UNIT 3: Space and Time Tradeoffs

Input Enhancement in String Matching: Boyer Moore

Boyer Moore algorithm

- Robert S. Boyer and J Strother Moore in 1977
- uses input enhancement idea: preprocesses the string being searched for (pattern), but not the string being searched in (text)

Boyer Moore algorithm: Strategy

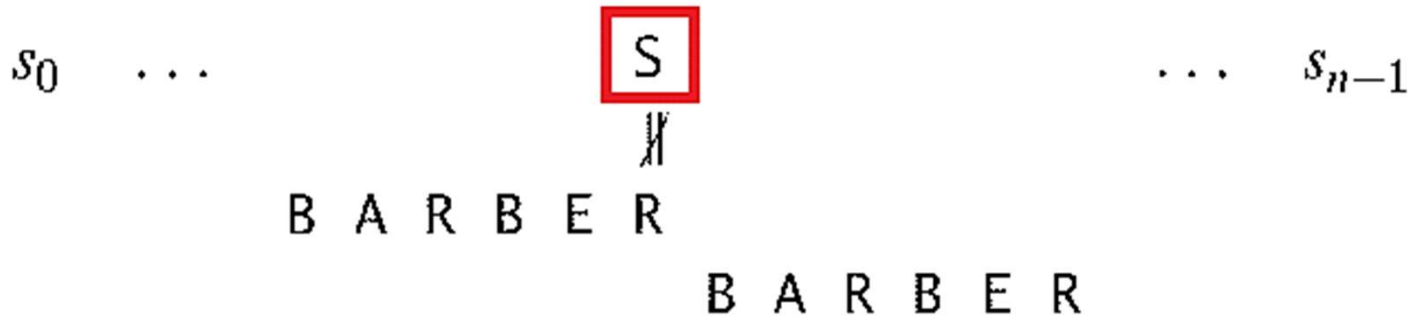
- Match the pattern right to left
- Construct bad symbol shift table
- Case 1 of Horspool's holds good in Boyer Moore
- But different strategy is used in the case where some positive number k($0 < k < m$) of the pattern's characters are matched successfully before a mismatch is encountered

Horspool's algorithm: Case 1
Boyer Moore algorithm

- If there are no letter say 'S' in the pattern, we can safely **shift the pattern by its entire length**

Example:

string is $s_0s_1 \ldots S \ldots s_{n-1}$ and pattern is "BARBER"



Boyer Moore algorithm:
Bad symbol shift (d_1) computation

$$d_1 = \max\{t_1(c) - k,\ 1\}$$

Where $t_1(c)$ is the entry in the shift table and k is the number of matched characters

Boyer Moore algorithm:
Good suffix shift (d_2) computation

find the longest prefix of size l < k that matches the suffix of the same size l

k	pattern	d_2
1	ABCB̄AB	2
2	ĀBCBAB	4
3	ĀBCBAB	4
4	ĀB̄CBAB	4
5	ĀB̄CBAB	4

good-suffix table

Boyer Moore algorithm:

Step 1: Construct bad-symbol shift table

Step 2: Construct good-suffix shift table

Step 3: Align the pattern against the beginning of the text.

Step 4:

Repeat until either a matching substring is found or pattern reaches beyond the last character of the text. Starting with the last character in the pattern, compare the corresponding characters in the pattern and the text until either all m character pairs are matched (then stop)

or a mismatching pair is encountered after k ≥ 0 character pairs are matched successfully. Retrieve the entry $t_1(c)$ from the c's column of the bad-symbol table where c is the text's mismatched character. If k > 0, also retrieve the corresponding d_2 entry from the good-suffix table. Shift the pattern to the right by the number of positions computed by the formula

$$d = \begin{cases} d_1 & \text{if } k = 0 \\ \max\{d_1, d_2\} & \text{if } k > 0 \end{cases},$$

$$\text{where } d_1 = \max\{t_1(c) - k,\ 1\}$$

Let’s check our understanding

Apply Boyer Moore algorithm

String: BESS_KNEW_ABOUT_BAOBABS

Pattern: BAOBAB

ALGORITHM *ShiftTable*$(P[0..m-1])$

//Fills the shift table used by Horspool's and Boyer-Moore algorithms
//Input: Pattern $P[0..m-1]$ and an alphabet of possible characters
//Output: *Table*$[0..size-1]$ indexed by the alphabet's characters and
// filled with shift sizes computed by formula initialize all the elements of *Table* with m

for $j \leftarrow 0$ **to** $m-2$ **do** $Table[P[j]] \leftarrow m-1-j$
return *Table*

Shift table for the pattern "BAOBAB"

c	A	B	C	D	. . .	O	. . .	Z	_
$t_1(c)$	1	2	6	6	6	3	6	6	6

good-suffix table

k	**pattern**	d_2
1	BAO$\overline{\text{B}}$AB	2
2	$\overline{\text{B}}$AOBAB	5
3	$\overline{\text{B}}$AOBAB	5
4	$\overline{\text{B}}$AOBAB	5
5	$\overline{\text{B}}$AOBAB	5

Pattern: "BAOBAB“
Text: “BESS_KNEW_ABOUT_BAOBABS”

B E S S _ K N E W _ A B O U T _ B A O B A B S
B A O B A B

$d_1 = t_1(K) - 0 = 6$

B A O B A B

$d_1 = t_1(\_) - 2 = 4$
$d_2 = 5$
$d = \max\{4, 5\} = 5$

B A O B A B

$d_1 = t_1(\_) - 1 = 5$
$d_2 = 2$
$d = \max\{5, 2\} = 5$

B A O B A B

Let’s check our understanding

Text: JIMY_HAILED_THE_LEADER_TO_STOP

Pattern: LEADER

- Apply Boyer – Moore algorithm
- Show the steps of the algorithm

Let’s check our understanding

String: G T A C T A G A G G A C G T A T G T A C T G

Pattern: A T G T A

- Apply Boyer – Moore’s algorithm

Let’s check our understanding

How many character comparisons will be made by Horspool's algorithm in searching for each of the following patterns in the binary text of 1000 zeros?

a. 00001
b. 10000
c. 01010

Let’s check our understanding

For searching in a text of length n for a pattern of length m (n >= m) with Horspool's algorithm, give an example of

a. worst-case input
b. best-case input

Let's check our understanding

How many character comparisons will the Boyer-Moore algorithm make in searching for each of the following patterns in the binary text of 1000 zeros?

a. 00001
b. 10000
c. 01010