INSTITUTO DE CIÊNCIA E TECNOLOGIA - ICT

BACHARELADO EM CIÊNCIA E TECNOLOGIA

ALGORITMOS E ESTRUTURAS DE DADOS II

TRABALHO 2 – RECUPERAÇÃO DE IMAGENS USANDO ÁRVORES B

Aluno: Pedro Spoljaric Gomes – R.A.: 112344

Orientador: Profª Drª Lilian Berton

São José dos Campos

10 de maio de 2017

1. INTRODUÇÃO

O trabalho consiste em criar um repositório de 100 imagens divididas em 20 categorias, ou seja, 5 imagens de cada categoria. Além disso, cada imagem deve ser salva com o nome de sua categoria e um índice de 1 a 5. Exemplo: gato1, gato2, gato3, gato4 e gato5. A partir desse repositório, o objetivo é implementar uma árvore B que organize esses arquivos e facilite o acesso a eles.

2. MATERIAIS E MÉTODOS

Escolhi as seguintes categorias para fazerem parte do desenvolvimento do algoritmo:

- Árvore;
- Biscoito;
- Cachoeira;
- Caranguejo;
- Castelo;
- Cavalo;
- Coelho;
- Coruja;
- Cupcake;
- Estrela;
- Flor;
- Gavião;
- Joaninha;
- Leão;
- Nemo;
- Onda;
- Pato;
- Ponte;
- Sol;
- Zebra.

A estrutura de dados que escolhi para organizar as imagens foi a árvore B tradicional com fator T = 4, que atende às seguintes propriedades:

1. Cada nó, exceto a raiz, contém no mínimo T-1(3) itens e T(4) ramificações e no máximo 2*T-1(7) itens e 2*T(8) ramificações.
2. Cada item possui uma chave maior que as chaves de todos os itens à sua esquerda e menor que as dos itens à sua direita.
3. Cada item possui uma chave maior que todas as chaves dos itens da página imediatamente à sua esquerda e das páginas subsequentes.
4. Cada item possui uma chave menor que todas as chaves dos itens da página imediatamente à sua direita e das páginas subsequentes.
5. Toda folha tem a mesma profundidade, que é a altura da árvore, e páginas nulas.

Utilizei as seguintes structs para manipular os dados da árvore:

```c
typedef struct TItem
{
    char id[20];
}TItem;
```

```c
typedef struct TNo
{
    TItem item[2*T-1];
    struct TNo *pagina[2*T];
    int nInfo;
}TNo;
```

```c
typedef struct TArvB
{
    TNo *raiz;
    int altura;
}TArvB;
```

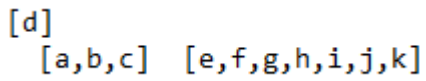
2.1 MÉTODOS

I. Inserção

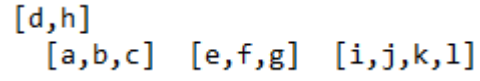
O método de inserção percorre a árvore até encontrar um nó folha para inserir o item, obedecendo as propriedades 2, 3 e 4 da árvore. Antes de acessar cada nó, é verificada a quantidade de itens nele e, se esta for igual ao máximo de itens permitido (2*T-1=7), o nó é dividido pela metade e o item do meio “sobe” para o nó pai (a árvore aumenta de tamanho caso a raiz esteja totalmente preenchida). Desse modo, sempre que o nó folha for encontrado, ele terá espaço disponível para a inserção.

Exemplo: Inserção da chave ‘l’ (L)

- Antes da inserção



- Depois da inserção



II. Remoção

O método de remoção percorre a árvore até encontrar o item. Nesse processo, antes de acessar cada página, caso ela possua somente o mínimo de itens (T-1=3), é feito com que ela receba pelo menos mais um item das seguintes formas (assim, quando o item for encontrado, não haverá problemas de remover um item de um nó que já tem o número mínimo de itens):

1. Se essa página for a página mais à direita do nó e
   a) A página imediatamente à sua esquerda possuir pelo menos T(4) itens:
      - Todos os itens e subpáginas da página à direita são deslocados uma posição para a direita, e o primeiro item dessa página copia o item que divide as duas páginas;
      - O item que divide as duas páginas copia o último item da página à esquerda;
      - A primeira subpágina da página à direita passa a apontar para a última subpágina da página à esquerda;
      - Os últimos item e subpágina da página à esquerda são removidos.
   b) A página imediatamente à sua esquerda também tem T-1(3) itens:
      - O item que divide as duas páginas ‘desce’ e une as duas páginas em apenas uma, mantendo todas as subpáginas em suas devidas posições;
      - A página mais à direita do nó se torna essa nova página.

2. Se essa página foi qualquer outra página do nó e
    a) A página imediatamente à sua direita possuir pelo menos T(4) itens:
        - Um item é adicionado no final da página à esquerda, copiando o item que divide as duas páginas;
        - O item que divide as duas páginas copia o primeiro item da página à direita;
        - A última subpágina da página à esquerda passa a apontar para a primeira subpágina da página à direita;
        - Todos os itens e subpáginas da página à direita são deslocados uma posição para a esquerda, removendo os primeiros item e subpágina.
    b) A página imediatamente à sua direita também tem T-1(3) itens:
        - O item que divide as duas páginas ‘desce’ e une as duas páginas em apenas uma, mantendo todas as subpáginas em suas devidas posições;
        - A página que estava à esquerda do item que ‘desceu’ se torna essa nova página e todos os itens e páginas que estavam à direita desse item são deslocados uma posição para a esquerda.

Após encontrar o item a ser removido, se o nó em que este item estiver for

1. Uma folha:
    - O item é removido e todos os itens à sua direita são deslocados uma posição para a esquerda.
2. Um nó interno e
    a) A página imediatamente à esquerda do item possui pelo menos T(4) itens:
        - O último item dessa página ‘sobe’, ocupando o lugar do item removido e o problema é passado recursivamente para o item que foi removido dessa página.
    b) A página imediatamente à direita do item possui pelo menos T(4) itens:
        - O primeiro item dessa página ‘sobe’, ocupando o lugar do item removido e o problema é passado recursivamente para o item que foi removido dessa página.

c) As duas páginas adjacentes possuem T-1(3) itens:
    - O item a ser removido ‘desce’, unindo essas duas páginas em uma e mantendo todas as subpáginas em suas devidas posições;
    - A página que estava à esquerda do item que ‘desceu’ se torna essa nova página e todos os itens e páginas que estavam à direita desse item são deslocados uma posição para a esquerda;
    - O problema é passado para esse item que desceu.

Exemplo 1: Remoção da chave ‘w’

- Antes da remoção

```
[d,h,l,p,t]
  [a,b,c]  [e,f,g]  [i,j,k]  [m,n,o]  [q,r,s]  [u,v,w,x,y,z]
```

- Depois da remoção

```
[d,h,l,p,t]
  [a,b,c]  [e,f,g]  [i,j,k]  [m,n,o]  [q,r,s]  [u,v,x,y,z]
```

Exemplo 2: Remoção da chave ‘t’

- Antes da remoção

```
[d,h,l,p,t]
  [a,b,c]  [e,f,g]  [i,j,k]  [m,n,o]  [q,r,s]  [u,v,x,y,z]
```

- Depois da remoção

```
[d,h,l,p,u]
  [a,b,c]  [e,f,g]  [i,j,k]  [m,n,o]  [q,r,s]  [v,x,y,z]
```

Exemplo 3: Remoção da chave ‘p’

- Antes da remoção

```
[d,h,l,p,u]
  [a,b,c]  [e,f,g]  [i,j,k]  [m,n,o]  [q,r,s]  [v,x,y,z]
```

- Depois da remoção

```
[d,h,l,u]
  [a,b,c]  [e,f,g]  [i,j,k]  [m,n,o,q,r,s]  [v,x,y,z]
```

III. Busca

O método de busca retorna todos os itens na árvore com rótulos que se iniciam com um dado texto.

Exemplo: Busca de ‘z’

- Árvore

```
[d,zc]
  [a,b,c]  [e,f,g,z,za,zb]  [zd,ze,zf,zg]
```

- Retorno

```
z
za
zb
zc
zd
ze
zf
zg
```

IV. Impressão

O método de impressão imprime todos os elementos da árvore organizadamente da seguinte forma:

Exemplo:

```
[p]
  [d,h,l]
    [a,b,c]    [e,f,g]    [i,j,k]    [m,n,o]
  [t,x,zb]
    [q,r,s]    [u,v,w]    [y,z,za]    [zc,zd,ze,zf,zg]
```

Onde os espaços no início de cada linha são proporcionais à profundidade dos nós nessa linha.

V. Exibição

O método exibir abre a imagem especificada no visualizador padrão do sistema operacional.

```
Inserção(1) Busca(2) Impressão(3) Remoção(4) Exibir(5)
2 arvore
arvore1.jpg
arvore2.jpg
arvore3.jpg
arvore4.jpg
arvore5.jpg
Inserção(1) Busca(2) Impressão(3) Remoção(4) Exibir(5)
5 arvore3
_imagens/arvore3.jpg
```

3. CONCLUSÃO

O desenvolvimento desse trabalho permitiu que eu aprendesse a fundo os conceitos de árvores B, precisando implementar métodos complicados de inserção e remoção a partir da teoria aprendida em aula. A maior dificuldade que encontrei foi em implementar o método de remoção possuindo apenas conhecimento de suas regras.

4. REFERÊNCIAS

BERTON, L., slides de aula. Primeiro semestre de 2017.