Módulo 5 – Codificação
Sistemas Multimédia

Ana Tomé
José Vieira

Departamento de Electrónica, Telecomunicações e Informática

Universidade de Aveiro

Sumário

- Códigos binários
  - Representação de informação com códigos
  - Representação de texto
    - ASCII
    - Unicode
  - Representação numérica
  - Árvores Binárias
- Codificadores Probabilísticos e Entropia
- Codificador Huffman

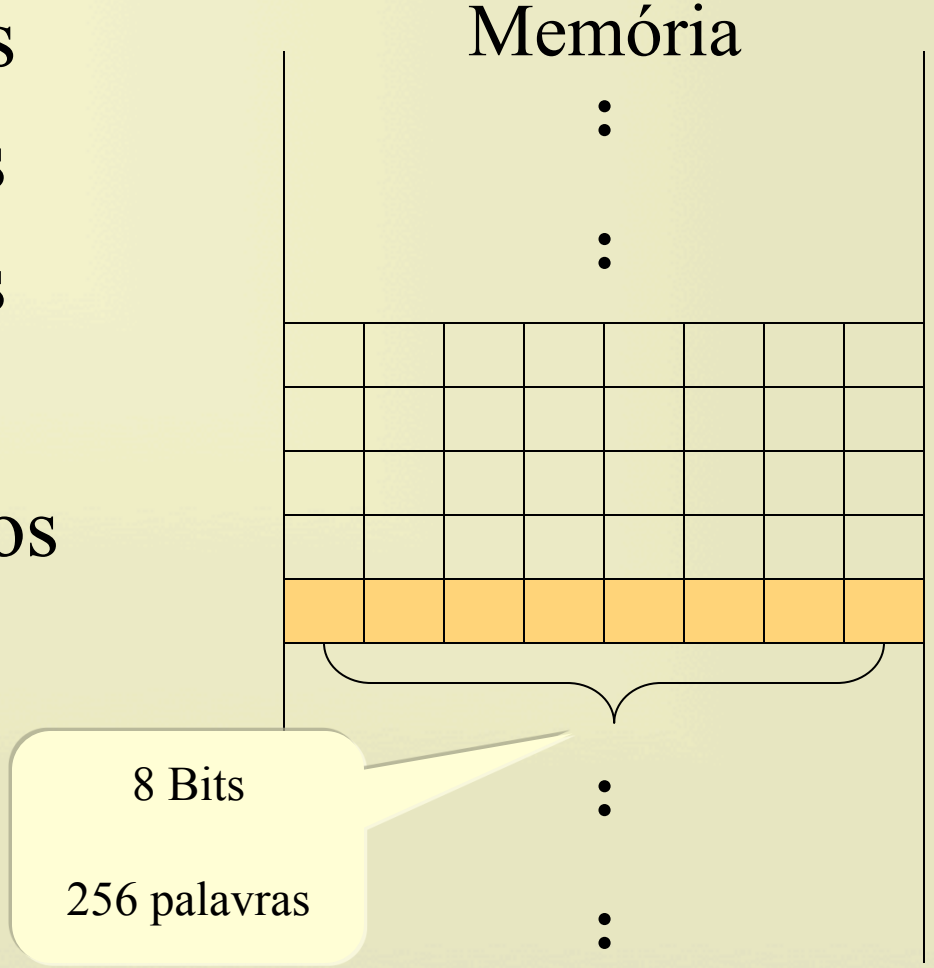
Códigos Binários

Codificação

- Os computadores armazenam toda a informação na forma mais elementar designada por bits.
- Cada bit pode tomar dois valores distintos “1” ou “0”. Um conjunto de 8 bits designa-se por Byte.
- 1024 Bytes = 1KByte.
- 1024 x 1024 Bytes = 1MByte.
- 1024 x 1MByte = 1GByte.
- Para armazenar informação proveniente das mais diversas fontes é necessário codificá-la.
- O conhecimento do código permite interpretar a informação armazenada na forma binária.

Capacidade de representação

- 1 Bit = 2 estados
- 2 Bits = 4 estados
- 3 Bits = 8 estados
- ...
- N Bits = 2^N estados

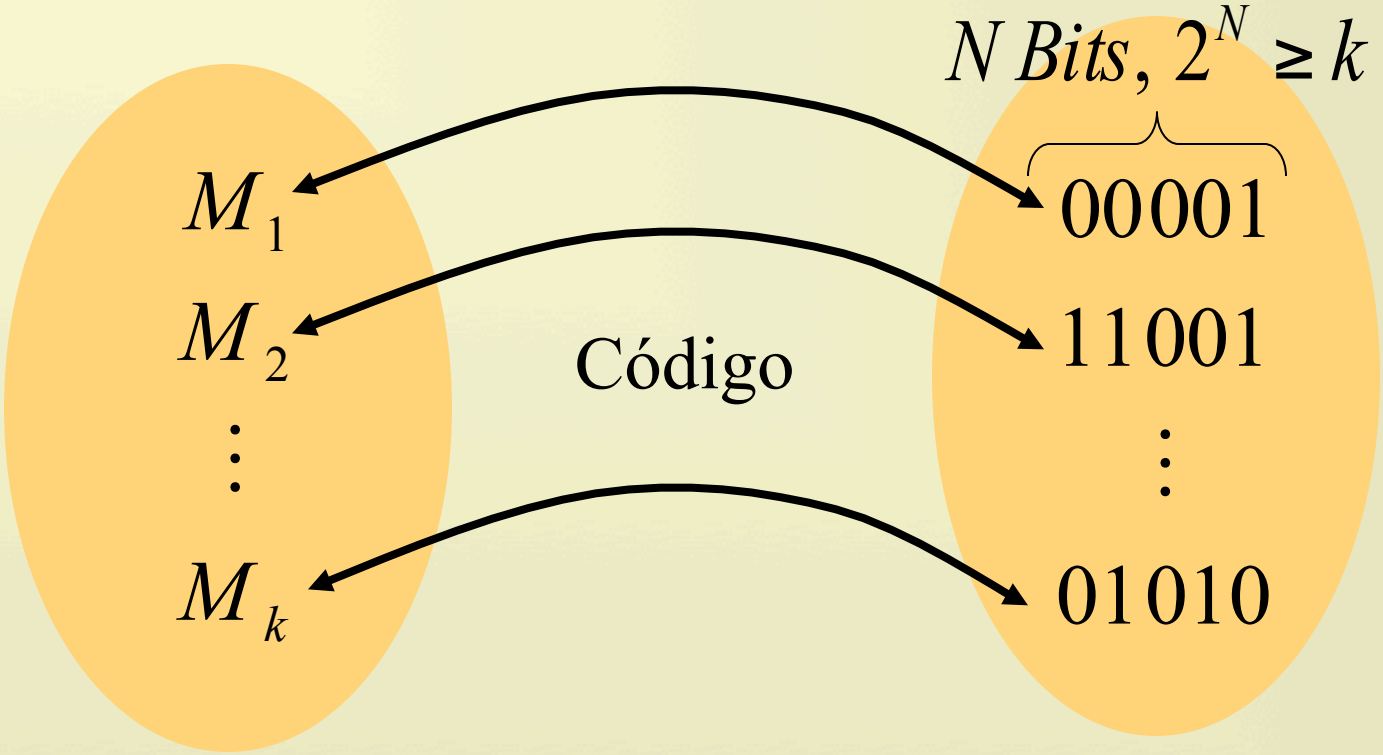


Capacidade de representação

- Exemplo do número de combinações que é possível gerar com 3 bits

b_2	b_1	b_0
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1

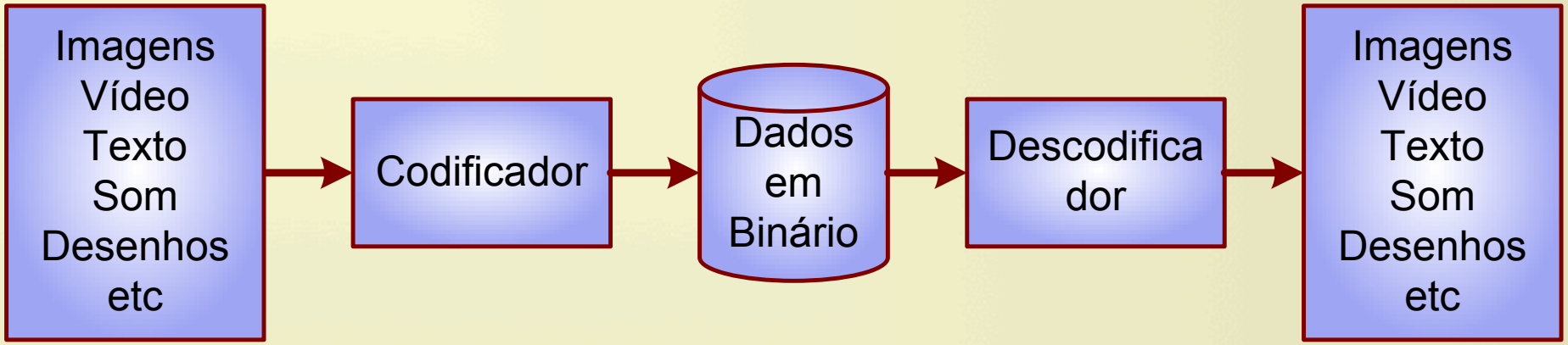
Códigos de representação



k Mensagens para transmitir/armazenar

k palavras binárias

Codificação/Representação



Os vários tipos de informação são codificados de forma diferente. Para interpretar cada um dos formatos é necessário um descodificador.

Código ASCII (texto)

- A primeira versão do código ASCII (American Standard Code for Information Interchange) foi criada em 1963 para normalizar a transmissão e armazenamento de texto. Em 1967 foram incluídas as letras minúsculas no código que no essencial permaneceu inalterado até aos nossos dias.

Código ASCII

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT	LF	VT	FF	CR	SO	SI
1	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
2	sp	!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	DEL

Exemplo de codificação para a letra “A”

4x16+1=64+1=65=100 0001

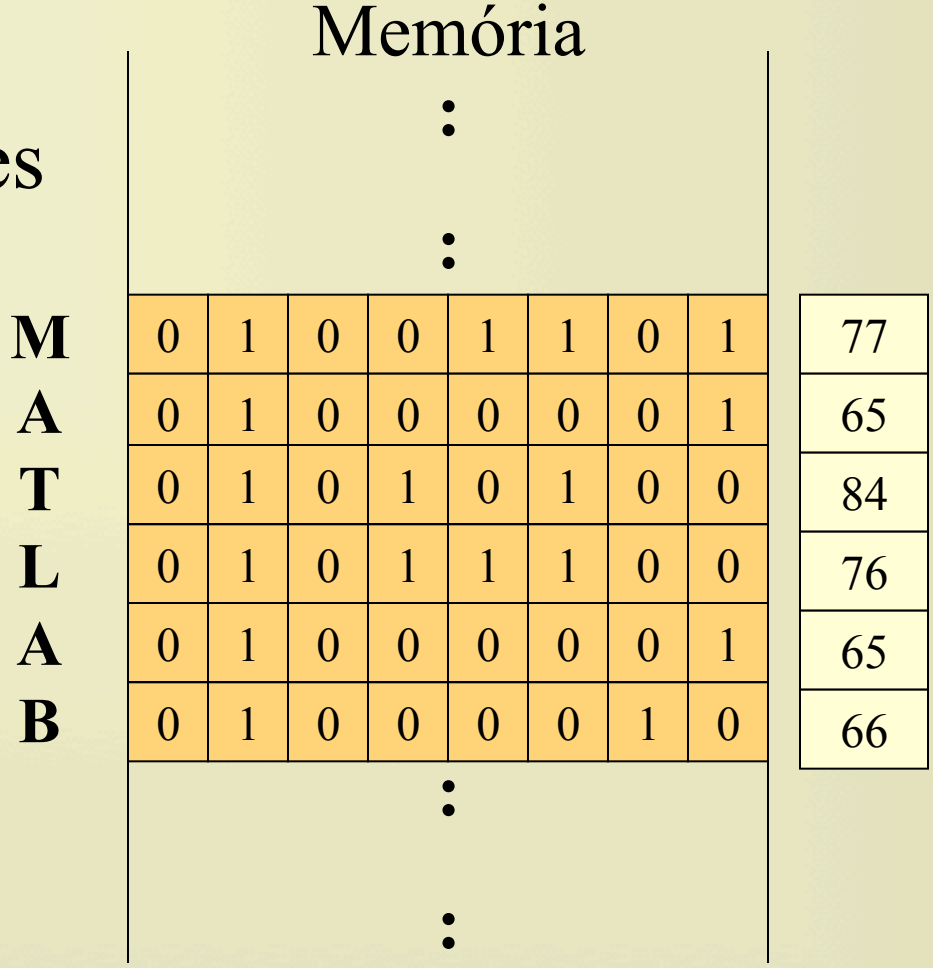
Letra “W”: 5x16+7=87=101 0111

Código ASCII

32		48	0	64	@	80	P	96	`	112	p
33	!	49	1	65	A	81	Q	97	a	113	q
34	“	50	2	66	B	82	R	98	b	114	r
35	#	51	3	67	C	83	S	99	c	115	s
36	$	52	4	68	D	84	T	100	d	116	t
37	%	53	5	69	E	85	U	101	e	117	u
38	&	54	6	70	F	86	V	102	f	118	v
39	‘	55	7	71	G	87	W	103	g	119	w
40	(	56	8	72	H	88	X	104	h	120	x
41	)	57	9	73	I	89	Y	105	i	121	y
42	*	58	:	74	J	90	Z	106	j	122	z
43	+	59	;	75	K	91	[	107	k	123	{
44	,	60	<	76	L	92	\	108	l	124	|
45	-	61	=	77	M	93	]	109	m	125	}
46	.	62	>	78	N	94	^	110	n	126	~
47	/	63	?	79	O	95	_	111	o	127	DEL

Exemplo código ASCII

- Código ASCII
- 7 Bits = 128 Caracteres



UNICODE

- O código ASCII possui a grande desvantagem de apenas permitir a representação de 2^8=256 símbolos diferentes.
- O código UNICODE pretende normalizar a codificação dos carateres utilizados por todas as escritas existentes no mundo. Utiliza 16 bits para codificar cada carater e encontra-se disponível nos sistemas informáticos mais recentes.
- Mais informações em http://www.unicode.org

Códigos binários

- Para representar números com bits é possível encontrar uma forma mais compacta do que a codificação ASCII.
- No sistema decimal utilizado para realizar cálculo, os números são representados fazendo uso da sua posição relativa:

$$1995_{10} = 1\times10^3 + 9\times10^2 + 9\times10^1 + 5\times10^0$$

Base 10

Códigos binários

- Se modificarmos a base de decimal para binária podemos utilizar o mesmo tipo de representação:

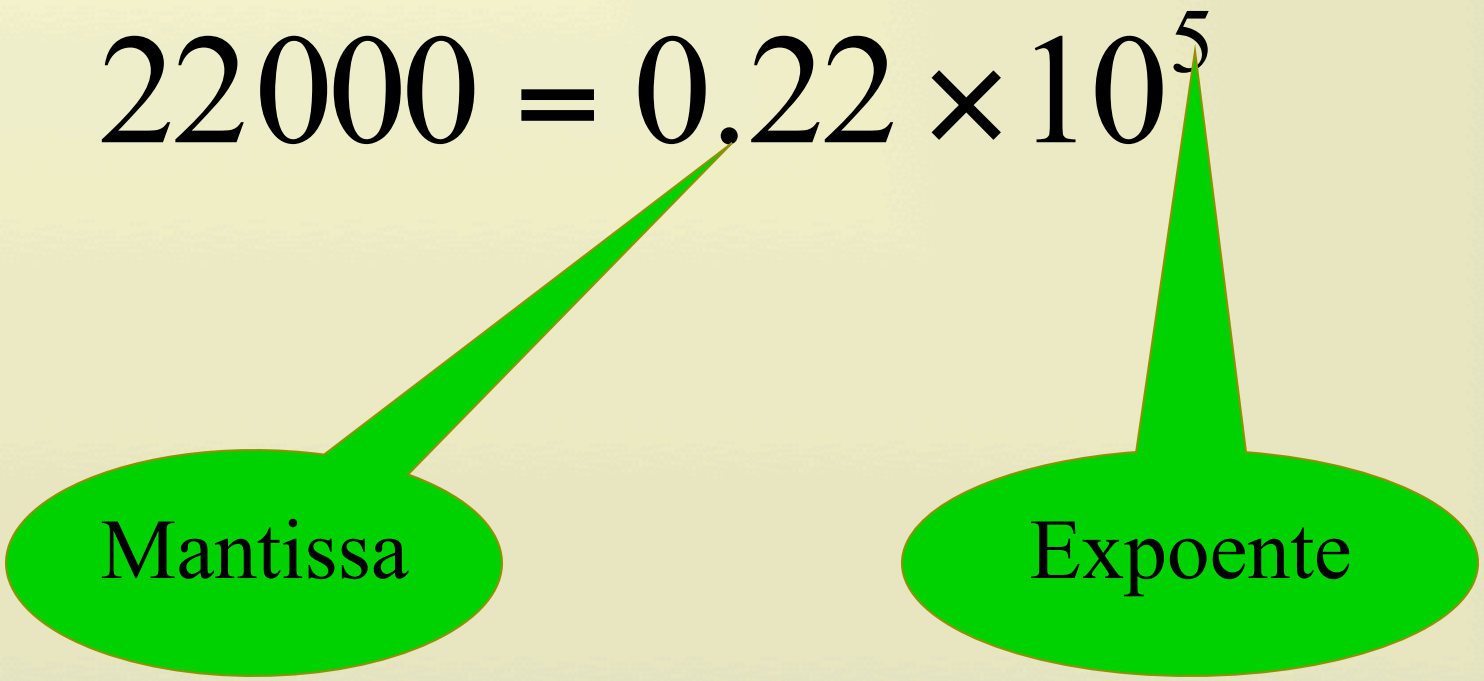
$$1001_2 = 1\times 2^3 + 0\times 2^2 + 0\times 2^1 + 1\times 2^0$$

- Note-se que o número anterior tem o valor em decimal de 8+0+0+1=9, sendo por isso uma das possíveis representações de números decimais em binário

Formato exponencial decimal

- Em formato decimal é útil representar os números utilizando a notação exponencial:

$$22000 = 0.22 \times 10^{5}$$



Formato exponencial binário

- No formato exponencial binário a mantissa e a base são representados em formato binário na base 2.

$$11000_b = 0.11_b \times 2^5$$

Formato numérico no Matlab

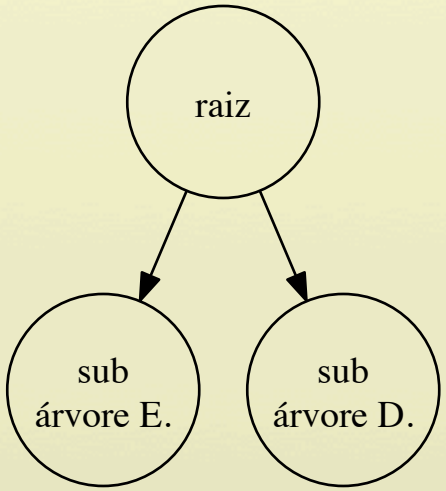
- O Matlab utiliza 64 bits para representar os números: 52bits para a mantissa e 12 para o expoente. A representação dos números é feita utilizando um formato exponencial que permite uma gama dinâmica muito grande.
- Para as imagens o Matlab tem um formato com 8 bits para representar inteiros sem sinal.

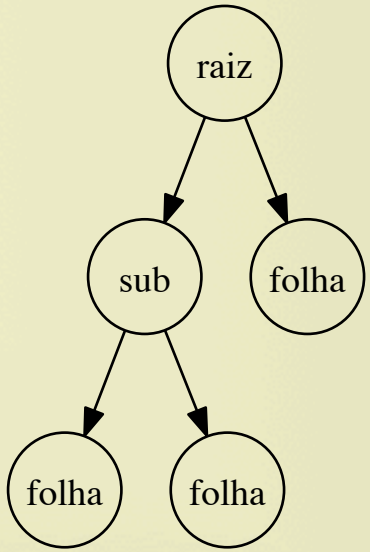
Codificadores Probabilísticos

Entropia

Árvores binárias

Uma árvore binária tem um elemento denominado raiz que aponta para duas sub-árvores binárias, esquerda e direita.





Nó inicial : raiz
Nó terminal: folha

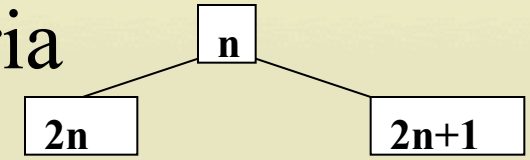
Codificação binária e árvores

Multiplicar por 2?

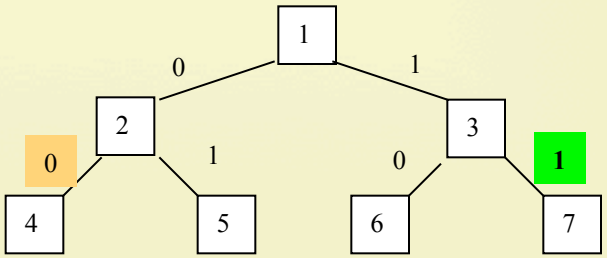
+1 é ?

4 (dec) =	**000100**	**5 (dec) =**	**000101**
8 (dec) =	**001000**	**9 (dec) =**	**001001**
16 (dec) =	**010000**	**17 (dec) =**	**010001**

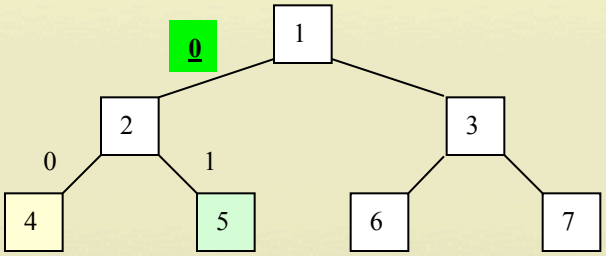
Árvore binária



Exemplo



Colocar
Raiz 1
0- nos ramos da esquerda
1-ramos da direita

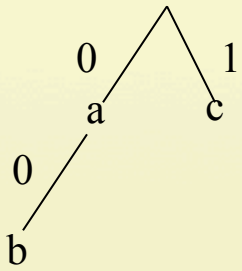


01

Código binário, percurso da folha para raíz

5 ⟹ 101

Códigos e propriedades



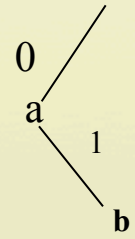
Símbolo a: **0**

Símbolo b: **00**

Símbolo c : **1**

001: “bc” ou “aac”

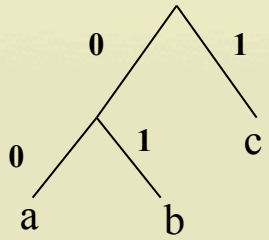
Código ambíguo



Símbolo a: **0**

Símbolo b: **01**

Código não instantâneo



Símbolo a: **00**

Símbolo b: **01**

Símbolo c: **1**

Símbolos em nós terminais

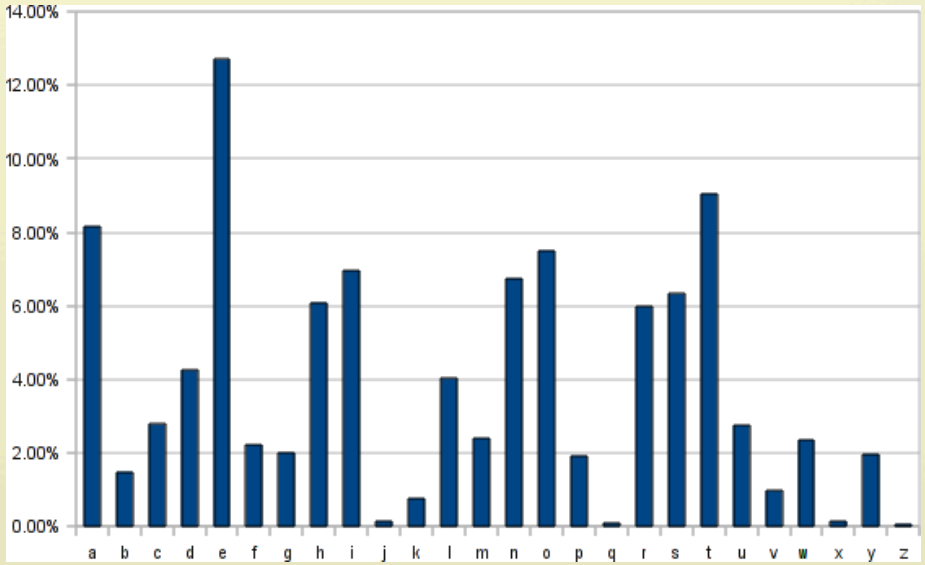
Instantâneo e não ambíguo

Mensagem e Alfabeto

- Para representar texto em formato ascii atribuímos 1byte para cada símbolo (caractere) para realizar a codificação.
- No entanto, nem todos os símbolos têm a mesma probabilidade de ocorrência num texto. Por exemplo o símbolo "@" aparece muito raramente mas atribuímos o mesmo número de bits que os necessários para representar o "a".
- Na língua Portuguesa por exemplo os diferentes caracteres têm diferentes probabilidades de ocorrer.
- Um código que usasse menos do que 8 bits nos caracteres mais frequentes e mais bits nos menos frequentes seria mais eficiente.

Código de Morse

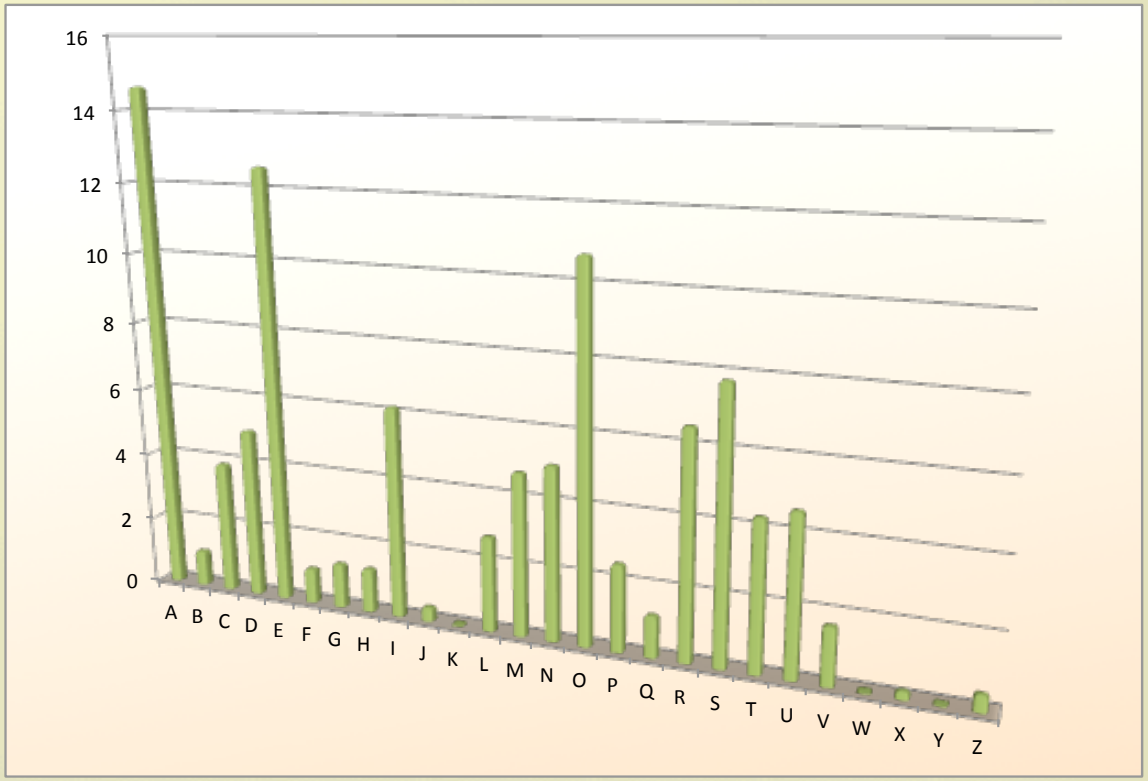
- No código de Morse os símbolos mais curtos são usados para as letras mais frequentes.



A •–	J •–––	S •••	1 •————
B –•••	K –•–	T –	2 ••–––
C –•–•	L •–••	U ••–	3 •••––
D –••	M ––	V •••–	4 ••••–
E •	N –•	W •––	5 •••••
F ••–•	O –––	X –••–	6 –••••
G ––•	P •––•	Y –•––	7 ––•••
H ••••	Q ––•–	Z ––••	8 –––••
I ••	R •–•	0 –––––	9 ––––•

Frequência Relativa das Letras no Português

Letra	Freq.%	Letra	Freq.%
A	14.63	N	5.05
B	1.04	O	10.73
C	3.88	P	2.52
D	4.99	Q	1.20
E	12.57	R	6.53
F	1.02	S	7.81
G	1.30	T	4.34
H	1.28	U	4.63
I	6.18	V	1.67
J	0.40	W	0.01
K	0.02	X	0.21
L	2.78	Y	0.01
M	4.74	Z	0.47



Mensagem e Alfabeto

Considere a mensagem (sequência de símbolos)

AABCAABDBCABADAA

- Mensagem com 16 símbolos
- Alfabeto {A,B,C,D} com 4 símbolos

1. Quantos bits para representar o alfabeto?
2. Quantos bits para representar a mensagem?

- Na mensagem os símbolos têm igual probabilidade?

Quantidade de Informação de um Símbolo

- Considere-se uma dado acontecimento s_i com uma probabilidade de ocorrência p_i.
- Qual a quantidade de informação contida neste evento?
- Vamos supor que p_i= 1/256. Neste caso podemos ter outros 255 acontecimentos de igual probabilidade e para os distinguir é necessário usar 1 byte.
- Sendo assim, a quantidade de informação contida neste acontecimento seria de 1 byte.

Entropia

- Medida da quantidade informação de um símbolo s_i

$$I(s_i) = \log_2(\tfrac{1}{p_i}) = -\log_2(p_i)$$

com probabilidade p_i

- A informação média (ENTROPIA) de uma mensagem com um alfabeto de *N* símbolos será então dada por

$$H(M) = \sum_{i=1}^{N} p_i \log_2(\tfrac{1}{p_i}) = -\sum_{i=1}^{N} p_i \log_2(p_i) \quad bps$$

bps –bits por símbolo

Mensagem e entropia

A mensagem: AABCAABDBCABADAA

Símbolos	Número de ocorrências
A	8
B	4
C	2
D	2

$$H(M) = -\left(\tfrac{1}{2}\log_2 \tfrac{1}{2} + \tfrac{1}{4}\log_2 \tfrac{1}{4} + 2\,\tfrac{1}{8}\log_2 \tfrac{1}{8}\right) =$$

$$= -\left(\tfrac{1}{2}(-1) + \tfrac{1}{4}(-2) + \tfrac{1}{4}(-3)\right) = 1.75$$

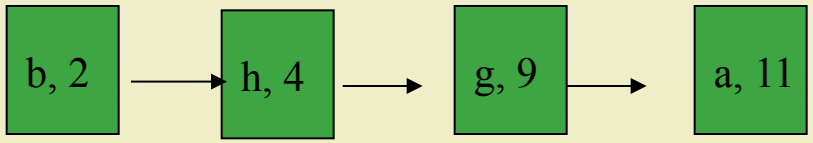
A mensagem precisa de 1.75 bits por símbolo (bps)

Exercício: Calcule a entropia da escrita Portuguesa com base na frequência relativa das letras usando o Matlab e textos de www.gutenberg.org.

Código de Huffman

Código de Huffman

Um conjunto de símbolos e número de ocorrências numa mensagem



Nota:

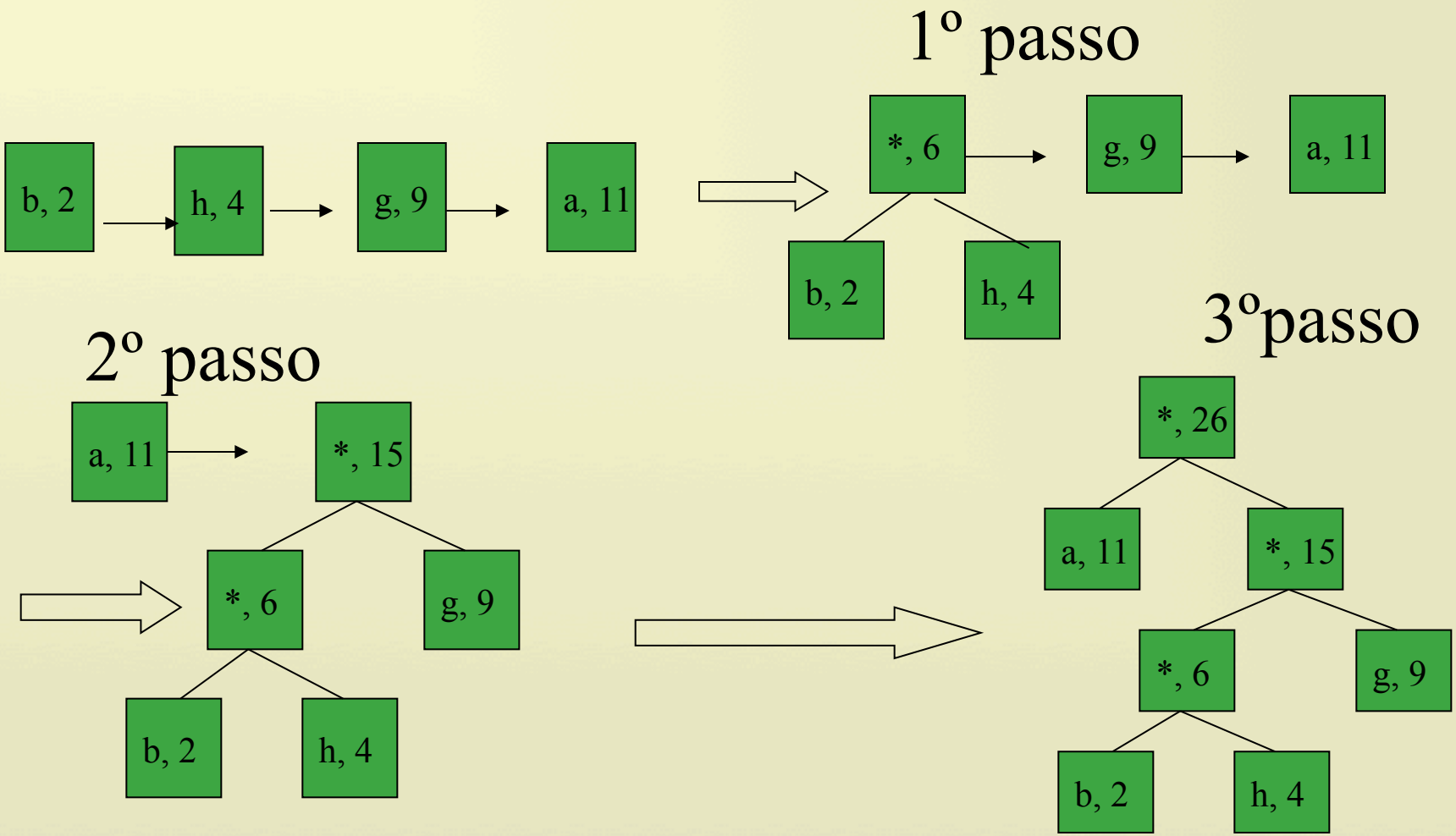
4 símbolos : 2 bps

Tamanho da mensagem: 52 bits.

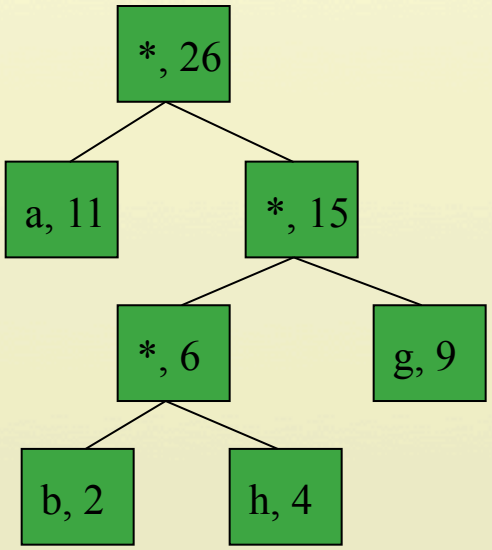
Como é que se constrói um código eficiente?

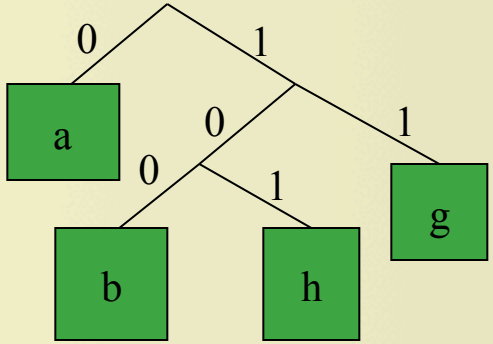
Qual é a entropia? *Sol: 1.755 bps*

Código Huffman e árvore binária



Huffman e árvore binária





símbolo	No. ocorrências	código
a	11	0
g	9	11
h	4	101
b	2	100

Tamanho da mensagem

Mensagem com 2bits/símbolo: 26x2=52 bits

Mensagem codificada com: 11x1+9x2+4x3+2x3=47 bits

Número médio de bits por símbolo: 47/26=1.807 bps

Valor próximo da entropia da mensagem

Símbolos mais frequentes código com menor número de bits.

Rácio de compressão=(original/ codificada)

Descodificar

100011

⇓

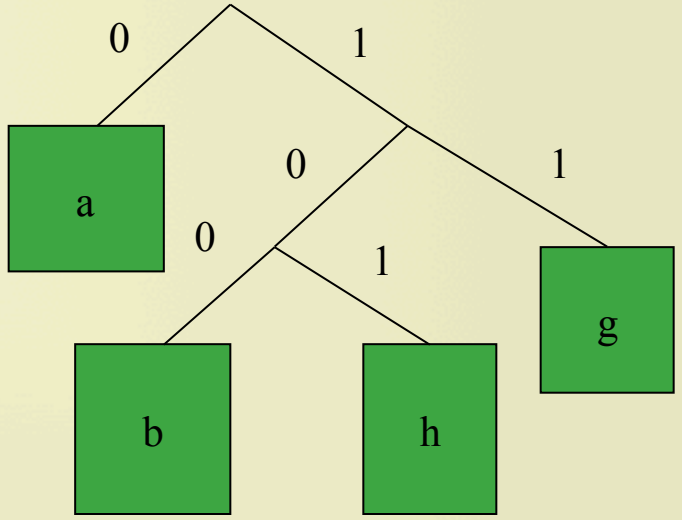
100011 b

⇓

100011 **a**

⇓

100011 **g**



Do nó raiz para os nós terminais

Código Huffman: propriedades

- Símbolos mais frequentes código com menor número de bits.
- Código não ambíguo
- Código de descodificação instantânea

Exercício

Mensagem e probabilidade de ocorrência de cada símbolo

Símbolos	Probabilidades
A	0.05
B	0.2
C	0.1
D	0.05
E	0.3
K	0.2
Z	0.1

Entropia ?
sol: 2.54 bps

Código de Huffman ?
sol: 3 símbolos com 2 bits, 1 símbolo 3 bits, 1 símbolo 4 bits, 2 símbolos com 5 bits

Utilizando o código Huffman , calcule o valor médio de bits por símbolo
sol: 2.6 bps