

# String Algorithms and Data Structures

## The Z-algorithm

CS 199-225

February 9, 2026

Brad Solomon



UNIVERSITY OF  
**ILLINOIS**  
URBANA - CHAMPAIGN

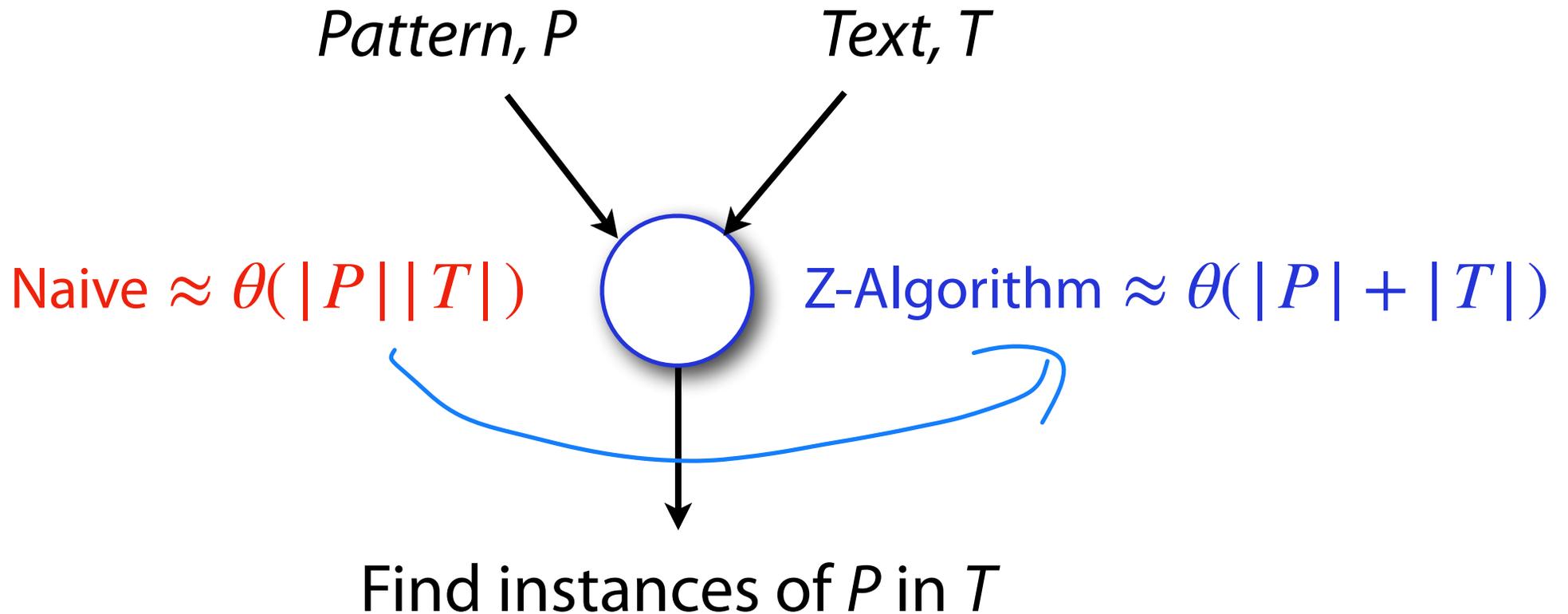
BA B BA

x  
0 1 2 0

(But fast)

Department of Computer Science

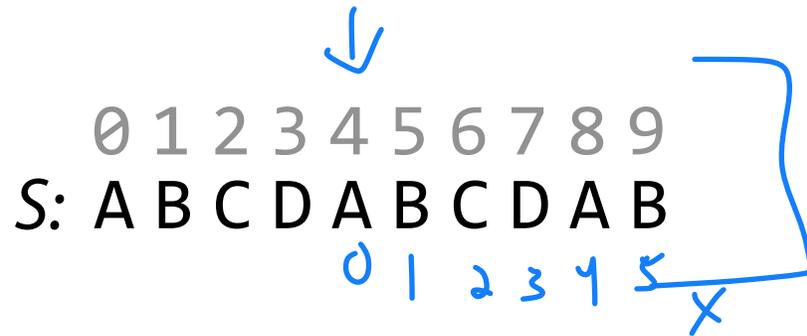
# Exact Pattern Matching w/ Z-algorithm



'instances': An exact, full length copy

# The Z-value [ $Z_i(S)$ ]

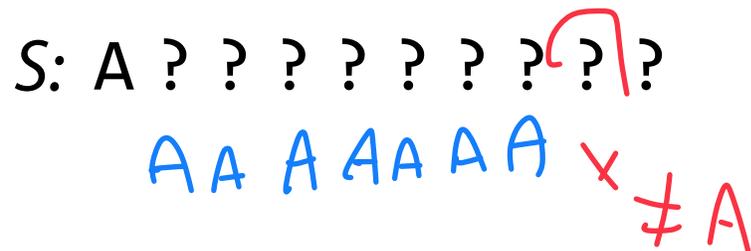
Given a string  $S$ ,  $Z_i(S)$  is the length of the longest substring in  $S$ , starting at position  $i > 0$ , that matches a prefix of  $S$ .



$$Z_4(S) = 6$$



$$Z_5(S) = 3$$



$$Z_1(S) = 7$$

# The Z-value [ $Z_i(S)$ ]

Given a string  $S$ ,  $Z_i(S)$  is the length of the longest substring in  $S$ , starting at position  $i > 0$ , that matches a prefix of  $S$ .

0 1 2 3 4 5 6 7 8 9  
S: A B C D A B C D A B

$$Z_4(S) = 6$$

S: C G C G A C G C X ?  
X ! = G

$$Z_5(S) = 3$$

S: A A A A A A A X ?  
X ! = A

$$Z_1(S) = 7$$

# The Z-Algorithm

$s: 1\ 0\ 1\ \$\ 1\ 0\ 1\ 0\ 1\ 1$

$0$	$1\ \$\ 1\ 0\ 1\ 0\ 1\ 1$	$1$
$1$	$1\ 0\ 1\ 0\ 1\ 1$	$+2$
$\$$	$1\ 0\ 1\ 0\ 1\ 1$	$1$
$1\ 0\ 1$	$0\ 1\ 1$	$4$
$0$	$1\ 0\ 1\ 1$	$1$
$1\ 0\ 1\ 1$		$2$
$0\ 1\ 1$		$1$
$1\ 1$		$2$
$1$		$1$

16 characters total

$s$  length 10

Not efficient at all :)

# The Z-Algorithm

$$Z_1 = 3$$

$$Z_2 =$$

corresponding Z val

"frontier"

0	1	2	3	4	5	6	7
A	A	A	A	B	B	B	B
A	A	A	A	B	B	B	B

We track our current knowledge of  $S$  using three values:  $i, r, l$

$i$  gets updated every iteration (as we compute  $Z_i$ )

$r$  gets updated when  $Z_i > 0$  AND  $r_{new} > r_{old}$

$l$  gets updated whenever  $r$  is updated (it stores the index of  $r$ 's Z-value)

# The Z-Algorithm

$z_4 = 3$

0	1	2	3	4	5	6	7	8	9
<del>1</del>	<del>0</del>	<del>1</del>	\$	<del>1</del>	<del>0</del>	<del>1</del>	0	1	1
1	0	1	\$	1	0	1	0	1	1

B/c
past
frontier
(r)
direct calc!

$i$ , the current index =

$r$ , the furthest match char =

$l$ , the furthest reaching Z-value =

Start

End

4

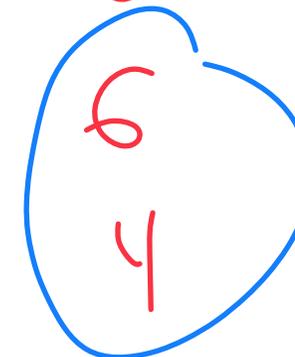
5

2

6

2

4



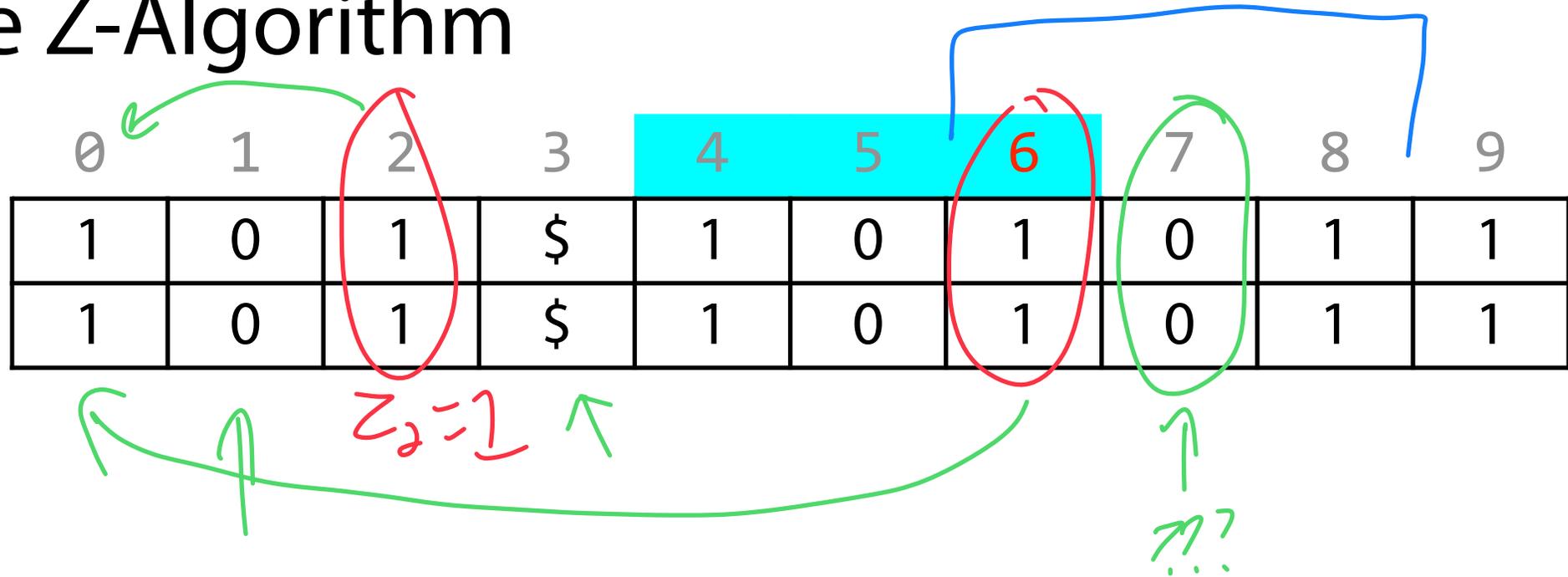
# The Z-Algorithm

0	1	2	3	4	5	6	7	8	9
1	0	1	\$	1	0	1	0	1	1
1	0	1	\$	1	0	1	0	1	1

$z_1$   $\rightarrow$   $z_5 = ?$  No work needed!

	Start	End
$i$ , the current index =	5	6
$r$ , the furthest match char =	6	6
$l$ , the furthest reaching Z-value =	4	4

# The Z-Algorithm



$i$ , the current index =

Start

6

End

7

$r$ , the furthest match char =

6

8

$l$ , the furthest reaching Z-value =

4

6



# The Z-Algorithm

We track our current knowledge of  $S$  using three values:  $i, r, l$

At a glance, there seems to be three distinct cases of calculation...

1) I know nothing (naive calculation needed)

2) I know everything already (no work needed)

3) I know some things (calculate only what is not known)



**Lets break down how we can tell which case we are in!**

# The Z-Algorithm

$z_1 = 2$

0	1	2	3	4	5	6	7
<del>A</del>	<del>A</del>	<del>A</del>	B	B	A	A	A
A	<del>A</del>	<del>A</del>	<del>B</del>	B	A	A	A

The values of  $i, r, l$  tell us how much work we need to do to compute  $Z_i$

Case 1:  $i > r$

Ex:  $i = 1, r = 0, l = 0$

We must compute  $Z_i$  explicitly!

# The Z-Algorithm

Prior Knowledge

0	1	2	3	4	5	6	7
A	A	A	<del>B</del>	<del>B</del>	A	A	A
A	A	A	B	B	A	A	A

$z_3=0$   $z_4=0$

$i$

frontier

The values of  $i, r, l$  tell us how much work we need to do to compute  $Z_i$

Case 1:  $i > r$

Ex:  $i = 5, r = 2, l = 1$

We must compute  $Z_i$  explicitly!

# The Z-Algorithm

Inside Z-box  
↓  
 $z_6 = 2$

0	1	2	3	4	5	6	7
A	A	A	B	B	A	A	A
A	A	A	B	B	A	A	A

$z_1 = 2$        $5 \ 6 \ 7 = 0 \ 1 \ 2$

The values of  $i, r, l$  tell us how much work we need to do to compute  $Z_i$

Case 2:  $i \leq r$

Ex:  $i = 6, r = 7, l = 5$

To find  $Z_6$ , we can save time by looking up the value  $Z_1$

# The Z-Algorithm

$z_1 = 0$   $z_5 = 0$

0	1	2	3	4	5	6	7
A	B	C	B	A	B	C	A
A	B	C	B	A	B	C	A
0	1	2		4	5	6	

The values of  $i, r, l$  tell us how much work we need to do to compute  $Z_i$

Case 2:  $i \leq r$

Ex:  $i = 5, r = 6, l = 4$

To find  $Z_5$ , we can save time by looking up the value      $z_1$

# The Z-Algorithm

$$01 = 34$$

0	1	2	3	4	5	6	7
A	A	B	A	A	A	B	C
A	A	B	A	A	A	B	C

frontier point  
work?

$$z_1 = 2 \quad \leftrightarrow \quad z_4 = 3$$

The values of  $i, r, l$  tell us how much work we need to do to compute  $Z_i$

Case 2:  $i \leq r$

Ex:  $i = 4, r = 4, l = 3$

To find  $Z_4$ , we can save time by looking up the value  $Z_l$  but... not all the cost



# The Z-Algorithm

Let  $l = 0, r = 0$ , for  $i = [1, \dots, |S| - 1]$ :

Compute  $Z_i$  using  $irl$ :

Case 1 ( $i > r$ ): Compute explicitly; update  $irl$  [if a match]

Case 2 ( $i \leq r$ ):

Use previous Z-values to avoid work

Explicitly compute only 'new' characters

How can we tell the difference between cases?

↩

3 sub cases

# The Z-Algorithm

$$i = 6, r = 7, l = 5$$

0	1	2	3	4	5	6	7	8
<b>A</b>	<b>A</b>	<b>A</b>	<b>A</b>	<b>C</b>	<b>A</b>	<b>A</b>	<b>A</b>	<b>B</b>
A	A	A	A	C	A	A	A	B
A	A	A	A	C	A	A	A	B

The amount of work required depends on two pieces of information

**1. # of characters at or after  $i$  that we have matched before**

*If  $z_6$  I might need info on 6 & 7*

**2. The Z-value that matches part or all of the string starting at  $i$**

*↳ Lets call this  $z_k$  (what I look up for info)*

# The Z-Algorithm

$i = 6, r = 7, l = 5$

0	1	2	3	4	5	6	7	8
<b>A</b>	<b>A</b>	<b>A</b>	<b>A</b>	<b>C</b>	<b>A</b>	<b>A</b>	<b>A</b>	<b>B</b>
A	A	A	A	C	A	A	A	B
A	A	A	A	C	A	A	A	B

*(Handwritten annotations: A red arrow points to index 6, and a red bracket above indices 6-7 is labeled with  $i = 6, r = 7, l = 5$ . Blue vertical lines are drawn at indices 6 and 7. A blue arrow points from the 'A' at index 7, row 2 to a handwritten  $|\beta|$  on the right.)*

The amount of work required depends on two pieces of information

**1. # of characters at or after  $i$  that we have matched before**

Call this value  $|\beta|$ . What is  $|\beta|$  in terms of  $i, r, l$ ?

$$|\beta| = r - i + 1$$

# The Z-Algorithm

$$i = 6, r = 7, l = 5$$

0	1	2	3	4	5	6	7	8
<b>A</b>	<b>A</b>	<b>A</b>	<b>A</b>	<b>C</b>	<b>A</b>	<b>A</b>	<b>A</b>	<b>B</b>
A	A	A	A	C	A	A	A	B
A	A	A	A	C	A	A	A	B

The amount of work required depends on two pieces of information

**2. The Z-value that matches part or all of the string starting at  $i$**

Call this value  $Z_k$ . What is  $k$  in terms of  $i, r, l$ ?

$$k = i - l \rightarrow \text{Define } Z_k \text{ (took up)}$$

# The Z-Algorithm



$$i = 6, r = 7, l = 5$$

0 1 2  
1 2 3

	0	1	2	3	4	5	6	7	8
	A	A	A	A	C	A	A	A	B
	A	A	A	A	C	A	A	A	<del>B</del>
$Z_k = Z_1 = 3$	A	A	A	<del>A</del>	C	A	A	A	B

The amount of work required depends on two pieces of information

**1. # of characters at or after  $i$  that we have matched before**

$$|\beta| = 7 - 6 + 1 = 2$$

**2. The Z-value that matches part or all of the string starting at  $i$**

$$k = 6 - 5 = 1$$

0 1 2 = 1 2 3 = 5 6 7

# The Z-Algorithm

$$i = \underline{5}, r = \underline{7}, l = 4$$

$Z_k = 2$



0	1	2	3	4	5	6	7
A	A	A	B	A	A	A	B
A	A	A	B	A	A	A	B

# of matches seen previously
# of characters I know stuff about

Case 2a:  $i \leq r$ ,  $Z_k < |\beta|$

$|\beta| = \underline{3}$ ,  $k = \underline{1}$ ,  $Z_k = \underline{2}$

$Z_i = \underline{2}$

$B/c \quad Z_l \quad 0123 = \underline{4567}$   
 $Z_k \quad 123 = 012$

# The Z-Algorithm

$$i = 5, r = 7, l = 4$$

$\emptyset$	1	2	3	4	5	6	7

Case 2a:  $i \leq r, Z_k < |\beta|$

$Z_l$  (defined by  $r, l$ ) tells us that  $\beta$  matches earlier.



# The Z-Algorithm

$$i = 5, r = 7, l = 4$$



why is  $z_3 = 7$ ?

$z_1 \rightarrow 0 \ 1 \ 2 \ 3 = 4 \ 5 \ 6 \ 7$

0	1	2	3	4	5	6	7
A	A	A	<del>B</del>	A	A	A	<del>B</del>

Case 2a:  $i \leq r, Z_k < |\beta|$

$Z_l$  tells us that  $\beta$  matches earlier.  $Z_k$  tells us how much matches the prefix.

Because  $Z_k < |\beta|, Z_i = \underline{Z_k}$  No work needed!

# The Z-Algorithm

$$i = 4, r = 4, l = 3$$

$Z_1 = Z_k = 1$

0	1	2	3	4	5	6	7
A	A	B	A	A	A	B	C
A	A	B	A	A	A	B	C

# characters matched prev      # of characters I can get info about

Case 2b:  $i \leq r, Z_k = |\beta|$

$$|\beta| = \underline{1}, k = \underline{1}, Z_k = \underline{1}$$

$$Z_i = \frac{1}{(Z_k)} + \text{stuff starting from 5}$$

# The Z-Algorithm

$$i = 4, r = 4, l = 3$$

0	1	2	3	4	5	6	7

Case 2b:  $i \leq r, Z_k = |\beta|$

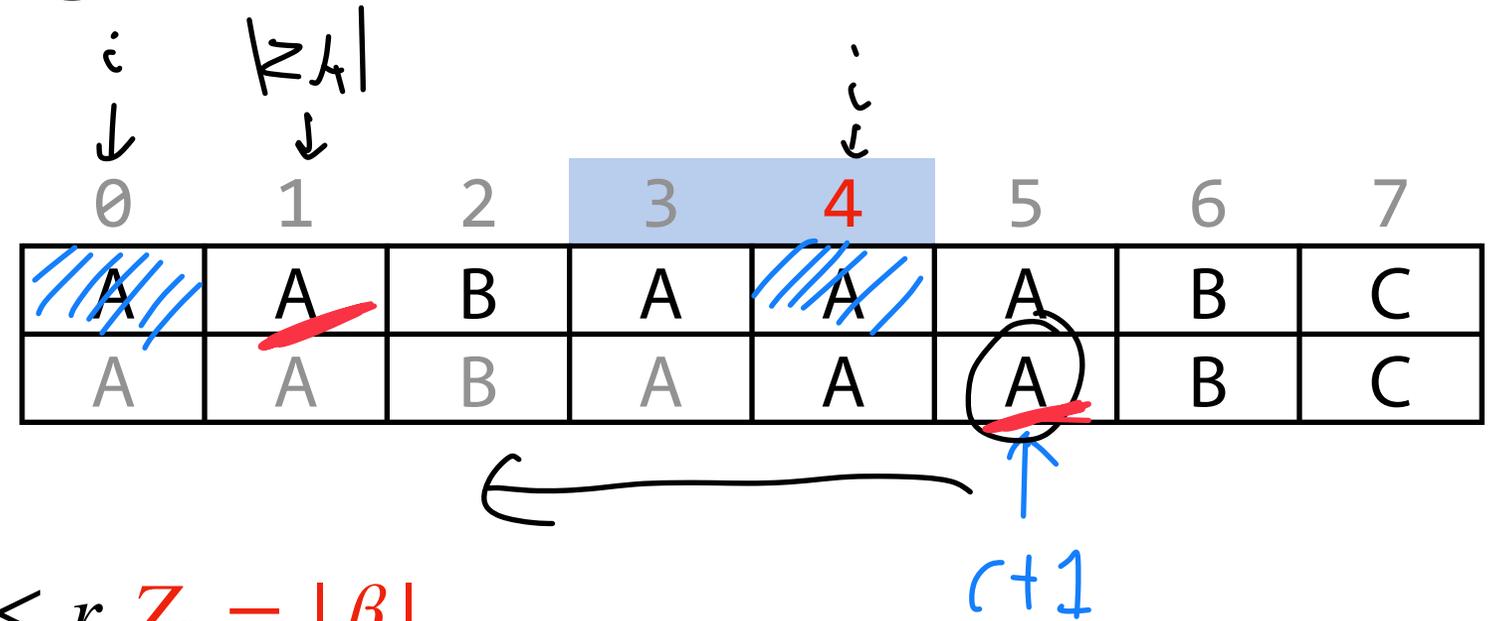
$Z_l$  (defined by  $r, l$ ) tells us that  $\beta$  matches earlier.

$$01 = 34$$



# The Z-Algorithm *\* Common mistake!*

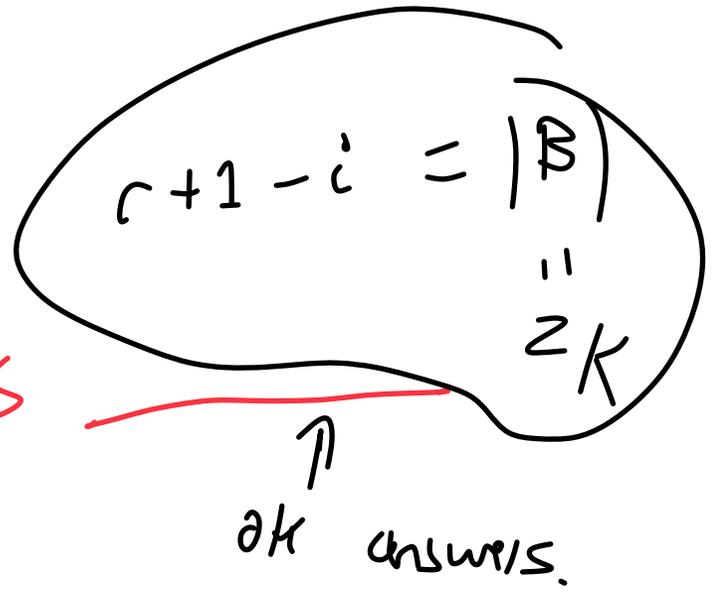
$i = 4, r = 4, l = 3$



Case 2b:  $i \leq r, Z_k = |\beta|$

$|\beta| = 1, k = 1, Z_k = 1$

$Z_i = Z_k +$  comparison starting at  $r+1$  *vs*



# The Z-Algorithm

$$i = 3, r = 5, l = 1$$

$z_3 = 4$

0	1	2	3	4	5	6	7
A /	A /	A /	A /	A	A	B	C
A	A	A /	A /	A /	A /	B	C

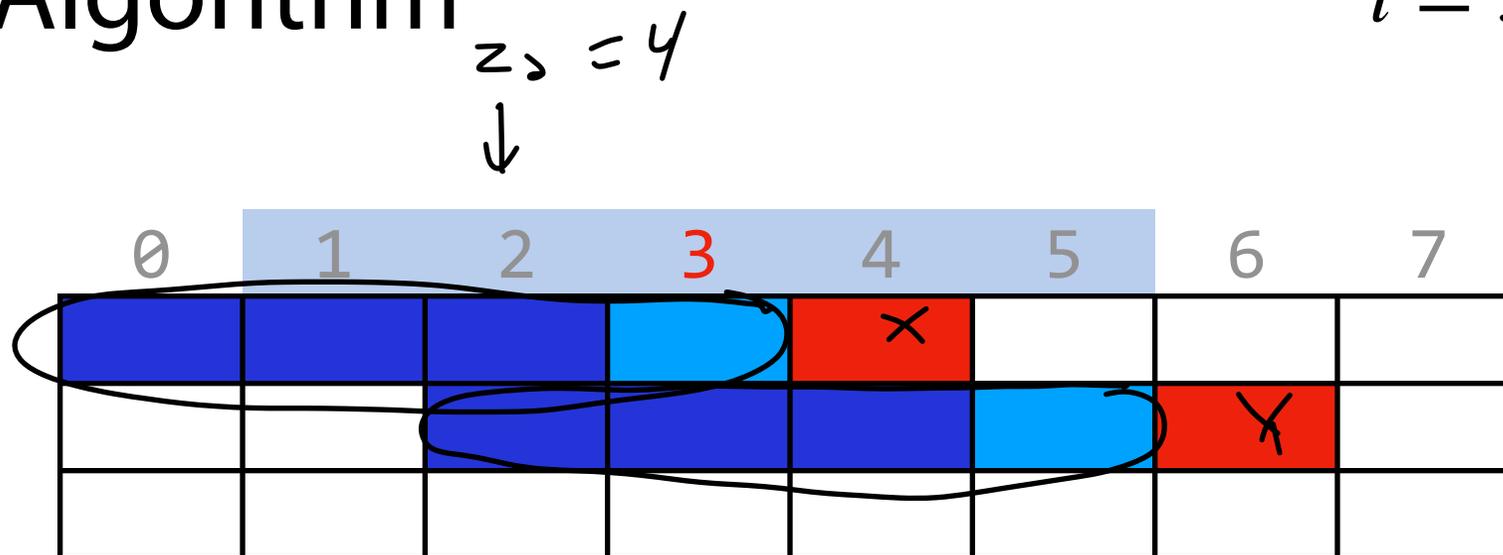
Case 2c:  $i \leq r, z_k > |\beta|$

$|\beta| = \underline{\quad 3 \quad}, k = \underline{\quad 2 \quad}, z_k = \underline{\quad 4 \quad}$

$z_i = \underline{\quad |\beta| \quad}$   
↑

# The Z-Algorithm

$$i = 3, r = 5, l = 1$$



Case 2c:  $i \leq r, Z_k > |\beta|$

$$0123 = 2345$$

$Z_k$  tells us how much matches the prefix.

# The Z-Algorithm

0 1 2 3 4 = 1 2 3 4 5  $i = 3, r = 5, l = 1$

0	1	2	3	4	5	6	7

Case 2c:  $i \leq r, Z_k > |\beta|$

$Z_l$  tells us that our entire range ( $\beta$  included) matches earlier

# The Z-Algorithm

$$i = 3$$

$$i = 3, r = 5, l = 1$$



0	1	2	3	4	5	6	7
Blue	Blue	Blue	Light Blue	Red	White	White	White
White	White	Blue	Light Blue	Blue	Light Blue	Red	White
White	White	White	Blue	Blue	Blue	Yellow	White



$$3 \quad 45 = 234 = 012$$

Case 2c:  $i \leq r, Z_k > |\beta|$

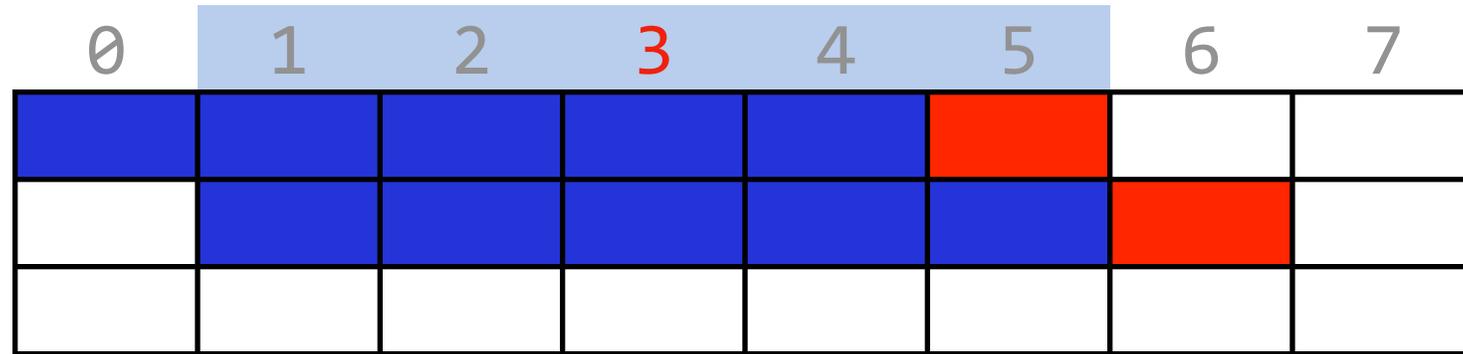
$Z_l$  tells us that  $\beta$  matches earlier.  $Z_k$  tells us how much matches the prefix.

Does 6 match 5 or 3

**What do we know about yellow?**

# The Z-Algorithm

$$i = 3, r = 5, l = 1$$



Case 2c:  $i \leq r, Z_k > |\beta|$

$$01234 = 12345$$

5 ≠ 6

$Z_l$  tells us that our entire range ( $\beta$  included) matches earlier

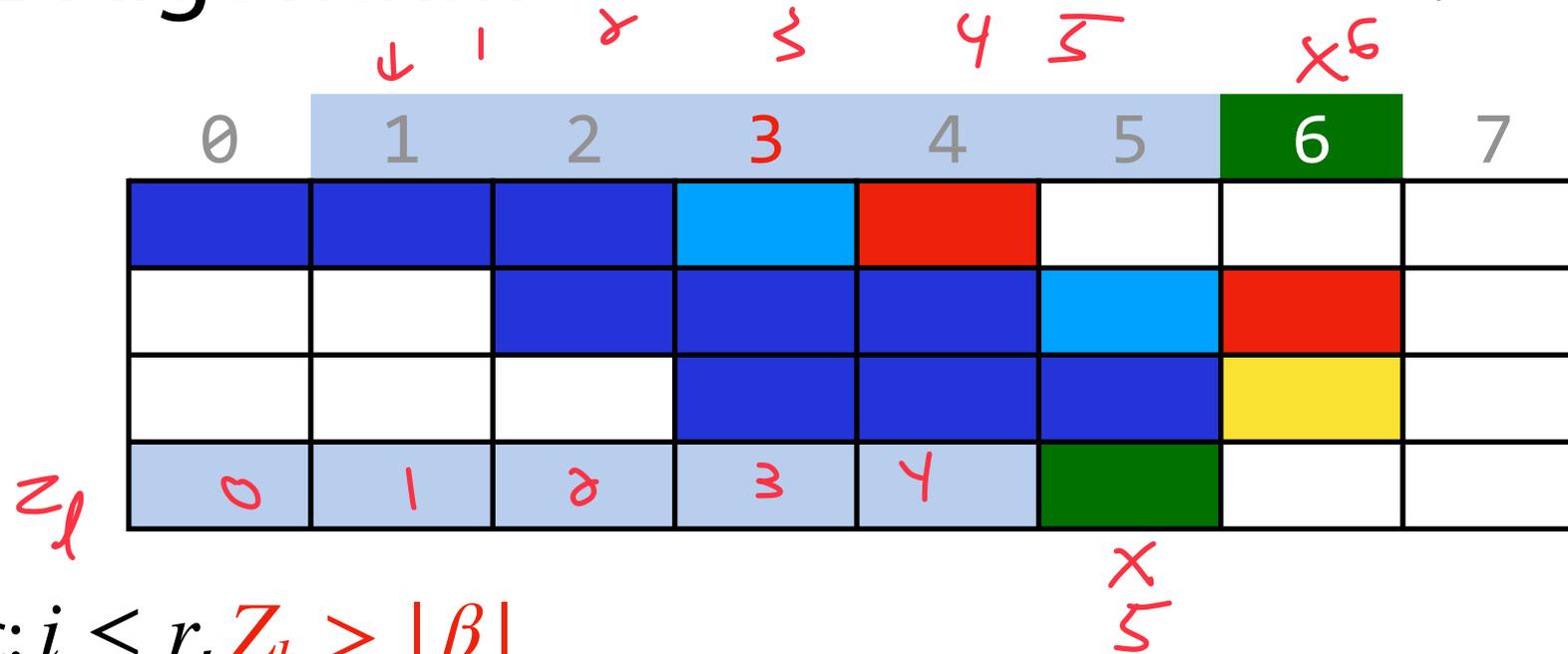


... and that it failed to match the next character.

we not equal!

# The Z-Algorithm

$i = 3, r = 5, l = 1$



Case 2c:  $i \leq r, Z_k > |\beta|$

$Z_l$  tells us that  $\beta$  matches earlier.  $Z_k$  tells us how much matches the prefix.

**$Z_l$  also tells us that yellow and green can't be equal!**

# The Z-Algorithm

$$i = 3, r = 5, l = 1$$



0	1	2	3	4	5	6	7
Blue	Blue	Blue	Light Blue	Red	White	White	White
White	White	Blue	Blue	Blue	Light Blue	Red	White
White	White	White	Blue	Blue	Blue	Yellow	White
Light Blue	Green	White	White				

Case 2c:  $i \leq r, Z_k > |\beta|$

$Z_l$  tells us that  $\beta$  is our prefix.  $Z_k$  is also a previously computed prefix.

Because  $Z_k > |\beta|$ ,  $Z_i = \underline{\hspace{2cm}}$



# The Z-Algorithm

Let  $l = 0, r = 0$ , for  $i = [1, \dots, |S| - 1]$ :

Compute  $Z_i$  using  $irl$ :

Case 1 ( $i > r$ ): Compute explicitly; update  $irl$  \* only update  $r$  &  $l$  if non-zero Z-value

Case 2 ( $i \leq r$ ):

2a: ( $Z_k < |\beta|$ ):  $Z_i = Z_k$  No work

2b: ( $Z_k = |\beta|$ ):  $Z_i = Z_k + \text{explicit}(\text{indices } r + 1 \text{ vs } Z_k)$ ; update  $irl$  \* make sure your bounds

2c: ( $Z_k > |\beta|$ ):  $Z_i = |\beta|$  No work

# Assignment 3: a\_zalg

Learning Objective:

Construct the full Z-algorithm and measure its efficiency

Demonstrate use of Z-algorithm in pattern matching

Consider: Our goal is  $\theta(|P| + |T|)$ . Does Z-alg search match this?

AAA

§

AAAAAA

A AA

AAA

# Next week:

If I gave you the pattern I was interested in ahead of time, what could you pre-compute to speed up search?

Ex: I'm going to try to look up the word '**arrays**' — but you don't know what text I'm going to search through.