Fast matching statistics for sets of long similar strings

Zsuzsanna Lipták[1] Martina Lucà[1] Francesco Masillo[1] Simon J. Puglisi[2]

[1]University of Verona

[2]University of Helsinki

PSC 2024, August 26-27, 2024 - Prague, Czech Republic

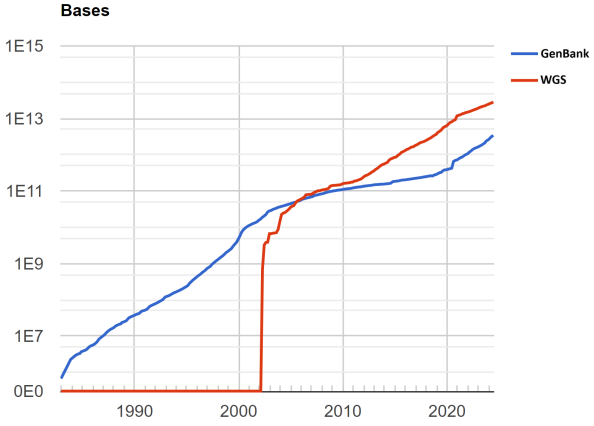
Introduction to Matching Statistics

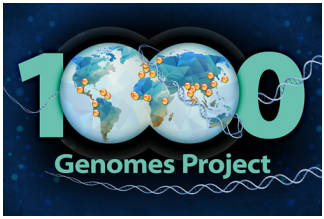
- data structure introduced by Chang and Lawler in 1994[1]
- computed in linear time in the size of the input
- many applications to classical string problems (see Gusfield's book[2]):
  - longest common substrings
  - exact matches
  - longest prefix matching
  - approximate pattern matching
  - ...
- many bioinformatics applications:
  - read alignment
  - MEM and MUM finding
  - computing phylogenies of a set of genomes
  - suffix array[3] and BWT[4] construction of sets of genomes
  - detection of SNVs or sequencing errors in read collections
  - ...

1. W. I. Chang and E. L. Lawler: Sublinear approximate string matching and biological applications. Algorithmica, 12(4/5) 1994, pp. 327–344.
2. D. Gusfield: Algorithms on Strings, Trees, and Sequences, Cambridge University Press, 1997.
3. Zs. Lipták, F. Masillo, and S. J. Puglisi: Suffix sorting via matching statistics. Algorithms Mol. Biol., 19(1) 2024, pp. 11:1–11:18.
4. F. Masillo: Matching statistics speed up BWT construction, Proc. of ESA 2023, pp. 83:1–83:15.

Motivation

- incredibly fast growth of data
- repetition of contents





Matching Statistics

- two strings R and S
- matching factor U, longest prefix of a suffix occurring as a substring in R
- $\texttt{MS}[i] = (p_i, \ell_i)$, where p_i is an occurrence of U in R, and ℓ_i is the length of U

Example

	i	1	2	3	4	5	6	7	8	9	10	11	12	13
R		A	G	T	A	C	A	A	A	T	A	G	G	#

	i	1	2	3	4	5	6	7	8	9	10	11	12
S		T	T	A	T	G	T	A	C	A	A	A	$
MS	p_i	9	9	7	3	2	3	4	5	6	7	8	0
	ℓ_i	1	2	2	1	7	6	5	4	3	2	1	0

Matching Statistics

- two strings R and S
- matching factor U, longest prefix of a suffix occurring as a substring in R
- $\mathtt{MS}[i] = (p_i, \ell_i)$, where p_i is an occurrence of U in R, and ℓ_i is the length of U

Example

	i	1	2	3	4	5	6	7	8	9	10	11	12	13
R		A	G	T	A	C	A	A	A	T	A	G	G	#

	i	1	2	3	4	5	6	7	8	9	10	11	12
S		T	T	A	T	G	T	A	C	A	A	A	$
MS	p_i	9	9	7	3	2	3	4	5	6	7	8	0
	ℓ_i	1	2	2	1	7	6	5	4	3	2	1	0

Matching Statistics

- two strings R and S
- matching factor U, longest prefix of a suffix occurring as a substring in R
- $\texttt{MS}[i] = (p_i, \ell_i)$, where p_i is an occurrence of U in R, and ℓ_i is the length of U

Example

	i	1	2	3	4	5	6	7	8	9	10	11	12	13
R		A	G	T	A	C	A	A	A	T	A	G	G	#

	i	1	2	3	4	5	6	7	8	9	10	11	12
S		T	T	A	T	G	T	A	C	A	A	A	\$
MS	p_i	9	9	7	3	2	3	4	5	6	7	8	0
	ℓ_i	1	2	2	1	7	6	5	4	3	2	1	0

Matching Statistics

- two strings R and S
- matching factor U, longest prefix of a suffix occurring as a substring in R
- $\texttt{MS}[i] = (p_i, \ell_i)$, where p_i is an occurrence of U in R, and ℓ_i is the length of U

Example

	i	1	2	3	4	5	6	7	8	9	10	11	12	13
R		A	G	T	A	C	A	A	A	T	A	G	G	#

	i	1	2	3	4	5	6	7	8	9	10	11	12
S		T	T	A	T	G	T	A	C	A	A	A	$
MS	p_i	9	9	7	3	2	3	4	5	6	7	8	0
	ℓ_i	1	2	2	1	7	6	5	4	3	2	1	0

Matching Statistics

- two strings R and S
- matching factor U, longest prefix of a suffix occurring as a substring in R
- $\texttt{MS}[i] = (p_i, \ell_i)$, where p_i is an occurrence of U in R, and ℓ_i is the length of U

Example

	i	1	2	3	4	5	6	7	8	9	10	11	12	13
R		A	G	T	A	C	A	A	A	T	A	G	G	#

	i	1	2	3	4	5	6	7	8	9	10	11	12
S		T	T	A	T	G	T	A	C	A	A	A	$
MS	p_i	9	9	8	3	2	3	4	5	6	7	8	9
	ℓ_i	1	2	2	1	7	6	5	4	3	2	1	0

Matching Statistics

- two strings R and S
- matching factor U, longest prefix of a suffix occurring as a substring in R
- $\texttt{MS}[i] = (p_i, \ell_i)$, where p_i is an occurrence of U in R, and ℓ_i is the length of U

Example

	i	1	2	3	4	5	6	7	8	9	10	11	12	13
R		A	G	T	A	C	A	A	A	T	A	G	G	#

	i	1	2	3	4	5	6	7	8	9	10	11	12
S		T	T	A	T	G	T	A	C	A	A	A	$
MS	p_i	9	9	8	3	2	3	4	5	6	7	8	9
	ℓ_i	1	2	2	1	7	6	5	4	3	2	1	0

Computing the Matching Statistics

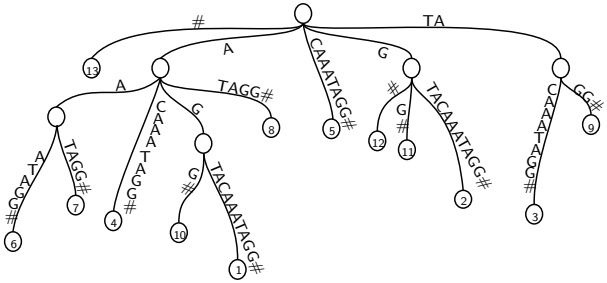
For each MS entry, we have two phases:

- right extension
- left contraction

	i	1	2	3	4	5	6	7	8	9	10	11	12
S		T	T	A	T	G	T	A	C	A	A	A	$
MS	p_i												
	ℓ_i												

R = AGTACAAATAGG#

Suffix Tree of R



Computing the Matching Statistics

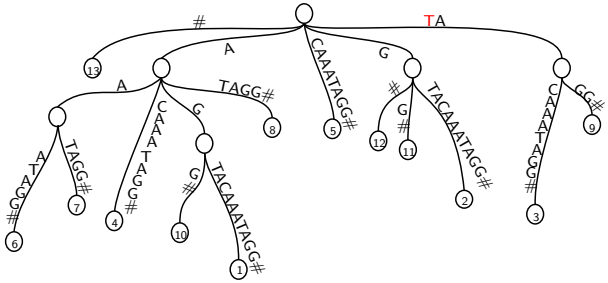
For each MS entry, we have two phases:

- right extension
- left contraction

	i	1	2	3	4	5	6	7	8	9	10	11	12
S		T	T	A	T	G	T	A	C	A	A	A	$
MS	p_i												
	ℓ_i												

R = AGTACAAATAGG#

Suffix Tree of R



Computing the Matching Statistics

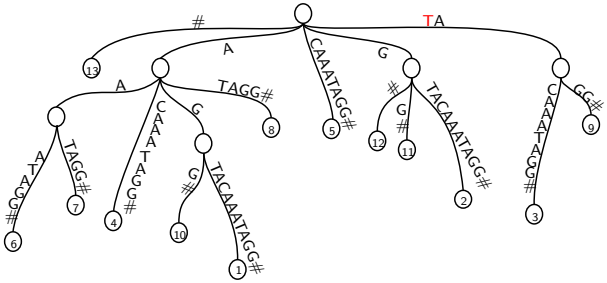
For each MS entry, we have two phases:

- right extension
- left contraction

	i	1	2	3	4	5	6	7	8	9	10	11	12
S		T	T	A	T	G	T	A	C	A	A	A	$
MS	p_i	9											
	ℓ_i	1											

R = AGTACAAATAGG#

Suffix Tree of R



Computing the Matching Statistics

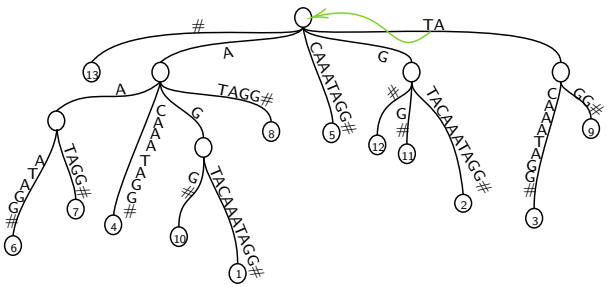
For each MS entry, we have two phases:

- right extension
- left contraction

	i	1	2	3	4	5	6	7	8	9	10	11	12
S		T	T	A	T	G	T	A	C	A	A	A	\$
MS	p_i	9											
	ℓ_i	1											

R = AGTACAAATAGG#

Suffix Tree of R



Computing the Matching Statistics

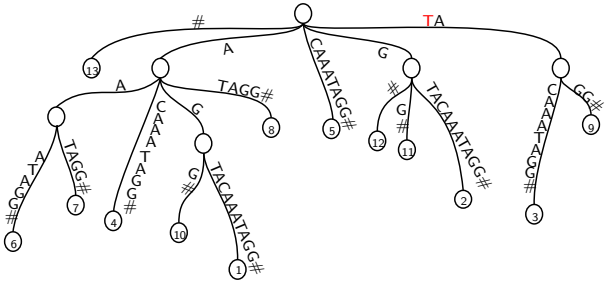
For each MS entry, we have two phases:

- right extension
- left contraction

	i	1	2	3	4	5	6	7	8	9	10	11	12
S		T	T	A	T	G	T	A	C	A	A	A	$
MS	p_i	9											
	ℓ_i	1											

R = AGTACAAATAGG#

Suffix Tree of R



Computing the Matching Statistics

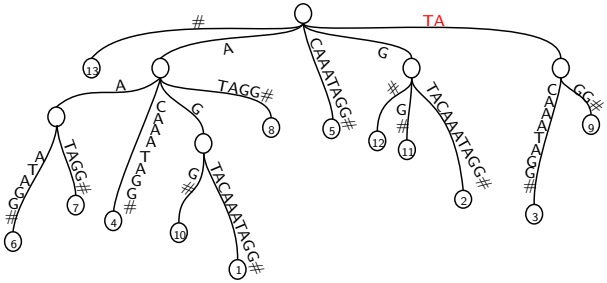
For each MS entry, we have two phases:

- right extension
- left contraction

	i	1	2	3	4	5	6	7	8	9	10	11	12
S		T	T	A	T	G	T	A	C	A	A	A	$
MS	p_i	9											
	ℓ_i	1											

R = AGTACAAATAGG#

Suffix Tree of R



Computing the Matching Statistics

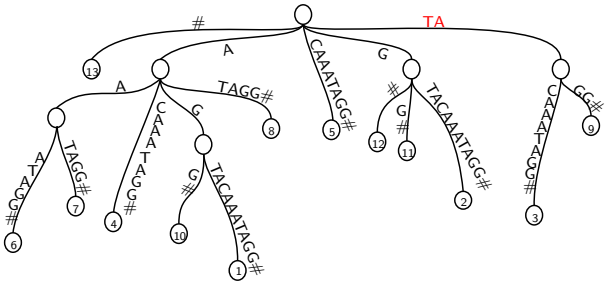
For each MS entry, we have two phases:

- right extension
- left contraction

	i	1	2	3	4	5	6	7	8	9	10	11	12
S		T	T	A	T	G	T	A	C	A	A	A	$
MS	p_i	9	9										
	ℓ_i	1	2										

R = AGTACAAATAGG#

Suffix Tree of R



Computing the Matching Statistics

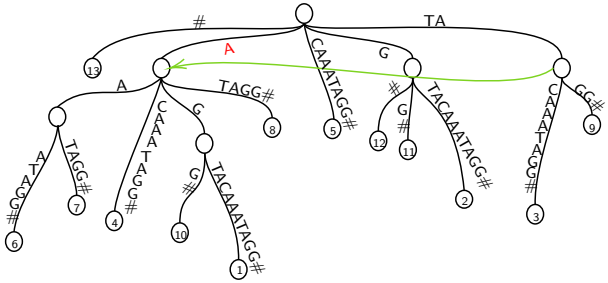
For each MS entry, we have two phases:

- right extension
- left contraction

	i	1	2	3	4	5	6	7	8	9	10	11	12
S		T	T	A	T	G	T	A	C	A	A	A	$
MS	p_i	9	9										
	ℓ_i	1	2										

R = AGTACAAATAGG#

Suffix Tree of R



Computing the Matching Statistics

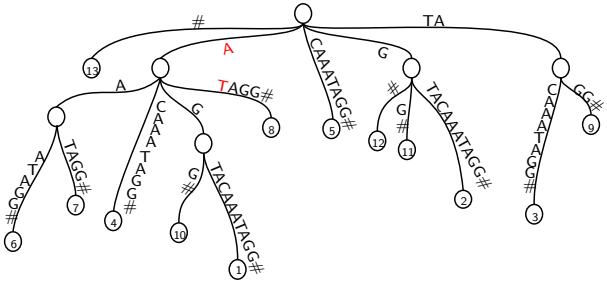
For each MS entry, we have two phases:

- right extension
- left contraction

	i	1	2	3	4	5	6	7	8	9	10	11	12
S		T	T	A	T	G	T	A	C	A	A	A	$
MS	p_i	9	9										
	ℓ_i	1	2										

R = AGTACAAATAGG#

Suffix Tree of R



Computing the Matching Statistics

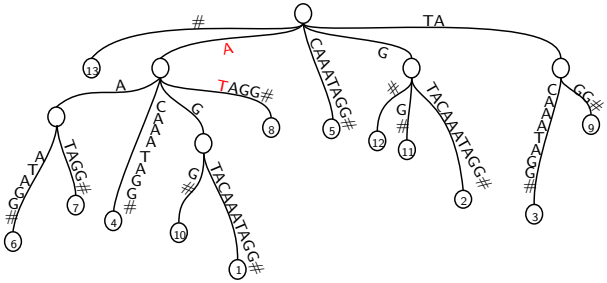
For each MS entry, we have two phases:

- right extension
- left contraction

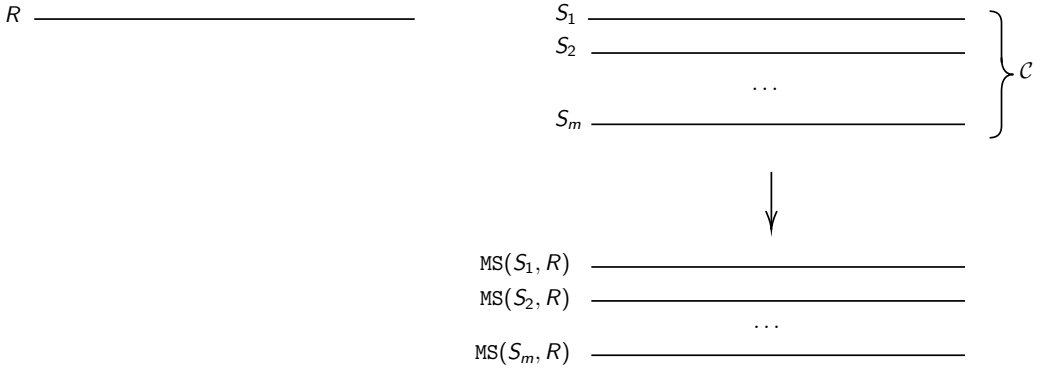
	i	1	2	3	4	5	6	7	8	9	10	11	12
S		T	T	A	T	G	T	A	C	A	A	A	$
MS	p_i	9	9	8									
	ℓ_i	1	2	2									

R = AGTACAAATAGG#

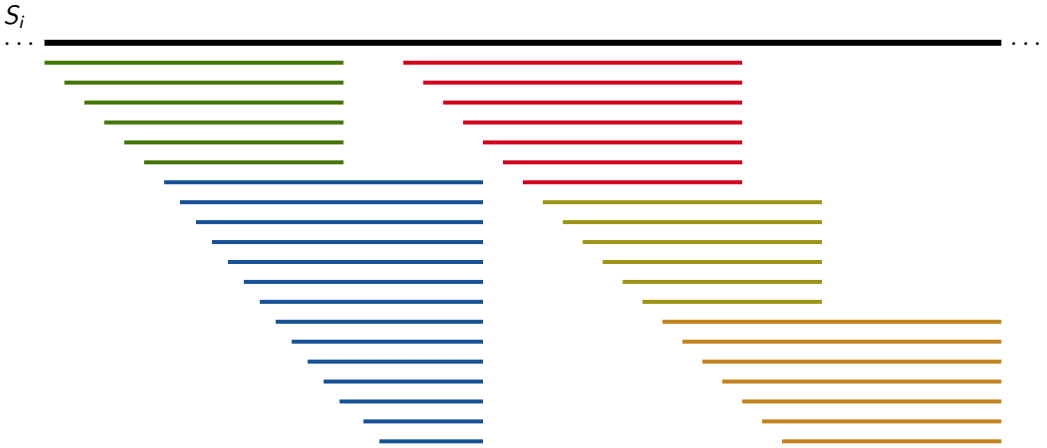
Suffix Tree of R



Matching Statistics in Our Setting

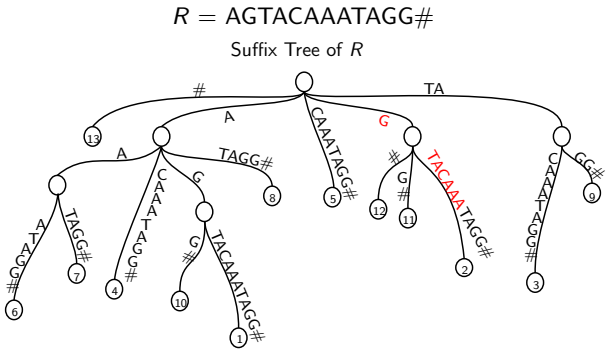


Matching Statistics in Our Setting - Property



Matching Statistics in Our Setting - Property (example)

	i	1	2	3	4	5	6	7	8	9	10	11	12
S		T	T	A	T	G	T	A	C	A	A	A	$
MS	p_i	9	9	7	3	2							
	ℓ_i	1	2	2	1	7							

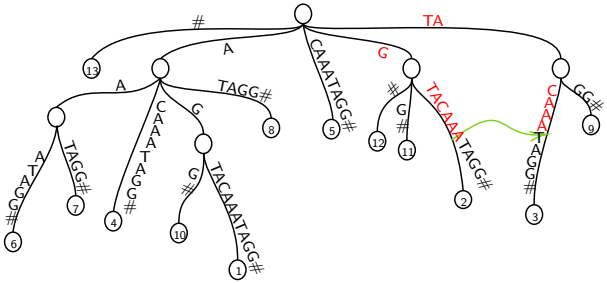


Matching Statistics in Our Setting - Property (example)

	i	1	2	3	4	5	6	7	8	9	10	11	12
S		T	T	A	T	G	T	A	C	A	A	A	$
MS	p_i	9	9	7	3	2							
	ℓ_i	1	2	2	1	7							

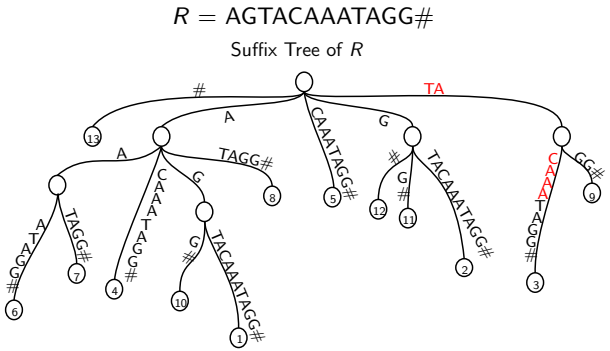
R = AGTACAAATAGG#

Suffix Tree of R



Matching Statistics in Our Setting - Property (example)

	i	1	2	3	4	5	6	7	8	9	10	11	12
S		T	T	A	T	G	T	A	C	A	A	A	$
MS	p_i	9	9	7	3	2	3						
	ℓ_i	1	2	2	1	7	6						

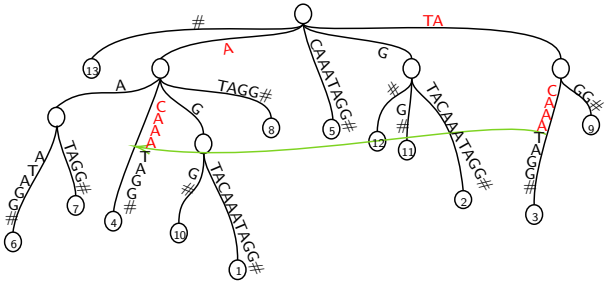


Matching Statistics in Our Setting - Property (example)

	i	1	2	3	4	5	6	7	8	9	10	11	12
S		T	T	A	T	G	T	A	C	A	A	A	$
MS	p_i	9	9	7	3	2	3						
	ℓ_i	1	2	2	1	7	6						

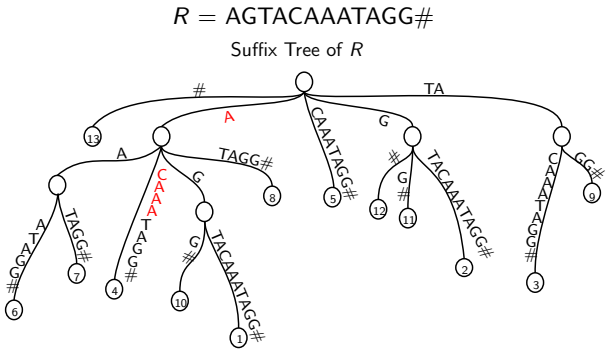
$R = \text{AGTACAAATAGG\#}$

Suffix Tree of R



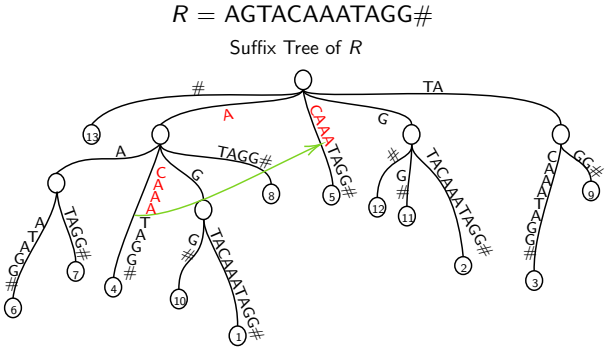
Matching Statistics in Our Setting - Property (example)

	i	1	2	3	4	5	6	7	8	9	10	11	12
S		T	T	A	T	G	T	A	C	A	A	A	$
MS	p_i	9	9	7	3	2	3	4					
	ℓ_i	1	2	2	1	7	6	5					



Matching Statistics in Our Setting - Property (example)

	i	1	2	3	4	5	6	7	8	9	10	11	12
S		T	T	A	T	G	T	A	C	A	A	A	\$
MS	p_i	9	9	7	3	2	3	4					
	ℓ_i	1	2	2	1	7	6	5					

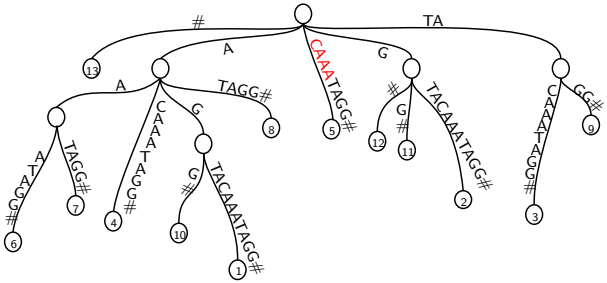


Matching Statistics in Our Setting - Property (example)

	i	1	2	3	4	5	6	7	8	9	10	11	12
S		T	T	A	T	G	T	A	C	A	A	A	$
MS	p_i	9	9	7	3	2	3	4	5	6	7	8	0
	ℓ_i	1	2	2	1	7	6	5	4	3	2	1	0

R = AGTACAAATAGG#

Suffix Tree of R



Heuristics

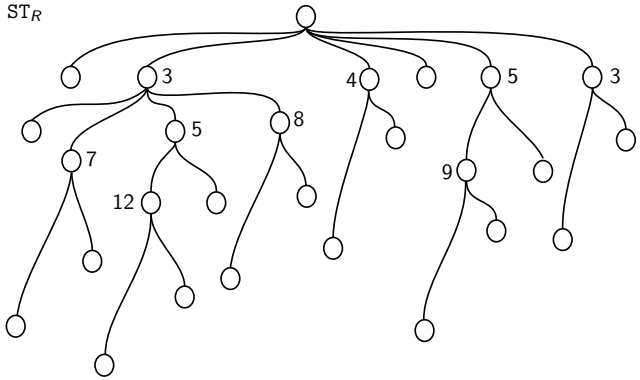
- max LCP
- block LCP
- longest repeated factor - LRF (new!)

Heuristics

- **max LCP**
- block LCP
- longest repeated factor - LRF (new!)

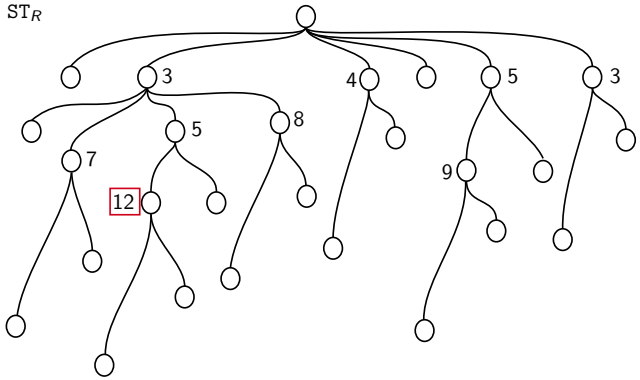
Heuristics

- **max LCP**
- block LCP
- longest repeated factor - LRF (new!)



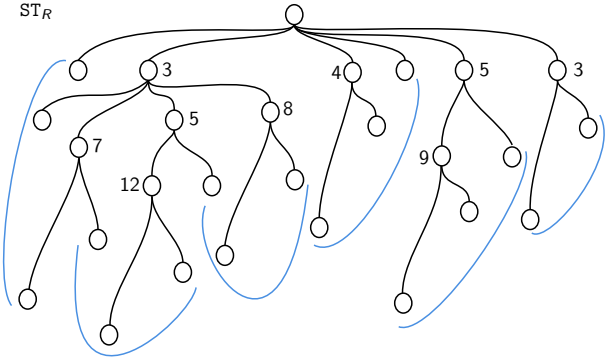
Heuristics

- **max LCP**
- block LCP
- longest repeated factor - LRF (new!)



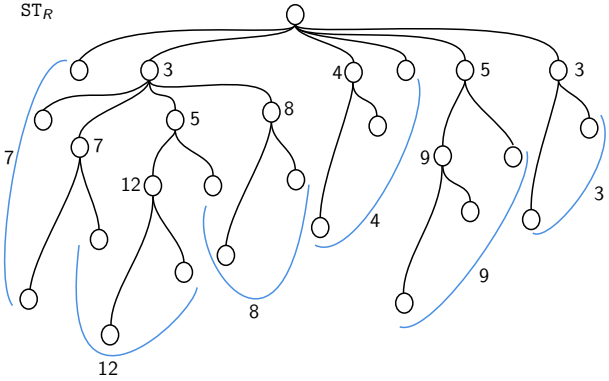
Heuristics

- max LCP
- **block LCP**
- longest repeated factor - LRF (new!)



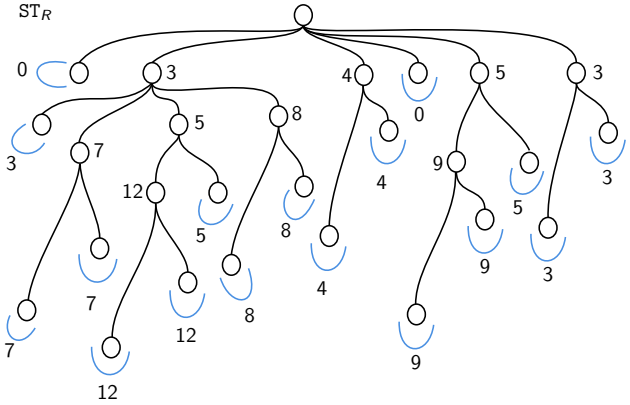
Heuristics

- max LCP
- **block LCP**
- longest repeated factor - LRF (new!)



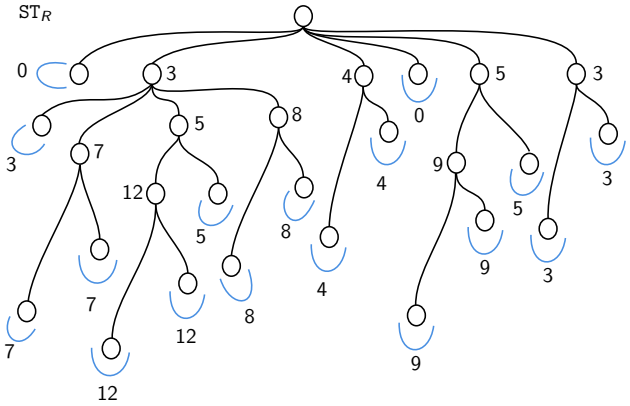
Heuristics

- max LCP
- block LCP
- **longest repeated factor - LRF (new!)**



Heuristics

- max LCP
- block LCP
- **longest repeated factor - LRF (new!)**



	···	i	$i+1$	$i+2$	···												$i+16$	···
LRF	···	7	7	5	3	4	5	0	4	12	9	9	0	8	3	3	8	12	···

LRF Heuristic

	$\cdots$	i	$i+1$	$i+2$	$\cdots$	$i+16$	$\cdots$
LRF	$\cdots$	7	7	5	3 4 5 0 4 12 9 9 0 8 3 3 8	12	$\cdots$

	$\cdots$	j	$j+1$	$j+2$	$\cdots$	$j+16$	$\cdots$
MS–ℓ_i	$\cdots$	25					$\cdots$
MS–p_i	$\cdots$	i			$\cdots$		$\cdots$

LRF Heuristic

	$\cdots$	i	$i+1$	$i+2$	$\cdots$	$i+16$	$\cdots$
LRF	$\cdots$	7	7	5	3 4 5 0 4 12 9 9 0 8 3 3 8	12	$\cdots$

	$\cdots$	j	$j+1$	$j+2$	$\cdots$	$j+16$	$\cdots$
MS–ℓ_i	$\cdots$	25	24				$\cdots$
MS–p_i	$\cdots$	i	$i+1$		$\cdots$		$\cdots$

LRF Heuristic

	$\cdots$	i	$i+1$	$i+2$	$\cdots$												$i+16$	$\cdots$
LRF	$\cdots$	7	7	5	3	4	5	0	4	12	9	9	0	8	3	3	8	12	$\cdots$
	$\cdots$	j	$j+1$	$j+2$	$\cdots$												$j+16$	$\cdots$
MS–ℓ_i	$\cdots$	25	24	23														$\cdots$
MS–p_i	$\cdots$	i	$i+1$	$i+2$	$\cdots$													$\cdots$

LRF Heuristic

	$\cdots$	i	$i+1$	$i+2$	$\cdots$												$i+16$	$\cdots$
LRF	$\cdots$	7	7	5	3	4	5	0	4	12	9	9	0	8	3	3	8	12	$\cdots$
	$\cdots$	j	$j+1$	$j+2$	$\cdots$												$j+16$	$\cdots$
MS–ℓ_i	$\cdots$	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10		$\cdots$
MS–p_i	$\cdots$	i	$i+1$	$i+2$	$\cdots$														$\cdots$

LRF Heuristic

	$\cdots$	i	$i+1$	$i+2$	$\cdots$	$i+16$	$\cdots$
LRF	$\cdots$	7	7	5	3 4 5 0 4 12 9 9 0 8 3 3 8	12	$\cdots$
	$\cdots$	j	$j+1$	$j+2$	$\cdots$	$j+16$	$\cdots$
MS–ℓ_i	$\cdots$	25	24	23	22 21 20 19 18 17 16 15 14 13 12 11 10	9	$\cdots$
MS–p_i	$\cdots$	i	$i+1$	$i+2$	$\cdots$	$i+16$	$\cdots$

Datasets and Tools

Four datasets (1 GB each):

name	no. seq.s	ref. seq. length	max LCP_R	mean LCP_R
SARS-CoV-2 genomes	36 201	29 783	17	6
Human Chromosome 19	17	59 126 939	3 099 999	81 379
Salmonella genomes	216	4 506 055	145	10
Rice Chromosome 1	23	43 992 113	12 893	27

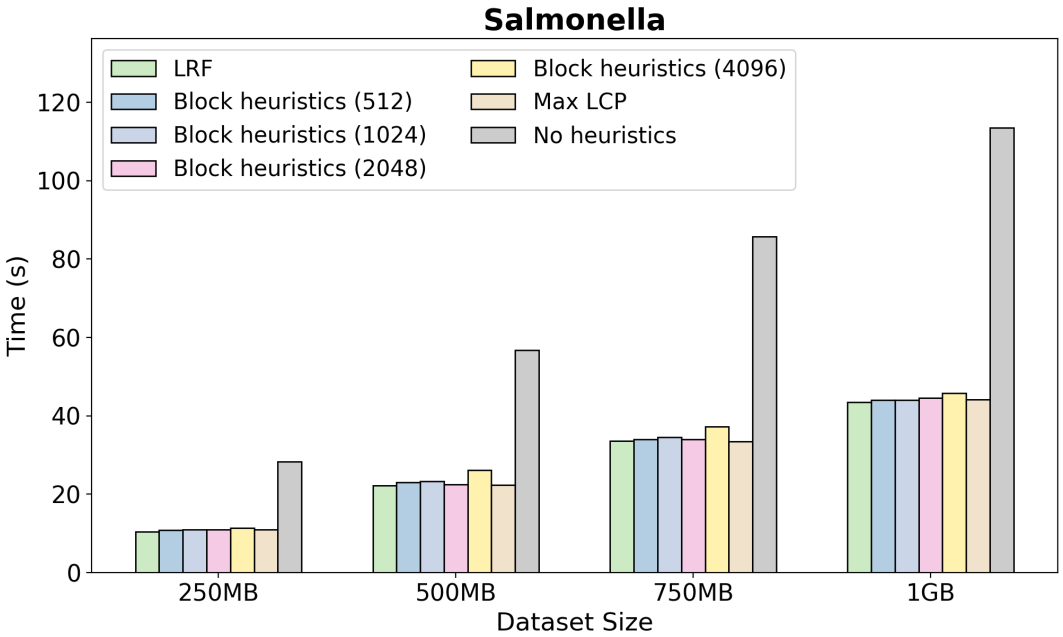
Heuristics compared:

1. no heuristic
2. max LCP (Lipták et al. 2024)
3. block LCP (Lipták et al. 2024)
4. LRF (our tool)

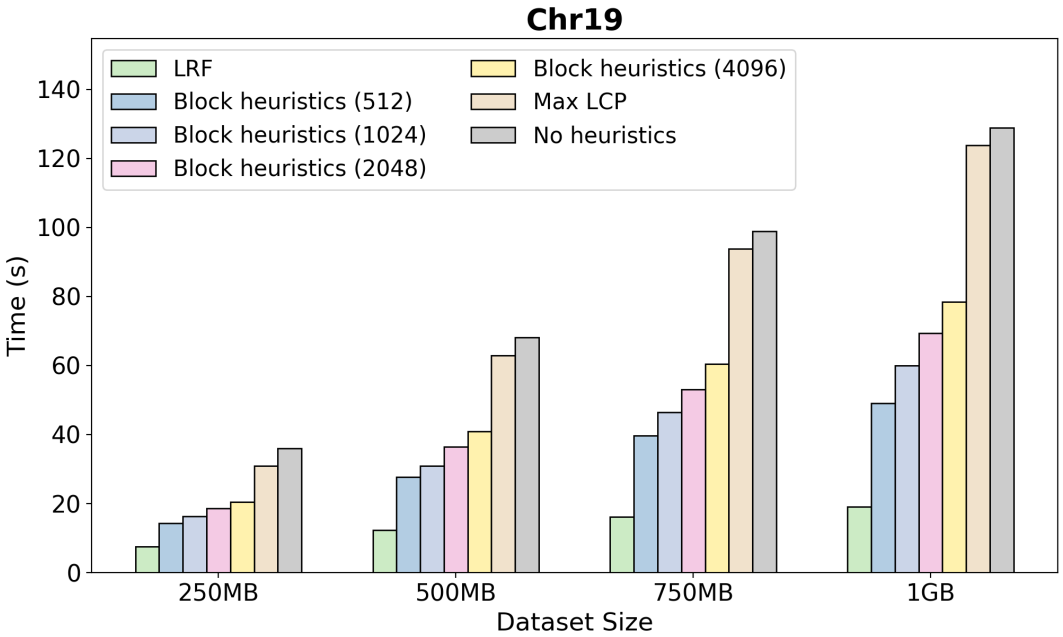
Competitor tools:

1. `indexed-ms` (Cunial et al. 2022)
2. `MONI` (Rossi et al. 2022)
3. `PHONI` (Boucher et al. 2021)
4. `AUG-PHONI` (Martínez-Guardiola et al. 2023)
5. `LRF-ms` (our tool)

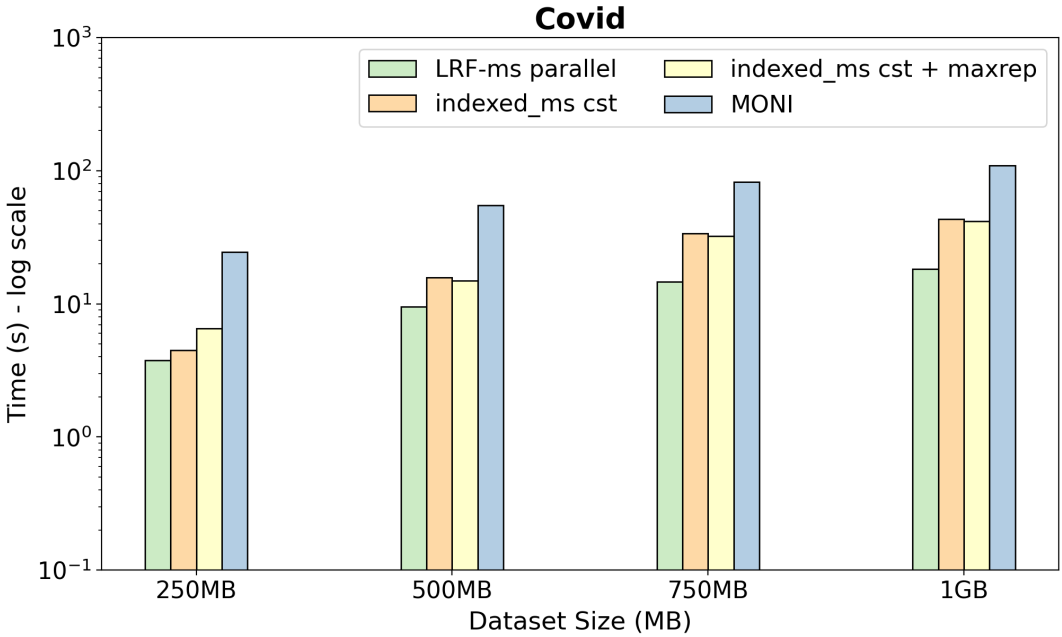
Heuristics Comparison



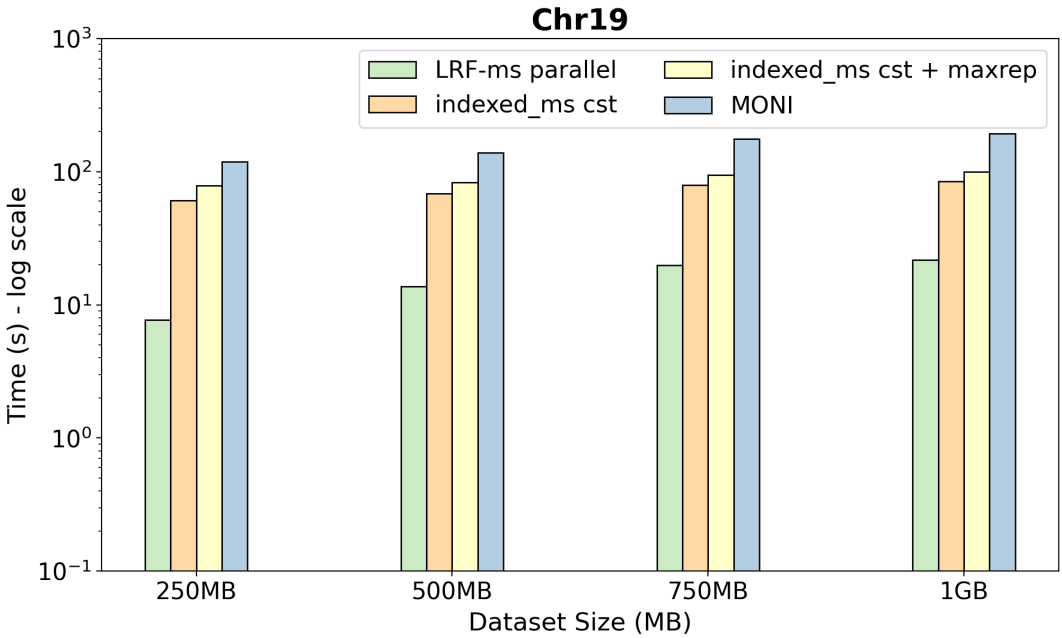
Heuristics Comparison (continued)



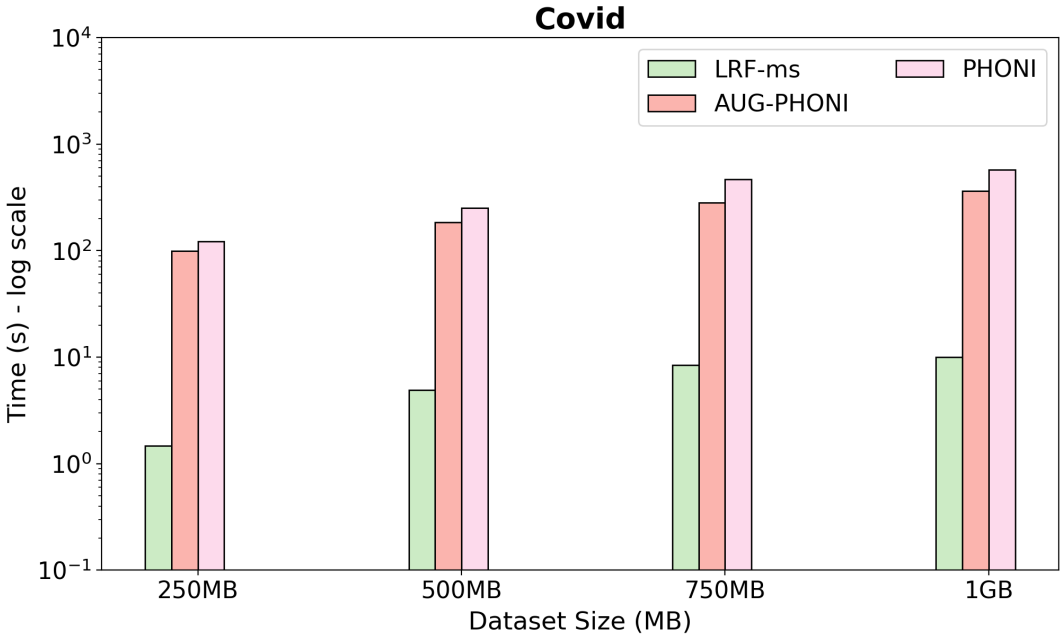
Parallel Software Comparison



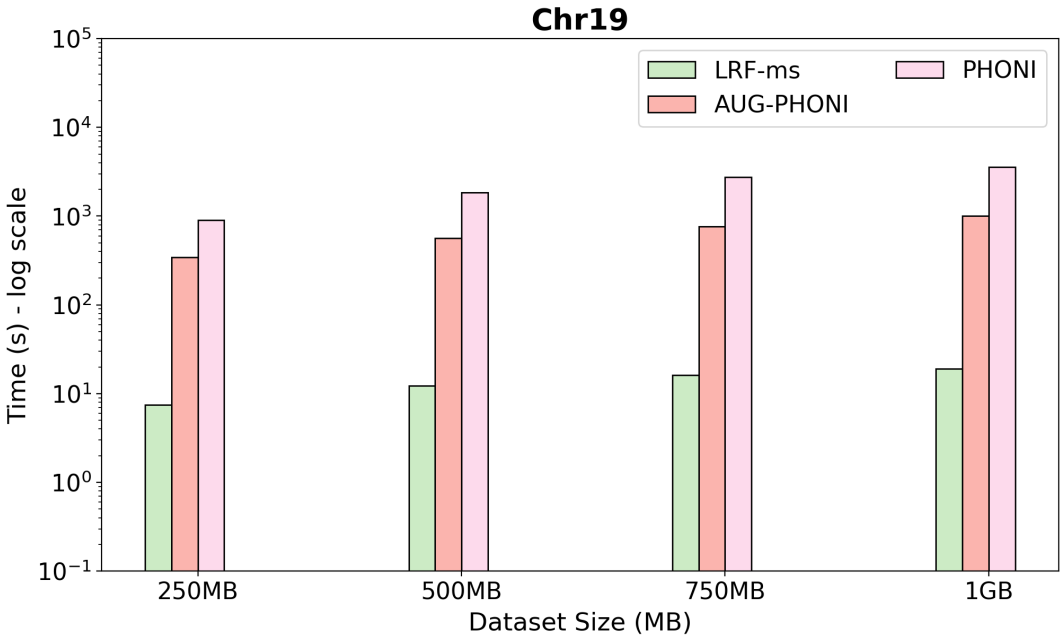
Parallel Software Comparison (continued)



Single Threaded Software Comparison



Single Threaded Software Comparison (continued)



Conclusions and Future Work

- presented a variant of an algorithm with a new heuristic (`LRF`)
- our implementation is highly competitive with other heuristics and widely known tools (`https://github.com/fmasillo/lrf-ms`)
- future work: apply LRF heuristic to a compressed index on R

Thank you for your attention!

Space Usage - Indexing Phase

	LRF-ms	indexed_ms	MONI	PHONI	AUG-PHONI
sars-cov2	**4 992**	13 488	15 052	14 524	14 640
chr19	**991 240**	1 003 788	1 428 540	1 428 536	1 428 536
salmonella	**80 380**	102 996	112 760	112 760	112 760
rice	727 924	**709 864**	1 039 352	1 039 344	1 039 348

Table: Peak memory (in KB) of the different tools for indexing the reference string R. The lowest value for each row is highlighted in bold.

Space Usage - Query Phase

	LRF-ms	indexed_ms	MONI	PHONI	AUG-PHONI
sars-cov2	90 716	617 820	17 044	**14 168**	14 180
chr19	1 077 920	**683 504**	13 078 520	2 175 716	2 393 404
salmonella	**166 552**	635 368	1 269 008	181 620	202 240
rice	814 900	**673 916**	10 859 492	1 628 136	1 816 560

Table: Peak memory (in KB) of the different tools for computing the matching statistics of the string collection w.r.t. R. The lowest value for each row is highlighted in bold.