Construção de códigos instantâneos

- Dada uma fonte com q símbolos, pretende-se obter q palavras-código com comprimentos $\ell_1, \ell_2, \ldots, \ell_q$ tal que o código resultante seja intantâneo.
- Começamos por ordenar os comprimentos ℓ_i por ordem crescente.
- Em seguida tomamos palavras dos comprimentos em causa sucessivamente mínimas na ordem lexicográfica de X^*, fixada uma ordem para o alfabeto, nunca recuando na ordem lexicográfica e excluindo todas as palavras que tenham como prefixo outras já escolhidas.
- Naturalmente, nem sempre o processo será bem sucedido, dependendo da cardinalidade do alfabeto X e dos números ℓ_i, como veremos adiante.

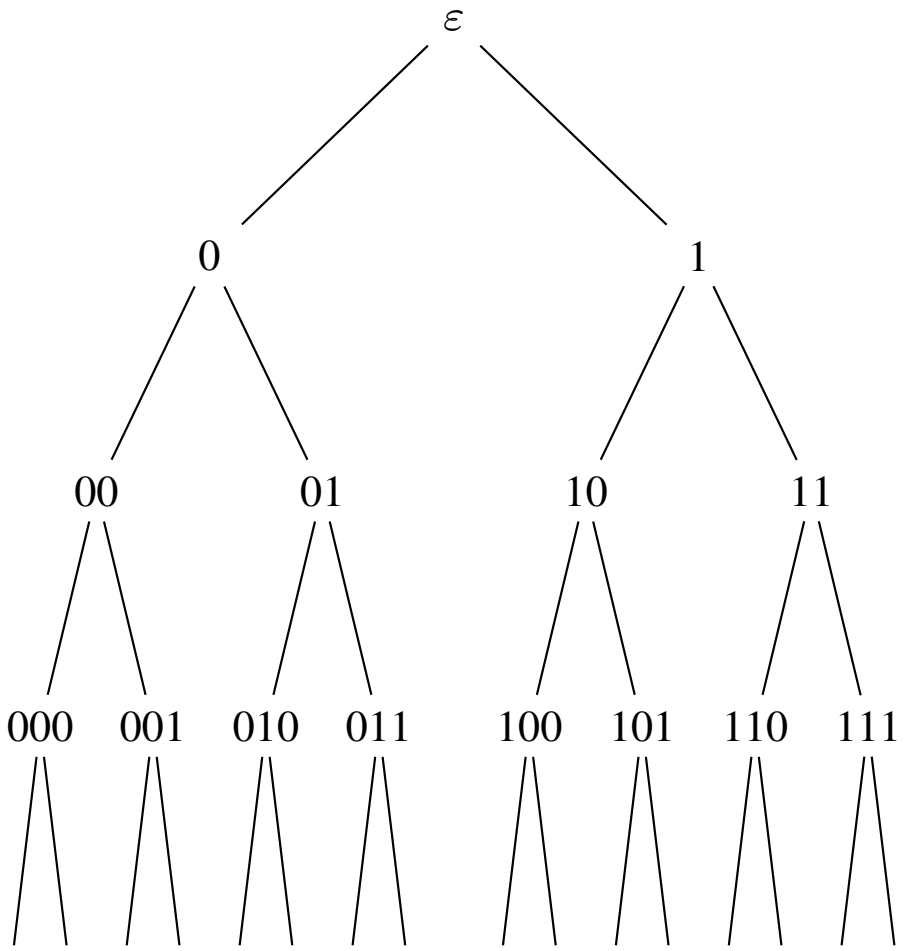
Exemplos

1. Código prefixo binário com comprimentos $3, 2, 3, 2, 2$:
   começamos por reordenar os comprimentos $2, 2, 2, 3, 3$ e tomamos $00, 01, 10, 110, 111$.
2. Código prefixo ternário com palavras-código de comprimentos $2, 3, 1, 1, 2$: ou seja de comprimentos por ordem crescente $1, 1, 2, 2, 3$, para o que tomamos $0, 1, 20, 21, 220$.
3. Código prefixo binário com comprimentos $2, 3, 2, 2, 2$? Aqui os comprimentos por ordem crescente são $2, 2, 2, 2, 3$, pelo que esgotamos as palavras de comprimento 2, $00, 01, 10, 11$ e qualquer palavra de comprimento 3 no alfabeto $\{0, 1\}$ tem uma daquelas como prefixo. Logo não há códigos prefixos nas condições pretendidas.

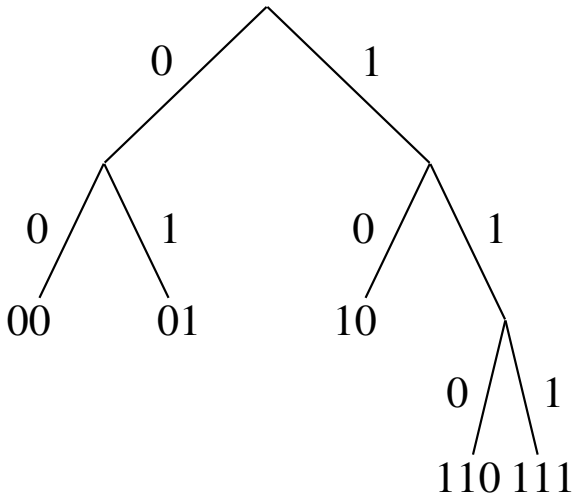
Representação em árvore de códigos prefixos

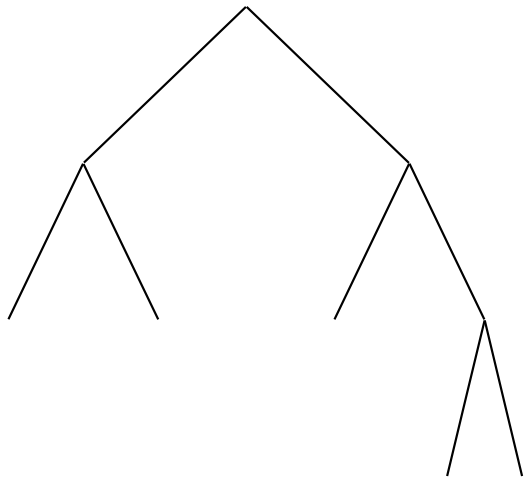
- Notemos que o monoide A^* pode ser parcialmente ordenado pela relação de *ser prefixo*: $u \leqslant v$ se u é prefixo de v.
- O diagrama do conjunto parcialmente ordenado A^* é uma *árvore* no sentido de que tem uma *raiz*, o elemento mínimo ε (a palavra vazia), e o conjunto de elementos menores ou iguais a qualquer elemento é uma cadeia, pelo que há um "único caminho" de ε até qualquer elemento de A^*.

Por exemplo, para o alfabeto binário, obtemos a árvore binária completa (aqui, como habitualmente, invertida e, naturalmente, truncada):

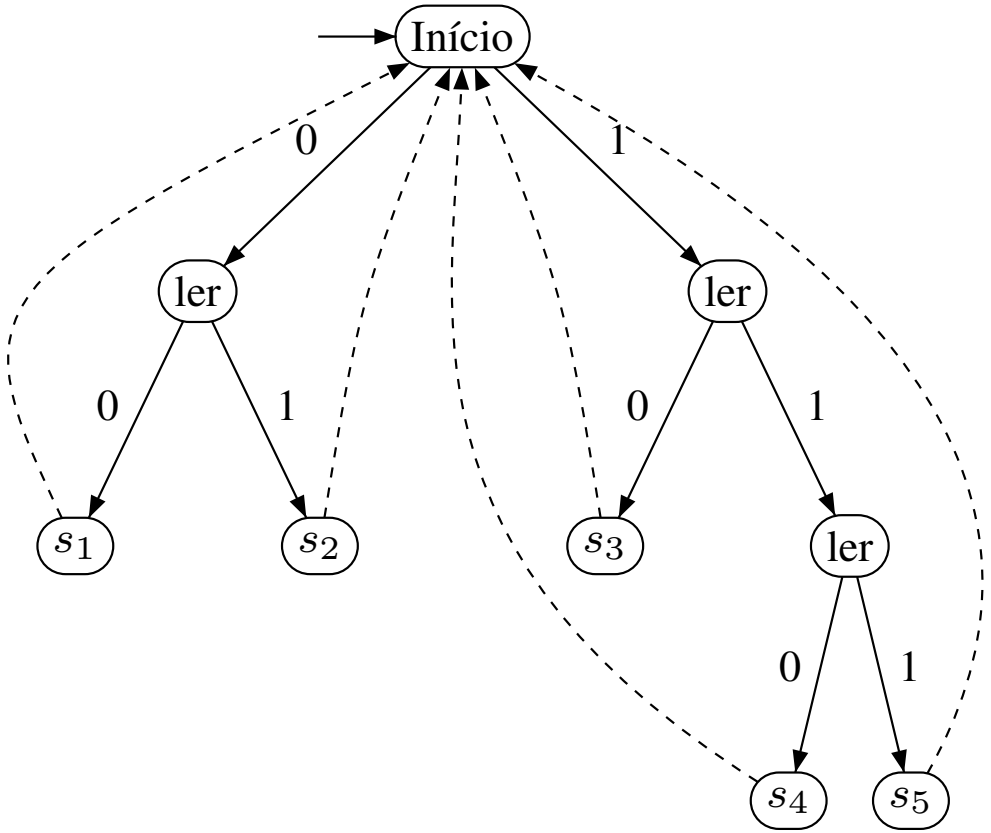


- A escolha de um código prefixo sobre o alfabeto A corresponde à escolha de pontos na correspondente árvore tais que nenhum deles fica no caminho único para outro a partir da raiz.
- Por exemplo, a seguinte árvore binária representa o código prefixo $00, 01, 10, 110, 111$:





- A decifração de uma mensagem escrita num código prefixo faz-se lendo na respetiva árvore os símbolos da mensagem codificada, regressando à raiz sempre que seja encontrada uma palavra do código.
- Por outras palavras, no exemplo anterior temos o seguinte *autómato*, onde as transições a tracejado não usam qualquer símbolo da mensagem codificada:

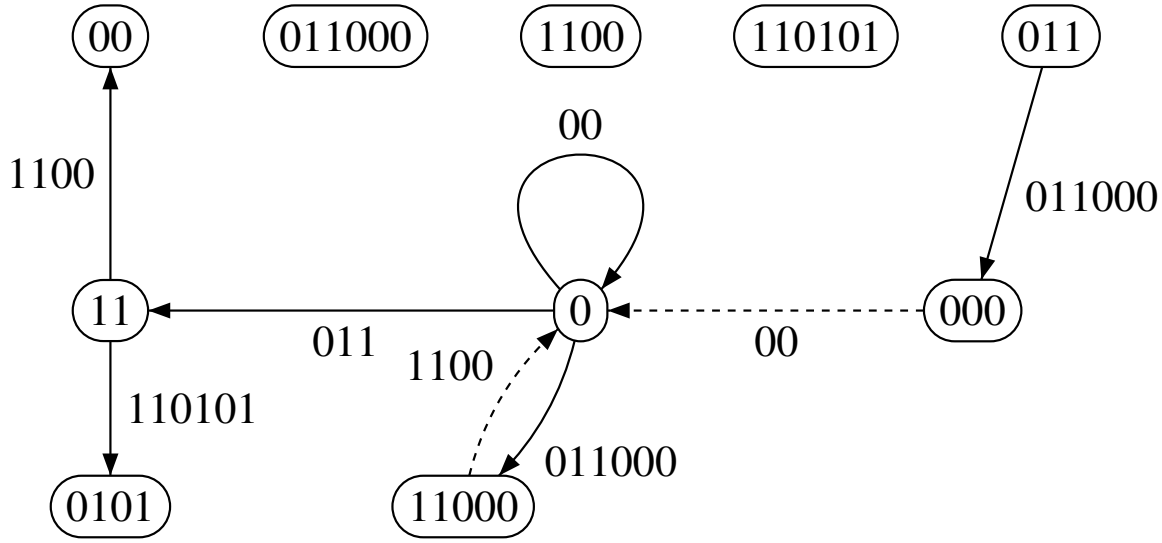


O algoritmo de Sardinas-Patterson

- Como determinar se uma sequência finita de palavras pode ser usada como codificação descodificável, i.e., como código?
- Seja $L \subseteq A^* \setminus \{\varepsilon\}$ um subconjunto finito.
- Definimos indutivamente um grafo $\Gamma(L)$ com dois tipos de arestas, a cheio e a tracejado, começando com L como conjunto de vértices:
  - se v é um vértice, $u \neq \varepsilon$ e $vu = w \in L$, então u é um vértice e temos uma aresta a cheio $v \xrightarrow{w} u$ etiquetada por w;
  - se $v \notin L$ é um vértice, $w \in L$ e $v = wu$, então u é um vértice e temos uma aresta a tracejado $v \overset{w}{\dashrightarrow} u$ etiquetada por w.

Exemplo

- Considere-se o conjunto $L = \{00, 011000, 1100, 110101, 011\}$. Eis o grafo correspondente:



- O caminho $(011, 000, 0, 0, 0, 11000, 0, 0, 11, 00)$ representa a seguinte igualdade:

$$011 \cdot 00 \cdot 00 \cdot 011000 \cdot 011 \cdot 00 = 011000 \cdot 00 \cdot 1100 \cdot 00 \cdot 1100,$$

ou seja, $\overline{011}\ \overline{00}\ \overline{0}\ \overline{0}\ \overline{0}\ \overline{1100}\ \overline{0}\ \overline{0}\ \overline{11}\ \overline{00}$, o que mostra que a descodificação é ambígua.

- Mais geralmente:

  Teorema 3.2 *A descodificação usando L como lista das palavras-código é possível se e só se no grafo associado $\Gamma(L)$ não existir nenhum caminho que comece e termine em vértices de L.*

  A demonstração consiste em generalizar o exemplo acima mostrando como um caminho representa uma certa igualdade.

  Temos dois tipos de arestas:

  - uma aresta a cheio $u \xrightarrow{c} v$ corresponde a uma igualdade $uv = c$, com $c \in L$;
  - uma aresta a tracejado $u \overset{c}{\dashrightarrow} v$ corresponde a uma igualdade $u = cv$, com $c \in L$.

Assim, uma igualdade não trivial $c_1 c_2 \cdots c_m = c'_1 c'_2 \cdots c'_n$, com $c_i, c'_j \in L$, que não seja obtida por concatenação de igualdades mais curtas deste tipo, pode ser lida no grafo $\Gamma(L)$ começando no vértice etiquetado pela mais curta das palavras c_1 e c'_1, digamos c_1, e seguindo a aresta a cheio etiquetada pela mais longa destas duas palavras, que vai dar num vértice v_1;
em seguida, se digamos $c_1 c_2$ for mais curta que c'_1, do vértice v_1 seguimos a aresta a tracejado etiquetada por c_2;
caso contrário, seguimos de v_1 a aresta a cheio etiquetada por c_2;
e assim por diante até esgotarmos os c_i e os c_j.

Os vértices intermédios que assim aparecem correspondem às sucessivas sobreposições entre os c_i e os c'_j.

- Note-se que, para um código prefixo L não há arestas em $\Gamma(L)$ e os únicos vértices são os de L.

Sensibilidade a erros em carateres do alfabeto do código

- Os códigos que temos vindo a considerar são em geral de *comprimento variável*, i.e., nem todas as palavras do código têm o mesmo comprimento.
- Os códigos em que todas as palavras-código têm o mesmo comprimento n também se dizem *códigos-bloco de comprimento* n.
- Os primeiros têm a vantagem de permitir obter mensagens mais curtas, ou seja comprimir a informação. No entanto, a alteração de um só carácter numa mensagem codificada pode ter efeitos globais na descodificação da mensagem.
- Pelo contrário, para os códigos-bloco, a alteração de um carácter na mensagem codificada só afeta a descodificação do carácter correspondente ao bloco que o contém.

Código ASCII

- Um exemplo de código-bloco muito utilizado é o código ASCII:

^@	0000000	^P	0010000		0100000	0	0110000	@	1000000	P	1010000	‘	1100000	p	1110000
^A	0000001	^Q	0010001	!	0100001	1	0110001	A	1000001	Q	1010001	a	1100001	q	1110001
^B	0000010	^R	0010010	"	0100010	2	0110010	B	1000010	R	1010010	b	1100010	r	1110010
^C	0000011	^S	0010011	#	0100011	3	0110011	C	1000011	S	1010011	c	1100011	s	1110011
^D	0000100	^T	0010100	$	0100100	4	0110100	D	1000100	T	1010100	d	1100100	t	1110100
^E	0000101	^U	0010101	%	0100101	5	0110101	E	1000101	U	1010101	e	1100101	u	1110101
^F	0000110	^V	0010110	&	0100110	6	0110110	F	1000110	V	1010110	f	1100110	v	1110110
^G	0000111	^H	0010111	’	0100111	7	0110111	G	1000111	W	1010111	g	1100111	w	1110111
^H	0001000	^X	0011000	(	0101000	8	0111000	H	1001000	X	1011000	h	1101000	x	1111000
^I	0001001	^Y	0011001	)	0101001	9	0111001	I	1001001	Y	1011001	i	1101001	y	1111001
^J	0001010	^Z	0011010	*	0101010	:	0111010	J	1001010	Z	1011010	j	1101010	z	1111010
^K	0001011	^[	0011011	+	0101011	;	0111011	K	1001011	[	1011011	k	1101011	{	1111011
^L	0001100	^\	0011100	,	0101100	<	0111100	L	1001100	\	1011100	l	1101100	|	1111100
^M	0001101	^]	0011101	-	0101101	=	0111101	M	1001101	]	1011101	m	1101101	}	1111101
^N	0001110	^^	0011110	.	0101110	>	0111110	N	1001110	^	1011110	n	1101110	~	1111110
^O	0001111	^_	0011111	/	0101111	?	0111111	O	1001111	_	1011111	o	1101111	^?	1111111

A desigualdade de Kraft

Já observámos que nem sempre existe um código prefixo cujas palavras-código tenham comprimentos prescritos sobre um dado alfabeto.

Teorema 3.3 (Desigualdade de Kraft) *Uma condição necessária e suficiente para que que existam códigos prefixo num alfabeto com r letras que possuam q palavras-código de comprimentos $\ell_1, \ell_2, \ldots, \ell_q$ é que se verifique a desigualdade*

$$K = \sum_{i=1}^{q} r^{-\ell_i} \leqslant 1.$$

Prova. Sem perda de generalidade, podemos supor que $\ell_1 \leqslant \ldots \leqslant \ell_q$. Sejam $w_1, \ldots, w_q$ palavras sobre um alfabeto A com r letras tais que $|w_i| = \ell_i$ $(i = 1, \ldots, q)$.
Vejamos que condições numéricas resultam de supor que w_1 não é prefixo de w_2:

- o número de palavras de A^* de comprimento ℓ_2 é r^{ℓ_2};
- dessas, $r^{\ell_2 - \ell_1}$ têm w_1 como prefixo;
- logo há $r^{\ell_2} - r^{\ell_2 - \ell_1}$ palavras de A^* de comprimento ℓ_2 que não têm w_1 como prefixo.

A existência de $w_2 \in A^*$ nestas condições é portanto equivalente à desigualdade
$r^{\ell_2} - r^{\ell_2 - \ell_1} \geqslant 1$.
Mais geralmente, supondo que $w_1, \ldots, w_{i-1} \in A^*$ forma um código prefixo, o número de $w_i \in A^*$ tais que $\{w_1, \ldots, w_i\}$ é ainda um código prefixo é dado pela expressão

$$r^{\ell_i} - r^{\ell_i - \ell_{i-1}} - r^{\ell_i - \ell_{i-2}} - \ldots - r^{\ell_i - \ell_1},$$

e portanto a condição necessária e suficiente para existência de um tal w_i é que aquele número seja $\geqslant 1$, i.e., multiplicando ambos os membros por $r^{-\ell_i}$:

$$r^{-\ell_i} + r^{-\ell_{i-1}} + \cdots + r^{-\ell_1} \leqslant 1.$$

Juntando todas estas condições necessárias e suficientes para a passagem de um código prefixo $\{w_1, \ldots, w_{i-1}\}$ a outro $\{w_1, \ldots, w_i\}$ satisfazendo as condições sobre os comprimentos, obtemos como condição necessária e suficiente para existência do código prefixo pretendido a conjunção das desigualdades

$$r^{-\ell_i} + r^{-\ell_{i-1}} + \cdots + r^{-\ell_1} \leqslant 1$$

com $i = 2, \ldots, q$.
Note-se que cada uma destas desigualdades implica a anterior, pelo que a sua conjunção é equivalente à última delas, correspondente a $i = q$:

$$r^{-\ell_q} + r^{-\ell_{q-1}} + \cdots + r^{-\ell_1} \leqslant 1,$$

que é precisamente a desigualdade de Kraft. □

O teorema de McMillan

Será que poderemos fazer melhor com códigos arbitrários? A resposta, negativa, segue do Teorema de Kraft juntamente com o seguinte resultado.

Teorema 3.4 (McMillan) *Seja $C \subseteq A^*$ um código com q palavras sobre um alfabeto com r letras e sejam ℓ_i $(i = 1, \ldots, q)$ os comprimentos dos seus elementos. Então verifica-se a desigualdade de Kraft:*

$$K = \sum_{i=1}^{q} r^{-\ell_i} \leqslant 1.$$

Prova. Consideremos a expressão

$$\left(\sum_{i=1}^{q} r^{-\ell_i} \right)^n = (r^{-\ell_1} + r^{-\ell_2} + \cdots + r^{-\ell_q})^n.$$

Expandindo a potência, obtemos q^n parcelas da forma $r^{-\ell_{i_1} - \ell_{i_2} - \cdots - \ell_{i_n}} = r^{-k}$, onde $k = \ell_{i_1} + \ell_{i_2} + \cdots + \ell_{i_n}$.

Para cada k, seja N_k o número de termos com valor r^{-k} na expansão da nossa expressão.
Note-se que uma condição necessária para que $N_k > 0$ é que $k \in [n, nm]$, onde $m = \max_i \ell_i$.
Logo

$$\left(\sum_{i=1}^{q} r^{-\ell_i} \right)^n = \sum_{k=n}^{nm} N_k r^{-k}.$$

Ora, sendo C um código, uma palavra (de comprimento k) de A^* que seja produto de elementos de C só admite uma factorização como tal produto.
Segue-se que $N_k \leqslant r^k$ pois N_k representa precisamente o número total de produtos de n elementos de C que têm comprimento k.
Concluímos que, para todo o $n \geqslant 1$, temos

$$\left(\sum_{i=1}^{q} r^{-\ell_i} \right)^n \leqslant \sum_{k=n}^{nm} r^k r^{-k} = \sum_{k=n}^{nm} 1 = nm - n + 1 \leqslant nm$$

donde segue que $\sum_{i=1}^{q} r^{-\ell_i} \leqslant (nm)^{\frac{1}{n}}$.
Como $m \geqslant 1$, tomando o limite quando $n \to \infty$, concluímos que $\sum_{i=1}^{q} r^{-\ell_i} \leqslant 1$. $\square$

- Logo, em qualquer análise de códigos em que só intervenham os comprimentos das palavras, basta considerar códigos prefixos.