RLBWT Tricks

Nathaniel K. Brown[1] Travis Gagie[1] Massimiliano Rossi[2]

[1]Faculty of Computer Science
Dalhousie University
Halifax, NS, Canada

[2]Department of Computer and Information Science and Engineering
University of Florida
Gainesville, FL, USA

Data Compression Conference, March 2022

Table of Contents

Introduction

- **String Indexing**: Support sub-string queries on text
- **FM-Index**: basis for key tools in computational genomics
  - Short read aligners such as BWA and Bowtie
  - Application of Burrows-Wheeler Transform (BWT)
- **Computational Pan-Genomics**:
  - Want to index many genomes in reasonable space
  - *Solution*: Versions of FM-Index based on run-length compressed BWT (RLBWT)

Introduction

- **Using Burrows-Wheeler Transform (BWT)**
  - Leverage last-to-first (LF) mapping

- **Pan-Genomic Indexes on run-length BWT (RLBWT)**
  - Conventionally, cannot compute LF steps in constant time

- **Nishimoto and Tabei's OptBWTR** (ICALP '21)
  - New, simple and constant-time implementation

Introduction

We show experimentally that their approach can be made practical for LF even without theoretical guarantees

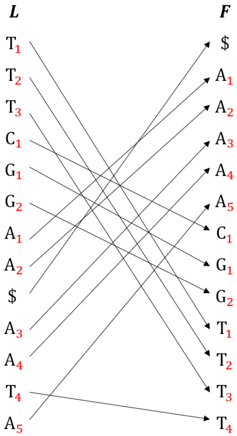
LF Permutation

$T = \text{GATTAGATACAT}$

L	F
T_1	$\$$
T_2	A_1
T_3	A_2
C_1	A_3
G_1	A_4
G_2	A_5
A_1	C_1
A_2	G_1
$\$$	G_2
A_3	T_1
A_4	T_2
T_4	T_3
A_5	T_4

LF Permutation

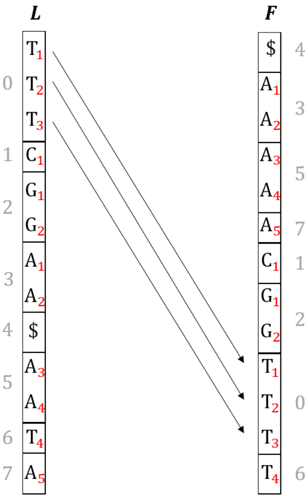
T = GATTAGATACAT



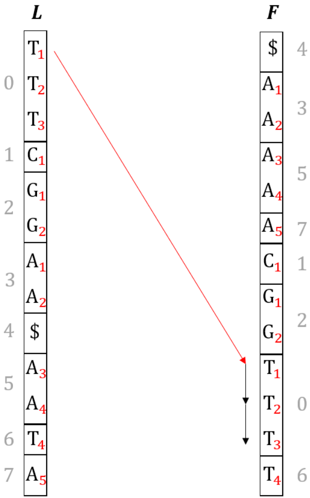
LF Runs

	L	***F***	
0	T_1	$\$$	4
	T_2	A_1	3
	T_3	A_2	
1	C_1	A_3	5
2	G_1	A_4	
	G_2	A_5	7
3	A_1	C_1	1
	A_2	G_1	2
4	$\$$	G_2	
5	A_3	T_1	0
	A_4	T_2	
6	T_4	T_3	
7	A_5	T_4	6

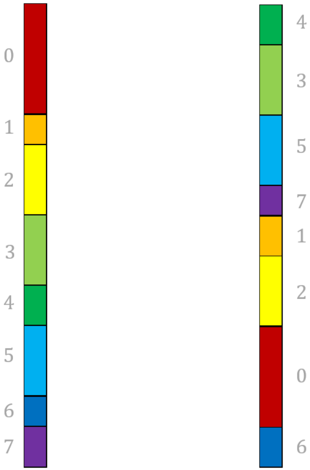
LF Runs


L
F
T1
0 T2
T3
1 C1
G1
2
G2
A1
3
A2
4 $
A3
5
A4
6 T4
7 A5
$ 4
A1
A2 3
A3
A4 5
A5 7
C1 1
G1
G2 2
T1
T2 0
T3
T4 6

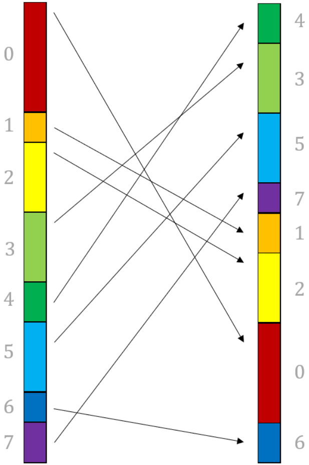
LF Runs


L
F
0 T_1 T_2 T_3
1 C_1
2 G_1 G_2
3 A_1 A_2
4 $
5 A_3 A_4
6 T_4
7 A_5
$ 4
A_1 A_2 3
A_3 A_4 5
A_5 7
C_1 1
G_1 G_2 2
T_1 T_2 T_3 0
T_4 6

Any Permutation



Any Permutation



LF Table

L	F	*run*	*offset*
T_1	\$	0	0
T_2	A_1	0	1
T_3	A_2	0	2
C_1	A_3	1	0
G_1	A_4	2	0
G_2	A_5	2	1
A_1	C_1	3	0
A_2	G_1	3	1
\$	G_2	4	0
A_3	T_1	5	0
A_4	T_2	5	1
T_4	T_3	6	0
A_5	T_4	7	0

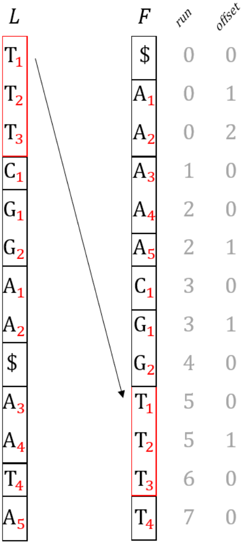
	character	*length*	*destination*	*offset*
0				
1				
2				
3				
4				
5				
6				
7				

LF Table

L	*F*	*run*	*offset*
T_1	$	0	0
T_2	A_1	0	1
T_3	A_2	0	2
C_1	A_3	1	0
G_1	A_4	2	0
G_2	A_5	2	1
A_1	C_1	3	0
A_2	G_1	3	1
$	G_2	4	0
A_3	T_1	5	0
A_4	T_2	5	1
T_4	T_3	6	0
A_5	T_4	7	0

	character	*length*	*destination*	*offset*
0	T	3		
1				
2				
3				
4				
5				
6				
7				

LF Table



L	F	run	offset
T_1	\$	0	0
T_2	A_1	0	1
T_3	A_2	0	2
C_1	A_3	1	0
G_1	A_4	2	0
G_2	A_5	2	1
A_1	C_1	3	0
A_2	G_1	3	1
\$	G_2	4	0
A_3	T_1	5	0
A_4	T_2	5	1
T_4	T_3	6	0
A_5	T_4	7	0

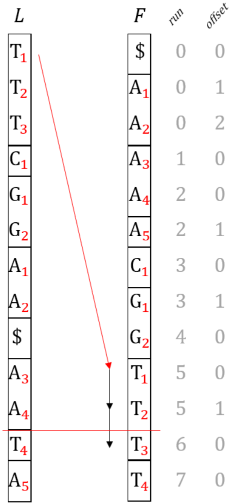
	character	length	destination	offset
0	T	3	5	0
1				
2				
3				
4				
5				
6				
7				

LF Table

L	*F*	*run*	*offset*
T_1	\$	0	0
T_2	A_1	0	1
T_3	A_2	0	2
C_1	A_3	1	0
G_1	A_4	2	0
G_2	A_5	2	1
A_1	C_1	3	0
A_2	G_1	3	1
\$	G_2	4	0
A_3	T_1	5	0
A_4	T_2	5	1
T_4	T_3	6	0
A_5	T_4	7	0

	character	*length*	*destination*	*offset*
0	T	3	5	0
1	C	1	3	0
2	G	2	3	1
3	A	2	0	1
4	\$	1	0	0
5	A	2	1	0
6	T	1	7	0
7	A	1	2	1

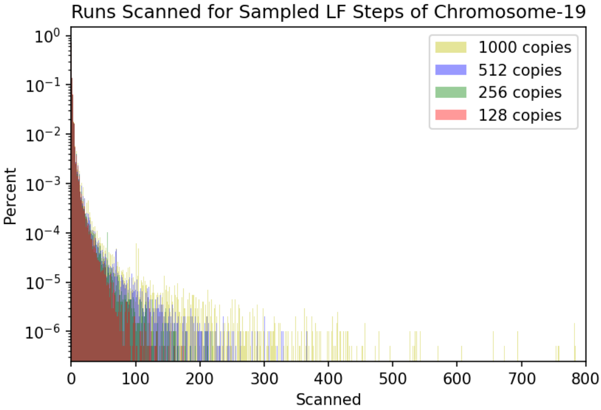
Crossing Boundaries



L	F	*run*	*offset*
T_1	$\$$	0	0
T_2	A_1	0	1
T_3	A_2	0	2
C_1	A_3	1	0
G_1	A_4	2	0
G_2	A_5	2	1
A_1	C_1	3	0
A_2	G_1	3	1
$\$$	G_2	4	0
A_3	T_1	5	0
A_4	T_2	5	1
T_4	T_3	6	0
A_5	T_4	7	0

Boundaries in Practice

- Nishimoto and Tabei limit crossings (additional space)
- 98% cross less than 5 boundaries



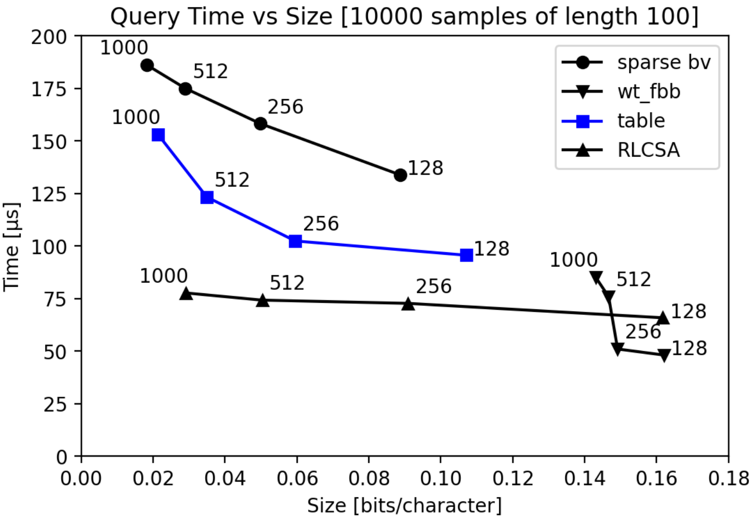
Compression

- Preliminary results vs. conventional approach:
  - LF steps ≈ 6 times faster
  - Table ≈ 14 times larger
- We devise a compression scheme specific to LF
  - To perform column-wise compression, partition into blocks to mitigate locality concerns
  - For alphabet size σ, LF mapping of run-heads forms σ non-decreasing subsequences

Setup

- Randomly sample 10000 patterns of length 100 and compute count queries
- Query against chromosome-19 genomes of 128, 256, 512 and 1000 copies
- **Data Structures**:
    - **sparse bv**: The sparse bitvector component of *r*-index
    - **wt_fbb** Fixed block boosting wavelet tree
    - **table** Our implementation of LF using Nishimoto and Tabei's approach
    - **RLCSA** BWT component of run-length encoded compressed suffix array

Results


Query Time vs Size [10000 samples of length 100]
Time [µs]
Size [bits/character]
200
175
150
125
100
75
50
25
0
0.00
0.02
0.04
0.06
0.08
0.10
0.12
0.14
0.16
0.18
sparse bv
wt_fbb
table
RLCSA
1000
512
256
128
1000
512
256
128
1000
512
256
128
1000
512
256
128

Thanks

- Email: `nathaniel.brown@dal.ca`
- Full Paper: `https://arxiv.org/abs/2112.04271`