Μιχάλης Αναστασίου
2017030185

Τεκμηρίωση Αποτελεσμάτων 1ης Εργαστηριακής Άσκησης

ΜΕΘΟΔΟΣ	ΚΑΤΑΣΚΕΥΗ ΑΡΧΕΙΟΥ	ΤΑΞΙΝΟΜΗΣΗ ΑΡΧΕΙΟΥ	ΣΕΙΡΙΑΚΗ ΑΝΑΖΗΤΗΣΗ(ΕΠΙΤΥΧΗΜΕΝΗ)	ΣΕΙΡΙΑΚΗ ΑΝΑΖΗΤΗΣΗ(ΑΠΟΤΥΧΗΜΕΝΗ)	Δυαδική Αναζήτηση (επιτυχημένη)	Δυαδική Αναζήτηση (αποτυχημένη)
Απόδοση	100	200	35-50	100	5-7	6-10

```
The buffers accesed the disk : 100 times for the 1st subquery
The buffers accesed the disk : 200 times to make the 10 files


LINEAR SEARCH RESULTS:

NUMBERS FOUND                          : 20
NUMBERS NOT FOUND                      : 20
Average of accesses at disk for FOUND     : 45
Average of accesses at disk for NOT FOUND : 100


BINARY SEARCH RESULTS :

NUMBERS FOUND                          :20
NUMBERS NOT FOUND                      :20
Average of accesses at disk for FOUND     :5
Average of accesses at disk for NOT FOUND :9
```

Κατασκευή Αρχείου :

Στην κατασκευή αρχείου έχουμε 100 προσβάσεις στον δίσκο όπως ήταν φυσικά αναμενόμενο αφού χρησιμοποιεί buffer μεγέθους 1000 int για την δημιουργία του αρχείου και πρέπει να καταχωρήσουμε 100.000 ακέραιους έτσι κάνουμε 100 προσβάσεις για αποθήκευση.

Ταξινόμηση αρχείου :

Χρησιμοποιούμε , επίσης , για την ταξινόμηση του αρχείου buffer 1000 ακεραίων όμως αυτήν την φορά για την ανάγνωση των ακεραίων έπειτα την ταξινόμηση τους και τέλος την αποθήκευση τους . Οπόταν έχουμε τις διπλάσιες προσβάσεις στον δίσκο από την κατασκευή του αρχείου όπως ήτανε απολύτως αναμενόμενο.

Σειριακή αναζήτηση (ΕΠΙΤΥΧΗΜΕΝΗ):

Σε αυτή την μέθοδο κάνουμε ανάγνωση από τον δίσκο χρησιμοποιώντας ξανά buffer 1000 ακεραίων. Ο μέσος όρος εύρεσης 20 τυχαίων αριθμών με επιτυχία είναι 35-50, εξαιτίας του γεγονότος ότι διαβάζει από την αρχή του αρχείου μέχρι να βρεθεί ο τυχαίος αριθμός. Αύτη η μέθοδος έχει πολυπλοκότητα O(N) ,αφού αν λάβει ένα μεγάλο αριθμό που είναι προς το τέλος της λίστας θα διαβάζει όλα τα στοιχεία του αρχείου μέχρι να το βρει.

Σειριακή αναζήτηση (ΑΠΟΤΥΧΗΜΕΝΗ):

Για την ίδια μέθοδο αλλά αυτήν την φορά με αποτυχημένη εύρεση, φαίνεται να διαβάζει εξαντλητικά όλο το αρχείο μέχρι το τέλος για να βεβαιωθεί ότι δεν βρήκε το στοιχείο που του δώσαμε για εύρεση. Σε αυτήν την περίπτωσή είναι αναμενόμενο να κάνει ΠΑΝΤΑ 100 φορές πρόσβαση στην μνήμη αφού έχουμε "worst case scenario" του αλγόριθμού μας (O(N)) δηλαδή διαβάζει μέχρι να φτάσει στο τέλος του αρχείου.

Δυαδική αναζήτηση (ΕΠΙΤΥΧΗΜΕΝΗ) :

Αντίθετα με την σειριακή αναζήτηση σε αυτόν τον αλγόριθμο χρησιμοποιούμε δυαδική αναζήτηση. Σε αυτήν την περίπτωση ο αλγόριθμός μας φαίνεται να έχει μέσο όρο προσβάσεων στον δίσκο 5-7 .Αυτό ήταν προσδοκώμενο λόγω του ότι δεν διαβάζει εξαντλητικά το αρχείο όπως η άλλη μέθοδος αλλά λειτουργεί πιο «έξυπνα» . Αναλόγως του αριθμού που εισήγαμε , αφού πάει στο κέντρο του αρχείου ,αποφασίζει αν θα προχωρήσει δεξιά ή αριστερά , μετά κάνει το ίδιο μέχρι την εύρεση του ακέραιου.

Δυαδική αναζήτηση (ΑΠΟΤΥΧΗΜΕΝΗ) :

Αντίστοιχα σε αύτη την μέθοδο παρουσιάζεται ραγδαία βελτίωση σε σύγκριση με την σειριακή αναζήτηση αφού , όπως προανέφερα πιο πάνω , λειτουργεί πολύ πιο αποτελεσματικά σαν μέθοδος.

Σύγκριση Δυαδικής και Σειριακής Αναζήτησης :

Συμπερασματικά, παρατηρούμε ότι η δυαδική αναζήτηση είναι έως και 10 φορές πιο αποτελεσματική από την σειριακή , καθώς οι προσβάσεις στον δίσκο είναι έως και 10 φορές λιγότερες. Η πολυπλοκότητα της σειριακής αναζήτησης είναι O(N) ενώ της δυαδικής αναζήτησης είναι O(logn) και στο "best case scenario" είναι O(1) . Αυτό σαφώς μας αποδεικνύει την διαφορά που έχουν οι 2 αλγόριθμοι και μας δικαιολογεί πλήρως τον μέσο όρο που παρουσίασαν στα αποτελέσματα του προγράμματός μας.

ΑΝΑΦΟΡΑ ΠΑΡΑΔΟΤΕΟΥ :

1) Στο πρώτο μέρος για την κατασκευή του αρχείου χρησιμοποιώντας την μέθοδο που μας παρείχατε με την χρήση 2 for loop γέμισα αρχικά τον buffer με 1000 τυχαίους αριθμούς χρησιμοποιώντας την μέθοδο rand. Έπειτα , τους αποθήκευσα στο αρχείο 100 φόρες αυξάνοντας κάθε φορά το offset.
2) Για το 2^{ο} μέρος διάβασα αρχικά με την χρήση «for loop» τους 1000 αριθμούς μου με το κατάλληλο offset 10 φορές για να γεμίσω έναν array με 10.000 αριθμούς .Έπειτα , με την χρήση quicksort ταξινόμησα τους 10000 ακεραίους, τους αποθήκευσα στον δίσκο ξαναχρησιμοποιώντας τον buffer 1000 ακεραίων. Το κάνω αυτό 10 φόρες μέχρι να δημιουργηθούν και τα 10 αρχεία στον δίσκο.

3) (2β)Σε αυτό το κομμάτι της άσκησης με την χρήση 11 buffers διάβασα αρχικά 1000 ακέραιους από το κάθε αρχείο. Επιπρόσθετα , σε ένα array 10 ακεραίων αποθήκευα τον πρώτο ακέραιο από κάθε αρχείο και εντόπισα τον μικρότερο και την θέση του στον πίνακα. Έπειτα , βάζω τον ακέραιο αυτό στον 11^{ο} buffer και , ανάλογα με την θέση που βρισκόταν , διαβάζω τον επόμενο ακέραιο από το συγκεκριμένο αρχείο. Σε ένα while loop αυτό συνεχίζεται μέχρι να διαβάσω όλους τους ακεραίους. Όταν ένα αρχείο αδειάσει σταματάω να διαβάζω(εντοπισμός με counters σε ένα άλλο array που αποθηκεύω την θέση κάθε αρχείου για να ξέρω ότι το εξάντλησα πλήρως). Δεν μου ήταν αναγκαίο να διαγράψω τα αρχεία έτσι δεν τα διέγραψα.

4) Σε αυτή την μέθοδο χρησιμοποιώντας while , for , γεννήτρια τυχαίων αριθμών εξετάζω 1000 τυχαίους αριθμούς για την εύρεση τους μέσα στο ταξινομημένο αρχείο μέχρι να έχω 20 επιτυχημένες και αποτυχημένες προσπάθειες. Εάν έχω περισσότερες από 20 επιτυχημένες ή αποτυχημένες προσπάθειες πριν να ολοκληρωθεί σταματάω να αθροίζω τον μέσο όρο προσβάσεων τους στον δίσκο. Μόλις συμπληρωθούν και τα 40 αποτελέσματα βγαίνω από το while και υπολογίζω τον μέσο όρο προσβάσεων στον δίσκο .

5) Τέλος , για αυτό το κομμάτι κώδικα χρησιμοποιώ την μέθοδο binary search που βρήκα στο διαδίκτυο και την τροποποίησα κατάλληλα στα μέτρα του προβλήματός μας . Βάζω σαν όρισμα στην μέθοδο binary search τον αριθμό που πρέπει να βρει στο αρχείο ακολουθώντας τα βήματα της εκφώνησης. Όταν τελειώσει μου επιστρέφει ένα πίνακα με 3 ακεραίους. Στην πρώτη θέση του πίνακα επιστρέφει «2» αν είναι επιτυχημένη και «-1» αν είναι αποτυχημένη . Στην δεύτερη θέση επιστρέφει 0 αν είναι αποτυχημένη , ενώ αν είναι επιτυχημένη τις προσβάσεις που έκανε στον δίσκο , αντίθετα στην 3 θέση αν είναι αποτυχημένη τις προσβάσεις που έκανε στον δίσκο ενώ αν είναι επιτυχημένη 0. Εν κατακλείδι , με την ίδια διαδικασία όπως στην σειριακή αναζήτηση λειτουργεί μέχρι να βρει 20 επιτυχημένες και 20 αποτυχημένες προσπάθειες , έπειτα υπολογίζει και τυπώνει τον μέσο όρο προσβάσεων στον δίσκο.

Το πρόγραμμα μου φαίνεται να λειτουργά σε όλες τις περιπτώσεις και δεν κάνει κάτι περισσότερο από αυτά που μας ζητάει η άσκηση.

Πηγές πληροφόρησης:

```
//Quick sort from : https://www.geeksforgeeks.org/quick-sort/
//Binary search : https://www.geeksforgeeks.org/binary-search/
```