Implementação de um codificador/decodificador baseado no algoritmo de Huffman aplicado à compactação de arquivos

Objetivos gerais

O objetivo deste EP é implementar um codificador/decodificador baseado no 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.

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.

O que deve ser entregue

• Código fonte do programa em C (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.

Revisão sobre código de Huffman:

Para enviar uma mensagem através de um cabo de transmissão, os caracteres da mensagem são enviados um a um, modificados através de alguma tabela de codificação. Em geral, este código forma um numero binário. Como a velocidade de transmissão é importante, e interessante tornar a mensagem tão curta quanto possível sem perder a capacidade de decodificar o texto enviado. Um algoritmo de determinação de códigos binários de comprimentos variados para caracteres (não e o caso que cada caractere seja representado com o mesmo numero de bits), baseado na frequência de uso destes caracteres foi desenvolvido em 1952 por David A. Huffman que era, na época, estudante de doutorado no MIT. A ideia é associar números binários com menos bits aos caracteres mais usados num texto. Desta maneira espera se que o numero de bits economizados para codificar os caracteres que ocorrem com maior frequência em um texto seja mais que o suficiente para cobrir o déficit que ocorrerão ao codificamos os caracteres que ocorrem raramente com cadeias longas de bits.

O algoritmo de Huffman que será implementado para codificar um arquivo consistira de três fases:

- Primeira etapa:

A frequência de cada caractere que ocorre no texto devera ser calculada.

- Segunda etapa:

Consiste em construir uma arvore binária baseada na frequência de uso destas letras de modo que as que aparecem com mais frequência fiquem mais perto da raiz do que as que aparecem com menos frequência. Este tipo de arvore é denominada arvore binária de Huffman.

Em cada passo desta fase tem-se teremos uma coleção de arvores binárias, ou seja, uma “floresta” formada por arvores binárias. As folhas de cada uma destas arvores correspondem a um conjunto de caracteres que ocorrem no texto. A raiz de cada uma destas arvores será associado um numero que corresponde à frequência com que os caracteres associados às folhas desta arvore ocorrem no texto. Escolherem os então as duas arvores desta floresta com a menor freqüência e as transformarem os em uma única arvore, ligando-as a uma nova raiz cujo valor será dado pela soma dos valores das frequências das duas sub arvores.

- Terceira etapa:

Nesta fase, a arvore de Huffman será usada para codificar e decodificar o texto.

A arvore de Huffamn construída durante a codificação será usada na decodificação (ela será armazenada junto com o texto que foi codificado).

CONSTRUÇÃO DE UMA ARVORE DE HUFFMAN:

Suponha uma mensagem composta pelos caracteres A, B, C, D, E, R e que a frequência de cada letra na mensagem seja respectivamente 22, 9, 10, 12, 16, 8.

A construção dessa arvore é recursiva, a partir da junção dos dois símbolos de menor probabilidade, ou seja, a construção é realizada de baixo para cima (das folhas para a raiz), tendo inicio com as letras de menor frequência até as de maior frequência, atingindo deste modo a raiz. As letra são representadas pelas folhas das arvores e seus vertices internos irão conter um número equivalente a soma das frequências de seus descendentes. Assim, o caractere de maior frequência irá possuir o código de menor comprimento que é exatamente o objetivo da compressão realizado pelo algoritmo de Huffman. O mesmo acontece com os demais caracteres, em ordem proporcionalmente crescente do número de bits, conforme a seguinte propriedade:

O tamanho médio dos caracteres (TMC) é dado pelo somatório dos produtos do tamanho de cada caracter (T(c)) pela sua probabilidade de limitações (P(c)). O mesmo resultado pode ser obtido pelo simples somatório das frequências de cada nó interno na árvore de Huffman).

	A	B	C	D	E	R
Frequência	22	9	10	12	16	8

Fazendo uso dos caracteres e dos seus respectivos valores de freqüência da tabela acima, demonstraremos graficamente a construção de uma arvore de Huffman.

Inicialmente cada um dos caracteres será uma árvore composta apenas pela raiz e cujo conteúdo é a frequência com que o caractere aparece na mensagem a ser codificada (veja Figura 1).

Figura 1

Em cada etapa do algoritmo existe um conjunto de arvores de Huffman. São escolhidos os dois caracteres com menor frequência (R e B) que formarão uma

arvore cuja raiz será correspondente a soma dos valores de frequência dos dois caracteres (8 + 9 = 17).

Escolhendo-se as duas raízes de menor frequência (arvores com R e B com frequências 8 e 9 , respectivamente) e juntando-as, essas duas arvores formarão uma nova arvore (veja Figura 2).

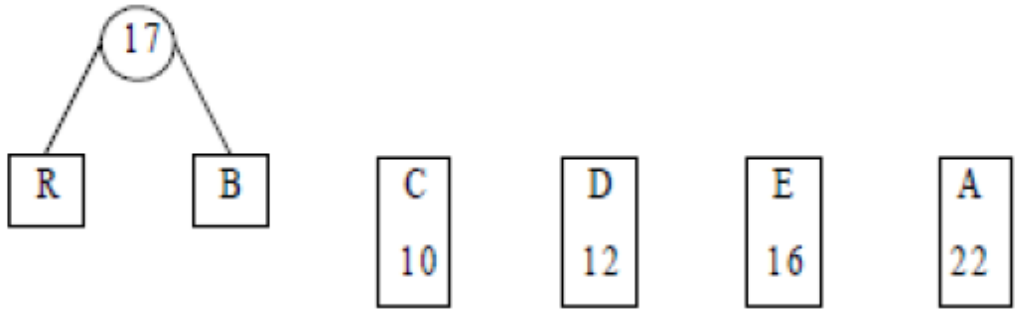

Figura 2

Repetindo processo anterior, escolhem-se as arvores seguintes com raízes de menor frequência (arvores C e D com frequências 10 e 12, respectivamente). Essas raízes são unidas, formando uma nova arvore, onde a raiz é, novamente, a soma de suas respectivas frequências (10 + 12 = 22) (Figura 3).

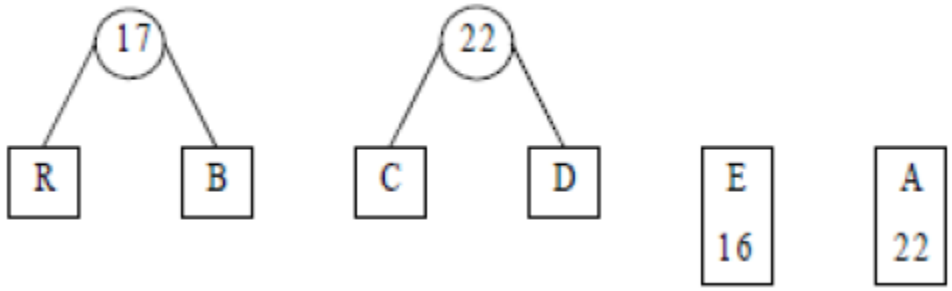

Figura 3

Dando continuidade ao processo, escolhem-se as arvores com frequências 16 (E) e 17, formando a arvore da Figura 4.

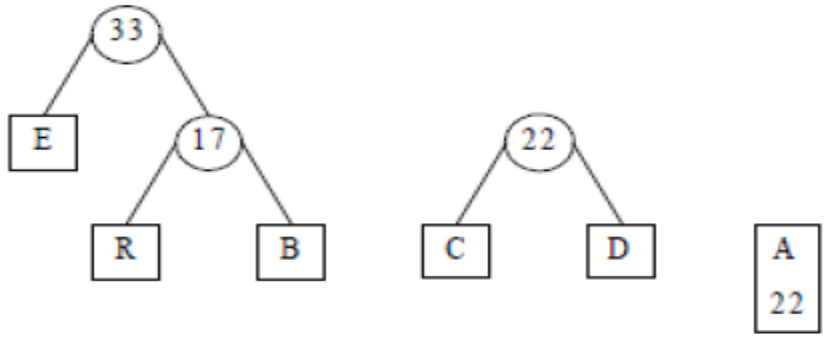

Figura 4

Unindo-se as arvores com freqüência 22 (A) e 22, obtem-se a arvore da Figura 5.

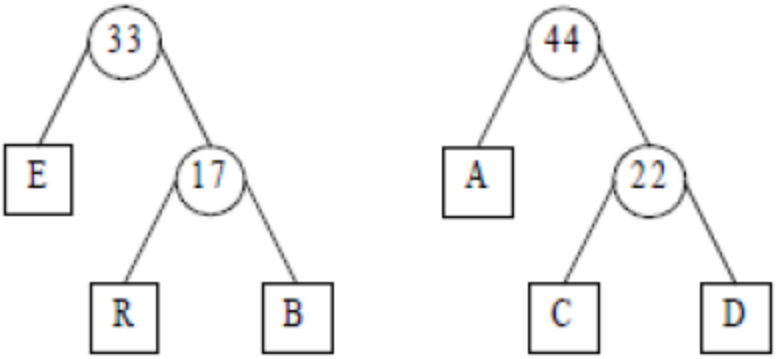

Figura 5

Finalmente, juntando as duas arvores restantes, se obtém a arvore da Figura 6. Deste modo, arvore de Huffman esta completamente construída. Lembrando sempre que o processo de construção de uma arvore de Huffman acompanha os valores de freqüência das raízes em ordem crescente, ou seja, da menor freqüência para a maior.

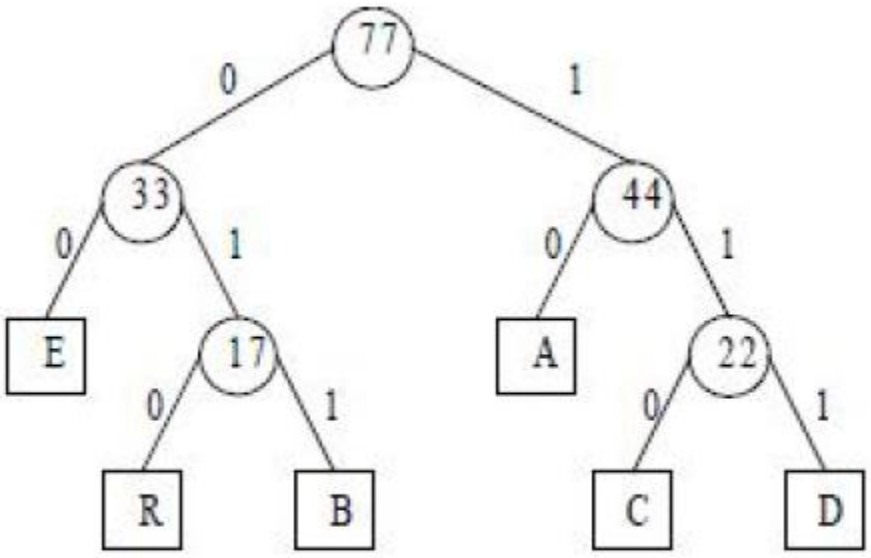

Figura 6. Árvore de Huffman.

CODIFICAÇÃO:

Associando 0 as arestas que ligam um nó pai com seu filho esquerdo e 1 as arestas que ligam um nó pai com seu filho direito. O código correspondente a cada

caractere será formado pelo numero binário associado ao caminho da raiz até a folha correspondente (onde esta o caractere a ser codificado).
Usando a arvore de Huffamn da figura 6 como exemplo, obtemos a seguinte tabela de códigos:

	A	B	C	D	E	R
Frequência	10	011	110	111	00	010

È interessante observar que na codificação alguns caracteres diminuem de tamanho (menos de 8 bits) enquanto outros, ao contrário, podem aumentar seu tamanho (até 256 bits). Em casos reais, a compressão obtida pode ser bem maior pois a frequência de alguns símbolos é consideravelmente grande (exemplo, as vogais "a", "e", "i", "o" e "u" ocorrem com bastante frequência) enquanto a de outros é quase nula(exemplo, letras como "y", "k" ou "z" aparecem com uma frequência muito menor).
Essa diferença faz com que os caracteres mais freqüentes, e conseqüentemente serão representados por símbolos mais curtos, aumentando bastante a taxa de compressão.

CODIFICAÇÃO:

Para decodificar uma mensagem em binários usando os caracteres da tabela acima, por exemplo:

1 0 0 1 1 0 1 0 1 0 1 1 0 1 0 1 1 1 1 0 0 1 1 0 1 0 1 0

Basta ir utilizando cada bit da mensagem para percorrer a arvore de Huffman desde a raiz até a folha onde está o caractere decodificado. Após decodificar o primeiro caractere, volta-se para a raiz e repete-se o processo, até que toda a mensagem tenha sido decodificada.
Para a seqüência de bits acima, ao texto correspondente é ABRACADABRA. Vale observar que o texto (ABRACADABRA) acima foi compactado de 11 bytes (88 bits) para 28 bits.