilon^2$ και $M = \lceil \log \omega_{max} + \log id_{max} \rceil$, έτσι καταλήγουμε στο παρακάτω θεώρημα :

When queried for the selectivity of P at time c:

1. $\ell = \min\{i | 0 \leq i \leq M, t_i \leq c\}$
2. If ℓ does not exist, then the algorithm fails, return.
3. If ℓ exists, then return
$$\frac{\sum_{e=(v,w,t,id) \in S_\ell} \text{RangeSample}(r_e^c, \ell) \cdot P(v,w)}{\sum_{e \in S_\ell} \text{RangeSample}(r_e^c, \ell)}$$

Σχήμα 2.5: Selectivity estimation για ακέραια συνάρτηση φθοράς.

Θεώρημα 2.7.11 *Ο αλγόριθμος του σχήματος 2.5 δίνει εκτιμητή $\hat{Q}$ του Q τέτοιο ώστε $Pr[|\hat{Q} - Q| < \epsilon] > 2/3$. Ο χρόνος απάντησης ενός ερωτήματος για το σελεστιτψ εστιματιου P είναι $O(\log M + \frac{1}{\epsilon^2} \log \omega_{max})$. Ο αναμενόμενος χρόνος για κάθε ενημέρωση είναι $O(\log \omega(\log \frac{1}{\epsilon} + \log t_{max} \log \omega_{max}))$. Η πολυπλοκότητα χώρου είναι $O(\frac{1}{\epsilon^2}(\log \omega_{max} \log id_{max}))$.*

Συχνά Στοιχεία και ϵ-approximate ϕ-quantiles

Θεώρημα 2.7.12 *Μπορούμε να απαντήσουμε σε ερωτήματα για ϵ-approximate ϕ-quantiles και συχνά στοιχεία χρησιμοποιώντας το ίδιο sketch, σε χρόνο $O(\log M + \frac{1}{\epsilon^2}\log\frac{\omega_{max}}{\epsilon^2})$. Ο αναμενόμενος χρόνος για κάθε ενημέρωση είναι $O(\log\omega(\log\frac{1}{\epsilon} + \log t_{max}\log\omega_{max}))$. Η πολυπλοκότητα χώρου είναι $O(\frac{1}{\epsilon^2}(\log\omega_{max}\log id_{max}))$.*

Αξίζει να σημειωθεί ότι ένας γρηγορότερος τρόπος υπολογισμού του μέσου όρου και των quantiles είναι να υπολογίσουμε τα απαιτούμενα quantiles ενός κατάλληλα βεβαρημένου δείγματος S_l όπου το l ορίζεται ως το μικρότερο επίπεδο τέτοιο ώστε $t_l \leq c$. Το αποτέλεσμα αυτού του υπολογισμού είναι το ίδιο με αυτό που ορίσαμε νωρίτερα, αλλά υπολογίζεται πιο γρήγορα. Επομένως, μετά τον υπολογισμό του επιπέδου δείγματος και των τιμών της $RangeSample$, διατάσσουμε το δείγματα κατά τιμή και υπολογίζουμε τα συχνά στοιχεία και τα quantiles, με ένα όμως επιπλέον κόστος $O(\tau\log\tau)$ στην πολυπλοκότητα χρόνου του selectivity estimation.

2.7.4 Φθορά Κυλιόμενου Παραθύρου

Η συνάρτηση κυλιόμενου παραθύρου είναι τέλειο παράδειγμα ακέραιας συνάρτησης, και για αυτό μπορούμε να το χρησιμοποιήσουμε εύκολα στον αλγόριθμο. Διευκολύνει την εφαρμογή του αλγορίθμου, αφού ο υπολογισμός της $expiry(e_j, i)$ αρκείται στον υπολογισμό $t_j + W$. Μπορούμε όμως να αποδείξουμε ένα πιο εύρωστο αποτέλεσμα : Αν θέσουμε $f(x) = 1$ όταν προσθέτουμε ένα στοιχείο, το στοιχείο δεν θα απορριφθεί ποτέ, και έτσι απορρίπτεται το στοιχείο με την παλαιότερη χρονοσφραγίδα όταν το δείγμα είναι γεμάτο. Μπορούμε κατ' αυτόν τον τρόπο να κρατάμε μια δομή δεδομένων που είναι κατάλληλη για οποιοδήποτε μέγεθος κυλιόμενου παραθύρου $W < \infty$, όπου το W μπορεί να οριστεί μετά τη δημιουργία της δομής, και να επιστρέφει έγκυρες εκτιμήσεις των συναθροιστικών συναρτήσεων.

Θεώρημα 2.7.13 *Η δομή δεδομένων μας μπορεί να απαντήσει σε ερωτήματα κυλιόμενου παραθύρου για το άθροισμα και το selectivity estimation, όπου η παράμετρος W παρέχεται*

τη στιγμή που τίθεται το ερώτημα. Πιο συγκεκριμένα, για $\tau = O(\frac{1}{\epsilon^2})$ *και* $M = O(\log \omega_{max} + \log id_{max})$*, παρέχουμε εκτιμητή* $\hat{V}$ *του decayed sum* V *τέτοιο ώστε* $Pr[|\hat{V} - V| < \epsilon V] > \frac{2}{3}$ *και εκτιμητή* $\hat{Q}$ *του selectivity estimation* Q*, τέτοιο ώστε* $Pr[|\hat{Q} - Q| < \epsilon] > \frac{2}{3}$*. Ο χρόνος απάντησης και για τα δύο ερωτήματα είναι* $O(\log M + \tau)$*.*

Κεφάλαιο 3

Υλοποίηση και Προετοιμασία Πειραμάτων

Στο παρόν κεφάλαιο αναλύουμε τα απαραίτητα βήματα που πραγματοποιήθηκαν για την υλοποίηση των τριών αλγορίθμων που παρουσιάστηκαν στο κεφάλαιο 2 για το πρόβλημα μέτρησης διαφορετικών τιμών, *PCSA*, *Loglog Counting* και *TDS*. Τα περιβάλλοντα, τα εργαλεία και οι παράμετρες που χρησιμοποιήθηκαν για την υλοποίηση παρουσιάζονται εδώ, καθώς επίσης και οι συσκευές στις οποίες εκτελέστηκαν τα πειράματα για την αξιολόγηση των αλγορίθμων.

3.1 Ανάπτυξη και Επαλήθευση Πηγαίου Κώδικα

Η υλοποίηση των αλγορίθμων πραγματοποιήθηκε στη γλώσσα προγραμματισμού **Ansi C** και συγκεκριμένα με τη χρήση του ελεύθερου, ανοιχτού κώδικα λογισμικού *GCC* (GNU Compiler Collection), στις κατάλληλες εκδόσεις του για υποστήριξη διαφορετικών αρχιτεκτονικών συστήματος. Η επιλογή της συγκεκριμένης, χαμηλού επιπέδου, γλώσσας προγραμματισμού είναι κατά τη γνώμη μας η καλύτερη δυνατή, αφού μας δίνει τις εξής δυνατότητες :

- να χρησιμοποιήσουμε το ελεύθερο, ανοιχτού κώδικα, αξιόπιστο και πολυχρησιμοποιη-

μένο *GCC*, αφού θέλουμε να εκτελέσουμε τον κώδικά μας στις διάφορες αρχιτεκτονικές με χρήση, φυσικά, της κατάλληλης έκδοσης του τελευταίου.

- να ελέγχουμε κατά την ανάπτυξη του κώδικα, τη μνήμη που θα χρησιμοποιήσουμε με αποδοτικό τρόπο, εφόσον στοχεύουμε συσκευές περιορισμένων δυνατοτήτων.
- να παράγουμε όσο το δυνατόν μικρότερα σε μέγεθος εκτελέσιμα αρχεία (≈ 24 bytes), λόγω της μικρής υπολογιστικής ισχύς των επεξεργαστών των συσκευών στις οποίες απευθυνόμαστε.
- να έχουμε στη διάθεσή μας αξιόπιστες υλοποιήσεις γνωστών συναρτήσεων, υπό την **Γενικού Κοινού Άδεια GNU**, όπως για παράδειγμα την υλοποίηση του αλγορίθμου *Επιτομής Μηνύματος MD4*.
- να είναι πιο εύκολη η προσαρμογή σε μια υψηλότερου επιπέδου, γλώσσα προγραμματισμού στην περίπτωση που επιθυμούμε να δοκιμάσουμε κάτι τέτοιο.

Αφού ολοκληρώθηκε η ανάπτυξη του κώδικα του κάθε αλγορίθμου, έπρεπε στη συνέχεια να επιβεβαιώσουμε την ορθότητα του πηγαίου κώδικα (source code), ώστε να είμαστε σίγουροι ότι θα λάβουμε έγκυρα πειραματικά αποτελέσματα. Η επαλήθευση πραγματοποιήθηκε μέσω διάφορων εκτελέσεων ελέγχου (test-runs) για κάθε αλγόριθμο, όπου για γνωστές εκδοχές της εισόδου εξετάσαμε και συγκρίναμε τα πειραματικά αποτελέσματα των εκτελέσεων με τα αντίστοιχα αναμενόμενα θεωρητικά. Οι εκτελέσεις ελέγχου πραγματοποιήθηκαν σε **x86** πλατφόρμα.

PCSA και Loglog Counting. Πιο συγκεκριμένα, δημιουργήσαμε είσοδο η οποία αποτελούνταν από συγκεκριμένο αριθμό στοιχείων και εισάγαμε κάθε φορά ελεγχόμενο αριθμό διπλότυπων. Έπειτα, τροφοδοτώντας τους αλγορίθμους καταγράψαμε τα αποτελέσματά τους, έτσι ώστε να μπορέσουμε να ελέγξουμε αν η προσέγγιση του καθενός ήταν στα όρια της εμπειρικής πόλωσης που παρέχονται από τους συγγραφείς. Υπενθυμίζουμε ότι πόλωση είναι ο λόγος της εκτιμώμενης τιμής προς την πραγματική τιμή, δηλαδή μπορούμε να δούμε πόσο καλά προσεγγίζει ο αλγόριθμος την τιμή που ζητείται. Προφανώς, όσο πιο κοντά στη μονάδα βρίσκεται η πόλωση τόσο καλύτερο το πειραματικό αποτέλεσμα. Για τους αλγορίθμους

Πίνακας 3.1: Εκτελέσεις ελέγχου για τον PCSA, $bbits = 32$, $m = 2^8 = 256$

Μέγεθος εισόδου	**Πόλωση με 0% διπλότυπα**	**Πόλωση με 40% διπλότυπα**	**Πόλωση με 80% διπλότυπα**
1000	1.092000	1.183333	2.265000
5000	1.005800	0.980667	1.092000
10000	0.989700	1.046667	1.009500
50000	0.967460	1.031467	0.989700
100000	0.954450	0.948417	1.000500
500000	0.958560	1.019220	0.954450
1000000	1.031261	1.000087	0.964845

PCSA και Loglog Counting χρησιμοποιήσαμε αριθμό ψηφίων για τον κάθε πίνακα ψηφίων $bbits = 32$, πλήθος πινάκων $m = 2^8$ $(= 256)$ και ως συνάρτηση κατακερματισμού τον αλγόριθμο *MD4* (εξηγείται αναλυτικά η επιλογή αυτή στην επόμενη ενότητα). Σύμφωνα με τη θεωρία οι πολώσεις για τους δύο αλγορίθμους δίνονται ως :

$$\text{PCSA : πόλωση } = 1 + 0.31/\sqrt{m}$$
$$\text{Loglog : πόλωση } = 1 + \theta_{1,n}, \qquad \text{όπου } |\theta_{1,n}| < 10^{-6}$$

Έτσι, έχουμε ποσοστό τυπικού σφάλματος $\approx 1.0193\%$ για τον PCSA και ≈ 1 για τον Loglog Counting (δηλαδή, τα αποτελέσματα του τελευταίου είναι ασυμπτωτικά μη-πολωμένα). Στους πίνακες 3.1 και 3.2 φαίνονται τα αποτελέσματα των εκτελέσεων ελέγχου για διαφορετικά μεγέθη εισόδου των αλγορίθμων και διαφορετικά ποσοστά διπλότυπων.

Παρατηρούμε ότι οι αποκρίσεις των δύο αλγορίθμων είναι όπως υποδεικνύεται από τη θεωρία, δίνοντας πολύ καλά προσεγγιστικά αποτελέσματα μέχρι και για πολύ μεγάλα μεγέθη εισόδου, με και χωρίς διπλότυπα. Η επαλήθευση λοιπόν του κώδικα για τους αλγορίθμους PCSA και Loglog Counting ήταν επιτυχής και με παραπάνω από ικανοποιητικά αποτελέσματα.

Πίνακας 3.2: Εκτελέσεις ελέγχου για τον Loglog Counting, $bbits = 32$, $m = 2^8 = 256$

Μέγεθος εισόδου	**Πόλωση με 0% διπλότυπα**	**Πόλωση με 40% διπλότυπα**	**Πόλωση με 80% διπλότυπα**
1000	1.092000	1.038333	1.160000
5000	0.973600	0.957000	1.092000
10000	1.014000	0.967500	1.045500
50000	1.043540	1.085800	1.014000
100000	1.098640	1.048267	1.079200
500000	1.121450	1.058497	1.098640
1000000	1.056599	1.084608	1.029525

Time-Decaying Sketches. Για τον TDS αλγόριθμο, δεν χρησιμοποιούμε την ιδιότητα της φθοράς στο χρόνο για τα στοιχεία, αφού θέλουμε απλά να μετρήσουμε το πλήθος των διαφορετικών στοιχείων στην ροή εισόδου. Για αυτό το λόγο, θέτουμε τα βάρη των τελευταίων σε μονάδες, $\omega_i = 1$, το μέγεθος του κυλιόμενου παραθύρου, W, πολύ μεγάλο ώστε να καλύπτει και την πιο παλιά χρονοσφραγίδα της ροής εισόδου, και χωρίς να μας ενδιαφέρουν οι τιμές v_i υπολογίζουμε το *decayed sum*. Κατ' αυτόν τον τρόπο, κάθε φορά που στη ροή δεδομένων εμφανίζεται ένα καινούριο στοιχείο προστίθεται το βάρος του στο συνολικό decayed sum, ενώ όταν στην αντίστοιχη κατάσταση είναι ένα διπλότυπο, απλά απορρίπτεται από τον αλγόριθμο. Αφού έχουμε επιλέξει τα βάρη να είναι μονάδες, τελικά το decayed sum θα μας δώσει τον αριθμό των διαφορετικών στοιχείων στη ροή εισόδου, λόγω του τύπου :

$$V = \sum_{(\omega,t,id)\in D} \omega \cdot f(c-t)$$

Τίθεται βέβαια, το ζήτημα της τιμής της παραμέτρου προσέγγισης, ϵ, σύμφωνα με την οποία ορίζεται το μέγεθος των δειγμάτων, $\tau = 60/\epsilon^2$, που διατηρούνται από τον αλγόριθμο. Τονίζουμε ότι το μέγεθος των δειγμάτων είναι ζωτικής σημασίας, αφού αν, για παράδειγμα, στο δείγμα χωρούν 100 στοιχεία τότε σύμφωνα με τη διαδικασία που ακολουθεί ο αλγόριθμος, δηλαδή η παραγωγή αποτελεσμάτων από το επιλεγόμενο δείγμα μόνο, δεν θα είναι σε θέση

Πίνακας 3.3: Εκτελέσεις ελέγχου για τον TDS, $\epsilon = 0.1$

Μέγεθος εισόδου	**ΣΣ (%) με 0% διπλότυπα**	**ΣΣ (%) με 40% διπλότυπα**	**ΣΣ (%) με 80% διπλότυπα**
10	0.000000	0.000000	0.000000
100	0.000000	0.000000	0.000000
1000	0.014000	0.021667	0.015000
2000	0.017000	0.020000	0.022500
3000	0.013333	0.020000	0.015000
4000	0.014250	0.016250	0.016250
5000	0.016000	0.017000	0.021000
6000	0.014667	0.017778	0.023333

να καταμετρήσει σωστά το μέγεθος της ροής δεδομένων. Πράγμα που σημαίνει ότι, αν η πραγματική είσοδος είναι μεγέθους 1000 στοιχείων και το μέγεθος των δειγμάτων 100, τότε σαν πλήθος στοιχείων ο αλγόριθμος θεωρεί το 100 και όχι το 1000. Επομένως, πρέπει να επιλέξουμε μεταξύ μεγέθους δειγμάτων και μνήμης, μέσω της παραμέτρου ϵ. Επιλέξαμε $\epsilon = 0.1$, δηλαδή $\tau = 6000$, τιμή με την οποία ο αλγόριθμος μπορεί να καταμετρήσει με ασφάλεια διαφορετικά στοιχεία ροών δεδομένων **μεγέθους** μέχρι και **6000 στοιχείων**.

Πραγματοποιήσαμε ικανοποιητικό αριθμό επαναλήψεων με διαφορετικά μεγέθη εισόδου και διαφορετικά ποσοστά διπλότυπων και κάθε φορά παίρναμε αποτελέσματα με πολύ μικρό σχετικό σφάλμα. Στον πίνακα 3.3 φαίνονται οι τιμές του ποσοστού του σχετικού σφάλματος για τις εκτελέσεις ελέγχου διαφορετικών μεγεθών ροής εισόδου με 0%, 40% και 80% διπλότυπα. Αυτά τα αποτελέσματα μας υποδεικνύουν ότι ο αλγόριθμος έχει υλοποιηθεί ορθά και μπορούμε να προχωρήσουμε σε πειραματικές μετρήσεις.

3.2 Ειδικά Ζητήματα Υλοποίησης

Σε αυτή την υποενότητα θα δούμε κάποια ζητήματα τα οποία προέκυψαν κατά την ανάπτυξη του κώδικα σε κάθε αλγόριθμο. Εφόσον είναι η πρώτη φορά, από όσο γνωρίζουμε, που

υλοποιούνται οι συγκεκριμένοι αλγόριθμοι για τέτοιου είδους, περιορισμένα σε υπολογιστικούς πόρους, περιβάλλοντα έπρεπε να κάνουμε κάποιες επιλογές σε κομμάτια που παίζουν σημαντικό ρόλο στην απόδοση των τελευταίων.

PCSA και Loglog Counting. Το μόνο ζήτημα το οποίο οι συγγραφείς των αλγορίθμων *PCSA* και *Loglog Counting* δεν προσδιόρισαν ρητά, αλλά κατέγραψαν τις ιδιότητες που υποθέτουν και είναι απαραίτητες για τη θεωρητική απόδοση τυπικού σφάλματος των αλγορίθμων, είναι η *επιλογή της συνάρτησης κατακερματισμού*, η οποία πρέπει να ανήκει σε συγκεκριμένη οικογένεια συναρτήσεων κατακερματισμού. Η συγκεκριμένη συνάρτηση κατακερματισμού πρέπει να υποστηρίζει την αντιστοίχιση κάθε τιμής από το σύνολο των στοιχείων, ομοιόμορφα τυχαία σε έναν ακέραιο στο καθορισμένο διάστημα.

Η επιλογή της συνάρτησης κατακερματισμού, αν και σημαντική, αποδείχθηκε εύκολη υπόθεση καθώς υπάρχουν πολλοί και εύρωστοι αλγόριθμοι επιτομής μηνυμάτων και συναρτήσεις κατακερματισμού, όπως ο αλγόριθμος *MD4* και η συνάρτηση *SHA-1*, τις οποίες και δοκιμάσαμε επιτυχώς με χρήση του ανοιχτού κώδικα εργαλείου *OpenSSL*, παράγοντας αποτελέσματα εντός των θεωρητικών ορίων για όλες τις εκδοχές εισόδου και διπλότυπων. Λόγω όμως, ευκολότερης υλοποίησης του αλγορίθμου MD4 εκτός OpenSSL, διότι δεν ήταν δυνατή η εγκατάσταση του τελευταίου σε κάποιες από τις συσκευές λόγω περιορισμένης μνήμης, τον επιλέξαμε για χρήση στην πειραματική διαδικασία και εκτίμηση των δύο αυτών αλγορίθμων.

Η υλοποίηση του MD4 που ενσωματώσαμε στον κώδικά μας είναι κατοχυρωμένη από την *RSA Data Security Inc.*, η οποία παρέχει ελεύθερα τον συγκεκριμένο κώδικα. Μετά από εξέταση και επαλήθευση του κώδικα για το συγκεκριμένο αλγόριθμο, τον ενσωματώσαμε στον δικό μας κώδικα για χρησιμοποίηση στους δύο πιθανοτικούς αλγορίθμους.

Time-Decaying Sketches. Τα θέματα που μας απασχόλησαν κατά την υλοποίηση του αλγορίθμου αυτού ήταν πολύ σημαντικά και επηρέασαν τη θεωρητική πολυπλοκότητα χρόνου ενημέρωσης του sketch. Το πρώτο ζήτημα αναφέρεται στον προσδιορισμό της συνάρτησης κατακερματισμού που χρησιμοποιείται για τη διατήρηση των δειγμάτων του αλγορίθμου. Το δεύτερο είναι η υλοποίηση της συνάρτησης $expiry()$, η οποία περιλαμβάνει τη συνάρτηση

$RangeSample()$. Τρίτο θέμα είναι η εύρεση του μέγιστου βάρους, ω_{max} και της μέγιστης ταυτότητας, id_{max}, που είναι χρήσιμα για την αρχικοποίηση του αλγορίθμου και συγκεκριμένα τον προσδιορισμό του πλήθους των δειγμάτων, $M = \lceil \log \omega_{max} + \log id_{max} \rceil$.

Εύρεση Πρώτου Αριθμού. Το πρώτο ζήτημα το οποίο αντιμετωπίσαμε είναι αυτό της εύρεσης του πρώτου αριθμού μέσα σε προσδιορισμένο τυχαίο διάστημα ακεραίων, που είναι απαραίτητος για τον προσδιορισμό της συνάρτησης κατακερματισμού που χρησιμοποιείται στη διατήρηση των δειγμάτων $S_0, S_1, \ldots, S_M$, αφού ουσιαστικά αποφασίζει για το αν θα εισχωρήσει το συγκεκριμένο στοιχείο στο επίπεδο για το οποίο εξετάζεται.

Έπειτα από αρκετά εκτενή έρευνα καταλήξαμε στην χρησιμοποίηση του παρακάτω απλού, αλλά ορθού και αξιόπιστου κώδικα :

```
for(i=d_bound+1; i<u_bound; i++)
{
    c=0;
    for(j=1; j<=i; j++)
    {
        if (i%j==0)
        {
           c=c+1;
        }
    }

    if (c==2)
    {
        hash_array[0] = i;
        break;
    }
}
```

Με *d_bound* και *u_bound* συμβολίζουμε το κάτω και πάνω όριο του διαστήματος ακεραίων στο οποίο ανήκει ο πρώτος αριθμός που επιλέγουμε. Ο κώδικας στηρίζεται στο ότι όλοι οι πρώτοι αριθμοί > 3 μπορούν να εκφραστούν μέσω των δύο τύπων $p = 6n - 1$ ή $p = 6n + 1$, όπου n ακέραιος. Υπάρχει η περίπτωση στο διάστημα που ορίζεται από τους συγγραφείς να ανήκουν παραπάνω από ένας πρώτοι αριθμοί, για αυτό εμείς επιλέγουμε τον πρώτο που συναντάται χωρίς βλάβη της γενικότητας.

Συνάρτηση $expiry(e, i)$**.** Η συνάρτηση $expiry(e, i)$ προσδιορίζει τον χρόνο λήξης του στοιχείου e για το επίπεδο i, ο οποίος μπορεί να είναι είτε μηδενικός, οπότε και δεν ανήκει ποτέ στο συγκεκριμένο επίπεδο, είτε άπειρος, που σημαίνει ότι ανήκει στο επίπεδο i χωρίς να έχει χρόνο λήξης. Για το σκοπό αυτό χρησιμοποιείται από την $expiry(e, i)$ η συνάρτηση $RangeSample()$. Φτάνοντας στην υλοποίηση της τελευταίας και πιο συγκεκριμένα της κύριας διαδικασίας $Hits$, η οποία μετρά αποδοτικά τον αριθμό των ακεραίων που ανήκουν σε ένα συγκεκριμένο διάστημα, συναντήσαμε σημαντικό πρόβλημα. Η διαδικασία $Hits$ είναι αναδρομική και σε κάθε κλήση της ανάγει το κυρίως πρόβλημα σε ένα σημαντικά μικρότερο, μέσω μείωσης του μεγέθους του διαστήματος ακεραίων το οποίο και εξετάζει. Για να το κάνει αυτό χρησιμοποιεί *αριθμητικές προόδους* και τις ιδιότητές τους, σε συνδυασμό με πράξεις *αριθμητικής υπολοίπων* (modulo arithmetic).

Εδώ ακριβώς εμφανίζεται το πρόβλημα, ενώ οι πράξεις σε πρώτη φάση αναφέρονται σε θεωρητική αριθμητική υπολοίπων, σύμφωνα με την οποία μπορεί να παραχθεί αρνητικό αποτέλεσμα από μια πράξη modulo, σε επόμενη φάση οι πράξεις γίνονται μέσω της υλοποιημένης για υπολογιστές πράξης modulo[1], η οποία παράγει μόνο θετικά αποτελεσματα. Προσπαθώντας να παράγουμε σωστά αποτελέσματα προχωρήσαμε στον υπολογισμό της σωστής προσέγγισης του αλγορίθμου αλλά δεν βρήκαμε κάποιο θεωρητικό μονοπάτι που να οδηγεί έστω και με συνδυασμό των δύο πράξεων σε ορθές αποκρίσεις.

Επομένως, έπρεπε να αντικαταστήσουμε τη διαδικασία $Hits$, όπως αυτή ορίζεται από τους συγγραφείς, και να προχωρήσουμε στην υλοποίηση ανάλογης διαδικασίας. Το πιο απλό παράδειγμα τέτοιας διαδικασίας είναι η ένα - προς - ένα καταμέτρηση του πλήθους των ακε-

[1] η πράξη modulo στους υπολογιστές αναφέρεται στον υπολογισμό του υπολοίπου μιας διαίρεσης

ραίων ενός συνόλου που ανήκουν σε ένα συγκεκριμένο διάστημα. Ονομάζουμε τη διαδικασία αυτή $DummyHits$, καθώς αποτελεί την χειρότερη περίπτωση χρονικής πολυπλοκότητας ενός ανάλογου αλγορίθμου, έχοντας πολυπλοκότητα *ανάλογη του μεγέθους του διαστήματος εισόδου*. Η πολυπλοκότητα χρόνου της συνάρτησης $RangeSample$ από $O(\log |r_e^c|)$ γίνεται μέσω της $DummyHits$ $O(|r_e^c|)$, όπου $|r_e^c|$ είναι το μέγεθος του διαστήματος των ακεραίων που ορίζεται για το στοιχείο e τη χρονική στιγμή c. Εφόσον η συνάρτηση $expiry$ βασίζεται στην $RangeSample$, αλλάζει και η πολυπλοκότητα της πρώτης, από $O(\log t_{max}\omega_{max})$ σε $O(|r_e^c| + \log t_{max})$. Αυτό φυσικά επηρεάζει άμεσα την χρονική πολυπλοκότητα του *TDS* αλγορίθμου, αφού για την ενημέρωση του sketch αν το στοιχείο που εξετάζεται δεν είναι κάποιο διπλότυπο, ο χρόνος που θα πάρει ο έλεγχος για τα διάφορα επίπεδα δειγμάτων στα οποία μπορεί να ανήκει, είναι λογαριθμικός στο μέγεθος του διαστήματος ακεραίων που έχει οριστεί για το στοιχείο αυτό, δηλαδή από $O(\log \omega(\log \tau + \log t_{max} \log \omega_{max}))$ γίνεται $O(\log \omega(\log \tau + |r_e^c| + \log t_{max}))$.

Προσδιορισμός των ω_{max} και id_{max}. Το δεύτερο θέμα που μας απασχόλησε κατά την υλοποίηση του αλγορίθμου TDS, ήταν ο προσδιορισμός της μέγιστης τιμής των βαρών και της μέγιστης τιμής των ταυτοτήτων των στοιχείων της ροής δεδομένων. Θυμηθείτε ότι από τις δύο αυτές τιμές εξαρτάται το μέγεθος των δειγμάτων, M, που διατηρεί ο αλγόριθμος. Οι συγγραφείς δεν αναφέρουν κάπου πως θα υπολογίσουμε τις τιμές αυτές χωρίς να χρειαστεί να σαρώσουμε τη ροή δεδομένων μία φορά παραπάνω, σε περίπτωση που ο χρήστης δεν γνωρίζει εκ των προτέρων τις τιμές αυτές. Επομένως, ο χρήστης είτε θα γνωρίζει τις τιμές, είτε θα έχει κάποιες εκτιμήσεις αυτών, ώστε να μπορέσει ο αλγόριθμος να παράγει αξιόπιστα και ορθά αποτελέσματα. Δηλαδή, κάποιος που επιθυμεί να χρησιμοποιήσει το συγκεκριμένο αλγόριθμο για κάποιο σκοπό, πρέπει από πριν να γνωρίζει το μέγεθος της ροής δεδομένων και τα βάρη που θέλει να χρησιμοποιήσει, έτσι ώστε να μην εκτιναχθεί η πολυπλοκότητα χώρου στα ύψη ή να μην "ξεμείνει" από χώρο στην περίπτωση μικρότερης από την απαιτούμενη εκτίμηση πολυπλοκότητας χώρου.

Στην περίπτωσή μας, όπου το μόνο που κάνουμε είναι η μέτρηση διαφορετικών στοιχείων, η μέγιστη τιμή βάρους είναι $\omega_{max} = 1$, όπως επεξηγείθηκε προηγουμένως, ενώ η μέγιστη

ταυτότητα δίνεται από το μέγεθος της ροής εισόδου με την οποία τροφοδοτούμε τον αλγόριθμο, $id_{max} = n$. Επομένως, οι πολυπλοκότητες χώρου και χρόνου δεν επηρεάζονται καθόλου, όπως άλλωστε υποθέτουν και οι συγγραφείς όταν δεν συνυπολογίζουν τη διαδικασία αυτή στις πολυπλοκότητες χώρου και χρόνου.

3.3 Πειραματική Διαδικασία

Αφού είδαμε τα σημαντικότερα ζητήματα που μας απασχόλησαν κατά την ανάπτυξη του κώδικα, σε αυτή την ενότητα θα περιγράψουμε τα θέματα της πειραματικής διαδικασίας και τις παραμέτρους των αλγορίθμων που χρησιμοποιήθηκαν για την τελευταία. Συγκεκριμένα, θα αναφέρουμε τις συσκευές στις οποίες έγιναν τα πειράματα και τα περιβάλλοντα που αυτές υποστηρίζουν, τη μορφή της εισόδου την οποία εισάγαμε για τις διάφορες μετρήσεις και τις παραμέτρους των αλγορίθμων που χρησιμοποιήθηκαν.

3.3.1 Συσκευές και Περιβάλλοντα Εκτέλεσης

Οι συσκευές στις οποίες πραγματοποιήθηκαν οι πειραματικές μετρήσεις χρονικής πολυπλοκότητας των αλγορίθμων είναι συσκευές περιορισμένων υπολογιστικών πόρων και μας δίνουν τη δυνατότητα για παραγωγικά συμπεράσματα όσον αφορά την χρησιμοποίηση των αλγορίθμων σε εφαρμογές του πραγματικού κόσμου. Αναφορικά αυτές είναι :

- **OpenMoko Neo Freerunner** (PDA-GPS-smartphone)
- **Crossbow NB100-Stargate Netbridge** (πύλη ενσωματωμένη σε δίκτυα αισθητήρων)
- **PC Engine Alix** (ενσωματωμένη υπολογιστική πλατφόρμα)
- **iSense Core Module** (συσκευή αισθητήρα)

Στην συνέχεια περιγράφουμε, εν συντομία, την κάθε συσκευή και τις τεχνικές της προδιαγραφές ώστε να έχει ο αναγνώστης μια καλή εικόνα του υλικού που χρησιμοποιήθηκε και τις εφαρμογές στις οποίες θα μπορούσε να ενσωματωθεί.

OpenMoko Neo Freerunner

Το *Neo Freerunner*, σχήμα 3.1, είναι το δεύτερο τηλέφωνο σχεδιασμένο για το λογισμικό *OpenMoko* και είναι ο απόγονος του *Neo 1973*. Με το όνομα OpenMoko αναφερόμαστε σε δύο πράγματα, τη Linux διανομή που είναι σχεδιασμένη για ανοιχτές κινητές υπολογιστικές πλατφόρμες (open mobile computing platforms), όπως τα κινητά τηλέφωνα και την εταιρία πίσω από τη συγκεκριμένη διανομή, η οποία είναι και η κατασκευάστρια εταιρία κινητών υπολογιστικών μονάδων, όπως τα Neo Freerunner και 1973.

Σχήμα 3.1: OpenMoko Neo Freerunner

Το Freerunner είναι εφυές κινητό τηλέφωνο (smartphone) με οθόνη επαφής (touch screen) και λειτουργεί ταυτόχρονα σαν GSM/GPRS τηλέφωνο, σαν PDA και σαν GPS (Global Positioning System) συσκευή (για την GPS λειτουργία χρησιμοποιείται *OpenStreetMap* λογισμικό). Τα τεχνικά χαρακτηριστικά της συσκευής δίνονται αμέσως παρακάτω :

- Επεξεργαστής **Samsung 2442B** στα **400MHz** (ARM920T πυρήνας, ARMv4T)

- **128MB SDRAM συνολική μνήμη**, 64MB στο εσωτερικό της CPU, 64MB εξωτερικά
- 256MB integrated flash μνήμη (επεκτάσιμη μέσω microSD ή microSDHC κάρτα)
- Οθόνη επαφής υψηλής ανάλυσης 2.84' (43mm x 58mm) 480x640 pixels
- microSD υποδοχή υποστηρίζοντας μέχρι και 8GB SDHC(Secure Digita High Capasity) κάρτες
- Εσωτερικό δέκτη GPS
- Υποστήριξη Bluetooth
- Υποστήριξη ασύρματης σύνδεσης μέσω 802.11 b/g Wi-Fi
- 2 3D επιταχυνσιόμετρα
- Tri-band GSM/GPRS
- Λειτουργία usb host με 500mA ισχύ, επιτρέποντας την τροφοδότηση usb συσκεύων για μικρά χρονικά διαστήματα

Το λογισμικό που, προς το παρόν, διατίθεται μαζί με τη συσκευή προορίζεται για προχωρημένους χρήστες και όχι για το γενικό κοινό. Είναι βασισμένο στο λειτουργικό Linux και συνίσταται κυρίως στους χρήστες του συγκεκριμένου λειτουργικού συστήματος. Οι τεχνικές προδιαγραφές του είναι πολύ υψηλές για συσκευή του είδους του και η ελευθερία χρήσης που δίνεται μέσω του λογισμικού είναι τεράστια. Η διανομή που χρησιμοποίήθηκε για τα πειράματα είναι η *FDOM* (Fat and Dirty OpenMoko), η οποία περιέχει πληθώρα εφαρμογών και μας παρείχε την λειτουργία του *τερματικού* (terminal) που ήταν απαραίτητη για τις οποιεσδήποτε ρυθμίσεις και εκτελέσεις προγραμμάτων.

Crossbow NB100-Stargate Netbridge

Το *Stargate Netbridge*, σχήμα 3.2, είναι μια ενσωματωμένη πύλη για δίκτυα αισθητήρων. Βασική της λειτουργία είναι να συνδέει τους Crossbow αισθητήρες σε ένα Ethernet δίκτυο.

Υποστηρίζει μία θύρα ethernet και δύο USB 2.0. Η συσκευή διατίθεται με λειτουργικό την *Debian* διανομή Linux, η οποία είναι η καλύτερη επιλογή για την αρχιτεκτονική **ARM**. Τα τεχνικά χαρακτηριστικά της συσκευής είναι :

Σχήμα 3.2: Crossbow Stargate Netbridge Gateway

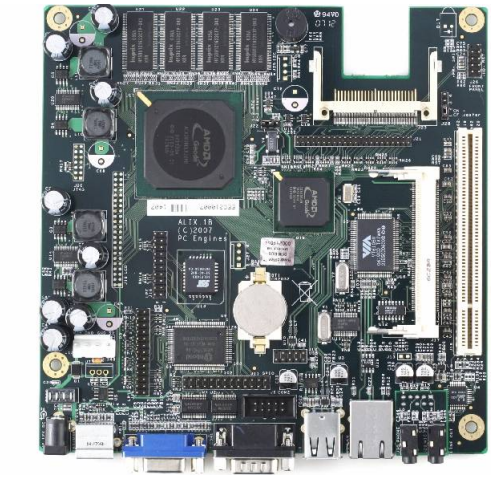
- Επεξεργαστής *Intel IXP420 XScale* στα **266MHz**
- **8MB Program FLASH, 32MB RAM** και ένα USB 2.0 System Disk χωρητικότητας 2GB
- Ενσωματωμένος Web Εξυπηρέτης (MoteExplorer)
- Ενσωματωμένο εργαλείο διαχείρισης δικτύων αισθητήρων (XServe)
- Διασύνδεση Web για πραγματικού χρόνου παρακολούθηση δεδομένων των αισθητήρων και κατάστασης του δικτύου
- Διασύνδεση Ethernet Backhaul για απομακρυσμένη πρόσβαση και παρακολούθηση

Η όλη διαδικασία στη συγκεκριμένη συσκευή έγινε μέσω SSH (Secure Shell). Φορτώσαμε στο USB System Disk τις ρυθμίσεις δικτύου και συνδέσαμε τη συσκευή σε τοπικό δίκτυο, από όπου και πραγματοποιήθηκαν μεταφορές αρχείων και εκτελέσεις προγραμμάτων μέσω SSH.

PC Engine Alix

Το *Alix* της *PC Engine*, σχήμα 3.3, είναι μια μικρής κλίμακα υπολογιστική μονάδα που χρησιμοποιείται σε ασύρματους δρομολογητές (wireless routers), τοίχη προστασίας (firewalls),

βιομηχανικές διασυνδέσεις χρήστη, ειδικού σκοπού δικτυακές συσκευές κ.α. Το συγκεκριμένο μοντέλο της σειράς Alix που χρησιμοποιήσαμε είναι το **alix1d**. Τα προϊόντα της σειράς διατίθενται με ανοιχτού κώδικα λειτουργικά συστήματα και BIOS, σε συνδυασμό με τα χαμηλού κόστους ολοκληρωμένα και περιφερειακά. Το λειτουργικό σύστημα στο οποίο πραγματοποιήθηκαν όλες οι μετρήσεις είναι η διανομή Linux *Xubuntu 8.04*, η οποία είναι ανεπτυγμένη για φορητούς υπολογιστές, εξυπηρέτες αλλά και επιτραπέζιους υπολογιστές που αποσκοπούν στην περιορισμένη κατανάλωση ενέργειας. Τα τεχνικά χαρακτηριστικά της συγκεκριμένης συσκευής φαίνονται αμέσως παρακάτω :



Σχήμα 3.3: PC Engine Alix, alix1d

- Επεξεργαστής **AMD Geode LX800** στα **500MHz**
- Μνήμη **256MB SDRAM**
- 1 θύρα Ethernet (μέσω VT6105M 10/100)
- 2 θύρες COM
- 4 θύρες USB 2.0

iSense Core Module

Το *iSense Core Module*, σχήμα 3.4, αποτελεί τη βασική μονάδα υλικού της iSense πλατφόρμας, που χρησιμοποιείται για διάφορες εφαρμογές ασύρματων δικτύων αισθητήρων. Μερικές από τις τελευταίες είναι η αυτοματοποίηση εργαστασίου, η επιτήρηση και ασφάλεια κτιρίων, η περιβαλλοντική και δομική επιτήρηση, καθώς και εφαρμογές που απαιτούν δίκτυα αισθητήρων μεγάλης κλίμακας. Αυτό που το ξεχωρίζει από τις υπόλοιπες εμπορικές συσκεύες αισθητήρων είναι ο συνδυασμός της υψηλής αποδοτικότητας και χαμηλής κατανάλωσης ενέργειας. Επιπλεόν, το iSense Core Module παρέχει μεγάλο πλήθος περιφερειακών διασυνδέσεων, όπως για παράδειγμα υποστήριξη I2C (Inter-Integraded Circuit), SPI (Serial Peripheral Interface), ένα τετρακάναλο 11-ψήφιο μετατροπέα αναλογικού σε ψηφιακό (Analog-to-Digital Converter), δύο 10-ψήφιους μετατροπείς ψηφιακού σε αναλογικό (Digital-to-Analog Converter), δύο *UARTs* (Univershal Asynchronous Receiver/Transmitter) κ.α.

Σχήμα 3.4: iSense Core Module

Τα κυριότερα τεχνικά χαρακτηριστικά της αρχιτεκτονικής πίσω από τη συγκεκριμένη συσκευή είναι :

- **RISC Controller** 32-ψηφίων στα **16MHz**
- **96Kb** διαμοιραζόμενη μνήμη προγραμμάτων και δεδομένων
- IEEE 802.15.4 συμβατός πομποδέκτης ραδιοκυμμάτων, 250 kbit/s, AES κρυπτογράφιση υλικού
- Υψηλής ακρίβειας (20ππμ) πραγματικού χρόνου ρολόι

Πίνακας 3.4: Βασικά Τεχνικά Χαρακτηριστικά Συσκευών

	Επεξεργαστής	**Αρχιτεκτονική**	**RAM**	**Λειτουργικό**
Stargate NB	266MHz Intel XScale	IXP420,ARMv5TE	32MB	Debian 2.6.18
OM Freerunner	400MHz Samsung 2442B	ARM920T,ARMv4T	128MB	GNU/Linux FDOM
Alix	500MHz AMD Geode LX800	x86	256MB	Xubuntu 8.10 / 2.6.27
iSense Core Module	16MHz, RISC Controller	-	96KB	iSense Firmware

- Ελεγχόμενος μέσω λογισμικού ρυσμιστής τάσης
- SMA (SubMiniature version A) διασυνδέτη ή κεραμική κεραία ολοκληρωμένων κυκλωμάτων
- Επεκτάσεις διασυνδετών για όλα τα είδη μονάδων και πηγών ενέργειας

Στον πίνακα 3.4 φαίνονται συγκεντρωμένα τα βασικά χαρακτηριστικά των συσκευών που χρησιμοποιήθηκαν. Μπορούμε εύκολα να δούμε ότι οι συσκευές που έχουν επιλεγεί είναι οι κατάλληλες για τα πειράματα που θέλουμε να πραγματοποιήσουμε, καθώς είναι συσκευές περιορισμένων πόρων αλλά έχουν και διαφορά στην αρχιτεκτονική. Έχουμε στη διάθεσή μας δύο διαφορετικές γενιές αρχιτεκτονικής ARM, μία x86 και μια πιο εξειδικευμένη αρχιτεκτονική σχεδιασμένη αποκλειστικά για δίκτυα αισθητήρων, γεγονός που μας δίνει τη δυνατότητα να εξετάσουμε τις πειραματικές αποκρίσεις των αλγορίθμων στις διαφορετικές πραγματικού-κόσμου πλατφόρμες και αρχιτεκτονικές.

Cross-Compiling Αφού ολοκλιρώσαμε την περιγραφή των συσκευών ας αναφερθούμε σύντομα στη διαδικασία του *cross-compiling*, η οποία κρίθηκε αναγκαία και διευκόλυνε κατά πολύ την όλη διαδικασία των πειραμάτων. Μετά την ανάπτυξη και επαλήθευση του πηγαίου κώδικα σε x86 πλατφόρμα έπρεπε να δημιουργήσουμε εκτελέσιμα προορισμένα για αρχιτε-

κτονική **ARM** (Advanced Risk Architecture), η οποία υποστηρίζεται στις δύο από τις τρείς συσκευές που έχουμε επιλέξει (*Stargate Netbridge* και *OpenMoko Neo Freerunner*).

Γενικά, η διαδικασία του *cross-compiling* αφορά την παραγωγή εκτελέσιμων προορισμένων για διαφορετική αρχιτεκτονική από την αρχιτεκτονική στην οποία λαμβάνει χώρα η μεταγλώττιση. Κατά αυτόν τον τρόπο, η μεταγλώττιση γίνεται πολύ πιο γρήγορα και δεν χρειάζεται η μεταφορά του πηγαίου κώδικα στην αντίστοιχη συσκευή αλλά μόνο το τελικό εκτελέσιμο. Το cross-compiling μπορεί να πραγματοποιηθεί μέσω εξομοίωσης πλατφόρμας με τη χρήση διάφορων εργαλείων, όπως για παράδειγμα τον εξομοιωτή *qemu*. Εξομοιώνοντας την κατάλληλη αρχιτεκτονική μπορεί κανείς να μεταγλωττίσει πηγαίο κώδικα για τη συγκεκριμένη αρχιτεκτονική, έπειτα να μεταφέρει το εκτελέσιμο στη συσκευή που επιθυμεί να το τρέξει.

Ένας άλλος τρόπος για την παραγωγή εκτελέσιμων διαφορετικής αρχιτεκτονικής από τον υπολογιστή που πραγματοποιεί τη μεταγλώττιση, είναι να εγκατασταθεί ο κατάλληλος μεταγλωττιστής για τη συγκεκριμένη αρχιτεκτονική. Αυτό δεν είναι πάντα εφικτό, εξαρτάται από την αρχιτεκτονική του επεξεργαστή, η οποία δεν είναι πάντα συμβατή με κάποιον από τους μεταγλωττιστές που υπάρχουν. Στην περίπτωσή μας ακολουθήσαμε αυτό τον τρόπο μεταγλώττισης, εγκαθιστώντας τους μεταγλωττιστές που προτείνουν οι κατασκευαστές των συσκευών.

3.3.2 Περιγραφή της Εισόδου

Για την εκτέλεση των πειραμάτων χρειαζόμαστε είσοδο που να προσομοιάζει όσο το δυνατόν πιστότερα δεδομένα παραγώμενα από κινητά δίκτυα ή ακόμα καλύτερα δίκτυα αισθητήρων. Τα δεδομένα τέτοιων δικτύων μπορούν να βρίσκονται σε οποιαδήποτε κατανομή, χωρίς αυτό να επηρεάζει τους αλγορίθμους που εξετάζουμε. Επομένως, επιλέξαμε *τυχαία ομοιόμορφα κατανεμημένα* δεδομένα με εισαγωγή σε αυτά συγκεκριμένου ποσοστού διπλότυπων. Η ίδια είσοδος τροφοδοτούνταν και στους τρεις αλγορίθμους κάθε φορά, έτσι ώστε να έχουμε στη διάθεσή μας μια δίκαια σύγκριση μεταξύ των αλγορίθμων.

Η μορφή της εισόδου ήταν στη μορφή που προτείνεται από τον αλγόριθμο TDS, όπου

ένα στοιχείο ορίζεται ως $e = (\omega, t, id)$ (χωρίς το v αφού δεν μας χρειάζεται). Το μόνο που είχαμε να κάνουμε ήταν να παράγουμε δύο τυχαία ομοιόμορφα κατανεμημένες ακολουθίες ακεραίων, η πρώτη αντιστοιχίζεται με τις ταυτότητες και η δεύτερη με τις χρονοσφραγίδες των στοιχείων. Έπειτα, επιλέγονται τυχαία κάποια από τα ήδη υπάρχοντα στοιχεία και ανάλογα με το ποσοστό διπλότυπων που επιθυμούμε να εισάγουμε αποθηκεύονται και κατανέμονται στη συνολική δομή των στοιχείων. Εφόσον βέβαια οι αλγόριθμοι είναι ανεκτικοί σε διπλότυπα, η κατανομή των διπλότυπων δεν τους επηρεάζει σε κάποια από τις πολυπλοκότητές τους.

Σε αυτό το σημείο πρέπει να τονισθεί ότι όσον αφορά τον αλγόριθμο TDS, το μέγιστο μέγεθος της εισόδου με την οποία μπορούμε να τον τροφοδοτήσουμε εξαρτάται από την επιλογή της προσεγγιστικής παραμέτρου, ϵ, όπως έχει αναλυθεί και σε προηγούμενη ενότητα. Η επιλογή μας όπως και για τις εκτελέσεις ελέγχου, είναι $\epsilon = 0.1$, έτσι ώστε να μην είναι πολύ μεγάλο το μέγεθος των δειγμάτων, τ, γιατί έχουμε να κάνουμε με συσκευές περιορισμένης χωρητικότητας μνήμης. Επομένως, σύμφωνα με την επιλογή μας αυτή το μέγιστο μέγεθος της εισόδου, που μπορούμε με ασφάλεια να υπολογίσουμε είναι $\tau = 60/\epsilon^2 = 6000$ στοιχεία.

Για τους δύο πιθανοτικούς αλγορίθμους, PCSA και Loglog Counting, δεν έχουμε περιορισμό στο μέγεθος της εισόδου. Για αυτό το λόγο, φτάνουμε σε μέγεθος του 10^6 έτσι ώστε να έχουμε μια πολύ καλή εικόνα της πραγματικής χρήσης των αλγορίθμων. Αν εξετάσουμε πραγματικά δεδομένα τέτοιων δικτύων, θα δούμε ότι φτάνουν τα $700000 - 800000$ στοιχεία, επομένως σταματώντας στα 10^6 στοιχεία είναι μια άριστη επιλογή μεγέθους για την πειραματική αξιολόγηση.

3.3.3 Επιλογή Παραμέτρων των Αλγορίθμων

Σε αυτή την υποενότητα θα αναφέρουμε τις επιλογές των παραμέτρων κάθε αλγορίθμου που χρησιμοποιήθηκαν για την πραγματοποίηση των πειραμάτων.

PCSA και Loglog Counting Όπως και στις εκτελέσεις ελέγχου, έτσι και εδώ οι παράμετρες που χρησιμοποιήσαμε όσον αφορά το πλήθος των ψηφίων του κάθε πίνακα ψηφίων, $bbits = 32$, και το πλήθος των πινάκων είναι $m = 2^8$ $(= 256)$. Είναι, ίσως η καταλληλότερη επιλογή

όταν πρόκειται να εκτελεστούν τα πειράματα σε περιορισμένα περιβάλλοντα όπως σε αυτά που έχουμε επιλέξει. Για παράδειγμα, η επιλογή $bbits = 64$ να μην ήταν και τόσο φρόνιμη, αφού τέτοιες συσκευές δεν έχουν χώρο για υποστήριξη επιπλέον λογισμικού υλοποίησης αριθμητικών πράξεων με λέξεις 64 ψηφίων. Το πλήθος των πινάκων, m, είναι και αυτό πολύ καλή επιλογή αφού η μνήμη είναι μέσα στα εφικτά όρια των συσκευών, αυτό άλλωστε είναι και το βασικό πλεονέκτημα των πιθανοτικών αλγορίθμων.

Time-Decaying Sketches Για τον αλγόριθμο TDS, η σημαντικότερη παράμετρος για την οποία έπρεπε να αποφασίσουμε είναι η προσεγγιστική παράμετρος, ϵ. Από αυτή εξαρτάται το μέγεθος του κάθε δείγματος που διατηρεί ο αλγόριθμος, αφού $\tau = 60/\epsilon^2$, και επομένως η πολυπλοκότητα μνήμης, αλλά και το μέγιστο μέγεθος ροής στην οποία μπορεί να αποκριθεί με ασφάλεια ο αλγόριθμος. Για να έχουμε τη δυνατότητα να εξετάσουμε εκτενέστερα, πειραματικά τη συμπεριφορά του αλγορίθμου χωρίς να επιβαρύνουμε κατά πολύ τη χωρητικότητα των συσκευών, επιλέξαμε $\epsilon = 0.1$, ώστε $\tau = 6000$ στοιχεία. Επομένως, μπορούμε να τροφοδοτήσουμε τον αλγόριθμο με ροές δεδομένων μεγέθους μέχρι και 6000 στοιχείων, λαμβάνοντας ορθές και αξιόπιστες αποκρίσεις.

Κεφάλαιο 4

Πειραματική Αξιολόγηση Αλγορίθμων

Το πρώτο μέρος του κεφαλαίου αυτού ασχολείται με την παρουσίαση του συνόλου των πειραματικών αποτελεσμάτων ανά πλατφόρμα, μέσω γραφικών παραστάσεων. Στο δεύτερο μέρος, γίνεται αξιολόγηση των αποτελεσμάτων αυτών, σημειώνοντας χρήσιμες παρατηρήσεις και σχόλια όπου χρειάζεται.

4.1 Πλήρης Παρουσίαση Γραφικών Παραστάσεων

Στην υποενότητα αυτή θα παρουσιάσουμε όλα τα πειραματικά αποτελέσματα των αλγορίθμων ανά αρχιτεκτονική. Σε πρώτη φάση, δίνουμε τις γραφικές παραστάσεις μέτρησης χρόνου στα διάφορα μεγέθη εισόδου, με και χωρίς εισαγωγή διπλότυπων. Σε δεύτερη φάση, παρουσιάζουμε τις γραφικές παραστάσεις της χρονικής απόδοσης συναρτήσει του ποσοστού διπλότυπων της ροής εισόδου. Τέλος, δίνονται οι γραφικές παραστάσεις του ποσοστού του σχετικού σφάλματος σε σχέση με το μέγεθος της εισόδου, με και χωρίς διπλότυπα.

4.1.1 iSense Core Module

Στη συγκεκριμένη συσκευή η υλοποίηση δεν ολοκληρώθηκε εξαιτίας διαφόρων θεμάτων της προσαρμογής του κώδικα στο firmware της συσκευής. Το κυριότερο πρόβλημα που αντι-

μετωπίσαμε ήταν η μη-υποστήριξη αριθμών κινητής υποδιαστολής από το βασικό firmware, τους οποίους χρειαζόμαστε και στους τρεις αλγορίθμους. Επόμενο θέμα ήταν το μέγεθος του κώδικα, το οποίο τελικά ήταν υπέρογκο για το συγκεκριμένο υλικό. Τέλος, η μη-υποστήριξη συγκεκριμένων αριθμητικών πράξεων που είναι απαραίτητοι για την υλοποίηση των αλγορίθμων (πχ. η ύψωση πραγματικού ή ακέραιου αριθμού σε δύναμη), ήταν ακόμη ένα αδιέξοδο το οποίο χρίζει επίλυσης.

Όσον αφορά το πρώτο θέμα, το ζήτημα των αριθμών κινητής υποδιαστολής, το ήδη υπάρχον λογισμικό για προσομοίωση των τελευταίων στο iSense firmware υπάρχει σαν επιπρόσθετο πακέτο λογισμικού, αλλά δεν έχει χρησιμοποιηθεί και ελεγχθεί ακόμα, ώστε να είναι πιστοποιημένη η ορθότητα της υλοποίησής του. Επιλέγοντας το πακέτο αυτό για ενσωματωση στο firmware αριθμών κινητής υποδιαστολής, αντιμετωπίσαμε προβλήματα κατά τη σύνδεση (linkint) του κώδικα με το δυαδικών δεδομένων αρχείο (binary data file) που φορτώνεται στη συσκευή. Η επίλυση του προβλήματος έγινε δυνατή μετά από τη μεσολάβηση του υπεύθυνου του λογισμικού της συσκεύης, ο οποίος ανέλαβε και επιδιόρθωσε τον κώδικα του συγκεκριμένου πακέτου στα διάφορα σημεία που ήταν απαραίτητο.

Ακόμα και τότε όμως, δεν ήταν δυνατή η ορθή υλοποίηση των αλγορίθμων λόγω πολύ περιορισμένης μνήμης της συσκευής (96KB συνολική μνήμη). Το μέγεθος του κώδικα με ενσωμάτωση του λογισμικού αριθμών κινητής υποδιαστολής αυξάνεται κατά πολύ, με αποτέλεσμα το συνολικό μέγεθος της υλοποίησης να είναι οριακά εντός των προδιαγραφών που υποδεικνύονται από τους υπεύθυνους του iSense υλικού (90KB).

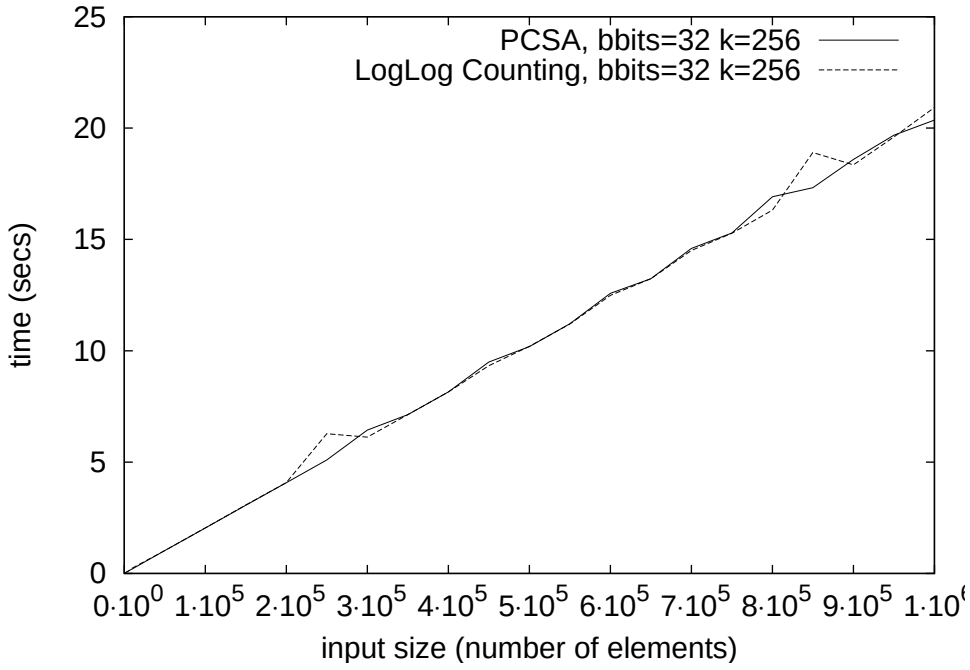
Το τελευταίο θέμα, της μη-υποστήριξης συγκεκριμένων αριθμητικών πράξεων, μας απασχόλησε ιδιαιτέρως και για αυτό το λόγο ψάξαμε τρόπους να υλοποιήσουμε τις περισσότερες από τις πράξεις που είναι απαραίτητες για τους αλγορίθμους. Αυτό όμως κρίθηκε αρκετά δύσκολο ζήτημα, αφού για παράδειγμα, η πράξη της ύψωσης πραγματικού αριθμού σε δύναμη στα περισσότερα πακέτα λογισμικού υλοποιείται στο επίπεδο συμβολικής γλώσσας (assembly). Βέβαια, τα όρια της μνήμης της συσκευής ήταν και πάλι ένα μεγάλο εμπόδιο.

Επομένως, μετά από πολύωρες προσπάθειες δεν καταφέραμε να εκπληρώσουμε τις παραπάνω ανάγκες της υλοποίησης, ώστε να μπορέσουμε να πάρουμε τα πειραματικά απο-

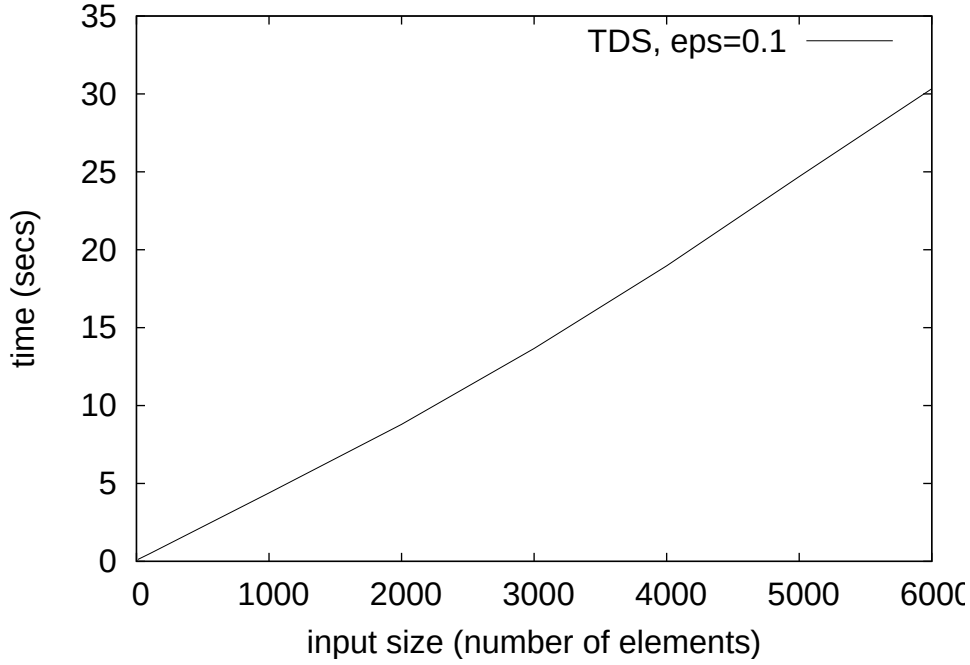
τελέσματα που είχαμε αρχικά σχεδιάσει.

4.1.2 Stargate NetBridge Gateway

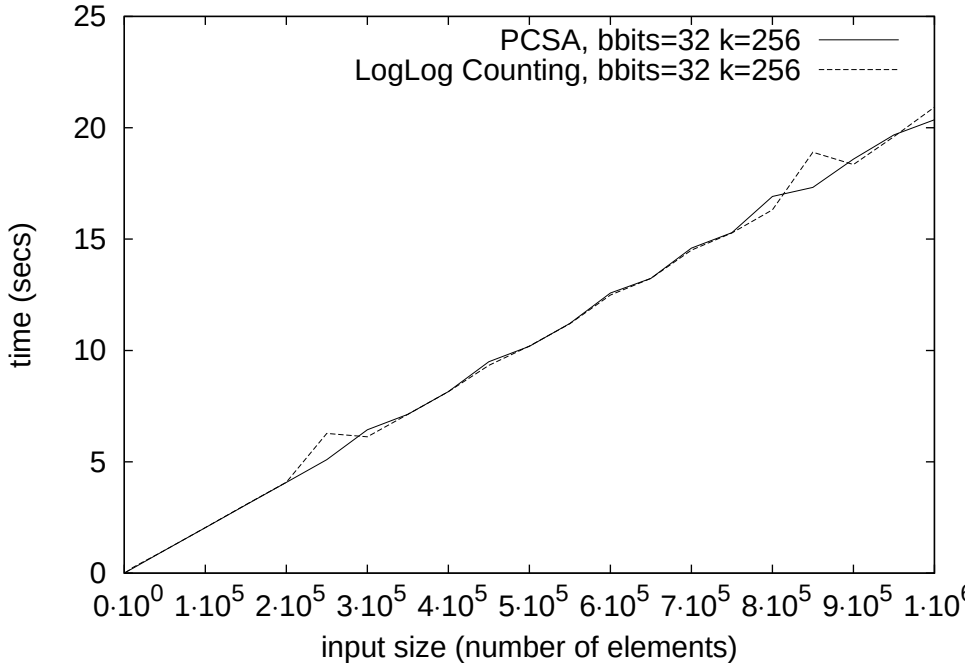
Γραφικές Παραστάσεις Χρόνου Χωρίς Διπλότυπα Παρακάτω φαίνονται οι γραφικές παραστάσεις των πειραματικών μετρήσεων χρόνου συναρτήσει του μεγέθους της εισόδου για τη συσκευή *Stargate Netbridge*. Παρατηρούμε ότι σε μια τόσο περιορισμένη σε υπολογιστικούς πόρους συσκευή η χρονική απόδοση των αλγορίθμων PCSA και Loglog Counting, σχήματα 4.1 και 4.3, είναι σαφώς καλύτερη από αυτή του TDS, σχήματα 4.2 και 4.4, αφού για μεγέθος 10^6 στοιχείων φτάνει τα ≈ 21 δευτερόλεπτα, τη στιγμή που για μέγεθος 6000 στοιχείων ο τελευταίος δίνει την τελική απάντηση μετά από 30 δευτερόλεπτα. Επιπλέον, παρατηρούμε ότι ενώ για τους δύο πιθανοτικούς αλγορίθμους ισχύουν τα ίδια όταν εισάγουμε 40% διπλότυπα στην είσοδο, για τον TDS από τα 30 δευτερόλεπτα που πραγματοποιεί χωρίς διπλότυπα, φτάνει περίπου στα 18 δευτερόλεπτα για 40% διπλότυπα.



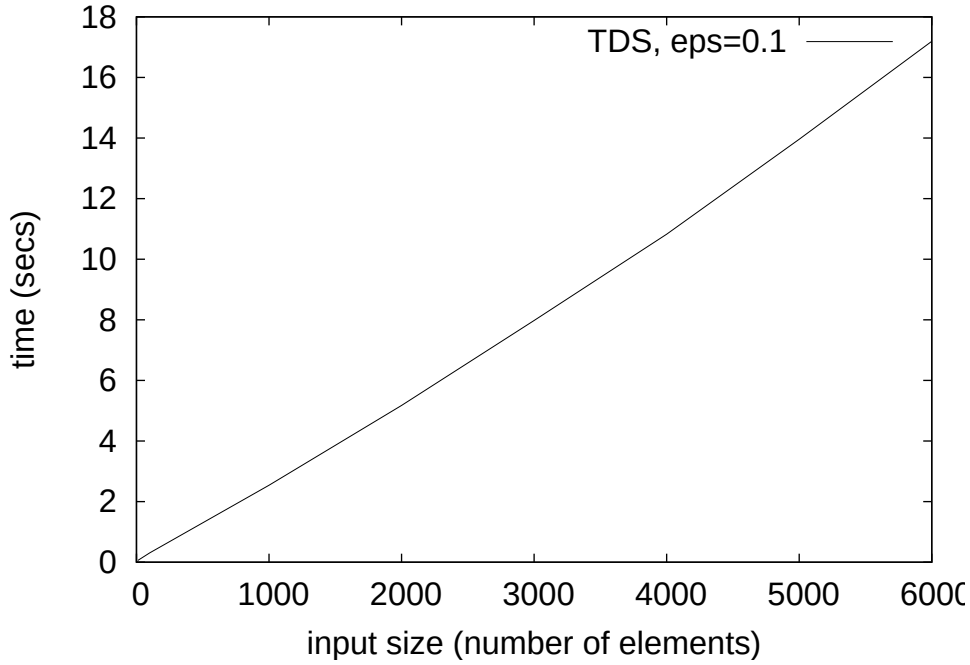
Σχήμα 4.1: Μετρήσεις χρόνου σε είσοδο χωρίς διπλότυπα στο **Stargate Netbridge**



Σχήμα 4.2: Μετρήσεις χρόνου σε είσοδο χωρίς διπλότυπα στο **Stargate Netbridge**

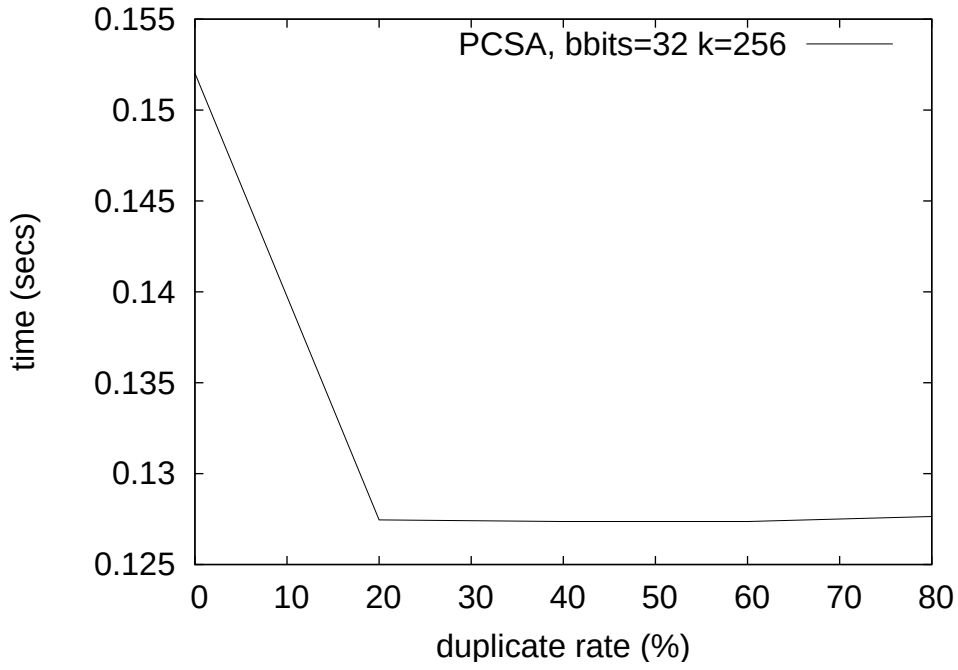


Σχήμα 4.3: Μετρήσεις χρόνου σε είσοδο με 40% διπλότυπα στο **Stargate Netbridge**

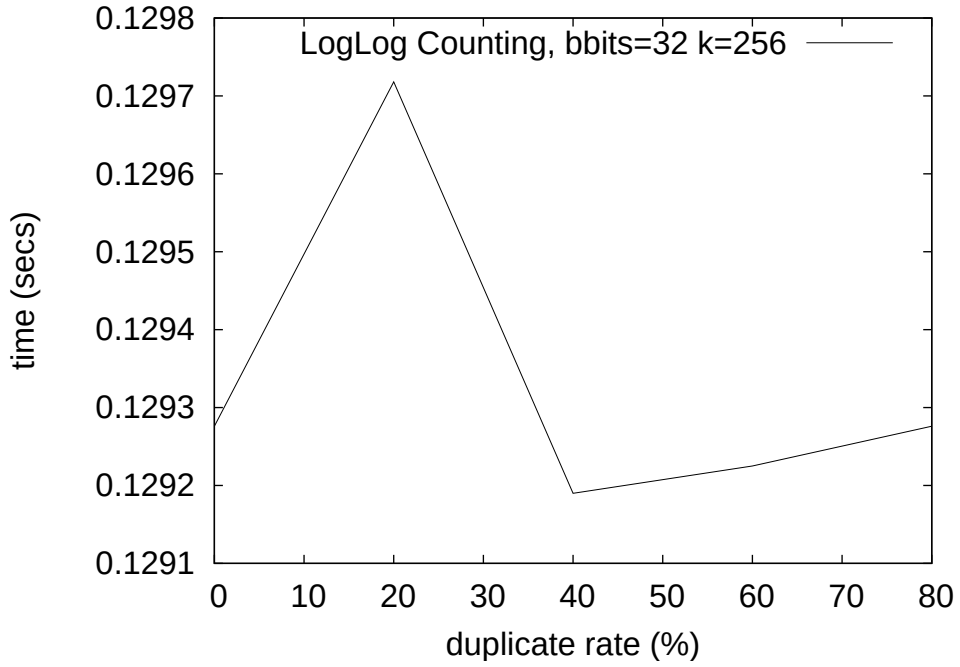


Σχήμα 4.4: Μετρήσεις χρόνου σε είσοδο με 40% διπλότυπα στο **Stargate Netbridge**

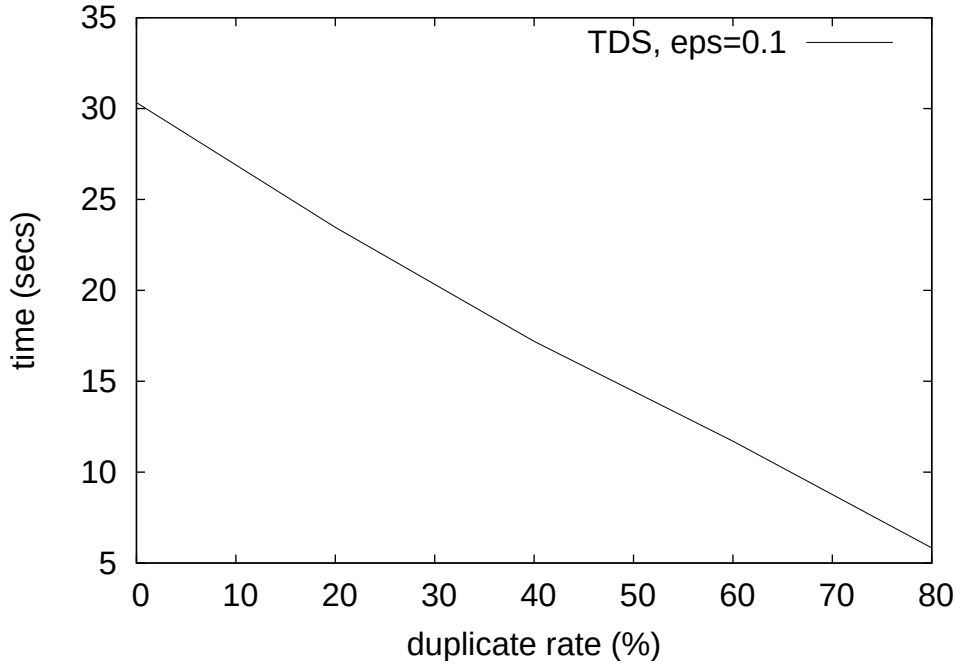
Γραφικές Παραστάσεις Χρόνου - Διπλότυπων Ακολουθούν οι γραφικές παραστάσεις της μέτρησης χρόνου συναρτήσει του ποσοστού διπλότυπων για ένα συγκεκριμένο μέγεθος εισόδου, $n = 6000$ στοιχεία. Παρατηρούμε ότι για τους δύο πιθανοτικούς αλγορίθμους το ποσοστό διπλότυπων στη ροή εισόδου δεν παίζει ουσιαστικά κανένα ρόλο, αφού για παράδειγμα για τον PCSA, σχήμα 4.5, ο χρόνος για 0% διπλότυπα είναι ≈ 0.1520 δευτερόλεπτα και για 80% είναι 0.1255 δευτερόλεπτα. Αντίστοιχα ισχύουν και για τον Loglog Counting. Από την άλλη, ο TDS ενώ για 0% διπλότυπα είναι στα 30 δευτερόλεπτα, με 80% διπλότυπα μειώνει το χρόνο κατά 24 δευτερόλεπτα.



Σχήμα 4.5: Χρονική απόδοση σε σχέση με ποσοστό διπλότυπων στο **Stargate Netbridge**

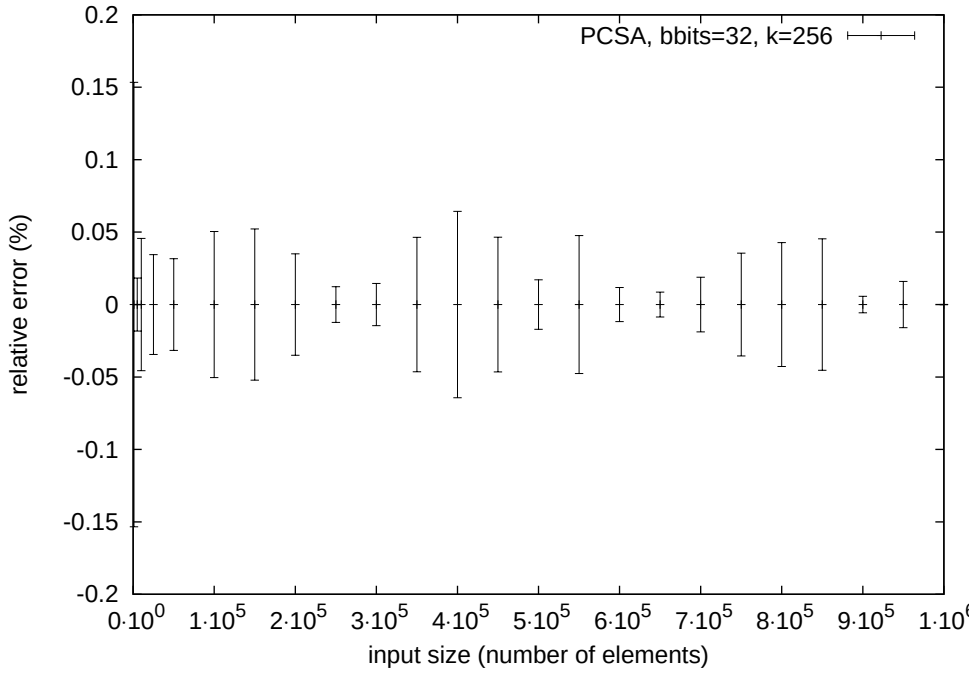


Σχήμα 4.6: Χρονική απόδοση σε σχέση με ποσοστό διπλότυπων στο **Stargate Netbridge**

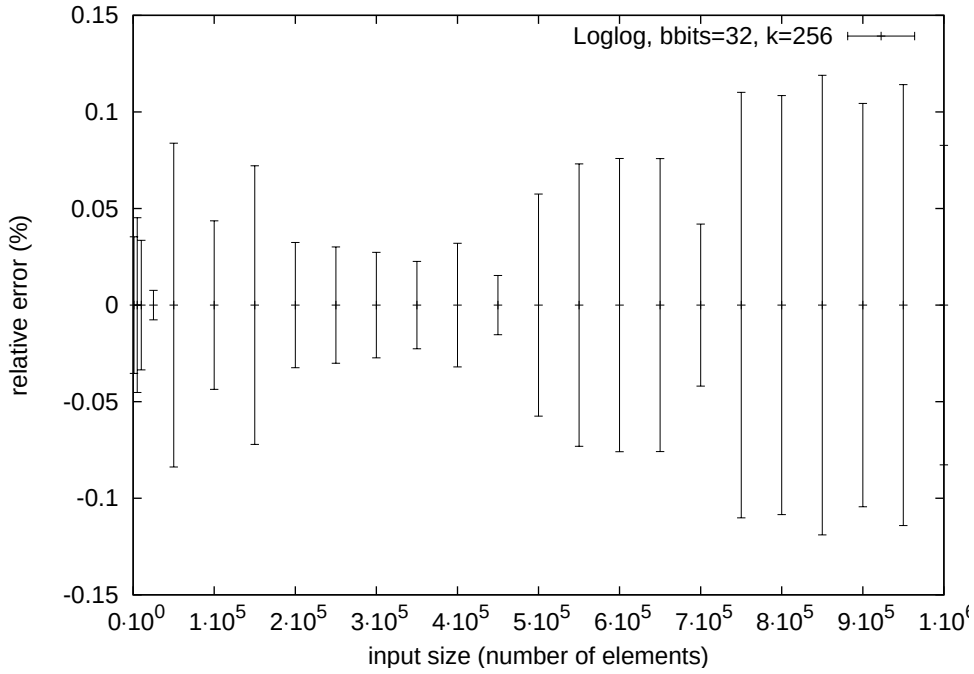


Σχήμα 4.7: Χρονική απόδοση σε σχέση με ποσοστό διπλότυπων στο **Stargate Netbridge**

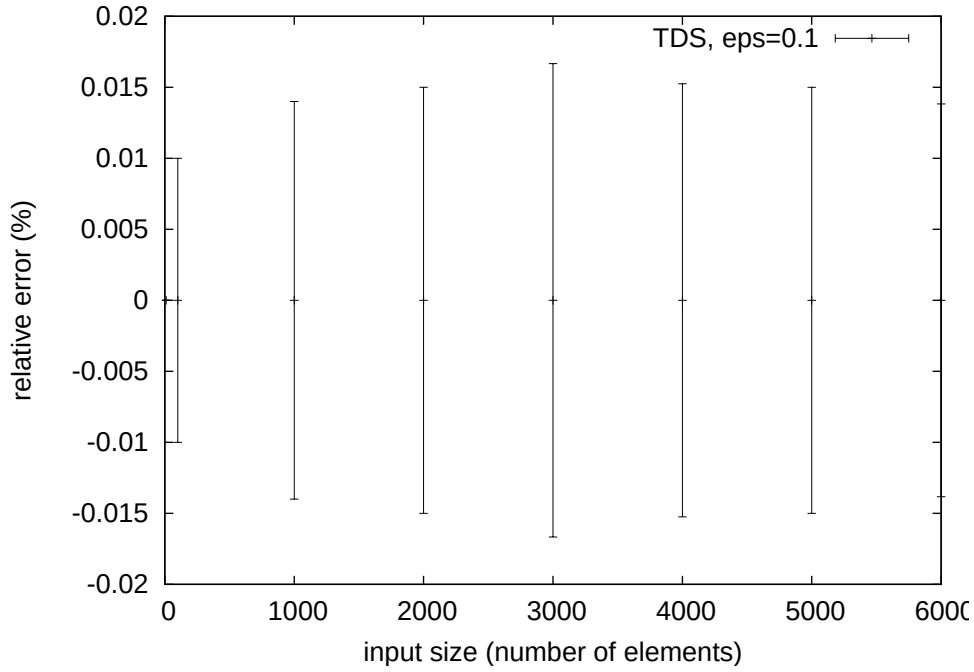
Γραφικές Παραστάσεις Σχετικού Σφάλματος Παρακάτω παρουσιάζονται οι γραφικές παραστάσεις του ποσοστού του σχετικού σφάλματος συναρτήσει του μεγέθους της εισόδου χωρίς και με εισαγωγή 40% διπλότυπων για κάθε αλγόριθμο. Από την σχηματική απεικόνιση των αποτελεσμάτων βλέπουμε ότι το ποσοστό σχετικού σφάλματος και για τους τρείς αλγορίθμους είναι πολύ μικρό, με μεγαλύτερο ποσοστό σφάλματος το 0.3% το οποίο παρουσιάζει ο Loglog Counting. Για τον PCSA το μέγιστο ποσοστό σφάλματος είναι 0.1%, ενώ για τον TDS είναι ακόμα μικρότερο, 0.03%. Με εισαγωγή 40% διπλότυπων τα ποσοστά σχετικού σφάλματος δεν επηρεάζονται για κανέναν από τους αλγορίθμους.



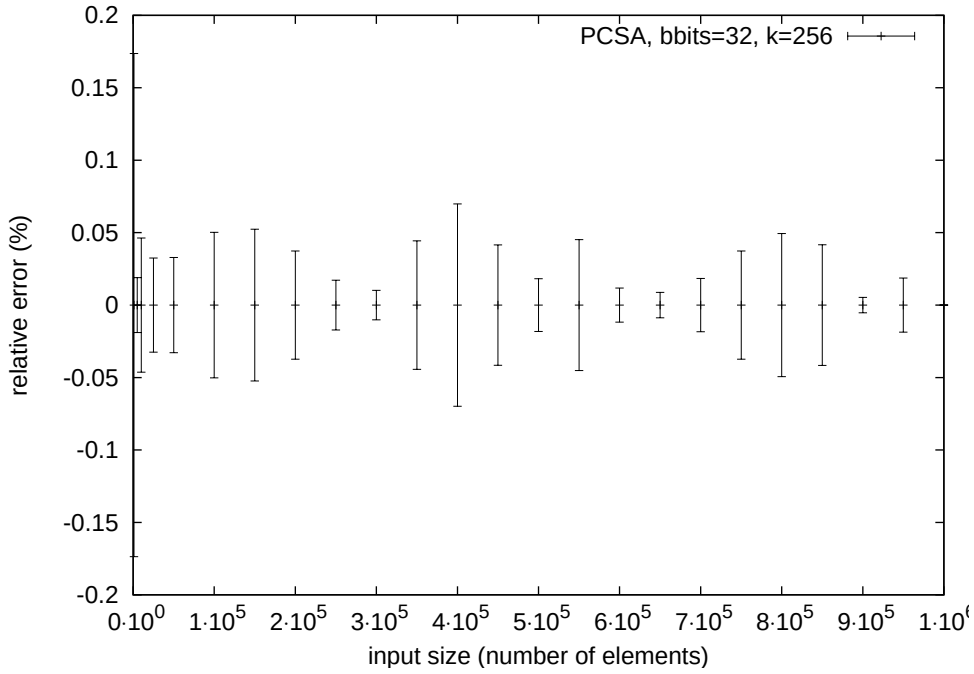
Σχήμα 4.8: Υπολογισμός σχετικού σφάλματος σε είσοδο χωρίς διπλότυπα στο **Stargate Netbridge**



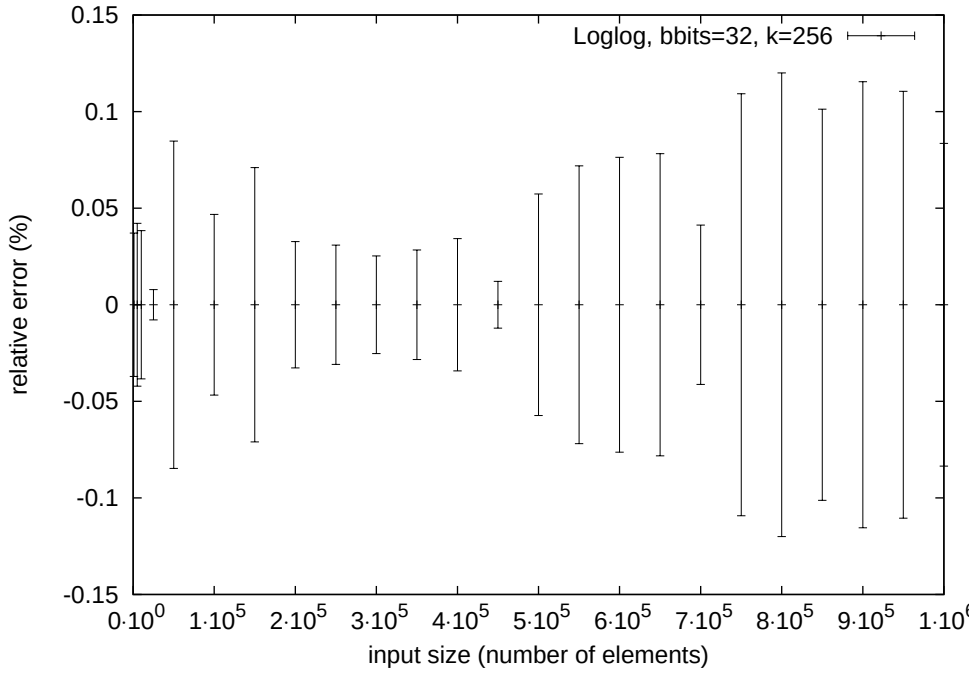
Σχήμα 4.9: Υπολογισμός σχετικού σφάλματος σε είσοδο χωρίς διπλότυπα στο **Stargate Netbridge**



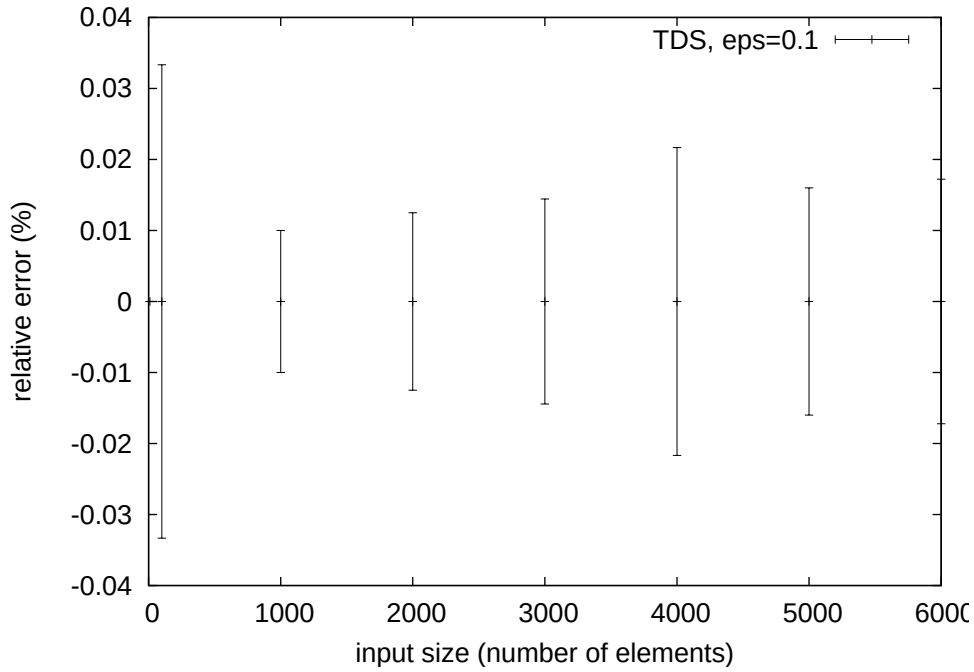
Σχήμα 4.10: Υπολογισμός σχετικού σφάλματος σε είσοδο χωρίς διπλότυπα στο **Stargate Netbridge**



Σχήμα 4.11: Υπολογισμός σχετικού σφάλματος σε είσοδο με 40% διπλότυπα στο **Stargate Netbridge**



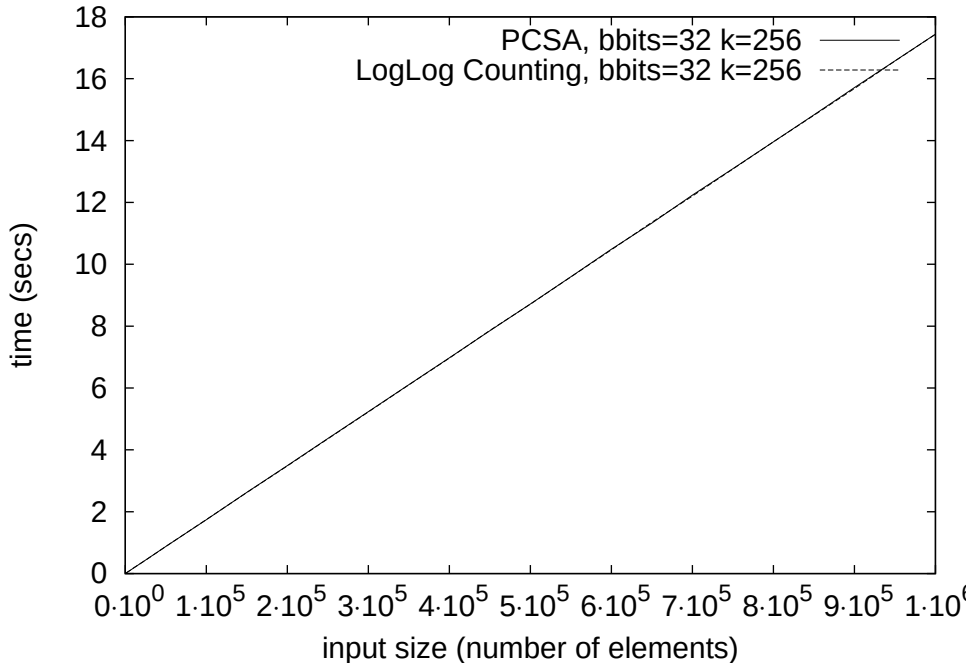
Σχήμα 4.12: Υπολογισμός σχετικού σφάλματος σε είσοδο με 40% διπλότυπα στο **Stargate Netbridge**



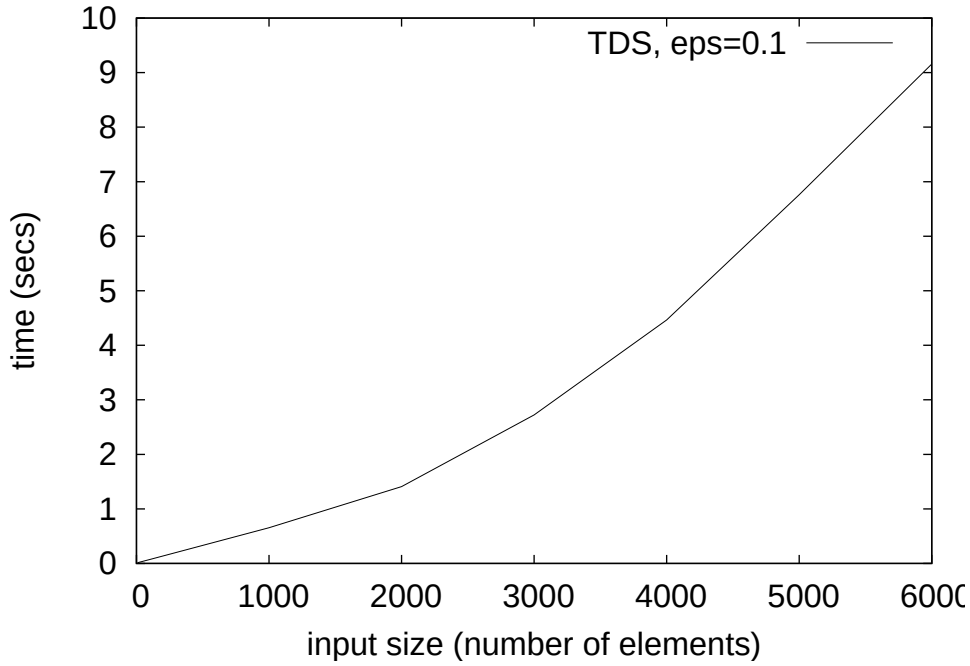
Σχήμα 4.13: Υπολογισμός σχετικού σφάλματος σε είσοδο με 40% διπλότυπα στο **Stargate Netbridge**

4.1.3 OpenMoko Neo Freerunner

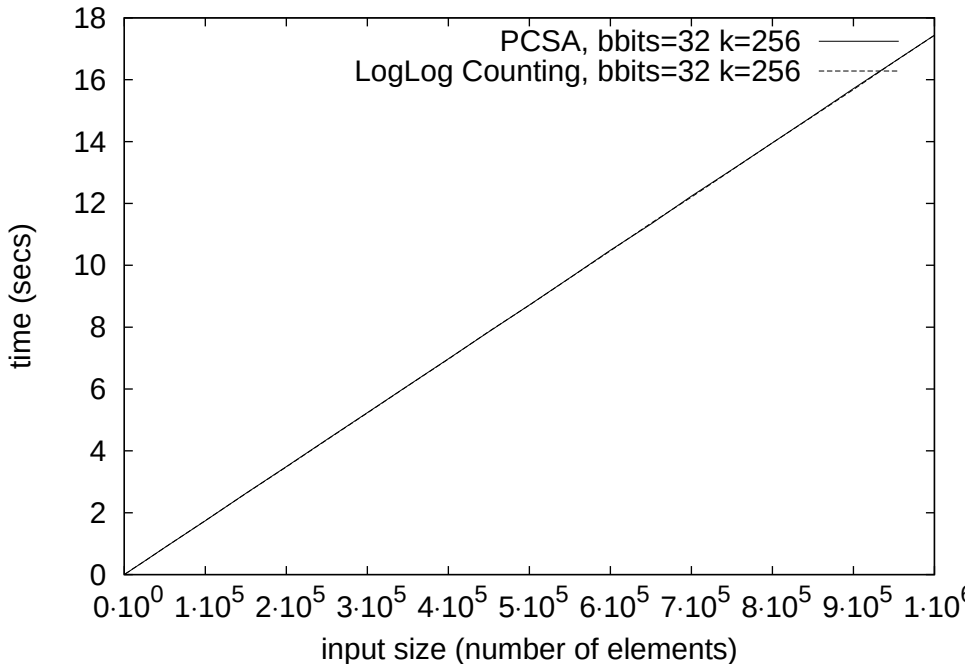
Γραφικές Παραστάσεις Χρόνου Παρακάτω φαίνονται οι γραφικές παραστάσεις των πειραματικών μετρήσεων χρόνου συναρτήσει του μεγέθους της εισόδου για τη συσκευή *OpenMoko Neo Freerunner*. Στο σχήματα 4.14 και 4.16 βλέπουμε ότι για το μέγεθος εισόδου 10^6 στοιχεία, οι δύο πιθανοτικοί αλγόριθμοι πραγματοποιούν τον υπολογισμό των διαφορετικών στοιχείων σε περίπου 17 δευτερόλεπτα, η εισαγωγή 40% διπλότυπων δεν τους επηρεάζει. Ο TDS αλγόριθμος χωρίς διπλότυπα φτάνει τα 9 δευτερόλεπτα και με 40% διπλότυπα είναι στα περίπου 4.3 δευτερόλεπτα.



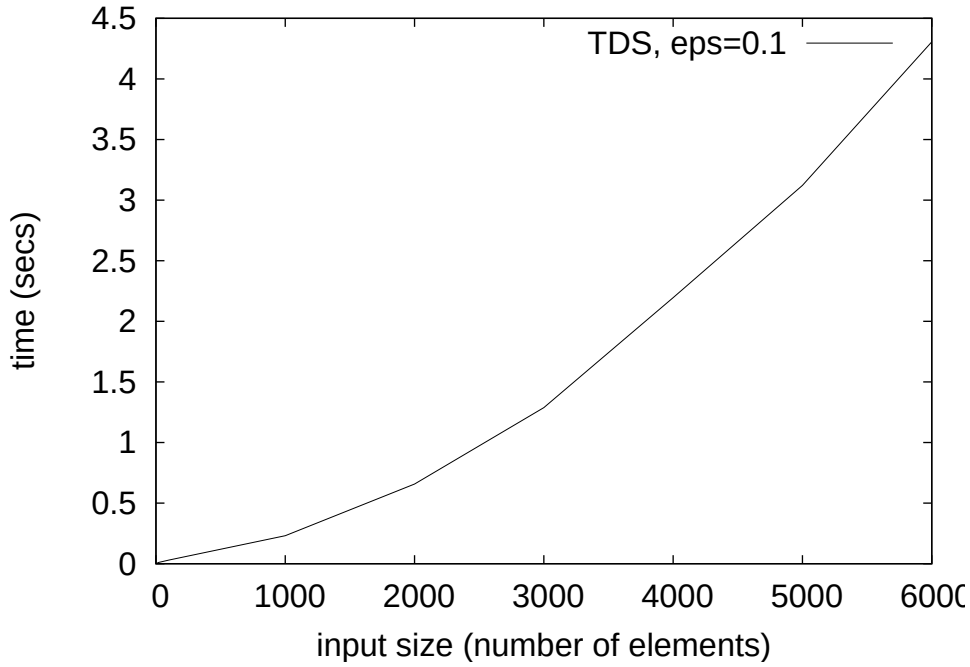
Σχήμα 4.14: Μετρήσεις χρόνου σε είσοδο χωρίς διπλότυπα στο **OpenMoko Neo Freerunner**



Σχήμα 4.15: Μετρήσεις χρόνου σε είσοδο χωρίς διπλότυπα στο **OpenMoko Neo Freerunner**

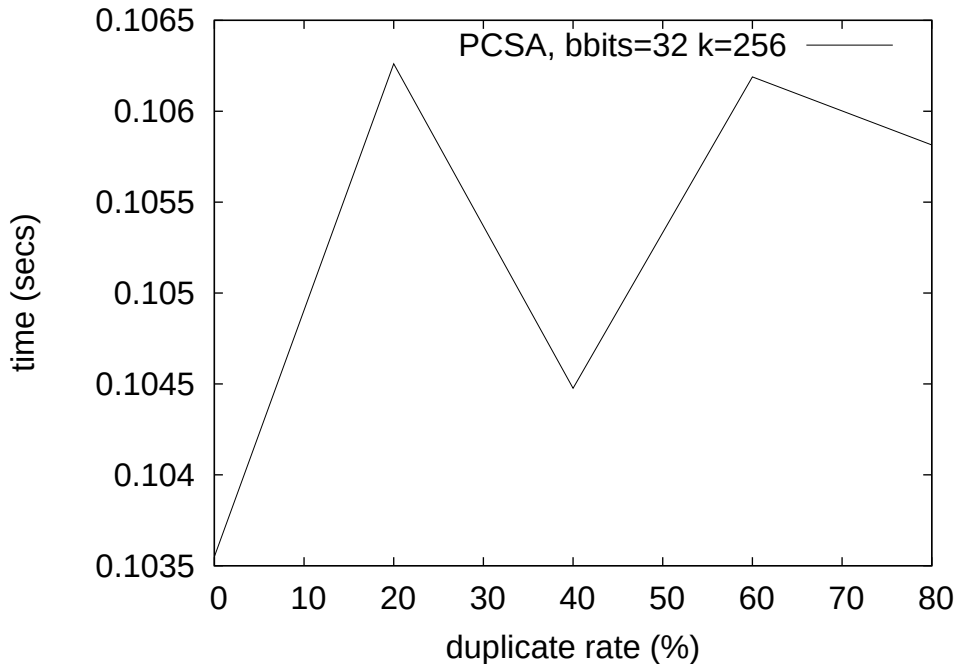


Σχήμα 4.16: Μετρήσεις χρόνου σε είσοδο με 40% διπλότυπα στο **OpenMoko Neo Freerunner**

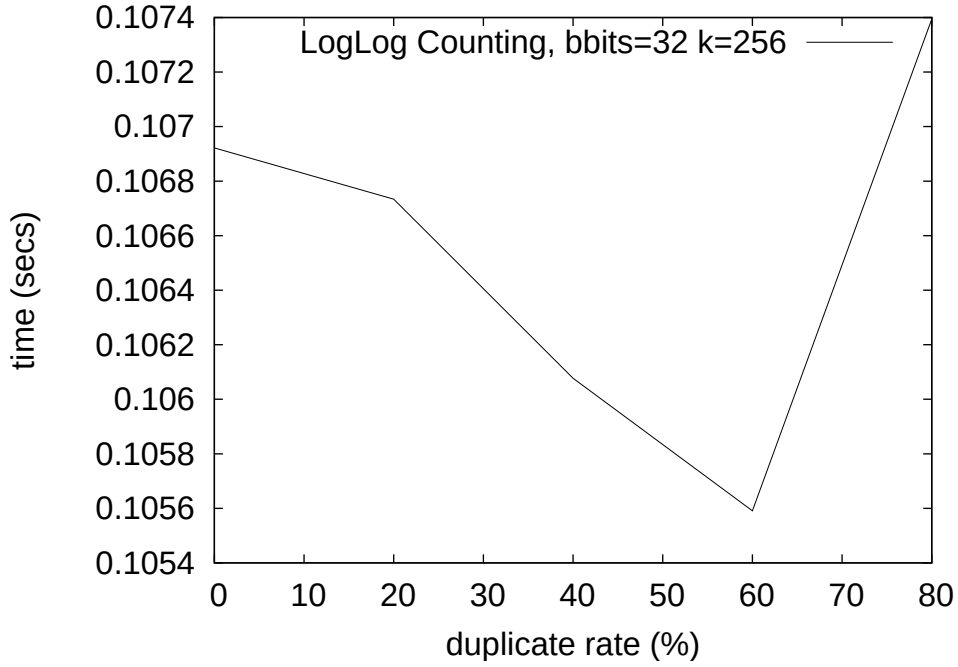


Σχήμα 4.17: Μετρήσεις χρόνου σε είσοδο με 40% διπλότυπα στο **OpenMoko Neo Freerunner**

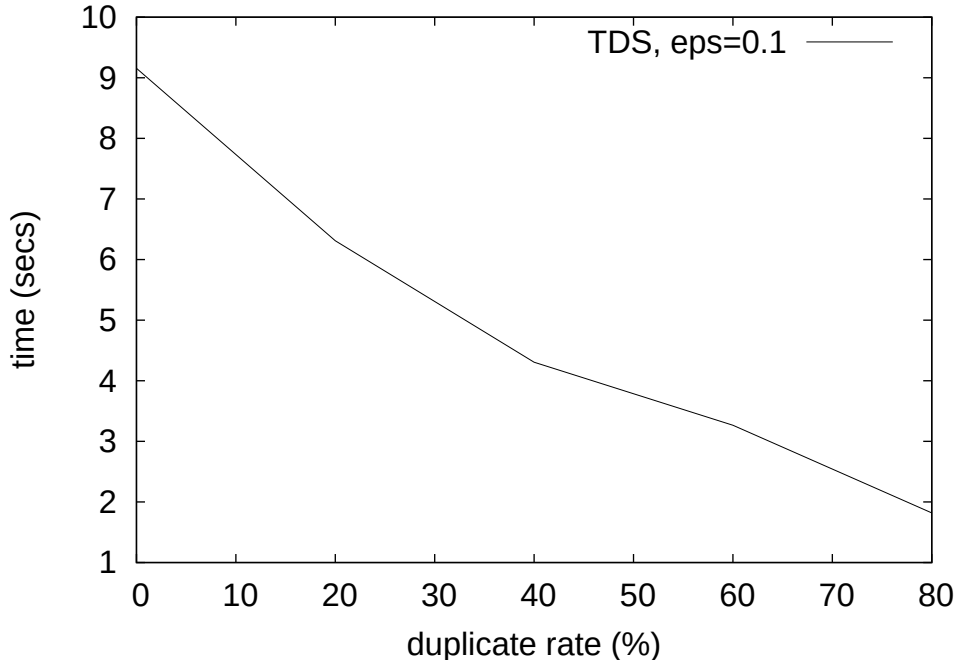
Γραφικές Παραστάσεις Χρόνου - Διπλότυπων Ακολουθούν οι γραφικές παραστάσεις της μέτρησης χρόνου συναρτήσει του ποσοστού διπλότυπων για ένα συγκεκριμένο μέγεθος εισόδου, $n = 6000$ στοιχεία. Στα σχήματα 4.18 και 4.19 παρατηρούμε ότι ο PCSA έχει περίπου 0.025 δευτερόλεπτα διακύμανση, ενώ ο Loglog Counting έχει διακύμανση περίπου 0.017 δευτερόλεπτα αντίστοιχα. Ο αλγόριθμος TDS, όπως βλέπουμε στο σχήμα 4.20, έχει μεγάλη βελτίωση ανάμεσα στις χρονικές αποδόσεις μεταξύ $0\% - 80\%$ διπλότυπων της τάξης των 7 δευτερολέπτων.



Σχήμα 4.18: Συμπεριφορά αλγορίθμου σε σχέση με ποσοστό διπλότυπων στο **OpenMoko Neo Freerunner**

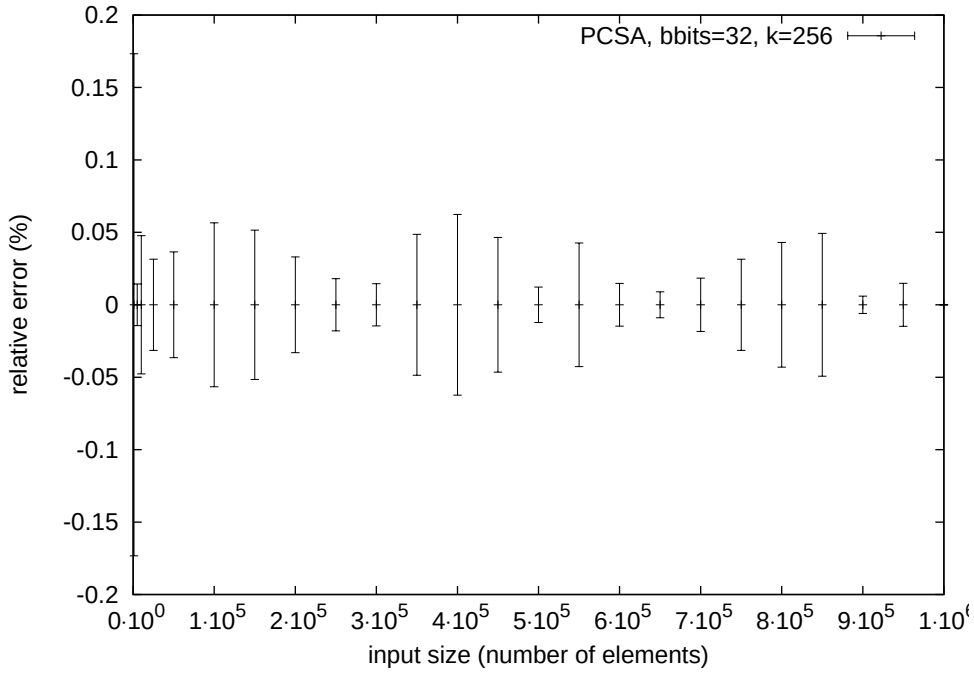


Σχήμα 4.19: Συμπεριφορά αλγορίθμου σε σχέση με ποσοστό διπλότυπων στο **OpenMoko Neo Freerunner**

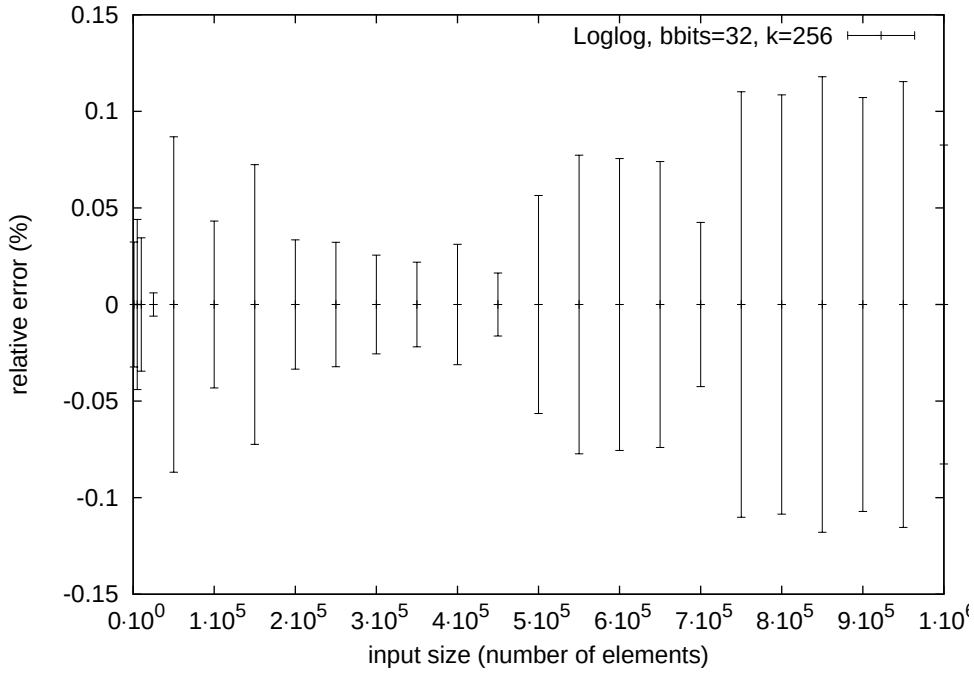


Σχήμα 4.20: Συμπεριφορά αλγορίθμου σε σχέση με ποσοστό διπλότυπων στο **OpenMoko Neo Freerunner**

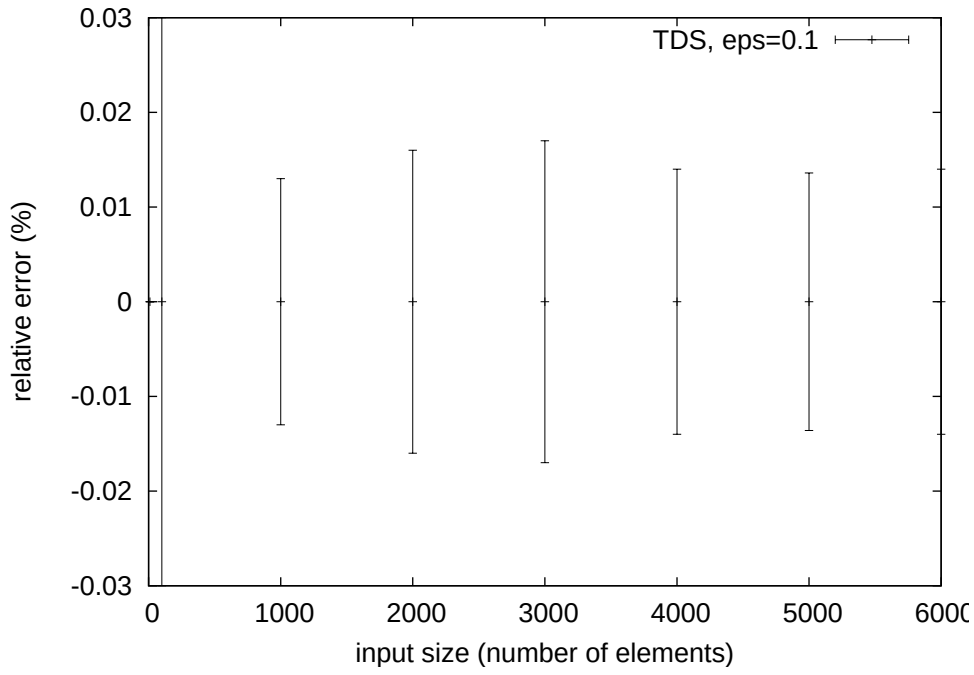
Γραφικές Παραστάσεις Σχετικού Σφάλματος Παρακάτω παρουσιάζονται οι γραφικές παραστάσεις σχετικού σφάλματος συναρτήσει του μεγέθους της εισόδου χωρίς και με εισαγωγή 40% διπλότυπων για κάθε αλγόριθμο. Στα σχήματα 4.21, 4.22, 4.24 και 4.25 παρατηρούμε ότι το μέγιστο ποσοστό σχετικού σφάλματος για τον PCSA είναι $\approx 0.1\%$, ενώ για τον Loglog Counting είναι $\approx 0.3\%$, με και χωρίς διπλότυπα. Για τον TDS το μέγιστο ποσοστό σχετικού σφάλματος φτάνει στο 0.03%.



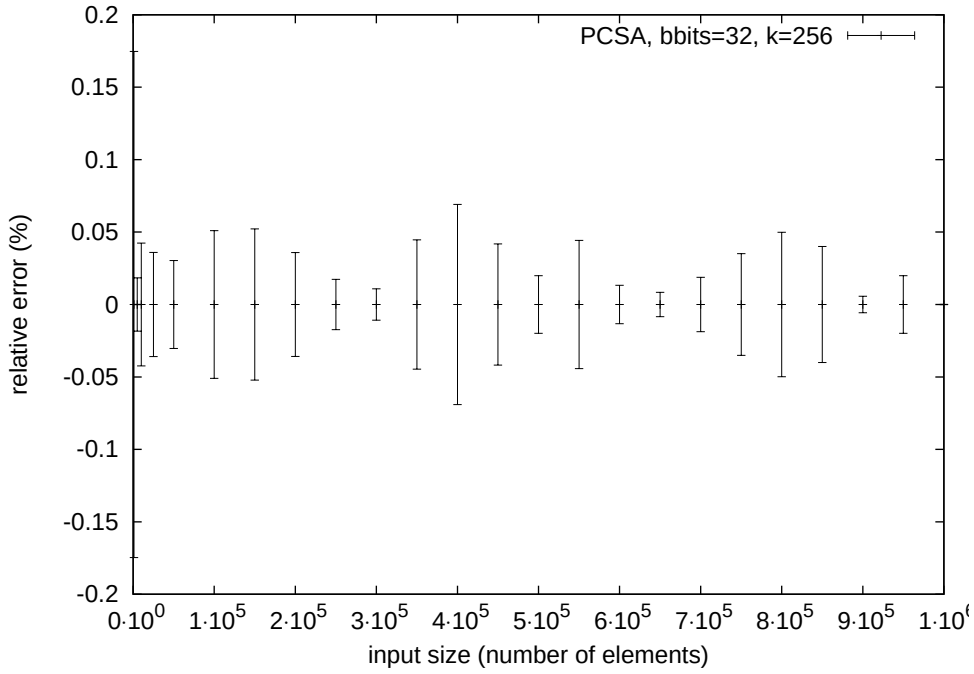
Σχήμα 4.21: Υπολογισμός σχετικού σφάλματος σε είσοδο χωρίς διπλότυπα στο **OpenMoko Neo Freerunner**



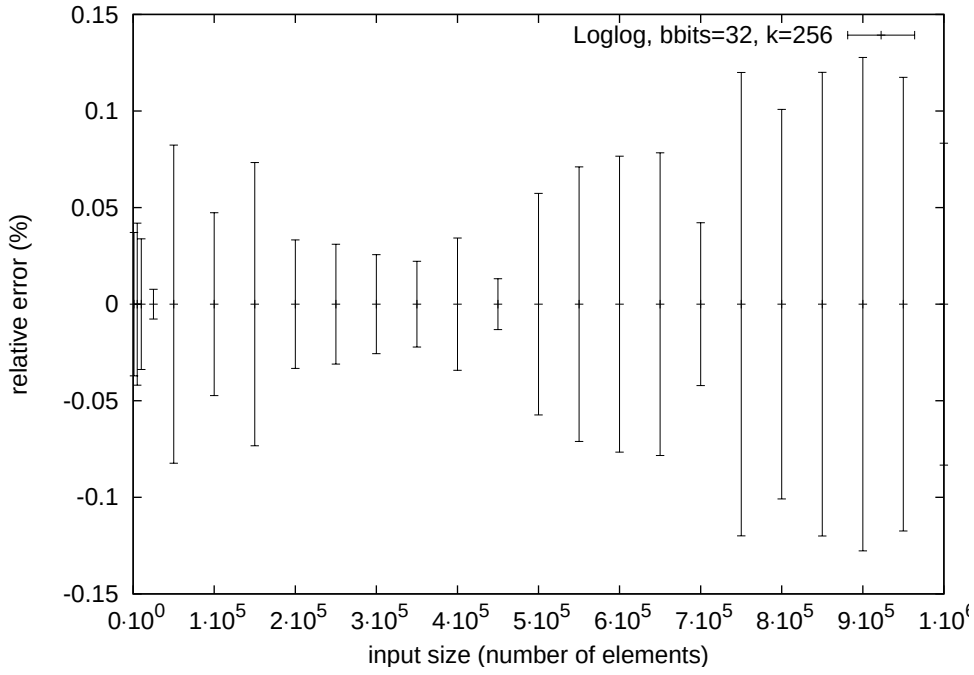
Σχήμα 4.22: Υπολογισμός σχετικού σφάλματος σε είσοδο χωρίς διπλότυπα στο **OpenMoko Neo Freerunner**



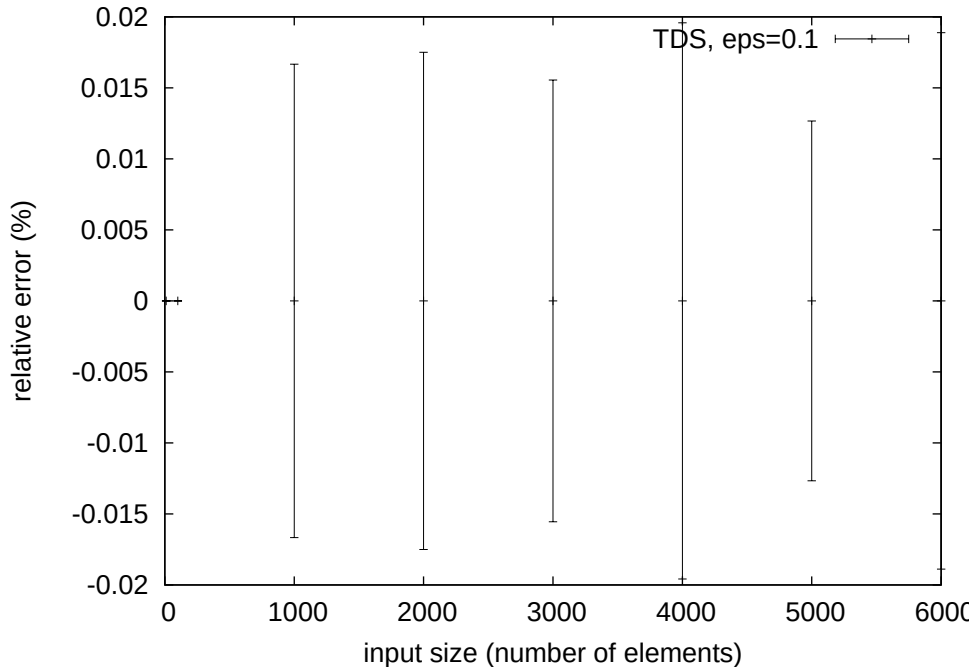
Σχήμα 4.23: Υπολογισμός σχετικού σφάλματος σε είσοδο χωρίς διπλότυπα στο **OpenMoko Neo Freerunner**



Σχήμα 4.24: Υπολογισμός σχετικού σφάλματος σε είσοδο με 40% διπλότυπα στο **OpenMoko Neo Freerunner**



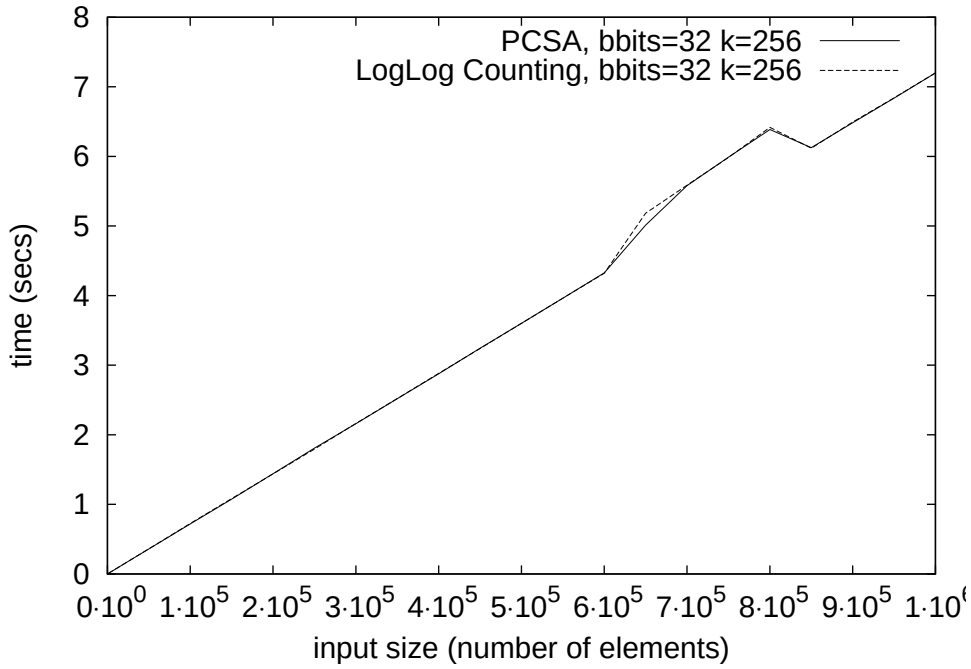
Σχήμα 4.25: Υπολογισμός σχετικού σφάλματος σε είσοδο με 40% διπλότυπα στο **OpenMoko Neo Freerunner**



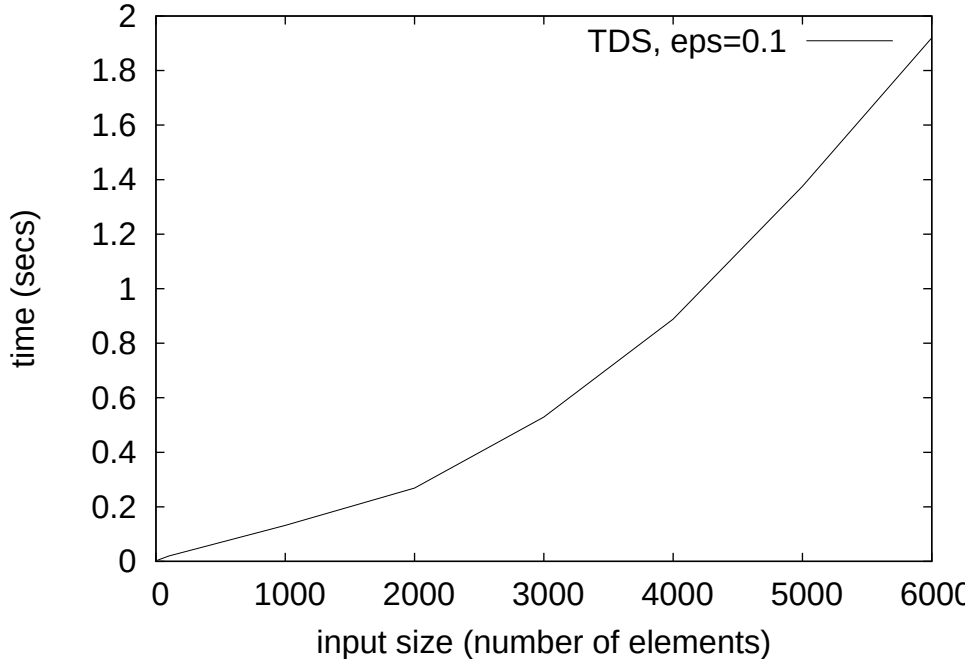
Σχήμα 4.26: Υπολογισμός σχετικού σφάλματος σε είσοδο με 40% διπλότυπα στο **OpenMoko Neo Freerunner**

4.1.4 PC Engine Alix

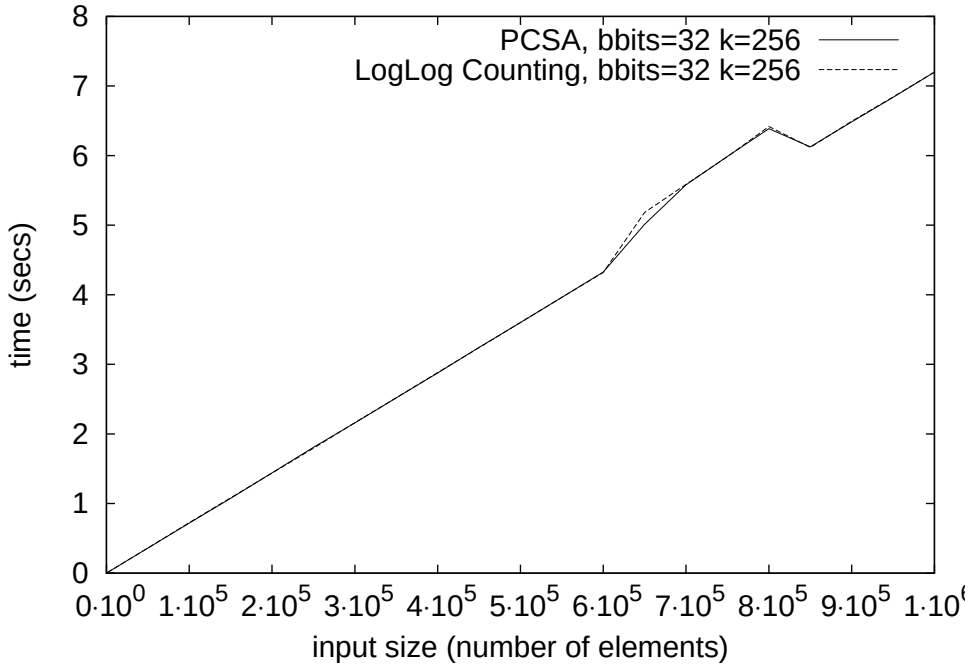
Γραφικές Παραστάσεις Χρόνου Παρακάτω φαίνονται οι γραφικές παραστάσεις των πειραματικών μετρήσεων χρόνου συναρτήσει του μεγέθους της εισόδου για τη συσκευή *PC Engine Alix*. Στα σχήματα 4.27 και 4.29 παρατηρούμε ότι για μέγεθος ροής εισόδου 10^6 στοιχεία οι δύο πιθανοτικοί αλγόριθμοι δίνουν την τελική απάντηση μετά από 7.2 δευτερόλεπτα περίπου, χωρίς να επηρεάζονται από εισαγωγή διπλότυπων. Από την άλλη, ο αλγόριθμος TDS επηρεάζεται από την εισαγωγή 40% διπλότυπων και ενώ χωρίς διπλότυπα απέδιδε στα 1.8 δευτερόλεπτα, με διπλότυπα αποδίδει 1 δευτερόλεπτο πιο γρήγορα για μέγεθος $n = 6000$.



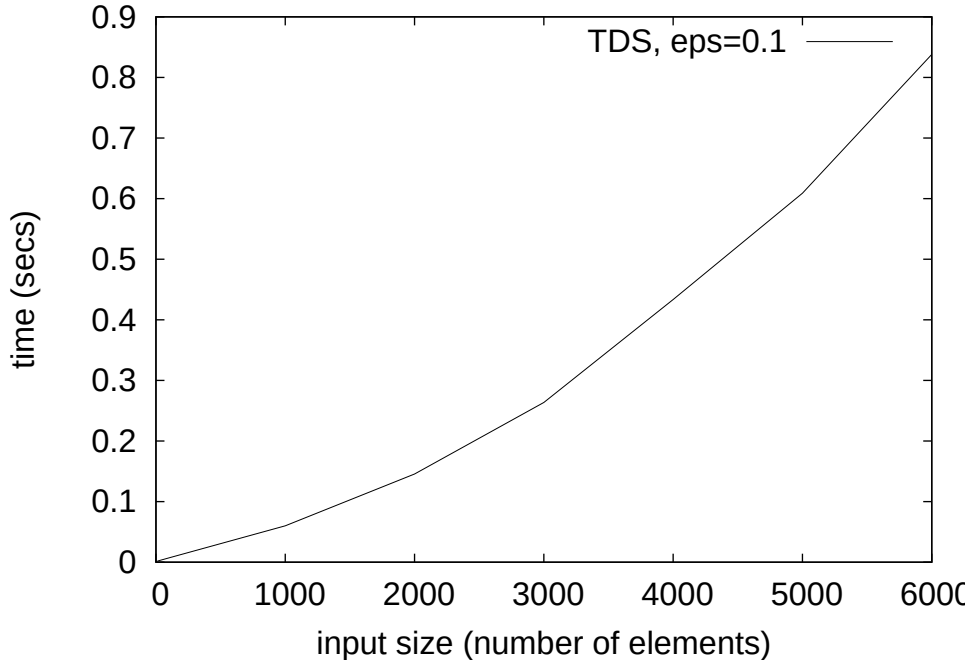
Σχήμα 4.27: Μετρήσεις χρόνου σε είσοδο χωρίς διπλότυπα στο **PC Engine Alix**



Σχήμα 4.28: Μετρήσεις χρόνου σε είσοδο χωρίς διπλότυπα στο **PC Engine Alix**

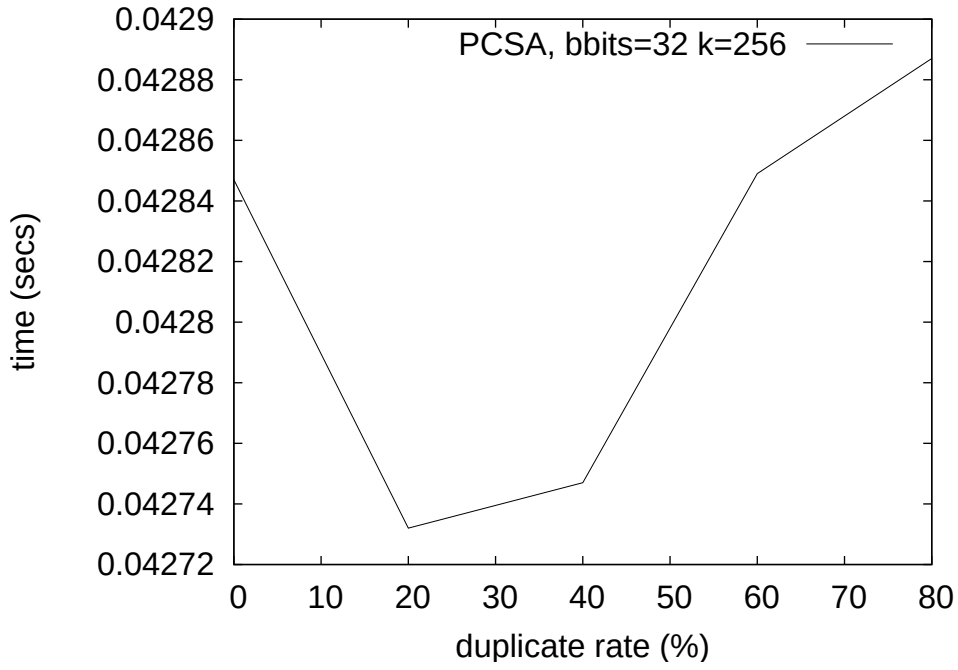


Σχήμα 4.29: Μετρήσεις χρόνου σε είσοδο με 40% διπλότυπα στο **PC Engine Alix**

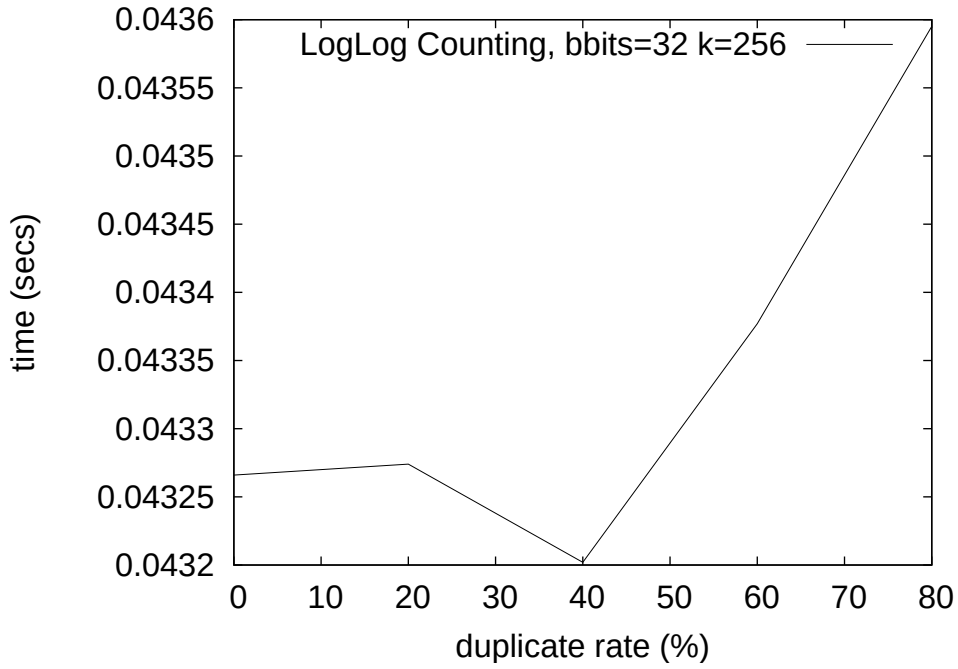


Σχήμα 4.30: Μετρήσεις χρόνου σε είσοδο με 40% διπλότυπα στο **PC Engine Alix**

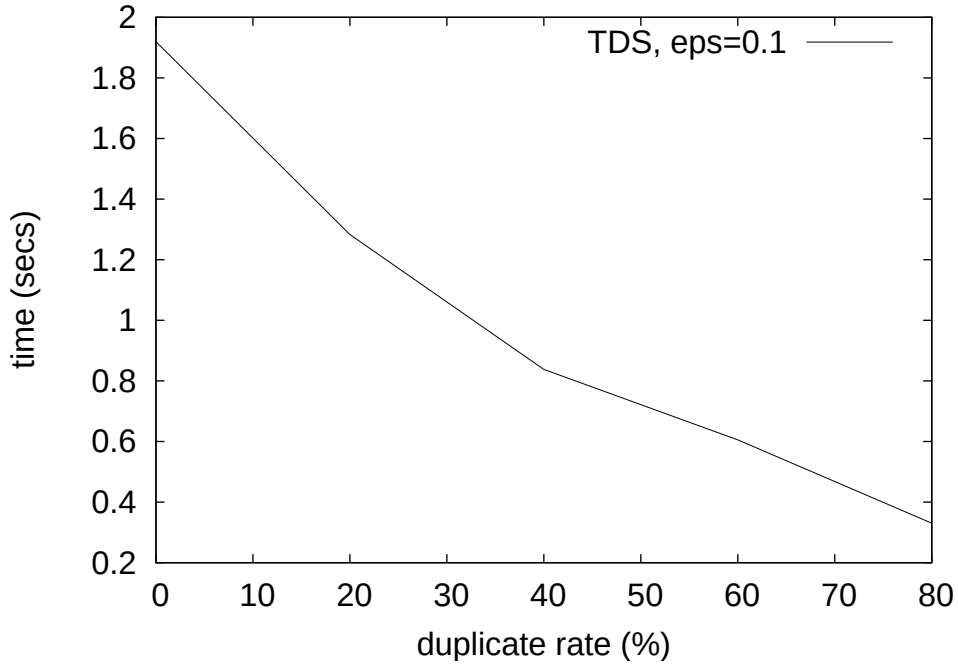
Γραφικές Παραστάσεις Χρόνου - Διπλότυπων Ακολουθούν οι γραφικές παραστάσεις της μέτρησης χρόνου συναρτήσει του ποσοστού διπλότυπων για ένα συγκεκριμένο μέγεθος εισόδου, $n = 6000$ στοιχεία. Στα σχήματα 4.31 και 4.32 παρατηρούμε ότι η μέγιστη διακύμανση χρόνου σε σχέση με το ποσοστό διπλότυπων για τον PCSA είναι 0.0015 δευτερόλεπτα, ενώ για τον Loglog Counting είναι 0.0004 δευτερόλεπτα αντίστοιχα. Για τον TDS όμως η διαφορά στο χρόνο με την αύξηση των διπλότυπων είναι πολύ μεγάλη, συγκεκριμένα ενώ με 0% διπλότυπα η χρονική απόδοση είναι 1.9 δευτερόλεπτα με 80% διπλότυπα μειώνεται στα 0.3 δευτερόλεπτα.



Σχήμα 4.31: Συμπεριφορά αλγορίθμου σε σχέση με ποσοστό διπλότυπων στο **PC Engine Alix**

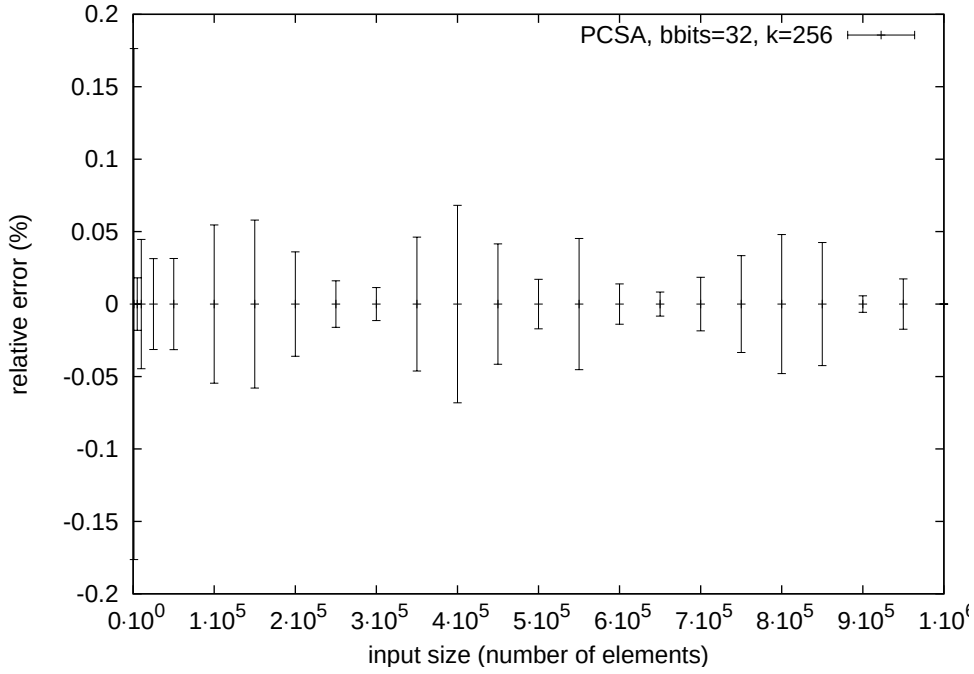


Σχήμα 4.32: Συμπεριφορά αλγορίθμου σε σχέση με ποσοστό διπλότυπων στο **PC Engine Alix**

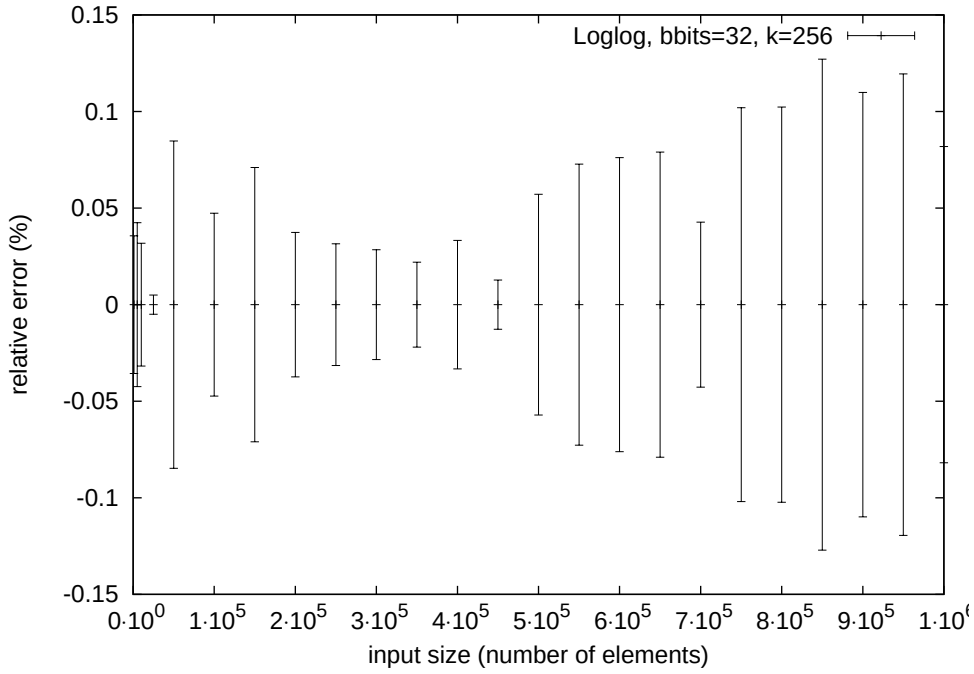


Σχήμα 4.33: Συμπεριφορά αλγορίθμου σε σχέση με ποσοστό διπλότυπων στο **PC Engine Alix**

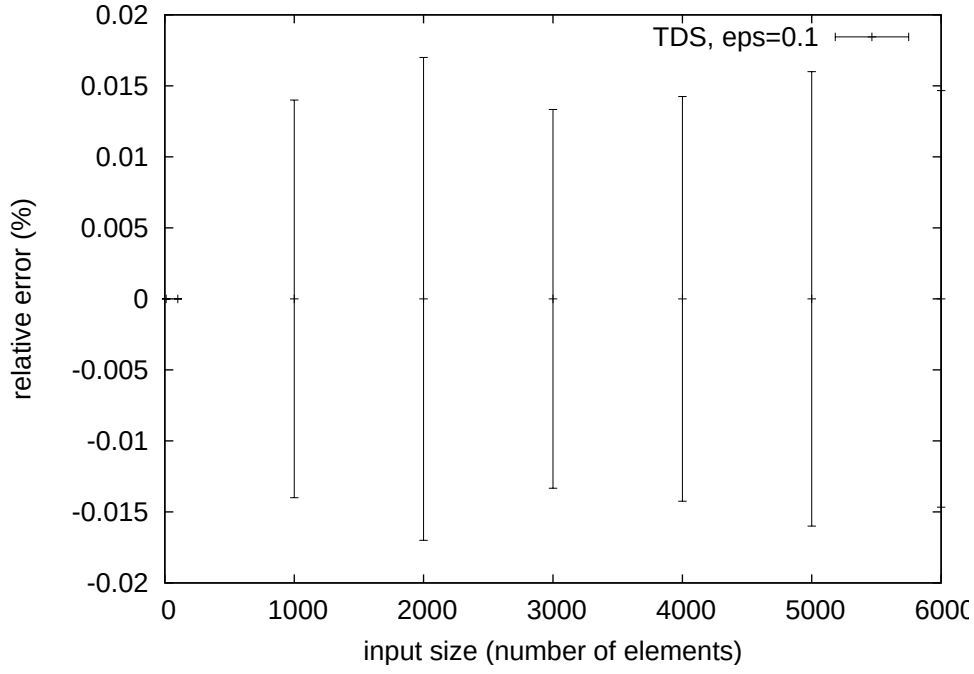
Γραφικές Παραστάσεις Σχετικού Σφάλματος Παρακάτω παρουσιάζονται οι γραφικές παραστάσεις σχετικού σφάλματος συναρτήσει του μεγέθους της εισόδου χωρίς και με εισαγωγή 40% διπλότυπων για κάθε αλγόριθμο. Στα σχήματα 4.34, 4.35, 4.37 και 4.38 παρατηρούμε ότι το μέγιστο ποσοστό σχετικού σφάλματος για τον PCSA είναι $\approx 0.125\%$, ενώ για τον Loglog Counting είναι $\approx 0.25\%$ με 40% και χωρίς διπλότυπα. Για τον TDS το μέγιστο ποσοστό σχετικού σφάλματος είναι μικρότερο και φτάνει στο 0.0325%, είτε υπάρχει εισαγωγή διπλότυπων είτε όχι.



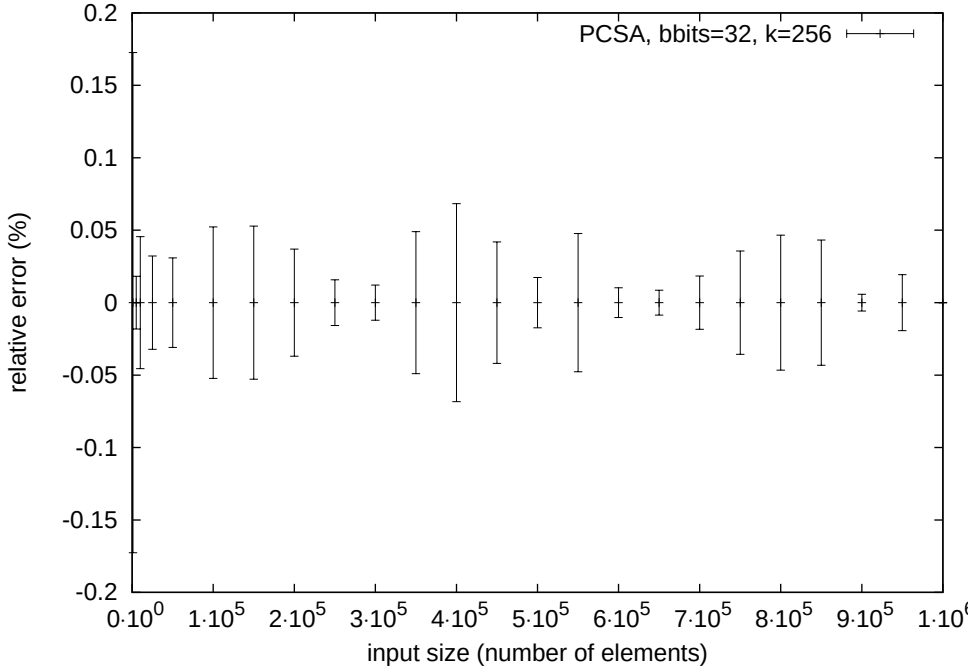
Σχήμα 4.34: Υπολογισμός σχετικού σφάλματος σε είσοδο χωρίς διπλότυπα στο **PC Engine Alix**



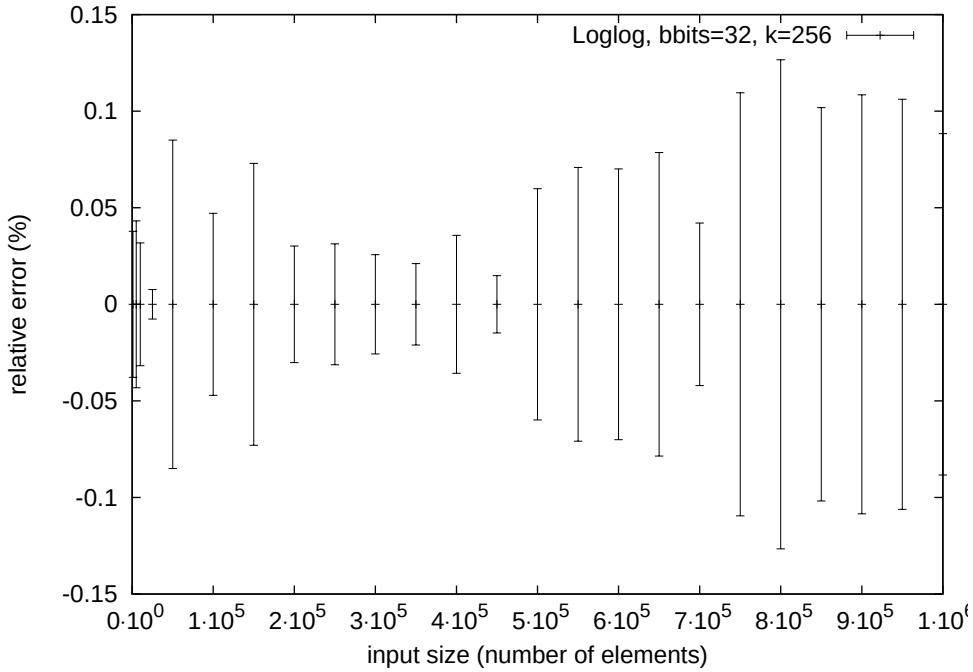
Σχήμα 4.35: Υπολογισμός σχετικού σφάλματος σε είσοδο χωρίς διπλότυπα στο **PC Engine Alix**



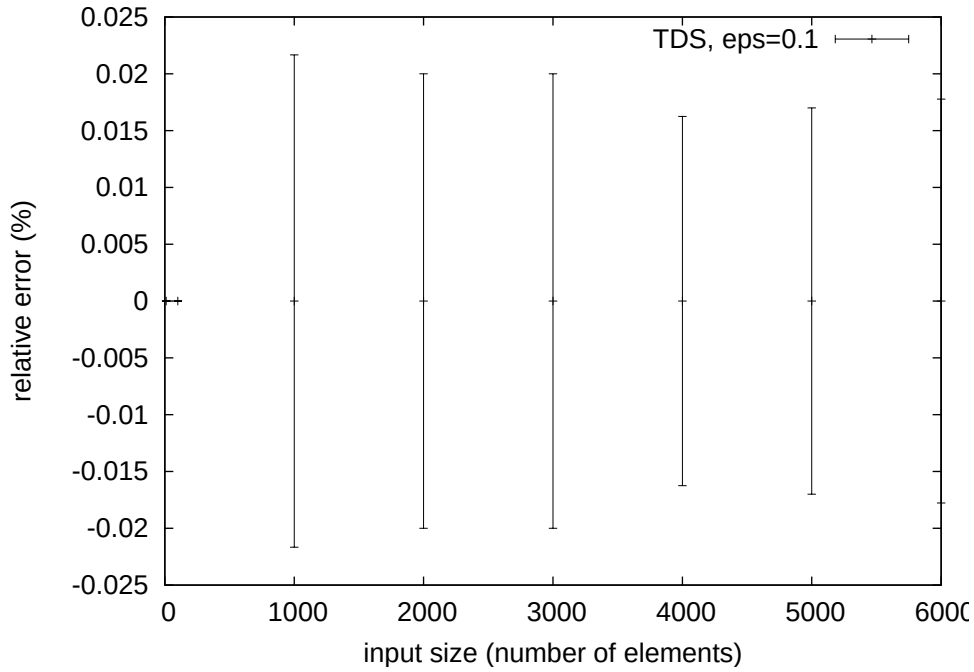
Σχήμα 4.36: Υπολογισμός σχετικού σφάλματος σε είσοδο χωρίς διπλότυπα στο **PC Engine Alix**



Σχήμα 4.37: Υπολογισμός σχετικού σφάλματος σε είσοδο με 40% διπλότυπα στο **PC Engine Alix**



Σχήμα 4.38: Υπολογισμός σχετικού σφάλματος σε είσοδο με 40% διπλότυπα στο **PC Engine Alix**



Σχήμα 4.39: Υπολογισμός σχετικού σφάλματος σε είσοδο με 40% διπλότυπα στο **PC Engine Alix**

4.2 Μέγεθος Εισόδου και Μετρήσεις Χρόνου

Το μέγεθος της ροής εισόδου γνωρίζουμε ότι παίζει σημαντικό ρόλο στη χρονική πολυπλοκότητα των αλγορίθμων, και φυσικά είναι απαραίτητο να αξιολογήσουμε την τελευταία, αφού η αποδοτικότητα σε χρόνο είναι θεμελιώδης απαίτηση σε πολλές περιπτώσεις εφαρμογών τέτοιων δικτύων.

Ας δούμε, λοιπόν, μερικές γραφικές παραστάσεις που δείχνουν τη γενική συμπεριφορά των αλγορίθμων ανεξάρτητα πλατφόρμας στην οποία πραγματοποιούνται τα πειράματα. Στα σχήματα 4.14 και 4.15, παρατηρούμε ότι στη συσκευή *OpenMoko Neo Freerunner* ο χρόνος, που έχει μετρηθεί για τα διάφορα μεγέθη ροής εισόδου χωρίς διπλότυπα, είναι ανάλογος του μεγέθους και για τους τρείς αλγορίθμους. Αυτό βέβαια, δε θα συνέβαινε αν για τον TDS αλγόριθμο είχαμε στη διάθεσή μας τη σωστή συνάρτηση $RangeSample$, με την οποία ο χρόνος της κάθε ενημέρωσης του sketch θα μειωνόταν με βάση ένα παράγοντα $O(\log |r_e^c|)$, ο οποίος είναι τώρα $O(|r_e^c|)$. Το ίδιο παρατηρείται και στις άλλες δύο συσκευές με διαφορετικούς, φυσικά, χρόνους αφού οι συσκευές μεταξύ τους έχουν περίβλεπτες διαφορές σε υπολογιστική δύναμη και χωρητικότητα μνήμης.

Αξιοσημείωτο είναι το γεγονός ότι οι δύο πιθανοτικοί αλγόριθμοι, αν και έχουν χρονική πολυπλοκότητα ανάλογη του μεγέθους της ροής εισόδου, n, οι χρόνοι τους είναι άκρως ικανοποιητικοί για δίκτυα περιορισμένων πόρων, αφού στην πιο αργή συσκευή, την Stargate Netbrigde δικτυακή πύλη, ο χρόνος απόκρισης για την εκτίμηση της μέτρησης διαφορετικών τιμών σε ροή εισόδου 400000 στοιχείων είναι περίπου 7.5 δευτερόλεπτα. Αυτό σημαίνει ότι ακόμα και σε δίκτυα αισθητήρων, στα οποία παράγονται τεράστιες ποσότητες δεδομένων, θα μπορούσαν να χρησιμοποιηθούν επιτυχώς οι δύο πιθανοτικοί αλγόριθμοι. Από την άλλη, ο αλγόριθμος TDS, δεν ενδείκνυται για χρήση σε τέτοιου είδους δίκτυα με σκοπό τη μέτρηση διαφορετικών τιμών της ροής εισόδου μόνο, αφού η χρονική απόδοσή του στις περισσότερες περιπτώσεις είναι απαγορευτική. Θα ήταν πιο φρόνιμο και αποδοτικό να χρησιμοποιηθεί ο αλγόριθμος αυτός για εκτίμηση συναθροιστικών συναρτήσεων, αφού εκεί τουλάχιστον η απάντηση που παρέχει ο αλγόριθμος είναι ολοκληρωμένη και εντός μικρών ορίων σφάλματος.

4.3 Σχετικό Σφάλμα και Μέγεθος Εισόδου

Αφού είδαμε τη χρονική συμπεριφορά των αλγορίθμων για συγκεκριμένα μεγέθη, ας δούμε κατά πόσο είναι εφικτό να χρησιμοποιηθούν σε εφαρμογές με απαιτήσεις υψηλής ακρίβειας αποτελεσμάτων. Γνωρίζουμε ότι οι αλγόριθμοι PCSA και Loglog Counting είναι πιθανοτικής φύσης και δίνονται οι αναμενόμενες τιμές τυπικού σφάλματος και πόλωσης στην πρώτη περίπτωση και τυπικού σφάλματος στη δεύτερη, ώστε να μας δείξουν ότι τα αποτελέσματά τους είναι παραπάνω από ικανοποιητικά. Όμως, είναι όντως ικανοποιητικά ή παρέχουν εκτιμήσεις περιορισμένης ακρίβειας που μπορούν να χρησιμοποιηθούν ως εργαλεία μιας πιο γενικής εκτίμησης της κατάστασης;

Στα σχήματα 4.8 και 4.12, βλέπουμε τον υπολογισμό του ποσοστού του σχετικού σφάλματος σε σχέση με το μέγεθος της ροής εισόδου για τους αλγορίθμους PCSA και Loglog Counting αντίστοιχα στην δικτυακή πύλη *Stargate Netbridge*. Παρατηρούμε ότι και για τους δύο αυτούς αλγορίθμους το μέγιστο ποσοστό σφάλματος δεν ξεπερνά το 0.06% και είναι ανεξάρτητο του μεγέθους της ροής εισόδου. Το σχετικό σφάλμα, ακόμα και για πολύ μεγάλα μεγέθη, παραμένει πολύ μικρό, ώστε η ακρίβεια που παρέχουν οι αλγόριθμοι να είναι παραπάνω από ικανοποιητική. Στο σχήμα 4.10, φαίνεται το ποσοστό σχετικού σφάλματος για τον αλγόριθμο TDS, ο οποίος είναι, από ότι δείχνουν τα αποτελέσματα, πιο ακριβής από τους άλλους δύο αλγορίθμους, το ποσοστό σφάλματος δεν ξεπερνά το 0.02%. Γεγονός αναμενόμενο λαμβάνοντας υπόψη δύο σημαντικά σημεία, τη θεωρία πίσω από τον αλγόριθμο, η οποία δίνει πολύ ακριβείς εκτιμήσεις όσο μικραίνει η προσεγγιστική παράμετρος, ϵ και την υλοποίηση κατά την οποία χρησιμοποιείται η $DummyHits$ στη συνάρτηση $RangeSample$, η οποία παίζει ίσως το σημαντικότερο ρόλο.

Επομένως, οι δύο πιθανοτικοί αλγόριθμοι προτιμούνται και πάλι, αφού ο TDS είναι αρκετά περιοριστικός στο θέμα του μεγέθους της ροής εισόδου. Αν αποφασίσουμε να χρησιμοποιήσουμε τον αλγόριθμο για εκτίμηση πλήθους διαφορετικών τιμών με μέγεθος ροής αρκετά μεγαλύτερο η πολυπλοκότητα χώρου αυξάνεται κατακόρυφα. Για παράδειγμα, υποθέστε ότι θέλουμε να μετρήσουμε τις διαφορετικές τιμές μιας ροής μεγέθους 600000 στοιχείων, αυτό σημαίνει ότι η προσεγγιστική παράμετρος, $\epsilon = 0.01$, και το μέγεθος του κάθε

δείγματος που διατηρεί ο αλγόριθμος θα είναι ίσο με $\tau = 60/0.01^2 = 600000$ στοιχεία, τιμή απαγορευτική για τέτοιου είδους συσκευές.

4.4 Εισάγοντας Διπλότυπα

Είδαμε τι γίνεται στην περίπτωση που δεν εισάγονται διπλότυπα στη ροή εισόδου. Στην υποενότητα αυτή θα εξετάσουμε κατά πόσο επηρεάζεται ο κάθε αλγόριθμος από το ποσοστό διπλότυπων της εισόδου, τόσο χρονικά όσο και σε ακρίβεια αποτελεσμάτων.

4.4.1 Χρονική Αξιολόγηση

Όσον αφορά τη χρονική αξιολόγηση με βάση το ποσοστό διπλότυπων βοηθούν αρκετά τα σχήματα 4.31, 4.32 και 4.33, στα οποία φαίνεται η χρονική συμπεριφορά του κάθε αλγορίθμου συναρτήσει του ποσοστού των διπλότυπων της ροής εισόδου στο *PC Engine ALix*. Οι γραφικές έγιναν για συγκεκριμένο μέγεθος ροής, το οποίο είναι $n = 6000$ στοιχεία. Παρατηρούμε ότι για τους πιθανοτικούς αλγορίθμους το ποσοστό των διπλότυπων δεν παίζει απολύτως κανένα ρόλο αφού η χρονική διαφορά που έχουν τα πειράματα μεταξύ 0% και 80% διπλότυπων στην είσοδο είναι πολύ μικρή, της τάξης του $1/1000$ του δευτερολέπτου. Από την άλλη, ο αλγόριθμος TDS λόγω της διαδικασίας που ακολουθεί, δηλαδή ελέγχοντας στην κύρια ρουτίνα του αν το στοιχείο έχει ήδη εξεταστεί ή όχι και αν ναι παρακάμπτει όλες τις άλλες πράξεις, αποδίδει πολύ γρηγορότερα όσο αυξάνεται το ποσοστό διπλότυπων στη ροή εισόδου. Πιο συγκεκριμένα, φαίνεται στο σχήμα 4.33, ότι ο χρόνος και το ποσοστό διπλότυπων της εισόδου είναι αντιστρόφως ανάλογα μεγέθη, αφού όσο περισσότερα στοιχεία αναφέρονται σε διπλότυπες παρατηρήσεις τόσο περισσότερες πράξεις αποφεύγονται.

4.4.2 Αξιολόγηση Ακρίβειας

Όπως φαίνεται από τα σχήματα 4.37, 4.38 και 4.39 για το *PC Engine Alix*, η ακρίβεια των τριών αλγορίθμων δεν επηρεάζεται από το ποσοστό διπλότυπων της ροής εισόδου. Το πο-

σοστό σχετικού σφάλματος είναι στα επίπεδα που ήταν όταν εξετάσαμε την ακρίβεια χωρίς διπλότυπα. Γεγονός το οποίο ήταν αναμενόμενο, αφού η ακρίβεια θεωρητικά δεν έχει σχέση με το ποσοστό διπλότυπων στη ροή εισόδου. Για τους δύο πιθανοτικούς αλγορίθμους, PCSA και Loglog Counting η ακρίβεια εξαρτάται από τη συνάρτηση κατακερματισμού που χρησιμοποιείται και την παράμετρο m, η οποία υποδηλώνει τον αριθμό των διαφορετικών πινάκων ψηφίων που υπολογίζονται από τους αλγορίθμους. Στην ϵ-approximation διαδικασία του TDS αλγορίθμου, η ακρίβεια εξαρτάται από την επιλογή της προσεγγιστικής παραμέτρου, ϵ, η οποία καθορίζει το μέγιστο μέγεθος του κάθε δείγματος και όχι από τη μορφή της εισόδου.

Κεφάλαιο 5

Συμπεράσματα και Μελλοντικές Κατευθύνσεις

Στην παρούσα διπλωματική παρουσιάστηκαν τρείς αλγόριθμοι σχετικοί με τον τομέα της *συγκέντρωσης/συνάθροισης δεδομένων* και την επίλυση του προβλήματος *μέτρησης διαφορετικών τιμών* μιας ροής δεδομένων. Ακολούθησε η περιγραφή των βημάτων της υλοποίησης των αλγορίθμων και η προετοιμασία του πηγαίου κώδικα με σκοπό την εκτέλεση πειραμάτων σε περιβάλλοντα περιορισμένων υπολογιστικών πόρων, τα οποία και παρουσιάζονται. Στη συνέχεια, δόθηκαν τα πειραματικά αποτελέσματα σε μορφή γραφικών παραστάσεων, συνοδευόμενα από τα κατάλληλα σχόλια και παρατηρήσεις.

5.1 Συμπεράσματα

Η πειραματική αξιολόγηση των αλγορίθμων σε σημαντικά περιορισμένα υπολογιστικά συστήματα, η οποία από όσο γνωρίζουμε πραγματοποιήθηκε για πρώτη φορά, μας οδηγεί σε χρήσιμα και πολύτιμα συμπεράσματα όσον αφορά την πρακτική εφαρμογή των τελευταίων σε τέτοιου είδους συσκευές. Δείξαμε ότι το υπάρχον εμπορικό υλικό διαθέτει ικανοποιητικά τεχνικά χαρακτηριστικά για να υποστηρίξει αποδοτικά την χρήση πιο γενικού σκοπού (PCSA, Loglog Counting), αλλά και πιο εξειδικευμένων (Time-decaying Sketches) αλγορίθμων. Τα

περιβάλλοντα στα οποία εκτελέστηκαν τα πειράματα κρίθηκαν κατάλληλα για να "αντέξουν" τους αλγορίθμους που εξετάσαμε χωρίς να χρειάζονται πολύ "προσεκτικές" κινήσεις στη διαχείριση της μνήμης, εκτός από αυτές που αναφέρονται στον εκάστοτε αλγόριθμο.

Επιπλέον, τα πειραματικά αποτελέσματα αποκαλύπτουν ότι η επιλογή του κατάλληλου αλγορίθμου δεν επίκειται στο υλικό που χρησιμοποιείται, αλλά στην εφαρμογή που αποσκοπούμε να υλοποιήσουμε. Η συμπεριφορά των αλγορίθμων είναι η ίδια σε κάθε συσκευή, το μόνο που αλλάζει είναι η τάξη των δεδομένων, η οποία εξαρτάται από τα χαρακτηριστικά της τελευταίας. Για παράδειγμα, θα προτιμούσαμε στην περίπτωση που θέλουμε να υπολογίσουμε πιο σύνθετα μεγέθη (μέσος όρος, ϕ-quantiles, ...) να χρησιμοποιήσουμε τον αλγόριθμο TDS, επειδή παρά την αυξημένη πολυπλοκότητα χώρου παρέχει πολύ ακριβείς αποκρίσεις σε πολυλογαριθμικό χρόνο.

Μια πολύ κρίσιμη υποσημείωση είναι ότι ενώ TDS παρουσιάζεται σαν ένας αλγόριθμος αιχμής για συγκέντρωση δεδομένων δικτύων αισθητήρων, η πρακτική του εφαρμογή είναι αδύνατη. Δεν γίνεται με την παρούσα διαδικασία που προτείνεται να παράγουμε τα θεωρητικά όρια για τις πολυπλοκότητες χώρου και χρόνου. Επιπροσθέτως, γίνεται ξεκάθαρο από την πειραματική διαδικασία ότι αν αυξήσουμε το μέγεθος της εισόδου και επαγωγικά το μέγεθος των δειγμάτων, ο χρόνος επεξεργασίας του αλγορίθμου θα είναι κάθε άλλο παρά ικανοποιητικός.

5.2 Μελλοντικές Κατευθύνσεις

Η δουλειά που έχει γίνει, αλλά και συνεχίζεται στο πεδίο της συνάθροισης δεδομένων μας βοηθά να ορίσουμε μονοπάτια για τη συνέχεια της διπλωματικής αυτής. Το επόμενο βήμα που έρχεται φυσικά είναι να θέσουμε ως είσοδο στους αλγορίθμους πραγματικά δεδομένα, τα οποία μπορούμε, όχι εύκολα, να συγκεντρώσουμε από δίκτυα αισθητήρων. Θα ήταν πολύ ενδιαφέρον να δούμε αν και πως θα επηρεαστεί η συμπεριφορά των αλγορίθμων με τροφοδότηση πραγματικής εισόδου, η οποία μπορεί να ακολουθεί κάποια συγκεκριμένη κατανομή ή να αποτελείται από σύνθετη κατανομή στοιχείων. Επομένο, βήμα θα μπορούσε να είναι η

αξιολόγηση των αλγορίθμων σε πιο εξεζητημένες συσκευές δικτύων αισθητήρων ώστε να αποφανθούμε αν η υλοποίησή τους είναι το ίδιο άνετη και εύκολη όσο στις συσκευές λίγο πιο υψηλού επιπέδου. Θεμελιώδες βήμα αποτελεί η ενσωμάτωση των αλγορίθμων σε εφαρμογές πραγματικού χρόνου ενός κινητού δικτύου ή δικτύου αισθητήρων, με σκοπό την ανάλυση της αποδοτικότητας της ολοκληρωμένης εφαρμογής. Τέλος, εφόσον μπορούμε να έχουμε στη διάθεσή μας πληθώρα πειραματικών αποτελεσμάτων θα μπορούσαμε να προτείνουμε έναν αλγόριθμο που συνδυάζει τα πλεονεκτήματα της κάθε προσέγγισης και θα προσαρμόζεται ανάλογα με την εφαρμογή.

Βιβλιογραφία

[1] P. Flajolet, G.N. Martin. *Probabilistic counting algorithms for data base applications*. Journal of Computer and System Sciences 31(2):182-209, 1985.

[2] M. Durand, P. Flajolet. *Loglog counting of large cardinalities*. In Proc. 16th Internation Symp. On Algorithms, pages 153-166, 2003.

[3] G. Cormode, S. Tirthapura, B. Xu. *Time-Decaying Sketches for Sensor Data Aggregation*. PODC '07.

[4] A. Pavan, S. Tirthapura. *Range-efficient computation of F_0 over massive data streams*. Journal on Computing, 37(2):359-379, 2007.

[5] Y.E Ioannidis. *The history of histograms(abridged)*. In Proc. VLDB, pages 19-30, 2003.

[6] C. Estan, G. Varghese, M. Fisk. *Bitmap algorithms for counting active flows on high speed links*. In Proc. SIGCOMM 02, pages 323-336, 2002.

[7] P.J. Haas, Y. Liu, L. Stokes. *An estimator of the number of species from quadrat sampling*. Biometrics, 62:135-141, 2006.

[8] Z. Bar-Yossef, T.S. Jayram, R. Kumar, D. Sivakumar, L. Trevisan. *Counting distinct elements in a data stream*. In Proc. RANDOM, pages 1-10, 2002.

[9] F. Giroire. *Order statistics and estimating cardinalities of massive datasets*. In Proc. Intl. Conf. Analysis Algorithms, pages 157-166, 2005.

[10] N. Alon, Y. Matias, M. Szegedy *The space complexity of approximating the frequency moments*. J. of Computer and System Science 58, pages 137-147, 1999.

[11] L. Hu, D. Evans *Localization for Mobile Sensor Networks*. In Tenth Annual International Conference on Mobile Computing and Networking (MobiCom 2004).

[12] R. Zheng, R. Barton *Towards Optimal Data Aggregation in Random Wireless Sensor Networks*. INFOCOM 2007

[13] B. Krishnamachari, D. Estrin, S. Wicker *The Impact of Data Aggregation in Wireless Sensor Networks*. Distributed Computing Systems Workshops, pages 575- 578, 2002

[14] A. L. L. de Aquino, C. M. S. Figueiredo, E. F. Nakamura, A. C. Frery, A. A. F. Loureiro, A. O. Fernandes *Sensor Stream Reduction For Clustered Wireless Sensor Networks*. SAC 2008

[15] A. Kamra, V. Misra, D. Rubenstein *CountTorrent: Ubiquitous Acess to Query Aggregates in Dynamic and Mobile Sensor Networks*. Sensys 2007

[16] K. Beyer, P. J. Hass, B. Reinwald, Y. Sismanis, R. Gemull *On Synopses for Distinct-Value Estimation Under Multiset Operations*. SIGMOD 2007

[17] A. L. L. Aquino, C. M. S. Figueiredo, E. F. Nakamura, L. S. Buriol, A. A. F. Loureiro, A. O. Fernandes, C. J. N. Coelho Jr. *Data Stream Based Algorithms For Wireless Sensor Network Applications*. AINA 2007

[18] Iternational Telecommunication Union *ITU Internet Reports: The Internet of Things Edition 2005*

[19] M. Weiser *Some Computer Science Issues In Ubiquitous Computing*. CACM 1993

[20] Επίσημη ιστιοσελίδα της υπηρεσίας ευρωπαϊκών στατιστικών, *Eurostat*, *http://epp.eurostat.ec.europa.eu*

[21] Επίσημη ιστιοσελίδα της Διεθνούς Ομοσπονδίας Φωνογραφικής Βιομηχανίας, *IFPI*. *http://www.ifpi.com/*