

Estructures de dades: Pràctica 4 Laboratori

Arbres Binaris

Semestre primavera 2014/2015
Universitat de Barcelona

Objectiu

En aquesta pràctica es pretén que l'estudiant es familiaritzi amb estructures de dades no lineals. En particular, es treballarà amb diferents implementacions dels arbres binaris, analitzant les seves característiques i les seves diferències.

Indexació de paraules en un text

L'aplicació que es vol implementar és la indexació de paraules en un text. Donat un text qualsevol es pretén construir una estructura de dades que permeti cercar paraules en el text de forma eficient. Aquesta estructura haurà de mantenir informació sobre les posicions en les que surt cada paraula, la paraula en el text i permetre accedir a aquesta informació de manera ràpida. Per exemple, en el següent text:

- 1 Binary search tree
 - 2
 - 3 From Wikipedia, the free encyclopedia
 - 4
 - 5 In computer science, binary search trees (BST), sometimes called ordered or sorted binary trees, are a class of data structures used to implement lookup tables and dynamic sets. They store data items, known as keys, and allow fast insertion and deletion of such keys, as well as checking whether a key is present in a tree.
 - 6
 - 7 Binary search trees keep their keys in sorted order, so that lookup and other operations can use the principle of binary search: when looking for a key in a tree (or a place to insert a new key), they traverse the tree from root to leaf, making comparisons to keys stored in the nodes of the tree and deciding, based on the comparison, to continue searching in the left or right subtrees. On average, this means that each comparison allows the operations to skip over half of the tree, so that each lookup/insertion/deletion takes time proportional to the logarithm of the number of items stored in the tree. This is much better than the linear time required to find items by key in an unsorted array, but slower than the corresponding operations on hash tables.

Si es cerques per la paraula binary, s'haurien d'obtenir els següents resultats, tenint en compte que entre la primera i segona línia de text hi ha una línia en blanc:

- línia 1, paraula 1
- línia 5, paraula 4
- línia 5, paraula 13
- línia 7, paraula 1
- línia 7, paraula 21

En aquesta pràctica es demana implementar aquesta funcionalitat utilitzant arbres de cerca binaria.

Exercici 1: Arbre binari

Implementeu la classe **BTree** que representi l'arbre binari. Teniu en compte que cada node de l'arbre estarà representat per una paraula on hi guardarem una llista de tuples que contindrà la línia i posició de la paraula en el text. Únicament s'han d'implementar aquells mètodes que siguin necessaris pel desenvolupament de la pràctica.

Exercici 2: Cercador de paraules

Implementeu un cercador de paraules **BTreeWordFinder** que tingui un objecte de tipus **BTree**. Aquesta classe ha de tenir un mètode **main** que realitzi les següents operacions:

- Demanar a l'usuari el nom del fitxer que volem utilitzar. Al campus virtual trobareu dos fitxers de text que podeu utilitzar per fer les proves: *smallText.txt* i *largeText.txt*.
- Crear un **BTreeWordFinder**.
- Inicialitzar el **BTreeWordFinder** a partir del contingut del fitxer. Guardeu totes les paraules en minúscula.
- Llegir el fitxer *dictionary.txt* que us proporcionem per testejar l'aplicació.
- Per a cada paraula del fitxer anterior fer una cerca de la paraula en l'arbre emmagatzemat a **BTreeWordFinder**. S'ha d'avaluar el temps necessari per realitzar la cerca de totes les paraules del fitxer.
- Generar i visualitzar l'index de les paraules.
- Visualitzar la profunditat de l'arbre.

Els altres mètodes que s'han de definir són els següents:

- **appendText(filename)**: Aquest mètode rep el nom d'un fitxer i emmagatzema el seu contingut dins de l'arbre.
- **insertWord(word, line, position)**: Aquest mètode rep el nom d'una paraula, la línia i la posició on apareix en el text i fa la inserció a l'arbre. Teniu en compte que l'arbre no té elements repetits, en cas que ja existeixi haureu d'actualitzar el contingut del node.

- `findOccurrences(word)`: Aquest mètode rep una paraula i retorna una llista de tuples que conté les línies i posicions de la paraula en el text.
- `viewIndex()`: Aquest mostra per pantalla l'índex de paraules que surten a l'arbre en ordre alfabètic, junt amb les posicions on apareixen.

Generació d'un índex de paraules

Un índex de paraules és un llistat complet de les paraules que surten en un text en ordre alfabètic, junt amb les posicions a on hi surten.

Per exemple, pel text mostrat en la secció 2, l'índex començaria així:

- (bst), [(5, 7)]
- (or [(7, 31)]
- a [(5, 16), (5, 50), (5, 55), (7, 26), (7, 29), (7, 32), (7, 36)]
- allow [(5, 37)]
- allows [(7, 80)]
- an [(7, 124)]
- and [(5, 26), (5, 36), (5, 40), (7, 13), (7, 58)]
- ...
- binary [(1, 1), (5, 4), (5, 13), (7, 1), (7, 21)]
- ...

Exercici 3: Arbres binaris balancejats

Implementeu la classe **BalancedBTree** que representi l'arbre binari balancejat. La funcionalitat d'aquesta classe ha de ser exactament la mateixa que la classe **BTree** de l'exercici 1.

Exercici 4: Cercador de paraules amb arbres binaris balancejats.

Implementeu una nova versió del cercador de paraules **BalancedBTreeWordFinder**. En aquest cas la classe té un objecte de tipus **BalancedBTree**. Implementeu exactament els mateixos mètodes dels definits a la classe **BTreeWordFinder**.

Exercici 5: Avaluació de les estructures

Feu una avaluació del rendiment de les dues implementacions anteriors (arbres binaris i arbres binaris balancejats), i raoneu les diferències: compteu el temps d'accés (cerca) per dos textos de diferent tamany; compteu el temps de generació de l'estructura per dos textos de diferent tamany. Raoneu els resultats de temps obtinguts.

Format del lliurament

El lliurament consistirà en tot el codi generat en els diferents punts de l'enunciat, juntament amb la memòria (en format PDF) on heu d'explicar els resultats i les conclusions que obteniu de l'exercici 5.

En concret, cal generar un fitxer en format zip que contingui el nom i cognoms dels dos membres del grup (sense accents), el format del nom del fitxer serà el següent:

- Cognom1Cognom2Nom_Cognom1Cognom2Nom_P4.zip

La data de lliurament de la pràctica és com a molt tard el proper dia 12 de Maig del 2015 al campus virtual de la UB.