

# Algoritmos e Estruturas de Dados III

Aula 8.3 – Métodos de Dicionário  
Conceitos, LZ77 e LZ78

Prof. Hayala Curto  
2022



**PUC Minas**

# Métodos de Dicionário



# Métodos de Dicionário

- Os símbolos (ou conjunto de símbolos) são substituídos por códigos a partir de um “dicionário”
- Os códigos possuem tamanho fixo
- Os dicionários podem ser estáticos ou dinâmicos
- Ex: LZ77 / LZ78 / LZW

# Métodos de Dicionário

Exemplo Simples (Dicionário Estático)

**Codificação:** considere que a palavra “algoritmo” tem o código 1025. Sua codificação será

00000000010000000001

⇒ Redução de 72 (9 bytes) para 20 bits

considere que a palavra “abc” não está no dicionário.

Ela não será codificada, e será necessário armazenar, além do bit extra, o número de bytes não codificados:

100000011011000010110001001100011

tam = 3 | a = 97 | b = 98 | c = 99

⇒ Aumento de 24 (3 bytes) para 33 bits

# Métodos de Dicionário

## Exemplo Simples (Dicionário Estático)

- um dicionário que possui um código para as palavras da língua portuguesa. Para 500.000 palavras, seriam necessários 19 bits para cada código ( $2^{19} = 524.288$ )
- o algoritmo irá substituir cada palavra do arquivo pelo seu código. Caso a palavra não existir no dicionário, ela é colocada sem substituição no arquivo de saída
- o arquivo compactado irá conter códigos e palavras

**Possível problema:** como saber diferenciar entre os códigos e as palavras (todos são conjuntos de bits)

**Solução:** usar um bit extra para informar se é um código ou não

# Métodos de Dicionário

## Exemplo Simples (Dicionário Estático)

No caso de muitas falhas (palavras não encontradas no dicionário), o tamanho do arquivo irá aumentar !!!

Perguntas:

- o que aconteceria se o texto estivesse em inglês?

Esse método de dicionário estático não é bom para compactadores de uso geral mas pode adequado para compactadores específicos

# Métodos de Dicionário

- Na verdade, os algoritmos de uso geral utilizam dicionários dinâmicos (métodos adaptativos)
- Os dicionários são construídos à medida que o texto vai sendo lido. Se uma palavra não existe no dicionário ela pode ser inserida, e palavras “fora de uso” podem ser removidas do dicionário
- Na década de 70, Abraham Lempel e Jacob Ziv criaram a base para os algoritmos de compactação baseados em dicionários dinâmicos
- A grande maioria dos métodos de dicionários existentes hoje são extensões ou modificações dos algoritmos de Lempel-Ziv (LZ)
- Boa parte dos programas comerciais (zip, compress, formato gif, etc.) se baseiam em variações/combinações destes algoritmos

LZ77





# LZ77

Parte do conteúdo já lido é usado como dicionário  
Utiliza o conceito de “janela deslizante” (sliding window)

Arquivo

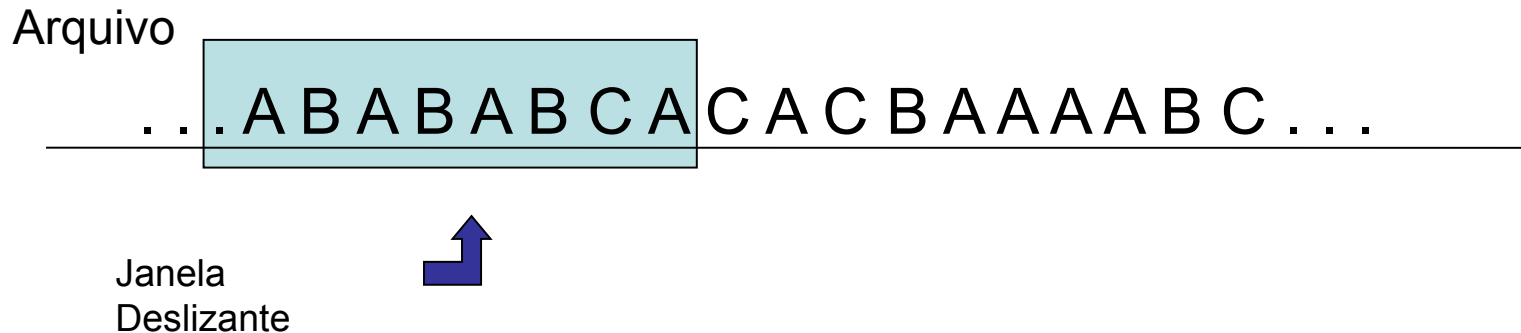
---

... A B A B A B C A C A C B A A A B C ...

---

# LZ77

Parte do conteúdo já lido é usado como dicionário  
Utiliza o conceito de “janela deslizante” (sliding window)



# LZ77

Parte do conteúdo já lido é usado como dicionário  
Utiliza o conceito de “janela deslizante” (sliding window)

Arquivo

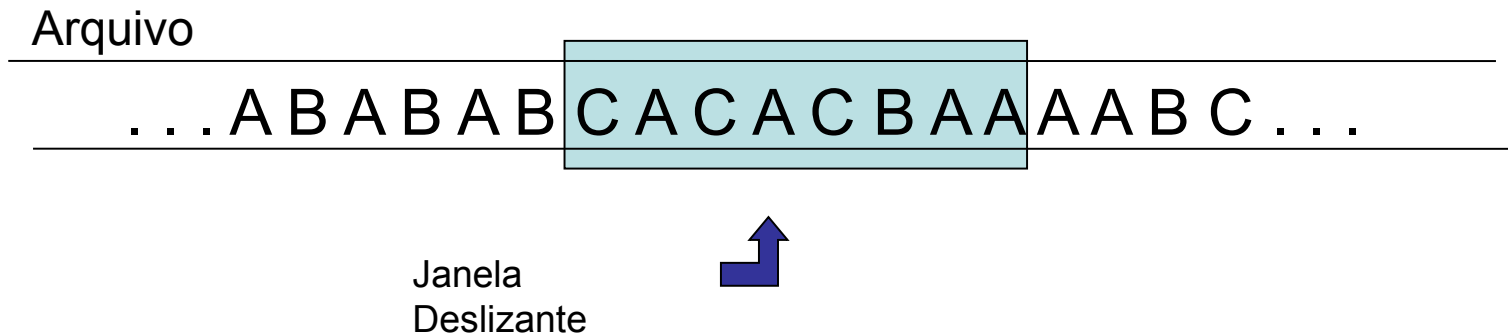
... A B A B A B C A C A C B A A A B C ...

Janela  
Deslizante



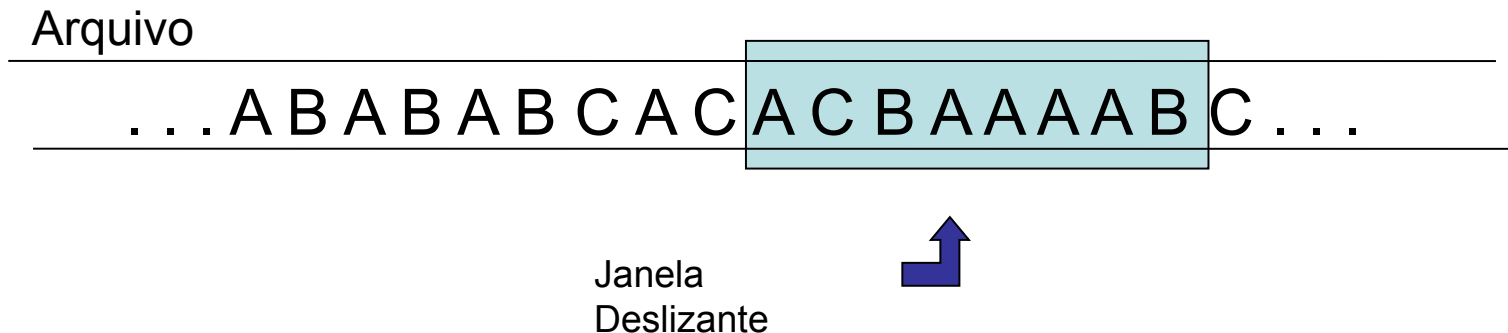
# LZ77

Parte do conteúdo já lido é usado como dicionário  
Utiliza o conceito de “janela deslizante” (sliding window)



# LZ77

Parte do conteúdo já lido é usado como dicionário  
Utiliza o conceito de “janela deslizante” (sliding window)

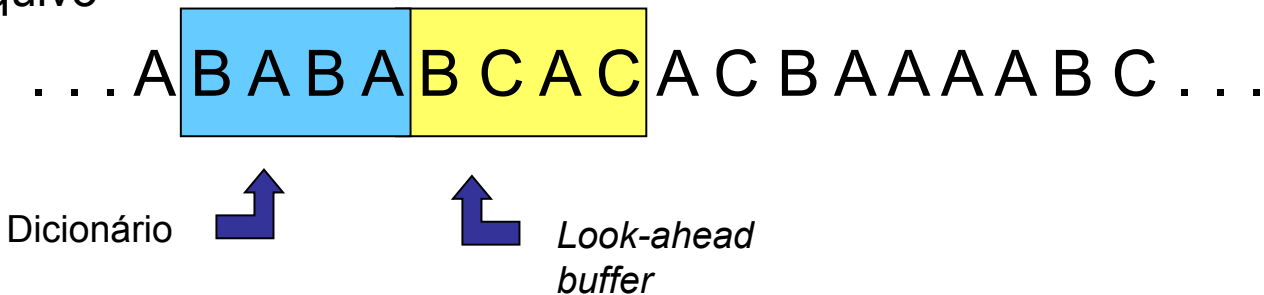


# LZ77

A janela deslizante é dividida em duas partes

- 1ª Parte: símbolos previamente lidos (dicionário ou *search buffer*)
- 2ª Parte: símbolos que serão processados (*look-ahead buffer*)

Arquivo

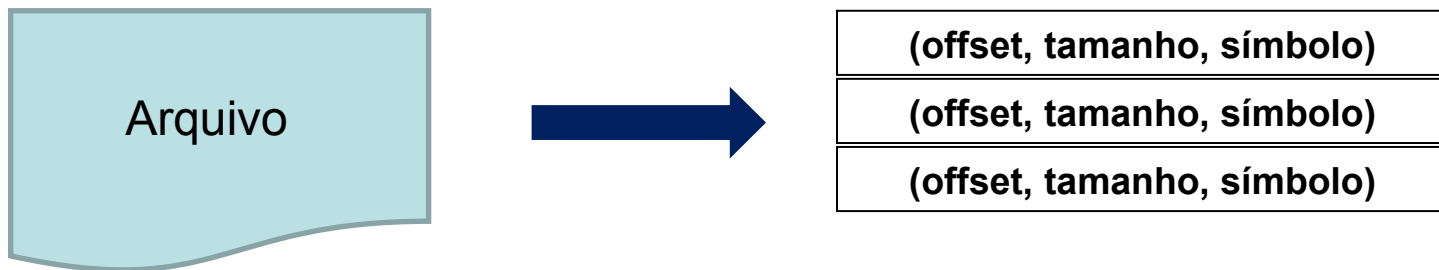


# LZ77 - Compactação



# Métodos de Dicionário

- Compactação
- Para cada conjunto de símbolos do buffer, o algoritmo procura pelo maior casamento possível no dicionário
- Se for encontrado, gera como saída um token contendo a posição no dicionário (offset), o tamanho da string e o próximo símbolo do buffer.
- Se não for encontrado, escreve 0 para o offset, 0 para o tamanho e o símbolo não encontrado

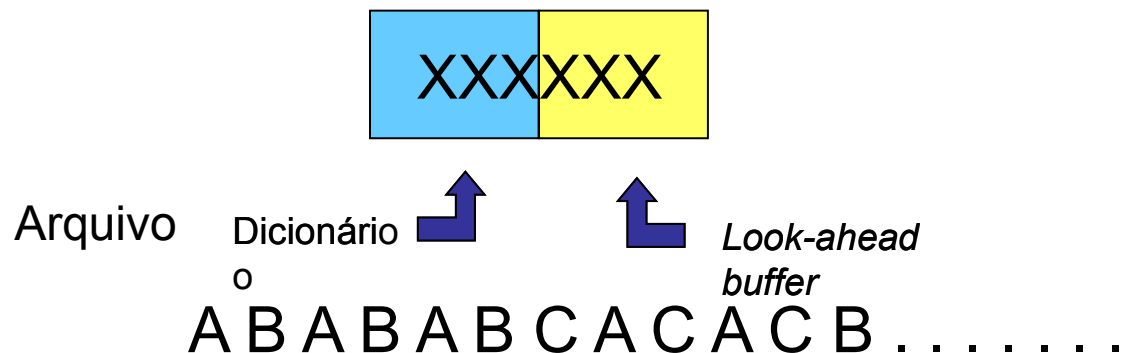




# LZ77

## Exemplo de compactação

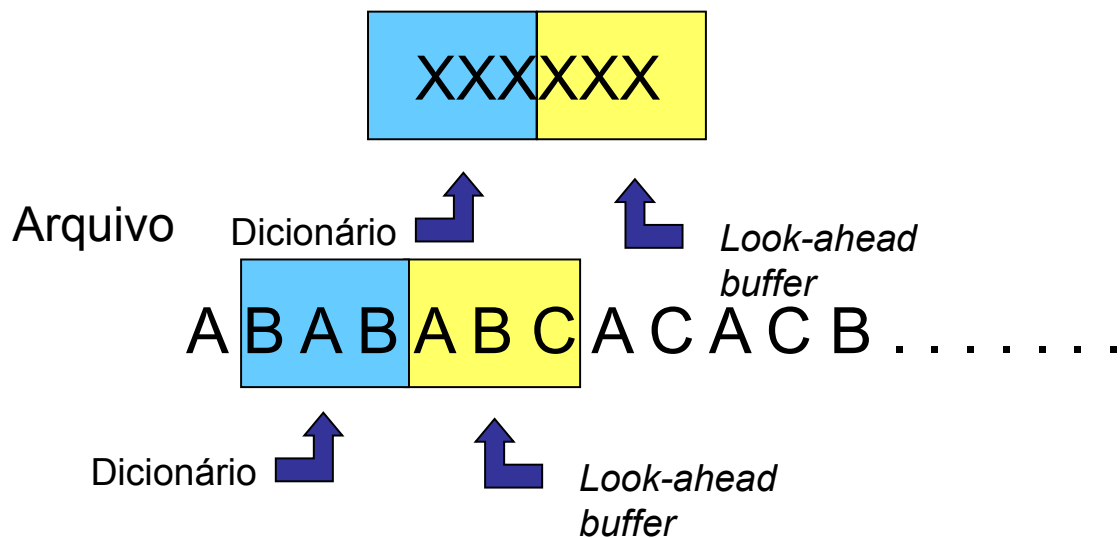
Tamanho da Janela = 6 (Dicionário = 3 / *Look-ahead buffer* = 3)



# LZ77

## Exemplo de compactação

Tamanho da Janela = 6 (Dicionário = 3 / *Look-ahead buffer* = 3)



## LZ77

## ● Exemplo de Compactação

# Arquivo



# LZ77

- Exemplo de Compactação

Arquivo

A B A B A B C A C A C B . . . . .



?



(0, 0, A)

Não usa nenhum símbolo do dicionário  
e escreve o próximo símbolo

# LZ77

- Exemplo de Compactação

Arquivo

A diagram illustrating the LZ77 sliding window. It shows a sequence of characters: A B A B A B C A C A C B . . . . . The first three characters 'A B A' are enclosed in a yellow box. To the left of this box is a light blue box. A red arrow points from the light blue box to the yellow box. A blue callout box points from the yellow box to the text '(0, 0, A)' and 'Avança a janela 1 posição'.

A B A B A B C A C A C B . . . . .

(0, 0, A)

Avança a janela 1 posição

# LZ77

- Exemplo de Compactação

Arquivo

A B A B A B C A C A C B . . . . .

(0, 0, A)

# LZ77

- Exemplo de Compactação

Arquivo

AB A B A B C A C A C B . . . . .



?

(0, 0, A)

# LZ77

- Exemplo de Compactação

Arquivo

AB A B A B C A C A C B . . . . .



?



(0, 0, A)

(0, 0, B)

Não usa nenhum símbolo do dicionário  
e escreve o próximo símbolo



# LZ77

- Exemplo de Compactação

Arquivo



(0, 0, A)

(0, 0, B)

Avança a janela 1 posição

# LZ77

- Exemplo de Compactação

Arquivo

ABABAB C A C A C B . . . . .



?

(0, 0, A)

(0, 0, B)

# LZ77

- Exemplo de Compactação

Arquivo



(0, 0, A)

(0, 0, B)

# LZ77

- Exemplo de Compactação

Arquivo

**AB****AB**A B C A C A C B . . . . .



(0, 0, A)

(0, 0, B)

**(2, 2, A)**

Volta 2 posições do dicionário, usa 2 símbolos e escreve o próximo símbolo do buffer

# LZ77

- Exemplo de Compactação

Arquivo



(0, 0, A)

(0, 0, B)

(2, **2**, A)

Avança a janela 1+(X) posições  
Onde X é o tamanho gravado na “trinca” anterior  
Aqui, X = 2

# LZ77

- Exemplo de Compactação

Arquivo

A B A B A B C A C A C B . . . . .

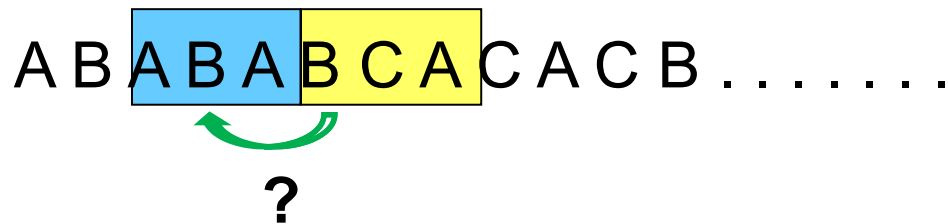
(0, 0, A)

(0, 0, B)

(2, 2, A)

- Exemplo de Compactação

Arquivo



(0, 0, A)

(0, 0, B)

(2, 2, A)

# LZ77

- Exemplo de Compactação

Arquivo

A B A B A B C A C A C B . . . . .



(0, 0, A)

(0, 0, B)

(2, 2, A)



# LZ77

## ● Exemplo de Compactação

Arquivo

A B A B A B C A C A C B . . . . .



(0, 0, A)

(0, 0, B)

(2, 2, A)


(2, 1, C)

Volta 2 posições do dicionário,  
usa 1 símbolo  
e escreve o próximo símbolo  
do buffer

## ● Exemplo de Compactação

Arquivo

A B A B A B C A C A C B . . . . .

A diagram illustrating the LZ77 sliding window. The sequence of characters is A B A B A B C A C A C B followed by ellipses. A blue box highlights the first 'A' and 'B' of the third match (A B). A yellow box highlights the 'B' and 'C' of the second match (B C). A red arrow points from the end of the yellow box to the start of the blue box, indicating the window shift.

(0, 0, A)

(0, 0, B)

(2, 2, A)

(2, 1, C)

Avança a janela 1+(X) posições  
Onde X é o tamanho gravado na “trinca” anterior  
Aqui, X = 1

- Exemplo de Compactação

Arquivo

A B A B **A B C** **A C A** C B . . . . .

(0, 0, A)

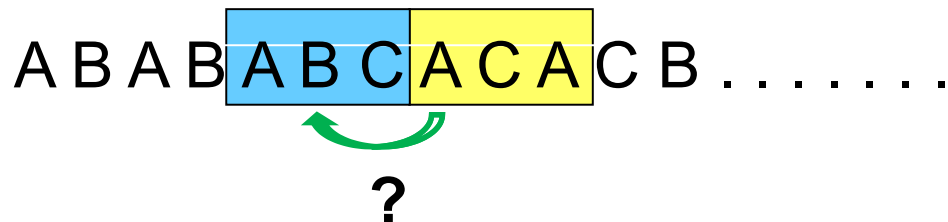
(0, 0, B)

(2, 2, A)

(2, 1, C)

- Exemplo de Compactação

Arquivo



(0, 0, A)

(0, 0, B)

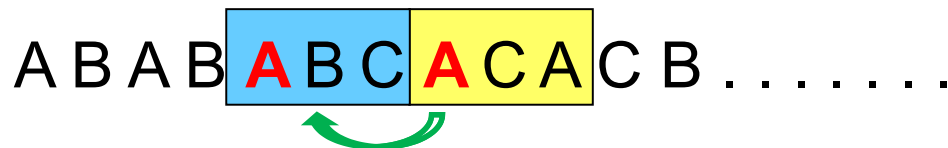
(2, 2, A)

(2, 1, C)

- Exemplo de Compactação

Arquivo

A B A B **A B C** **A C A** C B . . . . .



(0, 0, A)

(0, 0, B)

(2, 2, A)

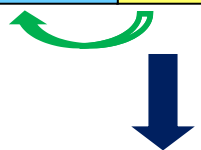
(2, 1, C)

# LZ77

## ● Exemplo de Compactação

Arquivo

A B A B **A B C** **A C A** C B . . . . .



(0, 0, A)

(0, 0, B)

(2, 2, A)

(2, 1, C)

**(3, 1, C)**

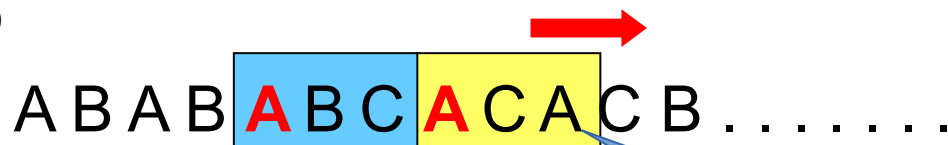
Volta 3 posições do dicionário,  
usa 1 símbolo e escreve  
o próximo símbolo do buffer

# LZ77

## ● Exemplo de Compactação

Arquivo

A B A B A B C A C A C B . . . . .



(0, 0, A)

(0, 0, B)

(2, 2, A)

(2, 1, C)

(3, 1, C)

Avança a janela 1+(X) posições  
Onde X é o tamanho gravado na  
“trinca” anterior  
Aqui, X = 1

# LZ77

- Exemplo de Compactação

Arquivo

A B A B A B **C A C** **A C B** . . . . .

(0, 0, A)

(0, 0, B)

(2, 2, A)

(2, 1, C)

(3, 1, C)



# LZ77

- Exemplo de Compactação

Arquivo

A B A B A B C A C A C B . . . . .



?

(0, 0, A)

(0, 0, B)

(2, 2, A)

(2, 1, C)

(3, 1, C)

# LZ77

- Exemplo de Compactação

Arquivo

A B A B A B C A C A C B . . . . .



(0, 0, A)

(0, 0, B)

(2, 2, A)

(2, 1, C)

(3, 1, C)

# LZ77

## ● Exemplo de Compactação

Arquivo

A B A B A B C A C A C B . . . . .



(0, 0, A)

(0, 0, B)

(2, 2, A)

(2, 1, C)

(3, 1, C)

**(2, 2, B)**

Volta 2 posições do dicionário,  
usa 2 símbolos e escreve  
o próximo símbolo do buffer

# LZ77

- Exemplo de Compactação

Arquivo original

A B A B A B C A C A C B . . . . .



Arquivo compactado

0 0 A 0 0 B 2 2 A 2 1 C 3 1 C 2 2 B . . . . .

ou

0 0 1 0 0 2 2 2 1 2 1 3 3 1 3 2 2 2 . . . . .

Obs:    A  $\equiv$  1    B  $\equiv$  2    C  $\equiv$  3

# LZ77 - Descompactação



# LZ77

- Exemplo de descompactação

Arquivo compactado

0 0 A 0 0 B 2 2 A 2 1 C 3 1 C 2 2 B . . . . .

# LZ77

- Exemplo de descompactação

Arquivo compactado

0 0 A 0 0 B 2 2 A 2 1 C 3 1 C 2 2 B . . . . .

A

# LZ77

- Exemplo de descompactação

Arquivo compactado

0 0 A 0 0 B 2 2 A 2 1 C 3 1 C 2 2 B . . . . .

A B



# LZ77

- Exemplo de descompactação

Arquivo compactado

0 0 A 0 0 B 2 2 A 2 1 C 3 1 C 2 2 B . . . . .

A B A B A

# LZ77

- Exemplo de descompactação

Arquivo compactado

0 0 A 0 0 B 2 2 A 2 1 C 3 1 C 2 2 B . . . . .

A B A B A B C

# LZ77

- Exemplo de descompactação

Arquivo compactado

0 0 A 0 0 B 2 2 A 2 1 C 3 1 C 2 2 B . . . . .

A B A B A B C A C

# LZ77

- Exemplo de descompactação

Arquivo compactado

0 0 A 0 0 B 2 2 A 2 1 C 3 1 C 2 2 B . . . . .

A B A B A B C A C A C B

# LZ77

- Exemplo de descompactação

Arquivo compactado

**0 0 A 0 0 B 2 2 A 2 1 C 3 1 C 2 2 B . . . . .**

Arquivo original:

A B A B A B C A C A C B

# LZ77

- Descompactação
  - Basta manter um buffer do tamanho do dicionário
  - Para cada **token** do arquivo verifica-se qual a sequência de símbolos correspondente no buffer escrevendo como saída essa sequência e o símbolo armazenado no **token**, deslocando o buffer para inserir esses novos símbolos



# LZ77

- Funciona bem quando padrões repetidos aparecem próximos uns dos outros no texto

# LZ77

- Funciona bem quando padrões repetidos aparecem próximos uns dos outros no texto
- Compressão depende do tamanho do **token**
  - Offset:  $\log_2(D)$ , onde D é o tamanho do Dicionário
  - Tamanho:  $\log_2(B-1)$ , onde B é o tamanho do Buffer
  - Símbolo:  $\log_2(\Sigma)$ , onde  $\Sigma$  é o tamanho do alfabeto
  - Tipicamente, o tamanho do dicionário é de milhares de bytes (para aumentar a chance de se encontrar um casamento), enquanto o *look-ahead buffer* tem algumas dezenas de bytes. Exemplo:
    - Offset: 12 bits (4K símbolos)
    - Tamanho: 5 bits (31+1 símbolos)
    - Símbolo: 8 bits (Tabela ASCII estendida)



LZ78



# LZ78

- Cria-se um dicionário de prefixos (ou sequências) já encontrados no conteúdo
- As sequências são formadas por símbolos que podem representar bytes, caracteres, cores, etc. (isto é, qualquer unidade básica de informação)
- Não utiliza o conceito de “janela deslizante”
- O dicionário é construído dinamicamente a partir das sequências encontradas e é armazenado em uma tabela na memória primária
- Ao final do processo de compactação (ou de descompactação) o dicionário é descartado

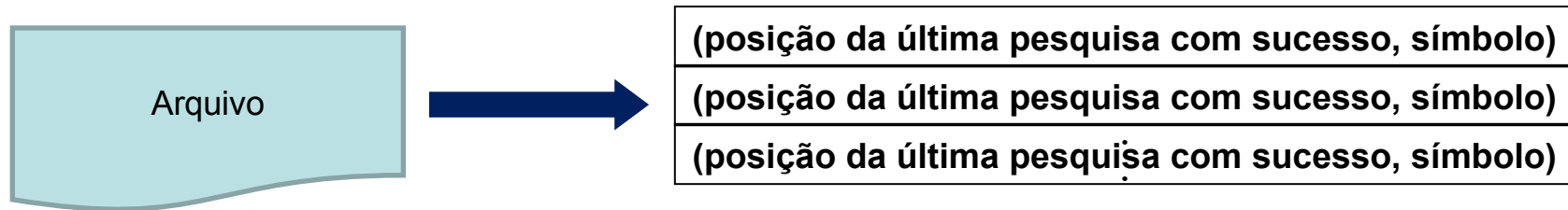
# LZ78

- Exemplo de Dicionário
  - Exemplo usando sequências de caracteres

Posição	Prefixo
1	A
2	B
3	AB
4	C
5	BC
6	ABC

# LZ78

- Informações importantes para execução
  - Símbolo lido: byte, caractere, etc
  - Prefixo (ou sequência) atual
    - Inicialmente deve o prefixo é nulo
  - Posição da tabela da última pesquisa com sucesso
    - Usar o valor ZERO para representar o caso em que ainda não se realizou nenhuma pesquisa na tabela



# LZ78

- Compactação (Repetir até o final do arquivo)
  - Ler o próximo símbolo do arquivo e concatenar o símbolo do final da sequência (ou prefixo)
  - Fazer uma busca na tabela para encontrar o prefixo
    - Se o prefixo for encontrado, atualizar o indicador de posição da última pesquisa com sucesso na tabela
    - Se o prefixo não for encontrado,
      - Gerar um token contendo a posição da última pesquisa com sucesso e o último símbolo lido (responsável pela falha na busca)
      - Inserir o prefixo na tabela para evitar novas falhas
      - Reinicializar o prefixo (com o valor NULO) e o indicador de posição de pesquisa com sucesso (com o valor ZERO)

# LZ78

ABAB CAB CAB CAAB CAB . . . . .

- Símbolo:
- Prefixo :

Posição	Prefixo
1	
2	
3	
4	
5	
6	

# LZ78

Ler



**A** B A B C A B C A B C A A B C A B . . . . .

- Símbolo: A
- Prefixo :

Posição	Prefixo
1	
2	
3	
4	
5	
6	

# LZ78

**A** B A B C A B C A B C A A B C A B . . . . .

Anexar



- Símbolo: A
- Prefixo : A

Posição	Prefixo
1	
2	
3	
4	
5	
6	



# LZ78

**A** B A B C A B C A B C A A B C A B . . . . .

- Símbolo: A
- Prefixo : A

Buscar



Posição	Prefixo
1	
2	
3	
4	
5	
6	

# LZ78

**A** B A B C A B C A B C A A B C A B . . . . .

- Símbolo: A
- Prefixo : A

Buscar



• Posição	• Prefixo
• 1	
• 2	
• 3	
• 4	
• 5	
• 6	

**FALHA !!!**

# LZ78

**A** B A B C A B C A B C A A B C A B . . . . .

- Símbolo: A
- Prefixo : A

• Posição	• Prefixo
• 1	
• 2	
• 3	
• 4	
• 5	
• 6	

Gerar  
token

(0, A)

**Dicionário**

# LZ78

**A** B A B C A B C A B C A A B C A B . . . . .

- Símbolo: A
- Prefixo : A

(0, A)

• Posição	• Prefixo
• 1	A
• 2	
• 3	
• 4	
• 5	
• 6	

Inserir  
prefixo



# LZ78

**A** B A B C A B C A B C A A B C A B . . . . .

– Símbolo:

– Prefixo :  Reinicializar

**(0, A)**

Posição	Prefixo
1	A
2	
3	
4	
5	
6	

# LZ78

**A** **B** A B C A B C A B C A A B C A B . . . . .

Ler



- Símbolo: B
- Prefixo :

(0, A)

Posição	Prefixo
1	A
2	
3	
4	
5	
6	

# LZ78

**A B** A B C A B C A B C A A B C A B . . . . .

Anexar



- Símbolo: B
- Prefixo : B

(0, A)

Posição	Prefixo
1	A
2	
3	
4	
5	
6	

# LZ78

**A** **B** A B C A B C A B C A A B C A B . . . . .

- Símbolo: B
- Prefixo : B

Buscar

(0, A)



Posição	Prefixo
1	A
2	
3	
4	
5	
6	




# LZ78


**A** **B** A B C A B C A B C A A B C A B . . . . .

- Símbolo: B
- Prefixo : B

Buscar

(0, A)



Posição	Prefixo
1	A
2	
3	
4	
5	
6	

**FALHA !!!**

# LZ78

**A** **B** A B C A B C A B C A A B C A B . . . . .

- Símbolo: B
- Prefixo : B

Posição	Prefixo
1	A
2	
3	
4	
5	
6	

Gerar  
token

(0, A)

(0, B)

# LZ78

**A B** A B C A B C A B C A A B C A B . . . . .

- Símbolo: B
- Prefixo : B

(0, A)

(0, B)

Posição	Prefixo
1	A
2	B
3	
4	
5	
6	

Inserir  
prefixo



# LZ78

**A B** A B C A B C A B C A A B C A B . . . . .

- Símbolo:
- Prefixo :  Reinicializar

(0, A)

(0, B)

Posição	Prefixo
1	A
2	B
3	
4	
5	
6	

# LZ78

**A B A** B C A B C A B C A A B C A B . . . . .

Ler



- Símbolo: A
- Prefixo :

(0, A)

(0, B)

Posição	Prefixo
1	A
2	B
3	
4	
5	
6	

# LZ78

**A B A** B C A B C A B C A A B C A B . . . . .

Anexar



- Símbolo: A
- Prefixo : A

(0, A)

(0, B)

Posição	Prefixo
1	A
2	B
3	
4	
5	
6	

# LZ78

**A B A** B C A B C A B C A A B C A B . . . . .

- Símbolo: A
- Prefixo : A

Buscar



Posição	Prefixo
1	A
2	B
3	
4	
5	
6	

(0, A)  
(0, B)

# LZ78


**A B A** B C A B C A B C A A B C A B . . . . .

- Símbolo: A
- Prefixo : A

Buscar

(0, A)

(0, B)



Posição	Prefixo
1	A
2	B
3	
4	
5	
6	

Sucesso



# LZ78

**A B A** B C A B C A B C A A B C A B . . . . .

- Símbolo: A
- Prefixo : A

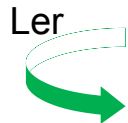
(0, A)

(0, B)

	Posição	Prefixo
Última Pos. de Sucesso →	1	A
	2	B
	3	
	4	
	5	
	6	

# LZ78

**A B A B** C A B C A B C A A B C A B . . . . .



- Símbolo: B
- Prefixo : A

(0, A)

(0, B)

	Posição	Prefixo
Última Pos. de Sucesso →	1	A
	2	B
	3	
	4	
	5	
	6	

# LZ78

**A B A B** C A B C A B C A A B C A B . . . . .

Anexar



- Símbolo: B
- Prefixo : AB

(0, A)

(0, B)

Última Pos.  
de Sucesso

Posição	Prefixo
1	A
2	B
3	
4	
5	
6	

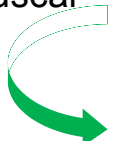
# LZ78

**A B A B** C A B C A B C A A B C A B . . . . .

- Símbolo: B
- Prefixo : AB

Buscar

Última Pos.  
de Sucesso



• Posição	• Prefixo
• 1	A
• 2	B
• 3	
• 4	
• 5	
• 6	

(0, A)


(0, B)

# LZ78


**A B A B** C A B C A B C A A B C A B . . . . .

- Símbolo: B
- Prefixo : AB

Buscar



Última Pos.  
de Sucesso



• Posição	• Prefixo
• 1	A
• 2	B
• 3	
• 4	
• 5	
• 6	

(0, A)

(0, B)

**FALHA !!!**

# LZ78

**A B A B** C A B C A B C A A B C A B . . . . .

- Símbolo: B
- Prefixo : AB

	• Posição	• Prefixo
Última Pos. de Sucesso	• 1	A
	• 2	B
	• 3	
	• 4	
	• 5	
	• 6	

Gerar  
token

(0, A)  
(0, B)  
(1, B)

# LZ78

**A B A B** C A B C A B C A A B C A B . . . . .

- Símbolo: B
- Prefixo : AB

(0, A)  
(0, B)  
(1, B)

	• Posição	• Prefixo
Última Pos. de Sucesso	• 1	A
	• 2	B
	• 3	AB
	• 4	
	• 5	
	• 6	

Inserir  
prefixo



# LZ78

**A B A B** C A B C A B C A A B C A B . . . . .

- Símbolo:
- Prefixo :  Reinicializar

(0, A)  
(0, B)  
(1, B)

• Posição	• Prefixo
• 1	A
• 2	B
• 3	AB
• 4	
• 5	
• 6	



# LZ78

**A B A B C** A B C A B C A A B C A B . . . . .

Ler



- Símbolo: C
- Prefixo :

(0, A)  
(0, B)  
(1, B)

• Posição	• Prefixo
• 1	A
• 2	B
• 3	AB
• 4	
• 5	
• 6	

# LZ78

**A B A B C** A B C A B C A A B C A B . . . . .

Anexar



- Símbolo: C
- Prefixo : C

(0, A)  
(0, B)  
(1, B)

• Posição	• Prefixo
• 1	A
• 2	B
• 3	AB
• 4	
• 5	
• 6	

# LZ78

**A B A B C** A B C A B C A A B C A B . . . . .

- Símbolo: C
- Prefixo : C

Buscar



• Posição	• Prefixo
• 1	A
• 2	B
• 3	AB
• 4	
• 5	
• 6	

(0, A)  
(0, B)  
(1, B)

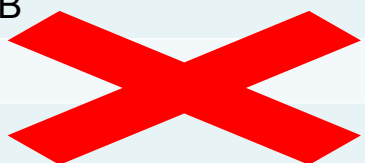
# LZ78

**A B A B C** A B C A B C A A B C A B . . . . .

- Símbolo: C
- Prefixo : C

Buscar



• Posição	• Prefixo
• 1	A
• 2	B
• 3	AB
• 4	
• 5	
• 6	

(0, A)  
(0, B)  
(1, B)

**FALHA !!!**

# LZ78

**A B A B C** A B C A B C A A B C A B . . . . .

- Símbolo: C
- Prefixo : C

• Posição	• Prefixo
• 1	A
• 2	B
• 3	AB
• 4	
• 5	
• 6	

Gerar  
token

(0, A)  
(0, B)  
(1, B)  
(0, C)

# LZ78

**A B A B C** A B C A B C A A B C A B . . . . .

- Símbolo: C
- Prefixo : C

• Posição	• Prefixo
• 1	A
• 2	B
• 3	AB
• 4	C
• 5	
• 6	

(0, A)  
(0, B)  
(1, B)  
(0, C)

Inserir  
prefixo



# LZ78

**A B A B C** A B C A B C A A B C A B . . . . .

- Símbolo:



Reinicializar

- Prefixo :

• Posição	• Prefixo
• 1	A
• 2	B
• 3	AB
• 4	C
• 5	
• 6	

(0, A)  
(0, B)  
(1, B)  
(0, C)

# LZ78

**A B A B C A** B C A B C A A B C A B . . . . .

Ler



- Símbolo: A
- Prefixo :

• Posição	• Prefixo
• 1	A
• 2	B
• 3	AB
• 4	C
• 5	
• 6	

(0, A)  
(0, B)  
(1, B)  
(0, C)



# LZ78

**A B A B C A** B C A B C A A B C A B . . . . .

Anexar



- Símbolo: A
- Prefixo : A

• Posição	• Prefixo
• 1	A
• 2	B
• 3	AB
• 4	C
• 5	
• 6	

(0, A)  
(0, B)  
(1, B)  
(0, C)

# LZ78

**A B A B C A** B C A B C A A B C A B . . . . .

- Símbolo: A
- Prefixo : A

Buscar



• Posição	• Prefixo
• 1	A
• 2	B
• 3	AB
• 4	C
• 5	
• 6	


(0, A)  
(0, B)  
(1, B)  
(0, C)

# LZ78

**A B A B C A** B C A B C A A B C A B . . . . .

- Símbolo: A
- Prefixo : A

Buscar



Posição	Prefixo
1	A
2	B
3	AB
4	C
5	
6	

Sucesso

(0, A)  
(0, B)  
(1, B)  
(0, C)

# LZ78

**A B A B C A** B C A B C A A B C A B . . . . .

- Símbolo: A
- Prefixo : A

(0, A)  
(0, B)  
(1, B)  
(0, C)

	Posição	Prefixo
Última Pos. de Sucesso →	1	A
	2	B
	3	AB
	4	C
	5	
	6	

# LZ78

**A B A B C A B** C A B C A A B C A B . . . . .

Ler



- Símbolo: B
- Prefixo : A

(0, A)  
(0, B)  
(1, B)  
(0, C)

Última Pos.  
de Sucesso

Posição	Prefixo
1	A
2	B
3	AB
4	C
5	
6	

# LZ78

**A B A B C A B** C A B C A A B C A B . . . . .

Anexar



- Símbolo: B
- Prefixo : AB

(0, A)  
(0, B)  
(1, B)  
(0, C)

Última Pos.  
de Sucesso


Posição	Prefixo
1	A
2	B
3	AB
4	C
5	
6	

# LZ78


**A B A B C A B** C A B C A A B C A B . . . . .

- Símbolo: B
- Prefixo : AB

Buscar



Última Pos.  
de Sucesso



Posição	Prefixo
1	A
2	B
3	AB
4	C
5	
6	

(0, A)  
(0, B)  
(1, B)  
(0, C)

# LZ78

**A B A B C A B** C A B C A A B C A B . . . . .

- Símbolo: B
- Prefixo : AB

Buscar

Última Pos.  
de Sucesso

Posição	Prefixo
1	A
2	B
3	AB
4	C
5	
6	

(0, A)  
(0, B)  
(1, B)  
(0, C)

Sucesso



# LZ78

**A B A B C A B** C A B C A A B C A B . . . . .

- Símbolo: B
- Prefixo : AB

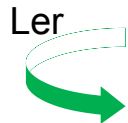
(0, A)  
(0, B)  
(1, B)  
(0, C)

Posição	Prefixo
1	A
2	B
3	AB
4	C
5	
6	

Última Pos.  
de Sucesso →

# LZ78

**A B A B C A B C** A B C A A B C A B . . . . .



- Símbolo: C

- Prefixo : AB

Posição	Prefixo
1	A
2	B
3	AB
4	C
5	
6	

Última Pos.  
de Sucesso



(0, A)  
(0, B)  
(1, B)  
(0, C)

# LZ78

**A B A B C A B C** A B C A A B C A B . . . . .

Anexar



- Símbolo: C
- Prefixo : ABC

(0, A)  
(0, B)  
(1, B)  
(0, C)

Posição	Prefixo
1	A
2	B
3	AB
4	C
5	
6	

Última Pos.  
de Sucesso



# LZ78

**A B A B C A B C** A B C A A B C A B . . . . .

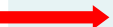
- Símbolo: C
- Prefixo : ABC

Buscar



Posição	Prefixo
1	A
2	B
3	AB
4	C
5	
6	

Última Pos.  
de Sucesso




(0, A)  
(0, B)  
(1, B)  
(0, C)

# LZ78

**A B A B C A B C** A B C A A B C A B . . . . .

- Símbolo: C
- Prefixo : ABC

Buscar



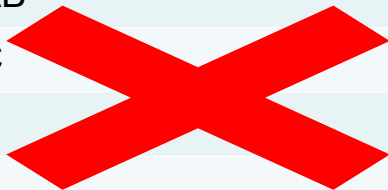
Posição	Prefixo
1	A
2	B
3	AB
4	C
5	
6	

Última Pos.  
de Sucesso



(0, A)  
(0, B)  
(1, B)  
(0, C)

**FALHA !!!**



# LZ78

**A B A B C A B C** A B C A A B C A B . . . . .

- Símbolo: C
- Prefixo : ABC

Posição	Prefixo
1	A
2	B
3	AB
4	C
5	
6	

Última Pos.  
de Sucesso



Gerar  
token



(0, A)  
(0, B)  
(1, B)  
(0, C)  
(3, C)

# LZ78

**A B A B C A B C** A B C A A B C A B . . . . .

- Símbolo: C
- Prefixo : ABC

Posição	Prefixo
1	A
2	B
3	AB
4	C
5	ABC
6	

Última Pos.  
de Sucesso



Inserir  
prefixo



(0, A)  
(0, B)  
(1, B)  
(0, C)  
(3, C)

# LZ78

**A B A B C A B C** A B C A A B C A B . . . . .

- Símbolo:
- Prefixo :  Reinicializar

Posição	Prefixo
1	A
2	B
3	AB
4	C
5	ABC
6	

(0, A)  
(0, B)  
(1, B)  
(0, C)  
(3, C)



# LZ78

**A B A B C A B C A** B C A A B C A B . . . . .

Ler



- Símbolo: A
- Prefixo :

Posição	Prefixo
1	A
2	B
3	AB
4	C
5	ABC
6	

(0, A)  
(0, B)  
(1, B)  
(0, C)  
(3, C)

# LZ78

**A B A B C A B C A** B C A A B C A B . . . . .

Anexar



- Símbolo: A
- Prefixo : A

Posição	Prefixo
1	A
2	B
3	AB
4	C
5	ABC
6	


(0, A)  
(0, B)  
(1, B)  
(0, C)  
(3, C)

# LZ78

**A B A B C A B C A** B C A A B C A B . . . . .

- Símbolo: A
- Prefixo : A

Buscar



Posição	Prefixo
1	A
2	B
3	AB
4	C
5	ABC
6	


(0, A)  
(0, B)  
(1, B)  
(0, C)  
(3, C)

# LZ78

**A B A B C A B C A** B C A A B C A B . . . . .

- Símbolo: A
- Prefixo : A

Buscar



Posição	Prefixo
1	A
2	B
3	AB
4	C
5	ABC
6	

Sucesso

(0, A)  
(0, B)  
(1, B)  
(0, C)  
(3, C)

# LZ78

**A B A B C A B C A** B C A A B C A B . . . . .

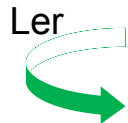
- Símbolo: A
- Prefixo : A

	• Posição	• Prefixo
Última Pos. de Sucesso →	• 1	A
	• 2	B
	• 3	AB
	• 4	C
	• 5	ABC
	• 6	

(0, A)  
(0, B)  
(1, B)  
(0, C)  
(3, C)

# LZ78

**A B A B C A B C A B** C A A B C A B . . . . .



- Símbolo: B
- Prefixo : A

	• Posição	• Prefixo
Última Pos. de Sucesso →	• 1	A
	• 2	B
	• 3	AB
	• 4	C
	• 5	ABC
	• 6	

(0, A)  
(0, B)  
(1, B)  
(0, C)  
(3, C)

# LZ78

**A B A B C A B C A B** C A A B C A B . . . . .

Anexar



- Símbolo: B
- Prefixo : AB

Última Pos.  
de Sucesso

• Posição	• Prefixo
• 1	A
• 2	B
• 3	AB
• 4	C
• 5	ABC
• 6	


(0, A)  
(0, B)  
(1, B)  
(0, C)  
(3, C)

# LZ78


**A B A B C A B C A B** C A A B C A B . . . . .

- Símbolo: B
- Prefixo : AB

Buscar



Última Pos.  
de Sucesso



• Posição	• Prefixo
• 1	A
• 2	B
• 3	AB
• 4	C
• 5	ABC
• 6	

(0, A)  
(0, B)  
(1, B)  
(0, C)  
(3, C)




# LZ78


**A B A B C A B C A B** C A A B C A B . . . . .

- Símbolo: B
- Prefixo : AB

Buscar



Última Pos.  
de Sucesso



• Posição	• Prefixo
• 1	A
• 2	B
• 3	AB
• 4	C
• 5	ABC
• 6	

Sucesso



(0, A)  
(0, B)  
(1, B)  
(0, C)  
(3, C)

# LZ78

**A B A B C A B C A B** C A A B C A B . . . . .

- Símbolo: B
- Prefixo : AB

• Posição	• Prefixo
• 1	A
• 2	B
• 3	AB
• 4	C
• 5	ABC
• 6	

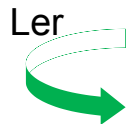
Última Pos.  
de Sucesso



(0, A)  
(0, B)  
(1, B)  
(0, C)  
(3, C)

# LZ78

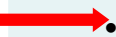
**A B A B C A B C A B C** A A B C A B . . . . .



- Símbolo: C
- Prefixo : AB

• Posição	• Prefixo
• 1	A
• 2	B
• 3	AB
• 4	C
• 5	ABC
• 6	

Última Pos.  
de Sucesso



(0, A)  
(0, B)  
(1, B)  
(0, C)  
(3, C)

# LZ78

**A B A B C A B C A B C** A A B C A B . . . . .

Anexar



- Símbolo: C
- Prefixo : ABC

• Posição	• Prefixo
• 1	A
• 2	B
• 3	AB
• 4	C
• 5	ABC
• 6	

Última Pos.  
de Sucesso



(0, A)  
(0, B)  
(1, B)  
(0, C)  
(3, C)

# LZ78

**A B A B C A B C A B C** A A B C A B . . . . .

- Símbolo: C
- Prefixo : ABC

Buscar



• Posição	• Prefixo
• 1	A
• 2	B
• 3	AB
• 4	C
• 5	ABC
• 6	

Última Pos.  
de Sucesso




(0, A)  
(0, B)  
(1, B)  
(0, C)  
(3, C)

# LZ78

**A B A B C A B C A B C** A A B C A B . . . . .

- Símbolo: C
- Prefixo : ABC

Buscar



• Posição	• Prefixo
• 1	A
• 2	B
• 3	AB
• 4	C
• 5	ABC
• 6	

Última Pos.  
de Sucesso



(0, A)  
(0, B)  
(1, B)  
(0, C)  
(3, C)

Sucesso



# LZ78

**A B A B C A B C A B C** A A B C A B . . . . .

- Símbolo: C
- Prefixo : ABC

• Posição	• Prefixo
• 1	A
• 2	B
• 3	AB
• 4	C
• 5	ABC
• 6	

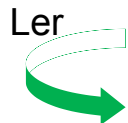
Última Pos.  
de Sucesso



(0, A)  
(0, B)  
(1, B)  
(0, C)  
(3, C)

# LZ78

**A B A B C A B C A B C A** A B C A B . . . . .



- Símbolo: A
- Prefixo : ABC

• Posição	• Prefixo
• 1	A
• 2	B
• 3	AB
• 4	C
• 5	ABC
• 6	

(0, A)  
(0, B)  
(1, B)  
(0, C)  
(3, C)

Última Pos.  
de Sucesso





# LZ78

**A B A B C A B C A B C A** A B C A B . . . . .

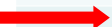
Anexar



- Símbolo: A
- Prefixo : ABCA

• Posição	• Prefixo
• 1	A
• 2	B
• 3	AB
• 4	C
• 5	ABC
• 6	

Última Pos.  
de Sucesso



(0, A)  
(0, B)  
(1, B)  
(0, C)  
(3, C)

# LZ78

**A B A B C A B C A B C A** A B C A B . . . . .

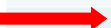
- Símbolo: A
- Prefixo : ABCA

Buscar



• Posição	• Prefixo
• 1	A
• 2	B
• 3	AB
• 4	C
• 5	ABC
• 6	

Última Pos.  
de Sucesso



(0, A)  
(0, B)  
(1, B)  
(0, C)  
(3, C)

# LZ78

**A B A B C A B C A B C A** A B C A B . . . . .

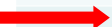
- Símbolo: A
- Prefixo : ABCA

Buscar



• Posição	• Prefixo
• 1	A
• 2	B
• 3	AB
• 4	C
• 5	ABC
• 6	

Última Pos.  
de Sucesso



**FALHA !!!**

(0, A)  
(0, B)  
(1, B)  
(0, C)  
(3, C)

# LZ78

**A B A B C A B C A B C A** A B C A B . . . . .

- Símbolo: A
- Prefixo : ABCA

• Posição	• Prefixo
• 1	A
• 2	B
• 3	AB
• 4	C
• 5	ABC
• 6	

Última Pos.  
de Sucesso →

Gerar  
token

(0, A)  
(0, B)  
(1, B)  
(0, C)  
(3, C)  
(5, A)

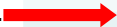
# LZ78

**A B A B C A B C A B C A** A B C A B . . . . .

- Símbolo: A
- Prefixo : ABCA

• Posição	• Prefixo
• 1	A
• 2	B
• 3	AB
• 4	C
• 5	ABC
• 6	ABCA

Última Pos.  
de Sucesso



(0, A)  
(0, B)  
(1, B)  
(0, C)  
(3, C)  
(5, A)

Inserir  
prefixo

# LZ78

**A B A B C A B C A B C A** A B C A B . . . . .

- Símbolo:



Reinicializar

- Prefixo :

• Posição	• Prefixo
• 1	A
• 2	B
• 3	AB
• 4	C
• 5	ABC
• 6	ABCA

(0, A)  
(0, B)  
(1, B)  
(0, C)  
(3, C)  
(5, A)

# LZ78

**A B A B C A B C A B C A A B C A B . . . . .**

- Símbolo:
- Prefixo : **ABCA**

• Posição	• Prefixo
• 1	A
• 2	B
• 3	AB
• 4	C
• 5	ABC
• 6	ABCA

(0, A)  
(0, B)  
(1, B)  
(0, C)  
(3, C)  
(5, A)

# LZ78

**A B A B C A B C A B C A A B C A B . . . . .**

- Símbolo:
- Prefixo : **ABCA**

• Posição	• Prefixo
• 1	A
• 2	B
• 3	AB
• 4	C
• 5	ABC
• 6	ABCA

Gerar  
token

(0, A)  
(0, B)  
(1, B)  
(0, C)  
(3, C)  
(5, A)  
(6, B)



# LZ78

Arquivo original

A B A B C A B C A B C A . . . . .



Arquivo compactado

0 A 0 B 1 B 0 C 3 C 5 A . . . . .

ou

0 1 0 2 1 2 0 3 3 3 5 1 . . . . .

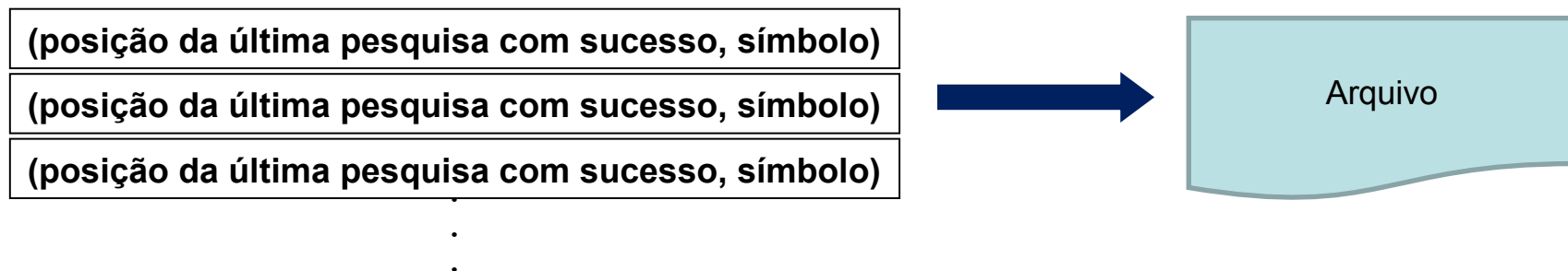
Obs: A ≡ 1    B ≡ 2    C ≡ 3

• Posição	• Prefixo
• 1	A
• 2	B
• 3	AB
• 4	C
• 5	ABC
• 6	ABCA

(0, A)  
(0, B)  
(1, B)  
(0, C)  
(3, C)  
(5, A)

# LZ78

- Descompactação
  - A medida que os **tokens** vão sendo lidos, o dicionário vai sendo reconstruído da mesma forma
  - Para cada **token** do arquivo compactado, é gerada como saída a sequência do dicionário indicada pelo campo posição concatenada com o símbolo armazenado no **token**



# LZ78

- Compressão depende do tamanho do **token**
  - Posição:  $\log_2(D)$ , onde  $D$  é o tamanho do Dicionário
  - Símbolo:  $\log_2(\Sigma)$ , onde  $\Sigma$  é o tamanho do alfabeto
- Características do dicionário
  - Uma implementação eficiente utiliza uma árvore digital (trie) para armazenar o dicionário
  - Normalmente o dicionário começa com sequências pequenas, mas sequências maiores vão sendo inseridas a medida que o arquivo vai sendo processado
  - O dicionário armazena prefixos de todo o arquivo (melhor que o LZ77 cujo dicionário só guarda as sequências mais recentes)
  - Desvantagem: o dicionário pode ficar cheio!

# LZ78

- O que fazer quando o dicionário ficar cheio?
  - “congelar” o dicionário:
    - as novas sequências param de ser inseridas no dicionário mas ele continua sendo usado para a compressão
    - perde adaptabilidade pois caso o conteúdo se altere muito o dicionário não será mais adequado para a compressão
    - não exige reinserção de sequências já encontradas anteriormente
  - “reiniciar” o dicionário:
    - todas as entradas são removidas e um novo dicionário começa a ser construído
    - é como se o arquivo de entrada ficasse dividido em blocos
    - perde o histórico de sequências já encontradas e armazenadas no dicionário que devem ser reinseridas
    - mantém a capacidade de adaptação a novos conteúdos
  - outras políticas: semelhantes a utilizadas no gerenciamento de memória, p.ex., retirar as entradas menos utilizadas, ...