Δομές δεδομένων και αρχείων 2 εργαστηριακή άσκηση

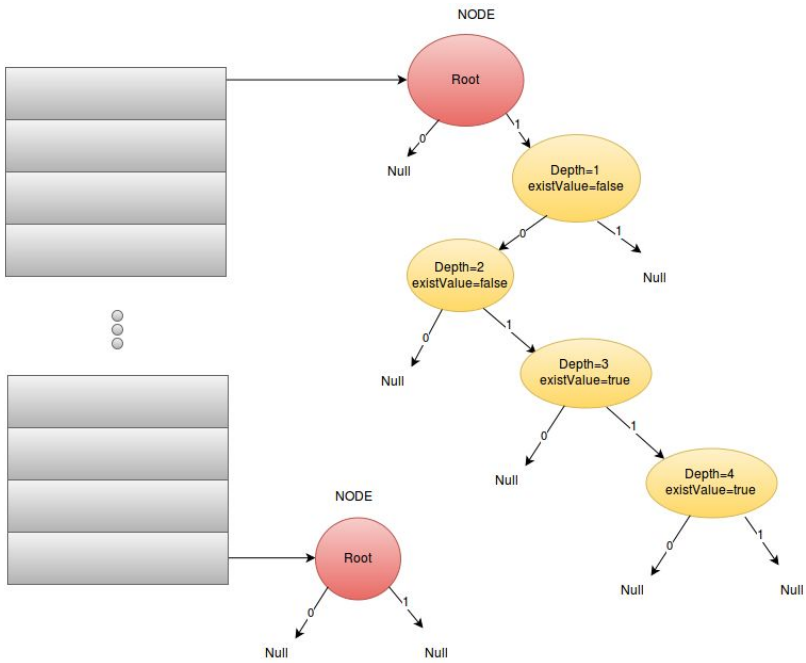
Σπυριδάκης Χρήστος
ΑΜ.2014030022

Σε αυτή την εργαστηριακή άσκηση έπρεπε να υλοποιήσουμε ένα Hash Table στο οποίο θα παίρναμε στατιστικές μετρήσεις (Average, σύμφωνα με διαφορετικό πλήθος στοιχείων τα οποία δίνουμε για εισαγωγή, αναζήτηση και διαγραφή στο Table μας).

Αρχικά για την δημιουργία του δυναμικού κατακερματισμού, δημιουργήθηκε ένας πίνακας ο οποίος σε κάθε θέση του είχε ένα κόμβο root, ο οποίος είναι ο αρχικός κόμβος ενός Trie.

Πιο συγκεκριμένα, δημιουργήθηκε για τις ανάγκες τις άσκησης η κλάση Node η οποία αναπαριστά έναν κόμβο του Trie. Σε αυτή την κλάση υπάρχουν μέσα 2 μεταβλητές Node οι οποίες και αυτές με την σειρά τους, μπορούν να κρατήσουν και από έναν κόμβο (τον Left και Right αντίστοιχο του 0 και 1). Επίσης μέσα σε αυτή την κλάση υπάρχουν οι μέθοδοι εισαγωγής, εύρεσης ή διαγραφής Path το οποίο αναπαριστά value το οποίο χρειαζόμαστε. Αυτές οι μέθοδοι καλούνται μέσω της Class HashingTable, και επιστρέφουν σε αυτή έναν αριθμό ο οποίος είναι ο αριθμός των ελέγχων που χρειάστηκαν για την εκάστοτε διαδικασία.

Στην Class HashingTable υπάρχει αυτούσιος ο πίνακας από Tries με σταθερό μέγεθος 100 θέσεων. Αυτή είναι επίσης η κλάση η οποία συνδέει την Main με τα Node καθώς υπάρχουν μέσα σε αυτή 3 μέθοδοι οι οποίοι παίρνουν τις τιμές που δίνονται στην Main για εισαγωγή/εύρεση/διαγραφή και αφού βρουν μέσω της HashFunction το Key, την θέση δηλαδή στον πίνακα που θα πάει, τον μετατρέπουν σε δυαδικό και τον περνάνε στη μέθοδο που υπάρχει στο Node για την λειτουργία που χρειάζεται, όπου με αναδρομικό τρόπο φτάνει στο σημείο που πρέπει και είτε προσθέτει το value που δώθηκε αν είναι εφικτό, είτε το διαγράφει, είτε το αναζητά.



Παρακάτω παρατίθεται ένα σχεδιάγραμμα, για την καλύτερη κατανόηση της λύσης.
Αναπαριστάται ένα τυχαίο Path για 2 τυχαίες θέσεις του πίνακα το πρώτο που βγάζει το Value :101 και το Value:1101 και το κενό Path, στο οποίο δεν έχουν μπει ακόμα τιμές. Ενώ έχοντας την μεταβλητή ValueExist μπορούμε να γνωρίζουμε αν ο αριθμός που μας έδωσαν για διαγραφή ή αναζήτηση όντως υπάρχει και δεν είναι απλά μέρος ενός αλλου Path.

Επίσης παρατίθεται ένα Screen Shot από τα αποτελέσματα της λειτουργίας του κώδικα:

```
For :20 Values
Average ADD with 20 values: 30
Average SEARCH with 20 values: 1
Average DELETE with 20 values: 1

For :100 Values
Average ADD with 100 values: 36
Average SEARCH with 100 values: 3
Average DELETE with 100 values: 3

For :1000 Values
Average ADD with 1000 values: 34
Average SEARCH with 1000 values: 7
Average DELETE with 1000 values: 7

For :10000 Values
Average ADD with 10000 values: 34
Average SEARCH with 10000 values: 10
Average DELETE with 10000 values: 10
```

Από τα αποτελέσματα του προγράμματος αρχικά βλέπουμε ότι το μέσω Path είναι περίπου 32 bit. Επειδή παίρνουμε τυχαίους αριθμούς μέσω της rand και από ότι φαίνεται επιστρέφει και αρκετούς μεγάλους αριθμούς.
Επίσης λόγο το ότι αρκετά από τα Path που θα προσπαθήσει να ψάξει, είτε για να διαγράψει το στοιχείο τους είτε για να δει αν υπάρχουν, είναι πολύ πιθανόν να μην υπάρχουν, αυτό σημαίνει ότι θα σταματήσει στο σημείο που επαληθεύει ότι το value που ψάχνουμε δεν υπάρχει. Με αποτέλεσμα αν υπάρχει ένα μικρό μέσω όρο συγκρίσεων,

Πηγές:

1) Απόλυτη java, Savith, Εκδόσεις Ιων
2) Δομές δεδομένων και Αλγόριθμοι σε Java, Goodrich, Δίαυλος