GSI024: Organização e Recuperação de Informação 1º semestre, 2016

Prática 4 — Construção de Índice Invertido

Prof. Marcelo Keese Albertini *Universidade Federal de Uberlândia*

Um índice invertido pode ser considerado a principal estrutura de dados utilizada por um buscador. Essa estrutura relaciona cada termo de busca com uma lista de documentos em que aparece.

Objetivo:

- Construir arquivos binários com referências e listas de `docIds` , que compõem um índice invertido. Esses arquivos serão usados para responder às buscas realizadas por usuários.

Entradas esperadas:

- Programa deve processar um arquivo binário contendo uma lista ordenada por `termId` de pares de números inteiros (`termId` , `docId`), criado conforme a prática 3. Esse arquivo pode ser obtido em http://www.facom.ufu.br/~albertini/ori/pares.dat

Saídas produzidas:

- Programa deve escrever as listas de `docIds` de cada `termId` em um arquivo binário chamado de `postings.dat`
  - Por exemplo, se no arquivo `pares.dat` tiver os seguintes pares: (0,1), (0,2), (0,2), (0,4), (1, 1), (1, 3) então em `postings.dat` serão escritos os números inteiros dos `docIds` 1, 2 e 4 referentes ao `termId` 0 e os números inteiros dos `docIds` 1 e 3 referentes ao `termId` 1. Note que repetições de `docIds` para um mesmo `termId` devem ser ignoradas nesta prática.
- Em um arquivo binário chamado `index.dat`, o seu programa deve escrever, para cada `termId` , o número inteiro que indica a posição em que começa a sua lista de `docIds` no arquivo `postings.dat`
  - Seguindo o exemplo anterior, no arquivo `index.dat` será escrito o número inteiro 0 referente à posição do byte em que a lista de `docIds` do `termId` 0 inicia em `postings.dat` e o número inteiro 3 referente à posição do byte em que a lista de `docIds` do `termId` 1 começa em `postings.dat`.

Habilidades praticadas:

1. Usar DataInputStream para leitura sequencial de arquivo binário:

```
DataInputStream dis = new DataInputStream(new FileInputStream("pares.dat"));
while (dis.available() > 0) {termId = dis.readInt(); docId = dis.readInt();}
```

2. Usar DataOutputStream para criar arquivos binários:

- ```
  DataOutputStream index =
  new DataOutputStream(new BufferedOutputStream(new FileOutputStream("index.dat")));
  ```
  ```