

String Matching

Input is a pattern $P[1 \dots m]$ and text $T[1 \dots n]$.
 Find a shift s s.t. $P[i] = T[s+i]$ for $1 \leq i \leq m$.
 Common application: grep, document search

NAIVE-STRING-MATCHER(T, P)

$n \leftarrow \text{length}(T), m \leftarrow \text{length}(P)$

for $s \leftarrow 0$ **to** $n - m$

do if $P[1 \dots m] = T[s + 1 \dots s + m]$
 then pattern occurs with shift s

NAIVE-STRING-MATCHER is $O(nm)$.

Consider $P = \text{aaaaa}$ and $T = \text{aaaabaaaab} \dots$

Rabin-Karp-Matcher

$P[1 \dots m]$ can be converted to a number:

$$p = \sum_{i=0}^{m-1} P[m-i] * d^i$$

where each character is in radix- d notation.

Similarly, $T[s + 1 \dots s + m]$ can be converted:

$$t = \sum_{i=0}^{m-1} T[s+m-i] * d^i$$

Idea: test $p \bmod q = t \bmod q$ before the more expensive $P[1 \dots m] = T[s + 1 \dots s + m]$.

Use $q = \text{prime}, q > m, q$ far from powers of 2.

```

RABIN-KARP-MATCHER( $T, P, d, q$ )
   $n \leftarrow \text{length}(T), m \leftarrow \text{length}(P)$ 
   $p \leftarrow 0, t \leftarrow 0$ 
  for  $i \leftarrow 1$  to  $m$  do
     $p \leftarrow (d * p + P[i]) \bmod q$ 
     $t \leftarrow (d * t + T[i]) \bmod q$ 
  for  $s \leftarrow 0$  to  $n - m$  do
    if  $s > 0$ 
      then  $t \leftarrow (d * t + T[s + m] - T[s] * d^m) \bmod q$ 
      if  $p = t$  and  $P[1 \dots m] = T[s + 1 \dots s + m]$ 
        then pattern occurs with shift  $s$ 

```

Average Case Analysis:

Probability Model:

Suppose there are v valid shifts.

If s is not a valid shift, then $p \bmod q = t \bmod q$ with probability $1/q$.

$O(m)$ per valid shift. $O(m)$ per “spurious hit.”

Expected number of spurious hits is $O(n/q)$.

$O(1)$ per iteration otherwise.

Expected running time is $O(m(v + n/q) + n)$, which is $O(mv + n)$ if $q \geq m$.

String Matching with Finite State Automata

A finite state automaton is defined by:

Q , a set of states

$q_0 \in Q$, the start state

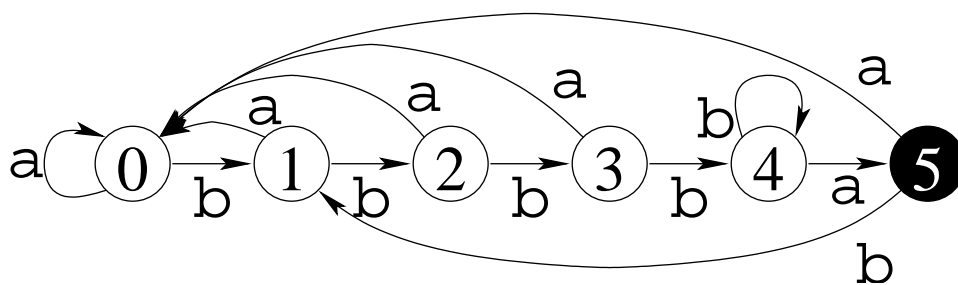
$A \subseteq Q$, the accepting states

Σ , the input alphabet

δ , the transition function, from $Q \times \Sigma$ to Q

$$\delta(q_i, x) = \begin{cases} q_{i+1} & \text{if } i < m \text{ and } x = P[i + 1] \\ q_j & \text{if } j \text{ is the max. value such that } x = P[j] \\ & \text{and } P[1 \dots j - 1] = P[i - j + 2 \dots i] \\ q_0 & \text{otherwise} \end{cases}$$

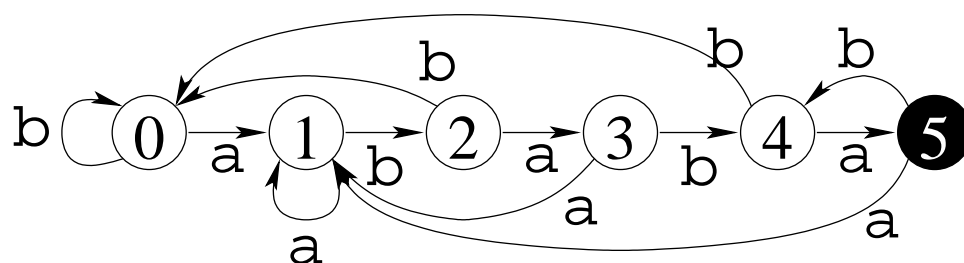
FSA for matching **bbbba**.



Transition Function

		States					
		0	1	2	3	4	5
Inputs	a	0	0	0	0	5	0
	b	1	2	3	4	4	1

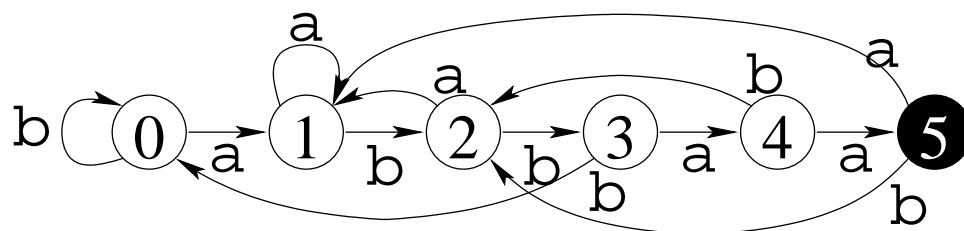
FSA for matching ababa



Transition Function

		State					
		0	1	2	3	4	5
Inputs	a	1	1	3	1	5	1
	b	0	2	0	4	0	4

FSA for matching abbaa



Transition Function

		State					
		0	1	2	3	4	5
Inputs	a	1	1	1	4	5	1
	b	0	2	3	0	2	2

Knuth-Morris-Pratt Algorithm

The Knuth-Morris-Pratt algorithm efficiently implements finite state automata. It is based on computing a prefix function:

$$\pi[q] = \max\{k : k < q \text{ and } P_k \text{ is a suffix of } P_q\}$$

where $0 \leq k < q \leq m$ and $P_k = P[1 \dots k]$.

P_k is a suffix of P_q if $P_k = P[q - k + 1 \dots q]$

COMPUTE-PREFIX-FUNCTION(P)

$m \leftarrow \text{length}(P)$

$\pi[1] \leftarrow 0$

$k \leftarrow 0$

for $q \leftarrow 2$ **to** m

do while $k > 0$ and $P[k + 1] \neq P[q]$

do $k \leftarrow \pi[k]$

if $P[k + 1] = P[q]$

then $k \leftarrow k + 1$

$\pi[q] \leftarrow k$

return π

Running time is $O(m)$. Count changes to k .
 $\pi[k] < k$, so $k \leftarrow \pi[k]$ decreases k .
 k is incremented $m - 1$ times and $k \geq 0$, so
 k can be decreased at most $m - 1$ times.

Show correctness of prefix computation.

Loop invariant is $\pi[q - 1] = k$.

$\pi[q - 1] = k$ before the first iteration.

In while loop,

If $P[q] = P[k + 1]$, then $\pi[q] = k + 1$.

If $P[q] \neq P[k + 1]$, then check $\pi[k]$ next

because $P_{\pi[k]}$ is a suffix of both P_k and P_{q-1} .

KMP-MATCHER(T, P)

$n \leftarrow \text{length}(T)$, $m \leftarrow \text{length}(P)$

$\pi \leftarrow \text{COMPUTE-PREFIX-FUNCTION}(P)$

$q \leftarrow 0$ \triangleright q is the state of the FSA.

for $i \leftarrow 1$ **to** n

do while $q > 0$ and $P[q + 1] \neq T[i]$

do $q \leftarrow \pi[q]$

if $P[q + 1] = T[i]$ **then** $q \leftarrow q + 1$

if $q = m$

then pattern occurs with shift $i - m$

$q \leftarrow \pi[q]$

Running time is $O(n+m)$. Count changes to q .
 $\pi[q] < q$, so $q \leftarrow \pi[q]$ decreases q .
 q is incremented $O(n)$ times and $q \geq 0$, so
 q can be decreased at most $O(n)$ times.

Show correctness of computation.

Loop invariant is $P_q = T[i - q \dots i - 1]$.

This is true before the first iteration.

In while loop,

If $P[q + 1] = T[i]$, then q is incremented.

If $P[q] \neq T[i]$, then check $\pi[q]$ next because
 $P_{\pi[q]}$ is also a suffix of T_{i-1} .