String matching algorithms

Given a text S and a pattern T, decide if T is a substring of S.

Naive algo: try all possible starting location in S
compare with T.
$O(|S| \cdot |T|)$ time.

Last time: build suffix tree of S $O(|S|)$ time
search for T $O(|T|)$ time

Today: two more algorithms in $O(|S|+|T|)$ time.

Knuth-Morris-Pratt (KMP)

Idea:

	i	i+1	+2	+3	+4	+5	+6	
S	a	a	a	b	a	a	a	...
T	a	a	a	b	a	a	b	...

mismatch at $S[i+6]$ when start at $S[i]$.

We don't need to start at $S[i+1]$

- $S[i+1], \ldots, S[i+5]$ must be "aabaa", not matching $T[0], \ldots, T[4]$

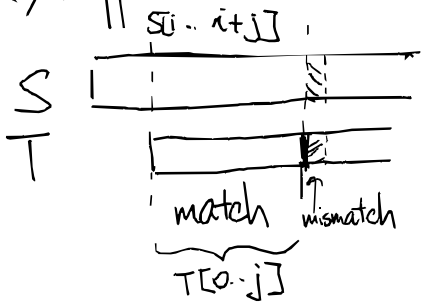
don't need to start at $S[i+2]$ either

- $S[i+2], \ldots, S[i+5]$ ... "abaa" ... $T[0], \ldots, T[3]$.

We can start at $S[i+4]$.

- $S[i+4], S[i+5]$ must be "aa" $=$ $T[0], T[1]$.
- can continue comparing $S[i+6]$ with $T[2]$...

In general, suppose



Then we don't have to compare T with $S[i+\Delta, \ldots]$ if $T[0..j-\Delta] \neq T[\Delta..j]$

because $S[i+\Delta..i+j] = T[\Delta..j] \neq T[0..j-\Delta]$

hence, $S[i+\Delta..] \neq T$

KMP preprocesses T so that it computes

- for every j, the smallest $\Delta > 0$, s.t.

$$T[0..j-\Delta] = T[\Delta..j]$$

for each prefix P_i the longest (proper) suffix of P_i that is also a prefix of P_i.

Denote the length $(j-\Delta+1)$ by $\pi[j]$

	0	1	2	3	4	5	6
T	a	a	a	b	a	a	b
π	0	1	2	0	1	2	0

Computing π

$\pi[0]=0$

for $j=1$ to $|T|-1$

$x=\pi[j-1]$

while $T[x] \neq T[j]$ and $x>0$

$x=\pi[x-1]$

if $T[x]=T[j]$

$\pi[j]=x+1$

else

$\pi[j]=0$ $\quad (x=0$ & $T[0] \neq T[j])$

Correctness.

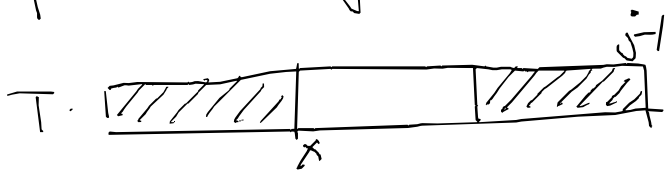
$$\pi[j] = \max_x \{x : T[0..x] = T[j+1-x..j]\}$$

$$= \max_x \{x : \text{(} T[0..x-1] = T[j+1-x..j-1] \text{)} \ \& \ T[x] = T[j]\}$$

Condition ◯ : length-x suffix of $T[0..j-1]$ is also a prefix.

Claim: The "while loop" enumerates all x satisfying ◯, from large to small and checks if $T[x] = T[j]$.

Proof sketch: The largest x s.t. ◯ holds is $\pi[j-1]$ (by definition)



find the largest $x' < x$ s.t. ◯ holds

$$\Updownarrow$$

$$T[0..x'-1] = \underbrace{T[j+1-x'..j-1]}$$

$$= T[x+1-x'..x-1]$$

$$\Rightarrow x' = \pi[x-1].$$

□

Running time

Everything other than the "while loop": $O(|T|)$ time

The "while loop":

x starts at $\pi[j-1]$

ends at $\pi[j]-1$ (or $\pi[j]$)

each iteration of while

x decreases.

For each j, "while" takes time $\leq O(\pi[j]-\pi[j-1]+1)$

$$\sum_j O(\pi[j]-\pi[j-1]+1) = O(|T|) + O(\pi[|T|-1]-\pi[0])$$
$$= O(|T|).$$

Matching

```
k=0
for i=0 to |S|-1
    while T[k]≠S[i] and k>0
        k=π[k-1]
    if T[k]=S[i]
        k=k+1
    if k=|T|, return true
return false
```

(T[0..k-1] already matches S[i-k..i-1])

Rabin-Karp

Idea: hash T into an integer

also hash all length-$|T|$ substrings of S.

compare

Suppose $S, T \in \{0, 1, \ldots, 9\}^*$

$T = 12351$

$S = 2563148762\ldots$

hash: view T as the integer 12351, and mod Q

for some Q

also need to hash

25631

56314

63148

$\ddots$

can compute these hashes iteratively.

$$16314 \bmod Q$$

$$= ((25631 \bmod Q) - (2 \times 10^4 \bmod Q)) \times 10 + 4 \bmod Q$$

Each hash can be computed in $O(1)$ time given the previous hash

Time $O(|S|+|T|)$ to compute all hashes and compare.

For general alphabet Σ, view strings as "integers" in base $|\Sigma|$.

Correctness.

If $T = S[i..i+|T|-1]$, then the hashes must equal.

If $T \neq S[i..i+|T|-1]$, then

hashes equal iff $\underbrace{T - S[i..i+|T|-1]}_{\text{as integers}}$ is a multiple of Q.

Also, $|T - S[\quad\quad\quad]| \leq 10^{|T|}$

Any integer N can have $\leq O(\frac{\log N}{\log\log N})$ prime factors.

There are $\Omega(\frac{W}{\log W})$ primes in $[1, W]$.

Picking Q to be a random prime $\leq \text{poly}(|S|, |T|)$ works with high prob.