

String Algorithms and Data Structures

Z-values and the Z-algorithm

CS 199-225

September 16, 2024

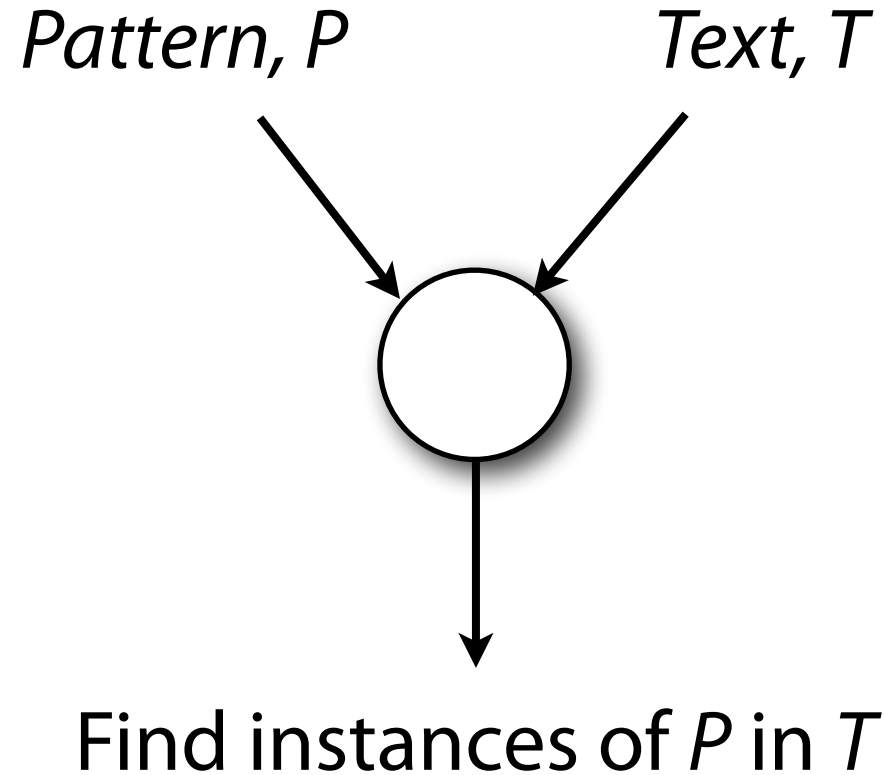
Brad Solomon



UNIVERSITY OF
ILLINOIS
URBANA - CHAMPAIGN

Department of Computer Science

Exact Pattern Matching



'instances': An exact, full length copy

Exact Pattern Matching

What's a simple algorithm for exact matching?

P: word

T: There would have been a time for such a word

word word word word word word word word word **word**

word word word word word word word word

word word word word word word word word

word word word word word word word word

word word word word word word word word

← One
occurrence

Try all possible alignments. For each, check if it matches. This is the *naïve algorithm*.

Exact Pattern Matching

What is good about the naive solution?

What is bad?

Exact Pattern Matching

What is our time complexity?

$$(n = |P|, \quad m = |T|)$$

Exact Pattern Matching

What is our time complexity?

$$(n = |P|, \quad m = |T|)$$

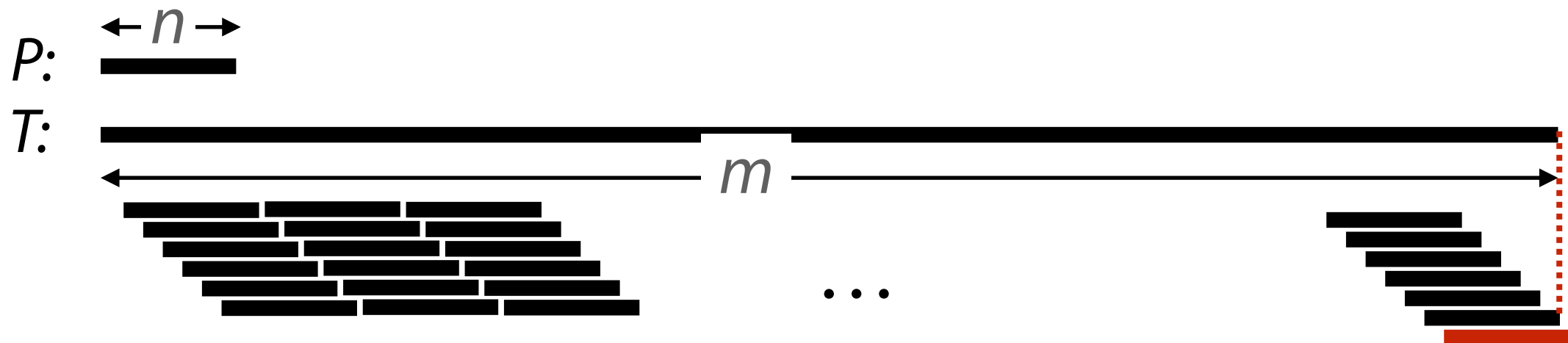
(# of alignments) x (cost of an alignment)

Exact Pattern Matching

What is our time complexity?

$$(n = |P|, \quad m = |T|)$$

(# of alignments) x (cost of an alignment)



P can fit at each 'position' along T except the **edge**

Exact Pattern Matching

What is our time complexity?

$$(n = |P|, \quad m = |T|)$$

(_____) x (cost of an alignment)

P: aaaa

T: aaa

aaaa aaaa aaaa aaaa aaaa aaaa aaaa aaaa aaaa

aaaa aaaa aaaa aaaa aaaa aaaa aaaa aaaa **aaaa**

aaaa aaaa aaaa aaaa aaaa aaaa aaaa aaaa **aaaa**

aaaa aaaa aaaa aaaa aaaa aaaa aaaa aaaa **aaaa**

aaaa aaaa aaaa aaaa aaaa aaaa aaaa aaaa

There are _____ positions which extend past the edge of T

Exact Pattern Matching

What is our time complexity?

$$(n = |P|, \quad m = |T|)$$

$$(m-n+1) \times (\text{cost of an alignment})$$

P: aaaa

T: aaa

aaaa aaaa aaaa aaaa aaaa aaaa aaaa aaaa aaaa

aaaa aaaa aaaa aaaa aaaa aaaa aaaa aaaa

aaaa aaaa aaaa aaaa aaaa aaaa aaaa aaaa

aaaa aaaa aaaa aaaa aaaa aaaa aaaa aaaa

aaaa aaaa aaaa aaaa aaaa aaaa aaaa aaaa

Each alignment compares _____ characters.

Exact Pattern Matching

What is our time complexity?

$(n = |P|, \quad m = |T|)$

$$\theta((m - n + 1) \times n)$$

String Algorithms in Genomics

P: Read ($n = \sim 50-150$)

CTCAAACCTCTGACCTTTGGTGATCCACCCGCCTAGGCCTTC

T: Reference ($m = \sim 3$ billion)

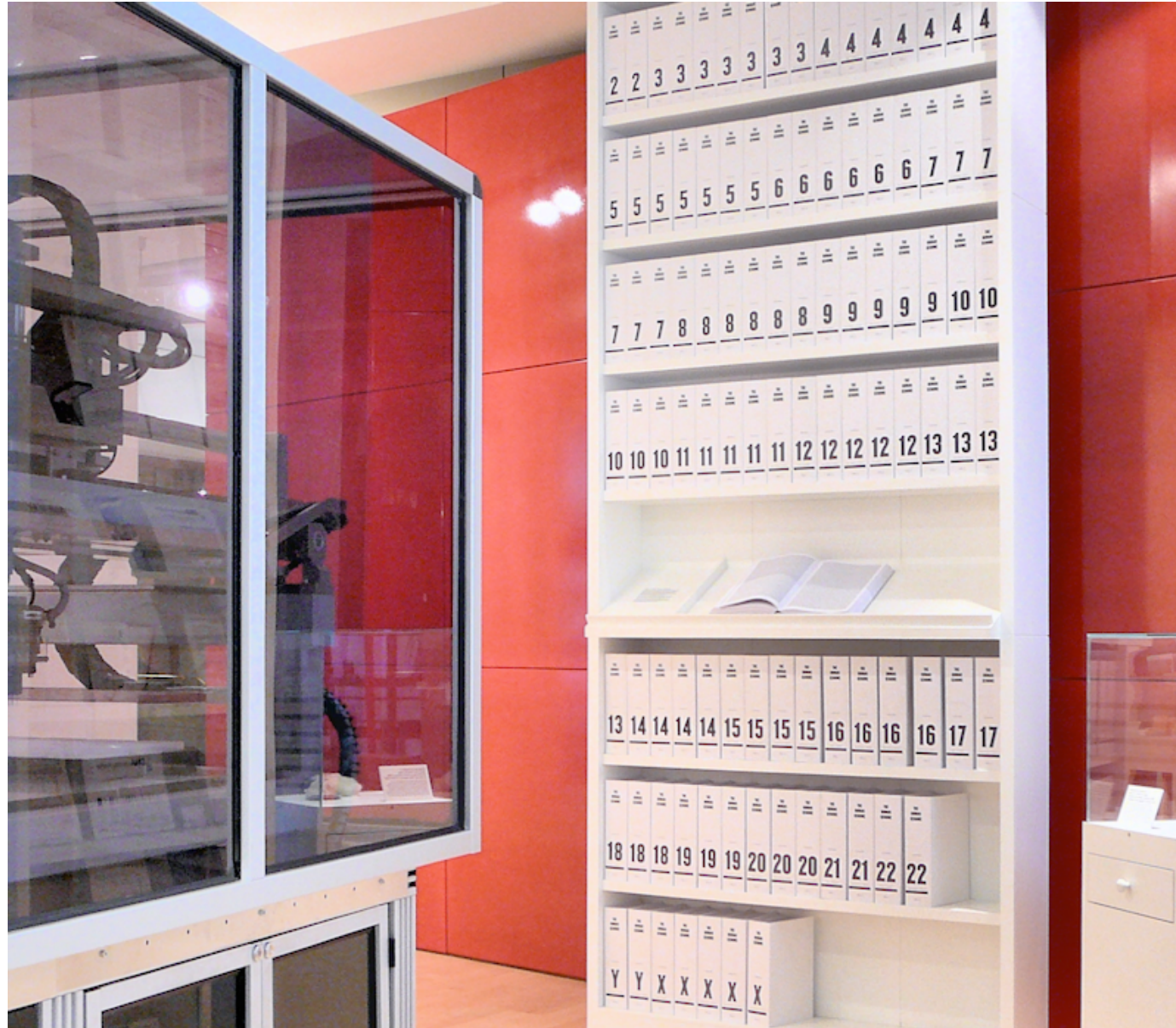
GATCACAGGTCTATCACCTATTAACCACTCACGGGAGCTCTCCATGCATTTGGTATTTT
CGTCTGGGGGATGCACGCGATAGCATTGCGAGACGCTGGAGCCGGAGCACCTATGTC
GCAGTATCTGTCTTTGATTCTGCCTCATCTATTATTTATCGCACCTACGTTCAATATT
ACAGGCGAACATACTTACTAAAGTGTGTTAATTAATTAATGCTTGTAGGACATAATAATA
ACAATTGAATGTCTGCACAGCCACTTTCCACACAGACATCATAACAAAAATTTCCACCA
AACCCCCCTCCCCGCTTCTGGCCACAGCACTTAAACACATCTCTGCCAAACCCAAAA
ACAAAGAACCCTAACACCAGCCTAACAGATTTCAAATTTTATCTTTTGGCGGTATGCAC
TTTTAACAGTCACCCCCAACTAACACATTATTTTCCCCTCCCCTCCACTCCATACTACTAAT
CTCATCAATACAACCCCCGCCATCCTACCCAGCACACACACACCCGCTGCTAACCCATA
CCCCGAACCAACCAACCCCAAGACACCCCCACAGTTTATGTAGCTTACCTCCTCAA
GCAATACACTGACCCGCTCAAACCTCTGGATTTTGGATCCACCCAGCGCCTTGGCCTAAA
CTAGCCTTTCTATTAGCTCTTAGTAAGATTACACATGCAAGCATCCCCGTTCCAGTGAGT
TCACCTCTAAATCACCACGATCAAAGGAACAAGCATCAAGCACGCAATGCAGCTC
AAAACGCTTAGCCTAGCCACACCCCCACGGGAAACAGCAGTGATTAACCTTAGCTATAA
ACGAAAGTTAACTAAGCTATACTAACCCAGGGTTGGTCAATTTGGTCCAGCCACCT
GGTCACACGATTAACCAAGTCAATAGAAGCCGGCGTAAAGAGGTTTGTAGATCACCC
TCCCCAATAAAGCTAAAACCTCACCTGAGTTGTAATAAACTCCGTTGACACAAAATAGAC
TACGAAAGTGGCTTTAACATATCTGAACACACAATAGCTAAGCCCAAACCTGGGATAGA
TACCCCACTATGCTTAGCCCTAAACCTCAACAGTTAAATCAAATAAACTGCTCGCCAGAA
CACTACGAGCCACAGCTTAAAACCTCAAAGGACCTGGCGGTGCTCATATCCCTCTAGAGG
AGCCTGTTCTGTAATCGATAAACCCCGATCAACCTCACCACCTCTGCTCAGCCTATAT
CCGCCATCTTACGCAAACCCTGATGAAGGCTACAAAGTAAGCGCAATACCCACGTA
ACGTTAGGTCAAGGTGTAGCCCATGAGGTGGCAAGAAATGGGCTACATTTCTACCCCA
AAAACCTACGATAGCCCTTATGAACTTAAAGGGTGAAGGTGGATTTAGCAGTAAACTAAG
AGTAGAGTGCTTAGTTGAACAGGGCCCTGAAGCGGTACACACCCGCCGTCACCTCCTC
AAGTATACTTCAAAGGACATTTAACTAAAACCCCTACGCATTTATATAGAGGAGACAAGT
CGTAACCTCAAACCTCTGCCTTTGGTGATCCACCCGCCTTGGCCTACCTGCATAATGAAG
AAGCACCCTAACCTTACACTTAGGAGATTTCAACTTAACTTACCCCTCTGAGCTAAAGCTA



String Algorithms in Genomics



String Algorithms in Genomics



Improving exact pattern matching



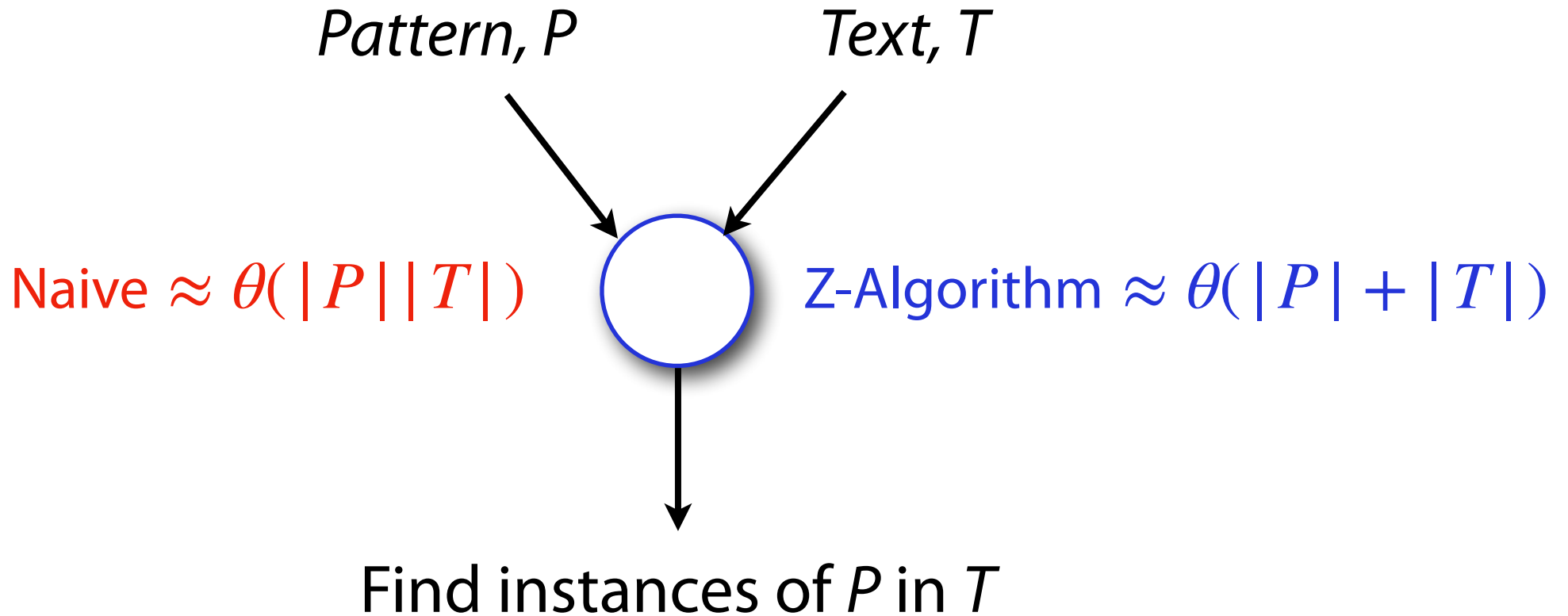
How can we do better than the naïve algorithm?

... If we have infinite space?

... If I tell you the pattern ahead of time?

... If I tell you the text ahead of time?

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 , that matches a prefix of S .

 0 1 2 3 4 5 6 7 8 9
S: **T T C G T T A G C G**

$Z_0(S) =$

$Z_3(S) =$

$Z_1(S) =$

$Z_4(S) =$

$Z_2(S) =$

$Z_5(S) =$

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 , that matches a prefix of S .

 0 1 2 3 4 5 6 7 8 9
S: **T T C G T T A G C G**

$$Z_0(S) = 10$$

$$Z_1(S) = 1$$

$$Z_2(S) = 0$$

$$Z_3(S) =$$

$$Z_4(S) =$$

$$Z_5(S) =$$

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: **T T C G T T A G C G**

$$Z_0(S) = 10$$

$$Z_1(S) = 1$$

$$Z_2(S) = 0$$

$$Z_3(S) = 0$$

$$Z_4(S) = 2$$

$$Z_5(S) = 1$$

Calculating the Z-values

Naive: Compute the Z-values by *explicitly* comparing characters (left-to-right scan):

$$Z_1 =$$

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

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

$$Z_5 =$$

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

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

What is our time complexity?

Calculating the Z-values

Naive: Compute the Z-values by *explicitly* comparing characters (left-to-right scan):

S : 1 1 0 1 1 0 0 1

What is our time complexity?

Calculating the Z-values



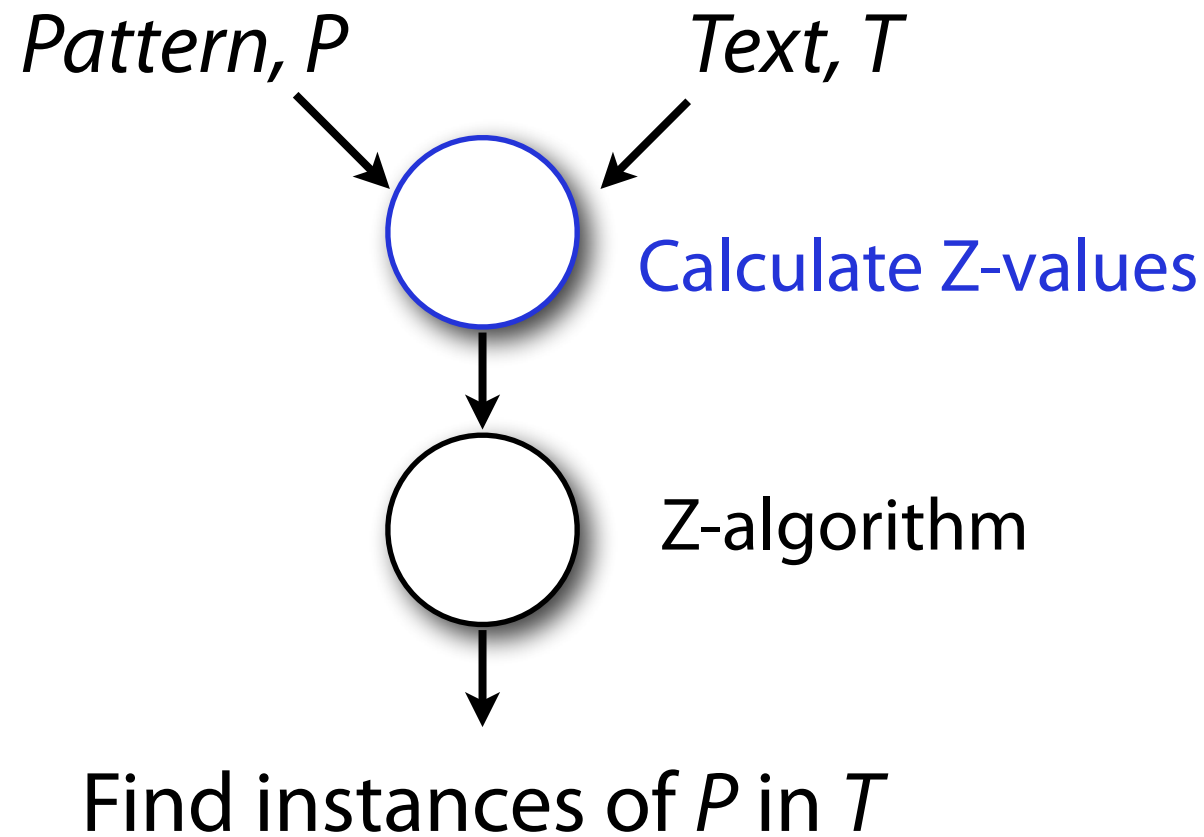
Naive: Compute the Z-values by *explicitly* comparing characters (left-to-right scan):

```
S : 1 1 0 1 1 0 0 1
    1 0 1 1 0 0 1
    0 1 1 0 0 1
    1 1 0 0 1
    1 0 0 1
    0 0 1
    0 1
    1
```

What is our time complexity?

Pattern matching with the Z-value

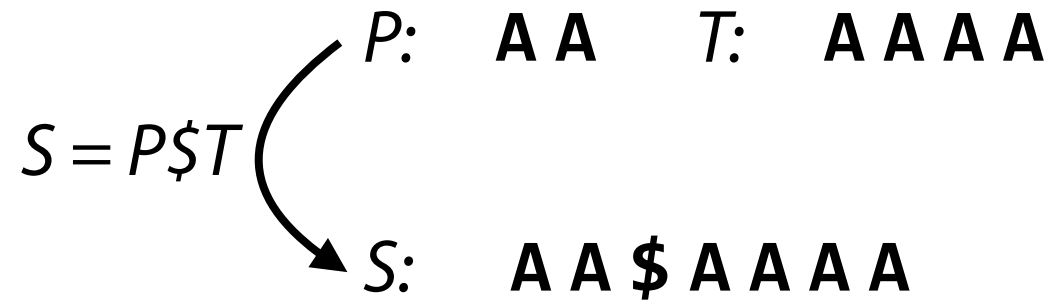
Given a Z_i value calculator, how do we solve pattern matching?



Z-value Pattern Matching

To solve pattern matching (given P and T), let $S = P\$T$

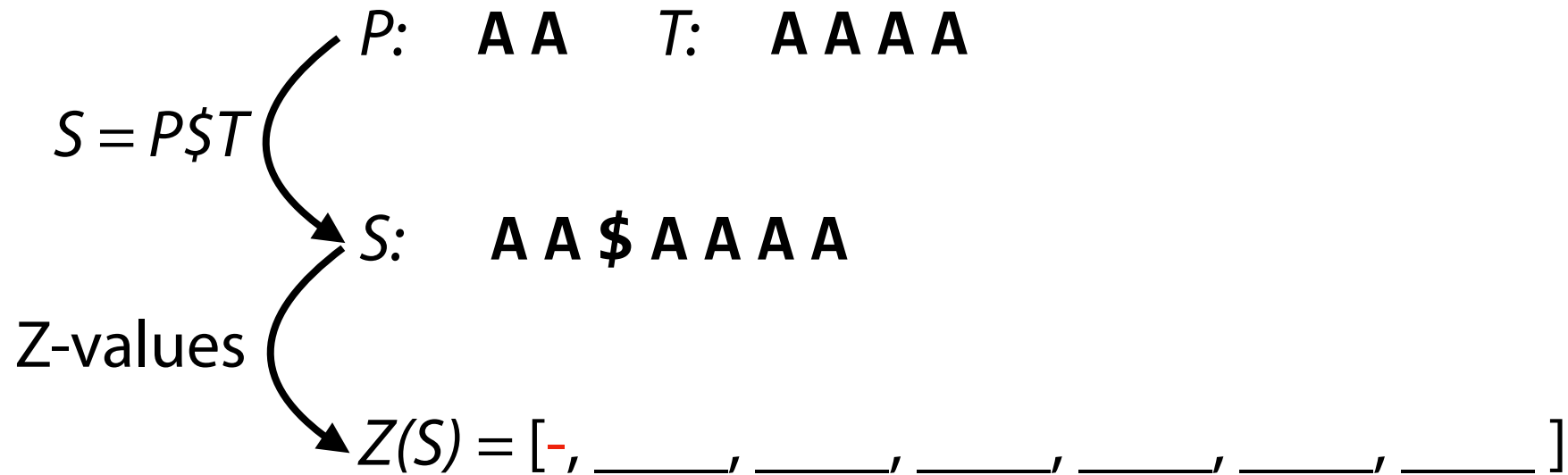
$\$$ = 'terminal character', outside alphabet



Z-value Pattern Matching

To solve pattern matching (given P and T), let $S = P\$T$

$\$$ = 'terminal character', outside alphabet



Z-value Pattern Matching

To solve pattern matching (given P and T), let $S = P\$T$

$\$$ = 'terminal character', outside alphabet

P : A A T : A A A A

0 1 2 3 4 5 6
 S : A A \$ A A A A
 0 1 2 3

$Z(S) = [-, 1, 0, 2, 2, 2, 1]$

What Z_i values are matches?

What are the matching indices in T ?

Z-value Pattern Matching



$P:$ TT $T:$ CTTA

Z-value search pseudo-code

$S:$

1. Concatenate ($S=P\$T$)

$Z(S):$

2. Calculate Z-values for S

3. For $i < 0$, match if $Z_i =$ _____

Match is **not** at i , but instead at

Assignment 2: a_zval

Learning Objective:

Construct a Z-value calculator and measure its efficiency

Demonstrate use of Z-values in pattern matching

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

End-of-class brainstorm

What information does a single Z-value tell us?

If I know $Z_{i-1}(S)$, can I use that information to help me compute $Z_i(S)$?

The Z-value (Take 2)

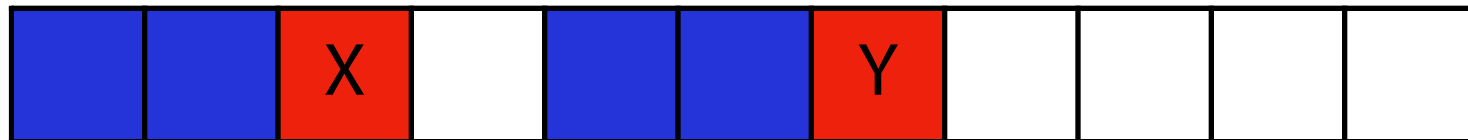
Given a string S , $Z_i(S)$ is the length of the longest substring in S , starting at position i , that matches a prefix of S .

What information does this give us?

S : **XYXYXABCD**

$$Z_4 = 2$$

0 1 2 3 4 5 6 7 8 9 ...

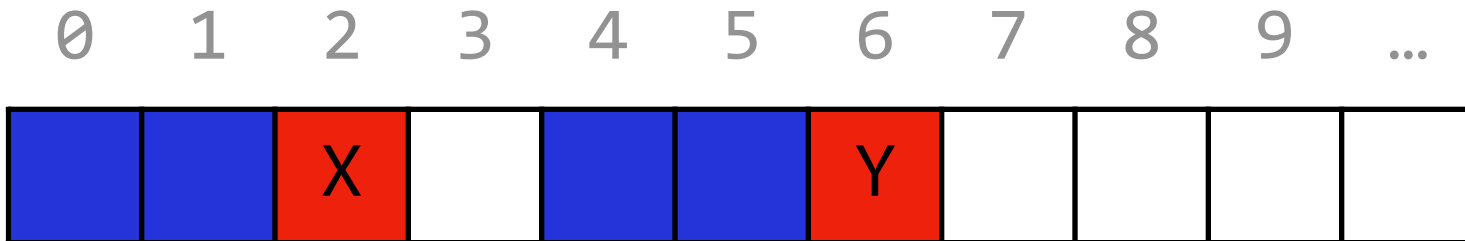


The Z-value (Take 2)

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

What information does this give us?

$\begin{matrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ S: & \boxed{T} & \boxed{T} & C & G & \boxed{T} & \boxed{T} & A & G & C & G \end{matrix} \quad Z_4 = 2$



The Z-Algorithm

Assume we've computed Z_1, \dots, Z_{i-1} and need to calculate Z_i

Case 1: We know nothing about the characters at $S[i]$

$Z_1 = ?$

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

Case 2: We know something about the characters at $S[i]$

$Z_2 = ?$

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

The Z-Algorithm

$$Z_1 = 3$$

$$Z_2 = ?$$

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 , the current index position being calculated

r , the index of the rightmost character which has ever been matched

l , the index of Z-value which r belongs too

The Z-Algorithm

Start

End

i , the current index =

r , the furthest match char =

l , the furthest reaching Z-value =

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

The Z-Algorithm

Start

End

i , the current index =

r , the furthest match char =

l , the furthest reaching Z-value =

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

The Z-Algorithm

Start

End

i , the current index =

r , the furthest match char =

l , the furthest reaching Z-value =

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

The Z-Algorithm

Start

End

i , the current index =

r , the furthest match char =

l , the furthest reaching Z-value =

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

The Z-Algorithm

Start

End

i , the current index =

r , the furthest match char =

l , the furthest reaching Z-value =

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

The Z-Algorithm

Start

End

i , the current index =

r , the furthest match char =

l , the furthest reaching Z-value =

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

The Z-Algorithm

Start

End

i , the current index =

r , the furthest match char =

l , the furthest reaching Z-value =

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

The Z-Algorithm



Intuition: We can use the previous Z_1, \dots, Z_i to compute Z_{i+1} !

Track 'what we know' using three integers: i, r, l

Next week: Review how integers are updated to define specific cases.