Estructures de Dades i Algorismes (EDA)

PER EL VECTOR TF-IDF

LIST < PAIR < Document, MAP < Integer , Double > > >

Per a representar un document de manera que poguem fer la Consulta per Semblança, hem decidit utilitzar el *vector model* per a cada document, emmagatzemat a la classe *Llibreria* (Conjunt de Documents). Això implica que un document està representat per un vector de tantes posicions com paraules hi ha al vocabulari, i on cada posició (paraula) té assignat un pes. Així, la relació semblança queda definida com el producte escalar de dos documents.

Tot i així, si ho pensem fredament, moltes posicions d'aquests vectors serà zero (doncs el document normalment només tindrà un subconjunt petit de paraules de tot el vocabulari). Ergo, són vectors *sparsed*. Per això hem utilitzat el MAP < Int, Double >, que mapeja una posició del vector a un valor. El cost mitjà d'inserció és constant, i el cost de look-up també; per tant es comporten com vectors.

A part d'això, una *Llibreria* és un conjunt de documents. Representem aquest conjunt com una llista de parells. Els parells estan ordenats a la llista per ordre d'inserció dels Documents. Els parells són d'una referència a document (left) i del HashMap que representa el vector TF-IDF que representa el document (right).

També hi ha una segona representació que, en compres d'utilitzar TF-IDF, utilitza com a pesos el nombre d'ocurrències de cada paraula al contingut del document.

TernaryTree

Per a la classe Vocabulari hem decidit utilitzar una estructura de dades per a indexar les paraules anomenat Ternary Search Tree. Aquest és un tipus d'arbre de prefixos que permet trobar un valor gairebé tan ràpid com un Trie, però utilitzant menys memòria. Podríem haver utilitzat un Set<Paraula>, que seria molt similar en temps d'execució i però utilitza més memòria.

TernaryTreeAutors

Per a la consultaAutors mitjançant els prefixos d'un autor, hem decidit utilitzar un arbre de prefixos. Hem utilitzat un Ternary Search Tree ja que és gairebé tan ràpid com un Trie, però utilitzant menys memòria. Aquest arbre permet fer una cerca dels autors mitjançant un prefix. Al cercar un prefix en qüestió, arribes a un node, que les fulles d'aquest sub-arbre serien els autors que contenen aquest prefix. Per a evitar

seguir fent una cerca per l'arbre una vegada trobat el prefix, hem afegit un conjunt d'autors a cada node.

ConsultaData

ArrayList<Document>

Aquest ArrayList s'utilitza per mantenir els documents donats d'alta en el sistema per ordre cronològic, és a dir per la seva data. Quan es vol fer la consulta i obtenir un conjunt de documents que compleixi una certa condició segons la seva data la ConsultaData fa dos recorreguts lineals en aquest ArrayList per obtenir els documents d'interés, un per obtenir el límit superior de l'*interval* (donat per la data mínima i la data màxima setejades com a paràmetre) i un altre per el límit inferior.

ConsultaTitol

HashMap<String, Set<Document>>

Aquesta estructura de dades ens permet desar diferents conjunts de documents. A la classe ConsultaTitol, cada clau del HashMap és l'autor, i el valor és conjunt de documents que pertanyen en aquest autor. El hashMap ens permet accedir i inserir valors de forma constant. La decisió d'utilitzar el Set<Document> envers altres estructures està justificada més abaix. Per a la consultaTitol podríem cercar per a tots els documents quins contenen l'autor en qüestió, però és massa lent.

ExpressionTree

Els atributs de la classe Expression Tree tenen l'objectiu de generar un arbre binari d'expressió a partir d'un string, aquesta generació consta de dues parts:

1- Interpretar el string I passar l'expresió booleana que ens arriva en infix a postfis mitjançant un algorisme "shunting yard", en aquesta part el stack fa la funció de emmagatzemar els operadors de manera que quan hem acabat la pila esta buida I la llista conté l'experssió sencera en postfix. També fem us del map ja que aquest emmagatzema el que ha de considerar l'algorisme shunting yard com a operador lògic per a diferenciar-los de la resta d'inputs que rep. Cada operador té assignat un valor de precedencia que ens ajuda a decidir l'ordre en el que han d'apareixen en l'expressió en postfix.

2- Generar un arbre binari d'expressio a partir de la llista de la meitat anterior.

ExpressioBooleana

La classe Expressió Booleana consisteix dels atribut presents per a facilitar el pont entre la creació de les expressions I la cerca propia. El controlador d'expressions booleanes guarda en un Map totes les expressions que s'han volgut guardar amb un nom fent servir aquest com a index per a facilitar la seva manipulació.

- nom: Aquest atribut es trivial, representa el nom que s'ha volgut assignar a una expressió booleana per a poder reusar-la en un futur

- cos: Representa el text que l'usuari escriu que es guarda com a string sense cap modificació I que s'interpretarà per a crear un arbre binari ExpressionTree

- root: Representa el node arrel de l'arbre binari mencionat. És generat en el moment en el que s'assigna un valor no null al cos de l'Expressió Booleana I es regenera cada vegada que aquest és modificat.

El motiu darrere d'emprar un arbre binari I no una altra estructura de dades és qie ens facilita l'implementació de la cerca. Els nodes fulla representen les paraules o sequències que es volen trobar I tornen un Set de documents que les conté cadascuna. D'aquesta manera, quan s'han resolt les cerques fulla només ens queden conjunts I operadors logics AND, OR I NOT, es a dir, nomes cal fer interseccions, unions o complementaris dels conjunts ja obtinguts. El node arrel es l'ultim operador logic que es resol I retorna el conjunt de documents que compleix totes les "subexpressions" de l'arbre.

- resultat: Aquest atribut també és trivial, es un Set de documents que conté el resultat de la cerca feta per root.

Altres:

Set<Document>

Per a algunes consultes hem decidit utilitzar un Set<Document> on conté un conjunt de documents. El hashSet té l'avantatge de inserir i cercar una clau de forma constant ja que no té els documents ordenats. Com que en aquestes consultes l'ordre el marca l'usuari, serà més endavant quan els ordenem.