An Adaptive Hybrid Pattern-Matching Algorithm on Indeterminate Strings

W. F. Smyth[1], Shu Wang[1] and Mao Yu[1]

[1] Algorithms Research Group
Department of Computing & Software
McMaster University, Canada
email: smyth,shuw@mcmaster.ca

The Prague Stringology Conference 2008

Outline

Outline

Outline

Outline

Outline

Outline

Outline

Regular Pattern Matching Algorithms

Over the last several decades, dozens of regular pattern-matching algorithms have been proposed.

- Window shifting: KMP [KMP77], BM [BM77], FJS [FJS06], etc.
- Bit-parallel: Shift-Or [Döm68, WM92, BYG92], BNDM [NR98], etc.

Regular Pattern Matching Algorithms

Over the last several decades, dozens of regular pattern-matching algorithms have been proposed.

- Window shifting: KMP [KMP77], BM [BM77], FJS [FJS06], etc.
- Bit-parallel: Shift-Or [Döm68, WM92, BYG92], BNDM [NR98], etc.

Indeterminate Pattern-Matching Algorithms

Intuitive approach: Modify existing regular pattern-matching algorithms to do indeterminate pattern-matching.

- Shift-Or: Can be modified to indeterminate pattern-matching easily, with the same speed of regular pattern-matching.
- iBMS: A very fast indeterminate pattern-matching algorithm based on BMS has been proposed in [HSW06b].
- iFJS: An indeterminate pattern-matching algorithm based on modified FJS (cut-off border array) has been proposed in [HSW06a].

Indeterminate Pattern-Matching Algorithms

Intuitive approach: Modify existing regular pattern-matching algorithms to do indeterminate pattern-matching.

- Shift-Or: Can be modified to indeterminate pattern-matching easily, with the same speed of regular pattern-matching.
- iBMS: A very fast indeterminate pattern-matching algorithm based on BMS has been proposed in [HSW06b].
- iFJS: An indeterminate pattern-matching algorithm based on modified FJS (cut-off border array) has been proposed in [HSW06a].

Indeterminate Pattern-Matching Algorithms

Intuitive approach: Modify existing regular pattern-matching algorithms to do indeterminate pattern-matching.

- Shift-Or: Can be modified to indeterminate pattern-matching easily, with the same speed of regular pattern-matching.
- iBMS: A very fast indeterminate pattern-matching algorithm based on BMS has been proposed in [HSW06b].
- iFJS: An indeterminate pattern-matching algorithm based on modified FJS (cut-off border array) has been proposed in [HSW06a].

Indeterminate String

A (regular) string x on Σ is a finite sequence of letters drawn from Σ. Two letters $\lambda, \mu \in \Sigma$ are said to *match* $(\lambda \approx \mu)$ iff $\lambda = \mu$. Consider any specified subset $S = \{\lambda_1, \lambda_2, \ldots, \lambda_j\}$ of Σ, $j \geq 2$. We introduce the idea of an indeterminate letter $\lambda = \lambda_S$ with the property that it matches every element of S (but no other letter); we write

$$\lambda \approx \lambda_1,\ \lambda \approx \lambda_2, \ldots, \lambda \approx \lambda_j.$$

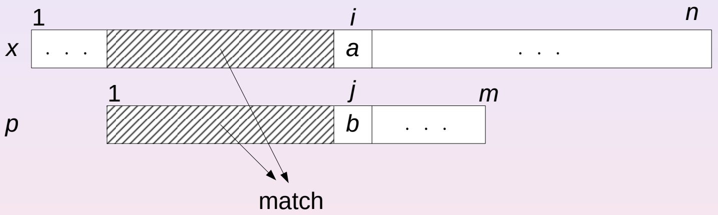
Given two subsets S, T of Σ, $|S| \geq 2$, $|T| \geq 2$, and indeterminate letters λ, μ associated with S, T respectively, $\lambda \approx \mu \Leftrightarrow S \cap T \neq \emptyset$. Given two indeterminate strings x and y, $x \approx y \Leftrightarrow (|x| = |y|) \wedge (\forall i \in 1..|x|, x[i] \approx y[i])$.

Outline

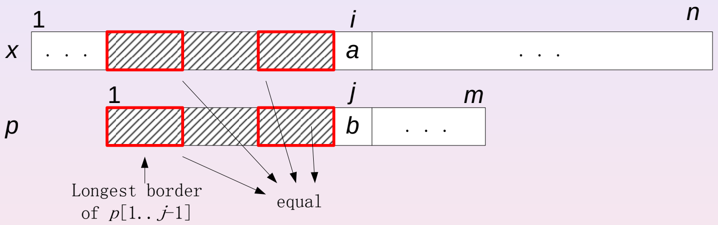
The Knuth-Morris-Pratt Algorithm

- A well-known linear time pattern-matching algorithm.
- Based on border array calculation.
- However, not very fast in practice.

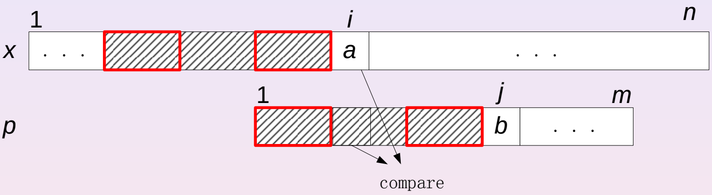
The KMP Algorithm - 1



The KMP Algorithm - 2



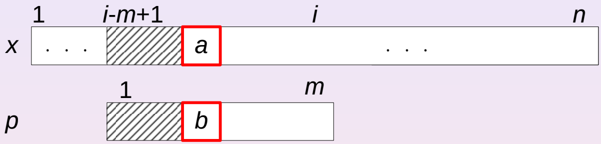
The KMP Algorithm - 3



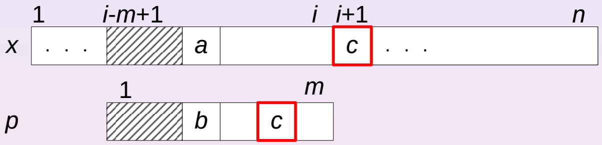
The Sunday Adaption of Boyer-Moore Algorithm

- A simplified version of the Boyer-Moore algorithms.
- Time complexity $O(mn)$.
- However, very fast in practice.

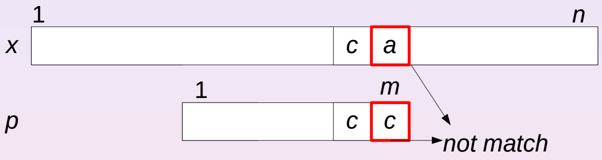
The BMS Algorithm - 1



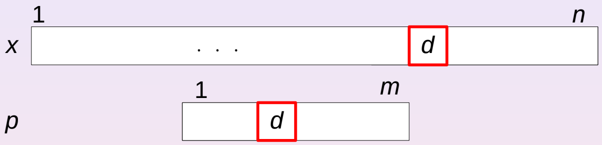
The BMS Algorithm - 2



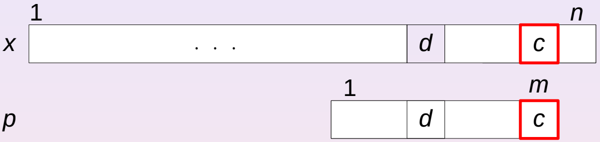
The BMS Algorithm - 3



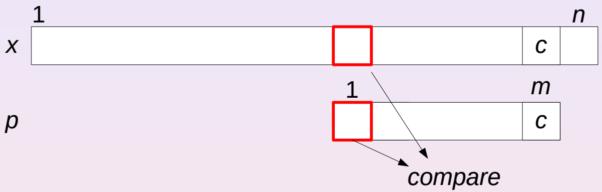
The BMS Algorithm - 4



The BMS Algorithm - 5



The BMS Algorithm - 6



The Shift-And Algorithm

- Makes use of the bit-parallel nature of computer.
- Time complexity $O(mn/w)$.
- Can be easily modified for indeterminate pattern-matching.

The Shift-And Algorithm

$m\backslash\Sigma$	A	C	G	T
A	1	0	0	0
A	1	0	0	0
T	0	0	0	1
C	0	1	0	0
G	0	0	1	0

Preprocessing();
BitArray: $D[1..n]$
$D[1] \leftarrow S_{x[1]}$
for $i = 2$ **to** n **do**
  $D[i] \leftarrow (\texttt{Shift}(D[i-1]) \& S_{x[i]})$;
  if $D_m \& 10^{m-1}$ **then** output $i - m + 1$;

The Franek-Jennings-Smyth Algorithm

- A hybrid algorithm that combines the KMP and BMS algorithm.
- Inherits the merits of both algorithms: very fast both asymptotically ($O(n)$) and in practice.

Outline of the FJS Algorithm

1. Perform **Sunday** shift along text.
2. When a match of letters is found at the end of the pattern, switch to **KMP** matching.
3. Continue **KMP** matching until no border can be used, then switch back to **Sunday** shift.

Outline

Example of Non-transitivity Effect

Suppose we are performing KMP matching along the text.

Index		1	2	3	4	5	6	7	
x	......	a	a	b	b	a	b	b	......
p		a	$*$	$*$	b	a	$*$	a	
$1st$ Shift				a	$*$	$*$	b	a	......
$2nd$ Shift						a	$*$	$*$	......
$3rd$ Shift							a	$*$	......

Table: First example of the non-transitivity effect

Proposition

Shifting the pattern to the right according to the longest border cannot guarantee a prefix match.

Proposition

A border of a border of x is not necessarily a border of x.

Index		1	2	3	4	5	6	7	
x	......	a	b	a	$*$	a	$*$	a	......
p		a	b	a	a	a	b	b	
Wrong Shift						a	b	a	......
Correct Shift				a	b	a	a	a	......

Table: Second example of the non-transitivity effect

Proposition

Shifting the pattern to the right according to the longest border can miss some occurrences in between.

Impact

- Because of these properties, transforming regular pattern-matching algorithms that use border arrays into indeterminate pattern-matching algorithms is non-trivial (KMP, FJS, etc.)
- However, since some of these regular algorithms are very fast in practice and have nice properties, we are motivated to invent indeterminate versions of them that avoid using border arrays.

Outline

A New Hybrid Algorithm

We propose a new hybrid algorithm that uses Shift-And and BMS as complementary shift engines.

- 1. Perform **Sunday shift** along text.
- 2. When a match of letters is found at the end of the pattern, switch to **Shift-And matching**.
- 3. Continue **Shift-And matching** until no match can be found at the current position ($D = 0$), then skip to next possible position and switch back to **Sunday** shift.

Shift-And Preprocessing

The usual Shift-And preprocessing is modified as follows:

for $i = 1$ **to** m
  for $j = 1$ **to** $|\Sigma|$
    if $\text{MATCH}(p[i], \Sigma[j])$ **then** $S[i, j] = 1$
    else $S[i, j] = 0$

Properties of Shift-Or

Notice some of the important properties of Shift-Or.

Proposition

$D[j] = 1 \Leftrightarrow p[1..j] \approx x[i - j + 1..i]$

Proposition

$D = 0$ *if and only if there doesn't exist any* $j \in 1..m$ *such that* $p[1..j] \approx x[i - j + 1..i]$

These properties enables us to move the pattern beyond $x[i]$ when we finish `ShiftAnd-Match`.

ShiftAnd-MATCH

$D \leftarrow 0$
do
  $D \leftarrow (D \ll 1) \& S_{x[i]}$
  if $D \& 10^m \neq 0$ **then output** i
  $i \leftarrow i + 1$
//If D =0, terminate loop according to previous proposition
while $(i \leq n$ **and** $D \neq 0)$

BMS Preprocessing

The usual BMS preprocessing is modified as follows.

for $i = 1$ **to** $|\Delta|$
  $\Delta[i] = m + 1$
for $i = 1$ **to** m
    for $j = 1$ **to** $|\Sigma|$
      if $\mathrm{MATCH}(p[i], \Sigma[j])$ **then** $\Delta[p[i]] = i$

Sunday-Shift

while not $\text{MATCH}(p[m], x[i'])$ **do**
  $i' \leftarrow i' + \Delta\left[x[i' + 1]\right]$
  if $i' > n$ **then return**

Algorithm Shift-And/Sunday

$i' \leftarrow m$; $m' \leftarrow m - 1$;
while $i' \leq n$ **do**
  `Sunday-Shift`();
  $i \leftarrow i' - m'$;
  //After Sunday-Shift stops, perform `ShiftAnd-MATCH`
  `ShiftAnd-Match`();
  //After ShiftAnd-Match stops, shift pattern to the right
  $i' \leftarrow i + m'$;

Example of The New Hybrid Algorithm

x	...	a	a	b	b	c	b	b	...
p		a	*	b	b				
D		0	0	0	0				
$1 \gg D$		1	0	0	0				
Sa		1	1	0	0				
D'		1	0	0	0				

Shift-Or
Match

Example of The New Hybrid Algorithm

x	...	a	a	b	b	c	b	b	...
p		a	$*$	b	b				
D		1	0	0	0				
$1 >> D$		1	1	0	0				
Sa		1	1	0	0				
D'		1	1	0	0				

Example of The New Hybrid Algorithm

x	...	a	a	b ↓	b	c	b	b	...
p		a	*	b	b				
D		1	1	0	0				
$1 \gg D$		1	1	1	0				
S_b		0	1	1	1				
D'		0	1	1	0				

Example of The New Hybrid Algorithm

x	. . .	a	a	b	b	c	b	b	. . .
p		a	$*$	b	b				
D		0	1	1	0				
$1 \gg D$		1	0	1	1				
S_b		0	1	1	1				
D'		0	0	1	1				

Example of The New Hybrid Algorithm

x	...	a	a	b	b	c	b	b	...
p		a	*	b	b				
D		0	0	1	1				
$1 \gg D$		1	0	0	1				
Sc		0	1	0	0				
D'		0	0	0	0				

Example of The New Hybrid Algorithm

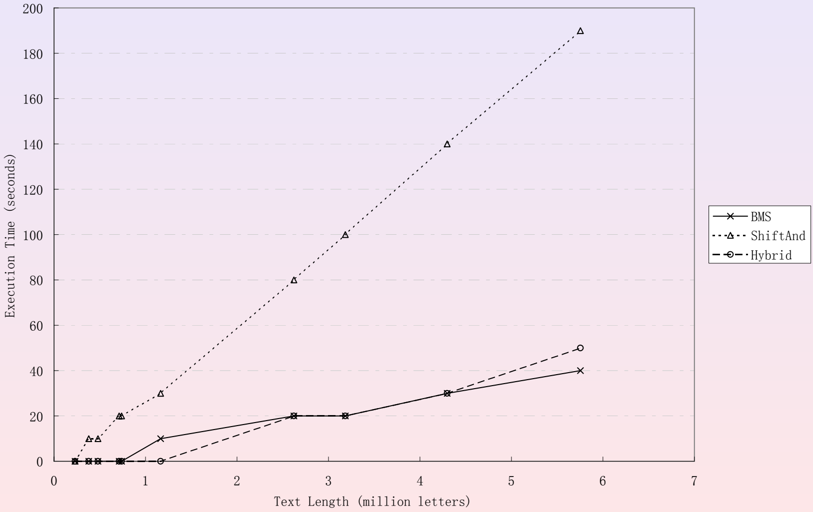
										↓		
x	...	**a**	a	b	**b**	c	b	b	b	b	a	...
p							**a**	*	b	**b**		

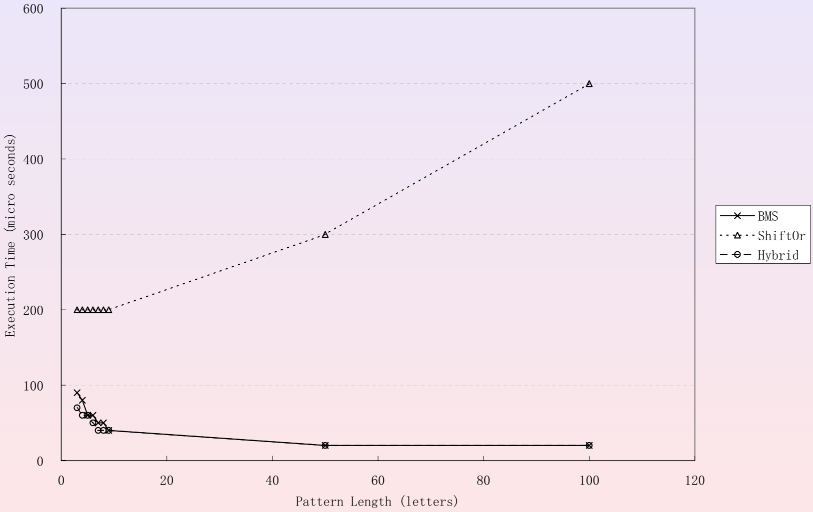
Begin *Sunday-Shift*

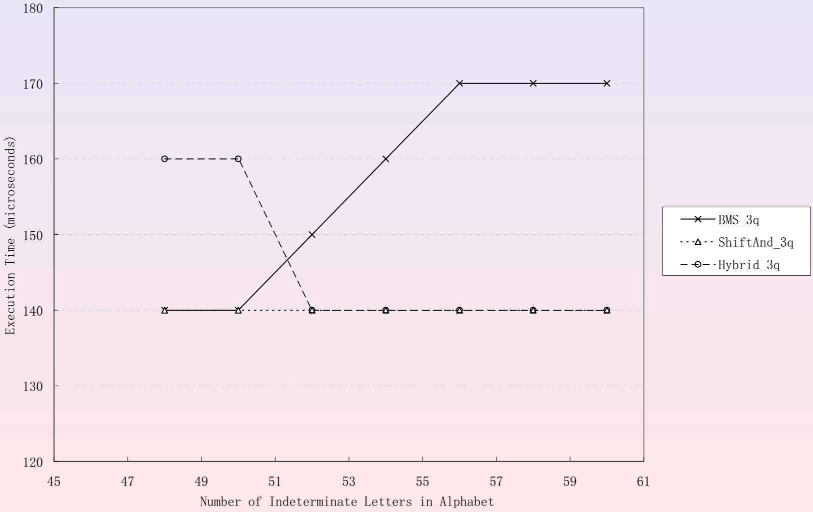
Example of The New Hybrid Algorithm

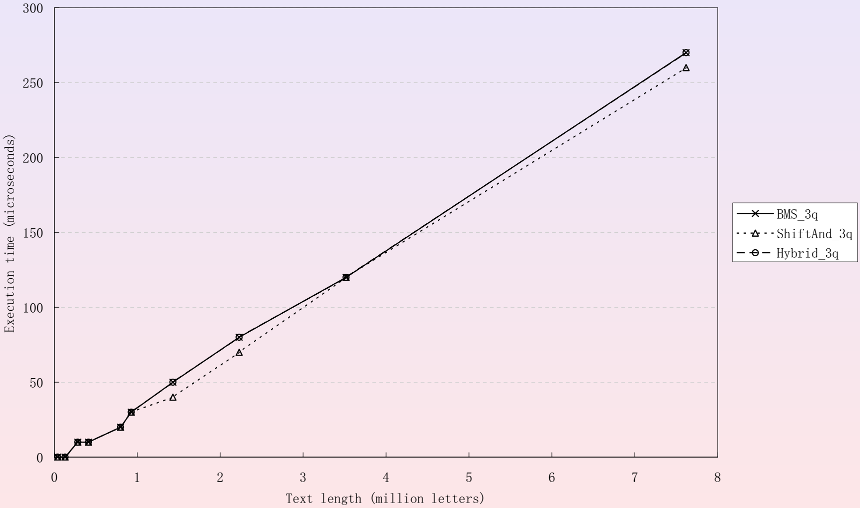
x	...	a	a	b	b	c	b	b	b	b	a	...	
p										a	*	b	b

Outline


200
180
160
140
120
100
80
60
40
20
0
Execution Time (seconds)
0
1
2
3
4
5
6
7
Text Length (million letters)
BMS
ShiftAnd
Hybrid


600
500
400
300
200
100
0
Execution Time (micro seconds)
0
20
40
60
80
100
120
Pattern Length (letters)
BMS
ShiftOr
Hybrid


180
170
160
150
140
130
120
Execution Time (microseconds)
45
47
49
51
53
55
57
59
61
Number of Indeterminate Letters in Alphabet
BMS_3q
ShiftAnd_3q
Hybrid_3q


300
250
200
150
100
50
0
Execution time (microseconds)
0
1
2
3
4
5
6
7
8
Text length (million letters)
BMS_3q
ShiftAnd_3q
Hybrid_3q

- In all of these tests, the hybrid algorithm's behaviour is very close to that of the better of BMS and Shift-And.
- The new algorithms's total running time is very competitive among these three algorithms being tested.

Outline

- A new algorithm that performs fast pattern-matching on both regular and indeterminate strings.
- Strong ability to adapt to the nature of text/pattern and to achieve faster performance over cases that arise in practice. This dynamic adaptivity is useful when we do not know the type of text or pattern: we don't need to make a decision ahead of time about which algorithm to use.
- Future work: Indeterminate pattern-matching algorithms based on variants of Shift-And such as BNDM and [Fre07], as well as on new convolution techniques [AAR07].

- Avivit Levy Amihood Amir and Liron Reuveni.
  The practical efficiency of convolutions in pattern matching.
  Fundamenta Informatica, page to appear, 2007.
- Robert S. Boyer and J. S Strother Moore.
  A fast string searching algorithm.
  CACM, 20(10):762–772, 1977.
- R.A. Baeza-Yates and G.H. Gonnet.
  A new approach to text searching.
  Communications of the ACM, 35(10):74–82, 1992.
- Bálint Dömölki.
  A universal computer system based on production rules.
  BIT, 8:262–275, 1968.

- Frantisek Franek, Christopher G. Jennings, and W. F. Smyth.
  A simple fast hybrid pattern-matching algorithm.
  Journal of Discrete Algorithms, to appear, 2006.
- Kimmo Fredriksson.
  Linear worst case time bndm.
  Information Processing Letters, page to appear, 2007.
- Jan Holub, W. F. Smyth, and Shu Wang.
  Hybrid pattern-matching algorithms on indeterminate strings.
  London Algorithmics and Stringology 2006, J. Daykin, M. Mohamed and K. Steinhoefel (eds.), King's College London Series Texts in Algorithmics, pages 115–133, 2006.

- Jan Holub, W. F. Smyth, and Shu Wang.
  Fast pattern-matching on indeterminate strings.
  Journal of Discrete Algorithms, to appear, 2006.
- D. E. Knuth, J. H. Morris, and V.R. Pratt.
  Fast pattern matching in strings.
  SIAM Journal on Computing, 6(2):323–350, 1977.
- G. Navarro and M. Raffinot.
  A bit-parallel approach to suffix automata: Fast extended string matching.
  In M. Farach-Colton, editor, *Proceedings of the 9th Annual Symposium on Combinatorial Pattern Matching*, number 1448, pages 14–33, Piscataway, NJ, 1998. Springer-Verlag, Berlin.

S. Wu and U. Manber.
Fast text searching with errors.
Communications of the ACM, 35(10):83–91, 1992.