Universidade Federal de Ouro Preto – UFOP
Instituto de Ciências Exatas e Biológicas – ICEB
Departamento de Computação – DECOM
Disciplina: Algoritmos e Estrutura de Dados I – CIC102 / 20092
Professor: David Menotti (menottid@gmail.com)

Trabalho Prático 6

Árvores Binárias de Pesquisa/Busca
Algoritmo de *Huffman* aplicado à Compactação de Arquivos

Valor: 0,5 pts (5% da nota total) **Data de entrega:**
Documentação não-Latex: -0,1 pts **09/12/2009**
Impressão não frente-verso: -0,05 pts (ecologicamente correto)
Interface gráfica: +0,3 pts

Objetivos

O objetivo deste trabalho é utilizar o algoritmo de compactação de mensagens de *Huffman* para criar arquivos binários (ou texto) compactados.

Descrição

O algoritmo de *Huffman* sugere um esquema de codificação para o alfabeto de uma mensagem em função da frequência de cada símbolo. Essa codificação é representada através de uma árvore binária. Cabe a você pesquisar a literatura e entender a forma de criar essa árvore e codificar a mensagem, apesar de você pode inserir em seu trabalho prático implementações já existentes na literatura para o algoritmo de *Huffman*.

O que deve ser implementado

Uma vez entendido o algoritmo de *Huffman*, a mensagem (arquivo binário/texto) deverá ser codificada e armazenada em arquivo. No entanto, antes é recomendado que a árvore que gera a codificação seja armazenada no cabeçalho do arquivo, para posterior descompactação. Lembre-se que o tamanho da árvore é variável – ele depende do tamanho do alfabeto. Cabe a você desenvolver o TAD para esta árvore e a forma de armazenamento. Ressalta-se que essa é a parte principal do trabalho, e cópias serão severamente punidas, não somente zerando o valor deste trabalho, mas também eliminando a maior nota dos testes teóricos. Você também deverá implementar uma função para descompactar o arquivo. Com essa segunda função implementada, você poderá facilmente testar se o seu trabalho prático está funcionando perfeitamente.

O seu programa deverá funcionar em linha de comando. Caso você implemente uma interface gráfica, similar a Winzip, você poderá ganhar até 0,3 pontos extras, dependendo da qualidade da interface.

Deseja-se também que o tamanho dos arquivos compactados pelo algoritmo de Huffman sejam comparados com os gerados por compactadores bem conhecimentos como ZIP e RAR, TAR. Uma análise dessa comparação deve ser feita.

Tipos e Tamanhos de Arquivos a serem utilizados

Procure utilizar arquivos texto, imagens (em diferentes formatos, com e sem compactação), vídeos,

áudios, de sistema (.dll, .exe, etc), etc. Escolha arquivos de diversos tamanhos variando desde 1 Kilobytes até 100 Megabytes.

A implementação do algoritmo de Huffman

Para este trabalho, está sendo indicado o uso da implementação do algoritmo de *Huffman* implementado em [1,2]. Todavia, outros códigos podem utilizados. Independente da fonte, o código do algoritmo de Huffman utilizado no trabalho prático deve ser apresentado na documentação. Eventuais adaptações e correções no código são objetivo de avaliação deste trabalho. Portanto "*débrouille-toi*"!

O que deve ser entregue

- Código fonte do programa em C ou C++ (bem identada e comentada).
- Documentação do trabalho. Entre outras coisas, a documentação deve conter:
  1. Introdução: descrição do problema a ser resolvido e visão geral sobre o funcionamento do programa.
  2. Implementação: descrição sobre a implementação dos algoritmos, incluíndo o algoritmo de Huffman. Deve ser detalhada a estrutura de dados utilizada (de preferência com diagramas ilustrativos), o funcionamento dos algoritmos, e decisões tomadas relativas aos detalhes de especificação que porventura estejam omissos no enunciado. Muito importante: os códigos utilizados na implementação devem ser inseridos na documentação.
  3. Análise dos testes: analisar os tamanhos dos arquivos compactados a partir dos experimentos realizados.
  4. Conclusão: comentários gerais sobre quais algoritmos são os mais eficientes dados vetores particulares.
  5. Bibliografia: bibliografia utilizada para o desenvolvimento do trabalho, incluindo sites da Internet se for o caso. Uma referência bibliográfica deve ser citada no texto quando da sua utilização
  6. Em Latex: Caso o trabalho não seja elaborado/escrito em latex, perde-se 0,1 pontos.
  7. Impressão: Caso o trabalho não seja impresso usando modo frente-verso, perde-se 0,05 pontos.
  8. Formato: mandatoriamente em PDF (http://www.pdf995.com/).

  Obs.: Veja modelo de como fazer o trabalho em latex:
  http://www.decom.ufop.br/prof/menotti/aedI092/tps/modelo.zip

Como deve ser feita a entrega:

A entrega DEVE ser feita via Moodle (www.decom.ufop.br/moodle) na forma de um **único** arquivo zipado, contendo o código, os arquivos e a documentação. Também deve ser entregue a documentação impressa na próxima aula teórica após a data de entrega do trabalho.

Comentários Gerais:

- Clareza, identação e comentários no programa também vão valer pontos;
- O trabalho é individual (grupo de UM aluno);

- Trabalhos copiados (e FONTE) terão nota zero;
- Trabalhos entregues em atraso serão aceitos, todavia a nota atribuída ao trabalho será zero;
- Evite discussões inócuas com o professor em tentar postergar a data de entrega do referido trabalho.

Referências:

[1] N. Ziviani, F.C. Botelho, Projeto de Algoritmos com implementações em Java e C++, Editora Thomson, 2006.

[2] Aaron M. Tenenbaum, Yedidyah Langsam, Moshe J. Augenstein, *Estruturas de Dados Usando C*, Makron Books/Pearson Education, 1995.