Parallel Computation of Matching Statistics and Average Common Substring

Fabio Garofalo Daniele Greco[2]
Giovanna Rosone[2] **Marinella Sciortino**[1]

[1]Dipartimento di Matematica e Informatica
Università di Palermo, Italy

[2]Dipartimento di Informatica
Università di Pisa, Italy

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

Motivation

- 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

Motivation

- 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:

Motivation

- 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

Motivation

- 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.

Motivation

- 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

- $\Sigma = \{c_1, c_2, \ldots, c_\sigma\}$ be a finite ordered alphabet
- $\mathcal{S}$ is a collection of m strings over Σ
- n_i is the length of the string s_i
- 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 $\mathcal{S}$
- Each string (or subset of strings) is identified by a specific color

EBWT (Mantaci, Restivo, Rosone, S. 2007)

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

String collection $\mathcal{S}$

	0	1	2	3	4	5	6
s_1	G	C	C	A	A	C	$\$_1$
s_2	G	A	G	C	T	C	$\$_2$
s_3	T	C	G	C	T	T	$\$_3$

EBWT (Mantaci, Restivo, Rosone, S. 2007)

Let $\mathcal{S} = \{s_1, s_2, \ldots, s_m\}$ 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 $\mathcal{S}$

	0	1	2	3	4	5	6
s_1	G	C	C	A	A	C	$\$_1$
s_2	G	A	G	C	T	C	$\$_2$
s_3	T	C	G	C	T	T	$\$_3$

EBWT (Mantaci, Restivo, Rosone, S. 2007)

Let $\mathcal{S} = \{s_1, s_2, \ldots, s_m\}$ 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 $\mathcal{S}$

	0	1	2	3	4	5	6
s_1	G	C	C	A	A	C	$\$_1$
s_2	G	A	G	C	T	C	$\$_2$
s_3	T	C	G	C	T	T	$\$_3$

- Sort all the suffixes of the strings in $\mathcal{S}$;

ebwt($\mathcal{S}$)	Sorted Suffixes of $\mathcal{S}$
C	$\$_1$
C	$\$_2$
T	$\$_3$
C	$AAC\$_1$
A	$AC\$_1$
G	$AGCTC\$_2$
A	$C\$_1$
T	$C\$_2$
C	$CAAC\$_1$
G	$CCAAC\$_1$
T	$CGCTT\$_3$
G	$CTC\$_2$
G	$CTT\$_3$
$\$_2$	$GAGCTC\$_2$
$\$_1$	$GCCAAC\$_1$
A	$GCTC\$_2$
C	$GCTT\$_3$
T	$T\$_3$
C	$TC\$_2$
$\$_3$	$TCGCTT\$_3$
C	$TT\$_3$

EBWT (Mantaci, Restivo, Rosone, S. 2007)

Let $\mathcal{S} = \{s_1, s_2, \ldots, s_m\}$ 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 $\mathcal{S}$

	0	1	2	3	4	5	6
s_1	G	C	C	A	A	C	$\$_1$
s_2	G	A	G	C	T	C	$\$_2$
s_3	T	C	G	C	T	T	$\$_3$

- Sort all the suffixes of the strings in $\mathcal{S}$;
- The output 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 $\mathcal{S}$.

ebwt($\mathcal{S}$)	Sorted Suffixes of $\mathcal{S}$
C	$\$_1$
C	$\$_2$
T	$\$_3$
C	$AAC\$_1$
A	$AC\$_1$
G	$AGCTC\$_2$
A	$C\$_1$
T	$C\$_2$
C	$CAAC\$_1$
G	$CCAAC\$_1$
T	$CGCTT\$_3$
G	$CTC\$_2$
G	$CTT\$_3$
$\$_2$	$GAGCTC\$_2$
$\$_1$	$GCCAAC\$_1$
A	$GCTC\$_2$
C	$GCTT\$_3$
T	$T\$_3$
C	$TC\$_2$
$\$_3$	$TCGCTT\$_3$
C	$TT\$_3$

LCP and Colored EBWT

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

$\mathrm{lcp}(\mathcal{S})$	$id(\mathcal{S})$	$\mathrm{ebwt}(\mathcal{S})$	Sorted Suffixes of $\mathcal{S}$
-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	A	$GCTC\$_2$
3	3	C	$GCTT\$_3$
0	3	T	$T\$_3$
1	2	C	$TC\$_2$
2	3	$\$_3$	$TCGCTT\$_3$
1	3	C	$TT\$_3$
-1			

LCP and Colored EBWT

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

$\mathsf{lcp}(\mathcal{S})$	$id(\mathcal{S})$	$\mathsf{ebwt}(\mathcal{S})$	Sorted Suffixes of $\mathcal{S}$
-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	A	$GCTC\$_2$
3	3	C	$GCTT\$_3$
0	3	T	$T\$_3$
1	2	C	$TC\$_2$
2	3	$\$_3$	$TCGCTT\$_3$
1	3	C	$TT\$_3$
-1			

LCP and Colored EBWT

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

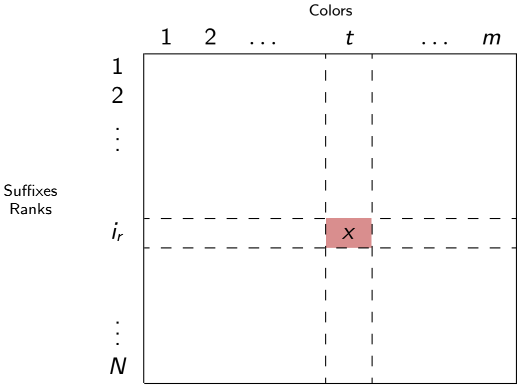
$\mathsf{lcp}(\mathcal{S})$	$id(\mathcal{S})$	$\mathsf{ebwt}(\mathcal{S})$	Sorted Suffixes of $\mathcal{S}$
-1	1	C	$\$_1$
0	2	C	$\$_2$
0	3	T	$\$_3$
0	1	C	$AAC\$_1$
1	1	A	$AC\$_1$
1	2	G	$AGCTC\$_2$
0	1	A	$C\$_1$
1	2	T	$C\$_2$
1	1	C	$CAAC\$_1$
1	1	G	$CCAAC\$_1$
1	3	T	$CGCTT\$_3$
1	2	G	$CTC\$_2$
2	3	G	$CTT\$_3$
0	2	$\$_2$	$GAGCTC\$_2$
1	1	$\$_1$	$GCCAAC\$_1$
2	2	A	$GCTC\$_2$
3	3	C	$GCTT\$_3$
0	3	T	$T\$_3$
1	2	C	$TC\$_2$
2	3	$\$_3$	$TCGCTT\$_3$
1	3	C	$TT\$_3$
-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 $\mathcal{S}$.



$x = \mathrm{LCP}$ value between the r-colored suffix of rank i_r and the nearest t-colored suffix in the sorted list of suffixes

Colored LCP - Formal Definition

Given $1 \leq i_r \leq N$ and $t = 1, \ldots m$, how $\text{cLCP}[i_r][t]$ is defined?

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

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

i	$\text{lcp}(\mathcal{S})$	$\text{id}(\mathcal{S})$
$\vdots$	$\vdots$	$\vdots$
$\text{prev}(i_r, t)$	l_1	t
$\vdots$	$\vdots$	$\neq t$
i_r	l	r
$\vdots$	$\vdots$	$\neq t$
$\text{next}(i_r, t)$	l_2	t
$\vdots$	$\vdots$	$\vdots$

Colored LCP - Formal Definition

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

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

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

i	$\mathrm{lcp}(\mathcal{S})$	$\mathrm{id}(\mathcal{S})$
$\vdots$	$\vdots$	$\vdots$
$\mathrm{prev}(i_r, t)$	l_1	t
$\vdots$	$\vdots$	$\neq t$
i_r	l	r
$\vdots$	$\vdots$	$\neq t$
$\mathrm{next}(i_r, t)$	l_2	t
$\vdots$	$\vdots$	$\vdots$

Upper Colored LCP $\quad \mathrm{UcLCP}[i_r][t] = \mathrm{LCP}(\mathrm{prev}(i_r, t), i_r)$

Colored LCP - Formal Definition

Given $1 \leq i_r \leq N$ and $t = 1, \ldots m$,
how $\mathrm{cLCP}[i_r][t]$ is defined?

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

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

i	$\mathrm{lcp}(\mathcal{S})$	$\mathrm{id}(\mathcal{S})$
$\vdots$	$\vdots$	$\vdots$
$\mathrm{prev}(i_r, t)$	l_1	t
$\vdots$	$\vdots$	$\neq t$
i_r	l	r
$\vdots$	$\vdots$	$\neq t$
$\mathrm{next}(i_r, t)$	l_2	t
$\vdots$	$\vdots$	$\vdots$

Lower Colored LCP $\quad \mathrm{LcLCP}[i_r][t] = \mathrm{LCP}(i_r, \mathrm{next}(i_r, t))$

Colored LCP - Formal Definition

Given $1 \leq i_r \leq N$ and $t = 1, \ldots m$, how $\mathrm{cLCP}[i_r][t]$ is defined?

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

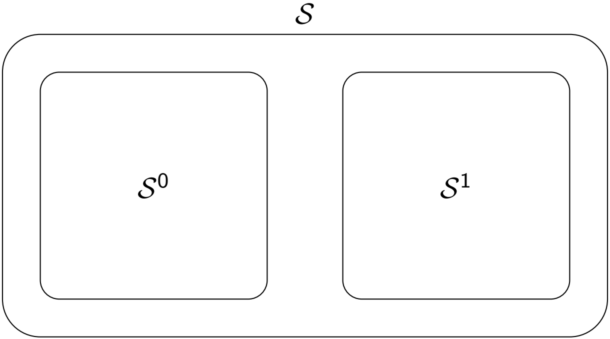
$next(i,t) = \min\{x \mid i < x \leq N, \mathrm{id}(\mathcal{S})[x] = t\}$

i	$\mathrm{lcp}(\mathcal{S})$	$\mathrm{id}(\mathcal{S})$
$\vdots$	$\vdots$	$\vdots$
$\mathrm{prev}(i_r, t)$	l_1	t
$\vdots$	$\vdots$	$\neq t$
i_r	l	r
$\vdots$	$\vdots$	$\neq t$
$\mathrm{next}(i_r, t)$	l_2	t
$\vdots$	$\vdots$	$\vdots$

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

Colored LCP on disjoint collections

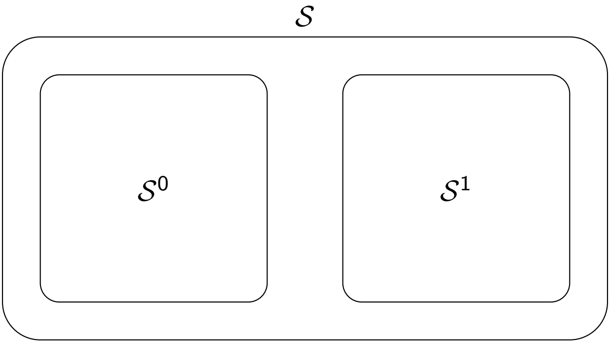
cLCP can also be defined for disjoint collections of strings.



The value cLCP$[i_r][t]$ is defined for each pair (i_r, t) such that $\mathrm{id}(\mathcal{S})[i_r] = r$, $t \in \mathrm{ID}$ 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 $\mathcal{S}^1 = \mathcal{S} \setminus \{s_\chi\}$.

$\chi - intervals$

A given string $s_\chi \in \mathcal{S}^0$ implicitly induces a partition of $\mathrm{lcp}(\mathcal{S})$ 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 $\mathrm{id}[i_r] = r$ and $s_r \in \mathcal{S}^1$.

i	$\mathrm{lcp}(\mathcal{S})$	$\mathrm{id}(\mathcal{S})$
$\vdots$	$\vdots$	$\vdots$
$\chi_1 = \mathrm{prev}(i_r, \chi)$		χ
$\vdots$	$\vdots$	$\neq \chi$
i_r		r
$\vdots$	$\vdots$	$\neq \chi$
$\chi_2 = \mathrm{next}(i_r, \chi)$		χ
$\vdots$	$\vdots$	$\vdots$

χ − *intervals*

A given string $s_\chi \in \mathcal{S}^0$ implicitly induces a partition of $\mathrm{lcp}(\mathcal{S})$ 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 $\mathrm{id}[i_r] = r$ and $s_r \in \mathcal{S}^1$.

i	$\mathrm{lcp}(\mathcal{S})$	$\mathrm{id}(\mathcal{S})$
$\vdots$	$\vdots$	$\vdots$
$\chi_1 = \mathrm{prev}(i_r, \chi)$		χ
$\vdots$	$\vdots$	$\neq \chi$
i_r		r
$\vdots$	$\vdots$	$\neq \chi$
$\chi_2 = \mathrm{next}(i_r, \chi)$		χ
$\vdots$	$\vdots$	$\vdots$

Inside a χ-interval, how to compute $\mathrm{UcLCP}[i_r][\chi]$ and $\mathrm{LcLCP}[i_r][\chi]$ by a lightweight strategy?

UcLCP$[i_r][\chi]$ Computation

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

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

.

	i	lcp($\mathcal{S}$)	id($\mathcal{S}$)
	$\vdots$	$\vdots$	$\vdots$
$\alpha \leftarrow \infty$	$\chi_1 = \mathrm{prev}(i_r, \chi)$		χ
	$\vdots$	$\vdots$	$\neq \chi$
	i_r		r
	$\vdots$	$\vdots$	$\neq \chi$
	$\chi_2 = \mathrm{next}(i_r, \chi)$		χ
	$\vdots$	$\vdots$	$\vdots$

UcLCP$[i_r][\chi]$ Computation

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

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

.

	i	$\mathsf{lcp}(\mathcal{S})$	$\mathsf{id}(\mathcal{S})$
	$\vdots$	$\vdots$	$\vdots$
	$\chi_1 = \mathsf{prev}(i_r, \chi)$		χ
	$\vdots$	$\vdots$	$\neq \chi$
$\alpha \leftarrow \mathrm{LCP}(\chi_1, i_r)$	i_r		r
	$\vdots$	$\vdots$	$\neq \chi$
	$\chi_2 = \mathsf{next}(i_r, \chi)$		χ
	$\vdots$	$\vdots$	$\vdots$

Example

$$s_\chi = ACGCGCC\$_\chi, \qquad s_1 = ACGAGACGAT\$_1, \qquad s_2 = AACGCCGCCGGCA\$_2$$

							UcLCP			LcLCP	cLCP	
#	id	$\mathcal{S}$	ebwt	lcp	lcp_χ	α	χ	1	2			Sorted suffixes of $\mathcal{S}$
1	χ	**1**	**C**	**-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								A $A C G C C G C C G G C A \$_2$
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								A C G C $C G C C G G C A \$_2$
9	χ	**1**	$\$_0$	**4**	**0**							**A C G C G C C** $\$_\chi$
10	1	0	G	1								A $G A C G A T \$_1$
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	χ	**1**	**G**	**1**	**1**	∞						**C C** $\$_\chi$
15	2	0	G	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								C G $A G A C G A T \$_1$
18	1	0	A	3								C G $A T \$_1$
19	χ	**1**	**G**	**2**	**1**							**C G C C** $\$_\chi$
20	2	0	A	4								C G C C $G C C G G C A \$_2$
21	2	0	C	5								C G C C $G G C A \$_2$
22	χ	**1**	**A**	**3**	**3**							**C G C G C C** $\$_\chi$
23	2	0	C	2								C G $G C A \$_2$
24	1	0	A	0								G $A C G A T \$_1$
25	1	0	C	2								G $A G A C G A T \$_1$
26	1	0	C	2								G $A T \$_1$
27	2	0	G	1								G C $A \$_2$
28	χ	**1**	**C**	**2**	**0**							**G C C** $\$_\chi$
29	2	0	C	3								G C C $G C C G G C A \$_2$
30	2	0	C	4								G C C $G G C A \$_2$
31	χ	**1**	**C**	**2**	**2**							**G C G C C** $\$_\chi$
32	2	0	C	1								G $G C A \$_2$
33	1	0	A	0								$T \$_1$
				-1	-1							

Example

$s_\chi = ACGCGCC\$_\chi, \qquad s_1 = ACGAGACGAT\$_1, \qquad s_2 = AACGCCGCCGGCA\$_2$

							UcLCP			LcLCP	cLCP	
#	id	$\mathcal{S}$	ebwt	lcp	lcp$_\chi$	α	χ	1	2			Sorted suffixes of $\mathcal{S}$
1	χ	**1**	**C**	**-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								A A C G C C G C C G G C A $\$_2$
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								A C G C C G C C G G C A $\$_2$
9	χ	**1**	$\$_0$	**4**	**0**							**A C G C G C C** $\$_\chi$
10	1	0	G	1								A G A C G A T $\$_1$
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	χ	**1**	**G**	**1**	**1**	∞						**C C** $\$_\chi$
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								C G A G A C G A T $\$_1$
18	1	0	A	3								C G A T $\$_1$
19	χ	**1**	**G**	**2**	**1**							**C G C C** $\$_\chi$
20	2	0	A	4								C G C C G C C G G C A $\$_2$
21	2	0	C	5								C G C C G G C A $\$_2$
22	χ	**1**	**A**	**3**	**3**							**C G C G C C** $\$_\chi$
23	2	0	C	2								C G G C A $\$_2$
24	1	0	A	0								G A C G A T $\$_1$
25	1	0	C	2								G A G A C G A T $\$_1$
26	1	0	C	2								G A T $\$_1$
27	2	0	G	1								G C A $\$_2$
28	χ	**1**	**C**	**2**	**0**							**G C C** $\$_\chi$
29	2	0	C	3								G C C G C C G G C A $\$_2$
30	2	0	C	4								G C C G G C A $\$_2$
31	χ	**1**	**C**	**2**	**2**							**G C G C C** $\$_\chi$
32	2	0	C	1								G G C A $\$_2$
33	1	0	A	0								T $\$_1$
				-1	-1							

Example

$s_\chi = ACGCGCC\$_\chi, \qquad s_1 = ACGAGACGAT\$_1, \qquad s_2 = AACGCCGCCGGCA\$_2$

							UcLCP			LcLCP	cLCP	
#	id	$\mathcal{S}$	ebwt	lcp	lcp_χ	α	χ	1	2			Sorted suffixes of $\mathcal{S}$
1	χ	1	C	-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								A A C G C C G C C G G C A $\$_2$
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								A C G C C G C C G G C A $\$_2$
9	χ	1	$\$_0$	4	0							A C G C G C C $\$_\chi$
10	1	0	G	1								A G A C G A T $\$_1$
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	χ	1	G	1	1	∞						C C $\$_\chi$
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 $\$_2$
17	1	0	A	1								C G A G A C G A T $\$_1$
18	1	0	A	3								C G A T $\$_1$
19	χ	1	G	2	1							C G C C $\$_\chi$
20	2	0	A	4								C G C C G C C G G C A $\$_2$
21	2	0	C	5								C G C C G G C A $\$_2$
22	χ	1	A	3	3							C G C G C C $\$_\chi$
23	2	0	C	2								C G G C A $\$_2$
24	1	0	A	0								G A C G A T $\$_1$
25	1	0	C	2								G A G A C G A T $\$_1$
26	1	0	C	2								G A T $\$_1$
27	2	0	G	1								G C A $\$_2$
28	χ	1	C	2	0							G C C $\$_\chi$
29	2	0	C	3								G C C G C C G G C A $\$_2$
30	2	0	C	4								G C C G G C A $\$_2$
31	χ	1	C	2	2							G C G C C $\$_\chi$
32	2	0	C	1								G G C A $\$_2$
33	1	0	A	0								T $\$_1$
				−1	−1							

Example

$$s_\chi = ACGCGCC\$_\chi, \qquad s_1 = ACGAGACGAT\$_1, \qquad s_2 = AACGCCGCCGGCA\$_2$$

							UcLCP			LcLCP	cLCP	
#	id	$\mathcal{S}$	ebwt	lcp	lcp_χ	α	χ	1	2			Sorted suffixes of $\mathcal{S}$
1	χ	**1**	**C**	**-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								A A C G C C G C C G G C A $\$_2$
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								A C G C C G C C G G C A $\$_2$
9	χ	**1**	$\$_0$	**4**	**0**							**A C G C G C C** $\$_\chi$
10	1	0	G	1								A G A C G A T $\$_1$
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	χ	**1**	**G**	**1**	**1**	∞						**C C** $\$_\chi$
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 $\$_2$
17	1	0	A	1		1	1					C G A G A C G A T $\$_1$
18	1	0	A	3								C G A T $\$_1$
19	χ	**1**	**G**	**2**	**1**							**C G C C** $\$_\chi$
20	2	0	A	4								C G C C G C C G G C A $\$_2$
21	2	0	C	5								C G C C G G C A $\$_2$
22	χ	**1**	**A**	**3**	**3**							**C G C G C C** $\$_\chi$
23	2	0	C	2								C G G C A $\$_2$
24	1	0	A	0								G A C G A T $\$_1$
25	1	0	C	2								G A G A C G A T $\$_1$
26	1	0	C	2								G A T $\$_1$
27	2	0	G	1								G C A $\$_2$
28	χ	**1**	**C**	**2**	**0**							**G C C** $\$_\chi$
29	2	0	C	3								G C C G C C G G C A $\$_2$
30	2	0	C	4								G C C G G C A $\$_2$
31	χ	**1**	**C**	**2**	**2**							**G C G C C** $\$_\chi$
32	2	0	C	1								G G C A $\$_2$
33	1	0	A	0								T $\$_1$
				-1	-1							

Example

$$s_\chi = ACGCGCC\$_\chi, \qquad s_1 = ACGAGACGAT\$_1, \qquad s_2 = AACGCCGCCGGCA\$_2$$

							UcLCP			LcLCP	cLCP	
#	id	$\mathcal{S}$	ebwt	lcp	lcp$_\chi$	α	χ	1	2			Sorted suffixes of $\mathcal{S}$
1	χ	**1**	**C**	**-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								$AACGCCGCCGGCA\$_2$
6	1	0	$\$_1$	1								$ACGAGACGAT\$_1$
7	1	0	G	4								$ACGAT\$_1$
8	2	0	A	3								$ACGCCGCCGGCA\$_2$
9	χ	**1**	$\$_0$	**4**	**0**							**ACGCGCC**$\$_\chi$
10	1	0	G	1								$AGACGAT\$_1$
11	1	0	G	1								$AT\$_1$
12	χ	**1**	**C**	**0**	**0**							**C**$\$_\chi$
13	2	0	G	1								$CA\$_2$
14	χ	**1**	**G**	**1**	**1**	∞						**CC**$\$_\chi$
15	2	0	G	2		2	2					$CCGCCGGCA\$_2$
16	2	0	G	3		2	2					$CCGGCA\$_2$
17	1	0	A	1		1	1					$CGAGACGAT\$_1$
18	1	0	A	3		1	1					$CGAT\$_1$
19	χ	**1**	**G**	**2**	**1**							**CGCC**$\$_\chi$
20	2	0	A	4								$CGCCGCCGGCA\$_2$
21	2	0	C	5								$CGCCGGCA\$_2$
22	χ	**1**	**A**	**3**	**3**							**CGCGCC**$\$_\chi$
23	2	0	C	2								$CGGCA\$_2$
24	1	0	A	0								$GACGAT\$_1$
25	1	0	C	2								$GAGACGAT\$_1$
26	1	0	C	2								$GAT\$_1$
27	2	0	G	1								$GCA\$_2$
28	χ	**1**	**C**	**2**	**0**							**GCC**$\$_\chi$
29	2	0	C	3								$GCCGCCGGCA\$_2$
30	2	0	C	4								$GCCGGCA\$_2$
31	χ	**1**	**C**	**2**	**2**							**GCGCC**$\$_\chi$
32	2	0	C	1								$GGCA\$_2$
33	1	0	A	0								$T\$_1$
				-1	-1							

Example

$$s_\chi = ACGCGCC\$_\chi, \qquad s_1 = ACGAGACGAT\$_1, \qquad s_2 = AACGCCGCCGGCA\$_2$$

							UcLCP			LcLCP	cLCP	
#	id	$\mathcal{S}$	ebwt	lcp	lcp$_\chi$	α	χ	1	2			Sorted suffixes of $\mathcal{S}$
1	χ	**1**	**C**	**-1**	**-1**	∞						$\$_\chi$
2	1	0	*T*	0		0	0					$\$_1$
3	2	0	*A*	0		0	0					$\$_2$
4	2	0	*C*	0		0	0					A $\$_2$
5	2	0	$\$_2$	1		0	0					A A C G C C G C C G G C A $\$_2$
6	1	0	$\$_1$	1		0	0					A C G A G A C G A T $\$_1$
7	1	0	*G*	4		0	0					A C G A T $\$_1$
8	2	0	*A*	3		0	0					A C G C C G C C G G C A $\$_2$
9	χ	**1**	$\$_0$	**4**	**0**	∞						**A C G C G C C** $\$_\chi$
10	1	0	*G*	1		1	1					A G A C G A T $\$_1$
11	1	0	*G*	1		1	1					A T $\$_1$
12	χ	**1**	**C**	**0**	**0**	∞						**C** $\$_\chi$
13	2	0	*G*	1		1	1					C A $\$_2$
14	χ	**1**	**G**	**1**	**1**	∞						**C C** $\$_\chi$
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 $\$_2$
17	1	0	*A*	1		1	1					C G A G A C G A T $\$_1$
18	1	0	*A*	3		1	1					C G A T $\$_1$
19	χ	**1**	**G**	**2**	**1**	∞						**C G C C** $\$_\chi$
20	2	0	*A*	4		4	4					C G C C G C C G G C A $\$_2$
21	2	0	*C*	5		4	4					C G C C G G C A $\$_2$
22	χ	**1**	**A**	**3**	**3**	∞						**C G C G C C** $\$_\chi$
23	2	0	*C*	2		2	2					C G G C A $\$_2$
24	1	0	*A*	0		0	0					G A C G A T $\$_1$
25	1	0	*C*	2		0	0					G A G A C G A T $\$_1$
26	1	0	*C*	2		0	0					G A T $\$_1$
27	2	0	*G*	1		0	0					G C A $\$_2$
28	χ	**1**	**C**	**2**	**0**	∞						**G C C** $\$_\chi$
29	2	0	*C*	3		3	3					G C C G C C G G C A $\$_2$
30	2	0	*C*	4		3	3					G C C G G C A $\$_2$
31	χ	**1**	**C**	**2**	**2**	∞						**G C G C C** $\$_\chi$
32	2	0	*C*	1		1	1					G G C A $\$_2$
33	1	0	*A*	0		0	0					T $\$_1$
				-1	-1							

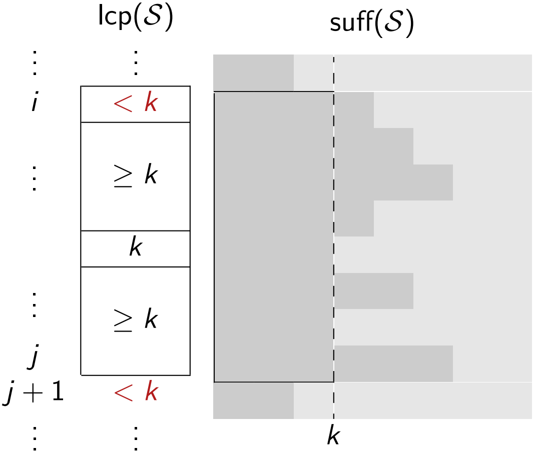
LcLCP$[i_r][\chi]$ Computation

i	$\mathsf{lcp}(\mathcal{S})$	$\mathsf{id}(\mathcal{S})$
$\vdots$	$\vdots$	$\vdots$
$\chi_1 = \mathsf{prev}(i_r, \chi)$		χ
$\vdots$	$\vdots$	$\neq \chi$
i_r		r
$\vdots$	$\vdots$	$\neq \chi$
$\chi_2 = \mathsf{next}(i_r, \chi)$		χ
$\vdots$	$\vdots$	$\vdots$

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

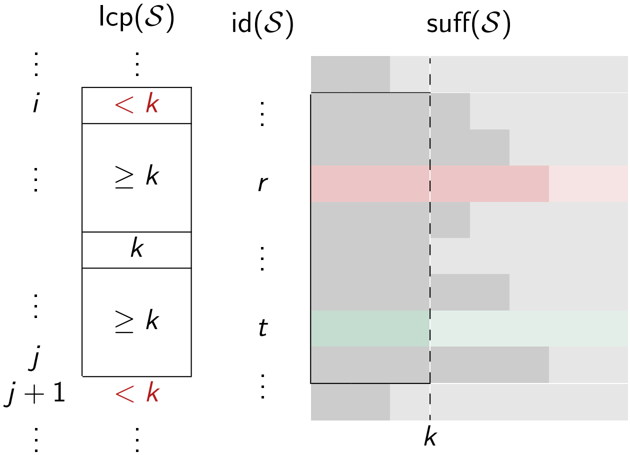
Colored k-lcp interval

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



Colored k-lcp interval

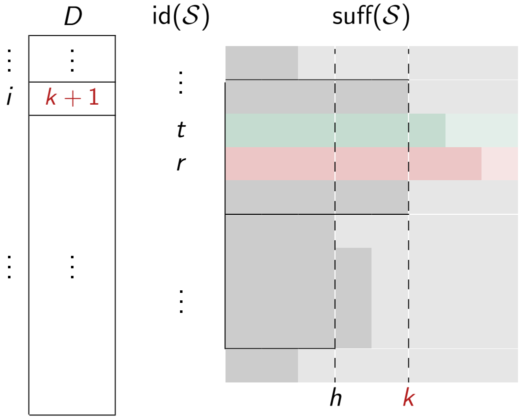
A colored k-lcp interval is an interval $[i, j]$ such that:



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 h-lcp interval starting at position i then $h \leq k-1$.



LcLCP$[i_r][\chi]$ Computation

Theorem

For any $1 \leq i_r \leq N$ *such that* $\mathrm{id}(\mathcal{S})[i_r] = r$,

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

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

LcLCP$[i_r][\chi]$ Computation

Theorem

For any $1 \leq i_r \leq N$ *such that* $\mathrm{id}(\mathcal{S})[i_r] = r$,

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

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

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

	i	$\mathrm{lcp}(\mathcal{S})$	D	$\mathrm{id}(\mathcal{S})$
	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$\zeta \leftarrow 0$	$\chi_1 = \mathrm{prev}(i_r, \chi)$			χ
	$\vdots$	$\vdots$	$\vdots$	$\neq \chi$
	i_r			r
	$\vdots$	$\vdots$	$\vdots$	$\neq \chi$
	$\chi_2 = \mathrm{next}(i_r, \chi)$			χ
	$\vdots$	$\vdots$	$\vdots$	$\vdots$

LcLCP$[i_r][\chi]$ Computation

Theorem

For any $1 \leq i_r \leq N$ *such that* $\mathrm{id}(\mathcal{S})[i_r] = r$*,*

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

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

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

	i	$\mathrm{lcp}(\mathcal{S})$	D	$\mathrm{id}(\mathcal{S})$
	$\vdots$	$\vdots$	$\vdots$	$\vdots$
	$\chi_1 = \mathrm{prev}(i_r, \chi)$			χ
	$\vdots$	$\vdots$	$\vdots$	$\neq \chi$
$\zeta \leftarrow \max \mathrm{D}(\chi_1, i_r]$	i_r			r
	$\vdots$	$\vdots$	$\vdots$	$\neq \chi$
	$\chi_2 = \mathrm{next}(i_r, \chi)$			χ
	$\vdots$	$\vdots$	$\vdots$	$\vdots$

Example

									UcLCP			LcLCP			cLCP	
#	id	$\mathcal{S}$	ebwt	lcp	D	lcp$_\chi$	α	ζ	χ	1	2	χ	1	2		Sorted suffixes of $\mathcal{S}$
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 $\$_2$
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							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 *C G C C G G C A* $\$_2$
9	χ	**1**	$\$_0$	**4**	**0**	**0**	∞									**A C G C G C C** $\$_\chi$
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	χ	**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**	∞									**C C** $\$_\chi$
15	2	0	*G*	2	0		2		2							C C *G C C G G C A* $\$_2$
16	2	0	*G*	3	0		2		2							C C *G G C A* $\$_2$
17	1	0	*A*	1	3		1		1							C G *A G A C G A T* $\$_1$
18	1	0	*A*	3	0		1		1							C G *A T* $\$_1$
19	χ	**1**	**G**	**2**	**5**	**1**	∞									**C G C C** $\$_\chi$
20	2	0	*A*	4	0		4		4							C G C C *G C C G G C A* $\$_2$
21	2	0	*C*	5	0		4		4							C G C C *G G C A* $\$_2$
22	χ	**1**	**A**	**3**	**0**	**3**	∞									**C G C G C C** $\$_\chi$
23	2	0	*C*	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							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** $\$_\chi$
32	2	0	*C*	1	0		1		1							G *G C A* $\$_2$
33	1	0	*A*	0	0		0		0							*T* $\$_1$
				-1		-1										

Example

									UcLCP			LcLCP			cLCP	
#	id	$\mathcal{S}$	ebwt	lcp	D	lcp_χ	α	ζ	χ	1	2	χ	1	2		Sorted suffixes of $\mathcal{S}$
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 $\$_2$
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							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 C G C C G G C A* $\$_2$
9	χ	**1**	$\$_0$	**4**	**0**	**0**	∞									**A C G C G C C** $\$_\chi$
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	χ	**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		2							C C *G C C G G C A* $\$_2$
16	2	0	*G*	3	0		2		2							C C *G G C A* $\$_2$
17	1	0	*A*	1	3		1		1							C G *A G A C G A T* $\$_1$
18	1	0	*A*	3	0		1		1							C G *A T* $\$_1$
19	χ	**1**	**G**	**2**	**5**	**1**	∞									**C G C C** $\$_\chi$
20	2	0	*A*	4	0		4		4							C G C C *G C C G G C A* $\$_2$
21	2	0	*C*	5	0		4		4							C G C C *G G C A* $\$_2$
22	χ	**1**	**A**	**3**	**0**	**3**	∞									**C G C G C C** $\$_\chi$
23	2	0	*C*	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							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** $\$_\chi$
32	2	0	*C*	1	0		1		1							G *G C A* $\$_2$
33	1	0	*A*	0	0		0		0							*T* $\$_1$
				−1		−1										

Example

									UcLCP			LcLCP			cLCP	
#	id	$\mathcal{S}$	ebwt	lcp	D	lcp$_\chi$	α	ζ	χ	1	2	χ	1	2		Sorted suffixes of $\mathcal{S}$
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 $\$_2$
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							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 C G C C G G C A $\$_2$
9	χ	**1**	$\$_0$	**4**	**0**	**0**	∞									**A C G C G C C** $\$_\chi$
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	χ	**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		2							C C G G C A $\$_2$
17	1	0	*A*	1	3		1		1							C G A G A C G A T $\$_1$
18	1	0	*A*	3	0		1		1							C G A T $\$_1$
19	χ	**1**	**G**	**2**	**5**	**1**	∞									**C G C C** $\$_\chi$
20	2	0	*A*	4	0		4		4							C G C C G C C G G C A $\$_2$
21	2	0	*C*	5	0		4		4							C G C C G G C A $\$_2$
22	χ	**1**	**A**	**3**	**0**	**3**	∞									**C G C G C C** $\$_\chi$
23	2	0	*C*	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							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** $\$_\chi$
32	2	0	*C*	1	0		1		1							G G C A $\$_2$
33	1	0	*A*	0	0		0		0							*T* $\$_1$
				−1		−1										

Example

#	id	$\mathcal{S}$	ebwt	lcp	D	lcp_χ	α	ζ	UcLCP χ	UcLCP 1	UcLCP 2	LcLCP χ	LcLCP 1	LcLCP 2	cLCP	Sorted suffixes of $\mathcal{S}$
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 $\$_2$
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							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 C G C C G G C A $\$_2$
9	χ	1	$\$_0$	4	0	0	∞									A C G C G C C $\$_\chi$
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	χ	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	1	0	A	1	3		1		1							C G A G A C G A T $\$_1$
18	1	0	A	3	0		1		1							C G A T $\$_1$
19	χ	1	G	2	5	1	∞									C G C C $\$_\chi$
20	2	0	A	4	0		4		4							C G C C G C C G G C A $\$_2$
21	2	0	C	5	0		4		4							C G C C G G C A $\$_2$
22	χ	1	A	3	0	3	∞									C G C G C C $\$_\chi$
23	2	0	C	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							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 $\$_\chi$
32	2	0	C	1	0		1		1							G G C A $\$_2$
33	1	0	A	0	0		0		0							T $\$_1$
				−1		−1										

Example

									UcLCP			LcLCP			cLCP	
#	id	$\mathcal{S}$	ebwt	lcp	D	lcp$_\chi$	α	ζ	χ	1	2	χ	1	2		Sorted suffixes of $\mathcal{S}$
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 $\$_2$
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							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 C G C C G G C A $\$_2$
9	χ	**1**	$\$_0$	**4**	**0**	**0**	∞									**A C G C G C C** $\$_\chi$
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	χ	**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	1	0	*A*	1	3		1	2	1			2				C G A G A C G A T $\$_1$
18	1	0	*A*	3	0		1		1							C G A T $\$_1$
19	χ	**1**	**G**	**2**	**5**	**1**	∞									**C G C C** $\$_\chi$
20	2	0	*A*	4	0		4		4							C G C C G C C G G C A $\$_2$
21	2	0	*C*	5	0		4		4							C G C C G G C A $\$_2$
22	χ	**1**	**A**	**3**	**0**	**3**	∞									**C G C G C C** $\$_\chi$
23	2	0	*C*	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							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** $\$_\chi$
32	2	0	*C*	1	0		1		1							G G C A $\$_2$
33	1	0	*A*	0	0		0		0							*T* $\$_1$
				-1		-1										

Example

									UcLCP			LcLCP			cLCP	
#	id	$\mathcal{S}$	ebwt	lcp	D	lcp$_\chi$	α	ζ	χ	1	2	χ	1	2		Sorted suffixes of $\mathcal{S}$
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 $\$_2$
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							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 C G C C G G C A $\$_2$
9	χ	**1**	$\$_0$	**4**	**0**	**0**	∞									**A C G C G C C** $\$_\chi$
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	χ	**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	1	0	A	1	3		1	2	1			2				C G A G A C G A T $\$_1$
18	1	0	A	3	0		1	2	1			2				C G A T $\$_1$
19	χ	**1**	**G**	**2**	**5**	**1**	∞									**C G C C** $\$_\chi$
20	2	0	A	4	0		4		4							C G C C G C C G G C A $\$_2$
21	2	0	C	5	0		4		4							C G C C G G C A $\$_2$
22	χ	**1**	**A**	**3**	**0**	**3**	∞									**C G C G C C** $\$_\chi$
23	2	0	C	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							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** $\$_\chi$
32	2	0	C	1	0		1		1							G G C A $\$_2$
33	1	0	A	0	0		0		0							T $\$_1$
				-1		-1										

Example

									UcLCP			LcLCP			cLCP	
#	id	$\mathcal{S}$	ebwt	lcp	D	lcp_χ	α	ζ	χ	1	2	χ	1	2		Sorted suffixes of $\mathcal{S}$
1	χ	**1**	**C**	**-1**	**0**	**-1**	∞	**0**								$\$_\chi$
2	1	0	*T*	0	0		0	0	0			0				$\$_1$
3	2	0	*A*	0	0		0	0	0			0				$\$_2$
4	2	0	*C*	0	2		0	1	0			1				A $\$_2$
5	2	0	$\$_2$	1	0		0	1	0			1				A *A C G C C G C C G G C A* $\$_2$
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		0	3	0			3				A C G *A T* $\$_1$
8	2	0	*A*	3	5		0	4	0			4				A C G C *C G C C G G C A* $\$_2$
9	χ	**1**	$\$_0$	**4**	**0**	**0**	∞	**0**								**A C G C G C C** $\$_\chi$
10	1	0	*G*	1	0		1	0	1			0				A *G A C G A T* $\$_1$
11	1	0	*G*	1	0		1	0	1			0				A *T* $\$_1$
12	χ	**1**	**C**	**0**	**2**	**0**	∞	**0**								**C** $\$_\chi$
13	2	0	*G*	1	0		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	1	0	*A*	1	3		1	2	1			2				C G *A G A C G A T* $\$_1$
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 C C G G C A* $\$_2$
21	2	0	*C*	5	0		4	0	4			3				C G C C *G G C A* $\$_2$
22	χ	**1**	**A**	**3**	**0**	**3**	∞	**0**								**C G C G C C** $\$_\chi$
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**	∞	**0**								**G C C** $\$_\chi$
29	2	0	*C*	3	0		3	2	3			2				G C C *G C C G G C A* $\$_2$
30	2	0	*C*	4	0		3	3	3			2				G C C *G G C A* $\$_2$
31	χ	**1**	**C**	**2**	**0**	**2**	∞	**0**								**G C G C C** $\$_\chi$
32	2	0	*C*	1	0		1	0	1			0				G *G C A* $\$_2$
33	1	0	*A*	0	0		0	0	0			0				*T* $\$_1$
				−1		−1										

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

For each χ-interval $[\chi_1, \chi_2]$, we are able to compute UcLCP[i_r][χ] and LcLCP[i_r][χ] , but not LcLCP[χ_1][r] and UcLCP[χ_2][r] values in a single sequential scan.

i	lcp($\mathcal{S}$)	id($\mathcal{S}$)
$\vdots$	$\vdots$	$\vdots$
χ_1		χ
$\vdots$	$\vdots$	$\neq \chi$
i_r		r
$\vdots$	$\vdots$	$\neq \chi$
χ_2		χ
$\vdots$	$\vdots$	$\vdots$

What is the reason of this asymmetry?

Missing LcLCP$[\chi_1][r]$, UcLCP$[\chi_2][r]$ values (2)

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

i	lcp($\mathcal{S}$)	id($\mathcal{S}$)
$\vdots$	$\vdots$	$\vdots$
χ_1		χ
$\vdots$	$\vdots$	$\neq \chi$
i_r		r
$\vdots$	$\vdots$	$\neq \chi$
χ_2		χ
$\vdots$	$\vdots$	$\vdots$

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

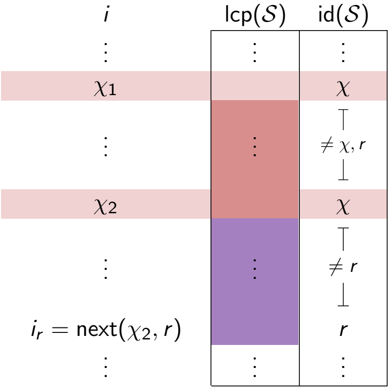
LcLCP$[\chi_1][r]$ Computation

- In fact, it is easy to compute LcLCP$[\chi_1][r]$ if there exists a r-colored suffix in (χ_1, χ_2):

i	$\mathrm{lcp}(\mathcal{S})$	$\mathrm{id}(\mathcal{S})$
$\vdots$	$\vdots$	$\vdots$
χ_1		χ
$\vdots$	$\vdots$	$\neq \chi, r$
$i_r = \mathrm{next}(\chi_1, r)$		r
$\vdots$	$\vdots$	$\neq \chi$
χ_2		χ
$\vdots$	$\vdots$	$\vdots$

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

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



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

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

Previous solution: Computational Complexity

Theorem

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

- Array D: in $\Theta(N)$ time and $\mathcal{O}(\max \mathrm{lcp}(\mathcal{S}))$ space.
- $\mathrm{UcLCP}[i_r][\chi]/\mathrm{LcLCP}[i_r][\chi]$: sequentially (forward) in $\Theta(N)$ time and $\mathcal{O}(1)$ space.
- $\mathrm{UcLCP}[\chi_2][r]/\mathrm{LcLCP}[\chi_1][r]$: add $\mathcal{O}(m|s_\chi|)$ time and $\mathcal{O}(m)$ space for forward and backward propagation.

Previous solution: Computational Complexity

Theorem

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

- Array D: in $\Theta(N)$ time and $\mathcal{O}(\max \mathrm{lcp}(\mathcal{S}))$ space.
- $\mathrm{UcLCP}[i_r][\chi]/\mathrm{LcLCP}[i_r][\chi]$: sequentially (forward) in $\Theta(N)$ time and $\mathcal{O}(1)$ space.
- $\mathrm{UcLCP}[\chi_2][r]/\mathrm{LcLCP}[\chi_1][r]$: 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 $\mathsf{lcp}(\mathcal{S})$ χ-induced segmentation as a shared reference for all the other suffixes, and keep track of *all* possible t-intervals, for $t \in [1, m]$.

i	$\mathsf{lcp}(\mathcal{S})$	$\mathsf{id}(\mathcal{S})$
$\vdots$	$\vdots$	$\vdots$
$t_1 = \mathsf{prev}(i_r, t)$		t
$\vdots$	$\vdots$	$\neq t$
i_r		r
$\vdots$	$\vdots$	$\neq t$
$t_2 = \mathsf{next}(i_r, t)$		t
$\vdots$	$\vdots$	$\vdots$

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:

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

.

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

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

.

Avoiding backward-forward propagation (3)

- For the computation of $\mathsf{UcLCP}[i_r][t]$, we have:

$$\mathsf{UcLCP}[i_r][t] = \min\{\mathsf{lcp}(\mathcal{S})[x] : x \in (t_1, i_r]\} = \alpha[t]$$

.

- For the computation of $\mathsf{LcLCP}[i_r][t]$, we use Theorem 1:

Theorem

For any $1 \leq i_r \leq N$ *such that* $\mathsf{id}(\mathcal{S})[i_r] = r$*,*

- *if* $\mathrm{LCP}(t_1, i_r) > \mathrm{LCP}(t_1, t_2)$ *then* $\mathrm{LCP}(i_r, t_2) = \mathrm{LCP}(t_1, t_2)$
- *otherwise*

$$\mathrm{LCP}(i_r, t_2) = \max\{\max\{\mathsf{D}_t[x] : t_1 < x \leq i_r\} - 1,\ \mathrm{LCP}(t_1, t_2)\}$$

Avoiding backward-forward propagation (3)

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

$$\mathsf{UcLCP}[i_r][t] = \min\{\mathsf{lcp}(\mathcal{S})[x] : x \in (t_1, i_r]\} = \alpha[t]$$

.

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

Theorem

For any $1 \leq i_r \leq N$ *such that* $\mathsf{id}(\mathcal{S})[i_r] = r$,

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

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

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

Matching Statistics

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	C	G	A	T

$\mathrm{MS}(s_0, s_1)$

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.

$$
\begin{array}{lcccccccccc}
 & \downarrow & & & & & & & & & \\
s_0 & A & C & G & C & G & C & C & & & \\
s_1 & A & C & G & A & G & A & C & G & A & T \\
 & & & & & & & & & & \\
\mathrm{MS}(s_0, s_1) & 3 & & & & & & & & &
\end{array}
$$

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.

		$\downarrow$								
s_0	A	C	G	C	G	C	C			
s_1	A	C	G	A	G	A	C	G	A	T
$\mathrm{MS}(s_0, s_1)$	3	2								

Matching Statistics

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.

$$
\begin{array}{lcccccccccc}
 & & & \downarrow & & & & & & & \\
s_0 & A & C & G & C & G & C & C & & & \\
s_1 & A & C & G & A & G & A & C & G & A & T \\
 & & & & & & & & & & \\
\mathrm{MS}(s_0, s_1) & 3 & 2 & 1 & & & & & & &
\end{array}
$$

Matching Statistics

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.

$$
\begin{array}{lcccccccccc}
 & & & & \downarrow & & & & & & \\
s_0 & A & C & G & C & G & C & C & & & \\
s_1 & A & C & G & A & G & A & C & G & A & T \\
 & & & & & & & & & & \\
\mathrm{MS}(s_0, s_1) & 3 & 2 & 1 & 2 & & & & & &
\end{array}
$$

Matching Statistics

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.

$$
\begin{array}{lcccccccccc}
 & & & & & \downarrow & & & & & \\
s_0 & A & C & G & C & G & C & C & & & \\
s_1 & A & C & G & A & G & A & C & G & A & T \\
 & & & & & & & & & & \\
\mathrm{MS}(s_0, s_1) & 3 & 2 & 1 & 2 & 1 & & & & &
\end{array}
$$

Matching Statistics

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.

$$
\begin{array}{lcccccccccc}
 & & & & & & \downarrow & & & & \\
s_0 & A & C & G & C & G & C & C & & & \\
s_1 & A & C & G & A & G & A & C & G & A & T \\
 & & & & & & & & & & \\
\mathrm{MS}(s_0, s_1) & 3 & 2 & 1 & 2 & 1 & 1 & & & &
\end{array}
$$

Matching Statistics

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.

$$\begin{array}{lcccccccccc} & & & & & & & \downarrow & & & \\ s_0 & A & C & G & C & G & C & C & & & \\ s_1 & A & C & G & A & G & A & C & G & A & T \\ & & & & & & & & & & \\ \mathrm{MS}(s_0, s_1) & 3 & 2 & 1 & 2 & 1 & 1 & 1 & & & \end{array}$$

Average Common Substring (ACS) measure

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

1. $\mathrm{MS}(s_r, s_t)$

Average Common Substring (ACS) measure

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

1. $\mathsf{MS}(s_r, s_t)$
2. $\mathsf{Score}(s_r, s_t) = \dfrac{\sum_{j=1}^{|s_r|} \mathsf{MS}(s_r, s_t)[j]}{|s_r|}$

Average Common Substring (ACS) measure

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

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

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

Average Common Substring (ACS) measure

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

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

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

Multi-String ACS Problem

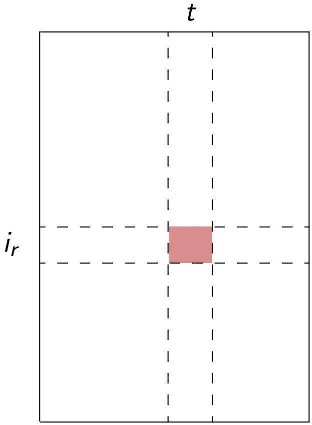
1-vs-all: Compute the pairwise ACS measure between a given string $s_\chi \in \mathcal{S}^0$ and each string of a set $\mathcal{S}^1$ of m strings, simultaneously.

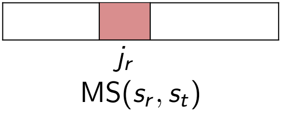
all-vs-all: Compute the pairwise ACS measure for any pair of strings s_r, $s_t \in \mathcal{S}$.

MS with cLCP (1)

Proposition

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

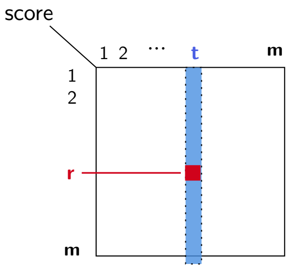




j_r the initial position of the r-colored suffix of rank i_r

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 $s_r, s_t \in \mathcal{S}$:



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

Score Matrix Computation (sketch)

Initialize $\mathsf{Score}[1,m][1,m] = [[0,\ldots,0],\ldots,[0,\ldots,0]]$
    for $t \leftarrow 1$ **to** m **do**
        $D_t \leftarrow generate\_D(t)$;
        $lcp_t \leftarrow generate\_lcp(t)$;
        $score[\,][t] \leftarrow \textsc{computeColumnScore}(t, D_t, lcp_t, \mathsf{lcp}(S), \mathsf{id}(S))$;
$\textsc{computeACS}(\text{score})$;

Score Matrix Computation (sketch)

Initialize $\mathsf{Score}[1,m][1,m] = [[0,\ldots,0],\ldots,[0,\ldots,0]]$
  for $t \leftarrow 1$ **to** m **do**
    $D_t \leftarrow \mathit{generate\_D}(t)$;
    $\mathit{lcp}_t \leftarrow \mathit{generate\_lcp}(t)$;
    $\mathit{score}[\,][t] \leftarrow \textsc{computeColumnScore}(t, D_t, \mathit{lcp}_t, \mathsf{lcp}(S), \mathsf{id}(S))$;
$\textsc{computeACS}(\text{score})$;

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 $\mathsf{Score}[1,m][1,m] = [[0,\ldots,0],\ldots,[0,\ldots,0]]$

for $t \leftarrow 1$ **to** m **do**

$D_t \leftarrow \mathit{generate\_D}(t)$;

$\mathit{lcp}_t \leftarrow \mathit{generate\_lcp}(t)$;

$\mathit{score}[\,][t] \leftarrow \textsc{computeColumnScore}(t, D_t, \mathit{lcp}_t, \mathsf{lcp}(S), \mathsf{id}(S))$;

$\textsc{computeACS}(\text{score})$;

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

COMPUTECOLUMNSCORE(t)

```
procedure (id(S)[1, N], lcp(S)[1, N+1], D_t[1, N], lcp_t[1, |s_t|, α[t], ζ[t])
    α[t] ← ∞
    ζ[t] ← 0
    h_t ← 1                                   ▷ index for scanning lcp_t
    for i ← 1 to N do
        if id(S)[i] ≠ t then                  ▷ We are inside a t-interval: [t_1, t_2]
            α[t] ← min{α[t], lcp[i]}
            ζ[t] ← max{ζ[t], D_t[i] − 1}
            if α[t] > lcp_t[i_t] then
                Score[id[i]][t]+ ← α[t]
            else
                Score[id[i]][t]+ ← max{α[t], ζ[t], lcp_t[i_t]}
        else                                  ▷ A new t-interval starts, next [t_1, t_2].
            h_t + +
            α[t] ← ∞
            ζ[t] ← 0
```

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 tool[1] (kmacs) with $k = 0$.

	Size (Gbytes)	Min length	Max length	Max lcp	Program	Wall clock (mm:ss)	Memory (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

[1] C.-A. Leimeister and M. Burkhard, **Kmacs: the k-mismatch average common substring approach to alignment-free sequence comparison.** *Bioinformatics*, 30(14), 2000-2008.

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!