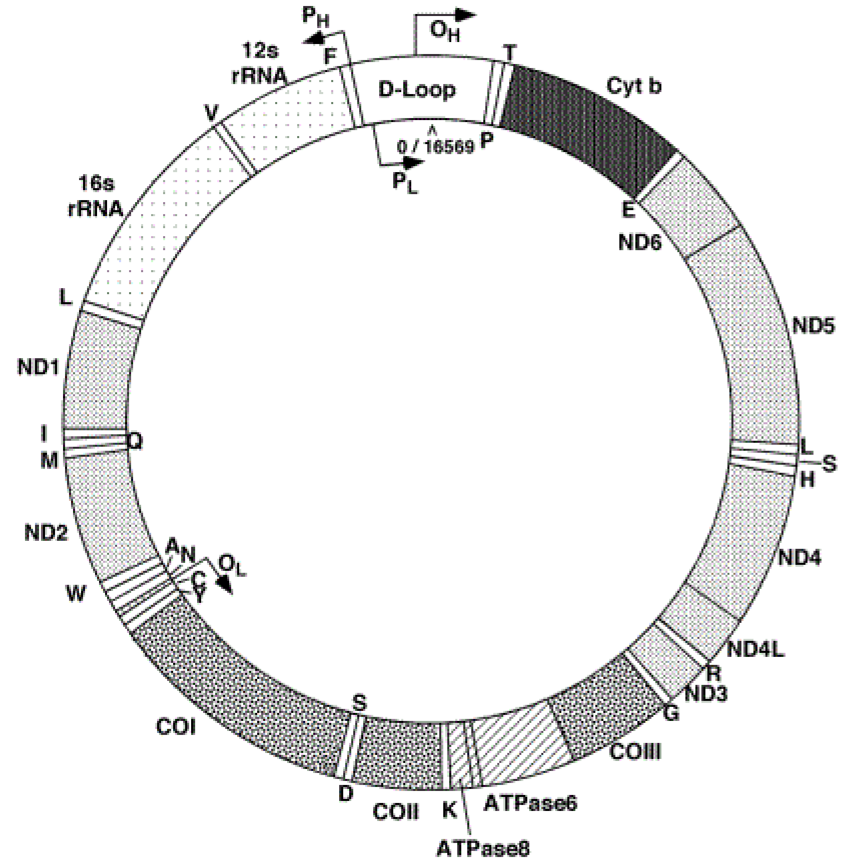
CIRCULAR SEQUENCE COMPARISON: ALGORITHMS AND APPLICATIONS



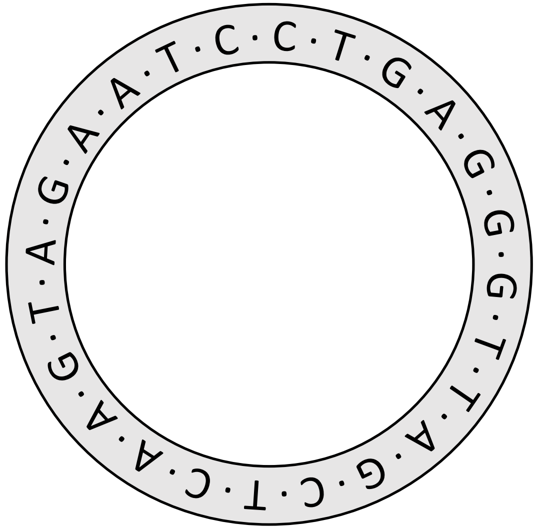
Circular Sequence Comparison: Algorithms and Applications

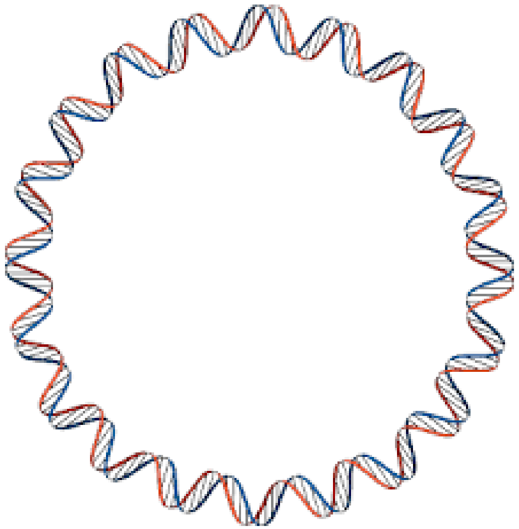
Authors:
Nadia Pisanti, Roberto Grossi (University of Pisa)
and 5 others (King’s College London).

Soon to be published on BMC Bioinformatics.

These slides are available at:
https://github.com/robzan8/csc

Circular Sequences

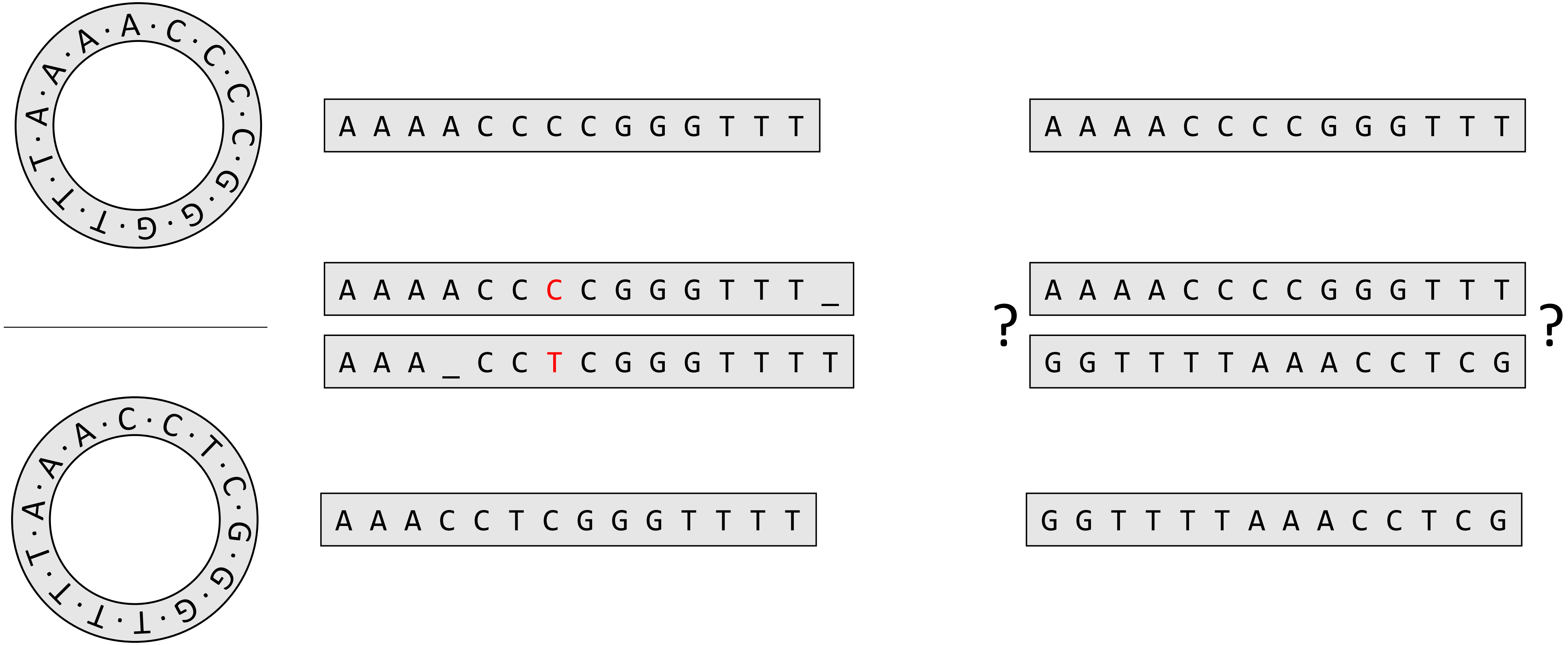

C · C · T · G · A · G · G · G · T · T · A · G · C · T · C · A · A · G · T · A · G · A · A · T



Examples:

Bacterial chromosomes and plasmids;
Mitochondrial DNA;
Viral genomes;
Circular proteins;
And more…

Comparisons



A A A A C C C C G G G T T T

A A A A C C C C G G G T T T _

A A A _ C C T C G G G T T T T

A A A C C T C G G G T T T T

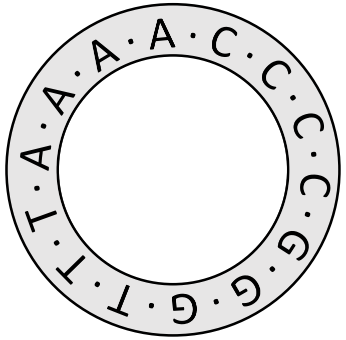
A A A A C C C C G G G T T T

A A A A C C C C G G G T T T

G G T T T T A A A C C T C G

G G T T T T A A A C C T C G

Rotation



x A A A A C | C C C G G G T T T

x^5 C C C G G G T T T | A A A A C

$(xx)[5 \ldots n+5]$ A A A A C | C C C G G G T T T | A A A A C | C C C G G G T T T

q-gram distance

Approximation of edit distance, defined as:

$$D_q(x, y) = \sum_{v \in \sum^q} |G_q(x)[v] - G_q(y)[v]|$$

where $G_q(x)[v]$ counts the number of occurrences of q-gram v in x.

Can be computed in linear time and space with an hash table that associates to each v the $diff\ G_q(x)[v] - G_q(y)[v]$

β-blockwise q-gram distance

x and y are divided in blocks, blockwise distance is the sum of the distances of all the block pairs

$D_{\beta,q}(x,y)$

x A A A A | C C C C | G G G T | T T A A

y A A A C | C T C G | G G T T | T T A A

Generalization of the q-gram distance, more accurate, ensures better locality.

Can be computed in linear time and $\mathcal{O}(\frac{m+n}{\beta})$ space.

Circular Sequence Comparison problem (CSC)

Input: strings x, y of lengths m and $n \geq m$, integers $\beta \geq 1$ and $q < m$

Output: i such that $D_{\beta,q}(x^i, y)$ is minimal

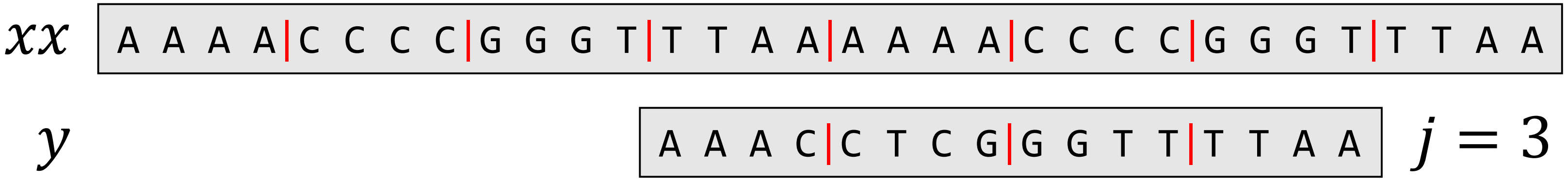
Naïve algorithm (nCSC) complexity: $\mathcal{O}(m(m+n))$

Heuristic algorithm (hCSC)

Step 1: divide xx in 2β blocks and y in β blocks.

Step 2: calculate $\delta_j = D_{\beta,q}(x^{j\frac{m}{\beta}}, y)$ shifting the window block by block.

Step 3: starting form position with best δ_j, refine search by moving left and right by m/β characters.



xx A A A A | C C C C | G G G T | T T A A | A A A A | C C C C | G G G T | T T A A

y A A A C | C T C G | G G T T | T T A A $j = 3$

Analysis of hCSC

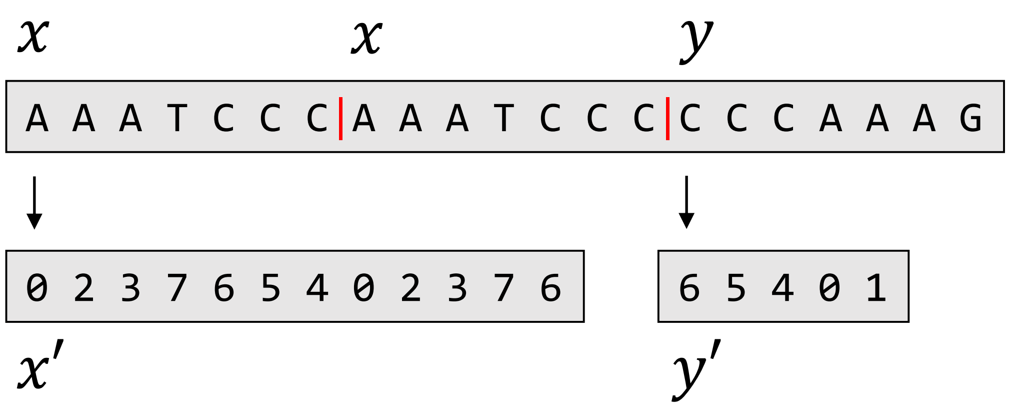
Step 2 (block-by-block search)	$\mathcal{O}(\beta(m+n))$
Step 3 (char-by-char local search)	$\mathcal{O}(\frac{m}{\beta}(m+n))$
Total	$\mathcal{O}\left(\left(\beta+\frac{m}{\beta}\right)(m+n)\right)$
By letting $\beta = \sqrt{m}$	$\mathcal{O}(\sqrt{m}(m+n))$

Linear additional space

Does not necessarily find global optimum!

Exact algorithm based on suffix array (saCSC)

Step 1: Construct the suffix array of xxy and create x' and y' from xx and y substituting each q-gram with its rank



A A A T C C C | A A A T C C C | C C C A A A G

0 2 3 7 6 5 4 0 2 3 7 6

6 5 4 0 1

SA

A A A	0
A A G	1
A A T	2
A T C	3
C A A	4
C C A	5
C C C	6
T C C	7

Exact algorithm based on suffix array (saCSC)

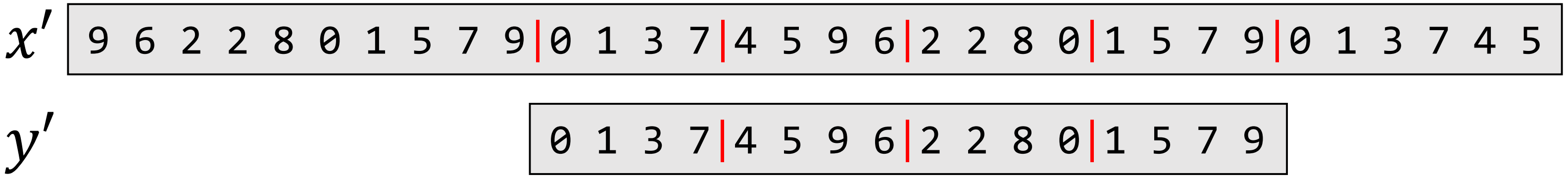
Step 2: Compute the blockwise q-gram distance for the initial window position, keeping a $diff$ array for each block

x' 9 6 2 2|8 0 1 5|7 9 0 1|3 7 4 5|9 6 2 2 8 0 1 5 7 9 0 1 3 7 4 5

y' 0 1 3 7|4 5 9 6|2 2 8 0|1 5 7 9

Exact algorithm based on suffix array (saCSC)

Step 3: Slide the window char-by-char, updating the $diff$s and the distance value accordingly;
keep track of the window position i with the lowest distance



x' 9 6 2 2 8 0 1 5 7 9|0 1 3 7|4 5 9 6|2 2 8 0|1 5 7 9|0 1 3 7 4 5

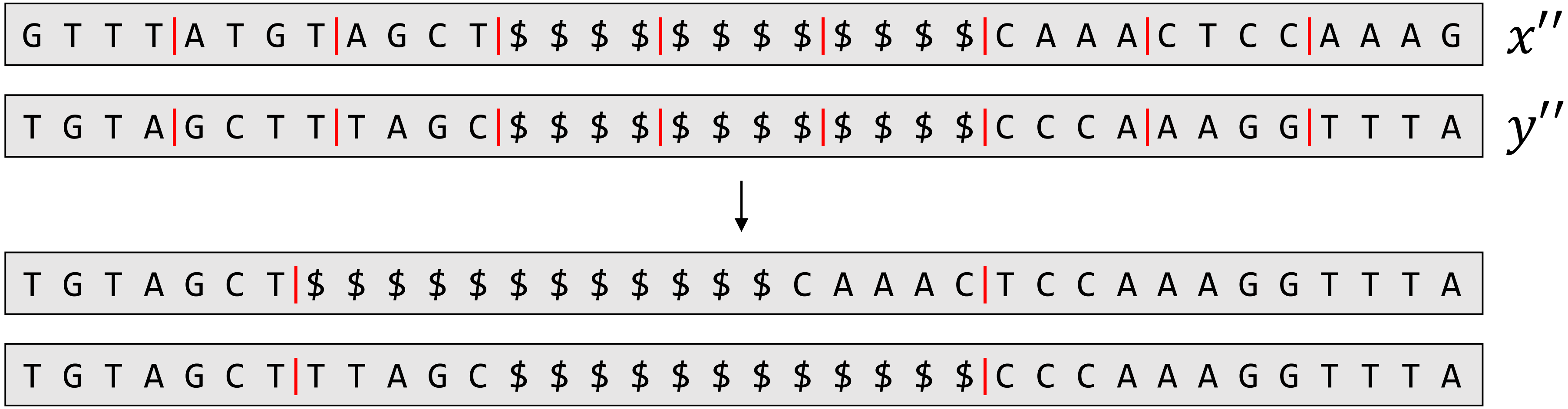
y' 0 1 3 7|4 5 9 6|2 2 8 0|1 5 7 9

Analysis of saCSC

Suffix array calculation	$\mathcal{O}(m+n)$
Distance with sliding window	$\mathcal{O}(\beta m+n)$
Total (for both time and space)	$\mathcal{O}(\beta m+n)$

saCSC refinement step (saCSCr)

Alignment of the first and last p blocks of x^i and y with Needleman-Wunsch, considering all possible rotations



G T T T | A T G T | A G C T | $ $ $ $ | $ $ $ $ | $ $ $ $ | C A A A | C T C C | A A A G x''

T G T A | G C T T | T A G C | $ $ $ $ | $ $ $ $ | $ $ $ $ | C C C A | A A G G | T T T A y''

T G T A G C T | $ $ $ $ $ $ $ $ $ $ $ $ C A A A C | T C C A A A G G T T T A

T G T A G C T | T T A G C $ $ $ $ $ $ $ $ $ $ $ $ | C C C A A A G G T T T A

$p = 3$

Time complexity: $\mathcal{O}\left(\left(p\frac{m}{\beta}\right)^3\right)$

Experimental results

Algorithm tested on various real and synthetic data

Applications on real data:
Chimpanzee - human MtDNA (from GenBank) comparison:
85% similarity and ~1200 gaps with EMBOSS Needle vs
91% similarity and 77 gaps with correct rotation;
Distance-based phylogenetic reconstruction:
MtDNA, viroid RNA, circular proteins.

saCSCr gives the same results as cNW (“brute force” Needleman-Wunsch), but with much smaller execution times!

Time performance

Experimental performance is in line with theoretical expectations

saCSC	(q-gram dist, suffix array)	$\mathcal{O}(\beta m + n)$
hCSC	(q-gram dist, heuristic)	$\mathcal{O}(\sqrt{m}(m + n))$
hSW	(Smith-Waterman, heuristic)	$\mathcal{O}(mn)$
nCSC	(q-gram dist, naïve)	$\mathcal{O}(m(m + n))$
cNW	(Needleman-Wunsch, naïve)	$\mathcal{O}(m^2 n)$

Conclusions

- β-blockwise q-gram distance can be computed efficiently and used effectively
- saCSC solves the CSC problem exactly and fast
- Refinement step bridges the gap between q-gram approximation and optimal solution
- saCSCr to be implemented in BEAR (state-of-the-art tool for multiple circular sequence alignment)

Q&A!