

Other examples of
source coding algorithms

So far we have seen examples of:

VARIABLE LENGTH CODING (VLC)

- Huffman
 - Shannon
 - Shannon-Fano
 - Arithmetic
-

Here we will consider other codes

The examples in these slides and the VLCs,
all belongs to the so called
Entropic Coding family
(lossless coding)

Run-Length Encoding (RLE)

Very simple idea exploiting repetitions....

Run-Length Encoding (RLE)

- Run-length encoding (RLE) is probably the simplest method of compression. It can be used to compress data made of any combination of symbols.
- It does not need to know the frequency of occurrence of symbols
- The general idea behind this method is to replace consecutive repeating occurrences of a symbol by one occurrence of the symbol followed by the number of occurrences.

Run-Length Encoding (RLE)

Run-length encoding example

57264444444443213333333333333333333312780000000000000000

a. Original data

5726#409321#3191278#015

The symbol 4 is
repeated 09 times.

The symbol 3 is
repeated 19 times.

The symbol 0 is
repeated 15 times.

b. Compressed data

Run-Length Encoding (RLE)

Run-length encoding other example

a. Original data

BBBBBBBBBAAAAAAAAAAAAAAAAANMMMMMMMMMM

b. Compressed data

B09A16N01M10

Run-length encoding (RLE) -another example

The following string:

- BBBBHHDDXXXXKKKKWWZZZZ

can be encoded as:

- 4B2H2D4X4K2W4Z

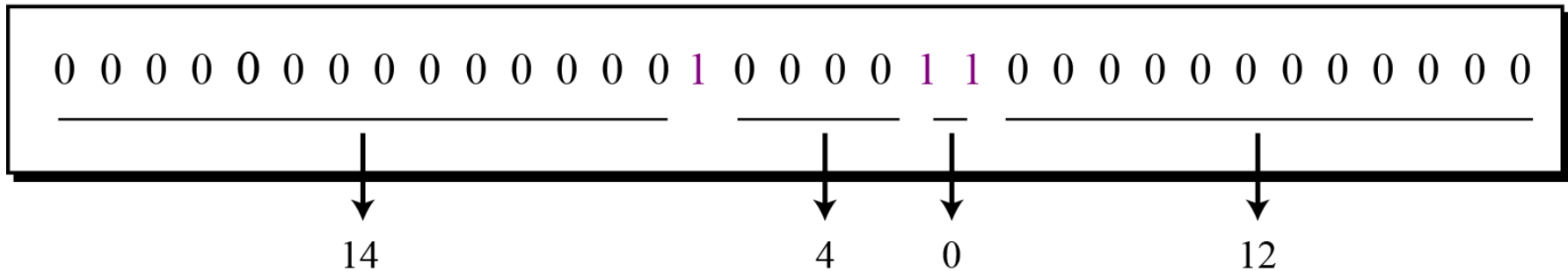
Here "4B" means four B's, and 2H means two H's, and so on.

In this case, the number of repetitions is put “before” instead of “later” (just another example).....

Run-length encoding (RLE) for two symbols

Consider the “space between 1’s”

a. Original data



Consider the “distance between 1’s”

1110 0100 0000 1100

b. Compressed data

$$1 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 = 14$$

$$0 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 0 \cdot 2^0 = 4$$

$$0 \cdot 2^3 + 0 \cdot 2^2 + 0 \cdot 2^1 + 0 \cdot 2^0 = 0$$

$$1 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 0 \cdot 2^0 = 12$$

Run-length encoding (examples)

As another example, consider the storage of a rectangular image. As a single color bitmapped image, it can be stored as:

[illegible]

The rectangular image can be compressed with run-length encoding by counting identical bits as follows:

- 0, 40
- 0, 40
- 0,10 1,20 0,10
- 0,10 1,1 0,18 1,1 0,10
- 0,10 1,1 0,18 1,1 0,10
- 0,10 1,1 0,18 1,1 0,10
- 0,10 1,20 0,10
- 0,40

The first line says that the first line of the bitmap consists of 40 0's. The third line says that the third line of the bitmap consists of 10 0's followed by 20 1's followed by 10 more 0's, and so on for the other lines

Run-Length Encoding (examples)

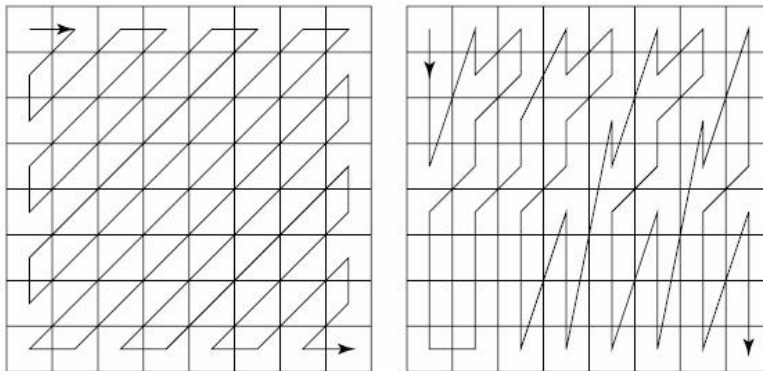
[illegible]

Zig-Zag scanning

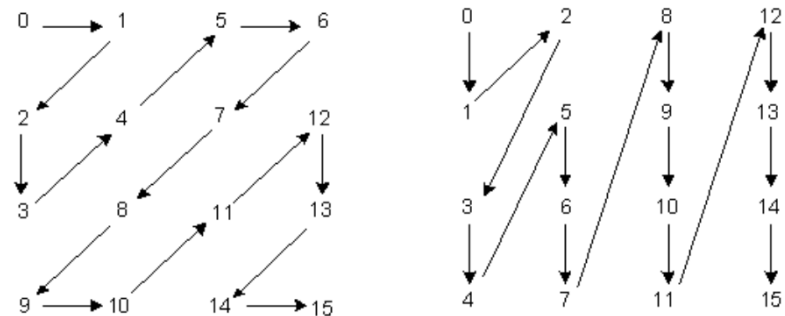
Run-Length Encoding (RLE)

- Codificadores de video como MPEG-2, H.264 y HEVC, utilizan codificación RLE pero no directamente sobre los pixeles de la imagen, sino aplicado a los coeficientes transformados y cuantificados resultados de la 2D-DCT, con distinto patrones especiales denominados **Zig-Zag scanning**.

MPEG-2 (8x8)



H.264 (4x4)



Dictionary – based algorithms

Example of Dictionary – based algorithm

Example

Suppose we have a source with a five-letter alphabet $\mathcal{A} = \{a, b, c, d, r\}$. Based on knowledge about the source, we build the dictionary shown in Table

TABLE **A sample dictionary.**

Code	Entry	Code	Entry
000	<i>a</i>	100	<i>r</i>
001	<i>b</i>	101	<i>ab</i>
010	<i>c</i>	110	<i>ac</i>
011	<i>d</i>	111	<i>ad</i>

Example of Dictionary – based algorithm

TABLE **A sample dictionary.**

Code	Entry	Code	Entry
000	<i>a</i>	100	<i>r</i>
001	<i>b</i>	101	<i>ab</i>
010	<i>c</i>	110	<i>ac</i>
011	<i>d</i>	111	<i>ad</i>

Suppose we wish to encode the sequence

abracadabra

The encoder reads the first two characters *ab* and checks to see if this pair of letters exists in the dictionary. It does and is encoded using the codeword 101. The encoder then reads the next two characters *ra* and checks to see if this pair occurs in the dictionary. It does not, so the encoder sends out the code for *r*, which is 100, then reads in one more character, *c*, to make the two-character pattern *ac*. This does exist in the dictionary and is encoded as 110. Continuing in this fashion, the remainder of the sequence is coded. The output string for the given input sequence is 101100110111101100000. ♦

Lempel Ziv encoding

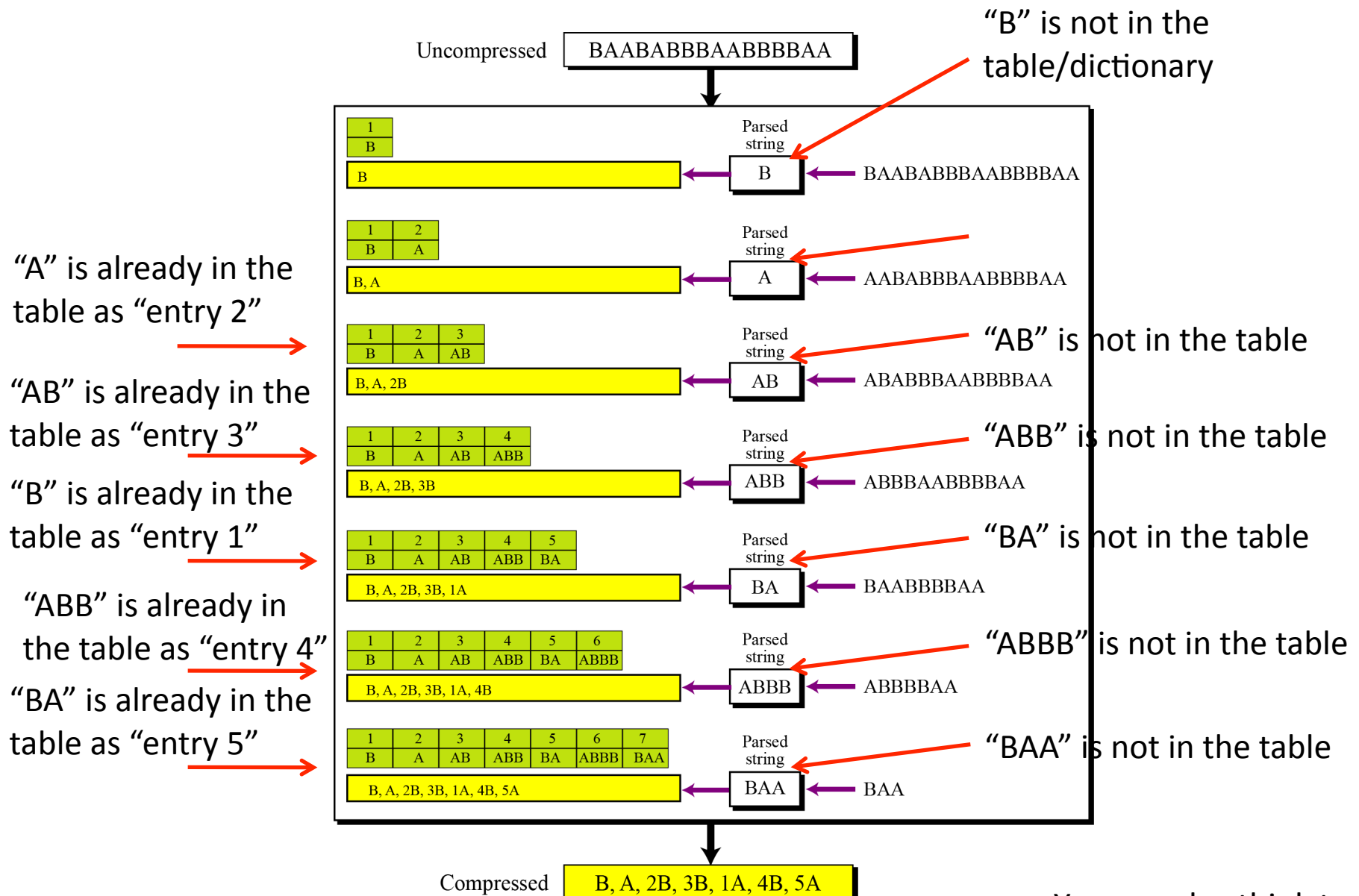
Lempel Ziv (LZ) encoding is an example of a category of algorithms called *dictionary-based* encoding.

-The idea is to create a *adaptively* dictionary (a table) of strings used during the communication session.

Basic idea:

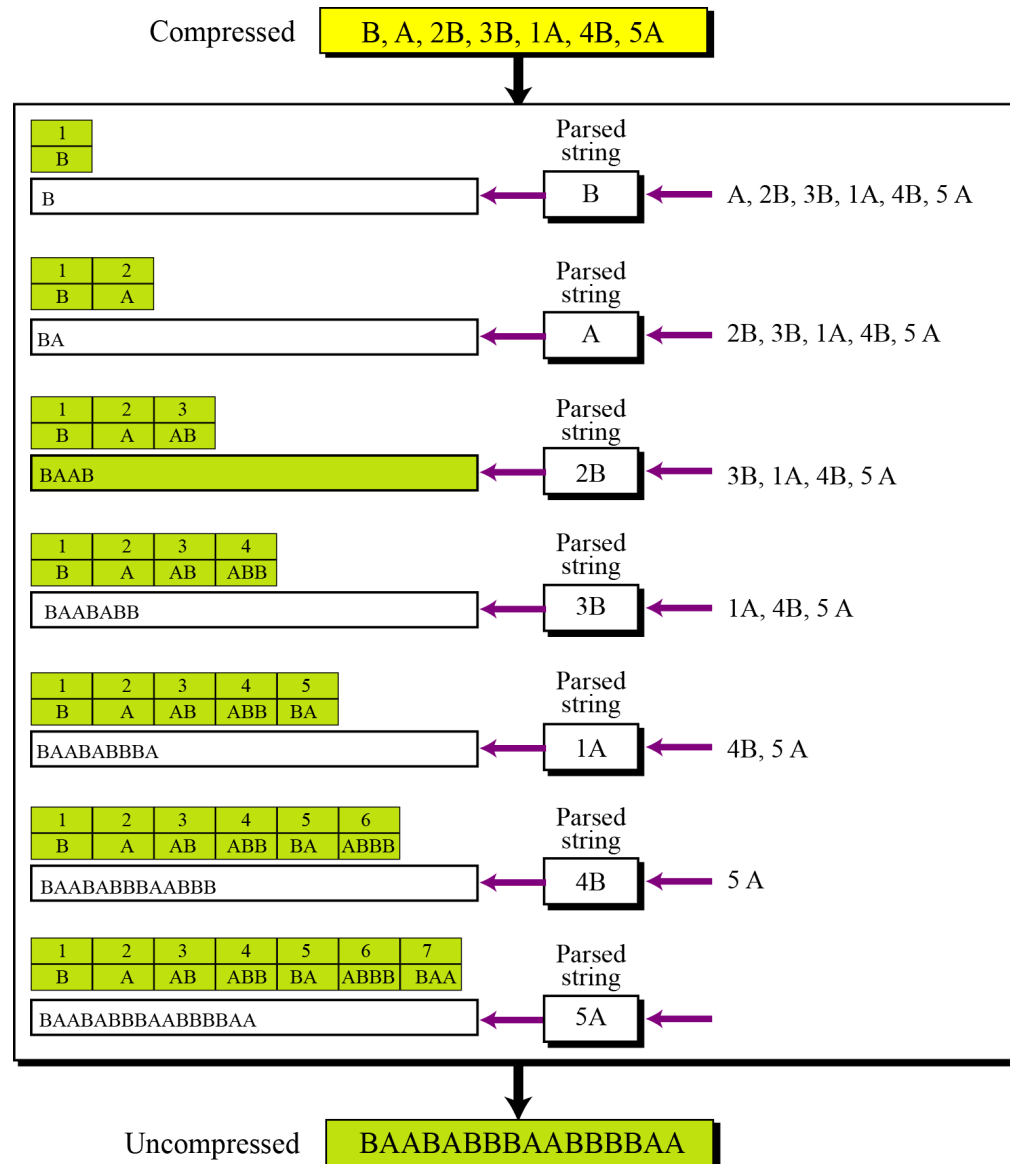
**In this phase there are two concurrent events:
(1) building an indexed dictionary,
(2) and compressing a string of symbols.**

-The algorithm extracts the smallest substring that cannot be found in the dictionary from the remaining uncompressed string. It then stores a copy of this substring in the dictionary as a new entry and assigns it an index value.



An example of Lempel Ziv encoding

Decoder



An example of Lempel Ziv decoding