Parallel Computation of Matching Statistics and Average Common Substring

Fabio Garofalo Daniele Greco²
Giovanna Rosone² Marinella Sciortino¹

¹Dipartimento di Matematica e Informatica Università di Palermo, Italy

> ²Dipartimento di Informatica Università di Pisa, Italy

BITS 2019 - Analysis of Big Omics Data June 26th, 2019

- Increased availability of large sets of biological sequences
- Tools for sequence comparison a fortiori need alignment-free based approaches
- Most alignment-free approaches require the computation of statistics when comparing sequences
- Such computations may not scale well in internal memory when very large collections of long sequences are considered

- Increased availability of large sets of biological sequences
- Tools for sequence comparison a fortiori need alignment-free based approaches
- Most alignment-free approaches require the computation of statistics when comparing sequences
- Such computations may not scale well in internal memory when very large collections of long sequences are considered
- Our contribution:

- Increased availability of large sets of biological sequences
- Tools for sequence comparison a fortiori need alignment-free based approaches
- Most alignment-free approaches require the computation of statistics when comparing sequences
- Such computations may not scale well in internal memory when very large collections of long sequences are considered
- Our contribution:
 - the Colored Longest Common Prefix array: a new conceptual data structure that uses an alignment-free approach and can be computed via sequential scans in semi-external memory

- Increased availability of large sets of biological sequences
- Tools for sequence comparison a fortiori need alignment-free based approaches
- Most alignment-free approaches require the computation of statistics when comparing sequences
- Such computations may not scale well in internal memory when very large collections of long sequences are considered
- Our contribution:
 - the Colored Longest Common Prefix array: a new conceptual data structure that uses an alignment-free approach and can be computed via sequential scans in semi-external memory
 - it implicitly stores all the information necessary to compute statistics on distinguishing, repeating, or matching substrings within collections of strings.

- Increased availability of large sets of biological sequences
- Tools for sequence comparison a fortiori need alignment-free based approaches
- Most alignment-free approaches require the computation of statistics when comparing sequences
- Such computations may not scale well in internal memory when very large collections of long sequences are considered
- Our contribution:
 - the Colored Longest Common Prefix array: a new conceptual data structure that uses an alignment-free approach and can be computed via sequential scans in semi-external memory
 - it implicitly stores all the information necessary to compute statistics on distinguishing, repeating, or matching substrings within collections of strings.
 - efficient lightweight strategy to solve the multi-string Average Common Substring (ACS) Problem and experimental results.

Preliminaries

- $\Sigma = \{c_1, c_2, \dots, c_{\sigma}\}$ be a finite ordered alphabet
- ullet ${\cal S}$ is a collection of m strings over Σ
- n_i is the length of the string s_i
- ullet A distinct end-marker symbol $\$_i < c_1$ is appended to each string s_i
- $N = \sum_{i=1}^{m} n_i + m$ is the length of the collection S
- Each string (or subset of strings) is identified by a specific color

Let $S = \{s_1, s_2, \dots, s_m\}$ be a collection of strings.

String collection ${\mathcal S}$							
	0	1	2	3	4	5	6
s ₁	G	С	С	Α	Α	С	\$1
s 2	G	Α	G	С	T	С	\$2
<i>5</i> 3	T	С	G	С	T	T	\$3

Let $S = \{s_1, s_2, \dots, s_m\}$ be a collection of strings.

• The extended Burrows-Wheeler Transform for a string collection \mathcal{S} , known as EBWT or multi-string BWT, is a reversible transformation that produces a string (denoted by $ebwt(\mathcal{S})$) that is a permutation of the characters of all strings in \mathcal{S}

String collection ${\cal S}$							
	0	1	2	3	4	5	6
<i>s</i> ₁	G	С	С	Α	Α	С	\$1
s 2	G	Α	G	С	T	С	\$2
<i>5</i> 3	T	С	G	С	T	T	\$3

Let $S = \{s_1, s_2, \dots, s_m\}$ be a collection of strings.

• The extended Burrows-Wheeler Transform for a string collection \mathcal{S} , known as EBWT or multi-string BWT, is a reversible transformation that produces a string (denoted by $ebwt(\mathcal{S})$) that is a permutation of the characters of all strings in \mathcal{S} • $ebwt(\mathcal{S})$ • Sorted Suffixes of \mathcal{S}

String collection ${\cal S}$							
	0	1	2	3	4	5	6
s ₁	G	С	С	Α	Α	С	\$1
s ₂	G	Α	G	С	T	С	\$2
5 3	T	С	G	С	T	T	\$3

• Sort all the suffixes of the strings in S;

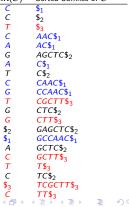
		lune 26th 2010	4 / 30
•	\$ ₃	TT\$3 4	199 (
	€ \$a	TCGCTT\$3	
	C	TC\$ ₂	
	T	GCTT\$ ₃ T\$ ₃	
	A C	GCTC\$2	
	\$ ₁	$GCCAAC$ $_1$	
	\$ 2	$GAGCTC\$_2$	
	G	CTT\$3	
	G	CTC\$ ₂	
	T	CGCTT\$3	
	G	CCAAC\$1	
	Ċ	CAAC\$ ₁	
	T	C\$2	
	A	C\$ ₁	
	G	AGCTC\$2	
	A	AC\$ ₁	
	Ċ	AAC\$ ₁	
	T	\$3	

Let $S = \{s_1, s_2, \dots, s_m\}$ be a collection of strings.

• The extended Burrows-Wheeler Transform for a string collection \mathcal{S} , known as EBWT or multi-string BWT, is a reversible transformation that produces a string (denoted by $ebwt(\mathcal{S})$) that is a permutation of the characters of all strings in \mathcal{S} $ebwt(\mathcal{S})$ Sorted Suffixes of \mathcal{S}

String collection ${\mathcal S}$							
	0	1	2	3	4	5	6
s ₁	G	С	С	Α	Α	С	\$ 1
s 2	G	Α	G	С	T	С	\$ ₂
<i>5</i> 3	T	С	G	С	T	T	\$3

- ullet Sort all the suffixes of the strings in ${\cal S}$;
- The output $\operatorname{ebwt}(\mathcal{S})$ is obtained by concatenating the symbols that (circularly) precede the first symbol of each suffix in the list of (lexicographically) sorted suffixes of \mathcal{S} .



LCP and Colored EBWT

• The longest common prefix (LCP) array of the collection S is the array lcp(S) of length N+1, such that lcp(S)[i], with $2 \le i \le N$, is the length of the longest common prefix between the suffixes associated to the positions i and i-1 in ebwt(S). By default, lcp(S)[1] = lcp(S)[N+1] = -1

$lcp(\mathcal{S})$	$id(\mathcal{S})$	$ebwt(\mathcal{S})$	Sorted Suffixes of ${\mathcal S}$
-1	1	С	\$ 1
0	2	C	\$ ₂
0	3	T	\$ ₃
0	1	C	AAC\$ ₁
1	1	Α	AC\$1
1	2	G	$AGCTC\$_2$
0	1	A	C\$ ₁
1	2	T	C\$ ₂
1	1	C	CAAC\$ ₁
1	1	G	CCAAC\$ ₁
1	3	T	$CGCTT\$_3$
1	2	G	CTC\$2
2	3	G	CTT\$3
0	2	\$ 2	GAGCTC\$2
1	1	\$ ₁	GCCAAC\$1
2	2	Ā	GCTC\$ ₂
3	3	C	$GCTT\$_3$
0	3	T	T\$3
1	2	С	TC\$ ₂
2	3	\$ ₃	TCGCTT\$3
1	3	Č	TT\$3

-1

LCP and Colored EBWT

- The longest common prefix (LCP) array of the collection S is the array lcp(S) of length N+1, such that lcp(S)[i], with $2 \le i \le N$, is the length of the longest common prefix between the suffixes associated to the positions i and i-1 in ebwt(S). By default, lcp(S)[1] = lcp(S)[N+1] = -1
- lcp(i,j) the length of the LCP between the suffixes at positions i and j, i.e. $min\{lcp(S)[l]: i < l < j\}$.

$lcp(\mathcal{S})$	$id(\mathcal{S})$	$ebwt(\mathcal{S})$	Sorted Suffixes of ${\cal S}$
-1	1	C	\$ ₁
0	2	C	\$ ₂
0	3	T	\$ ₃
0	1	C	AAC\$ ₁
1	1	A	AC\$ ₁
1	2	G	AGCTC\$2
0	1	A	C\$ ₁
1	2	T	C\$2
1	1	C	CAAC\$ ₁
1	1	G	CCAAC\$ ₁
1	3	T	CGCTT\$3
1	2	G	CTC\$2
2	3	G	$CTT\$_3$
0	2	\$ 2	GAGCTC\$2
1	1	\$ 1	GCCAAC\$1
2	2	Ā	GCTC\$2
3	3	C	GCTT\$3
0	3	T	T\$3
1	2	С	TC\$ ₂
2	3	\$2	TCGCTT\$3
1	3	\$ ₃	TT\$3
			. 3

-1

LCP and Colored EBWT

- The longest common prefix (LCP) array of the collection S is the array lcp(S) of length N+1, such that lcp(S)[i], with $2 \le i \le N$, is the length of the longest common prefix between the suffixes associated to the positions i and i-1 in ebwt(S). By default, lcp(S)[1] = lcp(S)[N+1] = -1
- lcp(i, j) the length of the LCP between the suffixes at positions i and j, i.e. $min\{lcp(S)[l]: i < l \le j\}$.
- The output string $\operatorname{ebwt}(\mathcal{S})$, $\operatorname{enhanced}$ with the N-integer array of colors $\operatorname{id}(\mathcal{S})$ where $\operatorname{id}(\mathcal{S})[i] = r$, with $1 \leq r \leq m$ and $1 \leq i \leq N$, if $\operatorname{ebwt}(\mathcal{S})[i]$ is a symbol of the string $s_r \in \mathcal{S}$, is called *colored* EBWT.

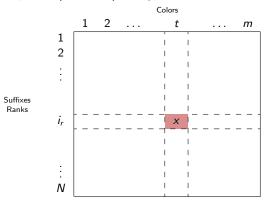
$lcp(\mathcal{S})$	id(S)	$ebwt(\mathcal{S})$	Sorted Suffixes of ${\mathcal S}$
-1	1	С	\$ ₁
0	2	С	\$ 2
0	3	T	\$ ₃
0	1	C	AAC\$ ₁
1	1	Α	AC\$ ₁
1	2	G	AGCTC\$ ₂
0	1	Α	C\$ ₁
1	2	T	C\$ ₂
1	1	C	CAAC\$ ₁
1	1	G	CCAAC\$ ₁
1	3	T	CGCTT\$3
1	2	G	CTC\$ ₂
2	3	G	CTT\$3
0	2	\$ 2	GAGCTC\$2
1	1	\$ 1	GCCAAC\$1
2	2	Α	$GCTC\$_2$
3	3	C	GCTT\$3
0	3	T	T\$3
1	2	\$3 C	TC\$2
2	3	\$ 3	TCGCTT\$3
1	3	Č	TT\$3
-1			•

Some recent lightweight implementations of EBWT and LCP

- Bauer, M., Cox, A., Rosone, G.: Lightweight algorithms for constructing and inverting the BWT of string collections. Theor. Comput. Sci. 483(0), 134–148 (2013) (known as BCR implementation)
- Cox, A.J., Garofalo, F., Rosone, G., Sciortino, M.: Lightweight LCP construction for very large collections of strings. J. Discrete Algorithms 37, 17–33 (2016)
- Egidi, L., Louza, F.A., Manzini, G., Telles, G.P.: External memory BWT and LCP computation for sequence collections with applications. WABI 2018
- Louza, F., Telles, G., Hoffmann, S., Ciferri, C.: Generalized enhanced suffix array construction in external memory. Algorithms Mol. Biol. 12(1), 26 (2017)

Colored Longest Common Prefix (cLCP) array

cLCP is an $(N \times m)$ -integer array representing the longest common prefix between any specific suffix of a (r-colored) string $s_r \in \mathcal{S}$ and the nearest suffixes of a specific (t-colored) string $s_t \in \mathcal{S}$ in the sorted list of suffixes of \mathcal{S} .



x = LCP value between the *r*-colored suffix of rank i_r and the nearest *t*-colored suffix in the sorted list of suffixes

Given $1 \le i_r \le N$ and t = 1, ... m, how cLCP $[i_r][t]$ is defined?

$$prev(i, t) = \max\{x \mid 1 \le x < i, id(\mathcal{S})[x] = t\}$$

$$next(i, t) = \min\{x \mid i < x \le N, id(\mathcal{S})[x] = t\}$$

i	$lcp(\mathcal{S})$	$id(\mathcal{S})$
:	:	
$prev(i_r, t)$	I_1	t
÷	:	Ţ ≠ t ⊥
i _r	1	r
÷	:	Ţ ≠ t ⊥
$next(i_r,t)$	I_2	t
:	i	i

Given $1 \le i_r \le N$ and t = 1, ... m, how cLCP $[i_r][t]$ is defined?

$$prev(i, t) = \max\{x \mid 1 \le x < i, id(\mathcal{S})[x] = t\}$$

$$next(i, t) = \min\{x \mid i < x \le N, id(\mathcal{S})[x] = t\}$$

i	$lcp(\mathcal{S})$	$id(\mathcal{S})$
:	:	:
$prev(i_r, t)$	I_1	t
:	÷	Ţ ≠ t ⊥
ir	1	r
:	:	Ţ ≠ t ⊥
$next(i_r,t)$	I_2	t
:	•	:

Upper Colored LCP

 $UcLCP[i_r][t] = LCP(prev(i_r, t), i_r)$

Given $1 \le i_r \le N$ and t = 1, ... m, how cLCP $[i_r][t]$ is defined?

$$prev(i, t) = \max\{x | 1 \le x < i, id(\mathcal{S})[x] = t\}$$

$$next(i, t) = \min\{x | i < x \le N, id(\mathcal{S})[x] = t\}$$

Lower Colored LCP $LcLCP[i_r][t] = LCP(i_r, next(i_r, t))$

Given $1 \le i_r \le N$ and t = 1, ... m, how cLCP $[i_r][t]$ is defined?

$$prev(i, t) = \max\{x \mid 1 \le x < i, id(\mathcal{S})[x] = t\}$$

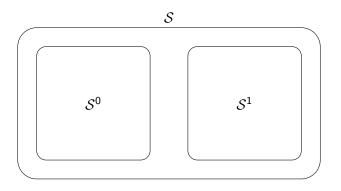
$$next(i, t) = \min\{x \mid i < x \le N, id(\mathcal{S})[x] = t\}$$

i	$lcp(\mathcal{S})$	$id(\mathcal{S})$
÷	:	:
$prev(i_r, t)$	I_1	t
:	:	Τ ≠ t 1
ir	1	r
÷	÷	Τ ≠ t ⊥
$next(i_r, t)$	<i>l</i> ₂	t
:	•	:

$$cLCP[i_r][t] = max{UcLCP[i_r][t], LcLCP[i_r][t]}$$

Colored LCP on disjoint collections

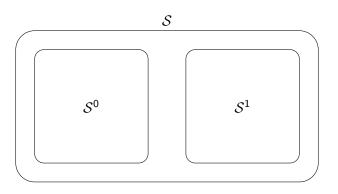
cLCP can also be defined for disjoint collections of strings.



The value cLCP[i_r][t] is defined for each pair (i_r , t) such that id(S)[i_r] = r, $t \in ID$ and s_r , s_t belongs to different collections.

Colored LCP on disjoint collections

cLCP can also be defined for disjoint collections of strings.



The value cLCP[i_r][t] is defined for each pair (i_r , t) such that id(\mathcal{S})[i_r] = r, $t \in \mathrm{ID}$ and s_r , s_t belongs to different collections.

Assume $\mathcal{S}^0 = \{s_\chi\}$ and $\mathcal{S}^1 = \mathcal{S} \setminus \{s_\chi\}$.

χ – intervals

A given string $s_\chi \in \mathcal{S}^0$ implicitly induces a partition of $lcp(\mathcal{S})$ into open intervals delimited by consecutive suffixes having color χ (or the positions 1 and N+1 of lcp), called χ -intervals. Let us consider a position i_r contained within a χ -interval such that $id[i_r] = r$ and $s_r \in \mathcal{S}^1$.

i	$lcp(\mathcal{S})$	$id(\mathcal{S})$
<u>:</u>	:	:
$\chi_1 = \operatorname{prev}(i_r, \chi)$		χ
: <i>i</i> _r :	:	$ \begin{array}{c} \top \\ \neq x \\ \bot \\ r \\ \top \\ \neq x \end{array} $
$\chi_2 = next(i_r, \chi)$		$\begin{array}{c c} & \bot \\ & \chi \end{array}$
:	÷	:

χ – intervals

A given string $s_\chi \in \mathcal{S}^0$ implicitly induces a partition of $lcp(\mathcal{S})$ into open intervals delimited by consecutive suffixes having color χ (or the positions 1 and N+1 of lcp), called χ -intervals. Let us consider a position i_r contained within a χ -interval such that $id[i_r] = r$ and $s_r \in \mathcal{S}^1$.

i	$lcp(\mathcal{S})$	$id(\mathcal{S})$
<u>:</u>	:	:
$\chi_1 = prev(i_r, \chi)$		χ
: i _r	:	
:	:	≠ x
$\chi_2 = \operatorname{next}(i_r, \chi)$		χ
i i	:	:

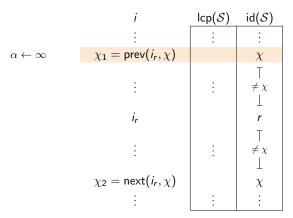
Inside a χ -interval, how to compute UcLCP[i_r][χ] and LcLCP[i_r][χ] by a lightweight strategy?

$UcLCP[i_r][\chi]$ Computation

ightharpoonup Keep track of the minimum lcp value since the beginning of each χ -interval.

$$UcLCP[i_r][\chi] = LCP(\chi_1, i_r) = \min\{lcp[x] : x \in (\chi_1, i_r]\} = \alpha$$

.

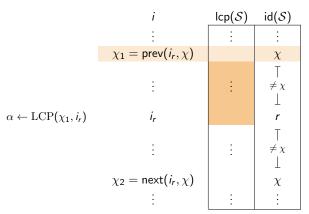


$UcLCP[i_r][\chi]$ Computation

ightharpoonup Keep track of the minimum lcp value since the beginning of each χ -interval.

$$UcLCP[i_r][\chi] = LCP(\chi_1, i_r) = \min\{lcp[x] : x \in (\chi_1, i_r]\} = \alpha$$

.



$$s_{\chi} = ACGCGCC\$_{\chi}$$
,

$$s_1 = ACGAGACGAT \$_1$$

$$s_{\chi} = ACGCGCC\$_{\chi}, \quad s_{1} = ACGAGACGAT\$_{1}, \quad s_{2} = AACGCCGCCGGCA\$_{2}$$

							ı	JcLCI	,	LcLCP		
	id	S	ebwt					1	_	ECEC	cLCP	Sorted suffixes of S
#	_		C	lcp -1	lcp _χ	α	χ	1	2			
1	χ	1	T		-1							şχ
2	2	0	l 'A	0								\$ ₁
		0	Ĉ	0								\$2
4 5	2	0	\$ ₂	1								A \$2 A A C G C C G C C G G C A \$
6	1	0	\$ ₁	1								A A C G C C G C C G G C A \$ A C G A G A C G A T \$1
7	1	0	G G	4								A C G A T \$1
8	2	0	A	3								ACGCCGGCA\$2
9		1	\$0	4	0							A C G C G C C \$x
10	χ	0	G	1	"							$\begin{array}{c ccccccccccccccccccccccccccccccccccc$
11	i	0	Ğ	i								A T \$1
12	x	1	Č	Ô	0							C S _x
13	2	0	G	1	-							C A \$2
14	χ	1	G	1	1	∞						
15	2	0	G	2								C C \$\x\ C C G C A \x\2
16	2	0	G	3								C C G G C A \$ ₂
17	1	0	A	1								CGAGACGAT\$1
18	1	0	A	3								C G A T \$1
19	χ	1	G	2	1							C G C C S _V
20	2	0	A	4								C G C C G C C A \$2
21	2	0	C	5								C G C C G G C A \$2
22	χ	1	Α	3	3							C G C C C S _X
23	2	0	С	2								C G G C A \$2
24	1	0	A	0	l		1	1	ı			GACGAT \$1
25	1	0	C	2	l		1	1	ı			G A G A C G A T \$1
26	1	0	C	2	l		1	1	ı			G A T \$1
27	2	0	G	1								G C A \$2
28	χ	1	С	2	0							G C C S _X
29	2	0	C	3								G C C G G C A \$2
30	2	0	C	4	L .							G C C G G C A \$2
31	χ	1	C	2	2							G C G C C \$\chi_{\chi}
32	2	0	C	1	l		1	1	ı			G G C A \$2 T \$1
33	1	0	A	0	١.		1	1	ı			1 \$1
		1	I	-1	-1	1	1	1		1		

$$s_{\chi} = ACGCGCC\$_{\chi}$$
,

$$s_1 = ACGAGACGAT \$_1$$

$$s_{\chi} = ACGCGCC\$_{\chi}, \quad s_{1} = ACGAGACGAT\$_{1}, \quad s_{2} = AACGCCGCCGGCA\$_{2}$$

							Ιι	JcLCI	>	LcLCP	cLCP	
#	id	S	ebwt	lcp	lcp_{χ}	α	_	1	2			Sorted suffixes of S
1	_	1	C	-1	-1	ш	χ	1	-			\mathbf{s}_{χ}
2	χ	0	T	0	-1							*χ \$1
3	2	0	l 'A	ŏ								\$ ₂
4	2	0	Ĉ	ő								A \$2
5	2	0	\$2	1								A A C G C C G C C G G C A
6	ī	0	\$1	i								A C G A G A C G A T \$1
7	ī	ő	Ĝ	4								A C G A T \$1
8	2	0	A	3								ACGCCGCCGGCA\$2
9	χ	1	\$0	4	0							A C G C G C C \$x
10	1	0	Ğ	1								A G A C G A T \$1
11	1	0	G	1					ı			A T \$1
12	χ	1	С	0	0							C \$ _{\chi}
13	2	0	G	1								C A \$2
14	χ	1	G	1	1	∞						C C \$\(\sigma \) C C G C A \$\(\sigma \)
15	2	0	G	2		2	2					C C G C C G G C A \$2
16	2	0	G	3								C C G G C A \$2
17	1	0	A	1								CGAGACGAT\$1
18	1	0	A	3								C G A T \$1
19	χ	1	G	2	1							$ \begin{array}{c ccccccccccccccccccccccccccccccccccc$
20	2	0	A	4								
21	2	0	С	5								C G C C G G C A \$2
22	χ	1	Α	3	3							C G C G C C S _X
23	2	0	C	2					ı			C G G C A \$2
24	1	0	A	0								G A C G A T \$1 G A G A C G A T \$1
25	1	0	C	2					ı			
26 27	1	0	G	2								G A T \$ ₁ G C A \$ ₂
28	2	1	C	2	0							
29	2	0	c	3	U							$ \begin{array}{c ccccccccccccccccccccccccccccccccccc$
30	2	0	c	4					ı			G C C G G C A \$2
31		1	c	2	2							G C G C C \$x
32	2	0	c	1								G C G C C \$\chi_X
33	1	0	Ā	ō					ı			G G C A \$2 T \$1
""	*	ا ا	l	-1	-1							• ••

$$s_{\chi} = ACGCGCC\$_{\chi}$$
,

$$s_1 = ACGAGACGAT$$

$$s_{\chi} = ACGCGCC\$_{\chi}, \quad s_{1} = ACGAGACGAT\$_{1}, \quad s_{2} = AACGCCGCCGGCA\$_{2}$$

_												
							Ų	JcLCI	P	LcLCP	cLCP	
#	id	S	ebwt	lcp	lcp_{χ}	α	χ	1	2			Sorted suffixes of S
1	χ	1	С	-1	-1							\$ _{\chi}
2	1	0	T	0								\$ ₁
3	2	0	A	0								\$ ₂
4	2	0	C	0								A \$2
5	2	0	\$ ₂	1								AACGCCGCCGGCAS
6	1	0	\$ ₁	1								ACGAGACGAT \$1
7	1	0	G	4								A C G A T \$1 A C G C C G C C G G C A \$2
8	2	0	A	3								
9	χ	1	\$0	4	0							A C G C G C C \$ _{\chi}
10	1	0	G	1								AGACGAT\$1
11	1	0	G	1	l		1	1				A T \$1
12	χ	1	C	0	0							C \$ _{\chi} C A \$ ₂
13	2	0	G	1								C A \$2
14	χ	1	G	1	1	∞						C C \$\x\ C C G C A \x\2
15	2	0	G	2		2	2					C C G C C G G C A \$2
16	2	0	G	3		2	2					C C G G C A \$2 C G A G A C G A T \$1
17	1	0	A	1								
18	1	0	A	3								C G A T \$1
19	χ	1	G	2	1							$ \begin{array}{c ccccccccccccccccccccccccccccccccccc$
20	2	0	A	4								C G C C G C C A \$2
21	2	0	C	5								C G C C G G C A \$2
22	χ	1	Α	3	3							C G C C C S _X
23	2	0	С	2								C G G C A \$2
24	1	0	A	0	l		1	1				GACGAT\$1
25	1	0	C	2	l		1	1				G A G A C G A T \$1
26	1	0	C	2								G A T \$1
27	2	0	G	1	l		1	1				G C A \$2
28	χ	1	С	2	0							G C C S _X
29	2	0	С	3								G C C G C A \$2
30	2	0	C	4	l		1	1				G C C G G C A \$2
31	χ	1	С	2	2							G C G C C S _X
32	2	0	C	1								G G C A \$2
33	1	0	A	0	l		1	1				T \$1
				-1	-1	l	1	l				

$$s_{\chi} = ACGCGCC\$_{\chi}$$
,

$$s_1 = ACGAGACGAT \$_1$$

$$s_{\chi} = ACGCGCC\$_{\chi}, \quad s_{1} = ACGAGACGAT\$_{1}, \quad s_{2} = AACGCCGCCGGCA\$_{2}$$

							Π.	JcLCI	_	LcLCP	cLCP	
			_	_	_	_	-	_		LCLCP	CLCP	
#	id	S	ebwt	lcp	lcp_{χ}	α	χ	1	2			Sorted suffixes of $\mathcal S$
1	χ	1	С	-1	-1							$\$_{\chi}$
2	1	0	T	0								\$ ₁
3	2	0	A	0								<u>\$2</u>
4	2	0	C	0								A \$2
5	2	0	\$2	1								A A C G C C G C C G G C A S
6	1	0	\$ ₁	1								A C G A G A C G A T \$1
7	1	0	G	4								A C G A T \$1
8	2	0	A	3	L .							ACGCCGCCGGCA\$2
9	χ	1	\$0	4	0							A C G C G C C \$x
10	1	0	G	1	l							$\begin{array}{c ccccccccccccccccccccccccccccccccccc$
11	1	0	G	1	_							A T \$1
12	χ	1	C	0	0							C \$ _{\chi}
13	2	0	G	1	-							C A \$2
14 15	2	0	G	2	1	2	2					
16		0	G	3		2	2					
17	2	0	A	1		1	1					C C G G C A \$2 C G A G A C G A T \$1
18	1	0	A	3		1						C G A T \$1
19		1	G	2	1							
20	2	0	A	4	-							
21	2	0	l ĉ	5								
22	X	1	A	3	3							C G C G C C \$x
23	2	0	Ĉ	2	,							C G G C A \$2
24	1	ō	Ā	l -								G A C G A T \$1
25	1	ō	C	2	l							G A G A C G A T \$1
26	1	ō	l č	2	l							G A T \$1
27	2	ō	Ğ	l ī	l							G C A \$2
28	χ	1	c	2	0							G C C \$\sum_\chi\$
29	2	0	С	3								GCCGCCGGCA\$2
30	2	0	C	4								G C C G G C A \$2
31	X	1	С	2	2							
32	2	0	C	1								G G C A \$2
33	1	0	A	0	l							T \$1
			I	_1	_1	I	I	I	1	1		

$$s_{\chi} = ACGCGCC\$_{\chi}$$
,

$$s_1 = ACGAGACGAT \$_1$$

$$s_{\chi} = ACGCGCC\$_{\chi}, \quad s_{1} = ACGAGACGAT\$_{1}, \quad s_{2} = AACGCCGCCGGCA\$_{2}$$

								JcLCI	<u> </u>	LcLCP	cLCP	
		_			_	_		JCLU		LCLCP	CLCP	
#	id	S	ebwt	lcp	lcp_{χ}	α	χ	1	2			Sorted suffixes of S
1	χ	1	С	-1	-1							$\$_{\chi}$
2	1	0	T	0								\$ ₁
3	2	0	A	0								<u>\$2</u>
4	2	0	C	0								A \$2
5	2	0	\$2	1								A A C G C C G C C G C A
6	1	0	\$1	1								ACGAGACGAT\$1
7	1	0	G	4								A C G A T \$1
8	2	0	A	3	L.							ACGCCGCCGCA\$2
9	χ	1	\$0	4	0							A C G C G C C \$x
10	1	0	G	1			1					$\begin{array}{c ccccccccccccccccccccccccccccccccccc$
11	1	0	G	1								A T \$1
12	X	1	C	0	0							C S _X
13	2	0	G	1	1							C A \$2
15	2	0	G	2	1	2	2					C C \$\sqrt{C} C G C A \$\sqrt{2}
16	2	0	G	3		2	2					
17	1	0	A	1		1	1					C G A G A C G A T \$1
18	1	0	Â	3		1	1					
19	X	1	Ĝ	2	1	-	-					$\begin{bmatrix} c & G & C & C \end{bmatrix}$
20	2	0	A	4	-							
21	2	ō	C	5								
22	χ	1	A	3	3							C G C G C C \$x
23	2	0	С	2								C G G C A \$2
24	1	0	A	0								GACGAT\$1
25	1	0	C	2			1					G A G A C G A T \$1
26	1	0	C	2			1					G A T \$1
27	2	0	G	1								G C A \$2
28	χ	1	С	2	0							G C C \$\chi_{\chi}
29	2	0	С	3								GCCGCCGGCA\$2
30	2	0	C	4			1					G C C G G C A \$2
31	χ	1	С	2	2							G C G C C \$χ G G C A \$2
32	2	0	C	1			1					G G C A \$2
33	1	0	A	0			1					<u>T</u> \$1
				_1	_1							

$$s_{\chi} = ACGCGCC\$_{\chi}$$
,

$$s_1 = ACGAGACGAT$$
\$₁

$$s_{\chi} = ACGCGCC\$_{\chi}, \quad s_{1} = ACGAGACGAT\$_{1}, \quad s_{2} = AACGCCGCCGGCA\$_{2}$$

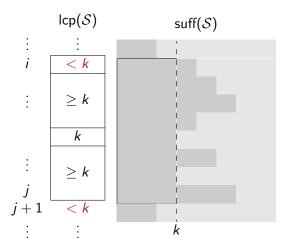
UcLCP LcLCP cLCP																										
#	id	S	ebwt	lcp	lcp_{χ}	α	χ	1	2			Sc	orted	suffix	es of a	s										
1	χ	1	С	-1	-1	∞							\$ χ													
2	1	0	T	0		0	0						\$1													
3	2	0	A	0		0	0						\$2													
4	2	0	C	0		0	0					ΙГ	Α	\$2												
5	2	0	\$ ₂	1		0	0						Α	A	С	G	С	С	G	С	С	G	G	С	Α	\$2
6	1	0	\$ ₁	1		0	0					l	Α	С	G	A	G	Α	С	G	Α	T	\$1			
7	1	0	G	4		0	0						Α	C	G	A	T	\$ 1								
8	2	0	A	3		0	0					l	Α	С	G	C	C	G	C	С	G	G	С	Α	\$2	
9	χ	1	\$0	4	0	∞							Α	C	G	C	G	С	С	\$ _x						
10	1	0	G	1		1	1						Α	G	Α	С	G	Α	T	\$1						
11	1	0	G	1		1	1						Α	T	\$1											
12	χ	1	С	0	0	∞							С	\$ _{\chi}												
13	2	0	G	1		1	1						C	Â	\$2											
14	χ	1	G	1	1	∞							С	C	\$ _{\chi}											
15	2	0	G	2		2	2						C	C	G	С	С	G	G	С	Α	\$ 2				
16	2	0	G	3		2	2						C	C	G	G	С	Α	\$ ₂ G							
17	1	0	A	1		1	1					lF	C	G	Α	G	Α	С	G	Α	T	\$ 1				
18	1	0	A	3		1	1						C	G	A	T	\$ 1									
19	χ	1	G	2	1	∞							С	G	С	С	\$ _{\chi}									
20	2	0	A	4		4	4						C	G	C	C	G	С	С	G	G	С	Α	\$ ₂		
21	2	0	C	5		4	4						C	G	C	C	G	G	C	Α	\$2					
22	χ	1	Α	3	3	∞							С	G	С	G	С	С	\$ χ							
23	2	0	C	2		2	2						С	G	G	C	Α	\$ 2								
24	1	0	A	0		0	0	1				ΙГ	G	A	C	G	Α	T	\$ ₁							
25	1	0	C	2		0	0						G	A	G	Α	С	G	Α	T	\$1					
26	1	0	C	2		0	0						G	A	T	\$1										
27	2	0	G	1		0	0	1					G	С	A	\$2										
28	χ	1	С	2	0	∞							G	С	С	\$ χ										
29	2	0	C	3		3	3						G	C	C	G	С	С	G	G	С	Α	\$ 2			
30	2	0	C	4		3	3	1				L	G	C	C	G	G	С	Α	\$ 2						
31	χ	1	С	2	2	∞							G	С	G	С	С	\$ χ								
32	2	0	C	1		1	1						G	G	C	Α	\$ 2									
33	1	0	A	0		0	0	1				"	T	\$1												
		ı	1	_1	_1	I	1	ı	I	1																

$LcLCP[i_r][\chi]$ Computation

Problem: LcLCP[i_r][χ] computation would require to look forward and to store many intermediate values.

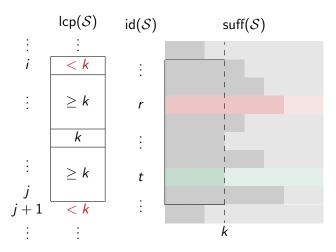
Colored k-lcp interval

A k-lcp interval is an interval [i, j] such that:



Colored k-lcp interval

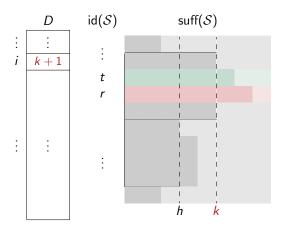
A colored k-lcp interval is an interval [i, j] such that:



 s_r and s_t belong one to \mathcal{S}^0 , the other to \mathcal{S}^1

Array D

Denote with D the (N+1)-integer array such that D[i]=k+1 if a colored k-lcp interval starts at position i and for every colored k-lcp interval starting at position i then $k \leq k-1$.



LcLCP[i_r][χ] Computation

Theorem

For any $1 \leq i_r \leq N$ such that $id(S)[i_r] = r$,

- if $LCP(\chi_1, i_r) > LCP(\chi_1, \chi_2)$ then $LCP(i_r, \chi_2) = LCP(\chi_1, \chi_2)$
- otherwise

$$LCP(i_r, \chi_2) = \max\{\max\{D[x] : \chi_1 < x \le i_r\} - 1, LCP(\chi_1, \chi_2)\}$$

LcLCP[i_r][χ] Computation

Theorem

For any $1 \leq i_r \leq N$ such that $id(S)[i_r] = r$,

- if LCP $(\chi_1, i_r) > \text{LCP}(\chi_1, \chi_2)$ then LCP $(i_r, \chi_2) = \text{LCP}(\chi_1, \chi_2)$
- otherwise

$$LCP(i_r, \chi_2) = \max\{\max\{D[x] : \chi_1 < x \le i_r\} - 1, LCP(\chi_1, \chi_2)\}$$

ho Keep track of the maximum D value since the beginning of each χ -interval.

track of th	ie maximum D v	aiue sin	ce the	beginni
	i	$lcp(\mathcal{S})$	D	$id(\mathcal{S})$
	:	:	:	:
$\zeta \leftarrow 0$	$\chi_1 = \operatorname{prev}(i_r, \chi)$			χ
	:	:	:	Ţ ≠ χ 1
	i_r			<i>r</i> т
	i:	:	:	≠ x 1
	$\chi_2 = \operatorname{next}(i_r, \chi)$			χ
	:	:	:	:

LcLCP[i_r][χ] Computation

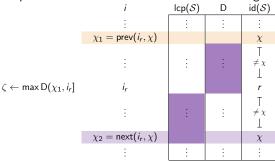
Theorem

For any $1 \leq i_r \leq N$ such that $id(S)[i_r] = r$,

- if LCP $(\chi_1, i_r) > \text{LCP}(\chi_1, \chi_2)$ then LCP $(i_r, \chi_2) = \text{LCP}(\chi_1, \chi_2)$
- otherwise

$$LCP(i_r, \chi_2) = \max\{\max\{D[x] : \chi_1 < x \le i_r\} - 1, LCP(\chi_1, \chi_2)\}$$

> Keep track of the maximum D value since the beginning of each χ -interval.



										JcLCI		г.	LcLC	n .	cLCP	
					-		_	Ι.			_		_	_	CLCP	
#	id	S	ebwt	lcp	D	lcp_{χ}	α	ς	χ	1	2	χ	1	2		Sorted suffixes of S
1	χ	1	С	-1	0	-1	∞									\mathbf{s}_{χ}
2	1	0	T	0	0		0		0							\$ ₁
3	2	0	A	0	0		0		0							\$2
4	2	0	C	0	2		0		0							A \$2
5	2	0	\$ ₂	1	0		0		0							A C G C C G C C G G C A \$2
	1	0	\$1	1	4		0		0							
7	1	0	G	4	0		0		0							
9	2	0	A	3	5	0	0		0							
10	χ 1	0	\$ ₀	1	0	U	∞		1							A C G C G C S _χ A G A C G A T S ₁
11	1	0	G	1	0		1		1							A T \$1
12		1	c	0	2	0	00		1							
13	χ 2	0	G	1	0	0	1		1							C \$χ C A \$2
14	χ	1	G	1	3	1	00		1							$\begin{vmatrix} c & c & s_{\chi} \end{vmatrix}$
15	2	0	G	2	0	-	2		2							CCGGCA\$
16	2	ō	Ğ	3	ő		2		2							C C G G C A \$2
17	1	0	A	1	3		1		1							CGAGACGATS
18	1	ō	A	3	ő		1		1							C G A T \$1
19	χ	1	G	2	5	1	∞									C G C C S _X
20	2	0	A	4	0		4		4							
21	2	0	C	5	0		4		4							C G C C G G C A \$2
22	χ	1	Α	3	0	3	∞									C G C G C C S _{\chi}
23	2	0	С	2	0		2		2							C G G C A \$2
24	1	0	A	0	2		0		0							G A C G A T \$1
25	1	0	C	2	0		0		0							G A G A C G A T \$1
26	1	0	C	2	0		0		0							G A T \$1
27	2	0	G	1	3		0		0							G C A \$2
28	χ	1	C	2	4	0	∞									G C C \$ _{\chi}
29	2	0	C .	3	0		3		3							GCCGGCA\$2
30	2	0	C	4	0		3		3							GCCGGCA\$2
31	X	1	C	2	0	2	000		١.							G C G C C \$x
32	2	0	С	1	0		1		1							G G C A \$2
33	1	0	A	0 -1	0	1	0	1	0	l			I	1	I	T \$1

										Jel Cl	_		LcLCI	_	cl CP	
					D	Ι.		٠.	-					_	CLCF	Sorted suffixes of S
#	id	S	ebwt	lcp		lcp_{χ}	α	5	χ	1	2	χ	1	2		
1	χ	1	С	-1	0	-1	∞									\mathbf{s}_{χ}
2	1	0	T	0	0		0		0							\$ ₁
3	2	0	A	0	0		0		0							\$2
4	2	0	C	0	2		0		0							A \$2
5	2	0	\$ ₂	1	0		0		0							A A C G C C G C C G G C A \$2
6	1	0	\$ ₁	1	4		0		0							
7	1	0	G	4	0		0		0							A C G C G G C A \$2
8	2	0	A	3	5		0		0							
9	χ	1	\$0	4	0	0	∞									A C G C G C C \$\chi_{\chi}
10	1	0	G	1	0		1		1		ı					A G A C G A T \$1 A T \$1
11	1	0		1	0		1		1							
12	χ	1	C	0	2	0	∞									C S _X
13	2	0	G	1	0		1		1							C A \$2
14	χ	1	G	1	3	1	~	0	_							
15	2	0	G	2	0		2 2		2							
16	2	0	G		0				2							
17	1	0	A	1	3		1		1							C G A G A C G A T \$1
18	1	0	G	3	0		1		1							C G A T \$1
19 20	2	0			5	1	∞									$\begin{array}{c ccccccccccccccccccccccccccccccccccc$
21	2	0	A C	4 5	0		4		4							
22				3		3										
23	χ	0	A	2	0	3	2		2							
24	2	0	C A	0	2		0		0		ı					
25	1	0	Ĉ	2	0		0		0		ı					G A C G A T \$ ₁ G A G A C G A T \$ ₁
26	1	0	6	2	0		0		0							G A T \$1
27	2	0	G	1	3		0		0		ı					G C A \$2
28		1	c	2	4	0			,							
29	2	0	c	3	0	U	3		3							$\begin{array}{c ccccccccccccccccccccccccccccccccccc$
30	2	0	l c	4	0		3		3							
31		1	c	2	0	2	00		3							G C G C C \$x
32	2	0	c	1	0	_	1		1							G G C A \$2
33	1	lő	Ā	0	0		0		0							T \$1
55	*	ľ	_ ^	-1	ľ	-1	ľ	I	ľ	l	1		l	l	l	. 41

									Π,	lcl Cl			LcLC	n	cl CP	
#	id	S	ebwt	1.	р	T	T		<u> </u>	1	2			2	CLCF	Sorted suffixes of S
	_		C	lcp -1	0	Icp _x	α	ς.	χ	1	-2	χ	1			
1	χ	1				-1	∞									\$ _{\chi}
2	1	0	T	0	0		0		0							\$ 1
3	2	0	A	0	0		0		0							\$2
5	2	0	\$ ₂	0	2		0		0							A S ₂ A A C G C C G C C G G C A S ₁
6	1	0	\$ ₁	1	4		0		0							A A C G C C G C C G G C A \$2
7	1	0	G G	4	0		0		0							A C G A T \$1
8	2	0	A	3	5		0		0							$\begin{array}{c ccccccccccccccccccccccccccccccccccc$
9		1	\$0	4	0	0	-		U							A C G C G C C S _X
10	χ 1	0	₽0 G	1	0	0	∞		1							$\begin{array}{c ccccccccccccccccccccccccccccccccccc$
11	1	0	G	1	l ŏ		1		1		ı			l		A T \$1
12		1	c	0	2	0	000		-							C S _X
13	χ 2	0	G	1	0	0	1		1							C A \$2
14	χ	1	G	1	3	1	000	0	-							C C \$χ
15	2	0	G	2	0		2	0	2			1				CCGCCGCA\$
16	2	ō	Ğ	3	0	1	2	-	2							C C G G C A \$2
17	1	0	A	1	3		1		1							CGAGACGAT \$1
18	1	ō	A	3	ő		1		1							C G A T \$1
19	χ	1	G	2	5	1	∞									C G C C S _X
20	2	0	A	4	0		4		4							
21	2	0	C	5	0		4		4							C G C C G G C A \$2
22	χ	1	Α	3	0	3	∞									C G C G C C S _X
23	2	0	С	2	0		2		2							C G G C A \$2
24	1	0	A	0	2		0		0							G A C G A T \$1
25	1	0	C	2	0		0		0							G A G A C G A T \$1
26	1	0	C	2	0		0		0							G A T \$1
27	2	0	G	1	3		0		0		ı			l		G C A \$2
28	χ	1	С	2	4	0	∞									G C C \$χ
29	2	0	С	3	0		3		3							G C C G C C G G C A \$2
30	2	0	C	4	0		3		3							G C C G G C A \$2
31	χ	1	C	2	0	2	∞									G C G C C \$x
32	2	0	C	1	0		1		1		ı			l		G G C A \$2
33	1	0	A	0	0	l .	0		0							T \$1
				-1	1	-1	1	1	l		1		l	l		

										lcl Cl					1.00	
-	_	_	_		_		_	_		JcLCI			LcLCI	_	cLCP	
#	id	S	ebwt	lcp	D	lcp_{χ}	α	ς	χ	1	2	χ	1	2		Sorted suffixes of S
1	χ	1	С	-1	0	-1	∞									\$ _X
2	1	0	T	0	0		0		0							\$ ₁
3	2	0	A A	0	0		0		0							\$2
4 5	2 2	0	\$ ₂	0	2		0		0							A S ₂ A A C G C C G C C G G C A S ₂
6	1	0	\$ ₁	1	4		0		0							A A C G C C G C C G G C A \$2
7	1	0	G	4	0		0		ő							$\begin{vmatrix} A & C & G & A & T & \$_1 \end{vmatrix}$
8	2	0	l ă	3	5		0		lő							A C G C C G G C A \$2
9	X	1	\$0	4	0	0	∞									A C G C G C C S _X
10	1	0	G	1	0		1		1							A G A C G A T \$1
11	1	0	G	1	0		1		1							A T \$1
12	X	1	C	0	2	0	∞									C \$ _{\chi}
13	2	0	G	1	0		1		1							C A \$2
14	χ	1	G	1	3	1	∞	0								C C \$ _{\chi}
15	2	0	G	2	0		2	0	2			1				C C G C C G G C A \$2
16	2	0	G	3	0		2	0	2			1				C C G G C A \$2
17 18	1	0	A A	1 3	3		1 1		1							C G A G A C G A T \$1 C G A T \$1
19	1	1	G	2	5	1	∞		1							C G C S _X
20	2	0	A	4	0		4		4							
21	2	0	Ιĉ	5	ő		4		4							
22	χ	1	A	3	0	3	∞									C G C G C C S _X
23	2	0	С	2	0		2		2							C G G C A \$2
24	1	0	A	0	2		0		0							G A C G A T \$1
25	1	0	C	2	0		0		0							G A G A C G A T \$1
26	1	0	C .	2	0		0		0							G A T \$1
27	2	0	G	1	3		0		0							G C A \$2
28	X	1	С	2	4	0	∞									G C C S _X
29 30	2 2	0	c	3	0		3		3							G C C G C C G G C A \$2 G C C G G C A \$2
31	_	1	c	2	0	2	∞		3							G C G C C \$x
32	2	0	C	1	0		1		1							$G G G C A S_2$
33	1	0	Ā	0	0		0		ō							T \$1
33	1	"	l .,	-1	~	-1	ľ		ا							. **

									Π.	JcLCI	_		LcLC	n	cLCP	
					_	Ι.		٠.			_			_	CLCP	0.1.7
#	id	S	ebwt	lcp	D	lcp_{χ}	α	ς	χ	1	2	χ	1	2		Sorted suffixes of S
1	χ	1	С	-1	0	-1	∞									\mathbf{s}_{χ}
2	1	0	T	0	0		0		0							\$ 1
3	2	0	A	0	0		0		0							\$2
4	2	0	C	0	2		0		0							A \$2
5	2	0	\$2	1	0		0		0							AACGCCGCCGGCA\$2
6	1	0	\$1	1	4		0	1	0							A C G A G A C G A T \$1
7	1	0	G	4	0		0		0							A C G A T \$1
8	2	0	A	3	5		0		0							A C G C G C C G G C A \$2
9	χ	0	\$ ₀		0	0	00		1							
11	1	0	G	1	0		1	1	1		ı					A G A C G A T \$1 A T \$1
12		1	C	0	2	0			1							
13	χ 2	0	G	1	0	U	∞		1							C \$χ C A \$2
14		1	G	1	3	1	000	0	1							
15	χ 2	0	G	2	0	-	2	0	2			1				C C \$\chi_{\chi} C C G C A \$\(\frac{1}{2}\)
16	2	lő	G	3	0		2	0	2			1				
17	1	ő	Ā	1	3		1	2	1			2				C G A G A C G A T \$1
18	1	lő	Â	3	0		1	-	i			-				
19	χ	1	Ĝ	2	5	1	00		1							c G C c s _x
20	2	0	A	4	0	-	4		4							
21	2	ő	C	5	ő		4		4							
22	χ	1	A	3	0	3	000									C G C G C C \$x
23	2	0	Ĉ	2	0		2		2							C G G C A \$2
24	1	ő	Ā	0	2		0	1	0		ı					$G A C G A T $_1$
25	1	ō	c	2	0		ő	1	ō							G A G A C G A T \$1
26	1	0	c	2	0		0	1	0		ı					G A T \$1
27	2	0	G	1	3		0	1	0							G C A \$2
28	χ	1	С	2	4	0	∞									G C C S
29	2	0	С	3	0		3		3							GCCGGCA\$2
30	2	0	C	4	0		3	1	3		ı					G C C G G C A \$2
31	χ	1	С	2	0	2	∞									G C G C C \$x
32	2	0	С	1	0		1		1							G G C A \$2
33	1	0	A	0	0		0	1	0							T \$1
				-1	ı	-1	1	1	1		1		l			

									Π.	Jcl Cl			LcLC	n	cl CP	
	id	s	ebwt	1.	D	L	Г .							2	CLCF	Sorted suffixes of S
#	_			lcp		Icp _x	α	5	χ	1	2	χ	1	2		
1	χ	1	C	-1	0	-1	∞									\$ _{\chi}
2	1	0	T	0	0		0		0							\$ 1
3	2	0	A	0	0		0		0							\$2
4	2	0	C	0	2		0		0							A A C G C C G C C G G C A \$2
5 6	2	0	\$ ₂	1	0		0		0							
			\$1		4											
7 8	1	0	G	3	0		0		0							
9	2	0	A	4	5	0	-		U							
10	χ	0	\$ ₀	1	0	0	∞		1							A C G C G C S _χ A G A C G A T \$ ₁
11	1	0	G	1	0		1		1							A G A C G A 7 \$1
12		1	c	0	2	0			1							
13	χ 2	0	G	1	0	U	∞		1							C \$\chi_C A \$\chi_2
14		1	G	1	3	1	000	0	1							
15	χ 2	0	G	2	0	-	2	0	2			1				
16	2	١ŏ	G	3	0		2	0	2			1				
17	1	l ö	A	1	3		1	2	1			2				$C G A G A C G A T $_1$
18	1	١ŏ	Â	3	0		1	2	i			2				
19	χ	1	Ĝ	2	5	1	00	_	1			-				c G C c s _x
20	2	0	A	4	0	-	4		4							
21	2	lõ	C	5	ő		4		4							
22	X	1	A	3	0	3	000									C G C G C C \$x
23	2	0	C	2	0	-	2		2							C G G C A \$2
24	1	Ó	l A	0	2		0		0							G A C G A T \$1
25	1	Ó	l c	2	0		0		o							GAGACGAT\$
26	1	0	c	2	0		0		0							G A T \$1
27	2	0	G	1	3		0		0							G C A \$2
28	χ	1	С	2	4	0	00									G C C S _X
29	2	0	С	3	0		3		3							GCCGCCGCA\$2
30	2	0	C	4	0		3		3							G C C G G C A \$2
31	χ	1	С	2	0	2	∞									G C G C C \$x
32	2	0	С	1	0		1		1							G G C A \$2
33	1	0	A	0	0		0		0		ı			l		T \$1
		1		-1		-1	1						l	l		

									Π,	lcl Cl	,		cLCI		cl CP	
					_				-						CLCF	
#	id	S	ebwt	lcp	D	lcp_{χ}	α	ζ.	χ	1	2	χ	1	2		Sorted suffixes of S
1	χ	1	C	-1	0	-1	∞	0								\$ _X
2	1	0	T	0	0		0	0	0			0				\$ ₁
3	2	0	A	0	0 2		0	0	0			0				\$2 A \$2
5	2	0	\$ ₂	0	0		0	1 1	0			1				A \$2 A A C G C C G C C G G C A \$;
6	1	0	\$ ₁	1	4		0	3	0			3				A C G A G A C G A T \$1
7	1	0	G	4	0		ő	3	0			3				$\begin{vmatrix} A & C & G & A & T & \$_1 \end{vmatrix}$
8	2	0	Ā	3	5		lő	4	0			4				$\begin{array}{ c c c c c c c c c c c c c c c c c c c$
9	χ	1	\$0	4	0	0	00	0	ľ			Ť				A C G C G C C \$x
10	1	0	G	1	0	,	1	0	1			0				$\begin{array}{c ccccccccccccccccccccccccccccccccccc$
11	1	0	Ğ	1	ő		1	١ŏ	1	l		0				A T \$1
12	χ	1	С	0	2	0	∞	0								c \$χ
13	2	0	G	1	0		1	0	1			1				C A \$2
14	χ	1	G	1	3	1	∞	0								C C \$x
15	2	0	G	2	0		2	0	2			1				C C G C C G G C A \$2
16	2	0	G	3	0		2	0	2			1				C C G G C A \$2
17	1	0	A	1	3		1	2	1			2				CGAGACGAT\$1
18	1	0	A	3	0		1	2	1			2				C G A T \$1
19	χ	1	G	2	5	1	∞	0								C G C C \$ _{\chi}
20	2	0	A	4	0		4	0	4			3				C G C C G G C A \$2
21	2	0	С	5	0		4	0	4			3				C G C C G G C A \$2
22	χ	1	Α	3	0	3	∞	0								
23	2	0	C	2	0		2	0	2			0				C G G C A \$2
24	1	0	A	0	2		0	1	0			1				G A C G A T \$1
25	1	0	C	2	0		0	1	0			1				G A G A C G A T \$1
26	1	0	C	2	0		0	1	0			1				G A T \$1
27	2	0	G	1	3		0	2	0			2				G C A \$2
28	χ	1	C	2	4	0	3	0	1			_				$\begin{array}{c ccccccccccccccccccccccccccccccccccc$
29 30	2	0	C	3	0		3	2	3			2				G C C G C C G G C A \$2 G C C G G C A \$2
31		1	c	2	0	2	-	0	3			2				G C G C C S _Y
32	2	0	c	1	0		∞ 1	0	1			0				$G G G C A S_2$
33	1	0	Ā	0	0		0	0	0	l		0				T \$1
55	- 1	١٠١	_ ^	-1	١٧	-1	١٧	1 "	ľ	ı		J	l		ı	

Missing LcLCP[χ_1][r], UcLCP[χ_2][r] values

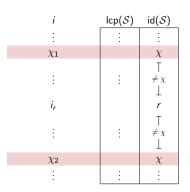
For each χ -interval $[\chi_1,\chi_2]$, we are able to compute $\mathrm{UcLCP}[i_r][\chi]$ and $\mathrm{LcLCP}[i_r][\chi]$, but not $\mathrm{LcLCP}[\chi_1][r]$ and $\mathrm{UcLCP}[\chi_2][r]$ values in a single sequential scan.

i	$lcp(\mathcal{S})$	$id(\mathcal{S})$
:	:	:
χ_1		χ
: <i>i</i> r	÷	⊺ ≠ χ ⊥ r
:	:	,
χ_2		χ
<u> </u>	:	:

What is the reason of this asymmetry?

Missing LcLCP[χ_1][r], UcLCP[χ_2][r] values (2)

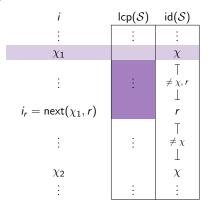
• The segmentation of lcp(S) in χ -intervals, induced by s_{χ} suffixes, is a shared reference amongst all the suffixes of any other color r.



• However, no lcp(S) segmentation in r-intervals (induced by $s_r \in S^1$ suffixes) is maintained for suffixes of s_χ during the sequential scan.

$LcLCP[\chi_1][r]$ Computation

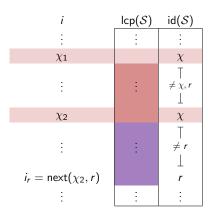
• In fact, it is easy to compute LcLCP[χ_1][r] if there exists a r-colored suffix in (χ_1, χ_2):



$$\mathsf{LcLCP}[\chi_1][r] = \mathsf{UcLCP}[i_r][\chi]$$

$LcLCP[\chi_1][r]$ Computation - Backward Propagation

• However, what if such a suffix does not exists?



We need to look beyond χ_2 and backward propagate LcLCP[χ_2][r]:

$$LcLCP[\chi_1][r] = min\{LCP(\chi_1, \chi_2), \ UcLCP[i_r][\chi]\}$$

21 / 39

Previous solution: Computational Complexity

Theorem

Let S a collection of m strings partitioned into S^0 and S^1 , let $s_\chi \in S^0$. Given $\mathrm{id}(S)$, $\mathrm{lcp}(S)$ and $\mathrm{lcp}(s_\chi)$, $\mathrm{cLCP}(S)$ can be computed by sequential scans in $\mathcal{O}(N+m|s_\chi|)$ time and $\mathcal{O}(m+\max|\mathrm{lcp}(S))$ space.

- Array D: in $\Theta(N)$ time and $\mathcal{O}(\max lcp(\mathcal{S}))$ space.
- UcLCP[i_r][χ]/LcLCP[i_r][χ]: sequentially (forward) in $\Theta(N)$ time and $\mathcal{O}(1)$ space.
- UcLCP[χ_2][r]/LcLCP[χ_1][r]: add $\mathcal{O}(m|s_\chi|)$ time and $\mathcal{O}(m)$ space for forward and backward propagation.

Previous solution: Computational Complexity

Theorem

Let S a collection of m strings partitioned into S^0 and S^1 , let $s_\chi \in S^0$. Given $\mathrm{id}(S)$, $\mathrm{lcp}(S)$ and $\mathrm{lcp}(s_\chi)$, $\mathrm{cLCP}(S)$ can be computed by sequential scans in $\mathcal{O}(N+m|s_\chi|)$ time and $\mathcal{O}(m+\max|\mathrm{lcp}(S))$ space.

- Array D: in $\Theta(N)$ time and $\mathcal{O}(\max lcp(\mathcal{S}))$ space.
- UcLCP[i_r][χ]/LcLCP[i_r][χ]: sequentially (forward) in $\Theta(N)$ time and $\mathcal{O}(1)$ space.
- UcLCP[χ_2][r]/LcLCP[χ_1][r]: add $\mathcal{O}(m|s_\chi|)$ time and $\mathcal{O}(m)$ space for forward and backward propagation.

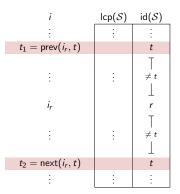
...affordable for solving the (1-vs-all) multi-string ACS problem, BUT unsuitable for an all-vs-all scenario!

Avoiding backward-forward propagation

Can we compute $UcLCP[\chi_2][r]/LcLCP[\chi_1][r]$ without relying on backward-forward propagation?

Avoiding backward-forward propagation

• **Idea**: Get rid of the lcp(S) χ -induced segmentation as a shared reference for all the other suffixes, and keep track of *all* possible t-intervals, for $t \in [1, m]$.



Avoiding backward-forward propagation (2)

• Maintain $\alpha[1, m]$ such that $\alpha[t]$ keeps track of the minimum lcp value since the beginning of each t-interval:

$$\alpha[t] = \min\{\mathsf{lcp}(\mathcal{S})[x] : x \in (\mathsf{prev}(i_r, t), i_r]\}$$

.

• Maintain $\zeta[1, m]$ such that $\zeta[t]$ keeps track of the maximum D_t value since the beginning of each t-interval:

$$\zeta[t] = \max\{D_t[x] : x \in (\operatorname{prev}(i_r, t), i_r]\}$$

.

Avoiding backward-forward propagation (3)

• For the computation of $UcLCP[i_r][t]$, we have:

$$\mathsf{UcLCP}[\mathit{i}_r][\mathit{t}] = \min\{\mathsf{lcp}(\mathcal{S})[x] : x \in (\mathit{t}_1, \mathit{i}_r]\} = \alpha[\mathit{t}]$$

• For the computation of LcLCP[i_r][t], we use Theorem 1:

Theorem

For any $1 \le i_r \le N$ such that $id(S)[i_r] = r$,

- if $LCP(t_1, i_r) > LCP(t_1, t_2)$ then $LCP(i_r, t_2) = LCP(t_1, t_2)$
- otherwise

$$LCP(i_r, t_2) = \max\{\max\{D_t[x] : t_1 < x \le i_r\} - 1, LCP(t_1, t_2)\}$$

Avoiding backward-forward propagation (3)

• For the computation of $UcLCP[i_r][t]$, we have:

$$\mathsf{UcLCP}[\mathit{i_r}][t] = \min\{\mathsf{lcp}(\mathcal{S})[x] : x \in (\mathit{t}_1, \mathit{i_r}]\} = \alpha[t]$$

• For the computation of LcLCP[i_r][t], we use Theorem 1:

Theorem

For any $1 \leq i_r \leq N$ such that $id(S)[i_r] = r$,

- if $\alpha[t] > LCP(t_1, t_2)$ then $LCP(i_r, t_2) = LCP(t_1, t_2)$
- otherwise

$$LCP(i_r, t_2) = \max\{\zeta[t] - 1, LCP(t_1, t_2)\}$$

No more forward-backward propagation: purely sequential $\mathcal{O}(Nm)$ time solution for an all-vs-all ACS computation scenario!

$$s_0$$
 A C G C G C C s_1 A C G A G A G A C G A T

$$MS(s_0, s_1)$$

Given two strings s_0 , s_1 , the matching statistics between of s_0 vs. s_1 is a $|s_0|$ -integer array such that, for any position i of s_0 , stores the length of the longest prefix of the suffix of s_0 starting at position i that is also substring of s_1 .

$$s_0$$
 $\stackrel{\downarrow}{A}$ $\stackrel{\downarrow}{C}$ $\stackrel{\downarrow}{G}$ $\stackrel{\downarrow}{C}$ $\stackrel{\downarrow}{G}$ $\stackrel{\downarrow}{A}$ $\stackrel{\downarrow}{C}$ $\stackrel{\downarrow}{G}$ $\stackrel{\downarrow}{G}$

 $MS(s_0, s_1)$ 3

Given two strings s_0 , s_1 , the matching statistics between of s_0 vs. s_1 is a $|s_0|$ -integer array such that, for any position i of s_0 , stores the length of the longest prefix of the suffix of s_0 starting at position i that is also substring of s_1 .

$$s_0$$
 A C G C G C C s_1 A C G A G A G A G A C G A T

 $MS(s_0, s_1)$ 3 2

$$s_0 \quad A \quad C \quad G \quad C \quad G \quad C \quad C \quad C \quad S_1 \quad A \quad C \quad G \quad A \quad G \quad A \quad C \quad G \quad A \quad T$$

$$MS(s_0, s_1) \quad 3 \quad 2 \quad 1 \quad 2 \quad 1 \quad 1 \quad 1$$

Given two strings s_r and s_t over the alphabet Σ of size σ , ACS is computed by proceeding in the following steps:

 $\mathbf{0}$ MS (s_r, s_t)

Given two strings s_r and s_t over the alphabet Σ of size σ , ACS is computed by proceeding in the following steps:

- $\mathbf{0}$ MS (s_r, s_t)
- $Score(s_r, s_t) = \frac{\sum_{j=1}^{|s_r|} MS(s_r, s_t)[j]}{|s_r|}$

Given two strings s_r and s_t over the alphabet Σ of size σ , ACS is computed by proceeding in the following steps:

- \bullet MS(s_r, s_t)
- Score $(s_r, s_t) = \frac{\sum_{j=1}^{|s_r|} \mathsf{MS}(s_r, s_t)[j]}{|s_r|}$ Norm $(\mathsf{Score}(s_r, s_t)) = \frac{\log_{\sigma} |s_t|}{\mathsf{Score}(s_r, s_t)} \frac{2\log_{\sigma} |s_r|}{|s_r| + 1}$

$$ACS(s_r, s_t) = \frac{Norm(Score(s_r, s_t)) + Norm(Score(s_t, s_r))}{2}$$

Given two strings s_r and s_t over the alphabet Σ of size σ , ACS is computed by proceeding in the following steps:

 \bullet MS(s_r, s_t)

$$Score(s_r, s_t) = \frac{\sum_{j=1}^{|s_r|} MS(s_r, s_t)[j]}{|s_r|}$$

Score $(s_r, s_t) = \frac{\sum_{j=1}^{|s_r|} MS(s_r, s_t)[j]}{|s_r|}$ Norm $(Score(s_r, s_t)) = \frac{\log_{\sigma} |s_t|}{Score(s_r, s_t)} - \frac{2 \log_{\sigma} |s_r|}{|s_r| + 1}$

$$ACS(s_r, s_t) = \frac{Norm(Score(s_r, s_t)) + Norm(Score(s_t, s_r))}{2}$$

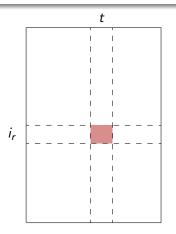
Multi-String ACS Problem

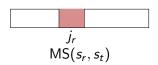
1-vs-all: Compute the pairwise ACS measure between a given string $s_v \in \mathcal{S}^0$ and each string of a set \mathcal{S}^1 of m strings, simultaneously. all-vs-all: Compute the pairwise ACS measure for any pair of strings s_r , $s_t \in \mathcal{S}$.

MS with cLCP (1)

Proposition

 $\mathsf{MS}(s_r, s_t)$ is a permutation of the values in $\mathsf{cLCP}(\mathcal{S})$ related to suffixes of s_r versus s_t .

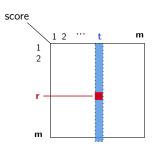




 j_r the initial position of the r-colored suffix of rank i_r

MS with cLCP (2)

Therefore, we only need to sum up cLCP values related to suffixes of s_r versus s_t (as they are computed) for each pair of strings $s_r, s_t \in \mathcal{S}$:



$$\mathsf{Score}(s_r, s_t) = \frac{\sum_{j=1}^{|s_r|} \mathsf{MS}(s_r, s_t)[j]}{|s_r|} = \left(\sum_{\substack{i_r \in [1...N] \\ \mathsf{id}(S)[i_r] = r}} \mathsf{cLCP}[i_r][t]\right) / |s_r|$$

Score Matrix Computation (sketch)

```
Initialize Score[1, m][1, m] = [[0, ..., 0], ..., [0, ..., 0]]

for t \leftarrow 1 to m do

D_t \leftarrow generate\_D(t);
lcp_t \leftarrow generate\_lcp(t);
score[][t] \leftarrow COMPUTECOLUMNSCORE(t, D_t, lcp_t, lcp(S), id(S));
COMPUTEACS(score);
```

Score Matrix Computation (sketch)

```
Initialize Score[1, m][1, m] = [[0, ..., 0], ..., [0, ..., 0]]

for t \leftarrow 1 to m do

D_t \leftarrow generate\_D(t);

lcp_t \leftarrow generate\_lcp(t);

score[][t] \leftarrow COMPUTECOLUMNSCORE(t, D_t, lcp_t, lcp(S), id(S));

COMPUTEACS(score);
```

Each column score[][t] can be computed separately from the others, provided D_t and lcp_t are pre-computed.



Score Matrix Computation (sketch)

```
Initialize Score[1, m][1, m] = [[0, ..., 0], ..., [0, ..., 0]]

for t \leftarrow 1 to m do

D_t \leftarrow generate\_D(t);

lcp_t \leftarrow generate\_lcp(t);

score[][t] \leftarrow COMPUTECOLUMNSCORE(t, D_t, lcp_t, lcp(S), id(S));

COMPUTEACS(score);
```

Straightforward parallel implementation: distinct columns assigned to distinct threads working concurrently.



COMPUTECOLUMNSCORE(t)

```
1: procedure (id(S)[1, N], lcp(S)[1, N+1], D_t[1, N], lcp_t[1, |s_t|, \alpha[t], \zeta[t])
          \alpha[t] \leftarrow \infty
 3: \zeta[t] \leftarrow 0
      h_t \leftarrow 1
                                                                                 \triangleright index for scanning lcp_t
      for i \leftarrow 1 to N do
 6:
                if id(S)[i] \neq t then
                                                      \triangleright We are inside a t-interval: [t_1, t_2]
                     \alpha[t] \leftarrow \min\{\alpha[t], \mathsf{lcp}[i]\}
 7:
                     \zeta[t] \leftarrow \max\{\zeta[t], \mathsf{D}_t[i] - 1\}
 8:
                     if \alpha[t] > \mathsf{lcp}_t[i_t] then
 9:
                           Score[id[i]][t] + \leftarrow \alpha[t]
10:
11:
                      else
                           Score[id[i]][t] + \leftarrow max\{\alpha[t], \zeta[t], lcp_t[i_t]\}
12:
                else
                                                           \triangleright A new t-interval starts, next [t_1, t_2].
13:
14:
                     h_t + +
                     \alpha[t] \leftarrow \infty
15:
                     \zeta[t] \leftarrow 0
16:
```

Preliminary Experiments

- Two collections of genomes (the first one contains 932 genomes and the second one contains 4,938 genomes).
- In both cases, the value $|s_\chi|$ is greater than the average length of the strings in the respective collection.
- k-Mismatch Average Common Substring approach tool¹ (kmacs) with k = 0.

	Size	Min length	Max length	Max Icp	Program	Wall clock	Memory
	(Gbytes)					(mm:ss)	(Kbytes)
1	3.434	1,080,084	10,657,107	1,711,194	new cLCP-mACS	2:34	110,412
					cLCP-mACS	13:37	10,716
					kmacs*	23:30	4,213,364
2	9.258	744	14,782,125	5,714,157	new cLCP-mACS	7:43	206,164
					cLCP-mACS	40:21	10,780
					kmacs*	57:43	9,637,964

 $|s_\chi|=5,650,368$ for the first collection and $|s_\chi|=3,571,103$ for the second one.

All tests were done on a MacBook Pro (13-inch), Intel Core i7 at 3,5 GHz, with 16 GB of RAM, HDD of type SSD

Future work

- Design a dynamic version of our tool (cLCP can be efficiently auto-updated by removing o inserting strings)
- Solve the many-to-many pairwise ACS problem on a collection of strings or between all strings of a collection versus all strings of another collection
- Use cLCP to define new similarity measures for string collections

Thanks for your attention!