CMSC423: Bioinformatic Algorithms, Databases and Tools

Exact string matching:

KMP algorithm

- Recap: Z values can be constructed in linear time
- Recap: Z values can be used to match strings in linear time

- Here: A more direct way of doing the matching

Revisiting the naïve matching

```
AAAAAAACAGTTCCCTCGACACCTACTACCTAAG
AAAAAT
  AAAAAT
   AAAAAT
    AAAAAT
```

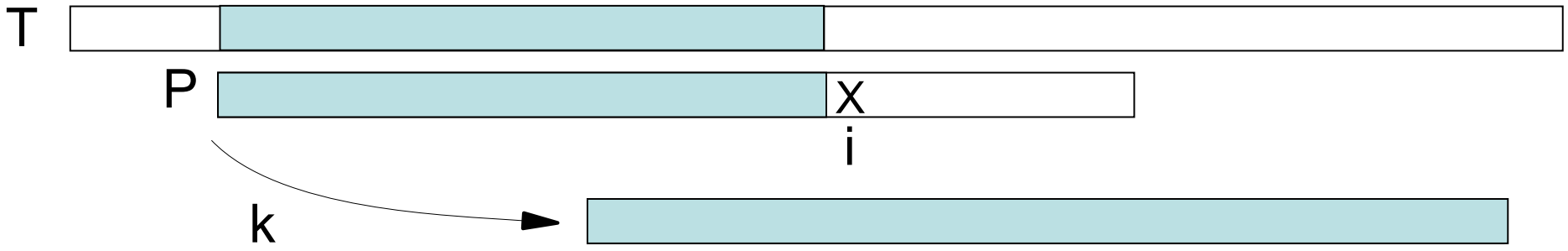
Text

Pattern

Intuition: After matching the characters in the box, we should know what matches exist after shifting the pattern.

Stop and Think!

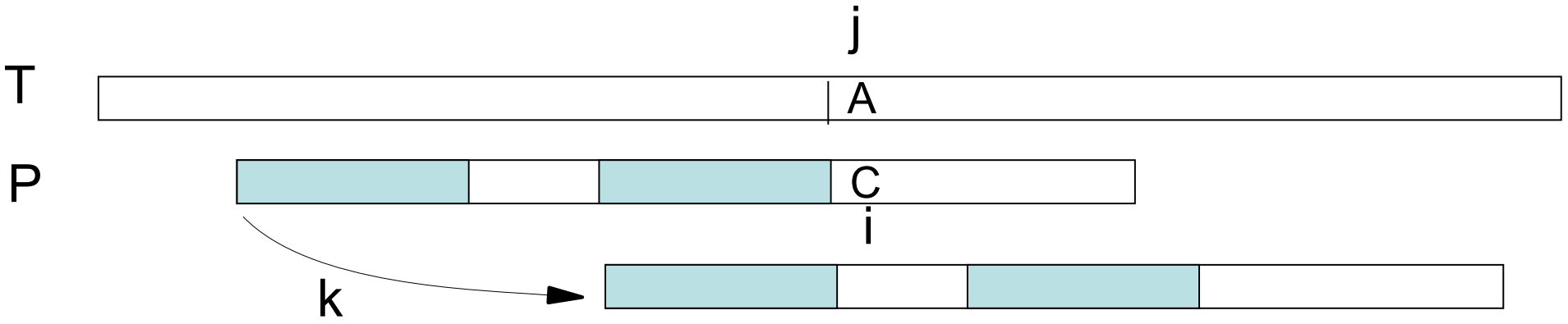
- Assume pattern matches the text up to position i in the pattern (see below)
- Assume the full pattern matches after a shift of k < i characters
- What relationships can we infer between substrings of the pattern?



The answer

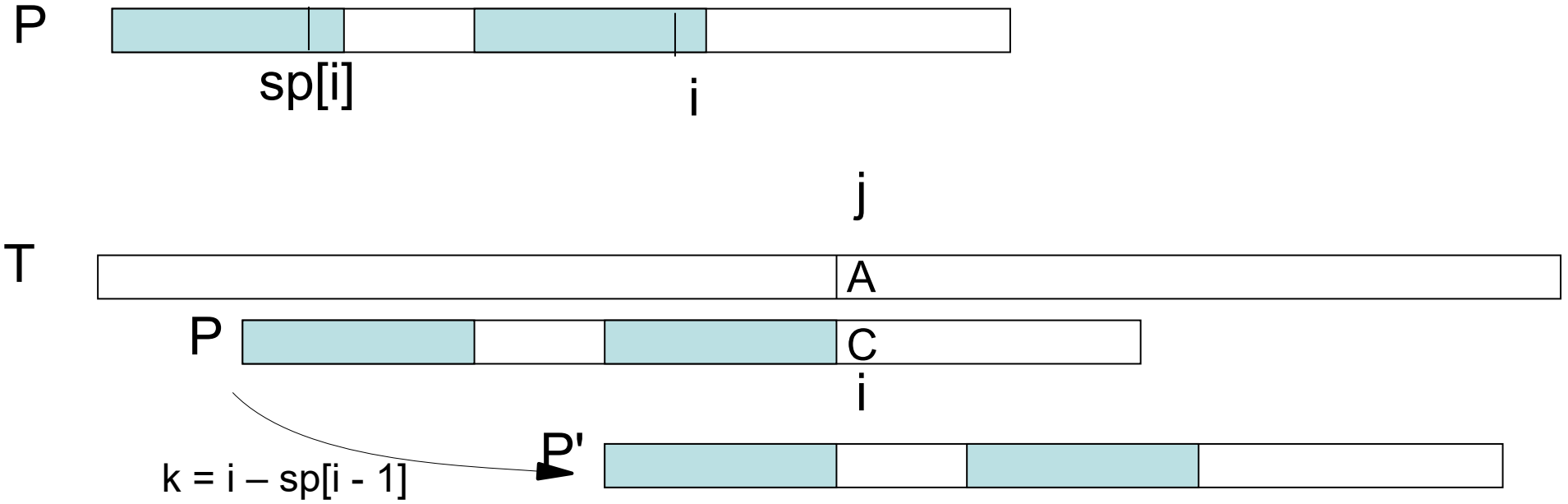
- The last i – k characters in the prefix of P that ends at position i match the first i – k characters of P

- Proof is obvious (?)



Knuth-Morris-Pratt algorithm

Given a Pattern and a Text, preprocess the Pattern to compute sp[i] = length of longest prefix of P that matches a suffix of P[0..i]

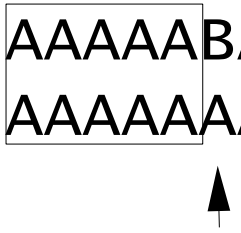


- Compare P with T until finding a mis-match (at coordinate i in P and j in T).

- Shift P such that first sp[i - 1] characters match T[j – sp[i - 1] + 1 .. j].
- Continue matching from T[j], P[sp[i – 1] + 1]

Walk-through

```
index:   0123456
pattern: AAAAAAA
sp:      0123456
```



First 5 characters match.

Mismatch at position i = 5

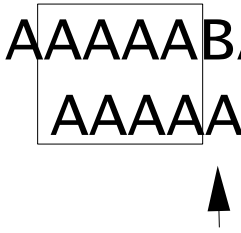
sp[i – 1] = sp[4] = 4

shift by i – sp[i - 1] = 5 – 4 = 1

number of comparisons = 6 (5 matched, one didn’t)

Walk-through

```
index:   0123456
pattern: AAAAAAA
sp:      0123456
```



First 4 characters match – no need to check.
Mismatch at position i = 4

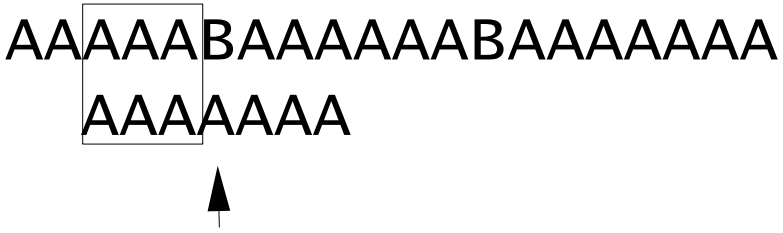
sp[i – 1] = sp[3] = 3

shift by i – sp[i - 1] = 4 – 3 = 1

number of comparisons = 1

Walk-through

```
index:   0123456
pattern: AAAAAAA
sp:      0123456
```

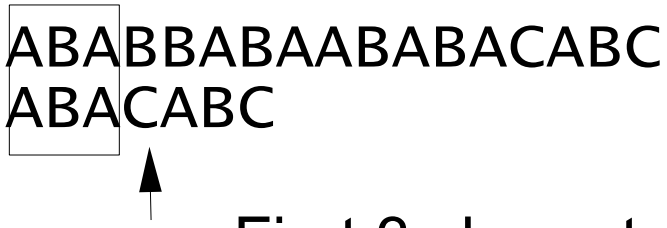


Keep checking the position marked with an arrow
number of comparisons = 1

and so on....

One more walkthrough

```
index:   0123456
pattern: ABACABC
sp:      0010120
```



First 3 characters match.
Mismatch at position i = 3

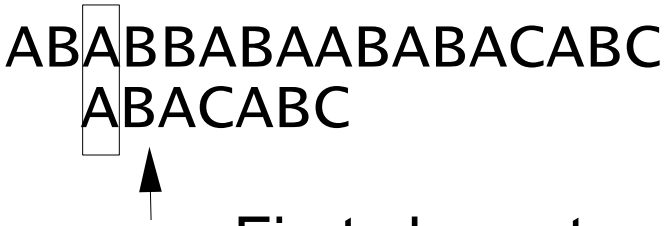
sp[i – 1] = 1

shift by i – sp[i – 1] = 3 – 1 = 2

number of comparisons = 4

One more walkthrough

```
index:   0123456
pattern: ABACABC
sp:      0010120
```



First character matches – no need to check.
Mismatch at position i = 1

sp[i – 1] = 0

shift by i – sp[i – 1] = 1

number of comparisons = 1

... and so on

KMP – Stop and Think!

- Does it work?

- Can you miss a match by shifting too far?

- How do you prove that?

Next: Run-time & computing sp values