**SIN110 Algoritmos e Grafos - Exercício E11**

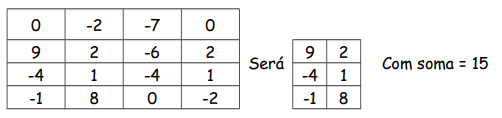
**Aluna:** Caroline Lopes Resek

**Matrícula:** 2017010113

1. Considere dois conjuntos de números inteiros A e B com m e n elementos, respectivamente. Eles não estão necessariamente ordenados, e assuma também que m ≤ n. Utilizando a estratégia de divisão e conquista, mostre como podemos computar A∪B e A∩B em um tempo proporcional a O(nlgm).
2. Dada uma matriz de números inteiros positivos e negativos, escreva e implemente um algoritmo que encontre um retângulo com a maior soma contido na matriz. A soma de um retângulo é a soma de todos os elementos dentro do retângulo.

Neste problema, o retângulo com a maior soma é referido como maximal. Um retângulo é qualquer submatriz (n X m), 1 ≤ n, m ≤ N, localizado dentro da matriz.

Como exemplo, o retângulo maximal da matriz:



1. Existe a necessidade de avisar um colega de vocês que determinado professor marcou prova. Para isso, será utilizado um canal extremamente rápido mas que possui um custo de utilização consideravelmente alto, logo, é interessante que sejam utilizados dados com compactação para diminuir o custo do envio. Para fazer a compactação, utilize o Código de Huffman para as frequências:



Assim que construída a árvore de Huffman, codifiquem a seguinte mensagem: “**aprovafoiadiada**”.

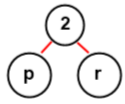
Aplicação do Código de Huffman:

**1º passo:** Ordeno a lista de caracteres em ordem crescente por frequência.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **p** | **r** | **f** | **v** | **s** | **n** | **h** | **d** | **m** | **i** | **o** | **e** | **a** |
| 1 | 1 | 2 | 2 | 2 | 3 | 4 | 4 | 6 | 6 | 8 | 10 | 15 |

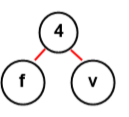
**2º passo:** Junto dois caracteres de menor frequência.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **f** | **v** | **s** | **p+r** | **n** | **h** | **d** | **m** | **i** | **o** | **e** | **a** |
| 2 | 2 | 2 | 2 | 3 | 4 | 4 | 6 | 6 | 8 | 10 | 15 |



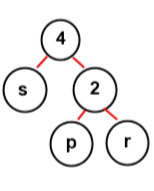
**3º passo:** Junto outros dois caracteres de menor frequência.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **s** | **p+r** | **n** | **h** | **d** | **f+v** | **m** | **i** | **o** | **e** | **a** |
| 2 | 2 | 3 | 4 | 4 | 4 | 6 | 6 | 8 | 10 | 15 |



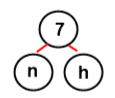
**4º passo:** Junto outros dois caracteres de menor frequência.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **n** | **h** | **d** | **f+v** | **p+r+s** | **m** | **i** | **o** | **e** | **a** |
| 3 | 4 | 4 | 4 | 4 | 6 | 6 | 8 | 10 | 15 |



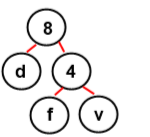
**5º passo:** Junto outros dois caracteres de menor frequência.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **d** | **f+v** | **p+r+s** | **m** | **i** | **n+h** | **o** | **e** | **a** |
| 4 | 4 | 4 | 6 | 6 | 7 | 8 | 10 | 15 |



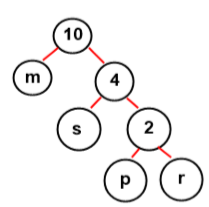
**6º passo:** Junto um caractere com um conjunto de caracteres de menor frequência.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **p+r+s** | **m** | **i** | **n+h** | **o** | **f+v+d** | **e** | **a** |
| 4 | 6 | 6 | 7 | 8 | 8 | 10 | 15 |



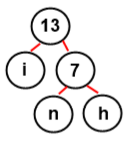
**7º passo:** Junto um caractere com um conjunto de caracteres de menor frequência.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **i** | **n+h** | **o** | **f+v+d** | **e** | **p+r+s+m** | **a** |
| 6 | 7 | 8 | 8 | 10 | 10 | 15 |



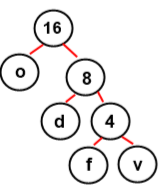
**8º passo:** Junto um caractere com um conjunto de caracteres de menor frequência.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **o** | **f+v+d** | **e** | **p+r+s+m** | **n+h+i** | **a** |
| 8 | 8 | 10 | 10 | 13 | 15 |



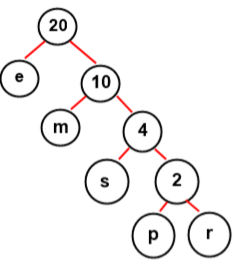
**9º passo:** Junto um caractere com um conjunto de caracteres de menor frequência.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **e** | **p+r+s+m** | **n+h+i** | **a** | **f+v+d+o** |
| 10 | 10 | 13 | 15 | 16 |



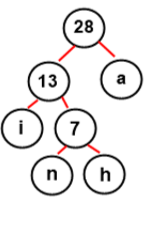
**10º passo:** Junto um caractere com um conjunto de caracteres de menor frequência.

|  |  |  |  |
| --- | --- | --- | --- |
| **n+h+i** | **a** | **f+v+d+o** | **p+r+s+m+e** |
| 13 | 15 | 16 | 20 |



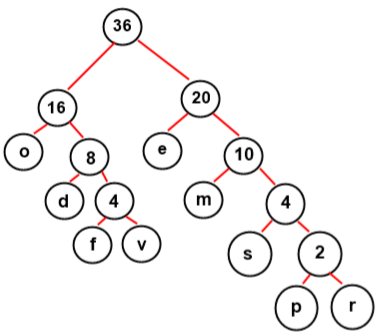
**11º passo:** Junto um caractere com um conjunto de caracteres de menor frequência.

|  |  |  |
| --- | --- | --- |
| **f+v+d+o** | **p+r+s+m+e** | **n+h+i+a** |
| 16 | 20 | 28 |



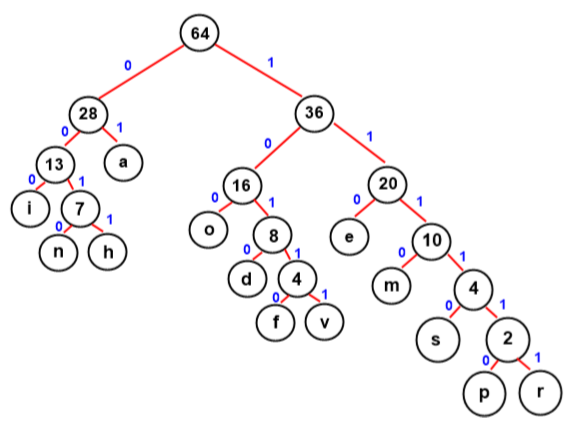
**12º passo:** Junto um conjunto de caracteres com outro conjunto de caracteres de menor frequência.

|  |  |
| --- | --- |
| **p+r+s+m+e+f+v+d+o** | **n+h+i+a** |
| 36 | 28 |



**13º passo:** Junto um conjunto de caracteres com outro conjunto de caracteres restante.

|  |
| --- |
| **n+h+i+a+p+r+s+m+e+f+v+d+o** |
| 64 |



Acima está a árvore de Huffman gerada. A partir dela é possível codificar a seguinte mensagem: “**aprovafoiadiada**”:

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **p** | **r** | **f** | **v** | **s** | **n** | **h** | **d** | **m** | **i** | **o** | **e** | **a** |
| 111110 | 111111 | 10110 | 10111 | 11110 | 0010 | 0011 | 1010 | 1110 | 000 | 100 | 110 | 01 |

**Codificação da mensagem:**

0111111011111110010111011011010000001101000001101001

1. Suponha dado um conjunto de livros numerados de 1 a n. Suponha que o livro i tem peso p[i] e que 0 < p[i] < 1 para cada i. Considere o problema de acondicionar os livros no menor número possível de envelopes de modo que cada envelope tenha no máximo 2 livros e o peso do conteúdo de cada envelope seja no máximo 1.

Escreva um algoritmo guloso *Min-Env(p,n)* que recebe um vetor p[1..n] e devolve o número mínimo de envelopes. O consumo de tempo do seu algoritmo deve ser O(nlgn). Procure mostrar que seu algoritmo está correto.

**Algoritmo guloso:** esse algoritmo é inspirado nas propriedades da estrutura ótima e da escolha gulosa, ele usa como sub-rotina o algoritmo HeapSort (A,n), o qual recebe e rearranja um vetor A[i...n] de modo que ele fique em ordem crescente. O consumo de tempo do algoritmo é O(nlogn).

**MinEnv(p,v)**

HeapSort(p,n) // pré-processamento:

i1 fn cont0

**enquanto** if **faça**

**se** p[i]+p[f]1

**então** ii+1

ff - 1

cont cont + 1

**devolva** cont