COL 351:
Analysis and Design of Algorithms

Lecture 14

String Matching

Given: String $S = [s_1, \ldots, s_n]$
pattern $P = [p_1, \ldots, p_k]$.

Find: Does there exists a sub-string of S that is identical to P.

Example:

S = "cuckoo **hash**ing is efficient"

P = "hash"

Yes

Subproblem

Given: Pattern $P = [p_1, \ldots, p_k]$.

Find: Table of size k satisfying

Table[i] = Length of longest common prefix and suffix of P[1, i]

i	1	2	3	4	5	6
P[i]	A	B	C	A	B	B
Table[i]	0	0	0	1	2	0

Question:

Can we find **All** common prefixes/ suffixes of $P[1, i]$?

Eg. P = (A B A B A B A B C)

the length of ALL common prefixes / suffixes of P[1...8] is 6, 4, 2

Subproblem

Given: Pattern $P = [p_1, \ldots, p_k]$.

Find: Table of size k satisfying

Table[i] = Length of longest common prefix and suffix of P[1, i]

i	1	2	3	4	5	6
P[i]	A	B	C	A	B	B
Table[i]	0	0	0	1	2	0

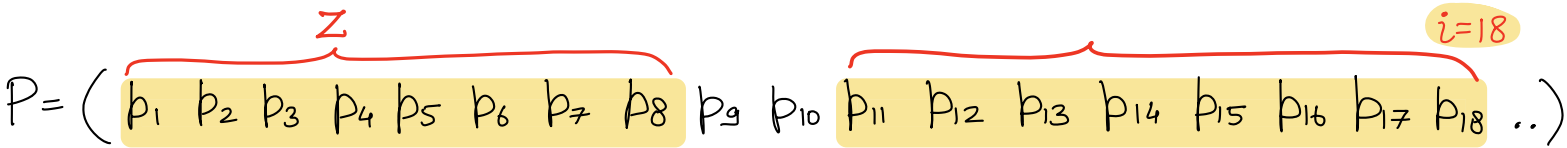
Subproblem

Lemma: Table[i] , Table[Table[i]] , Table[Table[Table[i]]] , ...

gives the length of ***ALL*** common prefixes-suffixes of $P[1, i]$.

Proof idea:

Table [18] = 8



Any common prefix / suffix of P [1, 18]

→ Either Z , or

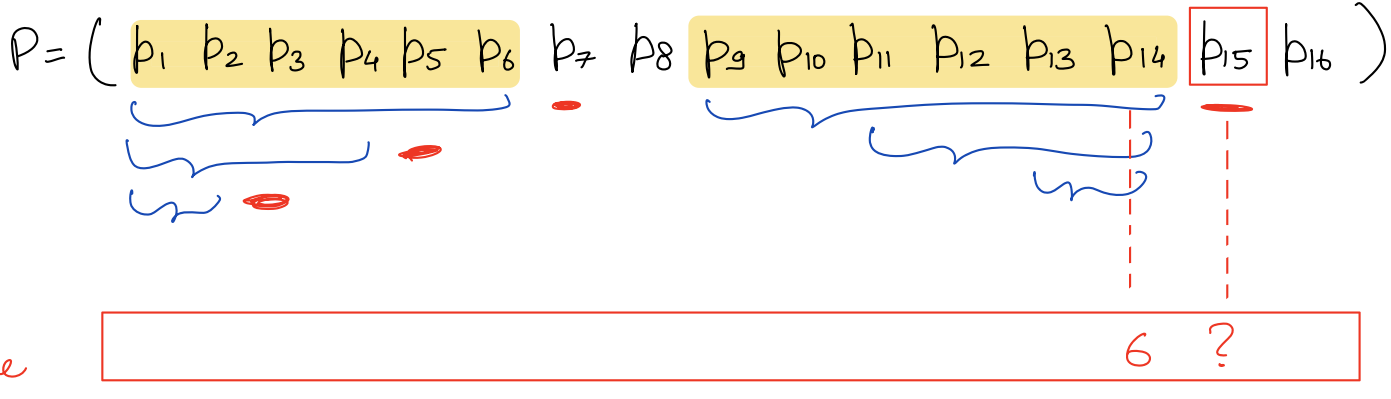
→ A common prefix / suffix of Z

Subproblem

Lemma: Table[i] , Table[Table[i]] , Table[Table[Table[i]]] , ...

gives the length of ***ALL*** common prefixes-suffixes of $P[1, i]$.

Question: *How to compute Table*[$i+1$]?



Subproblem

Lemma: Table[i] , Table[Table[i]] , Table[Table[Table[i]]] , ...

gives the length of ***ALL*** common prefixes-suffixes of $P[1, i]$.

Question: *How to compute Table*[$i+1$]?

L = Table[i];

While ($L > 0$ and P[$i+1$] ≠ P[L+1]): L = Table[L];

If (P[$i+1$] = P[L+1]): $L = L + 1$;

Table[$i+1$] = L;

Values of L:

Table[i], Table[Table[i]], T[T[T[i]]], ...

Stopping Criteria

$L>0$ $P[i+1] = P[L+1]$	$L=0$ $P[i+1] = P[L+1]$	$L=0$ $P[i+1] \neq P[L+1]$
Table[i+1] = L+1	Table[i+1] = 1	Table[i+1] = 0

Subproblem

Table[i] = Length of longest (non-trivial) common prefix and suffix of P[1, i]

```
Table ← Array of size k;
Table[1], L = 0;
For (i = 1 to k − 1):
    /* value of L is Table[i] */
    While (L > 0 and P[i +1] ≠ P[L+1]):  L = Table[L];
    If (P[i +1] = P[L+1]):  L = L + 1;
    Table[i +1] = L;
```

Subproblem

Table[i] = Length of longest (non-trivial) common prefix and suffix of P[1, i]

Table ← Array of size k;

Table[1], $L = 0$;

For ($i = 1$ to $k - 1$):

/* value of L is Table[i] */

While ($L > 0$ and P[$i+1$] ≠ P[$L+1$]): L = Table[L];

If (P[$i+1$] = P[$L+1$]): $L = L + 1$;

Table[$i+1$] = L;

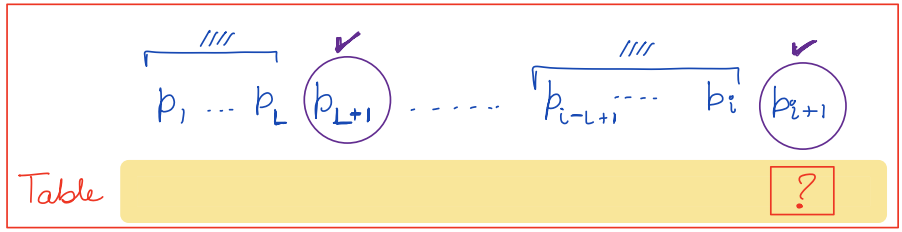
Invariant:
When FOR LOOP starts we have L = Table[i].

Values of L:
Table[i], Table[Table[i]], T[T[T[i]]], ...

STOPPING CRITERIA

P[$i+1$] = P[$L+1$]

$L = 0$
- P[1] = P[$i+1$] → Table[$i+1$] = 1
- P[1] ≠ P[$i+1$] → Table[$i+1$] = 0, L = 0



Subproblem

Table[i] = Length of longest (non-trivial) common prefix and suffix of P[1, i]

```
Table ← Array of size k;
Table[1], L = 0;
For (i = 1 to k − 1):
    /* value of L is Table[i] */
    While (L > 0 and P[i +1] ≠ P[L+1]):  L = Table[L];
    If (P[i +1] = P[L+1]):  L = L + 1;
    Table[i +1] = L;
```

Analysis:

Claim 1: L is incremented at most k times.

Claim 2: Throughout the algorithm L can decrease at most k times, so total number of iterations of While loop is at most k.

Therefore, time complexity is O(k).

Main Problem

Given: String $S = [s_1, \ldots, s_n]$ and pattern $P = [p_1, \ldots, p_k]$.

Find: Does there exists a sub-string of S that is identical to P.

Example:

S = "cuckoo **hash**ing is efficient"

P = "hash"

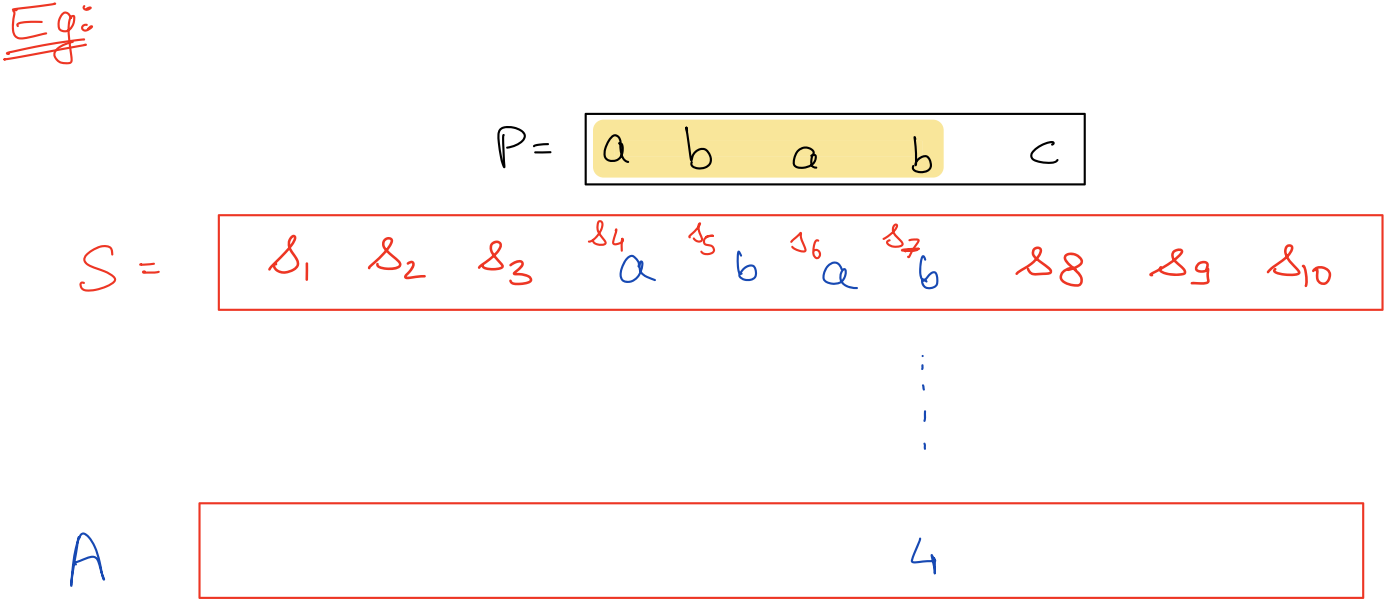
Yes

Main Problem

Table[i] := Length of longest (non-trivial) common prefix and suffix of P[1, i]

A[i] := Length of longest prefix of P that is also a suffix of S[1, i]

Eg:

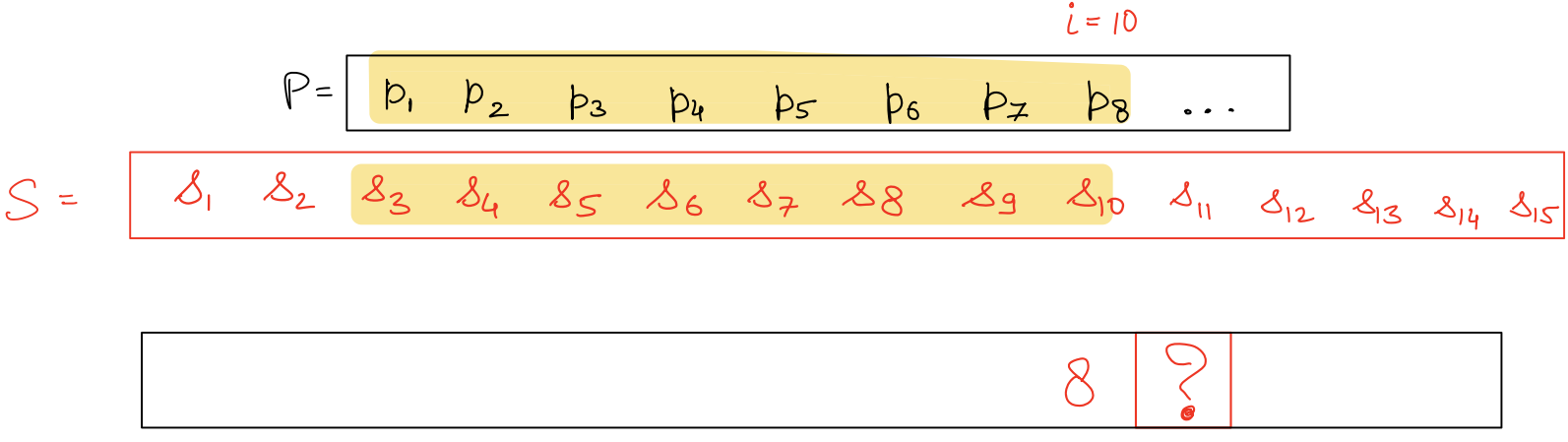


Main Problem

Table[i] := Length of longest (non-trivial) common prefix and suffix of P[1, i]

A[i] := Length of longest prefix of P that is also a suffix of S[1, i]

Question: *How to find* A[$i + 1$]?



Ques: How can you find ALL prefixes of P that are suffixes of $(s_1 \ldots s_{10})$?

Ans: By sequence: L = A[i], Table[L], Table[Table[L]], ...

Main Problem

Table[i] := Length of longest (non-trivial) common prefix and suffix of P[1, i]

A[i] := Length of longest prefix of P that is also a suffix of S[1, i]

Question: *How to find* A[i + 1]?

L = A[i];

While ($L > 0$ and S[i +1] $\neq$ P[L+1]): L = Table[L];

If (S[i +1] = P[L+1]): $L = L + 1$;

A[i +1] = L;

Main Problem

Table[i] := Length of longest (non-trivial) common prefix and suffix of P[1, i]

A[i] := Length of longest prefix of P that is also a suffix of S[1, i]

A ← Array of size $n + 1$ with $A[0] = 0$;

$L \leftarrow 0$;

For ($i = 0$ to $n - 1$):

/* value of L is A[i] */

While ($L > 0$ and S[i +1] ≠ P[L+1]): L = Table[L];

If (S[i +1] = P[L+1]): $L = L + 1$;

A[i +1] = L;

Knuth–Morris–Pratt (KMP) algorithm