IFSULDEMINAS, *campus* Muzambinho
Curso de Ciência da Computação

Compactação de Arquivos

Prof. Ricardo José Martins
ricardo.martins@muz.ifsuldeminas.edu.br

Curso de Bacharelado em Ciência da Computação
AED III – Algoritmo e Estruturas de Dados III

Compactadores de arquivos são softwares especializados em gerar uma representação mais eficiente de vários arquivos dentro de um único arquivo de modo que ocupem menos espaço na mídia de armazenamento ou o tempo de transferência deles sobre uma rede seja reduzido.

Os compactadores foram muito utilizados no passado quando as mídias de armazenamento tinham preços elevados e era necessário economizar espaço para armazenamento. Atualmente o uso deles é mais voltado a transferência de arquivos pela Internet para reduzir a massa de dados a ser transferida pela rede.

Run-length (ou RLE) é uma técnica para comprimir cadeias de caracteres onde existem seqüências longas de caracteres repetidos.

O princípio do funcionamento dessa codificação é simples: Quando temos a ocorrência de uma repetição contínua de determinado caractere, por exemplo, AAAAAAAAAAAA, é possível substituir sua representação pelo par (12, A). Entretanto não podemos simplesmente substituir no meio do texto a seqüencia de letras pelo número, senão uma frase como:

2. all is too well.

Se tornaria:

2. a2l is t2o we2l.

Como identificar se o número 2 estava realmente presente no texto original ou foi introduzido pela codificação? Neste caso precisamos identificar o início da codificação por um caractere especial. Assim, se usarmos por exemplo o símbolo de "@" como caractere especial, teremos:

2. a@2l is t@2o we@2l.

E conseguimos identificar exatamente onde o número 2 realmente existe na frase, e onde ele é parte da codificação. Entretanto, a cadeia ao invés de ser comprimida, foi na verdade expandida. Usando um caractere de escape a seqüencia mínima que podemos codificar sem causar expansão do arquivo precisa ter tamanho 3. Além disso o caractere especial não pode ser um dos caracteres que ocorrem dentro do texto.

Uma solução alternativa para o caractere de escape foi usada pelo protocolo MNP5, usado por fabricantes de modems. Nesse protocolo, ao invés de um caractere de escape, sempre que encontra uma repetição de 3 ou mais caracteres o codificador escreve os 3 primeiros caracteres repetidos no arquivo de saída, seguidos do número de repetições (além das 3 que já foram). Ou seja, se encontrar 3 caracteres "a" em seqüencia, imprime "aaa0", se forem 4 caracteres "b": "bbb1" e assim por diante. Repare que ainda assim existe um risco de expansão, mas ela é ligeiramente menor (e mais rara) do que no caso do caractere de escape.

Na compressão de imagens esta técnica é mais promissora pois imagens apresentam maiores áreas contínuas de uma mesma cor. Desenhos e outras imagens com número limitados de cores tendem a ser melhor comprimíveis usando esta técnica.

Uma abordagem interessante é a usada na compressão de imagens monocromáticas. Nessas imagens cada pixel é representado por apenas um bit. Assim o arquivo pode ser armazenado como uma lista dos tamanhos das seqüências alternadamente de pixels brancos (1) e negros (0), sem precisar indicar qual o valor da próxima seqüencia. Caso o primeiro pixel não seja da cor branca (1), o primeiro valor armazenado é 0, indicando uma seqüencia de tamanho 0 de pixels brancos.

Run-length bit a bit

Se a quantidade de valores forem bem definidas em um arquivo, podemos compactar o arquivo armazenando em apenas alguns bit ao invés de um byte todo: Por exemplo, imagine um arquivo apenas com as letras a, b, c e d. Como são apenas quatro valores, apenas dois bits são capazes de diferenciar uma informação da outra: a(00), b(01), c(10) e d(11). O texto a seguir

accaaddaadba

ao mudar as devidas associações, os valores binários ficam

00 10 10 00 00 11 11 00 00 11 01 00

convertendo para inteiro

40 60 52

finalmente para ASCII

(<4

Ou seja, o texto foi reduzido de 12 para 3 caracteres

Exercícios

1 - Implemente um compactador e um descompactador para o exemplo definido no slide anterior.

2 - Implemente um compactador e um descompactador para um arquivo que possui 8 dígitos diferentes: a, b, c, d, e, f, g e h.

Compressão Estatística

Baseia-se no uso de uma representação otimizada de caracteres ou grupos de caracteres afim de reduzir o tamanho dos dados. Caracteres de maior freqüência de ocorrência são representados por códigos binários pequenos, e os de menor freqüência são representados por códigos proporcionalmente maiores.

- **Algoritmo de Huffman**
- **Algoritmo de Shannon-Fano**

- **Algoritmo de Huffman**

  – **Necessita-se de uma tabela com a probabilidade de ocorrência de cada caractere;**

  Ex:

C1	0,5
C2	0,2
C3	0,2
C4	0,1

  – **Aplicação de um algoritmo para geração dos códigos otimizados de cada caractere através de uma árvore binária.**

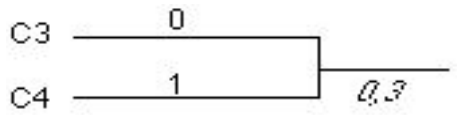
  – **Ordena-se a tabela de ocorrências (ordem crescente)**

  – **Retira-se da lista os dois caracteres de menor valor;**

- **Algoritmo de Huffman**

  – **Retira-se da lista os dois caracteres de menor probabilidade, sendo que estes formarão um novo ramo cuja probabilidade será a soma dos mesmos.**

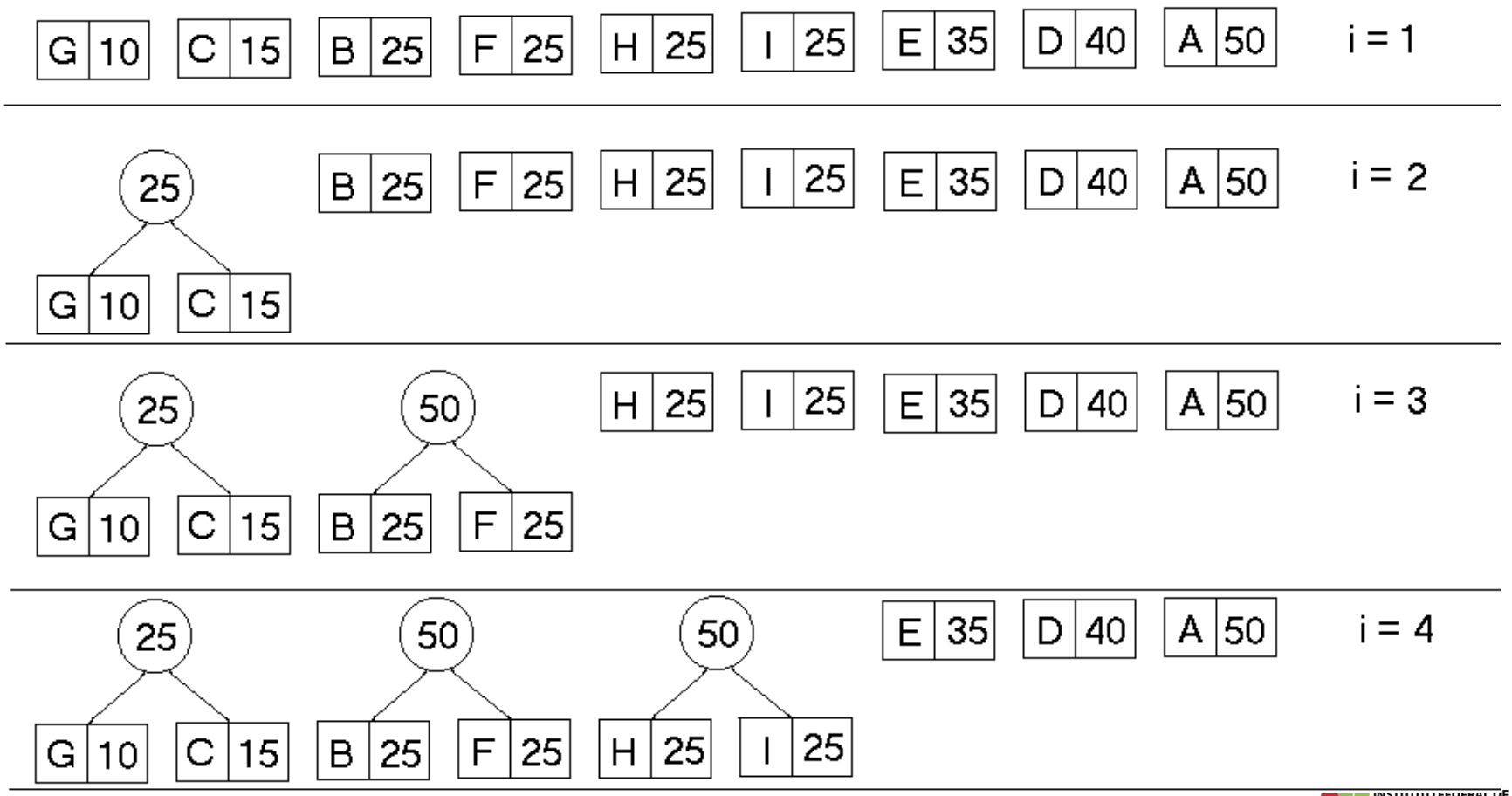
Ex:

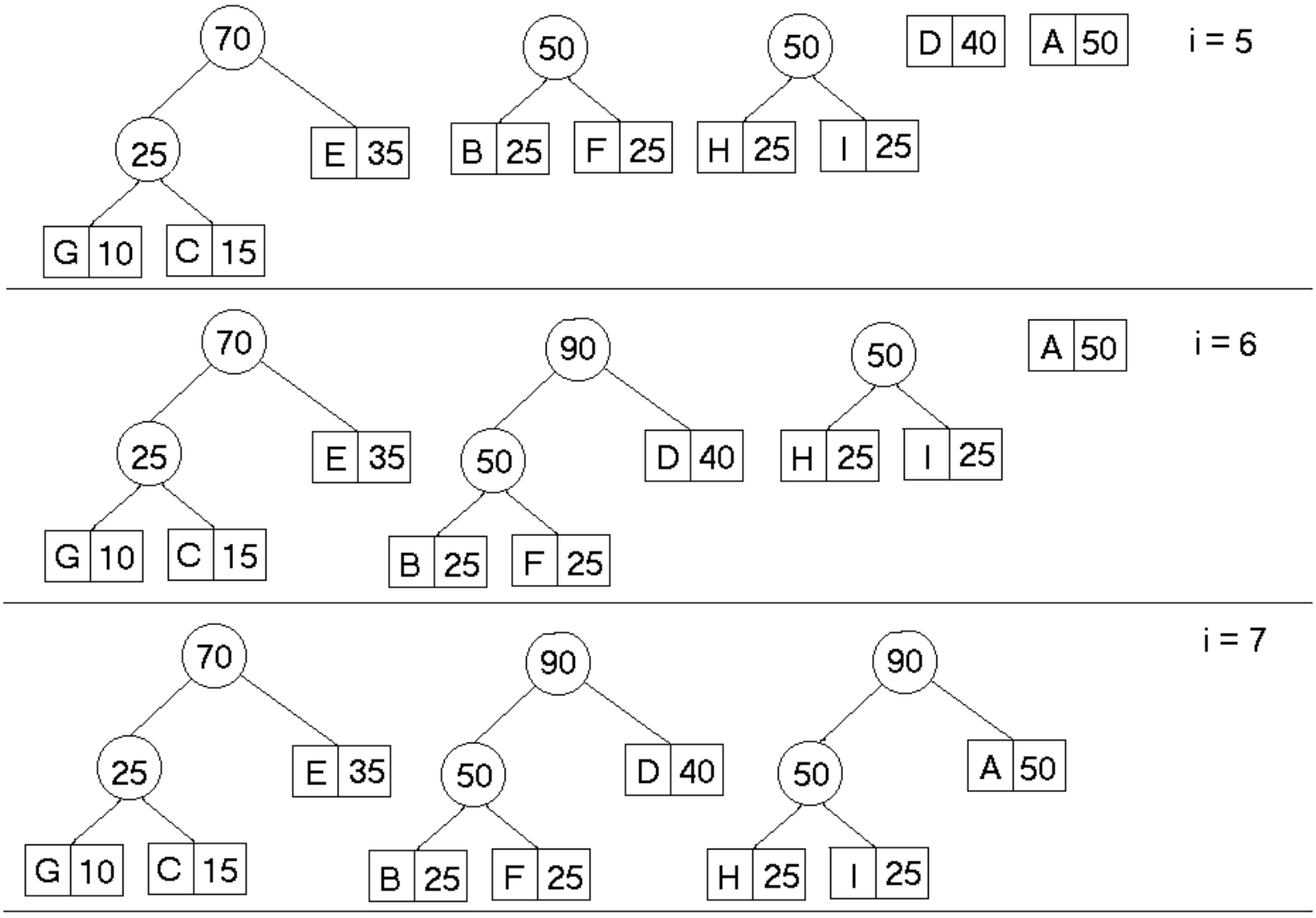


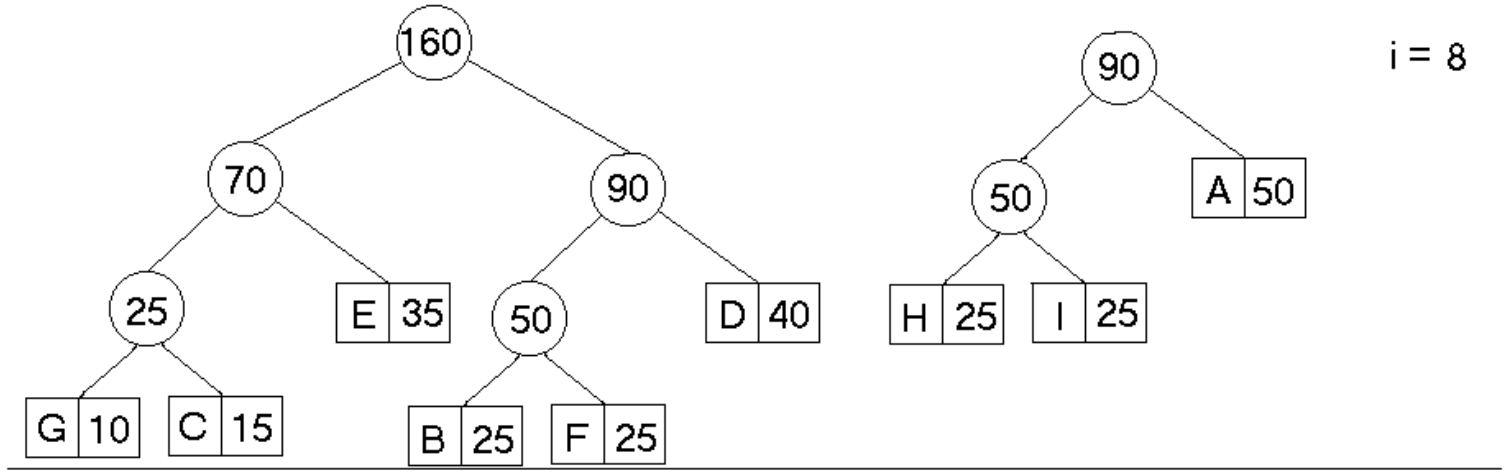
  – **Insere-se este novo caractere ORDENADAMENTE na lista.**

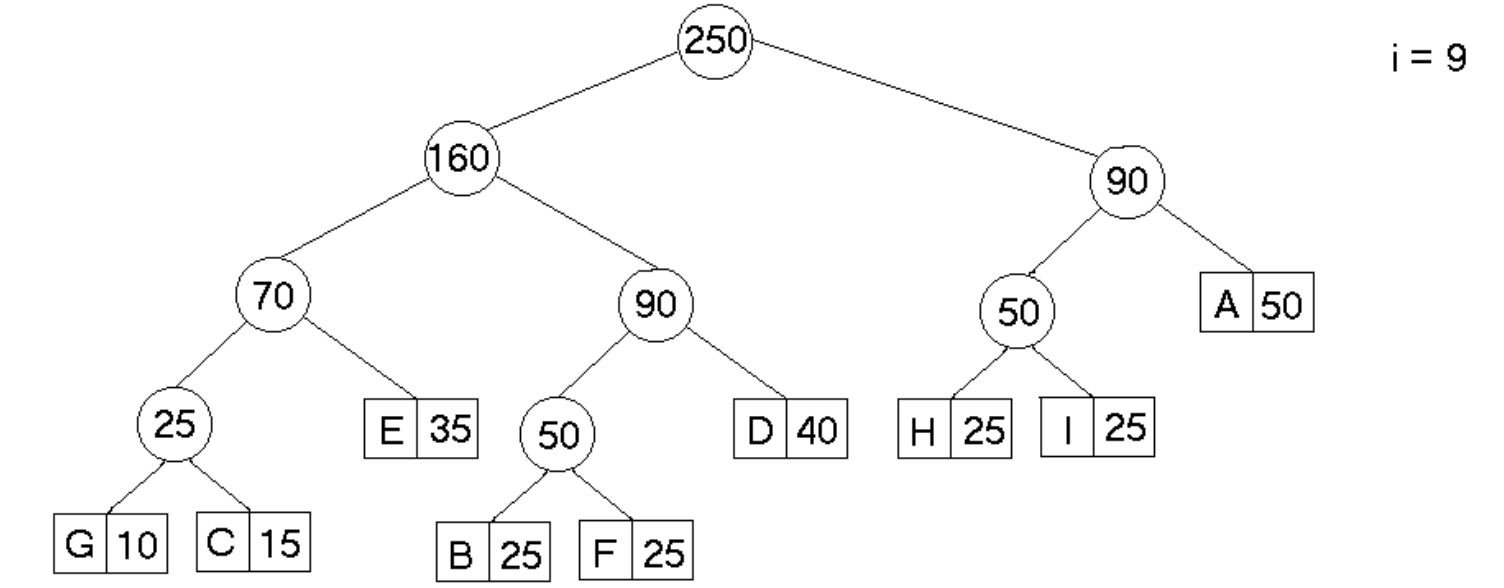
  – **Volta-se ao primeiro passo ate que reste somente um elemento na lista, cuja probabilidade seja 1.**

Exemplo de Codificação Huffman




70
25
E 35
G 10
C 15
50
B 25
F 25
50
H 25
I 25
D 40
A 50
i = 5
70
25
E 35
G 10
C 15
90
50
D 40
B 25
F 25
50
H 25
I 25
A 50
i = 6
70
25
E 35
G 10
C 15
90
50
D 40
B 25
F 25
90
50
A 50
H 25
I 25
i = 7


160
70
90
25
E 35
50
D 40
G 10
C 15
B 25
F 25
90
50
A 50
H 25
I 25
i = 8


250
160
90
70
90
50
A 50
25
E 35
50
D 40
H 25
I 25
G 10
C 15
B 25
F 25
i = 9

Para se montar o código, gera-se "0" cada vez que for para a esquerda e "1" cada vez que for para a direita; assim o código formado fica o seguinte:

	a	b	c	d	e	f	g	h	i
comprimento variável	11	0100	0001	011	001	0101	0000	100	101

- **Algoritmo de Shannon-Fano**

A forma desta técnica tem muitas semelhanças com a de Huffman. Necessita-se de uma tabela com a probabilidade de ocorrência de cada caractere, e de um procedimento para a codificação em binário. Por outro lado, o procedimento para a codificação, diferentemente de Huffman, baseia-se na divisão de conjuntos de probabilidades para a obtenção do código binário.

C4	0,5
C2	0,3
C1	0,1
C3	0,1

- **Algoritmo de Shannon-Fano**

Uma vez feito isso, divide-se a tabela em dois grupos cuja soma de probabilidades seja igual ou semelhante. No caso da tabela acima, serão obtidos dois grupos, um composto pelo caractere **C4** e outro pelos demais. O primeiro grupo recebe como primeiro valor de código o binário 0 e o segundo recebe 1:

C4	0
C2	1
C1	1
C3	1

- **Algoritmo de Shannon-Fano**

Para isso, repetimos o procedimento anterior, dividindo em dois subgrupos de probabilidades equivalentes. O caractere **C2** forma o primeiro subgrupo e os demais formam o segundo. Mais uma vez vamos colocar 0 para distinguir o primeiro e 1 para o segundo. O processo se repete ate que não haja mais grupos a dividir.

C4	0	
C2	1	0
C1	1	1
C3	1	1

C4	0		
C2	1	0	
C1	1	1	0
C3	1	1	1

C4	0		
C2	1	0	
C1	1	1	0
C3	1	1	1