Compressed Suffix Arrays for Automata

Jouni Sirén
University of Helsinki, Finland
with
Niko Välimäki and Veli Mäkinen (and others?)
University of Helsinki, Finland

Jouni Sirén, Niko Välimäki, Veli Mäkinen: **Indexing Finite Language Representation of Population Genotypes**. WABI 2011.

Extended version: arXiv:1010.2656, 2011.

Jouni Sirén: **Compressed Full-Text Indexes for Highly Repetitive Collections.** PhD thesis, 2012.

Some new content as well.

Veli Mäkinen, Gonzalo Navarro, Jouni Sirén, Niko Välimäki: **Storage and Retrieval of Highly Repetitive Sequence Collections**. Journal of Computational Biology, 2010. Earlier in SPIRE 2008, RECOMB 2009.

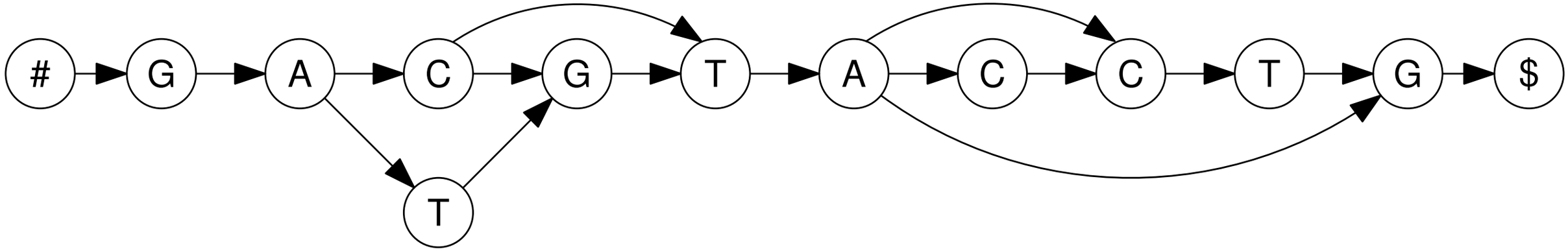
Collections of individual genomes or different versions of documents compress extremely well. With them, o(n) bits of overhead information in a CSA can be too much.

CSAs where overhead scales with compressed size.

The story begins...

- Veli visited Richard Durbin at Sanger in late 2009.
- 1000 genomes project was not planning to assemble individual genomes.
- They were going to store the reads as de Bruijn graphs.
- I misunderstood the problem.

This is what I imagined



What I thought

- They wanted to index recombinations of individual genomes in addition to the genomes themselves.
- We can probably use bit vectors to split and join paths in the automaton.
- RLCSA analysis: Substrings between SNPs are usually unique.

Backward searching

Suffixes	**BWT**
$	G
ACCTG$	T
ACGTACCTG$	G
CCTG$	A
CGTACCTG$	A
CTG$	C
G$	T
GACGTACCTG$	$
GTACCTG$	C
TACCTG$	G
TG$	C

Suffixes matching pattern AC

Suffixes starting with T

Backward searching

Suffixes	BWT
$	G
ACCTG$	T
ACGTACCTG$	G
CCTG$	A
CGTACCTG$	A
CTG$	C
G$	T
GACGTACCTG$	$
GTACCTG$	C
TACCTG$	G
TG$	C

Suffixes matching pattern AC

Suffixes matching pattern TAC

Nodes with label c must be in the same order as nodes having a predecessor with label c.

Requirements for the automaton

- Multiple suffixes can be recognized from most nodes.
- We should get the same order for the nodes, regardless of which suffix we use as a sort key.
- Each node should correspond to a lexicographic range of suffixes.

Definition.

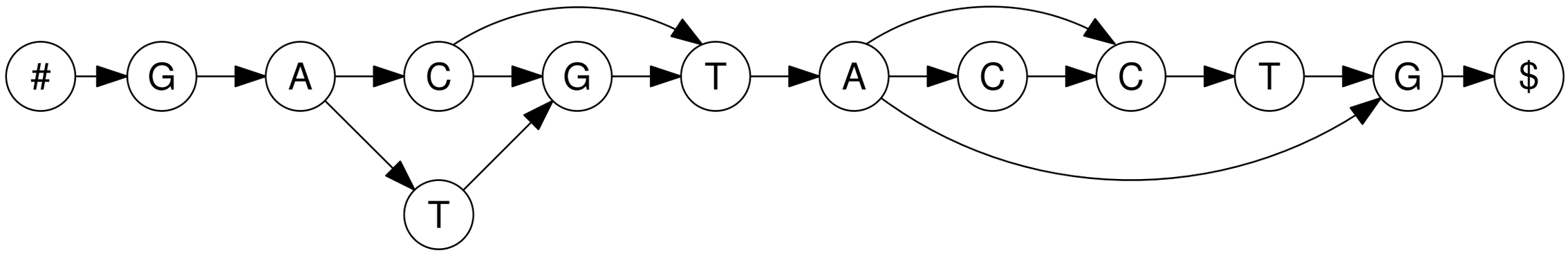
Let $A = (V, E)$ be a finite automaton, and let $v \in V$ be a node. Let $rng(v)$ be the smallest lexicographic range containing all suffixes that can be recognized from node v. Automaton A is *prefix-range-sorted*, if $rng(v) \cap rng(v') = \varnothing$ for all $v' \neq v$.

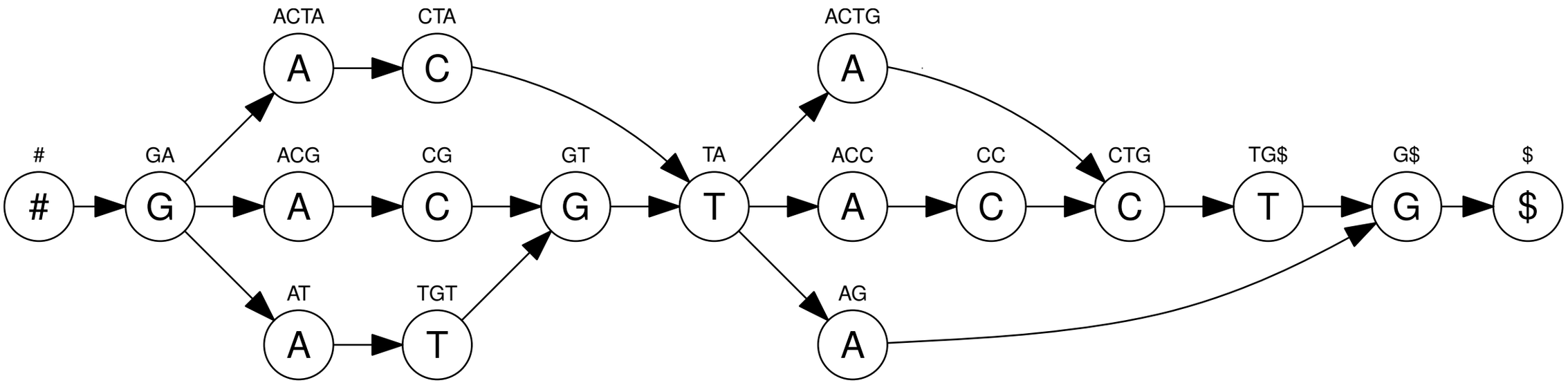
Prefix-doubling

- Find a prefix-range-sorted automaton equivalent to the original automaton.
- Nodes are paths in the original automaton.
- $(u, v, r) + (v, w, r') \longmapsto (u, w, (r, r'))$
- If all paths sharing a rank start from the same original node, we merge them.

Creating the edges

- We first merge paths with adjacent ranks starting from the same original node.
- $(u, v) + (v, v', r') \longmapsto (u, (v, v'))$
- Sort edges by $(l(u), r')$ and scan the lists.
- $(u, u', r) + (u, (v, v')) \longmapsto ((u, u'), (v, v'))$
- The edges were sorted by r.





	$	ACC	ACG	ACTA	ACTG	AG	AT	CC	CG	CTA	CTG	G$	GA	GT	TA	TG$	TGT	#
BWT	G	T	G	G	T	T	G	A	A	A	AC	AT	#	CT	CG	C	A	$
Edges	1	1	1	1	1	1	1	1	1	1	1	1	100	1	100	1	1	1

Index construction

- Human genome and genetic variation in the Finnish subpopulation of the 1000 genomes project.
- 4x Xeon X7550 (32 cores + HT, used 24 cores) and 1 TB of memory.
- Index construction took 10 hours, 181 GB.
- Final index takes 2.8 gigabytes.

Analysis

- Assume a random sequence of length n and random mutations with probability p.
- The expected number of paths of length k starting from a given position is $(1+p)^k$.
- For reasonable values of p, the expected number of nodes is $n(1+p)^{O(\log_\sigma n)} + O(1)$.
- This is $O(n)$ for $p = O(1 / \log_\sigma n)$.

Search performance

- Backward searching requires one extra bit vector operation per character.
- locate() can be slow due to duplicates.
- Theoretically 2x slower than a similar CSA.
- In approximate matching, GCSA is 1.5 to 2.5 times slower than RLCSA using the same algorithm.

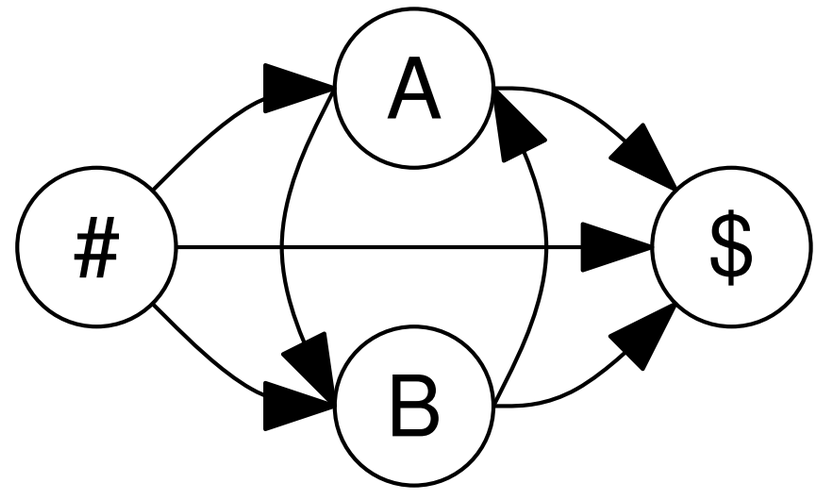
Multiple automata

- We can index multiple automata in the same way as multiple sequences.
- Indexing two identical automata results in exponential growth, as end markers are required to distinguish the paths.
- Maybe we can solve this by aligning the automata.

Class of languages

- Prefix-range-sorted automata exist for all finite languages (consider tries).

- Some infinite languages can also be recognized.



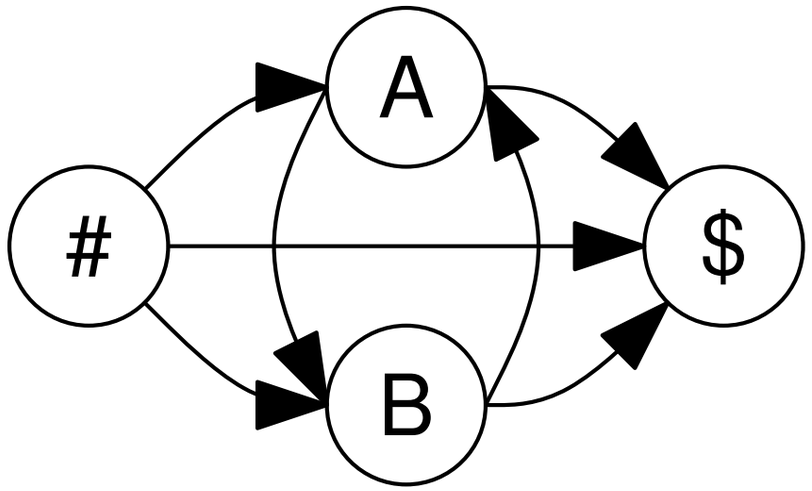
Theorem. Not all regular languages can be recognized by prefix-range-sorted automata.

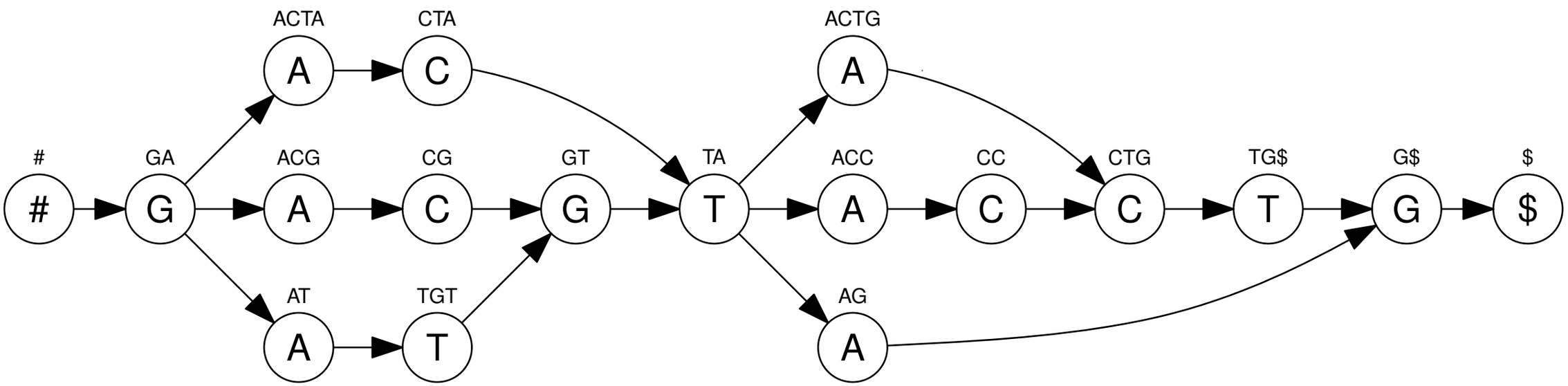
Consider the language $L = \{a, b\}^* \cup \{a, c\}^*$ and its suffixes of type $B_n = a^n b$ and $C_n = a^n c$.

If B_n and C_n can be recognized from the same node, then $bC_n \in L$ – a contradiction.

As $C_{n+1} < B_n < C_n$, there must be separate nodes to recognize suffixes C_i for all i – the automaton must be infinite.

De Bruijn graphs?





An order-k de Bruijn graph is a prefix-range-sorted automaton, where rng(v) is defined by a prefix of length k.

Alexander Bowe, Taku Onodera, Kunihiko Sadakane, Tetsuo Shibuya: **Succinct de Bruijn Graphs**. WABI 2012.

De Bruijn graphs with m edges in m(log σ + 2) + o(m) bits. Based on the XBW transform. Different terminology and different design choices, but the core combinatorial structure is essentially GCSA.

I had solved the right problem already in 2009, but nobody noticed!

THANK YOU!