Problemas de examen resueltos

Problemas de examen resueltos

Dados N libros que contienen un total de M palabras distintas, se va a utilizar un TDA palabras para manejar dicha información. Considérese la siguiente función:

```
list<int> buscar(const string&pal1,const string&pal2,int k) const;
```

La cual devuelve una lista de libros (identificados por un entero) que contienen las palabras pal1 y pal2 a una distancia k. Se pide:

- Define una representación adecuada de forma que la operación buscar tenga una eficiencia O(logM+ndoc), donde ndoc es el número de documentos en los que aparecen la palabra pal1 o la palabra pal2.
- Implementar dicha función.

Problemas de examen resueltos

Este problema es particularmente feo porque todas las aproximaciones que se nos ocurren van a depender de: buscar las palabras (O(logM)), recorrer los distintos documentos en los que aparecen (O(ndoc)) y recorrer las distintas apariciones de cada palabra en cada documento (O(npal)).

Lo más probable la primera vez que lo abordamos es que nos salga O(logM+ndoc*npal), que es mayor que O(logM+ndoc).

El truco está en aprovechar el hecho de que tenemos **LIBERTAD** para escoger cualquier representación. Escogeremos una representación que, aunque sea muy ineficiente en cuanto a memoria, será muy eficiente para realizar esta búsqueda en particular.

Problemas de examen resueltos

```
map< string, map< string, vector<int> > > bd;

 //pal1 -> (pal2 -> (k -> ints id de libro)).
   map      map      vector

list<int> buscar(const string&pal1,const string&pal2,int k) const{
  return bd[pal1][pal2][k];
  //Busca  map   map   vector
}
```

$T=\log(M^2)+1$

$T=2\log(M)+1$

$O(\log(M))$