CMSC423: Bioinformatic Algorithms, Databases and Tools

Exact string matching:

The Z algorithm

- Recap: Naïve string matching algorithm runs in O(m n)
  m = len(pattern), n = len(text)

- Stop and think: What inefficiencies can you notice in the naïve algorithm?

Intuition

```
T  =  CCCCCCCCCCCCCC
       ||||X
P  =  CCCCG
        ||||X
       CCCCG
```

Naïve algorithm re-computes information it should already know

Specifically, once the Cs match at the first location, we should know they will match after a 1-letter shift

Can we capture the self-similarity of the string to help matching?

Quantifying self-similarity

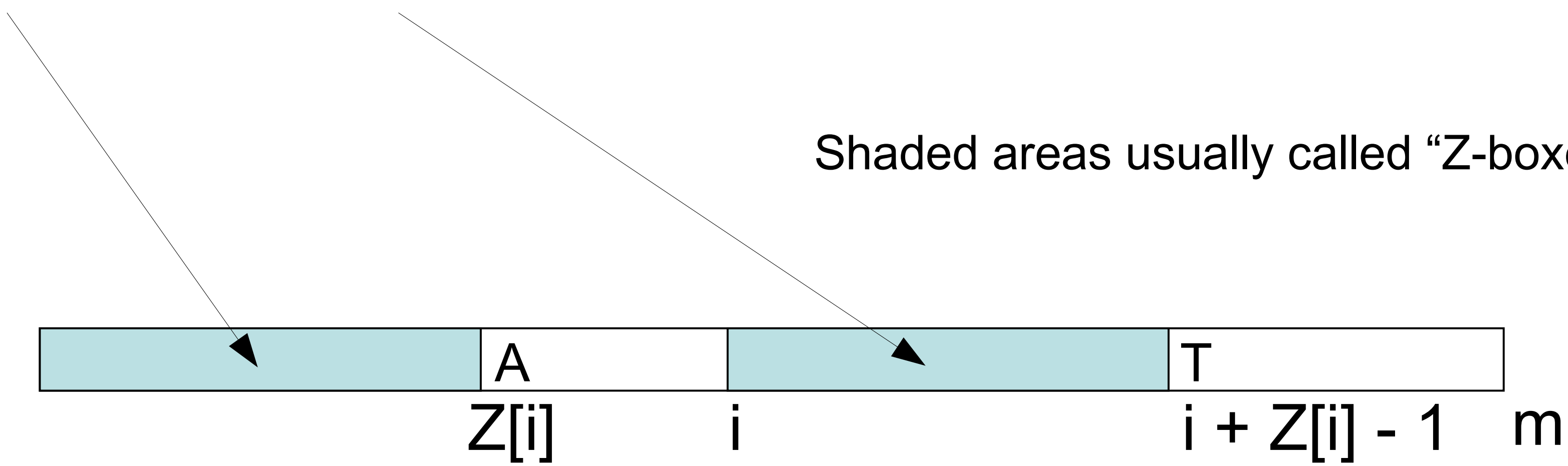
the Z algorithm (Gusfield)

For a string/text T

Z[i] = length of the longest prefix of T[i..m] that matches a prefix of T.
Z[i] = 0 if the prefixes don't match. Z[0] = 0 (by definition)

T[0 .. Z[i]] == T[i .. i+Z[i] -1]

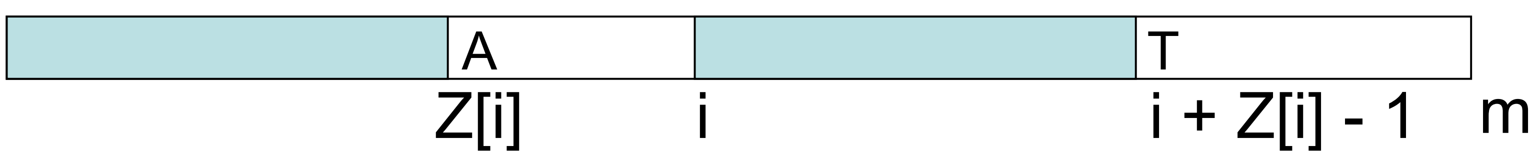
Shaded areas usually called “Z-boxes”



Quick aside: off-by-1 errors

- Is it i, or i + 1?
- Is a range inclusive or exclusive?
- Do coordinates start at 0 or 1?

- These are common issues that arise when implementing string algorithms.
- It is important to carefully trace an example on paper when writing your code. It will save you hours of debugging.



AGGTCCTAGGCGCT

i = 7
Z[i] = 3

Are the numbers in the diagram above correct, or off by 1?

Example Z values

Please write the Z values below each character in this string.

```
ACAGGTACAGTTCCCTCGACACCTACTACCTAAG
```

Example Z values

```
ACAGGTACAGTTCCCTCGACACCTACTACCTAAG
0010004010000000003020002002000110
```

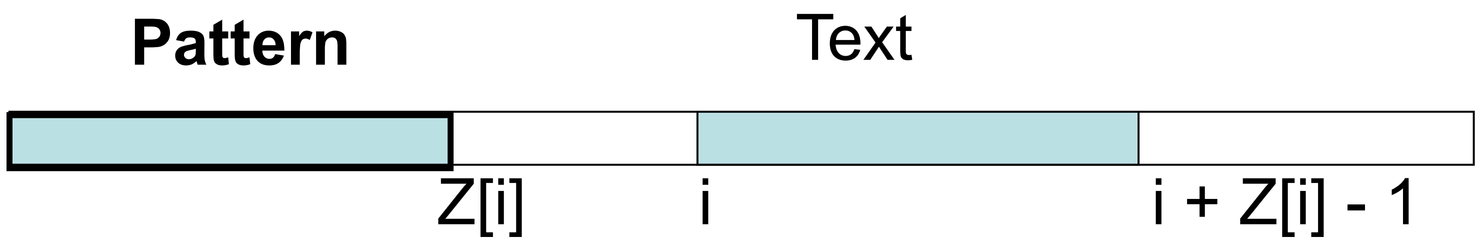
Stop and Think!

- We only talked about one string

- If you are given a pattern P and a text T, can you use the Z values to find out if the pattern matches the text, and where?

Can the Z values help in matching?

Create string Pattern$Text where $ is not in the alphabet



If there exists i, such that Z[i] = length(Pattern)
Pattern occurs in the Text starting at i

Stop and think: Assuming Z values are computed. What is the runtime?

Example

```
CCTACT$ACAGGTACAGTTCCCTCGACACCTACTACCTAAG
01001000100000100002310100106100100410000
```

runtime = O(m + n)   (simply scan an array to find the matches)

- Stop and think! What is the largest Z value possible?

NEXT: Can you compute the Z values efficiently?