Οργάνωση αρχείων

Τεκμηρίωση των αποτελεσμάτων

Πίνακας αποτελεσμάτων

Μέθοδος	Α. τρόπος Οργάνωσης αρχείου	Β. τρόπος οργάνωσης αρχείου	Γ. τρόπος οργάνωσης αρχείου	Δ. τρόπος οργάνωσης αρχείου	Απόδοση Εξωτερικής Ταξινόμησης περίπτωση Γ	Απόδοση Εξωτερικής Ταξινόμησης περίπτωση Δ
Απόδοση (αριθμός προσβάσεων δίσκου)	1328	332	10	7	5000	1250

Ο Α. τρόπος οργάνωσης αρχείου είναι ο πιο αργός, διότι για την εύρεση του κλειδιού πρέπει να αναζητήσουμε σειριακά πάρα πολλές σελίδες, συνεπώς θα έχουμε μεγάλο αριθμό προσβάσεων στο δίσκο.

Ο Β. τρόπος οργάνωσης αρχείου παρατηρείται ότι είναι πιο γρήγορος από τον Α ,καθώς η αναζήτηση του κλειδιού γίνεται σε ένα αρκετά μικρότερο αρχείο που περιέχει μόνο το κλειδί και την σελίδα που βρίσκεται το κλειδί στο μεγάλο αρχείο. Αυτό μας διευκολύνει διότι κάνουμε μόνο 4 αναζητήσεις, στη σελίδα που μας δόθηκε από το μικρό αρχείο, στο μεγάλο αρχείο.

Ο Γ. Τρόπος οργανώνει το αρχείο με τον ίδιο τρόπο με τον Α, με μόνη διαφορά ότι η εύρεση του κλειδιού γίνεται με δυαδικό τρόπο με αποτέλεσμα την σημαντική μείωση του αριθμού προσβάσεων.

Ο πιο γρήγορος τρόπους απ' όλους παρατηρήθηκε ότι είναι ο Δ. τρόπος που οργανώνει το αρχείο σύμφωνα με τον Β τρόπο οργάνωσης, προβαίνοντας σε ακόμα λιγότερες προσβάσεις στο δισκό κατά την δυαδική αναζήτηση του κλειδιού.

Ο τρόπος ταξινόμησης που χρησιμοποιήθηκε για το Γ και το Δ αρχείο είναι ουσιαστικά ο ίδιος. Το αρχικό αρχείο διαβάστηκε σε ένα array με 10000 nodes και αφού ταξινομήθηκε αποθηκεύτηκε στο καινούργιο αρχείο. Η μόνη διαφορά που είχε επίδραση στην απόδοση την ταξινόμησης των δύο περιπτώσεων, είναι ότι το αρχικό αρχείο που διαβάστηκε για να δημιουργηθεί το αρχείο Δ ήταν σημαντικά μικρότερο (625 σελίδες) συγκριτικά με με το αρχικό αρχείο (2500 σελίδες) που διαβάστηκε για να δημιουργηθεί το Γ Αρχείο.

Περιγραφή κώδικα

Η πιο σημαντική κλάση για την υλοποίηση της άσκησης είναι η 'FileManager'. Μεταβλητές στην κλάση αυτή ορίστηκαν, το όνομα του αρχείου , ο αριθμός των σελιδών, η τρέχουσα θέση του κέρσορα στο αρχείο και το 'MyFile' που είναι τύπου RandomAccessFile. Με τη μέθοδο 'CreateFile', δημιουργείται ένα αρχείο με μία σελίδα όπου περιέχει τον αριθμό των σελιδών και το όνομα του αρχείου. Επίσης ,αρχικοποιείται ο αριθμός των σελιδών με μηδέν και η τρέχουσα θέση στο 1. Η υπόλοιπη κλάση, αποτελείται από μεθόδους υπεύθυνες για την επεξεργασία του αρχείου.

Τα υπόλοιπα ερωτήματα υλοποιήθηκαν σε μία μόνο κλάση την 'Modes'. Πιο συγκεκριμένα, περιέχει την αρχικοποίηση των αρχείων για τους Α , Β, Γ και Δ τρόπους οργάνωσης καθώς και τις απαραίτητες μεθόδους για την σειριακή και δυαδική αναζήτηση των 20 κλειδιών σε κάθε αρχείο.

Εδικότερα, η μέθοδος που δημιουργεί το αρχείο με τον πρώτο τρόπο οργάνωσης γεμίζει την κάθε σελίδα των 128 bytes με 4 Nodes που το κάθε ένα περιέχει ένα μοναδικό κλειδί και ένα string που παράγονται τυχαία.

Συνεχίζοντας, το σκεπτικό του Β τρόπου οργάνωσης αρχείου είναι η δημιουργία 2 αρχείων , ενός που περιέχει τα κλειδιά και τις πληροφορίες (case.bin) και ενός που περιέχει μόνο το κλειδί και τη θέση της σελίδας που βρίσκεται αυτό το κλειδί στο μεγάλο αρχείο (caseBindexFile.bin). Τα δύο αυτά αρχεία δημιουργήθηκαν παράλληλα σε μία μέθοδο. Πιο συγκεκριμένα, ανά 4 σελίδες που δημιουργούνται στο αρχείο 'case.bin' δημιουργείται μία σελίδα (που περιέχει 16 nodes) στο αρχείο 'caseBindexFile.bin'.

Η σειριακή αναζήτηση έγινε και στις δύο περιπτώσεις με τα ίδια 20 τυχαία κλειδιά. Η σειριακή αναζήτηση των δύο αρχείων υλοποιείται σε μία μέθοδο . Στην περίπτωση που ψάχνουμε στο 'caseBindexFile.bin' μας γυρνάει την θέση της σελίδας στο αρχείο 'case.bin' του κλειδιού που αναζητούμε. Κατόπιν, η θέση αυτή γίνεται όρισμα για μία άλλη μέθοδο που ψάχνει σειριακά για το κλειδί στη σελίδα του αρχείου 'case.bin'.

Για την ταξινόμηση του αρχείου με τον Γ τρόπο οργάνωσης ,αρχείου χρησιμοποιείται μία μέθοδος όπου ανοίγει το αρχείο 'caseA.bin' , το διαβάζει ανά σελίδα και το χωρίζει σε 4 Nodes οι οποίοι αργότερα τοποθετούνται σε ένα array από 10000 Nodes οι οποίοι ταξινομούνται και γράφονται σε ένα νέο αρχείο το 'caseC.bin'.

Ο Δ τρόπος οργάνωσης υλοποιήθηκε με μία μέθοδο όπου ανοίγει το αρχείο 'caseBindexFile.bin' το διαβάζει και ανά σελίδα το χωρίζει σε 16 IndexNodes που

αργότερα τοποθετούνται σε ένα array από 10000 IndexNodes οι οποίοι ταξινομούνται και γράφονται ανά 16 σε ένα νέο αρχείο το ‘caseDindexFile.bin’ .

Η δυαδική αναζήτηση έγινε και στις δύο περιπτώσεις με τα ίδια 20 τυχαία κλείδια που χρησιμοποιήθηκαν και για την σειριακή. Στην περίπτωση Γ, η μέθοδος που την υλοποιεί δέχεται ορίσματα το αριστερό και το δεξί άκρο της αναζήτησης, το κλειδί, έναν FileManager που περίεχει το ανοιχτό αρχείο που θα υποστεί την αναζήτηση καθώς και έναν αριθμό που θα χρησιμοποιηθεί ως όρισμα στη μέθοδο getLastKey και βοηθάει για να πάρουμε τη σωστή συνάρτηση για το κάθε αρχείο (αυτό έγινε για να γίνει η getLastKey κοινή και για τις 2 περιπτώσεις). Για την μέθοδο της περίπτωσης Δ, ισχύουν ακριβώς τα ίδια με μόνη διαφορά ότι με το που βρεθεί το κλειδί στο αρχείο ‘caseBindexFile.bin’, παίρνουμε την θέση της σελίδας που του αντιστοιχεί και ψάχνουμε στο αρχείο ‘caseB.bin’ στην θέση αυτή το κλειδί σειριακά.

Η μόνη πηγή πληροφόρησης που χρησιμοποιήθηκε είναι η παρακάτω:

https://stackoverflow.com/questions/2183240/java-integer-to-byte-array/2183259

Κώδικας που πάρθηκε: byte[] INT = ByteBuffer.allocate(4).putInt(i).array();

Τέλος, υπήρξε ανταλλαγή ιδεών με τον συνάδελφο Νικόλαο Αγγελίδη.