

Δομές Δεδομένων και Αρχείων

Κωνσταντίνος Κασφίκης
2013030108

Περιγραφή Άσκησης

Στην συγκεκριμένη άσκηση ζητείται η υλοποίηση ενός B+ Tree στον δίσκο. Τα B+ Trees αποτελούν μια ιδιαίτερη μορφή B Trees η οποία επιτρέπει την αναζήτηση εύρους.

Όπως και τα B Trees τα B+ Trees αποτελούνται από δύο ειδών κόμβους, τους εσωτερικούς και τα φύλλα. Οι εσωτερικοί κόμβοι (ή αλλιώς index) ουσιαστικά χρησιμεύουν στην ευκολότερη αναζήτηση δεδομένων, περιέχοντας ο καθένας μερικά από το σύνολο των κλειδιών. Έτσι, μέσω σύγκρισης του κάθε κλειδιού του εκάστοτε εσωτερικού κόμβου επιλέγεται ο κατάλληλος δείκτης προς το παρακάτω επίπεδο και διασχίζεται το δέντρο μέχρι να βρεθεί το φύλλο που περιέχει το κλειδί που μας ενδιαφέρει. Οι εσωτερικοί κόμβοι περιέχουν τόσο κλειδιά όσο και δείκτες σε παρακάτω κόμβους, ενώ τα φύλλα περιέχουν κλειδιά και δεδομένα. Ως βαθμός του B+ Tree (n) ορίζεται ο μέγιστος αριθμός δεικτών που μπορεί να περιέχει ένας εσωτερικός κόμβος.

Η περιγραφή της λειτουργίας του B+ Tree είναι η εξής: κατά την εισαγωγή του πρώτου κλειδιού δημιουργείται ένα φύλλο που λειτουργεί και ως ρίζα του δέντρου. Με την εισαγωγή $n-1$ στοιχείων στο δέντρο το φύλλο αυτό διχοτομείται σε 2 φύλλα και την θέση του καταλαμβάνει ένας εσωτερικός κόμβος ενός στοιχείου (συγκεκριμένα του $n/2$ στοιχείου που εισήχθη) που περιέχει 2 δείκτες (ο πρώτος στο πρώτο παραγόμενο φύλλο που περιέχει τα 1 μέχρι $((n/2)-1)-1$ κλειδιά που εισήχθησαν, και ο άλλος στο δεύτερο παραγόμενο φύλλο που περιέχει τα κλειδιά $(n/2)-1$ μέχρι $n-1$). Η εισαγωγή κλειδιών συνεχίζεται κανονικά εισάγωντας το εκάστοτε κλειδί στο κατάλληλο φύλλο και σχηματίζοντας παράλληλα τους κατάλληλους εσωτερικούς κόμβους μέχρι κάποιο φύλλο ή εσωτερικός κόμβος να γεμίσει οπότε πραγματοποιείται διχοτόμηση στον κόμβο αυτόν.

Μέρος 1^ο

Στον δίσκο, κάθε φύλλο και κάθε εσωτερικός κόμβος στο παραγόμενο αρχείο έχουν την παρακάτω μορφή:

INTERNAL

<u>-200</u>	Max	Num	Parent	PTR-1	KEY-1	PTR-n	KEY-n
-------------	-----	-----	--------	-------	-------	-------	-------	-------	-------

LEAF

<u>-300</u>	m	Num	Next	Parent	KEY-1	DATA-1	KEY-n	Data-n
-------------	---	-----	------	--------	-------	--------	-------	-------	--------

Όπου Num ο αριθμός που δείχνει πόσα ακριβώς κλειδιά περιέχει ο εκάστοτε κόμβος (δείχνει επίσης πόσα δεδομένα περιέχει ένα φύλλο και τον αριθμό των δεικτών ενός εσωτερικού κόμβου μειωμένο κατά 1), Parent δείκτης προς τον παραπάνω κόμβο του εκάστοτε κόμβου, KEY₁ έως KEY_n τα κλειδιά που περιέχει ο κόμβος. Για τα φύλλα, m είναι ο αριθμός bytes που το κάθε δεδομένο κλειδιού καταλαμβάνει, Next είναι ο δείκτης στο επόμενο φύλλο (δεδομένο το οποίο επιτρέπει την αναζήτηση εύρους) και DATA₁ έως DATA_n τα δεδομένα m bytes που συνοδεύουν το κάθε κλειδί. Για τους εσωτερικούς κόμβους, Max είναι ο μέγιστος αριθμός κλειδιών, PTR έως PTR_n είναι το σύνολο των δεικτών προς τους παρακάτω κόμβους (children) του κάθε εσωτερικού. Ο πρώτος ακέραιος του κάθε κόμβου (-200 ή -300) χρησιμοποιείται για να διευκρινιστεί αν πρόκειται για φύλλο ή εσωτερικό κόμβο. Πολλές από τις παραπάνω βοηθητικές μεταβλητές των κόμβων χρησιμοποιούνται προκειμένου το αρχείο με το παραγόμενο δέντρο να μπορεί να τροποποιηθεί και εκτός του συγκεκριμένου προγράμματος (π.χ. m).

Για την υλοποίηση της άσκησης υπολογίζεται το n με τον εξής τρόπο:

$$n = ((N - 20) / m + 4) + 1$$

Όπου N το μέγεθος της σελίδας δίσκου, m το μέγεθος των δεδομένων σε byte που συνοδεύουν κάθε κλειδί. Ο υπολογισμός του n γίνεται με τρόπο τέτοιο ώστε ο αριθμός κλειδιών στους εσωτερικούς κόμβους να ισούται με τον αριθμό κλειδιών στα φύλλα. Τα 20 bytes που αφαιρούνται από το σύνολο των bytes της

σελίδας δίσκου αποτελούν τις βοηθητικές μεταβλητές για την "ανάγνωση" ή "εγγραφή" που κάθε κόμβος περιέχει.

Κανονικά το n θα έπρεπε να υπολογίζεται από τον εξής τύπο:

$$n=(N-20)/8$$

Όπου από το σύνολο των bytes της σελίδας αφαιρούνται 16 bytes που δεσμεύουν οι 4 πρώτοι ακέραιοι και 4 bytes του 1ου δείκτη. Ο παραγόμενος αριθμός διαιρείται με το 8 για τα ζεύγη των ακεραίων δεικτών-κλειδιών. Ωστόσο, ο παραπάνω τύπος δεν χρησιμοποιείται στην συγκεκριμένη υλοποίηση του B+ Tree γιατί τα φύλλα και οι εσωτερικοί κόμβοι θα είχαν διαφορετικό αριθμό κλειδιών, χρησιμοποιείται ωστόσο μόνο για μέγεθος δεδομένων $m=4$ bytes.

Στο ερώτημα αυτό πρέπει να πραγματοποιηθεί η διαδικασία της εισαγωγής στο δέντρο, της αναζήτησης και της αναζήτησης εύρους.

Εισαγωγή:

Η εισαγωγή βασίζεται σε πολλές μεθόδους με βασικές την NonFullNode και την split που βρίσκονται στην κλάση BPlusTree. Η NonFullNode είναι η μέθοδος η οποία εισάγει το κάθε κλειδί στο κατάλληλο φύλλο, διασχίζοντας όλους τους εσωτερικούς κόμβους μέχρι να βρει το κατάλληλο φύλλο, ενώ η split διχοτομεί τους γεμάτους κόμβους δημιουργώντας έτσι τους εσωτερικούς κόμβους ή προσθέτοντας δείκτες σε αυτούς προς τους επιπλέον παραγόμενους κόμβους, αυξάνοντας κατά αυτόν τον τρόπο συνεχώς την "χωρητικότητα" του δέντρου. Η εισαγωγή τελικά γίνεται μέσω της μεθόδου insert της κλάσης BPlusTree.

Αναζήτηση:

Η αναζήτηση πραγματοποιείται με "είσοδο" ένα κλειδί του οποίου πρέπει να επιστραφούν τα δεδομένα (m bytes). Χρησιμοποιείται αναδρομική συνάρτηση (η οποία χρησιμοποιεί πολλές βοηθητικές μεθόδους) η οποία καλεί τον εαυτό της μέχρι να βρει φύλλο. Έτσι, κάθε φορά που καλείται η αναδρομική αυτή συνάρτηση αν ο κόμβος που εξετάζει είναι εσωτερικός ψάχνει τα κλειδιά του μέχρι να βρει τον κατάλληλο δείκτη προς το παρακάτω επίπεδο, αλλιώς αν εξετάζει φύλλο ψάχνει τα κλειδιά αυτού μέχρι να βρει το ζητούμενο και επιστρέφει τα δεδομένα του. Η αναζήτηση πραγματοποιείται από την μέθοδο search της κλάσης BPlusTree.

Αναζήτηση Εύρους:

Η αναζήτηση εύρους πραγματοποιείται με "είσοδο" 2 κλειδιά, τα οποία δείχνουν την ελάχιστη και την μέγιστη τιμή του εύρους. Προκειμένου να υλοποιηθεί η αναζήτηση εύρους αρχικά καλείται η συνάρτηση `getNodeLeafSearch` η οποία επιστρέφει τον κόμβο στον οποίο βρίσκεται το κλειδί με την ελάχιστη τιμή. Με δεδομένο το αρχικό φύλλο που επέστρεψε η παραπάνω συνάρτηση η συνάρτηση `rangeSearch` διασχίζει τα φύλλα (χρησιμοποιώντας το `Next` που κάθε φύλλο περιέχει) μέχρι να βρει την μέγιστη τιμή του εύρους αναζήτησης ή κάποια μεγαλύτερη τιμή. Η `rangeSearch` θα επιστρέψει τελικά έναν πίνακα τύπου `Data` (όπου `Data` είναι η κλάση στην οποία παράγονται και αποθηκεύονται τα δεδομένα του κάθε κλειδιού) με το σύνολο όλων των δεδομένων των κλειδιών που βρίσκονται μέσα στο εύρος αναζήτησης.

Η υλοποίηση του προγράμματος γίνεται στον δίσκο. Έτσι για κάθε μία από τις παραπάνω λειτουργίες όσο και για την διαγραφή είναι απαραίτητη η μετατροπή συγκεκριμένων δεδομένων σε ακέραιους. Την διαδικασία αυτή πραγματοποιούν βοηθητικές συναρτήσεις (π.χ `getChildNodes()`, `getNumKeys()` κ.ο.κ). Επίσης, για την εγγραφή των δεδομένων του κάθε κόμβου χρησιμοποιούνται μέθοδοι οι οποίες μετατρέπουν τα στοιχεία από ακέραιους σε bytes (`integer=4bytes`) και τα τοποθετούν στις κατάλληλες θέσεις σε κάθε κόμβο-σελίδα δίσκου (π.χ. `writeInternal()`, `writeLeaf()`) . Σημειώνεται επίσης ότι για την διευκόλυνση του χρήστη κάθε δεδομένο περιέχει στα τελευταία 4 bytes του έναν integer ίσο με τον κλειδί του, ενώ τα υπόλοιπα bytes έχουν παραχθεί τυχαία.

Στο πρώτο μέρος της άσκησης ζητείται να πραγματοποιηθούν η εισαγωγή κλειδιών από το $1-10^5$ και στην συνέχεια η αναζήτηση 20 τυχαίων κλειδιών στο δέντρο αυτό, μετρώντας ταυτόχρονα το μέσο όρο προσβάσεων στο δίσκο. Τέλος ζητείται η πραγματοποίηση 20 αναζητήσεων εύρους και να μετρηθεί ο μέσος όρος προσβάσεων στο δίσκο.

Για $N=1024$, $m=20$ στις αναζητήσεις ο μέσος όρος προσβάσεων υπολογίζεται περίπου στο 3 ενώ στις αναζητήσεις εύρους υπολογίζεται στο 10.

Αναζήτηση:

```
-----
The 31147 key has data:[B83d4eac69 and the four last bytes of these data are(converted to integer):31147 and it was found with 3 disk accesses.
-----
The 19401 key has data:[284e2a1792 and the four last bytes of these data are(converted to integer):19401 and it was found with 3 disk accesses.
-----
The 27849 key has data:[B875b56c32 and the four last bytes of these data are(converted to integer):27849 and it was found with 3 disk accesses.
-----
The 92906 key has data:[B86bc7c334 and the four last bytes of these data are(converted to integer):92906 and it was found with 3 disk accesses.
-----
The 42471 key has data:[B8332204a1 and the four last bytes of these data are(converted to integer):42471 and it was found with 3 disk accesses.
-----
The 39982 key has data:[B84aa29fb7 and the four last bytes of these data are(converted to integer):39982 and it was found with 3 disk accesses.
-----
The 77172 key has data:[B87d4991ad and the four last bytes of these data are(converted to integer):77172 and it was found with 3 disk accesses.
-----
The 37190 key has data:[B82ed33b30 and the four last bytes of these data are(converted to integer):37190 and it was found with 3 disk accesses.
-----
The 14615 key has data:[B81b6d3336 and the four last bytes of these data are(converted to integer):14615 and it was found with 3 disk accesses.
-----
The 29626 key has data:[B845546170 and the four last bytes of these data are(converted to integer):29626 and it was found with 3 disk accesses.
-----
The 22142 key has data:[D87a214402 and the four last bytes of these data are(converted to integer):22142 and it was found with 3 disk accesses.
-----
The 12326 key has data:[B81940e19d and the four last bytes of these data are(converted to integer):12326 and it was found with 3 disk accesses.
-----
The 22745 key has data:[B8677327b6 and the four last bytes of these data are(converted to integer):22745 and it was found with 3 disk accesses.
-----
The 94893 key has data:[B814ae5a5 and the four last bytes of these data are(converted to integer):94893 and it was found with 3 disk accesses.
-----
The average number of disk accesses for the search is: 3
```


Αναζήτηση εύρους:

```
The 15426 key has data:[B07692d9cc and the four last bytes of these data are(converted to integer):15426
The 15427 key has data:[B075f32542 and the four last bytes of these data are(converted to integer):15427
The 15428 key has data:[B07f1302d6 and the four last bytes of these data are(converted to integer):15428
The 15429 key has data:[B043ee72e6 and the four last bytes of these data are(converted to integer):15429
The 15430 key has data:[B023529fee and the four last bytes of these data are(converted to integer):15430
The 15431 key has data:[B04fe767f3 and the four last bytes of these data are(converted to integer):15431
The 15432 key has data:[B02805c96b and the four last bytes of these data are(converted to integer):15432
The 15433 key has data:[B0184cf7cf and the four last bytes of these data are(converted to integer):15433
The 15434 key has data:[B02fd6b6c7 and the four last bytes of these data are(converted to integer):15434
The 15435 key has data:[B05bfa9431 and the four last bytes of these data are(converted to integer):15435
The 15436 key has data:[B05db250b4 and the four last bytes of these data are(converted to integer):15436
The 15437 key has data:[B0223f3642 and the four last bytes of these data are(converted to integer):15437
The 15438 key has data:[B038c5cc4c and the four last bytes of these data are(converted to integer):15438
The 15439 key has data:[B037918c79 and the four last bytes of these data are(converted to integer):15439
The 15440 key has data:[B078e94dcf and the four last bytes of these data are(converted to integer):15440
The 15441 key has data:[B0233fe9b6 and the four last bytes of these data are(converted to integer):15441
The 15442 key has data:[B0358ee631 and the four last bytes of these data are(converted to integer):15442
The 15443 key has data:[B0ec756bd and the four last bytes of these data are(converted to integer):15443
The 15444 key has data:[B03c72f59f and the four last bytes of these data are(converted to integer):15444
The 15445 key has data:[B060dcc9fe and the four last bytes of these data are(converted to integer):15445
The 15446 key has data:[B0222114ba and the four last bytes of these data are(converted to integer):15446
The 15447 key has data:[B016e7dcfd and the four last bytes of these data are(converted to integer):15447
The 15448 key has data:[B03d121db3 and the four last bytes of these data are(converted to integer):15448
The 15449 key has data:[B03b07a0d6 and the four last bytes of these data are(converted to integer):15449
The 15450 key has data:[B011a9e7c8 and the four last bytes of these data are(converted to integer):15450
The 15451 key has data:[B03901d134 and the four last bytes of these data are(converted to integer):15451
The 15452 key has data:[B014d3bc22 and the four last bytes of these data are(converted to integer):15452
The 15453 key has data:[B012d4bf7e and the four last bytes of these data are(converted to integer):15453
The 15454 key has data:[B04c1d9d4b and the four last bytes of these data are(converted to integer):15454
The 15455 key has data:[B07b227d8d and the four last bytes of these data are(converted to integer):15455
The 15456 key has data:[B07219ec67 and the four last bytes of these data are(converted to integer):15456
The 15457 key has data:[B045018215 and the four last bytes of these data are(converted to integer):15457
The 15458 key has data:[B065d6b83b and the four last bytes of these data are(converted to integer):15458
The 15459 key has data:[B0d706f19 and the four last bytes of these data are(converted to integer):15459
The 15460 key has data:[B030b7c004 and the four last bytes of these data are(converted to integer):15460
The 15461 key has data:[B079efed2d and the four last bytes of these data are(converted to integer):15461
The 15462 key has data:[B02928854b and the four last bytes of these data are(converted to integer):15462
The 15463 key has data:[B027ae2fd0 and the four last bytes of these data are(converted to integer):15463
The 15464 key has data:[B029176cc1 and the four last bytes of these data are(converted to integer):15464
The 15465 key has data:[B02f177a4b and the four last bytes of these data are(converted to integer):15465
The 15466 key has data:[B04278a03f and the four last bytes of these data are(converted to integer):15466
The 15467 key has data:[B0147ed70f and the four last bytes of these data are(converted to integer):15467
The 15468 key has data:[B061dd025 and the four last bytes of these data are(converted to integer):15468
```

The average number of disk accesses for the range search is: 10
=====

Μέρος 2ο

Στο δεύτερο μέρος ζητείται η διαγραφή 20 τυχαίων κλειδίων από το δέντρο και ο υπολογισμός του μέσου όρου προσβάσεων στον δίσκο για την διαγραφή ενός στοιχείου.

Η διαγραφή πραγματοποιείται με την κλήση της μεθόδου `delete` της κλάσης `BPlusTree`. Υπάρχουν 2 διαφορετικές περιπτώσεις διαγραφής:

1) Διαγραφή μη πρώτου κλειδιού από φύλλο

2) Διαγραφή του πρώτου κλειδιού από φύλλο

Στην πρώτη περίπτωση πρέπει να διαγραφεί το κλειδί από το φύλλο και στη συνέχεια να αλλάξει η τιμή του συγκεκριμένου κλειδιού στον πατέρα κόμβο (`parent`) στην τιμή του δεύτερου κλειδιού του φύλλου - έστω x - (πριν την διαγραφή του πρώτου κλειδιού). Αν και στον πατέρα του πατέρα κόμβου το κλειδί αυτό είναι το πρώτο κλειδί τότε πρέπει να αλλάξει η τιμή του σε x . Η διαδικασία αυτή επαναλαμβάνεται μέχρι να μην υπάρχει το κλειδί αυτό σε κανέναν κόμβο.

Στην δεύτερη περίπτωση η διαγραφή του κλειδιού από το φύλλο αρκεί.

Η μέθοδος `delete` όπως και η μέθοδος `rangeSearch` καλεί την μέθοδο `getNodeLeafSearch` η οποία εντοπίζει το φύλλο στο οποίο είναι τοποθετημένα τα κλειδί που θέλουμε να διαγράψουμε και τα δεδομένα του. Στη συνέχεια η `delete` διαγράφει για κάθε περίπτωση το στοιχείο μειώνοντας παράλληλα κατά ένα τον αριθμό κλειδίων του επηρεαζόμενου κόμβου-φύλλο.

Ο μέσος όρος προσβάσεων στο δίσκο για την διαγραφή ενός στοιχείου υπολογίζεται στο 6.

```

The 92435 key and its data were successfully deleted from this b+ tree with 6 disk accesses.
-----
The 50007 key and its data were successfully deleted from this b+ tree with 6 disk accesses.
-----
The 90331 key and its data were successfully deleted from this b+ tree with 6 disk accesses.
-----
The 71770 key and its data were successfully deleted from this b+ tree with 6 disk accesses.
-----
The 36924 key and its data were successfully deleted from this b+ tree with 6 disk accesses.
-----
The 12795 key and its data were successfully deleted from this b+ tree with 6 disk accesses.
-----
The 56543 key and its data were successfully deleted from this b+ tree with 6 disk accesses.
-----
The 17518 key and its data were successfully deleted from this b+ tree with 6 disk accesses.
-----
The 22965 key and its data were successfully deleted from this b+ tree with 6 disk accesses.
-----
The 74129 key and its data were successfully deleted from this b+ tree with 6 disk accesses.
-----
The 95804 key and its data were successfully deleted from this b+ tree with 6 disk accesses.
-----
The 34994 key and its data were successfully deleted from this b+ tree with 6 disk accesses.
-----
The 81406 key and its data were successfully deleted from this b+ tree with 6 disk accesses.
-----
The 16308 key and its data were successfully deleted from this b+ tree with 6 disk accesses.
-----
The 49692 key and its data were successfully deleted from this b+ tree with 6 disk accesses.
-----
The average number of disk accesses for delete is: 6

```


(Bonus) Μέρος 3ο

Γίνονται μετρήσεις για $N=1024$ και $m=20$.

Ο μέσος όρος προσβάσεων στον δίσκο για:

Αναζήτηση : 3

Αναζήτηση Εύρους: 10

Διαγραφή: 6

Επαναλαμβάνονται οι ίδιες μετρήσεις για $N=1024$ και $m=50$

Ο μέσος όρος προσβάσεων στον δίσκο για:

Αναζήτηση : 2

Αναζήτηση Εύρους: 8

Διαγραφή: 5

Η διαφορά ανάμεσα στο σύνολο των μετρήσεων για διαφορετικά m οφείλεται στο γεγονός ότι με την υλοποίηση του συγκεκριμένου B+ Tree το m είναι αντιστρόφως ανάλογο του n (και επομένως του μέγιστου αριθμού κλειδιών που ένας κόμβος μπορεί να περιέχει). Όσο μικρότερο είναι το m τόσο περισσότερα κλειδιά χωράνε σε έναν κόμβο. Εφόσον αυξάνεται ο αριθμός των κλειδιών του κάθε κόμβου, μειώνεται ο αριθμός των φύλλων και των εσωτερικών κόμβων του δέντρου με αποτέλεσμα οι εύρεση του φύλλου με το κλειδί που μας ενδιαφέρει να γίνεται διασχίζοντας λιγότερους εσωτερικούς κόμβους. Ο παραπάνω παράγοντας αφορά όλες τις λειτουργίες του προγράμματος. Ωστόσο η αναζήτηση εύρους με μικρότερο m επηρεάζεται και από το γεγονός ότι τα φύλλα περιέχουν περισσότερα κλειδιά και επομένως δεδομένα, με αποτέλεσμα να μεταφέρονται σε λιγότερα επόμενα φύλλα.

Μετά από την υλοποίηση των παραπάνω μετρήσεων το πρόγραμμα μεταβαίνει σε ένα menu που δίνει στον χρήστη την δυνατότητα να εισάγει, διαγράψει και να αναζητήσει κλειδιά. Του δίνεται ακόμα να δημιουργήσει B+ Tree με N, m που ο ίδιος θα καθορίσει.

Το πρόγραμμα υλοποιήθηκε σε γλώσσα προγραμματισμού JAVA σε περιβάλλον Eclipse για Windows.