Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών

Βάσεις Δεδομένων

2η Σειρά Γραπτών Ασκήσεων

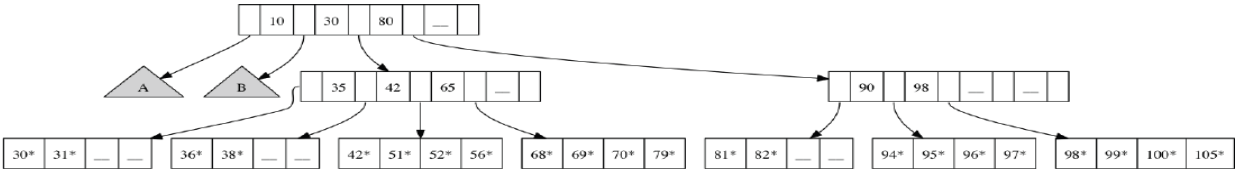
Διονύσης Ζήνδρος <dionyziz@gmail.com>
03106601

1η Φεβρουαίου 2012

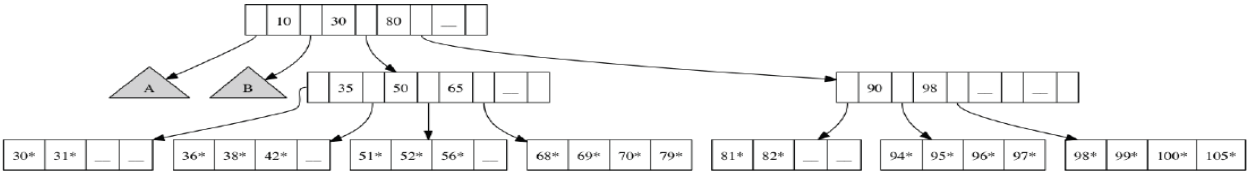
Άσκηση 1

(α)

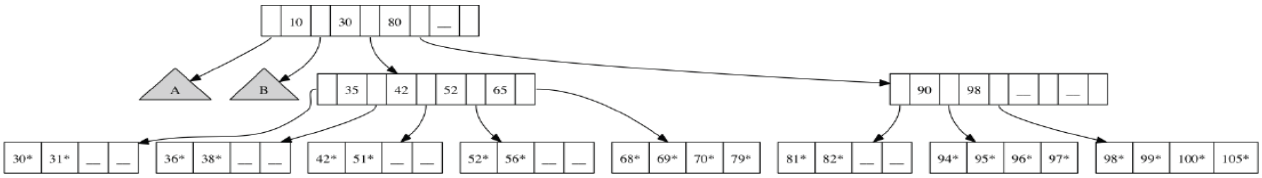
(1) Ο κόμβος συχγωνεύεται με το δεξιό του αδερφό. Επειδή περιέχει το 42 που είναι μικρότερο από όλα τα υπόλοιπα, θα βρίσκεται στην πρώτη θέση του νέου κόμβου. Θα πρέπει επίσης να γίνουν οι κατάλληλες αλλαγές στον πατέρα, δηλαδή να διαγραφεί το 42, και το 50 να αλλάξει σε 42.



(2) Στην περίπτωση συγχώνευσης με τον αριστερό αδερφό, το 42 θα μπει στο τέλος του νέου κόμβου, αφού είναι το μεγαλύτερο από όλα. Αυτή τη φορά ο πατέρας δεν χρειάζεται να αλλάξει, και άρα αλώς αφαιρείται το 42 από αυτόν.



(3) Σε περίπτωση ανακατανομής, το 51 που είναι ο μικρότερος αριθμός του δεξιού κόμβου, μεταφέρεται στον αριστερό. Έτσι ενημερώνεται και ο πατέρας του δεξιού κόμβου και γίνεται 52.



(β) Το μικρότερο τοπικά κόστος επιτυγχάνεται με τη συγχώνευση αφού δεν απαιτείται διαγραφή κάποιου κόμβου-πατέρα.

(γ) Η συγχώνευση σε κοινό κόμβο είναι προτιμότερη σε περίπτωση που πρόκειται να ακολουθήσουν διαγραφές, αφού μειώνεται ο πλήθος των κόμβων, ενώ η ανακατανομή είναι προτιμότερη σε περίπτωση που πρόκειται να ακολουθήσουν εισαγωγές, αφού αφήνει περισσότερο κενό χώρο μέσα στους κόμβους για νέες τιμές.

Άσκηση 2

Υπάρχουν οι 23 δείκτες σε blocks που φαίνονται στο σχήμα. Λόγω του ότι το δέντρο πρέπει να είναι ισορροπημένο, άρα υπάρχουν τουλάχιστον 12 δείκτες σε blocks συνολικά στα A και B υποδέντρα (και άρα ακριβώς 12 αφού αυτά σύμφωνα με την εκφώνηση είναι ελάχιστα). Άρα υπάρχουν 35 blocks δηλαδή το πολύ 700 πλειάδες.

Άσκηση 3

Αφού το γνώρισμα Α είναι υποψήφιο κλειδί, άρα κάθε εγγραφή έχει μοναδική τιμή για αυτό, δηλαδή η σχέση εγγραφής → τιμής κλειδιού είναι «ένα-προς-ένα». Επειδή το πλήθος των τιμών που παίρνει το Α είναι ίσο με το πλήθος των εγγραφών, η σχέση είναι επίσης και «επί». Επειδή οι εγγραφές είναι ταξινομημένες στο αρχείο με βάση το γνώρισμα Α, άρα η πρώτη εγγραφή έχει Α = 0, η δεύτερη εγγραφή έχει Α = 1, κ.ό.κ. μέχρι την τελευταία εγγραφή που έχει Α = 5,999,999. Επειδή σύμφωνα με την εκφώνηση κάθε block χωράει ακριβώς 10 εγγραφές, ξέρουμε σε ποιο block θα είναι κάθε εγγραφή με συγκεκριμένη τιμή γνωρίσματος χωρίς να χρησιμοποιήσουμε καθόλου κλειδιά. Επιπλέον, αυτή η μέθοδος προσπέλασης είναι βέλτιστη. Άρα η χρήση ευρετηρίου στη συγκεκριμένη περίπτωση είναι άχρηστη. Έχουμε συνεπώς τους ακόλουθους χρόνους προσπέλασης:

(1) σΑ < 60,000 προσπέλαση στα πρώτα 6,000 blocks
(2) σΑ = 60,000 μία προσπέλαση στο 6,000ό block
(3) σΑ > 60,000 AND σΑ < 60,0010 μία προσπέλαση στο 6,000ό block
(4) σΑ <> 60,000 προσπέλαση σε όλα τα blocks

Ο λόγος που η μέθοδος αυτή είναι βέλτιστη είναι το ότι περιορίζεται στην ανάγνωση των blocks που περιέχουν τις ίδιες τις εγγραφές, οι οποίες ούτως ή άλλως θα πρέπει να διαβαστούν.

Σε περίπτωση, όμως, που αποφασίσουμε να βάλουμε περισσότερες από 10 εγγραφές σε ένα block (π.χ. μία εγγραφή να μπει η μισή σε ένα block και η μισή στο επόμενο block) δεν μπορεί να χρησιμοποιηθεί αυτή η μέθοδος.

Τότε αν χρησιμοποιηθεί το ευρετήριο κατακερματισμού με χρήση συνάρτησης κατακερματισμού έχουμε τα εξής:

(1) σΑ < 60,000 Η χρήση κατακερματισμού δεν βελτιώνει την άμεση πρόσβαση στο δίσκο σε αυτή την περίπτωση, οπότε δεν χρησιμοποιούμε το ευρετήριο. Διαφορετικά θα έπρεπε να κοιτάξουμε όλους τους h(1), h(2), …, h(60,000) που κάδους που ενδέχεται να είναι και 60,000.
(2) Η χρήση κατακερματισμού είναι αποδοτική σε αυτή την περίπτωση, αφού μπορούμε να εντοπίσουμε την εγγραφή με 1 μόνο I/O.
(3) Θα πρέπει να ελέγξουμε τους κάδους h(60,000), h(60,001), …, h(60,010) δηλαδή να κάνουμε 10 προσπελάσεις I/O.
(4) Η χρήση κατακερματισμού δεν βελτιώνει το αποτέλεσμα σε αυτή την περίπτωση, άρα μπορούμε να χρησιμοποιήσουμε απευθείας πρόσβαση στο δίσκο. Διαφορετικά θα πρέπει να περάσουμε από όλους τους κάδους.

Σε περίπτωση που χρησιμοποιήσουμε B+ δέντρο με *b* = 100 fan-out έχουμε:

(1) σΑ < 60,000 ακολουθούμε την ίδια μέθοδο με πριν ξεκινώντας από την αρχή του αρχείου μέχρι να βρούμε την τελευταία εγγραφή που θα είναι εκείνη με σΑ = 60,000. Το κόστος είναι και πάλι βέλτιστο με πρόσβαση σε 6,000 blocks. Δεν χρησιμοποιούμε το ευρετήριο, ή το χρησιμοποιούμε κάνοντας απλώς γραμμική αναζήτηση στο επίπεδο των φύλλων.

(2) Κάνουμε δυαδική αναζήτηση με $\log_b(6,000,000) = 4$ I/O προσπελάσεις για να βρούμε το block στο οποίο είναι η εγγραφή που ζητάμε.
(3) Κάνουμε δυαδική αναζήτηση όπως πριν με $\log_b(6,000,000) = 4$ προσπελάσεις και στη συνέχεια οι εγγραφές που μας ενδιαφέρουν υπάρχουν στο τελευταίο block που διαβάσαμε.
(4) Θα πρέπει να διαβάσουμε όλες τις εγγραφές, συνεπώς το ερευτήριο δεν βοηθά. Μπορούμε να το χρησιμοποιήσουμε για να κάνουμε γραμμική αναζήτηση στο επίπεδο των φύλλων.

Άσκηση 4

(1) Στη χειρότερη περίπτωση, όλες οι εγγραφές θα καταλήξουν στο πρώτο bucket, εκτός από μία εγγραφή που θα μπει στο καθένα από τα υπόλοιπα για να αναγκαστούμε να δεσμεύσουμε τον αντίστοιχο χώρο. Άρα το πρώτο bucket θα έχει 99,001 εγγραφές. Άρα προκύπτουν 1990 blocks στο δίσκο.
(2) Στην καλύτερη περίπτωση, οι εγγραφές θα κατανεμηθούν σε όλα τα buckets ομοιόμορφα, οπότε χρειαζόμαστε 1000 blocks, ένα για κάθε bucket.
(3) Η χειρότερη περίπτωση εδώ επιτυγχάνεται δουλεύοντας ως εξής. Πρώτα βάζουμε από μία εγγραφή σε κάθε κάδο για να καταλάβει από ένα block ο καθένας. Στη συνέχεια βάζουμε 100 εγγραφές σε κάθε κάδο ώστε να καταλάβει ένα δεύτερο block σε όσους μπορούμε. Έτσι γεμίζουμε αρχικά 1000 blocks και μετά 990 δεύτερα blocks μέχρι να μας τελειώσουν οι εγγραφές. Δηλαδή θα έχουμε 1990 blocks στο δίσκο και πάλι.