Alunos: Vinicius Guimarães RA: 802431
Vitor Enzo RA: 802123

1) O código de Huffman é um algoritmo guloso utilizado para realizar compressão de dados sem haver perda de informação. A ideia geral é que o algoritmo irá estabelecer código binários para cada caractere de acordo com a probabilidade de ocorrência dos mesmos. Assim, um caractere que aparece várias vezes em um texto, terá código binário menor do que quando comparado a um caractere que aparece poucas vezes.

Por exemplo, considerando $B = (a, b, c, d, e)$ como o alfabeto de símbolos a serem codificados e w_i a probabilidade de ocorrência de B_i nos dados, estão temos:

	a	b	c	d	e	f	
w	0.37	0.12	0.09	0.25	0.10	0.07	→ Soma das probabilidades dá 1

Utilizando a forma tradicional, teremos que utilizar 3 bits pois $2^2 = 4$ e temos 6 caracteres, obtendo a seguinte codificação quando iniciada pelo símbolo a:

$a = 000$, $b = 001$, $c = 010$, $d = 011$, $e = 100$, $f = 101$

O que nos dá um tamanho médio do código de:

$$I(c) = \sum_{j=1}^{n} w_j \cdot \text{tamanho do código do caractere } B_j =$$

$$= 0,37.3 + 0,12.3 + 0,09.3 + 0,25.3 + 0,10.3 + 0,07.3 = \boxed{3}$$

Ou seja, em média é preciso 3 bits para codificar 1 caractere.

Porém, com a aplicação do algoritmo de huffman, teremos os seguintes códigos:

$a = 11$, $b = 011$, $c = 001$, $d = 10$, $e = 010$, $f = 000$

O que gera um tamanho médio de:

$$I(c) = 0,37.2 + 0,12.3 + 0,09.3 + 0,25.2 + 0,1.3 + 0,07.3 = \boxed{2,38}$$

Portanto, sendo o tamanho médio de cada caractere como sendo 2,38, temos que isso é aproximadamente 20,6% menor do que a codificação tradicional. Portanto, por gerar códigos de acordo com a probabilidade de ocorrência em um texto, o algoritmo de Huffman consegue gerar códigos menores do que a forma tradicional

(2) Dado o código abaixo

```
Huffman (α){
  n = |α|
  for each xi ∈ α.
    Insert(Q, αi)
  for i = 1 to n-1 {
    Create a new node z)
    X = ExtractMin(Q)
    Y = ExtractMin(Q)
    z.left = X
    z.right = Y
    w(z) = w(X) + w(Y)
    Insert(Q, z)
  }
  return ExtractMin(Q)
}
```

→ Estamos inserindo cada elemento do conjunto α dentro de Q (Onde Q é uma estrutura de dado que servirá para pegarmos a árvore de menor probabilidade)

→ Se tivermos 3 caracteres por exemplo, teremos que realizar duas funções de árvores

→ Começamos a criar um nó z, que será a raiz da árvore e adicionamos os dois menores elementos de Q dentro de z. Além disso, a probabilidade de z é atualizada para ser o somatório das probabilidades de X e Y.

→ Em seguida inserimos z dentro de Q para continuar o processo

→ Ao final da execução do for, teremos a árvore de Huffman, que será retornada

O código de Huffman é guloso pois a cada execução do for, são pegos da fila de prioridade os 2 nós que estarão mais longe da raiz da árvore quando comparado com aqueles que ainda estão na lista de prioridade, ou seja, aqueles com a maior prioridade (menor probabilidade)

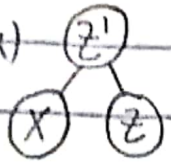
(3)

e	r	s	t	n	l	z	x
34	22	24	28	15	10	9	8

j=1	e	r	s	t	n	l	z	x
Q:	34	22	24	28	15	10	9	8

Escolhidos: x, z

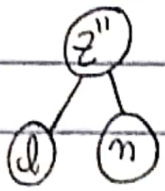
frequência z^I = w(z)+w(x) = 17



j=2	e	r	s	t	n	l	z^I
Q:	34	22	24	28	15	10	17

Escolhidos: l, n

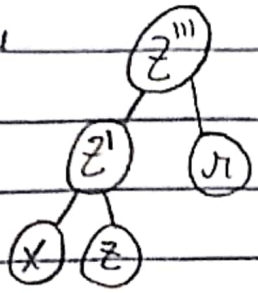
frequência de z^{II} = 25



j=3	e	r	s	t	z^I	z^{II}
Q:	34	22	24	28	17	25

Escolhidos: z^I; r

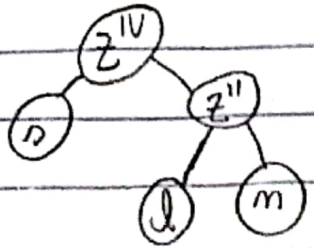
frequência de z^{III} = 39



j=4	e	s	t	z^{II}	z^{III}
Q=	34	24	28	25	39

Escolhidos: s, z^{II}

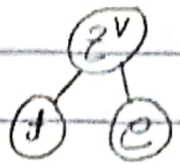
frequência de z^{IV} = 49



j=5	e	t	z^{III}	z^{IV}
Q:	34	28	39	49

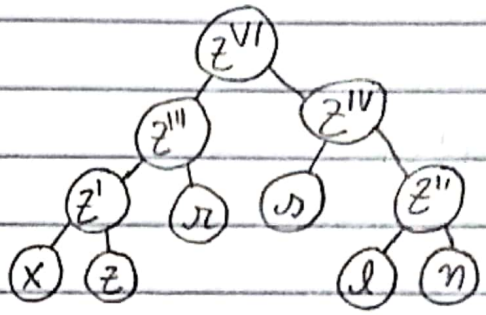
Escolhidos: t, e

frequência de z^V = 62



j=6	z^{III}	z^{IV}	z^V
Q:	39	49	62

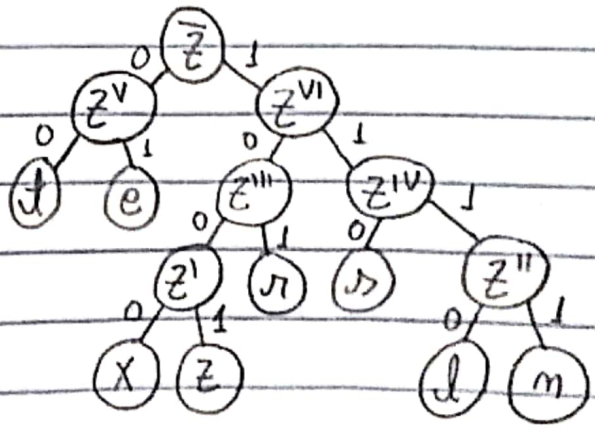
frequência de z^{VI} = 39+49 = 88



j=7	z^V	z^{VI}
Q:	62	88

Escolhidos: z^V, z^{VI}

frequência de $\bar{z}$ = 62+88 = 150



Assim, tendo a árvore de Huffman calculada, obtemos os seguintes códigos binários para os símbolos:

e = 01, r = 101, a = 110, t = 00, m = 1111, l = 1110, z = 1001, X = 1000

Se convertermos a estratégia padrão, teríamos 3 bits para cada caractere.

Com Huffman temos: $\frac{34.2 + 22.3 + 24.3 + 29.2 + 15.4 + 10.4 + 9.4 + 9.4}{150} = 2,1866$

Que significa uma redução de cerca de 5%.

AA2 - Algoritmos gulosos (Código de Huffman)

Questão 04)

$\alpha = (a, b, c, d, e, f, g, h)$

$w = (1, 1, 2, 3, 5, 8, 13, 21)$

Aplicando o algoritmo de Huffman:

$i=1$ $Q^{(1)} = \{ \boxed{\begin{matrix} a \\ 1 \end{matrix}} \; \boxed{\begin{matrix} b \\ 1 \end{matrix}} \; c \; d \; e \; f \; g \; h \}$ $x = a$, $y = b$, $w(z^1) = 2$

$i=2$ $Q^{(1)} = \{ \boxed{\begin{matrix} z^1 \\ 2 \end{matrix}} \; \boxed{\begin{matrix} c \\ 2 \end{matrix}} \; d \; e \; f \; g \; h \}$ $x = z^1$, $y = c$, $w(z^2) = 4$

$i=3$ $Q^{(3)} = \{ \boxed{\begin{matrix} z^2 \\ 4 \end{matrix}} \; \boxed{\begin{matrix} d \\ 3 \end{matrix}} \; e \; f \; g \; h \}$ $x = d$, $y = z^2$, $w(z^3) = 7$

$i=4$ $Q^{(4)} = \{ \boxed{\begin{matrix} z^3 \\ 7 \end{matrix}} \; \boxed{\begin{matrix} e \\ 5 \end{matrix}} \; f \; g \; h \}$ $x = e$, $y = z^3$, $w(z^4) = 12$

$i=5$: $Q^{(5)} = \{ \boxed{\begin{matrix} z^4 \\ 12 \end{matrix}} \; \boxed{\begin{matrix} f \\ 8 \end{matrix}} \; g \; h \}$ $x = f$, $y = z^4$, $w(z^5) = 20$

$i=6$ $Q^{(6)} = \{ \boxed{\begin{matrix} z^5 \\ 20 \end{matrix}} \; \boxed{\begin{matrix} g \\ 13 \end{matrix}} \; h \}$ $x = g$, $y = z^5$, $w(z^6) = 33$

$i=7$: $Q^{(7)} = \{ \boxed{\begin{matrix} z^6 \\ 33 \end{matrix}} \; \boxed{\begin{matrix} h \\ 21 \end{matrix}} \}$ $x = h$, $y = z^6$, $w(z^7) = 54$

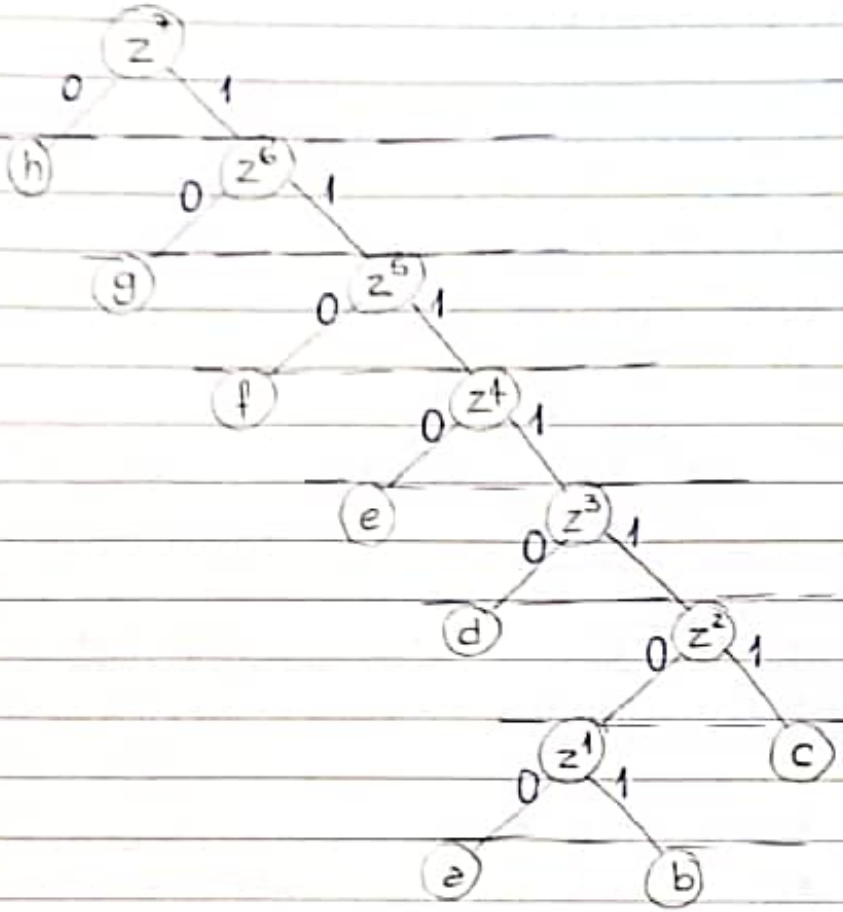


Tabela de Simbolos

h = 0	d = 11110
g = 10	c = 111111
f = 110	b = 1111101
e = 1110	a = 1111100

Assim, a cadeia de bits de "fdheg" seria:

fdheg = 110|11110|0|1110|10

Generalizando, para $n = 2$, temos:

$s_1 = 0$

$s_2 = 1$

para $n \geq 3$:

$s_1 = (n-3)\cdot'1' + '00'$ $\quad s_i = (n-i)\cdot'1' + '0'$

$s_2 = (n-3)\cdot'1' + '01'$

$s_3 = (n-3)\cdot'1' + '1'$

:

Questão 05) Sabe-se que, de acordo com Claude Shannon e sua teoria da informação, a quantidade de informação do símbolo α_i é dada por

$$h(\alpha_i) = \log_2 \frac{1}{w_i},$$

ou seja, é inversamente proporcional à probabilidade

A entropia associada à distribuição de probabilidades do vetor $\vec{w}$ é dada por

$$H(\vec{w}) = \sum_i w_i \cdot h(\alpha_i)$$

$$= \sum_i w_i \cdot \log_2 \frac{1}{w_i}$$

$$= \sum_i w_i \cdot \log_2 w_i^{-1}$$

$$H(\vec{w}) = -\sum_i w_i \cdot \log_2 w_i$$

A prova, consiste em notar que ~~não~~ o tamanho do código c $\ell(c)$ será igual a $H(\vec{w})$ quando $\ell(\alpha_i) = h(\alpha_i)$.

Deseja-se mostrar que $H(\vec{w}) \leq \ell(c)$. Iremos iniciar calculando $H(\vec{w}) - \ell(c)$,

$$H(\vec{w}) - \ell(c) = \sum_j w_j \cdot \log_2 \frac{1}{w_j} - \sum_j w_j \cdot \ell(\alpha_j)$$

Note que

$$\ell(\alpha_j) = \log_2 2^{\ell(\alpha_j)}$$

Logo,

$$H(\vec{w}) - \ell(c) = \sum_j w_j \cdot \log_2 \frac{1}{w_j} + \sum_j w_j \cdot \log_2 2^{-\ell(\alpha_j)}$$

$$= \sum_j w_j \cdot \log_2 \frac{2^{-\ell(\alpha_j)}}{w_j}$$

Realizando troca de base no logaritmo temos

$$\log_2 \left(\frac{2^{-\ell(\alpha_j)}}{w_j}\right) = \frac{\ln\left(\frac{2^{-\ell(\alpha_j)}}{w_j}\right)}{\ln 2} = \frac{1}{\ln 2} \cdot \ln \frac{2^{-\ell(\alpha_j)}}{w_j}$$

Portanto,

$$H(\vec{w}) - \ell(c) = \frac{1}{\ln 2} \sum_j w_j \ln \frac{2^{-\ell(a_j)}}{w_j}$$

Tomaremos como verdade que $\ln x \leq x-1$, segue

$$H(\vec{w}) - \ell(c) \leq \frac{1}{\ln 2} \cdot \sum_j w_j \left(\frac{2^{-\ell(a_j)}}{w_j} - 1 \right)$$

$$\leq \frac{1}{\ln 2} \left(\sum_j 2^{-\ell(a_j)} - \sum_j w_j \right)$$

$$\leq \frac{1}{\ln 2} \left(\sum_j 2^{-\ell(a_j)} - 1 \right)$$

Seja $\ell_{max} = \max\{\ell(a_1), \ell(a_2), \dots, \ell(a_n)\}$. Então,

$$\sum_{j=1}^{\ell_{max}} 2^{\ell_{max} - \ell_j} = 2^{\ell_{max}-1} + 2^{\ell_{max}-2} + \cdots 2^{\ell_{max} - \ell_{max}}$$

O número máximo de nós folhas é $2^{\ell_{max}}$. Então,

$$\sum_{j=1}^{\ell_{max}} 2^{\ell_{max} - \ell_j} < 2^{\ell_{max}}$$

Dividindo ambos lados por $2^{\ell_{max}}$, tem-se

$$\sum_{j=1}^{\ell_{max}} 2^{-\ell_j} < 1$$

Portanto,

$$H(\vec{w}) - \ell(c) \leq 0$$

$$H(\vec{w}) \leq \ell(c)$$

no código de Huffman, quanto menor a probabilidade, maior a informação, o que resulta em $\ell(c) = H(\vec{w})$.