

On the Parameterized Complexity of the Repetition Free Longest Common Subsequence Problem

Guillaume Blin¹ Paola Bonizzoni² Riccardo Dondi³
Florian Sikora⁴

¹LIGM, Université Paris Est, CNRS – France

²DISCo, Università Milano-Bicocca – Italy

³Università Bergamo – Italy

⁴LAMSADE, Université Paris Dauphine, CNRS – France

florian.sikora@dauphine.fr

On the Parameterized Complexity of the Repetition Free Longest Common Subsequence Problem

Guillaume Blin¹ Paola Bonizzoni² Riccardo Dondi³
Florian Sikora⁴

¹LIGM, Université Paris Est, CNRS – France

²DISCo, Università Milano-Bicocca – Italy

³Università Bergamo – Italy

⁴LAMSADE, Université Paris Dauphine, CNRS – France

florian.sikora@dauphine.fr



On the Parameterized Complexity of the Repetition Free Longest Common Subsequence Problem

Guillaume Blin¹ Paola Bonizzoni² Riccardo Dondi³
Florian Sikora⁴

¹LIGM, Université Paris Est, CNRS – France

²DISCo, Università Milano-Bicocca – Italy

³Università Bergamo – Italy

⁴LAMSADE, Université Paris Dauphine, CNRS – France

florian.sikora@dauphine.fr



On the Parameterized Complexity of the Repetition Free Longest Common Subsequence Problem

Guillaume Blin¹ Paola Bonizzoni² Riccardo Dondi³
Florian Sikora⁴

¹LIGM, Université Paris Est, CNRS – France

²DISCo, Università Milano-Bicocca – Italy

³Università Bergamo – Italy

⁴LAMSADE, Université Paris Dauphine, CNRS – France

florian.sikora@dauphine.fr



Outline

Introduction

Kernelization

FPT

Outline

Introduction

Kernelization

FPT

Genomes

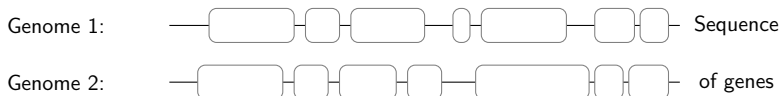
Genome 1: ...cgtaggtcgttacgtaactattacgaggttcagtacactcgctagactgca...

Genome 2: ...catacgtcgttacgtatctatagtgaggtctcagtacacacgtgactgca...

Genomes

Genome 1: ...cgtaggtcgttacgtaactattacgaggttcagtacactcgctagactgca...

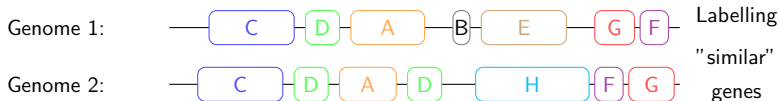
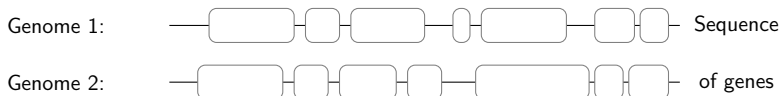
Genome 2: ...catacgtcgttacgtatctatagtgaggtctcagtacacacgtgactgca...



Genomes

Genome 1: ...cgtaggtcgttacgtaactattacgaggttcagtacactcgctagactgca...

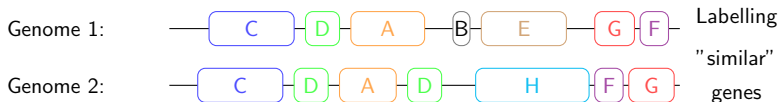
Genome 2: ...catacgtcgttacgtatctatagtgaggtctcagtacacacgtgactgca...



Genomes

Genome 1: ...cgtaggtcgttacgtaactattacgaggttcagtacactcgctagactgca...

Genome 2: ...catacgtcgttacgtatctatagtgaggtctcagtacacacgtgactgca...



Genome 1: CDABEGF

Genome 2: CDADHFG

Strings

Motivations

- ▶ Recent approach: Consider **evolutionary model** for genomes.
- ▶ Goal: inference ancestral genomes, evolutionary events that originated two genomes.
- ▶ Approach based on **alignment of genomes**, measure similarities and differences.

Longest Common Subsequence (LCS)

- ▶ Well known problem in Computational Biology.

LCS:

- ▶ **Input:** Two strings.
- ▶ **Output:** A string of maximum length which is a **subsequence** of both strings.

Longest Common Subsequence (LCS)

- ▶ Well known problem in Computational Biology.

LCS:

- ▶ **Input:** Two strings.
 - ▶ **Output:** A string of maximum length which is a **subsequence** of both strings.
-
- ▶ Polynomial for 2 (or a fixed number of) strings in the input via DP [HSU, DU, 1984].
 - ▶ NP-hard otherwise, even with alphabet of size 2 [MAIER, 1978].

Exemplar model

- ▶ Fact: evolution implies that genomes contain **multiple copies** of a gene.
- ▶ Exemplar model [SANKOFF 1999]: for each family of duplicated genes there is an **exemplar** one, representative gene from which all other genes have originated.

Exemplar model

- ▶ Fact: evolution implies that genomes contain **multiple copies** of a gene.
- ▶ Exemplar model [SANKOFF 1999]: for each family of duplicated genes there is an **exemplar** one, representative gene from which all other genes have originated.
- ▶ LCS problem under the exemplar model: **no repetition** of a symbol in the subsequence solution.

Exemplar model

- ▶ Fact: evolution implies that genomes contain **multiple copies** of a gene.
- ▶ Exemplar model [SANKOFF 1999]: for each family of duplicated genes there is an **exemplar** one, representative gene from which all other genes have originated.
- ▶ LCS problem under the exemplar model: **no repetition** of a symbol in the subsequence solution.

Repetition Free LCS: [Adi et al. 2010]

- ▶ **Input:** Two strings s_1, s_2 over an alphabet.
 - ▶ **Output:** A string s of maximum length which is a subsequence of both strings, s.t. each symbol of the alphabet **occurs at most once** in s .
-
- ▶ Following the parsimony principle, the exemplar genes are those s.t. the resulting similarity is the best.

Exemplar model

- ▶ Fact: evolution implies that genomes contain **multiple copies** of a gene.
- ▶ Exemplar model [SANKOFF 1999]: for each family of duplicated genes there is an **exemplar** one, representative gene from which all other genes have originated.
- ▶ LCS problem under the exemplar model: **no repetition** of a symbol in the subsequence solution.

Decision Repetition Free LCS: [Adi et al. 2010]

- ▶ **Input:** Two strings s_1, s_2 over an alphabet, **an integer k** .
 - ▶ **Output:** A string s of length **at least k** which is a subsequence of both strings, s.t. each symbol of the alphabet **occurs at most once** in s .
-
- ▶ Following the parsimony principle, the exemplar genes are those s.t. the resulting similarity is the best.

Example

s_1 :



s_2 :

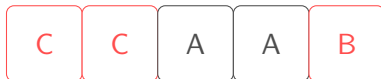


Example

s_1 :



s_2 :



LCS:



Example

 s_1 : s_2 :

LCS:



A RFLCS:



RFLCS - Known results

- ▶ Polynomial-time solvable if [ADI ET AL. 2010]:
 - ▶ Each symbol **occurs at most once** in one string (=LCS).
 - ▶ There is a fixed number of symbols with multiple occurrences.
- ▶ APX-hard, even with at most 2 occurrences of each symbol [ADI ET AL. 2010].
- ▶ “Trivially” o -approximable, where o is the maximum number of occurrences of a symbol [ADI ET AL. 2010].
 - ▶ Compute a LCS (an upper-bound) and remove repetitions...
- ▶ In FPT for the parameter size of the solution by color-coding, $O^*((2e)^k)$ if randomized [BONIZZONI ET AL. 2010].

Outline

Introduction

Kernelization

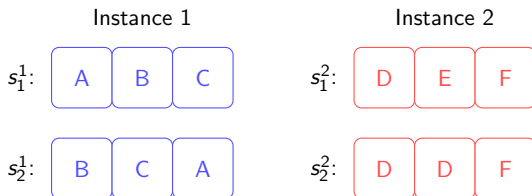
FPT

A negative result

- ▶ RFLCS is in FPT: admit a kernel.
- ▶ Our result: RFLCS **does not admit a polynomial-size kernel** (unless $\text{NP} \subseteq \text{coNP}/\text{Poly}$).
 - ▶ The derived classical problem is NP-complete, remains to show the OR-composition.

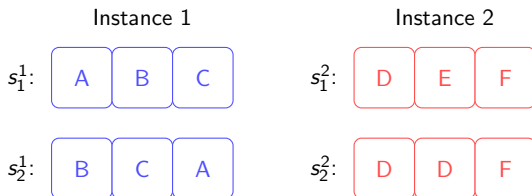
Compose two instances

- We can assume that the 2 instances are on different alphabet (otherwise, we can build an equivalent instance).

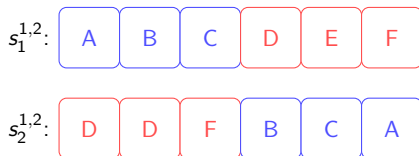


Compose two instances

- We can assume that the 2 instances are on different alphabet (otherwise, we can build an equivalent instance).

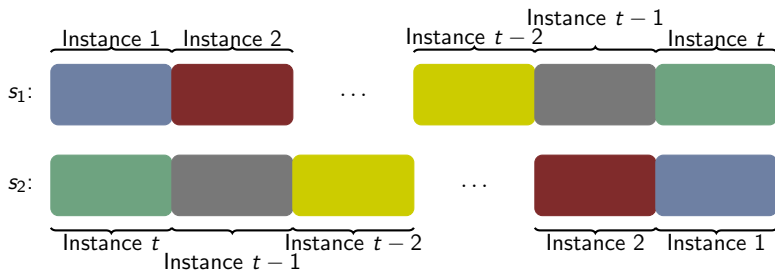


Composed instance

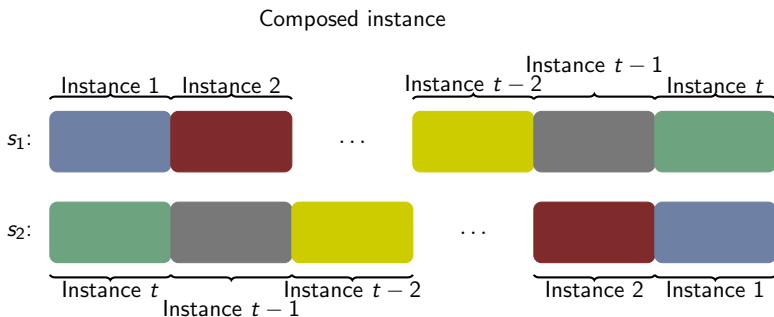


OR-composition

Composed instance



OR-composition



- Since the alphabet are disjoint, there is a RFLCS of size k **in one of the t instances** iff there is a RFLCS of size k **in the composed instance**.

Outline

Introduction

Kernelization

FPT

RFLCS and polynomials

- ▶ Known: RFLCS in FPT w.r.t. the parameter size of the solution.
- ▶ Randomized algorithm in $O^*((2e)^k)$ time and $O^*(2^k)$ space, via color-coding (worse running time for the deterministic one).
- ▶ Improvement of these complexity using the framework of Koutis and Williams [2008,2009].
- ▶ Key result:
 - ▶ There is a randomized algorithm to decide in **time $O^*(2^k)$ and polynomial space**, if a polynomial represented by an arithmetic circuit contains a **multilinear monomial of degree k** .

Polynomials

- ▶ Framework successfully applied for different problems:
 - ▶ k -PATH.
 - ▶ k -TREE.
 - ▶ k -LEAF SPANNING TREE.
 - ▶ t -DOMINATING SET.
 - ▶ GRAPH MOTIF.
 - ▶ EXEMPLAR BREAKPOINT DISTANCE.
 - ▶ ...

Polynomials

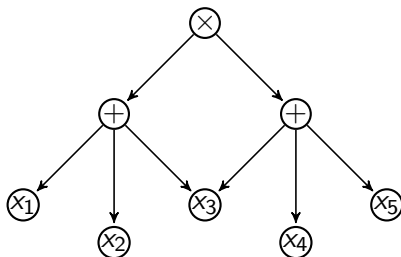
- ▶ A monomial is **multilinear** if each variable of the monomial occurs at most once.
- ▶ By definition, the degree of a multilinear monomial is the number of its variables.
- ▶ Example: $P(X) = (x_1^2 x_3 x_5 + x_1 x_2 x_4 x_6)$:
 - ▶ $x_1 x_2 x_4 x_6$ is a multilinear monomial of degree 4.
 - ▶ $x_1^2 x_3 x_5$ is not a multilinear monomial.

Polynomials

- ▶ An **arithmetic circuit** over a set of variables X is a DAG s.t.:
 - ▶ internal nodes are the operations $\times$ or $+$,
 - ▶ leafs are the elements of X .

Polynomials

- ▶ An **arithmetic circuit** over a set of variables X is a DAG s.t.:
 - ▶ internal nodes are the operations $\times$ or $+$,
 - ▶ leafs are the elements of X .
- ▶ Example for $P(X) = (x_1 + x_2 + x_3)(x_3 + x_4 + x_5)$.

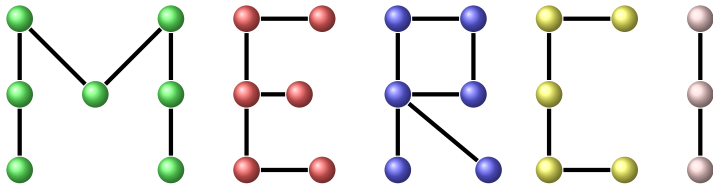


RFLCS and polynomials

- ▶ To solve RFLCS:
 - ▶ We introduce **a variable for each symbol** of the alphabet.
 - ▶ We build a **circuit** representing the subsequences of s_1 and s_2 .
 - ▶ There is a **multilinear** monomial of degree k in the circuit iff there is a common subsequence of size k **without repetitions**.
 - ▶ We can solve RFLCS in $O^*(2^k)$ time and polynomial space by a randomized algorithm.

Conclusion

- ▶ Better parameters? (e.g. when there is few duplications, difference between smallest string and the solution?)
- ▶ Prove a lower-bound on the time-complexity? Find a deterministic $O^*(2^k)$ algorithm?
- ▶ Constant factor approximable?



Paldies!

Construction of the circuit

- ▶ Via Dynamic Programming.
- ▶ Idea: The set of monomials representing the LCS of length k between $s_1[0\dots i]$ and $s_2[0\dots j]$ is the sum of:
 - ▶ the set of monomials representing the LCS of length k between $s_1[0\dots i-1]$ and $s_2[0\dots j]$,
 - ▶ the set of monomials representing the LCS of length k between $s_1[0\dots i]$ and $s_2[0\dots j-1]$,
 - ▶ the set of monomials representing the LCS of length $k-1$ between $s_1[0\dots i-1]$ and $s_2[0\dots j-1]$ times x_a if $s_1[i] = s_2[j] = a$, otherwise of the set for the LCS of length k .