Universidade Federal do Amazonas

Banco de Dados I – 2020/2

Profº Altigran Soares da Silva

Trabalho Prático II – Sistema de Arquivos

Equipe:

- Davi Ricardo - 21602648
- Marcos Guerreiro - 21555273
- Rógenis Silva - 21650332

1. Introdução

O objetivo deste trabalho é o de exercitar alguns conceitos de organização e indexação de arquivos através da implementação de um arquivo de dados que parte da inserção de uma massa de dados como entrada. Estes dados ou registros serão inseridos de forma unitária, ou seja, registro a registro em um arquivo organizado por hash. A medida que o arquivo de dados é criado, o endereço de cada registro será usado para a criação de uma árvore B+ para fazer a indexação primária dos registros. A busca de dados poderá ser realizada tanto usando técnicas de indexação sobre um arquivo de índices primários ou um arquivo de índices secundários, indexados por uma a árvore B+ de índice primário e uma árvore B+ de indice secundário, respectivamente, quanto sobre a função hash. O objetivo deste documento é o de apresentar uma breve descrição dos dados (seção 2), apresentar a estrutura do arquivo de dados organizado por hash (seção 3) e dos arquivos de índices (seção 4), informar a organização dos códigos (seção 5), descrever as funções mais relevantes (seção 6) e a explanar a distribuição do trabalho entre os integrantes (seção 7).

2. Descrição dos Dados

Os dados serão extraídos de um arquivo de entrada específico, o artigo.csv. Estes dados são uma série de registros, que apresentam, na maioria dos casos, a seguinte estrutura:

- Id: Código identificador do artigo
- Título: Título do artigo

- Ano: Ano de publicação do artigo
- Autores: Lista dos autores do artigo
- Citações: Número de vezes que o artigo foi citado
- Atualização: Data e hora da última atualização dos dados
- Snippet: Resumo textual dos dados do artigo.

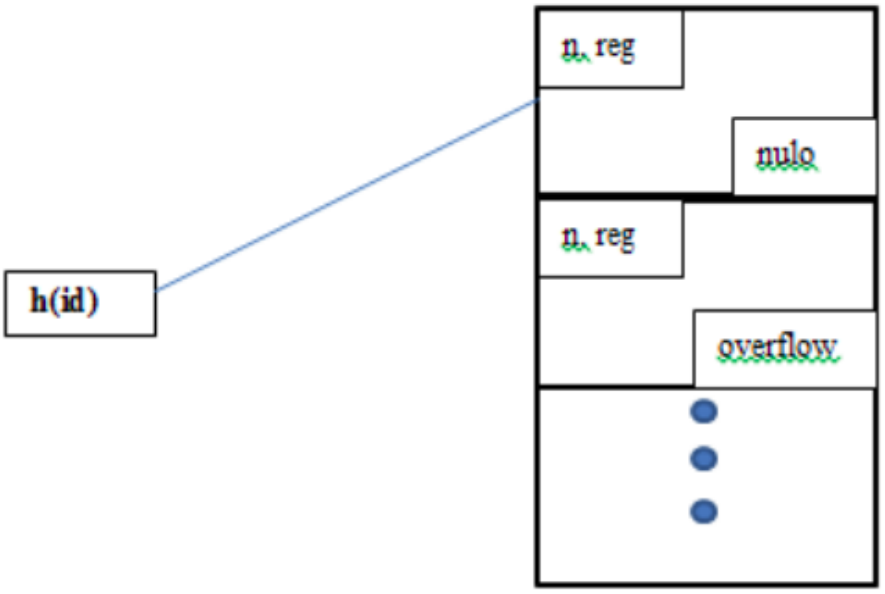
É válido mencionar que os registros nem sempre apresentam esta estrutura, pois há registros terminados em NULL indicando que todos os campos em diante não existem. Os campos Autores e Snippet foram truncados para apenas 100 caracteres alfanuméricos.

3. Estrutura dos Arquivos Organizado por Hash

Um dos arquivos deste trabalho está organizado por um hash de overflow, portanto é necessário explicar a estrutura do arquivo de registros organizados por buckets, este arquivo será chamado de arquivo de dados, e a estrutura do arquivo que irá conter eventuais registros cujos buckets excederam a quantidade de registros, este arquivo será chamado de arquivo de overflow. A estrutura que armazenará um registro qualquer possui apenas os campos necessários para representar os dados apresentados na seção 2 e será usada constantemente por outras estruturas e funções.

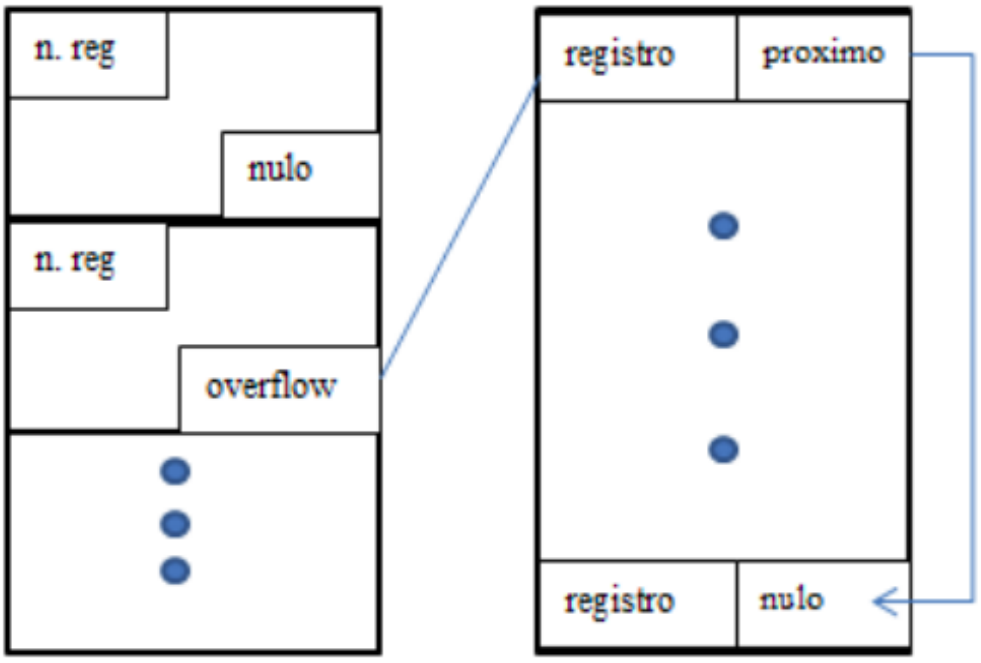
3.1. Estrutura do Arquivo de Dados

O arquivo de dados é dividido em 51.200 buckets referenciados pelo hash, na qual cada bucket possui 2 blocos de 4096 bytes. Cada bloco possui um vetor de registros de tamanho fixo que são escritos de forma não espalhada. Além disso, cada bloco possui um campo que armazena o número de registros inseridos e um campo que armazena o local de um registro, se este existir, que pertence ao mesmo bucket no arquivo de overflow, levando em consideração estes campos há aproximadamente 400 bytes de desperdício para cada bloco. Segue uma ilustração aproximada da estrutura do arquivo.



3.2. Estrutura do Arquivo de Overflow

O arquivo de overflow é formado por registros diferentes, os registros de overflow. Cada registro de overflow é uma estrutura que possui apenas um registro contendo os dados desejados e um campo para armazenar o local do próximo registro, gerando então um encadeamento, se existir algum registro a ser encadeado. Segue uma ilustração aproximada da estrutura do arquivo.

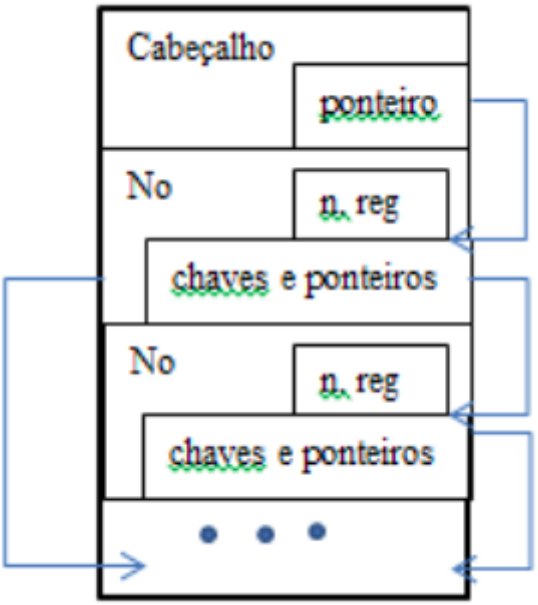


4. Estrutura dos Arquivos de Índices

A estrutura das duas árvores são semelhantes, entretanto as chaves diferem, logo são gerados dois arquivos, o arquivo de índice primário para o campo id e o arquivo de índice secundário para o campo título.

4.1. Estrutura do Arquivo de Índice Primário

Os primeiros 4096 bytes são reservados para o cabeçalho que contém a altura da árvore e o ponteiro para raiz. Os demais blocos representam os nós, externos ou internos, da árvore B+ que são estruturas que possuem um campo que indica o número de registros, um vetor de pares de ponteiro e chave e um último ponteiro visto que o número de ponteiros é o número de chaves somado em um. Segue uma ilustração aproximada da estrutura do arquivo.



4.2. Estrutura do Arquivo de Índice Secundário

O arquivo de índice secundário segue a mesma estrutura do arquivo de índice primário. A única diferença está no valor das chaves que é o campo título e não o campo id.

5. Organização dos Códigos

Os códigos estão divididos primariamente entre bibliotecas no diretório headers e o código fonte no diretório sources. Todas as funções possuem algumas linhas de comentários para uma breve explicação. Neste documento iremos abordar apenas as funções requisitadas pelo trabalho. Basta executar o comando make para compilação e geração dos diretórios e os comandos segundo a especificação do trabalho.

6. Descrição das Funções

As funções mais relevantes ao escopo do trabalho serão descritas em detalhes nas subseções seguintes.

6.1. upload_file

A função toma como parâmetro o caminho do arquivo de entrada, a partir disto são criados arquivos de dados, de overflow, de índice primário e de índice secundário. Cada registro do arquivo de entrada será lido, formatado e armazenado na estrutura de registro descrita na seção 3 para que possa ser inserida no arquivo de dados organizado pelo hash.

A função de inserção no hash irá armazenar o registro no arquivo de dados se possível, mas no caso de um bucket estar cheio haverá a escrita deste registro no arquivo de overflow. Toda tentativa de inserção do registro no bloco irá verificar primeiramente o número de registros no bloco, se o valor for igual ao fator de bloco, então a função irá tentar inserir no próximo bloco do bucket até que seja necessária a escrita no arquivo de overflow.

Esta função irá retornar o endereço do registro e indicará em qual arquivo este registro está, se está no arquivo de dados ou de overflow, tais informações servirão como parâmetro para a criação da B+ de índice primário.

6.2. find_record_datafile

A função requer apenas o valor do id de um registro provido pelo usuário para que haja a busca do registro desejado no arquivo de dados ou, eventualmente, no de overflow. A busca começará com a leitura do primeiro bloco do bucket de endereço retornado pela função hash, com o bloco em memória a busca pelo id será sequencial. A busca continuará para o segundo e último bloco do bucket se o

registro não for encontrado no primeiro bloco, sendo então necessário outro carregamento do bloco para a memória para buscar sequencialmente.

Caso o registro não seja encontrado no segundo bloco, verifica-se o valor do campo que armazena o endereço do próximo registro do bucket no arquivo de overflow, se o valor for válido haverá uma busca sequencial no arquivo de overflow na qual serão carregados apenas registros, de forma unitária, e não blocos.

7. Divisão do Trabalho

Segue a divisão do trabalho entre os integrantes:

- Davi Ricardo:
    1. Documentação e relatório
    2. Arquivo organizado em hash (hash de overflow)
- Marcos Guerreiro
    1. Busca de um registro pelo hash dado um id (findrec)
    2. Estrutura e organização dos arquivos e makefile
- Rógenis Silva
    1. Implementação das funções (upload)
    2. Parser B+ de índice primário