String Matching

1

String Matching

DESIGN &
ANALYSIS OF
ALGORITHM

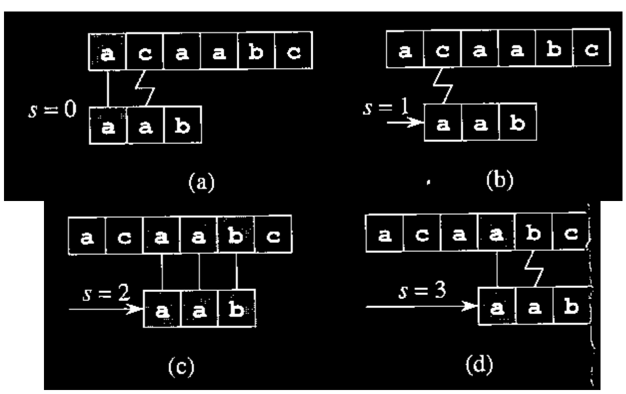
- T= text
- P=pattern
- n=text length
- m=pattern length.
- Z=alphabet size.

ALG99F, javed@kent.edu

Naïve Method

3

Naïve Method



Complexity?

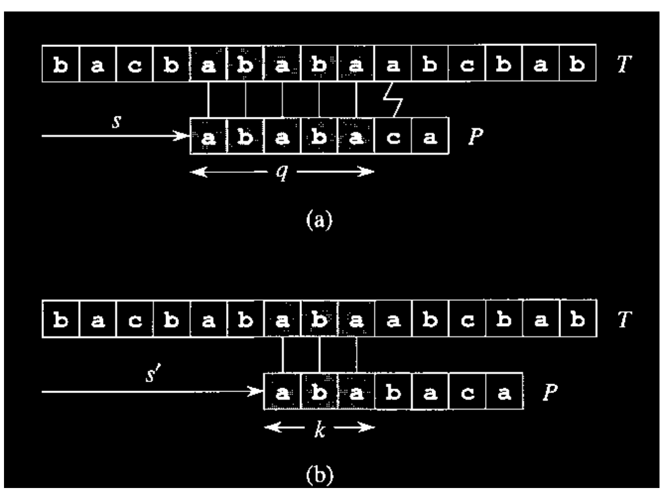
DESIGN & ANALYSIS OF ALGORITHM

ALG99F, javed@kent.edu

Knuth-Morris-Pratt Algorithm

5

Idea of Jumping



DESIGN & ANALYSIS OF ALGORITHM

LECT-15, S-6
ALG99F, javed@kent.edu
Javed I. Khan@1999

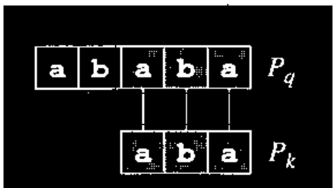
Jump Table

i	1	2	3	4	5	6	7	8	9	10
$P[i]$	a	b	a	b	a	b	a	b	c	a
$\pi[i]$	0	0	1	2	3	4	5	6	0	1

KMP Algorithm


```
KMP-MATCHER(T, P)
n ← length[T]
m ← length[P]
π ← COMPUTE-PREFIX-FUNCTION(P)
q ← 0
for i ← 1 to n
     do while q > 0 and P[q + 1] ≠ T[i]
             do q ← π[q]
        if P[q + 1] = T[i]
          then q ← q + 1
        if q = m
          then print "Pattern occurs with shift" i − m
               q ← π[q]
```

Computing Jumps



i	1	2	3	4	5	6	7	8	9	10
$P[i]$	a	b	a	b	a	b	a	b	c	a
$\pi[i]$	0	0	1	2	3	4	5	6	0	1

```
COMPUTE-PREFIX-FUNCTION(P)
  m ← length[P]
  π[1] ← 0
  k ← 0
  for q ← 2 to m
      do while k > 0 and P[k + 1] ≠ P[q]
             do k ← π[k]
         if P[k + 1] = P[q]
           then k ← k + 1
         π[q] ← k
  return π
```

- Jumps can be computed by preprocessing the pattern, by comparing the pattern with itself

DESIGN & ANALYSIS OF ALGORITHM

Complexity

- The running time for COMPUTE-PREFIX-FUNCTION is:
  - O(m)
- The total jump cannot exceed (m+n),= Thus complexity is O(m+n).

DESIGN & ANALYSIS OF ALGORITHM

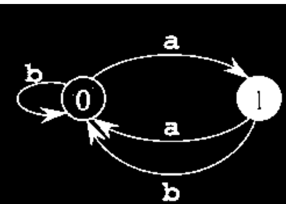
State Machine

11

FSM for Detecting String Which Ends with Even 1s

state	input a	input b
0	1	0
1	0	0

(a)



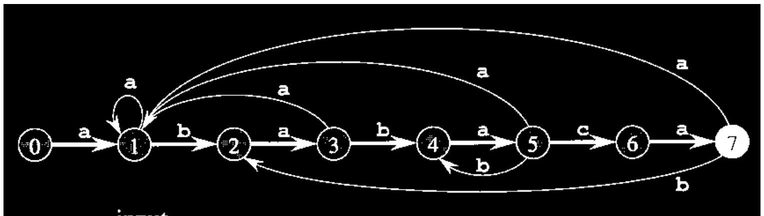
(b)

Figure 34.5 A simple two-state finite automaton with state set $Q = \{0, 1\}$, start state $q_0 = 0$, and input alphabet $\Sigma = \{\mathtt{a}, \mathtt{b}\}$. **(a)** A tabular representation of the transition function δ. **(b)** An equivalent state-transition diagram. State 1 is the only accepting state (shown blackened). Directed edges represent transitions. For example, the edge from state 1 to state 0 labeled b indicates $\delta(1, \mathtt{b}) = 0$. This automaton accepts those strings that end in an odd number of a's. More precisely, a string x is accepted if and only if $x = yz$, where $y = \varepsilon$ or y ends with a b, and $z = \mathtt{a}^k$, where k is odd. For example, the sequence of states this automaton enters for input abaaa (including the start state) is $\langle 0, 1, 0, 1, 0, 1 \rangle$, and so it accepts this input. For input abbaa, the sequence of states is $\langle 0, 1, 0, 0, 1, 0 \rangle$, and so it rejects this input.

DESIGN & ANALYSIS OF ALGORITHM

ALG99F, javed@kent.edu
Javed I. Khan@1999

State Transition Diagram for String Matching



	input			
state	a	b	c	P
0	1	0	0	a
1	1	2	0	b
2	3	0	0	a
3	1	4	0	b
4	5	0	0	a
5	1	4	6	c
6	7	0	0	a
7	1	2	0	

i	—	1	2	3	4	5	6	7	8	9	10	11
$T[i]$	—	a	b	a	b	a	b	a	c	a	b	a
state $\phi(T_i)$	0	1	2	3	4	5	4	5	6	$\boxed{7}$	2	3

Operation of FSM on string abababacaba

DESIGN & ANALYSIS OF ALGORITHM

Complexity

- Fastest Compute State Transition is:
  - O(m z)
- Total WC running time is:
  - O(n+mz)

DESIGN & ANALYSIS OF ALGORITHM

Boyer-Moore Algorithm (1976)

15

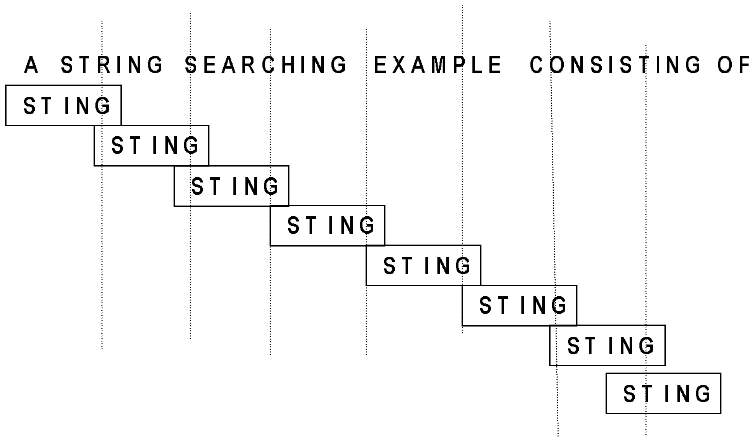
Boyer-Moore Algorithm

DESIGN & ANALYSIS OF ALGORITHM

- Comparing from the Right to Left:
  - in the pattern, each time, there is a mismatch, see how many position the pattern can be shifted left.
- More Look ahead in the Preprocessing
  - bring into consideration the character that caused the mismatch while considering what to do next.

LECT-15, S-16
ALG99F, javed@kent.edu

Example



DESIGN & ANALYSIS OF ALGORITHM

Complexity

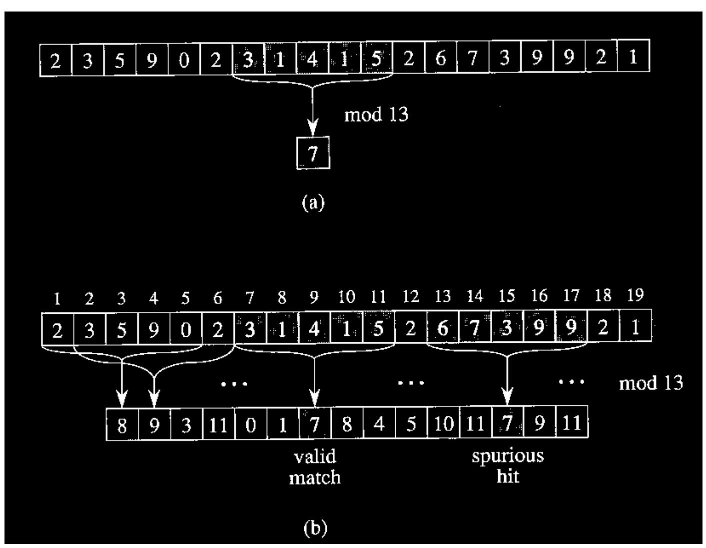
- Boyer-Moore string search algorithm never uses more than M+N character comparisons, and uses about N/M steps of the alphabet is not small and the pattern is not long.

DESIGN & ANALYSIS OF ALGORITHM

Rabin-Karp Method (1980)

19

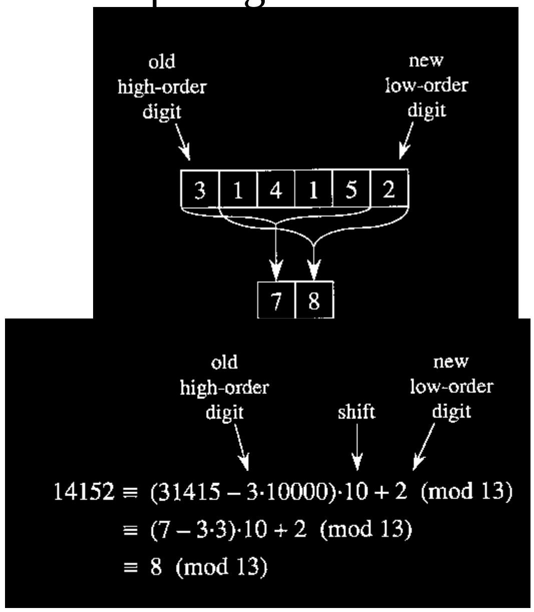
Rabin-Karp Algorithm



DESIGN & ANALYSIS OF ALGORITHM

LECT-15, S-20
ALG99F, javed@kent.edu

Computing Hash Value



DESIGN & ANALYSIS OF ALGORITHM

RK Algorithm

```
RABIN-KARP-MATCHER(T, P, d, q)
n ← length[T]
m ← length[P]
h ← d^(m-1) mod q
p ← 0
t_0 ← 0
for i ← 1 to m
      do p ← (dp + P[i]) mod q
         t_0 ← (dt_0 + T[i]) mod q
for s ← 0 to n − m
      do if p = t_s
            then if P[1..m] = T[s+1..s+m]
                    then "Pattern occurs with shift" s
         if s < n − m
            then t_{s+1} ← (d(t_s − T[s+1]h) + T[s+m+1]) mod q
```

- Radix is d. The prime is q.

DESIGN & ANALYSIS OF ALGORITHM

Complexity

DESIGN & ANALYSIS OF ALGORITHM

- In the worst case the running time is O((n-m+1) m).
  - (case $T=a^n$ and $P=a^m$)
  - Each evaluation after the first one is O(1) in text.
- Average Case Complexity
  - Only one match in most cases O(1)
  - Thus running time is O(n+m).

LECT-15, S-23
ALG99F, javed@kent.edu