

Dictionary methods

- encode "phrases", not single symbols into code words
- output words/symbols are equiprobable

Ex: Encoding English text

'the' → 1
'aw' → 2
'is' → 3
...

- based on "transformation table" → **DICTIONARY**
- table is dynamic → coding improves with the number of symbols processed (usually!)

Historical remark: **RLE (Run-Length Encoding)**

PCX (i.i.r.c. max 16 colors)

Ex: binary B&W bitmap

0... black
1... white

1111000000101111

4,1,6,0,1,1,1,0,4,1

can be removed if binary



8 bit packed

3 bits ⇒ 8 colours

5 bits ⇒ max. 32 repetitions

can be implemented as a dictionary, BUT it is computationally ineffect.

Sliding window

- given by its width in "elements" (source symbols)
- moves through the data set

Ex: sliding window width 8

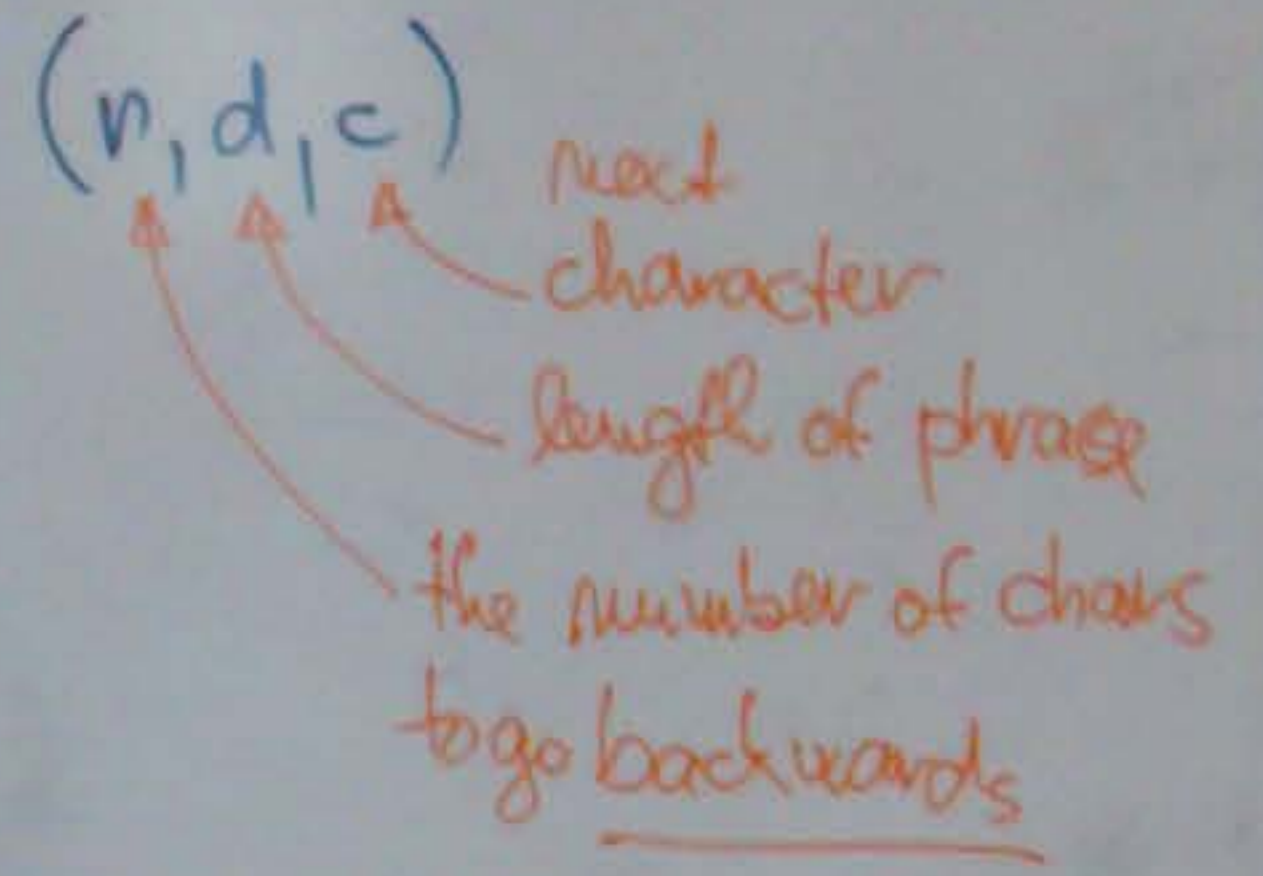
abbedbabedba

position 0

position 1

[Z77] (Lempel-Ziv)

- use a sliding window to search for repetitions of a phrase;
- if found, the current phrase is replaced with a reference



K d y b y u n e b y l y u b y u c h y b y

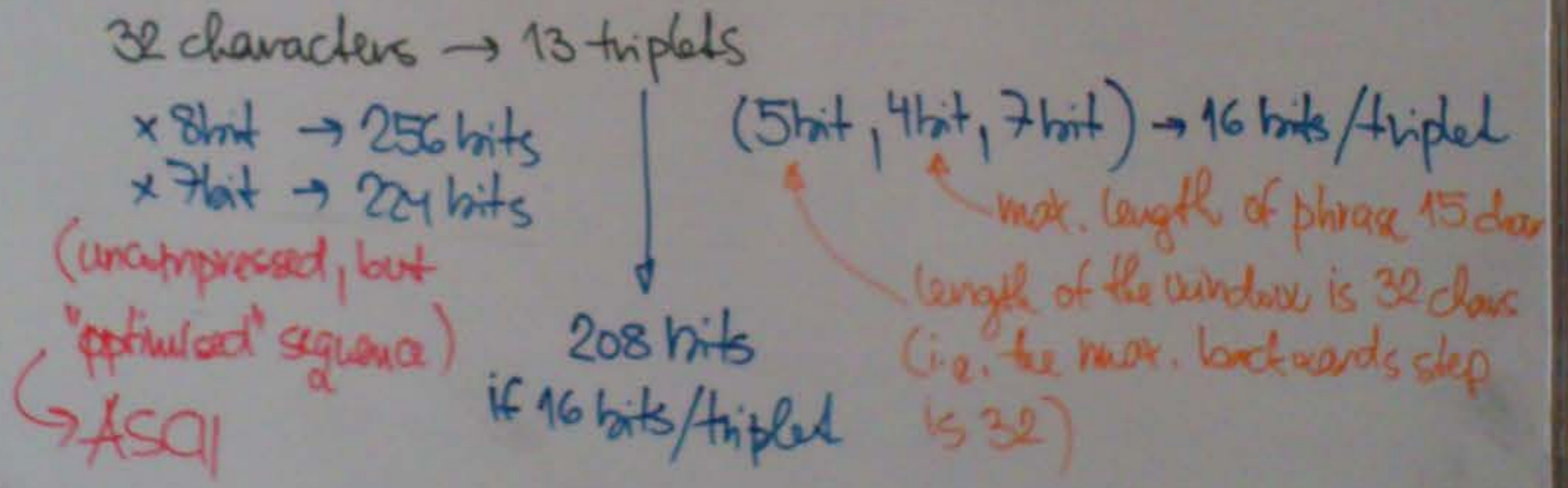
K d y b y u n e b y l y u b y u c h y b y

(0,0,K) (0,0,d) (0,0,y) (0,0,b) (2,1,u) (3,2,e) (5,2,r) (10,4,n)

no previous reference found

e b y l y u b y u c h y b y

(0,0,e) (12,5,b) (3,2,c) (0,0,h) (16,3,.)



LZW (Lempel-Ziv-Welch)

• search in a long sliding window is slow

→ use a tree-based dictionary

a) dictionary is adaptive
 (can be discarded after compression and re-created when decompressing)

b) pre-defined basis

K d y b y _ b y l y _ r y b y _ b y _ c h y b y

0 1 2 3 2 4 15 5 16 6 14 16 7 8 18 16 15

+ + + +
 kd dy yb by
 this goes to dictionary

_ c h y b y . EOF
 4 9 10 22 11

32 characters 2 symbols à 6 bits = 132 bits
 ↳ 256 bits
 224 bits

Dictionary:

k → 0	kd → 12	byly → 26
d → 1	dy → 13	yub → 27
y → 2	yb → 14	by_ → 28
b → 3	by → 15	_c → 29
_ → 4	y_ → 16	ch → 30
l → 5	_b → 17	hy → 31
r → 6	byl → 18	yby. → 32
y → 7	ly → 19	
e → 8	yur → 20	
c → 9	ry → 21	
h → 10	yby → 22	
. → 11	yun → 23	
(basis)	ne → 24	
	eb → 25	