Δυαδικό δένδρο έρευνας

Απόδοση και Τεκμηρίωση Αποτελεσμάτων

Πίνακας αποτελεσμάτων

Μέθοδος	Μέσος αριθμός συγκρίσεων/ εισαγωγή	Μέσος αριθμός συγκρίσεων/ τυχαία αναζήτηση	Μέσος αριθμός συγκρίσεων/ αναζήτηση εύρους(K=100)	Μέσος αριθμός συγκρίσεων/ αναζήτηση εύρους(K=1000)
ΔΔΕ Α	60	55	85	437
Νηματοειδές ΔΔΕ Β	62	75	115	555
Ταξινομημένο πεδίο		41	58	146

Να σημειωθεί ότι για να επιτευχθεί καλύτερη σύγκριση των αποτελεσμάτων, σε όλες τις δομές έχουν χρησιμοποιηθεί τα ίδια κλειδιά σε κάθε ερώτημα.

Αρχικά, παρατηρείται μεταξύ του ΔΔΕ και του νηματοειδούς ΔΔΕ ότι ο μέσος αριθμός συγκρίσεων κατά την εισαγωγή είναι σχεδόν ο ίδιος. Πιο συγκεκριμένα, βρέθηκε ότι πάντα ο αριθμός συγκρίσεων κατά την εισαγωγή του νηματοειδούς ΔΔΕ είναι κατά δύο μεγαλύτερος.

Κατά την τυχαία αναζήτηση 100 κλειδιών, παρατηρήθηκε ότι το ταξινομημένο πεδίο έχει τον μικρότερο αριθμό συγκρίσεων με το ΔΔΕ να ακολουθεί με 15 περίπου περισσότερες και τέλος, με την χειρότερη απόδοση το νηματοειδές ΔΔΕ με 30 παραπάνω συγκρίσεις .

Για τον μέσο αριθμό συγκρίσεων στην την αναζήτηση εύρους, σχεδιάστηκε πειραματικά η γραφική παράσταση για K=100, K=1000 και K=2000 και βρέθηκε ότι όλες οι δομές παρουσίαζαν γραμμική αύξηση με διαφορετικό ρυθμό αύξησης όπως φαίνεται και από τον πίνακα αποτελεσμάτων. Το πιο αποδοτικό και με τον μικρότερο ρυθμό αύξησης βρέθηκε ότι είναι το ταξινομημένο πεδίο. Το αμέσως αποδοτικότερο και με μεγαλύτερο ρυθμό αύξησης από προηγουμένως, είναι το ΔΔΕ. Τέλος, το νημετοειδές ΔΔΕ παρουσιάζει τον μεγαλύτερο ρυθμό αύξησης και

συνεπώς την χειρότερη απόδοση από τα τρία κατά την αναζήτηση εύρους.

Συνοψίζοντας, την καλύτερη απόδοση γενικώς την έχει το ταξινομημένο πεδίο.

Περιγραφή κώδικα

Να σημειωθεί ότι η αναλυτική λειτουργία των μεθόδων θα επεξηγούνται με σχόλια στον κώδικα οπότε για συντομία λόγου δεν θα επαναληφθούν όλα στο παρών έγγραφο.

Η εργασία μου αποτελείται από τα πακέτα BST και others. Το πρώτο πακέτο, περιέχει τις κλάσεις bst, ThreadedBST, SortedArray και Console. Το πακέτο others, περιέχει τις κλάσεις Generator και MultiCounter οι οποίες είναι ίδιες με αυτές που χρησιμοποιήθηκαν και στην προηγούμενη εργασία

Ξεκινώντας, η κλάση bst είναι μία κλάση που υλοποιεί το δυαδικό δένδρο έρευνας. Στον κατασκευαστή, αρχικοποιείται η λίστα διαθέσιμων θέσεων στο array δύο διαστάσεων. Κατόπιν, η μέθοδος insert(int key) καλεί την μέθοδο inserthelp(int rt, int key) η οποία με χρήση της αναδρομής εισάγει κλειδιά στο δυαδικό δένδρο. Συνεχίζοντας, για την αναζήτηση τυχαίων κλειδιών χρησιμοποιείται η μέθοδος findhelp(int rt, int key), που καλείται από την find(int key), η οποία υλοποιείται επίσης με αναδρομή. Για την εκτύπωση εύρους τιμών χρησιμοποιούνται οι μέθοδοι printRange(int low, int high) και printRangehelp(int root, int low, int high) με το ίδιο σκεπτικό με πριν. Τέλος, έχουν γραφτεί και οι μέθοδοι για να υλοποιηθεί η Inorder εκτύπωση του δένδρου η οποία όμως χρησιμοποιήθηκε μόνο για τον έλεγχο της σωστής λειτουργίας της insert(int key) (δεν καλείται πουθενά).

Η κλάση ThreadedBST υλοποιεί το νηματοειδές ΔΔΕ. Η βασική διαφορά της από το απλό ΔΔΕ είναι ότι το array έχει 2 παραπάνω στήλες που αναπαριστούν τα νήματα και παίρνουν τις τιμές -2 αν είναι False (δεν δείχνει σε παιδί) και -3 αν είναι True (δείχνει σε παιδί). Αρχικά, αρχικοποιείται το array με -1 (null) στις θέσεις 0 και 1 όπου βρίσκονατι το κλειδί και το Left, -2 στις θέσεις 2 και 4 όπου βρίσκονται το left thread και το right thread αντίστοιχα και τέλος η θέση 3 του array αρχικοποιείται με τις διευθύνσεις όλων των διαθέσιμων θέσεων. Όπως

και στην κλάση bst, όλα τα ερωτήματα υλοποιούνται με μία μέθοδο που καλεί την αντίστοιχη μέθοδο help (εκτός από την printRange) ώστε να της δώσει στο όρισμα int rt το root της κλάσης. Στην inserthelp δεν χρησιμοποιείται αναδρομή αλλά while loop και if συνθήκες. Στο while loop βρίσκεται ο πατέρας του παιδιού που θα προσθέσουμε και στη συνέχεια με χρήση μίας από τις τρεις if (ανάλογα την συνθήκη), θα προστεθεί το παιδί. Συνεχίζοντας, η findhelp ακολουθάει το ίδιο σκεπτικό με την findhelp της κλάσης bst μόνο που χρησιμοποιεί while loop και όχι αναδρομή. Τέλος, για την αναζήτη εύρους τιμών, χρησιμοποιείται η μέθοδος printRange(int low, int high) η οποία καλεί την μέθοδο find με όρισμα low και μας γυρνάει την θέση του κλειδιού low αν υπάρχει ή το πιο κοντινό του αν δεν υπάρχει. Κατόπιν, εκτυπώνεται το κλειδί και καλείται η inorderSuccessor(int ptr) που μας δίνει το αμέσως μεγαλύτερο κλειδί στη σειρά.

Η τελική δομή που μας ζητείται να φτιάξουμε είναι ένας ταξινομημένος πίνακας. Η κλάση υπεύθυνη γι' αυτό είναι η SortedArray. Αρχικά, αρχικοποιείται στον κατασκευαστή της το sortedarray με τα ίδια 100000 κλειδιά που χρησιμοποιήθηκαν και για τις άλλες δομές και στη συνέχεια ταξινομούνται. Για την εύρεση τυχαίου κλειδιού, χρησιμοποιείται η μέθοδος binSearch(int leftEdge, int rightEdge, int key) όπου ψάχνει το κλειδί με χρήση while loop και μας γυρνάει την διεύθυνση του κλειδιού ή τη διεύθυνση του κοντινότερου αριθμού στο κλειδί. Η αναζήτηση εύρους τιμών, υλοποιείται με δύο μεθόδους, την findlow(int leftEdge, int right-Edge, int key) και printRange(int leftEdge, int rightEdge, int low, int high). Η πρώτη, καλεί την συνάρτηση binSearch για να βρεί την διεύθυνση του κλειδιού που βρίσκεται πιο κοντά και κατόπιν την αυξάνει μέχρι να βρεθεί το κλειδί που είναι αμέσως μεγαλύτερο από το low. Αυτή η μέθοδος καλείται αργότερα στην printRange και ύστερα από την διεύθυνση που γυρνάει εκτυπώνονται σειριακά τα κλειδιά μέχρι το high.

Τέλος, η κλάση Console περιέχει την main και δημιουργεί όλα τα απαραίτητα κλειδιά για δημιουργήσει τις δομές και να υλοποιήσει τα ερωτήματα. Επίσης, κάνει τις απαραίτητες μετρήσεις των συγκρίσεων χρησιμοποιώντας την κλάση MultiCounter.

Πηγές

Η πηγές που χρησιμοποιήθηκαν για την συγγραφή του κώδικά είναι οι πηγές που μας δόθηκαν και οι έτοιμοι κώδικες που μας προτάθηκαν να χρησιμοποιήσουμε.