Efficient pattern matching in degenerate strings with the Burrows–Wheeler transform

Jacqueline W. Daykin [1,2,3] Richard Groult [4,3] Yannick Guesnet [3]
Thierry Lecroq [3] Arnaud Lefebvre [3] Martine Léonard [3] Laurent Mouchard [3] Élise Prieur-Gaston [3] Bruce Watson [5,6]

[1] Aberystwyth Univ. (Mauritius Branch Campus), Mauritius
[2] King's College London, UK
[3] Normandie Univ, UNIROUEN, LITIS, 76000 Rouen, France
[4] Univ. de Picardie Jules Verne, Amiens, France
[5] Stellenbosch Univ., South Africa
[6] CAIR, CSIR Meraka, Pretoria, South Africa

Seqbio 2017, Lille, France, November 7th, 2017

Outline

1. Introduction
2. Pattern matching in degenerate strings with the BWT
3. Experiments

Outline

1. Introduction
2. Pattern matching in degenerate strings with the BWT
3. Experiments

Burrow-Wheeler Transform (BWT)

Definition

Let x be a string built on a finite alphabet Σ.
The BWT of x is defined as the pair (L, h) where L is the last column of the matrix M_x formed by all the sorted cyclic rotations of x and h is the index of x in this matrix.

Burrow-Wheeler Transform (BWT)

Definition

Let x be a string built on a finite alphabet Σ.
The BWT of x is defined as the pair (L, h) where L is the last column of the matrix M_x formed by all the sorted cyclic rotations of x and h is the index of x in this matrix.

$x = \texttt{BANANA}$

1	1	B	A	N	A	N	A
2	2	A	N	A	N	A	B
3	3	N	A	N	A	B	A
4	4	A	N	A	B	A	N
5	5	N	A	B	A	N	A
6	6	A	B	A	N	A	N

Burrow-Wheeler Transform (BWT)

Definition

Let x be a string built on a finite alphabet Σ.
The BWT of x is defined as the pair (L, h) where L is the last column of the matrix M_x formed by all the sorted cyclic rotations of x and h is the index of x in this matrix.

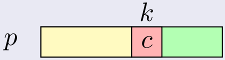
$x = \mathtt{BANANA}$

1	1	B	A	N	A	N	A
2	2	A	N	A	N	A	B
3	3	N	A	N	A	B	A
4	4	A	N	A	B	A	N
5	5	N	A	B	A	N	A
6	6	A	B	A	N	A	N

	SA						BWT
1	6	A	B	A	N	A	N
2	4	A	N	A	B	A	N
3	2	A	N	A	N	A	B
4	1	B	A	N	A	N	A
5	5	N	A	B	A	N	A
6	3	N	A	N	A	B	A

$BWT(\mathtt{BANANA}) = (\mathtt{NNBAAA}, 4)$

Backward search



Assume

(i, j) is the interval in the SA of a text t of the suffixes of t starting with $p[k+1\,..\,m]$

then

(i', j') is the interval in the SA of t of the suffixes of t starting with $p[k\,..\,m]$

with

$$i' = C[c] + \mathit{rank}_c(\mathit{BWT}, i-1) + 1 \text{ and } j' = C[c] + \mathit{rank}_c(\mathit{BWT}, j)$$

where

$c = p[k]$, $C[c] = \sharp\{i \mid t[i] < c\}$ and
$\mathit{rank}_c(x, i)$ gives the number of occurrences of the letter c in the prefix $x[1\,..\,i]$.

Backward search – example

$\rightarrow$	1	6	A	B	A	N	A	N
	2	4	A	N	A	B	A	N
	3	2	A	N	A	N	A	B
	4	1	B	A	N	A	N	A
	5	5	N	A	B	A	N	A
$\rightarrow$	6	3	N	A	N	A	B	A

$(1, 6)$ is the interval in the SA of BANANA of suffixes starting with ε

Backward search – example

$\rightarrow$	1	6	A	B	A	N	A	N
	2	4	A	N	A	B	A	N
	3	2	A	N	A	N	A	B
	4	1	B	A	N	A	N	A
	5	5	N	A	B	A	N	A
$\rightarrow$	6	3	N	A	N	A	B	A

$(1, 6)$ is the interval in the SA of BANANA of suffixes starting with ε

$\rightarrow$	1	6	A	B	A	N	A	N
	2	4	A	N	A	B	A	N
$\rightarrow$	3	2	A	N	A	N	A	B
	4	1	B	A	N	A	N	A
	5	5	N	A	B	A	N	A
	6	3	N	A	N	A	B	A

$(6, 6)$ is the interval in the SA of BANANA of suffixes starting with A

Backward search – example

	1	6	A	B	A	N	A	N
$\rightarrow$	2	4	A	N	A	B	A	N
$\rightarrow$	3	2	A	N	A	N	A	B
	4	1	B	A	N	A	N	A
	5	5	N	A	B	A	N	A
	6	3	N	A	N	A	B	A

$(2, 3)$ is the interval in the SA of BANANA of suffixes starting with AN

Backward search – example

	1	6	A	B	A	N	A	N
$\rightarrow$	2	4	A	N	A	B	A	N
$\rightarrow$	3	2	A	N	A	N	A	B
	4	1	B	A	N	A	N	A
	5	5	N	A	B	A	N	A
	6	3	N	A	N	A	B	A

$(2,3)$ is the interval in the SA of BANANA of suffixes starting with AN

	1	6	A	B	A	N	A	N
	2	4	A	N	A	B	A	N
	3	2	A	N	A	N	A	B
	4	1	B	A	N	A	N	A
	5	5	N	A	B	A	N	A
$\Rightarrow$	6	3	N	A	N	A	B	A

$(6,6)$ is the interval in the SA of BANANA of suffixes starting with NAN

Degenerate strings

Definition

Given an alphabet Σ we define a new alphabet Δ_Σ as the non-empty subsets of Σ:
$\Delta_\Sigma = \mathcal{P}(\Sigma) \setminus \{\emptyset\}$
Singletons are called *solid* letters.
Degenerate or *indeterminate* strings on an alphabet Σ are strings of Δ_Σ.

$\Sigma = \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \mathtt{d}, \mathtt{e}\}$
$t = \{\mathtt{c}, \mathtt{e}\} \cdot \{\mathtt{c}, \mathtt{d}\} \cdot \{\mathtt{a}, \mathtt{b}, \mathtt{c}\} \cdot \{\mathtt{a}, \mathtt{d}\} \cdot \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$

Pattern matching on degenerate strings

Definition

Given 2 degenerate strings p and t find all the positions $0 \leq j < |t| - |p|$ on t where $p[i] \cap t[i+j] \neq \emptyset$ for $0 \leq i < |p|$.

$p = \{\mathtt{a}\} \cdot \{\mathtt{c}, \mathtt{d}\}$ occurs at positions 3 and 4 in
$t = \{\mathtt{c}, \mathtt{e}\} \cdot \{\mathtt{c}, \mathtt{d}\} \cdot \{\mathtt{a}, \mathtt{b}, \mathtt{c}\} \cdot \{\mathtt{a}, \mathtt{d}\} \cdot \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$

Pattern matching on degenerate strings

Definition

Given 2 degenerate strings p and t find all the positions $0 \leq j < |t| - |p|$ on t where $p[i] \cap t[i+j] \neq \emptyset$ for $0 \leq i < |p|$.

$p = \{\mathtt{a}\} \cdot \{\mathtt{c}, \mathtt{d}\}$ occurs at positions 3 and 4 in
$t = \{\mathtt{c}, \mathtt{e}\} \cdot \{\mathtt{c}, \mathtt{d}\} \cdot \{\mathtt{a}, \mathtt{b}, \mathtt{c}\} \cdot \{\mathtt{a}, \mathtt{d}\} \cdot \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$

Pattern matching on degenerate strings

Definition

Given 2 degenerate strings p and t find all the positions $0 \leq j < |t| - |p|$ on t where $p[i] \cap t[i+j] \neq \emptyset$ for $0 \leq i < |p|$.

$p = \{\mathtt{a}\} \cdot \{\mathtt{c}, \mathtt{d}\}$ occurs at positions 3 and 4 in
$t = \{\mathtt{c}, \mathtt{e}\} \cdot \{\mathtt{c}, \mathtt{d}\} \cdot \{\mathtt{a}, \mathtt{b}, \mathtt{c}\} \cdot \{\mathtt{a}, \mathtt{d}\} \cdot \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$

Burrows-Wheeler transform on degenerate strings (D-BWT)

Given an order on Δ_Σ denoted by the usual symbol $<$, we can compute the BWT of a degenerate string x in the same way as for a regular string.

$t = \{\mathtt{c}, \mathtt{e}\} \cdot \{\mathtt{c}, \mathtt{d}\} \cdot \{\mathtt{a}, \mathtt{b}, \mathtt{c}\} \cdot \{\mathtt{a}, \mathtt{d}\} \cdot \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$

$\mathtt{A} = \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}, \mathtt{B} = \{\mathtt{a}, \mathtt{d}\}, \mathtt{C} = \{\mathtt{c}, \mathtt{d}\}, \mathtt{D} = \{\mathtt{c}, \mathtt{e}\}$

$\mathtt{A} < \mathtt{B} < \mathtt{C} < \mathtt{D}$

1	1	D	C	A	B	A
2	2	A	D	C	A	B
3	3	B	A	D	C	A
4	4	A	B	A	D	C
5	5	C	A	B	A	D

1	3	A	B	A	D	B
2	1	A	D	C	A	C
3	4	B	A	D	A	D
4	2	C	A	B	B	A
5	5	D	C	A	C	A

Burrows-Wheeler transform on degenerate strings (D-BWT)

Jacqueline W. Daykin and Bruce Watson
A Text Transformation Scheme for Degenerate Strings
Proceedings of the 2nd International Conference on Algorithms for Big Data, Palermo, Italy, April 07-09, 2014 pp 23–29

Jacqueline W. Daykin and Bruce Watson
Indeterminate String Factorizations and Degenerate Text Transformations
Mathematics in Computer Science **11**(2) (2017) 209–218

BWBBLE

Lin Huang, Victoria Popic and Serafim Batzoglou
Short read alignment with populations of genomes
Bioinformatics **29**(13) (2013) i361–i370

Represent a collection of genomes called *reference multi-genome* and do pattern matching

BWBBLE

SNPs (aka SNVs or substitutions)

4-letter alphabet $\{\mathtt{A}, \mathtt{C}, \mathtt{G}, \mathtt{T}\}$ $\rightarrow$ 16-letter IUPAC encoding $\rightarrow$ 4-bit Gray code (to minimize $\sharp$ separate intervals during the search with the BWT)

Indels (insertions-deletions)

Corresponding sequences padded with surrounding bases (length depending on read length) are concatenated at the end of the reference multi-genome (separated by a special character)

Inversions, translocations and duplications

only both ends of the events are concatenated at the end of the reference multi-genome

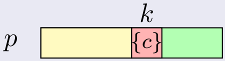
Outline

1 Introduction

2 Pattern matching in degenerate strings with the BWT

3 Experiments

Backward search on the D-BWT: formalization



$$
\begin{aligned}
&\mathit{OneStep}(H, k, C, \mathit{BWT} = (L, h), p) = \\
&(((r, s)) \quad | \quad r = C[c] + \mathit{rank}_c(L, i-1) + 1, \\
&\qquad s = C[c] + \mathit{rank}_c(L, j), \\
&\qquad r \le s,\ (i, j) \in H,\ c \in \Delta_\Sigma \text{ and } c \cap p[k] \ne \emptyset).
\end{aligned}
$$

Let $\mathit{Step}(m, C, \mathit{BWT}, p) = \mathit{OneStep}(\{(1, n)\}, m, C, \mathit{BWT}, p)$ and $\mathit{Step}(i, C, \mathit{BWT}, p) = \mathit{OneStep}(\mathit{Step}(i+1, C, \mathit{BWT}, p), i, C, \mathit{BWT}, p)$ for $1 \le i \le m-1$.

In words, $\mathit{Step}(i, C, \mathit{BWT}, p)$ applies step m through to i of the backward search.
Then $\mathit{Step}(1, C, \mathit{BWT}, p)$ contains the intervals in the SA of t of the suffixes of t starting with p.

Backward search on the D-BWT: correctness

Lemma

The interval $(i, j) \in \mathit{Step}(k, C, \mathit{BWT}, p)$ *if and only if* $p[k..m]$ *is a degenerate prefix of* $M_t[h]$ *for* $i \leq h \leq j$.

Corollary

The interval $(i, j) \in \mathit{Step}(1, C, \mathit{BWT}, p)$ *if and only if* p *is a degenerate prefix of* $M_t[h]$ *for* $i \leq h \leq j$.

Intervals do not overlap

Lemma

The intervals in OneStep$(\{(i, j)\}, k, C, \mathit{BWT}, p)$ *do not overlap.*

Intervals do not overlap

$t = \{c, e\} \cdot \{c, d\} \cdot \{a, b, c\} \cdot \{a, d\} \cdot \{a, b, c\}$
$A = \{a, b, c\}, B = \{a, d\}, C = \{c, d\}, D = \{c, e\}$
$A < B < C < D$
$p = \{c\} \cdot \{d\} \cdot \{b\}$
$b \in A$ and $d \in B, C$

$\rightarrow$	1	3	A	B	A	D	C
$\rightarrow$	2	5	A	D	C	A	B
	3	4	B	A	D	C	A
	4	2	C	A	B	A	D
	5	1	D	C	A	B	A

Intervals do not overlap

$t = \{\mathtt{c}, \mathtt{e}\} \cdot \{\mathtt{c}, \mathtt{d}\} \cdot \{\mathtt{a}, \mathtt{b}, \mathtt{c}\} \cdot \{\mathtt{a}, \mathtt{d}\} \cdot \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$

$\mathtt{A} = \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}, \mathtt{B} = \{\mathtt{a}, \mathtt{d}\}, \mathtt{C} = \{\mathtt{c}, \mathtt{d}\}, \mathtt{D} = \{\mathtt{c}, \mathtt{e}\}$

$\mathtt{A} < \mathtt{B} < \mathtt{C} < \mathtt{D}$

$p = \{\mathtt{c}\} \cdot \{\mathtt{d}\} \cdot \{\mathtt{b}\}$

$\mathtt{b} \in \mathtt{A}$ and $\mathtt{d} \in \mathtt{B}, \mathtt{C}$

	1	3	A	B	A	D	C
	2	5	A	D	C	A	B
⇒	3	4	B	A	D	C	A
⇒	4	2	C	A	B	A	D
	5	1	D	C	A	B	A

Intervals do not overlap

Lemma

The intervals in OneStep$(\{(i, j), (i', j')\}, k, C, BWT, p)$ *with* $i \leq j < i' \leq j'$ *do not overlap.*

Corollary

Let H *be a set of non-overlapping intervals. The intervals in OneStep*(H, k, C, BWT, p) *do not overlap.*

Intervals do not overlap

$t = \{\mathtt{c}, \mathtt{e}\} \cdot \{\mathtt{c}, \mathtt{d}\} \cdot \{\mathtt{a}, \mathtt{b}, \mathtt{c}\} \cdot \{\mathtt{a}, \mathtt{d}\} \cdot \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$

$\mathtt{A} = \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}, \mathtt{B} = \{\mathtt{a}, \mathtt{d}\}, \mathtt{C} = \{\mathtt{c}, \mathtt{d}\}, \mathtt{D} = \{\mathtt{c}, \mathtt{e}\}$

$\mathtt{A} < \mathtt{B} < \mathtt{C} < \mathtt{D}$

$p = \{\mathtt{c}\} \cdot \{\mathtt{d}\} \cdot \{\mathtt{b}\}$

$\mathtt{b} \in \mathtt{A}$, $\mathtt{d} \in \mathtt{B}, \mathtt{C}$ and $\mathtt{c} \in \mathtt{A}, \mathtt{D}$

	1	3	A	B	A	D	C
	2	5	A	D	C	A	B
⇒	3	4	B	A	D	C	A
⇒	4	2	C	A	B	A	D
	5	1	D	C	A	B	A

Intervals do not overlap

$t = \{\mathtt{c},\mathtt{e}\} \cdot \{\mathtt{c},\mathtt{d}\} \cdot \{\mathtt{a},\mathtt{b},\mathtt{c}\} \cdot \{\mathtt{a},\mathtt{d}\} \cdot \{\mathtt{a},\mathtt{b},\mathtt{c}\}$

$\mathtt{A} = \{\mathtt{a},\mathtt{b},\mathtt{c}\}, \mathtt{B} = \{\mathtt{a},\mathtt{d}\}, \mathtt{C} = \{\mathtt{c},\mathtt{d}\}, \mathtt{D} = \{\mathtt{c},\mathtt{e}\}$

$\mathtt{A} < \mathtt{B} < \mathtt{C} < \mathtt{D}$

$p = \{\mathtt{c}\} \cdot \{\mathtt{d}\} \cdot \{\mathtt{b}\}$

$\mathtt{b} \in \mathtt{A}$, $\mathtt{d} \in \mathtt{B},\mathtt{C}$ and $\mathtt{c} \in \mathtt{A},\mathtt{D}$

$\Rightarrow$	1	3	A	B	A	D	C
	2	5	A	D	C	A	B
	3	4	B	A	D	C	A
	4	2	C	A	B	A	D
$\Rightarrow$	5	1	D	C	A	B	A

Consecutive intervals can merge

Lemma

$\textit{Merge}(\textit{OneStep}(((i,j),(j+1,j')),k,C,\textit{BWT},p)) = \textit{OneStep}(((i,j')),k,C,\textit{BWT},p).$

Consecutive intervals can merge

$t = \{c, e\} \cdot \{c, d\} \cdot \{a, b, c\} \cdot \{a, d\} \cdot \{a, b, c\}$
$A = \{a, b, c\}, B = \{a, d\}, C = \{c, d\}, D = \{c, e\}$
$A < B < C < D$
$p = \{c\} \cdot \{d\} \cdot \{b\}$
$b \in A$, $d \in B, C$ and $c \in A, D$

	1	3	A	B	A	D	C
	2	5	A	D	C	A	B
⇒	3	4	B	A	D	C	A
⇒	4	2	C	A	B	A	D
	5	1	D	C	A	B	A

Consecutive intervals can merge

$t = \{\mathtt{c},\mathtt{e}\} \cdot \{\mathtt{c},\mathtt{d}\} \cdot \{\mathtt{a},\mathtt{b},\mathtt{c}\} \cdot \{\mathtt{a},\mathtt{d}\} \cdot \{\mathtt{a},\mathtt{b},\mathtt{c}\}$

$\mathtt{A} = \{\mathtt{a},\mathtt{b},\mathtt{c}\}, \mathtt{B} = \{\mathtt{a},\mathtt{d}\}, \mathtt{C} = \{\mathtt{c},\mathtt{d}\}, \mathtt{D} = \{\mathtt{c},\mathtt{e}\}$

$\mathtt{A} < \mathtt{B} < \mathtt{C} < \mathtt{D}$

$p = \{\mathtt{c}\} \cdot \{\mathtt{d}\} \cdot \{\mathtt{b}\}$

$\mathtt{b} \in \mathtt{A}$, $\mathtt{d} \in \mathtt{B}, \mathtt{C}$ and $\mathtt{c} \in \mathtt{A}, \mathtt{D}$

	1	3	A	B	A	D	C
	2	5	A	D	C	A	B
$\rightarrow$	3	4	B	A	D	C	A
$\rightarrow$	4	2	C	A	B	A	D
	5	1	D	C	A	B	A

Consecutive intervals can merge

$t = \{\mathsf{c},\mathsf{e}\} \cdot \{\mathsf{c},\mathsf{d}\} \cdot \{\mathsf{a},\mathsf{b},\mathsf{c}\} \cdot \{\mathsf{a},\mathsf{d}\} \cdot \{\mathsf{a},\mathsf{b},\mathsf{c}\}$

$\mathtt{A} = \{\mathsf{a},\mathsf{b},\mathsf{c}\}, \mathtt{B} = \{\mathsf{a},\mathsf{d}\}, \mathtt{C} = \{\mathsf{c},\mathsf{d}\}, \mathtt{D} = \{\mathsf{c},\mathsf{e}\}$

$\mathtt{A} < \mathtt{B} < \mathtt{C} < \mathtt{D}$

$\mathsf{b} \in \mathtt{A}$, $\mathsf{d} \in \mathtt{B},\mathtt{C}$ and $\mathsf{c} \in \mathtt{A},\mathtt{D}$

$\Rightarrow$	1	3	A	B	A	D	C
	2	5	A	D	C	A	B
	3	4	B	A	D	C	A
	4	2	C	A	B	A	D
$\Rightarrow$	5	1	D	C	A	B	A

Pattern matching in conservative degenerate strings

A degenerate string is said to be *conservative* if its number of non-solid letters is upper-bounded by a fixed positive constant q.

Theorem

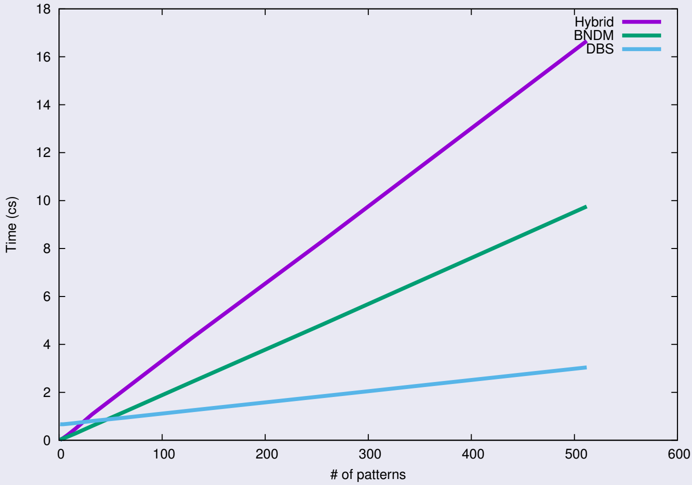
Let t be a conservative degenerate string over a constant size alphabet. Let the number of degenerate letters of t be bounded by a constant q. Then given the BWT of t, all the intervals in the BWT of occurrences of a pattern p of length m can be detected in time $O(qm^2)$.

Outline

1. Introduction
2. Pattern matching in degenerate strings with the BWT
3. Experiments

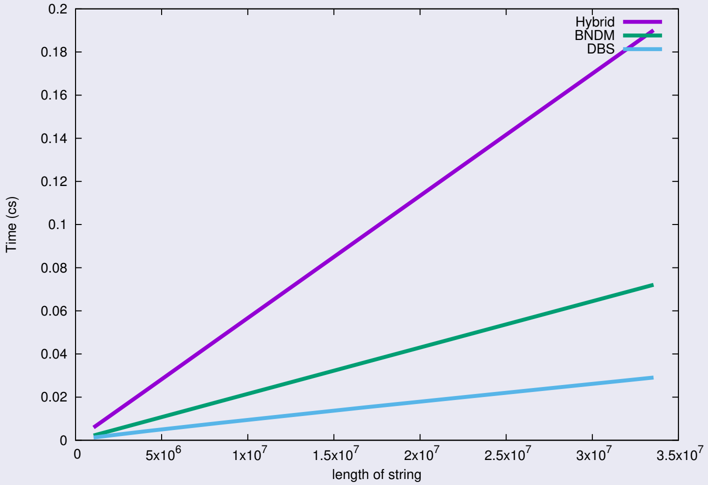
Experiments

Degenerate patterns of length 8 in a solid string of length 5MB, $\sigma = 4$



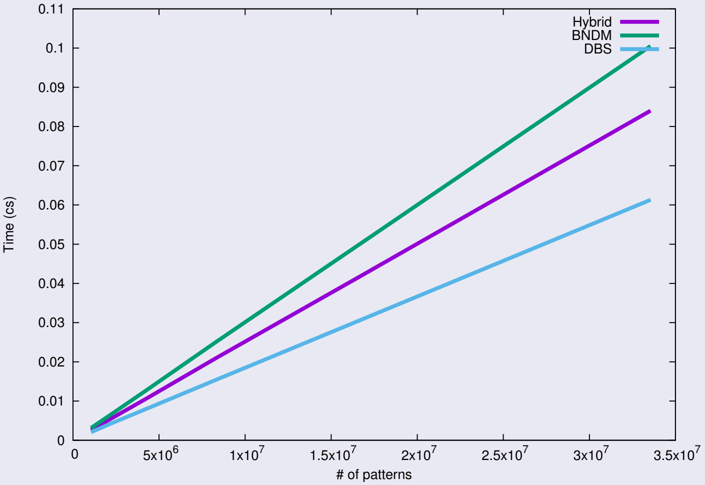
Experiments

One degenerate pattern of length 8 in a conservative degenerate string



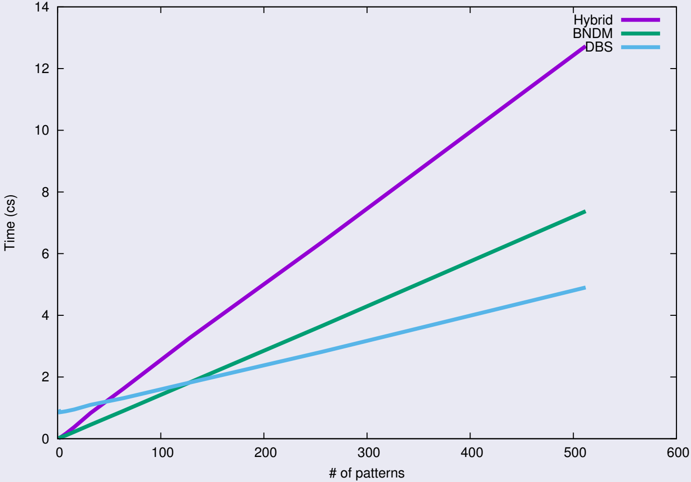
Experiments

Degenerate patterns of length 8 in a solid string, $\sigma = 4$



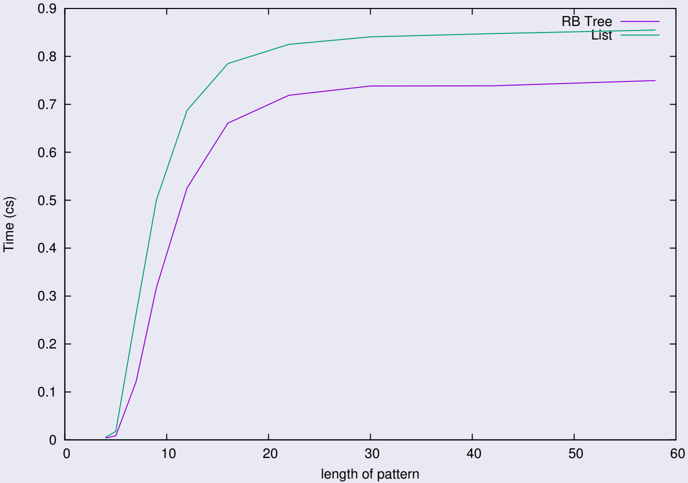
Experiments

Degenerate patterns of length 8 in a solid string, $\sigma = 8$



Experiments

Intervals



Perspectives

- Average case analysis
- Efficient data structure for handling intervals
- Using different order on the alphabet
- ...

Thank you for your attention!