ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ
ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΚΙΝΗΤΟΣ ΚΑΙ ΔΙΑΧΥΤΟΣ ΥΠΟΛΟΓΙΣΜΟΣ
ΧΕΙΜΕΡΙΝΟ ΕΞΑΜΗΝΟ

PROJECT

ΚΑΘΗΓΗΤΗΣ
Κατσαρός Δημήτριος

ΦΟΙΤΗΤΗΣ
Κουκουγιάννης Δημήτριος
ΑΕΜ 2537
dkoukougiannis@uth.gr

Βόλος, Ιανουάριος 2021

Περιεχόμενα

1 Εισαγωγή **2**

2 Το πρόβλημα **2**

3 Αλγόριθμος **2**
3.1 Heuristic 1 3

4 Ανάπτυξη κώδικα **3**
4.1 Δημιουργία συνόλων ερωτημάτων 3
4.2 Εφαρμογή ευρετικής 3
4.3 Το κυρίως πρόγραμμα 4

5 Γραφικές παραστάσεις **4**
5.1 Συζήτηση 4
5.1.1 Αιτία 8
5.1.2 Ανάλυση χειρότερης περίπτωσης 8

6 Συμπεράσματα **9**

Περίληψη

Στις μέρες μας τα ασύρματα περιβάλλοντα συναντώνται σε πλήθος εφαρμογών και έχουν γίνει αναπόσπαστο κομμάτι της καθημερινής ζωής. Ο καθένας μπορεί να χρησιμοποιήσει μία συσκευή για να λάβει κάποια στοιχεία. Σε αυτήν την εργασία μου ανατέθηκε το αρθρο που βρίσκεται στις αναφορές ώστε να γίνει η ανάπτυξη του κώδικα για την ευρετική 1.

1 Εισαγωγή

Ο συγγραφέας προσπαθεί να βρει λύση στο εξής πρόβλημα: Αν υπάρχει ένας server, ο οποίος εκπέμπει αντικείμενα σε πολλαπλά κανάλια κυκλικά, πως θα πρέπει να γίνει η εκπομπή τους; Ο εξυπηρετητής έχει μία μνήμη στην οποία καταγράφει τα αιτήματα, ενώ οι πελάτες ακούνε χρησιμοποιώντας μία συσκευή. Τονίζεται ότι μέχρι τώρα αντίστοιχα άρθρα προσπαθούσαν να αναπτύξουν αλγορίθμους, ώστε ο server να εκπέμπει τα πιο δημοφιλή στοιχεία έναντι των λιγότερων δημοφιλών με την ελπίδα ότι θα μειωθεί ο μέσος χρόνος αναμονής για τα στοιχεία. Το πρόβλημα με αυτήν την προσέγγιση είναι ότι μερικοί χρήστες μπορεί να μη λάμβαναν καθόλου κάποια στοιχεία, τα οποία ζήτησαν. Ακόμη σημειώνεται ότι ο μέσος χρόνος αναμονής δεν είναι η μοναδική μετρική, η οποία πρέπει να λαμβάνεται υπόψιν κατά τη σχεδίαση, καθώς ένα σύστημα πρέπει να είναι δίκαιο προς όλους τους χρήστες του.

2 Το πρόβλημα

Πριν την ανάπτυξη του κώδικα θα πρέπει να γίνει η αναφορά στους όρους που χρησιμοποιούνται. Έστω $\mathbf{AP}(\mathrm{m}) = \{Q_1, Q_2, ..., Q_m\}$ το πρότυπο προσπέλασης με Q_k να είναι ένα σύνολο από ερωτήματα, το οποίο περιλαμβάνει στοιχεία d_k ίσου μεγέθους. Για δεδομένο σύνολο S ορίζουμε ώς |S| τον αριθμό των στοιχείων που περιλαμβάνει. Επίσης ορίζουμε ως $\pi = \{P_1, P_2, ...\}$ τη διαμέριση του προτύπου προσπέλασης με τέτοιο τρόπο, ώστε τα P να μην έχουν τομή και η ένωση τους να μας δίνει το πρότυπο προσπέλασης. Το $\epsilon(\pi)$ είναι η συνάρτηση του χειρότερου χρόνου προσπέλασης. Στο 1 διακρίνονται συνοπτικά οι όροι. Για τη μελέτη του πρβλήματος γίνονται οι παρακάτω παραδοχές:

- Το πρότυπο προσπέλασης μπορεί να οριστεί εκ των προτέρων.
- Ο χρήστης κατεβάζει τα στοιχεία που θέλει από ένα μόνο κανάλι.
- Η βασική μονάδα εκπομπής είναι ένα σύνολο και όχι ατομικά κάθε στοιχείο.
- Το κανάλι εκπομπής C χωρίζεται σε χρονικά διαστήματα, κατά τα οποία το κόστος είναι η εκπομπή ενός αντικειμένου.
- Ισχύει για τη συνάρτηση χειρότερου κόστους ότι $|P_1| \geq |P_2| \geq |P_3| \geq ...$, οπότε $\epsilon(\pi) = |P_1|$.
- Το κανάλι έχει index.

Το πρόβλημα λοιπόν είναι το εξής όπως αυτό περιεγράφηκε: "*Given C and AP(m), the broadcast scheduling problem is to find the optimal partition* π^* *of AP(m) such that* π^* *minimizes* $\epsilon(\pi)$*, where each* $P_i \in \epsilon(\pi)$ *will be organized into a broadcast program and broadcast cyclically on channel i within* $\epsilon(\pi^*)$ *buckets.*"

3 Αλγόριθμος

Ο αλγόριθμος ο οποίος χρησιμοποιήθηκε από το συγγραφέα έχει 3 ευρετικές. Εγώ ασχολήθηκα με την πρώτη. Πριν όμως ξεκινήσει η ανάλυση του αλγορίθμου πρέπει να δοθούν κάποιοι ορισμοί. Έστω δύο σύνολα Q_1, Q_2. Αυτά ονομάζονται επικαλυπτόμενα (overlapped), εαν έχουν τουλάχιστον ένα κοινό στοιχείο δηλαδή αν η τομή τους δεν είναι το κενό σύνολο. Για δύο τυχαία αντικείμενα d_i, d_j ορίζουμε ως μονοπάτι μεταξύ τους μία σειρά από μη επικαλυπτόμενα σύνολα τα οποία είναι επικαλυπτόμενα. Ένα σύνολο Ω από τέτοια σύνολα ονομάζεται συνδεδεμένο αν υπάρχει μονοπάτι που να συνδέει δύο οποιαδήποτε στοιχεία.

Symbol	Description
d_k	Data item
N	Number of data items
C	Number of channels
Q_j	Query set
m	Number of query sets
$\boldsymbol{AP}(m)$	Access pattern, $\boldsymbol{AP}(m) = \{Q_1, Q_2, \ldots, Q_m\}$
$\|S\|$	Number of data items in a set S
π	Partition of $\boldsymbol{AP}(m)$
P_i	ith part of partition $\pi = \{P_1, P_2, \ldots, P_C\}$
$\varepsilon(\pi)$	Access time function, $Min(Max\{\|P_1\|, \|P_2\|, \ldots, \|P_C\|\})$
π^*	Optimal solution to $\varepsilon(\pi)$
$\varepsilon(\pi^*)$ or ε^*	Minimal access time

Σχήμα 1: Πίνακας σημασίας συμβόλων

3.1 Heuristic 1

Σε αυτή την ευρετική επικαλυπτόμενα σύνολα πρέπει να είναι αποθηκευμένα μαζί. Έτσι προσπαθούμε να δημιουργήσουμε επικαλυπτόμενα σύνολα για κάθε σύνολο ερωτημάτων. Σκοπός της είναι να μειώσει το χειρότερο χρόνο εκπομπής, ο οποίος είναι ίσος με το μέγεθος του μεγαλύτερου συνδεδεμένου στοιχείου που θα δημιουργηθεί.

4 Ανάπτυξη κώδικα

Για την προσομοίωση της λειτουργίας ενός server σε κάποιο τυχαίο ασύρματο περιβάλλον χρησιμοποιήθηκε η γλώσσα προγραμματισμού Java. Ακολουθήθηκαν τρία στάδια τα οποία αφορούσαν τη δημιουργία των συνόλων ερωτημάτων, την εφαρμογή της ευρετικής για τη δημιουργία προτύπου εκπομπής και τη μέτρηση του χειρότερου χρόνου εκτέλεσης. Στο τέλος έγιναν οι γραφικές παραστάσεις με τη χρήση της Python για οπτικοποίηση των αποτελεσμάτων.

4.1 Δημιουργία συνόλων ερωτημάτων

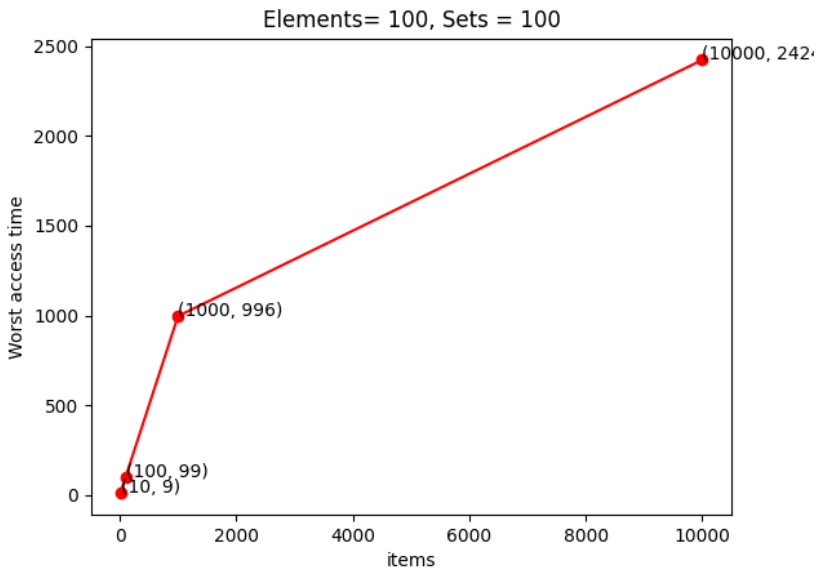
Για να γίνει η προσομείωση του ασύρματου περιβάλλοντος δημιουργήθηκε η κλάση Create_AP. Αυτή λαμβάνει τρία ορίσματα: τον αριθμό των στοιχείων, των αριθμό των συνόλων και τον μέγιστο αριθμό στοιχείων σε κάθε σύνολο. Για να δημιουργηθεί ένα πρότυπο διατάσσουμε τα στοιχεία με αύξουσα σειρά του id τους ξεκινώντας από το ένα. Στη συνέχεια αναθέτω μία τιμή για κάθε στοιχείο ίση με $\frac{\sqrt{id}}{N}$, οπου N το μέγιστο πλήθος στοιχείων. Ο παρακάτω πίνακας είναι ενδεικτικός της μέχρι τώρα διαδικασίας.

1	2	3	...	n
$\sqrt{1}/N$	$\sqrt{2}/N$	$\sqrt{3}/N$	...	$\sqrt{n}/N$

Μετά διαλέγω μία τυχαία τιμή μεταξύ 0 και της μέγιστης που βρέθηκε προηγουμένως και ανάλογα το διάστημα στο οποίο ανήκει η συγκεκριμένη τιμή διαλέγω εκείνο το σημείο. Τα άκρα κάθε διαστήματος ορίζονται ως η πιθανότητα του πίνακα συν την προηγούμενη τιμή ενω συμβατικά ισχύει ότι το διάστημα ξεκινάει στο 0. Αν ένα σύνολο έχει δύο φορές την ίδια τιμή τότε θεωρώ ότι ισχύει μόνο η μία τιμή (αν και δεν τη διαγράφω).

4.2 Εφαρμογή ευρετικής

Στη συνέχεια δημιουργήθηκε η κλάση NewPattern μέσω της οποίας θα δημιουργηθεί το νέο πρότυπο προσπέλασης. Ο κατασκευαστής αυτής της κλάσης κάνει μία επανάληψη στην οποία αποθηκεύει στη μεταβλητή pendingQueries τα ερωτήματα τα οποία δημιουργούν ένα μονοπάτι. Στη συνέχεια και όταν έχει τελειώσει κάθε επανάληψη αφαιρούντα τα συγκεκριμένα σύνολα από την αρχική λίστα και η διαδικασία συνεχίζεται μέχρι να αδειάσει αυτή. Μέσω της δημιουργίας αυτού του νέου αντικειμένου θα δημιουργήσουμε τα ζητούμενα υπερσύνολα και εν συνεχεία τον χειρότερο χρόνο εκτέλεσης. Αυτός θα βρεθεί μέσω της μεθόδου της κλάσης, η οποία βρίσκει ποιο υπερσύνολο έχει τα περισσότερα στοιχεία. Ο αριθμός που επιστρέφεται είναι ο χειρότερος χρόνος εκτέλεσης σε buckets.



4.3 Το κυρίως πρόγραμμα

Το κυρίως πρόγραμμα εκτελεί διάφορες επαναλήψεις για να σχεδιαστούν στη συνέχεια οι γραφικές τους παραστάσεις. Για να το κάνει αυτό χρησιμοποιεί τρεις πίνακες τους οποίους προσπελαύνει διαδοχικά. Για να είναι δυνατή η ανακατασκευή των αποτελεσμάτων σε κάθε εκτέλεση δημιουργούνται κατάλληλα αρχεία στα οποία αποθηκεύονται τα προγράμματα εκπομπής που δημιουργήθηκαν αρχικά. Επίσης στον παρακάτω πίνακα διακρίνονται οι διάφορες τιμές για τις παραμέτρους που χρειάζονται. Στο τέλος εκτυπώνω στην οθόνη τον κώδικα Python που θα χρειαστεί να εκτελεστεί για να γίνουν οι γραφικές παραστάσεις.

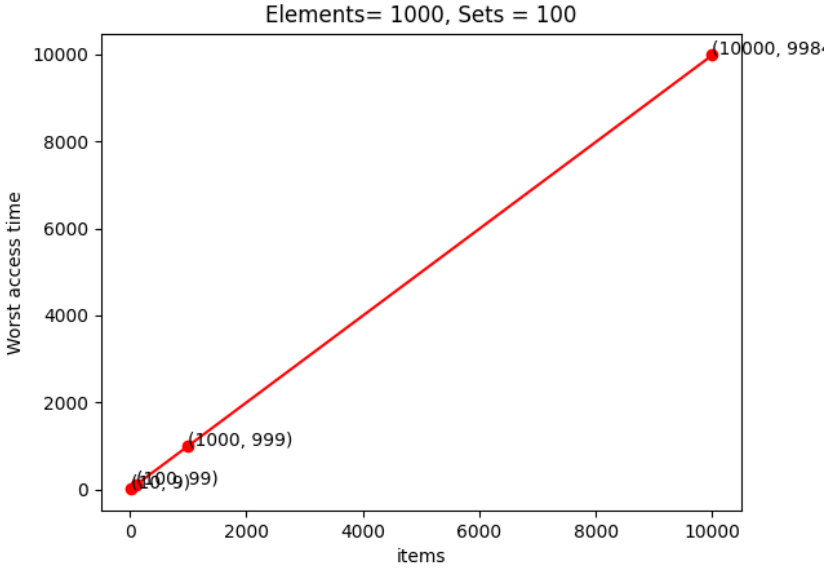
5 Γραφικές παραστάσεις

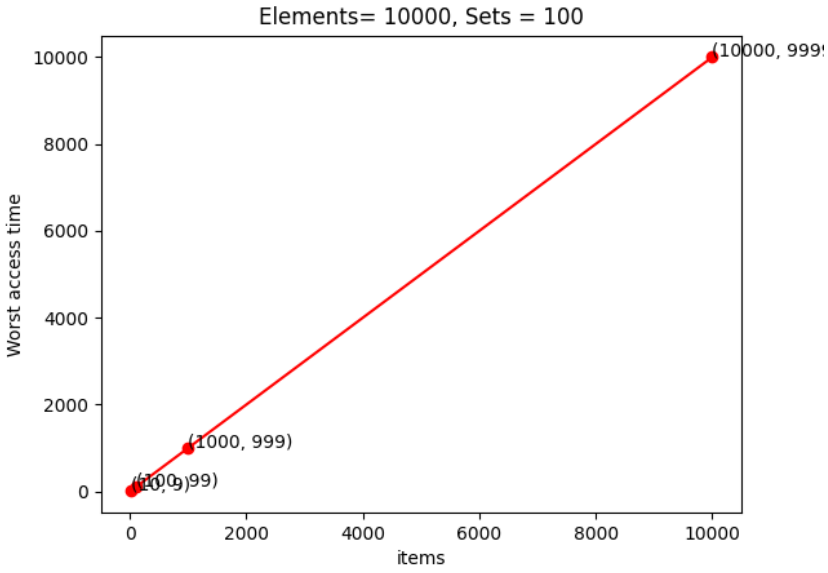
Σε αυτήν την ενότητα θα δοθούν οι γραφικές παραστάσεις για την εφαρμογή της ευρετικής. Οι μεταβλητές που παρουσιάζονται είναι οι παρακάτω:

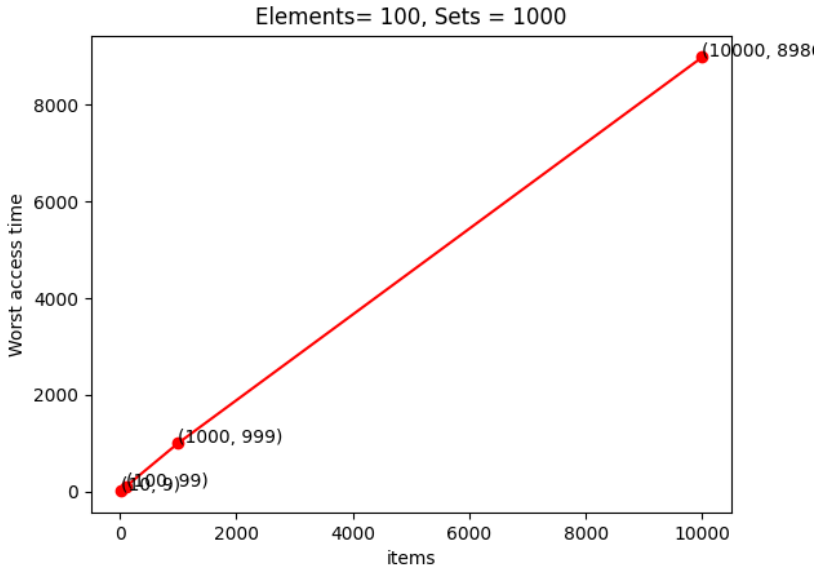
- Worst access time: Ο χειρότερος χρόνος προσπέλασης αντικειμένου (είναι ίσος με το μέγεθος του μεγαλύτερου συνόλου). Μετριέται σε buckets, οπου κάθε στοιχείο καταλαμβάνει ένα.
- Items: Ο μέγιστος αριθμός των μοναδικών στοιχείων που μπορεί να βρίσκονται σε ένα σύνολο.
- Sets: Ο αριθμός των συνόλων των ερωτημάτων.

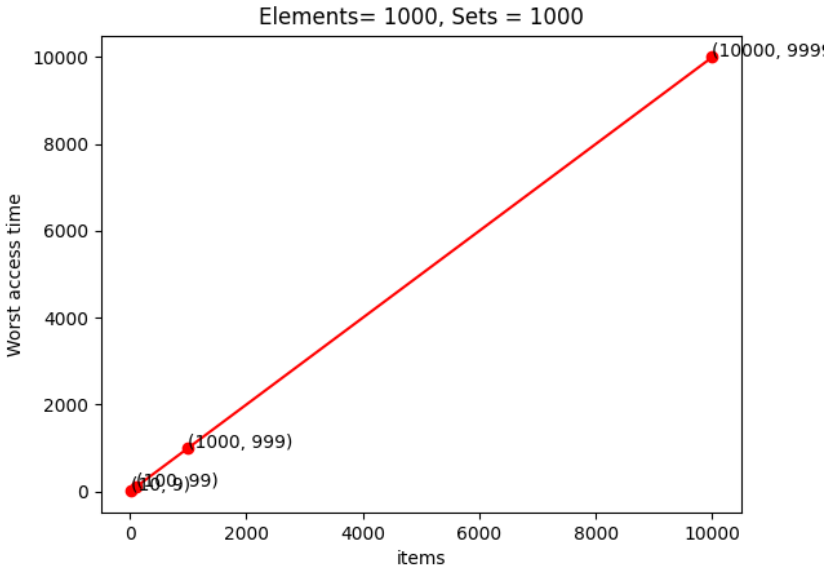
5.1 Συζήτηση

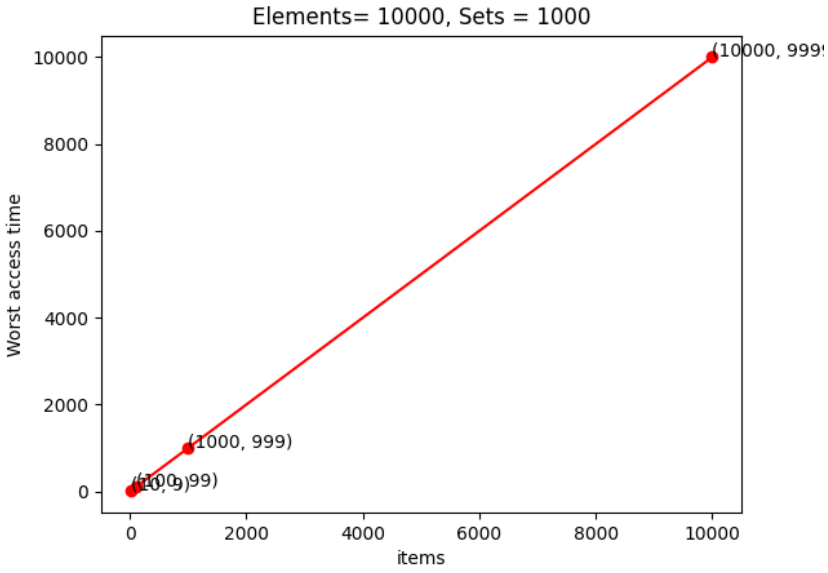
Η ευρετική 1 παρουσιάζει καλύτερα αποτελέσματα όταν υπάρχει μικρός αριθμός συνόλων και μοναδικών στοιχείων. Όταν αυξάνεται πολύ ο αριθμός των μέγιστων στοιχείων σε ένα σύνολο η συμπεριφορά είναι πολύ κακή και αυξάνεται δραματικά για πολλά μοναδικά στοιχεία, ενώ όταν τα μοναδικά στοιχεία είναι λίγα ο μέγιστος χρόνος προσπέλασης είναι ο ίδιος. Τέλος για μικρό αριθμό μοναδικών στοιχείων αλλά μεγάλο αριθμό συνόλων είναι μεγαλύτερη σε σχέση με πριν. Συνοψίζοντας λοιπόν γίνεται αντιληπτό ότι όσο πιο λίγα μοναδικά στοιχεία υπάρχουν και όσο λιγότερα σύνολα δεδομένων η συμπεριφορά είναι η βέλτιστη. Αν αυξηθούν μόνο τα σύνολα η συμπεριφορά είναι η ίδια με πριν για μικρό αριθμό μέγιστων αντικειμένων σε ένα σύνολο, αλλά γίνεται χειρότερη αν αυξηθούν πολύ τα αντικείμενα. Τέλος αν αυξηθούν και τα elements, sets η συμπεριφορά είναι πολύ κακή και παραμένει η ίδια για κάθε προσομείωση.

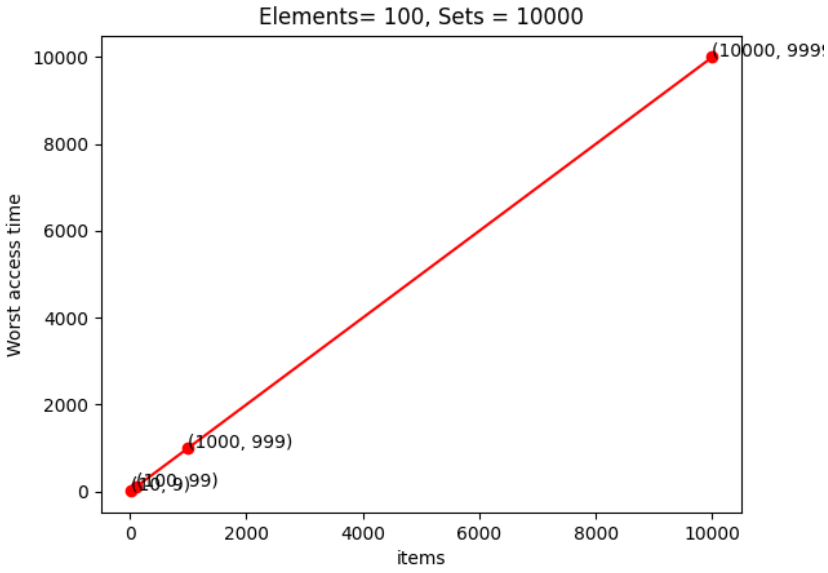

Elements= 1000, Sets = 100
Worst access time
items
(1000, 999)
0
2000
4000
6000
8000
10000

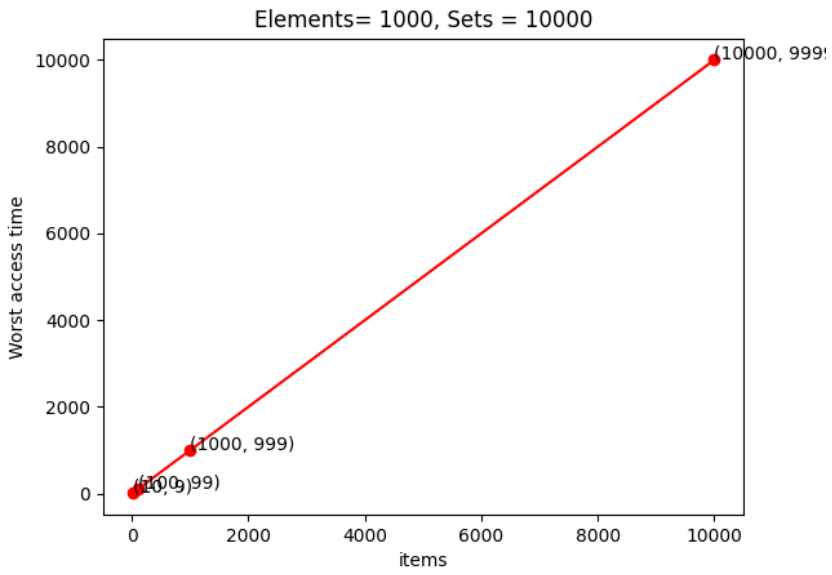

Elements= 10000, Sets = 100
Worst access time
items
(1000, 999)
0
2000
4000
6000
8000
10000


Elements= 100, Sets = 1000
Worst access time
items
0
2000
4000
6000
8000
10000
(1000, 999)


Elements= 1000, Sets = 1000
Worst access time
items
0
2000
4000
6000
8000
10000
(1000, 999)


Elements= 10000, Sets = 1000
Worst access time
items
(1000, 999)
0
2000
4000
6000
8000
10000


Elements= 100, Sets = 10000
Worst access time
items
(1000, 999)
0
2000
4000
6000
8000
10000



5.1.1 Αιτία

Elements

Όταν τα στοιχεία σε κάθε σετ είναι λίγα μειώνεται το απόλυτο μέγεθος του ίδιου του συνόλου αλλά και η πιθανότητα να γίνει ένωση μεταξύ sets. Ακόμα και στην περίπτωση που αυτό συμβαίνει το προκύπτων σύνολο αποτελείται από μικρότερα σύνολο οπότε και μειώνεται και ο χρόνος. Αν ο αριθμός των στοιχείων του συνόλου αυξηθεί σημαντικά το υπερσύνολο θα αποτελείται από μεγαλύτερα υποσύνολα.

Sets

Λιγότερα σύνολα αυξάνουν την πιθανότητα ένωσης τους, οπότε και μειώνεται ο χειρότερος χρόνος.

Items

Όταν υπάρχει μικρότερος αριθμός από Items τότε η πιθανότητα ενα μοναδικό στοιχείο να βρίσκεται περισσότερες φορές μέσα σε ένα σύνολο αυξάνεται (σε αυτήν την περίπτωση 'μετράει' μόνο το ένα bucket). Έπίσης αυξάνεται και η πιθανότητα μεγάλυς επικάλυψης μεταξύ συνόλων γεγονός που μειώνει το συνολικό τους μέγεθος. Άρα το υπερσύνολο αποτελείται από μικρότερα σύνολα και έτσι μειώνονται τα buckets από τα οποία αποτελείται.

5.1.2 Ανάλυση χειρότερης περίπτωσης

Η χειρότερη περίπτωση είναι να υπάρχει ακριβώς ένα επικαλυπτόμενο στοιχείο μεταξύ διαδοχικών συνόλων και όλα τα άλλα ους στοιχεία να είναι διαφορετικά. Έστω λοιπόν N ο αριθμός των sets, M των items και K των elements. Για τη συγκεκριμένη περίπτωση αν $worst\_time$ είναι ο μέγιστος χρόνος προσπέλασης τότε:

$$worst\_time = K - 1 + K - 1 + K - 1 + ... + K - 1 = N \cdot (K - 1)$$

Από τα παραπάνω μπορεί να γίνει γρήγορα αντιληπτό ότι ο αλγόριθμος προσεγγίζει αυτή την κατάσταση όταν τα μοναδικά αντικείμενα είναι πολλά οπότε και η πιθανότητα συνύπαρξης και των δύο σε ένα σύνολο είναι μικρή. Έτσι αν αυξηθούν και τα σύνολα αυξάνεται και το συνολικό N οπότε και αυξάνεται και ο παρακάτω χρόνος. Αν και τα μέγιστα στοιχεία σε ένα σύνολο είναι πολλά η μεταβλητή $worst\_time$ γίνεται πολύ μεγάλη.

6 Συμπεράσματα

Η πρώτη ευρετική του άρθρου είναι μία πολύ καλή λύση για εφαρμογή σε προβλήματα όπου υπάρχει μικρός αριθμός αιτημάτων (sets) και μοναδικών αντικειμένων. Αν τα items είναι πολλά καλό θα ήταν να χωρίστούν τα ερωτήματα με τέτοιον τρόπο, ώστε τα σύνολα ερωτημάτων να είναι λίγα και να αποτελούνται από μικρό αριθμό στοιχείων. Μία βελτίωση που θα πρότεινα είναι να γίνεται ένωση συνόλων, τα οποία έχουν όσο το δυνατόν μεγαλύτερη επικάλυψη μετακύ τους.

Αναφορές

[1] Set-based broadcast scheduling for minimizing the worst access time of multiple data items in wireless environments, Jen-Ya Wang