Relative FM-indexes

Jouni Sirén, University of Chile
with
Djamal Belazzougui, University of Helsinki
Travis Gagie, University of Helsinki
Simon Gog, Karlsruhe Institute of Technology
Giovanni Manzini, University of Eastern Piedmont

Relative data compression

- Individual genomes are represented by their differences to the reference genome.

- Version control systems store revisions of a document as insertions and deletions to older revisions.

- Relative Lempel-Ziv parsing (Kuruppu et al., SPIRE 2010) represents a string as a concatenation of substrings of another string.

Compressed data structures

- Provide the functionality of a data structure, while using similar space as a compressed representation of the data.

- Take advantage of the repetitiveness of the data to store it in less space than the information-theoretic minimum.

- Are always trade-offs between space usage and query performance.

Relative data structures

- Let R and S be similar datasets.
- If we build data structure D for the datasets, D(R) and D(S) should also be similar.
- We can encode D(S) relative to D(R) as D(S | R).
- Given D(R) and D(S | R), we can query D(S) efficiently or decompress it.

Relative data structures	Compressed data structures
Individual data structures for each of the datasets.	A single data structure containing all the datasets.
The encoding of S depends only on S and R.	The encoding of S may depend on all the other datasets.
Compression depends on the similarity between S and R.	Compression may take advantage of the similarities between all the datasets.
Construction for multiple datasets is easy to distribute.	Construction for multiple datasets may require significant resources and specialized algorithms.
Datasets can be added and removed easily.	Insertion and deletion require specialized algorithms.

The suffix array (SA) is an array of pointers to the suffixes of the text in lexicographic order.

The Burrows-Wheeler transform (BWT) lists the characters preceding the suffixes.

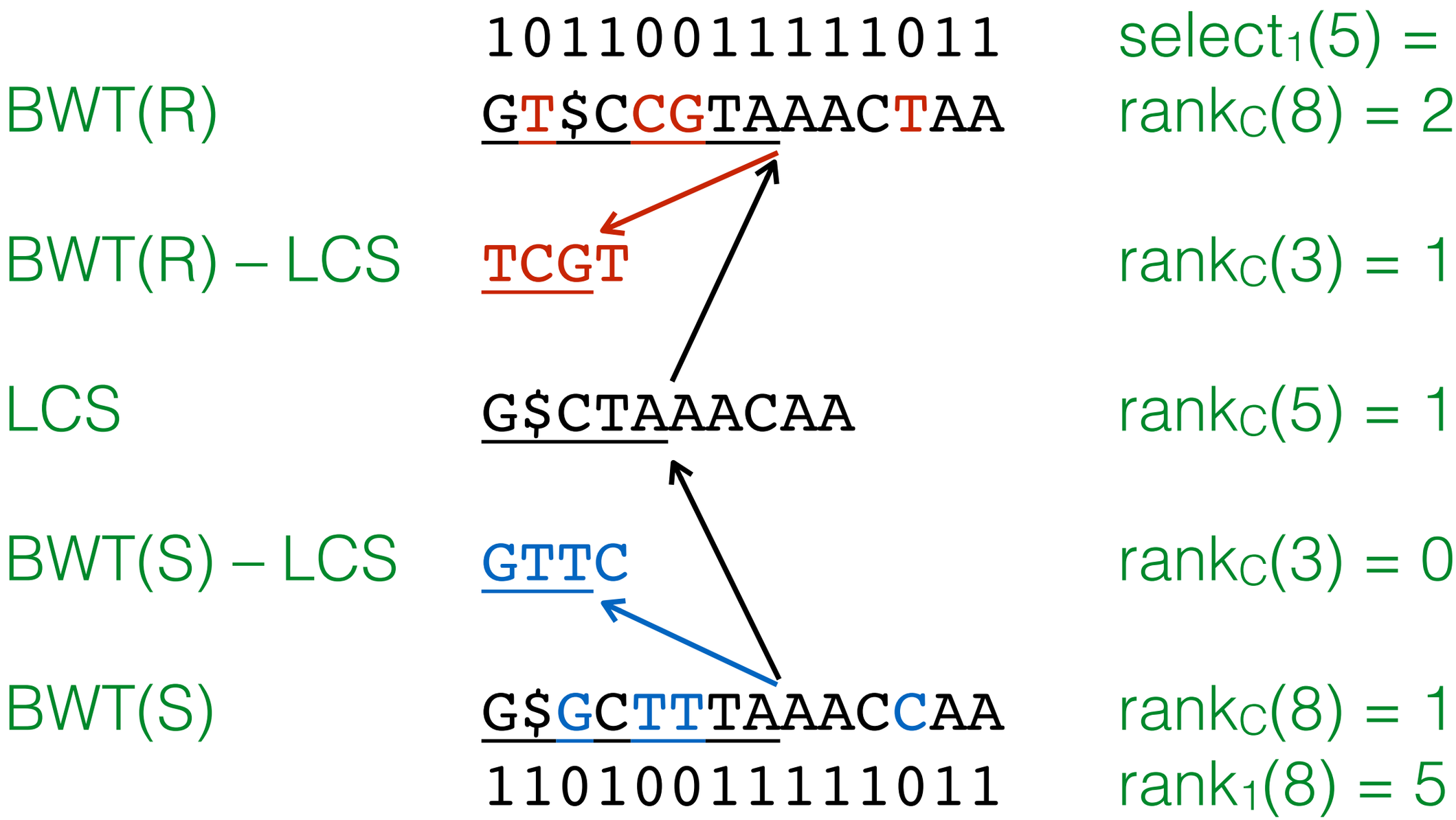
The FM-index (Ferragina, Manzini, JACM 2005) simulates the SA with BWT and additional structures to support $\text{rank}_c(\text{BWT}, i)$.

SA	BWT	Sorted suffixes
14	G	$
7	T	ACATCAG$
1	$	ACGATTACATCAG$
12	C	AG$
9	C	ATCAG$
4	G	ATTACATCAG$
11	T	CAG$
8	A	CATCAG$
2	A	CGATTACATCAG$
13	A	G$
3	C	GATTACATCAG$
6	T	TACATCAG$
10	A	TCAG$
5	A	TTACATCAG$

ACGATTACATCAG$	**BWT**	**BWT**	**ACGACTATATCAG$**
$	G	G	$
ACATCAG$	T	$	ACGACTATATCAG$
ACGATTACATCAG$	$	G	ACTATATCAG$
AG$	C	C	AG$
ATCAG$	C	T	ATATCAG$
ATTACATCAG$	G	T	ATCAG$
CAG$	T	T	CAG$
CATCAG$	A	A	CGACTATATCAG$
CGATTACATCAG$	A	A	CTATATCAG$
G$	A	A	G$
GATTACATCAG$	C	C	GACTATATCAG$
TACATCAG$	T	C	TATATCAG$
TCAG$	A	A	TATCAG$
TTACATCAG$	A	A	TCAG$

ACGATTACATCAG$	**BWT**	**BWT**	**ACGACTATATCAG$**
$	G	G	$
ACATCAG$	T	$	ACGACTATATCAG$
ACGATTACATCAG$	$	G	ACTATATCAG$
AG$	C	C	AG$
ATCAG$	C	T	ATATCAG$
ATTACATCAG$	G	T	ATCAG$
CAG$	T	T	CAG$
CATCAG$	A	A	CGACTATATCAG$
CGATTACATCAG$	A	A	CTATATCAG$
G$	A	A	G$
GATTACATCAG$	C	C	GACTATATCAG$
TACATCAG$	T	C	TATATCAG$
TCAG$	A	A	TATCAG$
TTACATCAG$	A	A	TCAG$

Computing $rank_C(i)$



Relative FM-index

- Sequences BWT(R) – LCS and BWT(S) – LCS with rank support (e.g. as wavelet trees [Grossi et al., SODA 2003]).

- Compressed bitvectors marking LCS in BWT(R) and BWT(S).

- We use SDSL (Gog et al., SEA 2014) in the implementation.

Index construction

- Finding the longest common subsequence (LCS) of two long strings is expensive.

- Myers' algorithm (Algorithmica, 1986) takes O(nD) time, which is roughly $n^2/50$ for the BWTs of human genomes.

- We approximate the LCS by partitioning the BWTs and using Myers' algorithm for each pair of partitions.

$	G	G	$
ACATCAG$	T	$	ACGACTATATCAG$
ACGATTACATCAG$	$	G	ACTATATCAG$
AG$	C	C	AG$
ATCAG$	C	T	ATATCAG$
ATTACATCAG$	G	T	ATCAG$
CAG$	T	T	CAG$
CATCAG$	A		
CGATTACATCAG$	A	A	CGACTATATCAG$
		A	CTATATCAG$
G$	A	A	G$
GATTACATCAG$	C	C	GACTATATCAG$
TACATCAG$	T	C	TATATCAG$
		A	TATCAG$
TCAG$	A	A	TCAG$
TTACATCAG$	A		

Datasets

- Reference: 1000 Genomes Project assembly of the human reference genome (3096M bases).

- YanHuang: An older assembly of the genome of a Han Chinese male (3002M bases).

- NA12878: The maternal haplotype of a Utah female from the 1000 Genomes Project (3036M bases).

- Patterns: 3.68 million reads of length 108 from the 1000 Genomes Project individual HG00122 (British female).

Dataset	LCS	Construction	Plain FM-index	RRR FM-index	Relative FM-index
YanHuang	2935M 94.82% 97.79%	708 s	1090 MB 56.45 s	628 MB 328.86 s	288 MB 621.47 s
NA12878	2992M 96.65% 98.54%	589 s	1090 MB 57.31 s	636 MB 325.48 s	218 MB 619.81 s
NA12878 (reference without chr Y)	2991M 98.51% 98.51%		1090 MB	636 MB	181 MB

Conclusions

- Compressing data structures relative to similar datasets seems like an interesting idea.
- The relative FM-index is roughly 5x smaller and 10x slower than a plain FM-index, or 3x smaller and 2x slower than a compressed FM-index.
- There is a report of some ongoing work on data structures based on relative Lempel-Ziv parsing in the workshop.