LOWER AND UPPER BOUNDS FOR STRING MATCHING IN LABELLED GRAPHS

Massimo Equi

Supervisor: Veli Mäkinen
Co-supervisor: Alexandru I. Tomescu
Opponent: Nicola Prezza

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

22 June 2022

A classic problem

String Matching in Plain Text

$T = \texttt{A C G C G T C G G A A T G T C A G C T A T A A G}$

$P = \texttt{A A T G T C}$

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

A classic problem

String Matching in Plain Text

$T = $ A C G C G T C G G A A T G T C A G C T A T A A G

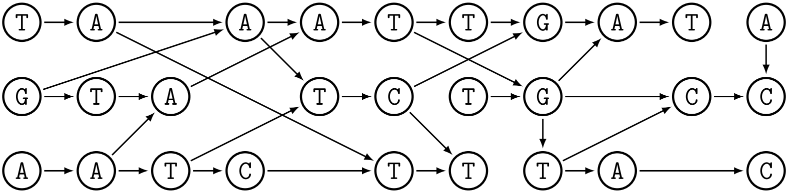
$P = $ A A T G T C

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

Our problem

String Matching in Labelled Graphs (SMLG)

SMLG for graph $G = (V, E, \ell)$ and pattern string P



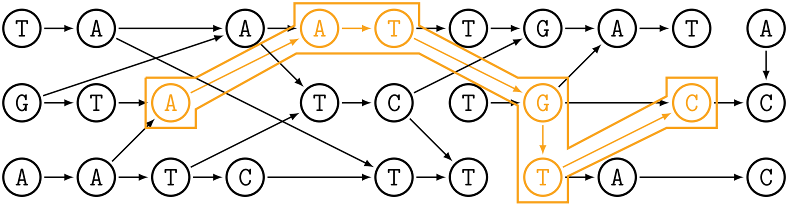
$P = \mathtt{A\ A\ T\ G\ T\ C}$

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

Our problem

String Matching in Labelled Graphs (SMLG)

SMLG for graph $G = (V, E, \ell)$ and pattern string P



$P = \mathtt{A\ A\ T\ G\ T\ C}$

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

What did we study?

Lower and upper bounds for the time complexity of SMLG

- Lower bound: **minimum** number of operations needed by an algorithm to solve the problem

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

What did we study?

Lower and upper bounds for the time complexity of SMLG

- Lower bound: **minimum** number of operations needed by an algorithm to solve the problem
- E.g. $\Omega(|V|)$, checking at least all nodes in the graph

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

What did we study?

Lower and upper bounds for the time complexity of SMLG

- Lower bound: **minimum** number of operations needed by an algorithm to solve the problem
- E.g. $\Omega(|V|)$, checking at least all nodes in the graph
- Upper bound: **maximum** number of operations needed by an algorithm to solve the problem

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

What did we study?

Lower and upper bounds for the time complexity of SMLG

- Lower bound: **minimum** number of operations needed by an algorithm to solve the problem
- E.g. $\Omega(|V|)$, checking at least all nodes in the graph
- Upper bound: **maximum** number of operations needed by an algorithm to solve the problem
- E.g. $O(|V|^{|P|})$, checking all possible paths of length $|P|$

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

What did we study?

Trivial lower and upper bounds: $\Omega(|V|)$ and $O(|V|^{|P|})$

What did we study?

Trivial lower and upper bounds: $\Omega(|V|)$ and $O(|V|^{|P|})$

Ideally, we want to make them match

- $\Omega(|V|) \rightarrow ? \leftarrow O(|V|^{|P|})$

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

What did we study?

Trivial lower and upper bounds: $\Omega(|V|)$ and $O(|V|^{|P|})$

Ideally, we want to make them match

- $\Omega(|V|) \rightarrow ? \leftarrow O(|V|^{|P|})$

If they match, the algorithm is **optimal**

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

What did we study?

Trivial lower and upper bounds: $\Omega(|V|)$ and $O(|V|^{|P|})$

Ideally, we want to make them match

- $\Omega(|V|) \rightarrow ? \leftarrow O(|V|^{|P|})$

If they match, the algorithm is **optimal**

There are many possibilities in between:

- $O(|E| + |V|)$, $O(|E| \log |E|)$, $O((|E||V|)^c)$

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

Practical Motivations

Back in the 90s: querying the Web

Practical Motivations

Back in the 90s: querying the Web

Nowadays: bioinformatics, graph databases, graph mining for network analysis

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

Practical Motivations

Back in the 90s: querying the Web

Nowadays: bioinformatics, graph databases, graph mining for network analysis

Main focus for us: Bioinformatics

- Pangenomics
- Sequence Alignment in Variation Graphs

Theoretical Motivations

State of the art for *SMLG*				
Year	**Authors**	**Graph**	**Exact/ Approximate**	**Time**
1992	Manber, Wu	DAG	approximate	$O(\|E\|\|P\| + occ \log\log \|P\|)$
1993	Akutsu	Tree	exact	$O(N)$
1995	Park, Kim	DAG	exact *(2)*	$O(N + \|E\|\|P\|)$
1997	**Amir et al.**	**general**	**exact**	$\boldsymbol{O(N + \|E\|\|P\|)}$
1997	Amir et al.	general	approximate *(1)*	NP-Hard
1997	Amir et al.	general	approximate	$O(N\|P\| \log N + \|E\|\|P\|)$
1998	Navarro	general	approximate	$O(N\|P\| + \|E\|\|P\|)$
2017	Vadaddi et al.	general	approximate	$O((\|V\| + 1)\|E\|\|P\|)$
2017	**Rautiainen, Marschall**	**general**	**approximate**	$\boldsymbol{O(N + \|E\|\|P\|)}$

Table: *(1)* errors in the graph, *(2)* matches span only one edge.

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

Our Work

Our Work

Four publications

Our Work

Four publications

- Paper I

  On the Complexity of String Matching for Graphs

  Massimo Equi, Roberto Grossi, Veli Mäkinen, Alexandru I. Tomescu

  ICALP 2019

- Paper II

  Graphs Cannot Be Indexed in Polynomial Time for Sub-quadratic Time String Matching, Unless SETH Fails

  Massimo Equi, Veli Mäkinen, Alexandru I. Tomescu

  SOFSEM 2021

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

Our Work

Four publications

- Paper I
  On the Complexity of String Matching for Graphs
  Massimo Equi, Roberto Grossi, Veli Mäkinen, Alexandru I. Tomescu
  ICALP 2019
- Paper II
  Graphs Cannot Be Indexed in Polynomial Time for Sub-quadratic Time String Matching, Unless SETH Fails
  Massimo Equi, Veli Mäkinen, Alexandru I. Tomescu
  SOFSEM 2021

They provide quadratic conditional lower bounds based on SETH

- Paper I deals with the **online** case
- Paper II deals with the **indexed** case

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

Our Work

Four publications

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

Our Work

Four publications

- Paper III

  Linear Time Construction of Indexable Founder Block Graphs

  Veli Mäkinen, Bastien Cazaux, **Massimo Equi**, Tuukka Norri, Alexandru I. Tomescu

  WABI 2020

- Paper IV

  Algorithms and Complexity on Indexing Elastic Founder Graphs

  Massimo Equi, Tuukka Norri, Jarno Alanko, Bastien Cazaux, Alexandru I. Tomescu, Veli Mäkinen

  ISAAC 2021

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

Our Work

Four publications

- Paper III
  Linear Time Construction of Indexable Founder Block Graphs
  Veli Mäkinen, Bastien Cazaux, **Massimo Equi**, Tuukka Norri, Alexandru I. Tomescu
  WABI 2020
- Paper IV
  Algorithms and Complexity on Indexing Elastic Founder Graphs
  Massimo Equi, Tuukka Norri, Jarno Alanko, Bastien Cazaux, Alexandru I. Tomescu, Veli Mäkinen
  ISAAC 2021

They provide efficient algorithms for a special class of graphs

- Paper III introduces the techniques for **Founder Block Graphs**
- Paper IV extends Paper III to **Elastic Founder Graphs**

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

Quick Complexity Background

Quick Complexity Background

Orthogonal Vectors (OV) Problem

Let $X, Y \subseteq \{0,1\}^d$ be two sets of n binary vectors of length d.

Determine whether there exist $x \in X, y \in Y$ such that $x \cdot y = 0$

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

Quick Complexity Background

Orthogonal Vectors (OV) Problem

Let $X, Y \subseteq \{0,1\}^d$ be two sets of n binary vectors of length d.

Determine whether there exist $x \in X, y \in Y$ such that $x \cdot y = 0$

$$X = \begin{pmatrix} 001 \\ \mathbf{010} \\ 011 \\ 101 \end{pmatrix} \qquad \begin{pmatrix} 010 \\ 011 \\ \mathbf{100} \\ 111 \end{pmatrix} = Y$$

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

Quick Complexity Background

Orthogonal Vectors (OV) Problem

Let $X, Y \subseteq \{0,1\}^d$ be two sets of n binary vectors of length d.

Determine whether there exist $x \in X, y \in Y$ such that $x \cdot y = 0$

$$X = \begin{pmatrix} 001 \\ \mathbf{010} \\ 011 \\ 101 \end{pmatrix} \qquad \begin{pmatrix} 010 \\ 011 \\ \mathbf{100} \\ 111 \end{pmatrix} = Y$$

Orthogonal Vectors Hypothesis (OVH)

No algorithm can solve *Orthogonal Vectors* in time $O(n^\alpha \, \mathrm{poly}(d))$, $\alpha < 2$

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

Quick Complexity Background

Orthogonal Vectors (OV) Problem

Let $X, Y \subseteq \{0,1\}^d$ be two sets of n binary vectors of length d.

Determine whether there exist $x \in X, y \in Y$ such that $x \cdot y = 0$

$$X = \begin{pmatrix} 001 \\ \mathbf{010} \\ 011 \\ 101 \end{pmatrix} \qquad \begin{pmatrix} 010 \\ 011 \\ \mathbf{100} \\ 111 \end{pmatrix} = Y$$

Orthogonal Vectors Hypothesis (OVH)

No algorithm can solve *Orthogonal Vectors* in time $O(n^{\alpha}\,\mathrm{poly}(d))$, $\alpha < 2$

CNF-SAT can be reduced to OV, thus SETH $\Rightarrow$ OVH

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

Paper I

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

Theorem

For any $\epsilon > 0$ and any DAG G, the exact SMLG problem on G cannot be solved in either $O(|E|^{1-\epsilon}\,|P|)$ or $O(|E|\,|P|^{1-\epsilon})$ time unless OVH is false.

Paper I

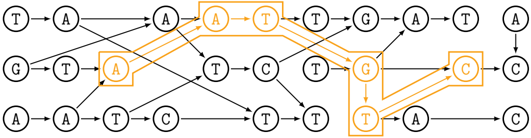
Theorem

For any $\epsilon > 0$ and any DAG G, the exact SMLG problem on G cannot be solved in either $O(|E|^{1-\epsilon}\,|P|)$ or $O(|E|\,|P|^{1-\epsilon})$ time unless OVH is false.

OV

$$\begin{pmatrix} 001 \\ \mathbf{010} \\ 011 \\ 101 \end{pmatrix} \begin{pmatrix} 010 \\ 011 \\ \mathbf{100} \\ 111 \end{pmatrix} \leq$$

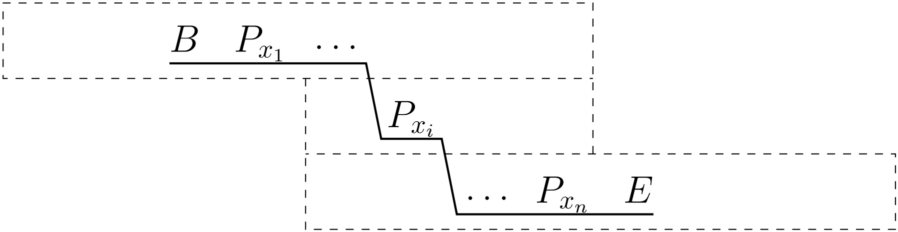
SMLG



HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

Set X $\rightarrow$ String P
Set Y $\rightarrow$ Graph G

$$\underline{B\, P_{x_1} P_{x_2}\ \ldots\ P_{x_n}\ E}$$

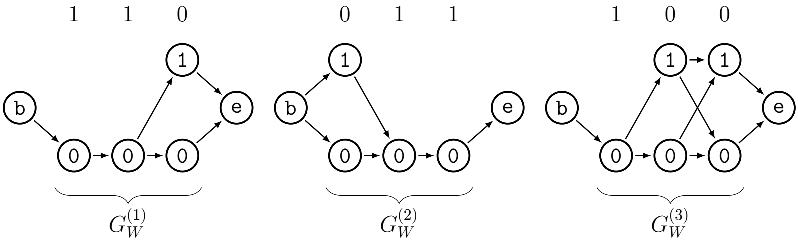


Paper I

Vectors: $x_1, \ldots, x_n \in X$, $y_1, \ldots, y_n \in Y$ Alphabet: $\Sigma = \{\mathtt{b}, \mathtt{e}, 0, 1\}$

$$X = \{\, \overset{x_1}{(100)}, \overset{x_2}{(010)}, \overset{x_3}{(110)} \,\} \to P = \mathtt{bb}\ \overset{P_{x_1}}{\mathtt{100}}\ \mathtt{e\,b}\ \overset{P_{x_2}}{\mathtt{010}}\ \mathtt{e\,b}\ \overset{P_{x_3}}{\mathtt{110}}\ \mathtt{ee}$$

$$Y = \{\, \overset{y_1}{(110)}, \overset{y_2}{(011)}, \overset{y_3}{(100)} \,\}$$



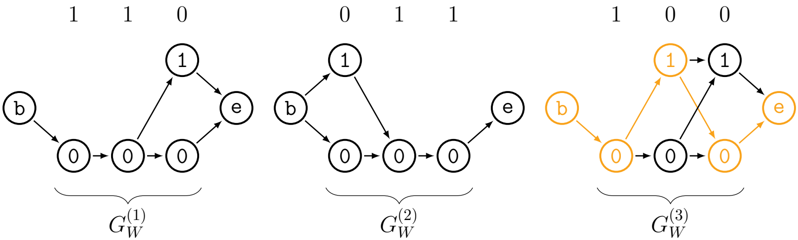
HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

Paper I

Vectors: $x_1, \ldots, x_n \in X$, $y_1, \ldots, y_n \in Y$ Alphabet: $\Sigma = \{\mathtt{b}, \mathtt{e}, 0, 1\}$

$$X = \{ \overset{x_1}{(100)}, \overset{x_2}{(010)}, \overset{x_3}{(110)} \} \rightarrow P = \mathtt{bb}\ \overset{P_{x_1}}{\mathtt{100\ e\ b}}\ \overset{P_{x_2}}{\mathtt{010\ e\ b}}\ \overset{P_{x_3}}{\mathtt{110}}\ \mathtt{ee}$$

$$Y = \{ \overset{y_1}{(110)}, \overset{y_2}{(011)}, \overset{y_3}{(100)} \}$$



$P_{x_2} = \mathtt{b\ 010\ e}$ has a match; $P_{x_3} = \mathtt{b\ 110\ e}$ has no match

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

Paper II

Paper II

Theorem

For any $\alpha, \beta, \delta > 0$, with $\beta < 1$ or $\delta < 1$, there is no algorithm preprocessing a labeled graph $G = (V, E, \ell)$ in time $O(|E|^\alpha)$ such that for any pattern string P we can solve the SMLG problem on G and P in time $O(|P| + |E|^\delta |P|^\beta)$, unless OVH is false.

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

Paper II

Theorem

For any $\alpha, \beta, \delta > 0$, with $\beta < 1$ or $\delta < 1$, there is no algorithm preprocessing a labeled graph $G = (V, E, \ell)$ in time $O(|E|^\alpha)$ such that for any pattern string P we can solve the SMLG problem on G and P in time $O(|P| + |E|^\delta |P|^\beta)$, unless OVH is false.

Linear independent-components (lic) reduction

Paper II

Theorem

For any $\alpha, \beta, \delta > 0$, with $\beta < 1$ or $\delta < 1$, there is no algorithm preprocessing a labeled graph $G = (V, E, \ell)$ in time $O(|E|^\alpha)$ such that for any pattern string P we can solve the SMLG problem on G and P in time $O(|P| + |E|^\delta |P|^\beta)$, unless OVH is false.

Linear independent-components (lic) reduction

$$\begin{array}{cc} X & Y \\ \begin{pmatrix} 001 \\ \mathbf{010} \\ 011 \\ 101 \end{pmatrix} & \begin{pmatrix} 010 \\ 011 \\ \mathbf{100} \\ 111 \end{pmatrix} \end{array} \leq_{lic} \quad \text{Any problem respecting a } \textit{lic} \text{ reduction}$$

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

Paper II

Theorem

For any $\alpha, \beta, \delta > 0$, with $\beta < 1$ or $\delta < 1$, there is no algorithm preprocessing a labeled graph $G = (V, E, \ell)$ in time $O(|E|^{\alpha})$ such that for any pattern string P we can solve the SMLG problem on G and P in time $O(|P| + |E|^{\delta}|P|^{\beta})$, unless OVH is false.

Linear independent-components (lic) reduction

$$\begin{array}{cc} X & Y \\ \begin{pmatrix} 001 \\ \mathbf{010} \\ 011 \\ 101 \end{pmatrix} & \begin{pmatrix} 010 \\ 011 \\ \mathbf{100} \\ 111 \end{pmatrix} \end{array} \leq_{lic} \quad \text{Any problem respecting a } \textit{lic} \text{ reduction}$$

lic reduction: P depends only on X, G depends only on Y

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

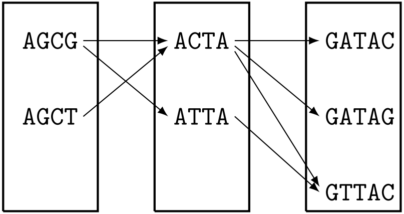
Paper III

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

From an MSA[m, n] we can build a Founder Block Graph (FBG)

AGCG ACTA GATAC
AGCT ACTA GATAG
AGCG ATTA GTTAC
AGCT ACTA GTTAC

$\rightarrow$



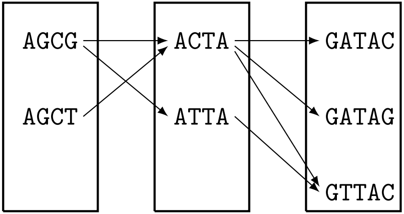
HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

Paper III

From an MSA$[m, n]$ we can build a Founder Block Graph (FBG)

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

$\rightarrow$

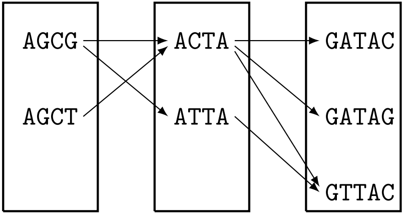


- $(v, w) \in E \Leftrightarrow \ell(v)\ell(w)$ matches an MSA row at the corresponding segments

From an MSA[m, n] we can build a Founder Block Graph (FBG)

```
AGCG|ACTA|GATAC
AGCT|ACTA|GATAG
AGCG|ATTA|GTTAC
AGCT|ACTA|GTTAC
```

$\rightarrow$



- $(v, w) \in E \Leftrightarrow \ell(v)\ell(w)$ matches an MSA row at the corresponding segments
- **Repeat-free** property: every node label $\ell(v)$ appears nowhere else in the graph

Paper III

Repeat-free Founder Block Graphs

Repeat-free Founder Block Graphs

- Can be constructed from an MSA optimising different functions

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

Paper III

Repeat-free Founder Block Graphs

- Can be constructed from an MSA optimising different functions
- Allow indexing in polynomial time
- Allow queries in time $O(|P| \log \sigma)$

Repeat-free Founder Block Graphs

- Can be constructed from an MSA optimising different functions
- Allow indexing in polynomial time
- Allow queries in time $O(|P| \log \sigma)$

Indexing algorithm

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

Paper III

Repeat-free Founder Block Graphs

- Can be constructed from an MSA optimising different functions
- Allow indexing in polynomial time
- Allow queries in time $O(|P| \log \sigma)$

Indexing algorithm

- Aho-Corasick automaton of the node labels

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

Paper III

Repeat-free Founder Block Graphs

- Can be constructed from an MSA optimising different functions
- Allow indexing in polynomial time
- Allow queries in time $O(|P| \log \sigma)$

Indexing algorithm

- Aho-Corasick automaton of the node labels
- Forward and a backward trie for each block

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

Paper III

Repeat-free Founder Block Graphs

- Can be constructed from an MSA optimising different functions
- Allow indexing in polynomial time
- Allow queries in time $O(|P| \log \sigma)$

Indexing algorithm

- Aho-Corasick automaton of the node labels
- Forward and a backward trie for each block

Querying algorithm

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

Paper III

Repeat-free Founder Block Graphs

- Can be constructed from an MSA optimising different functions
- Allow indexing in polynomial time
- Allow queries in time $O(|P| \log \sigma)$

Indexing algorithm

- Aho-Corasick automaton of the node labels
- Forward and a backward trie for each block

Querying algorithm

- **Key property**: a match spanning at least three nodes is "**unique**"

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

Paper III

Repeat-free Founder Block Graphs

- Can be constructed from an MSA optimising different functions
- Allow indexing in polynomial time
- Allow queries in time $O(|P| \log \sigma)$

Indexing algorithm

- Aho-Corasick automaton of the node labels
- Forward and a backward trie for each block

Querying algorithm

- **Key property**: a match spanning at least three nodes is "**unique**"
- The Aho-Corasick automaton locates a matching node

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

Paper III

Repeat-free Founder Block Graphs

- Can be constructed from an MSA optimising different functions
- Allow indexing in polynomial time
- Allow queries in time $O(|P| \log \sigma)$

Indexing algorithm

- Aho-Corasick automaton of the node labels
- Forward and a backward trie for each block

Querying algorithm

- **Key property**: a match spanning at least three nodes is "**unique**"
- The Aho-Corasick automaton locates a matching node
- Extend such match using the tries

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

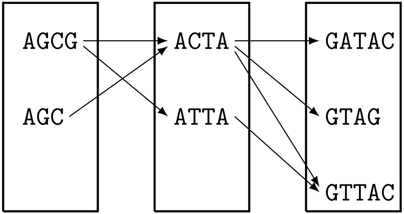
Paper IV

Paper IV

From an MSA$[m, n]$ we can build an EFG $G = (\ell, V, E)$

```
A G C G | A C T A | G A T A C
A G C - | A C T A | G - T A G
A G C G | A T T A | G T T A C
A G C - | A C T A | G T T A C
```

$\rightarrow$

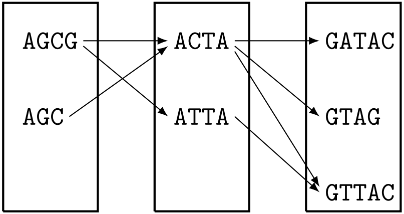


Paper IV

From an MSA$[m, n]$ we can build an EFG $G = (\ell, V, E)$

```
A G C G|A C T A|G A T A C
A G C -|A C T A|G - T A G
A G C G|A T T A|G T T A C
A G C -|A C T A|G T T A C
```

$\rightarrow$



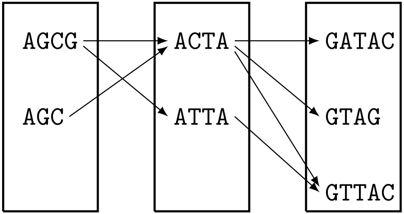
- $(v, w) \in E \Leftrightarrow \ell(v)\ell(w)$ matches an MSA row at the corresponding segments

Paper IV

From an MSA$[m, n]$ we can build an EFG $G = (\ell, V, E)$

```
AGCG|ACTA|GATAC
AGC-|ACTA|G-TAG
AGCG|ATTA|GTTAC
AGC-|ACTA|GTTAC
```

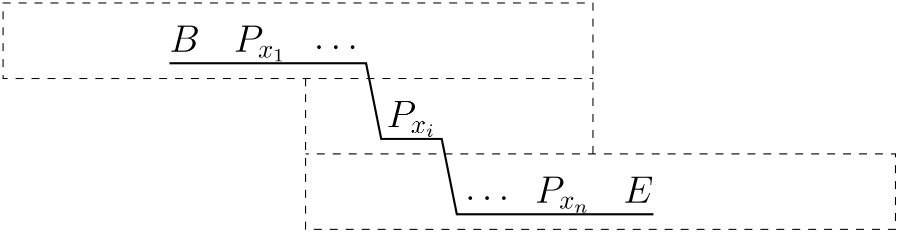
$\rightarrow$



- $(v, w) \in E \Leftrightarrow \ell(v)\ell(w)$ matches an MSA row at the corresponding segments
- **Semi-repeat-free** property: every $\ell(v)$ can appear only as a prefix of $\ell(w)$, where v and w belong to the same block

Set $X \rightarrow$ Query string P
Set $Y \rightarrow$ Graph G

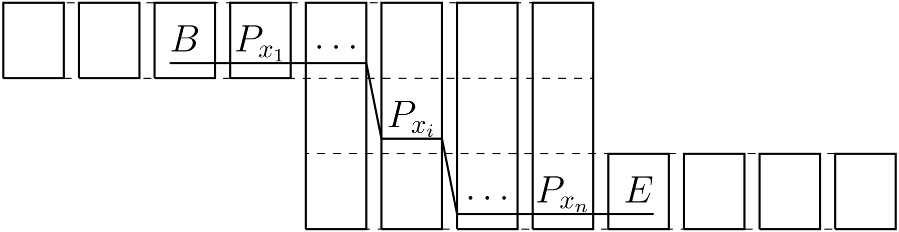
$$\underline{B\, P_{x_1} P_{x_2} \,\ldots\, P_{x_n}\, E}$$



Paper IV

Set X → Query string P
Set Y → Graph G

$$\underline{B\,P_{x_1}P_{x_2}\;\ldots\;P_{x_n}\;E}$$



HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

Paper IV

We refine the previous techniques to achieve the same results also for the gapped case

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

Paper IV

We refine the previous techniques to achieve the same results also for the gapped case

Semi-repeat-free EFGs

- Can be constructed from a **gapped** MSA optimising different functions

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

Paper IV

We refine the previous techniques to achieve the same results also for the gapped case

Semi-repeat-free EFGs

- Can be constructed from a **gapped** MSA optimising different functions
- Allow indexing in polynomial time
- Allow querying in time $O(|P| \log \sigma)$

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

Conclusions and Open Questions

This thesis

- **Paper I**: SMLG cannot be solved in less than $O(|E||P|)$ time
- **Paper II**: SMLG cannot be solved in less than $O(|E||P|)$ time even after polynomial indexing
- **Paper III**: SMLG on FBGs can be solved with query time $O(|P|)$, after polynomial time indexing
- **Paper IV**: SMLG on general EFGs respects the lower bound, (semi)-repeat-free EFGs give similar guarantees of FBG

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

Conclusions and Open Questions

This thesis

- **Paper I**: SMLG cannot be solved in less than $O(|E||P|)$ time
- **Paper II**: SMLG cannot be solved in less than $O(|E||P|)$ time even after polynomial indexing
- **Paper III**: SMLG on FBGs can be solved with query time $O(|P|)$, after polynomial time indexing
- **Paper IV**: SMLG on general EFGs respects the lower bound, (semi)-repeat-free EFGs give similar guarantees of FBG

Open Questions

- Non-repeat-free FBGs
- Trade-offs for a more general class of graphs, e.g. $O(|E|^2)$ indexing, $O(|P| \log |P|)$ queries
- SMLG under other models of computation, e.g. quantum computing

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

OV Indexing Lower Bound

(N, M)-OV

OV over sets X and Y , where $|X| = N$ and $|Y| = M$

We reduce OV to (N, M)-OV

- (X_i, Y_j) is a (N, M)-OV instance
- Time to index: $O\left(d^{O(1)}|X_i|^{\alpha}\right)$
- Time to query: $O\left(d^{O(1)}|X_i|^{\delta}|Y_j|^{\beta}\right)$

Total time, for all instances

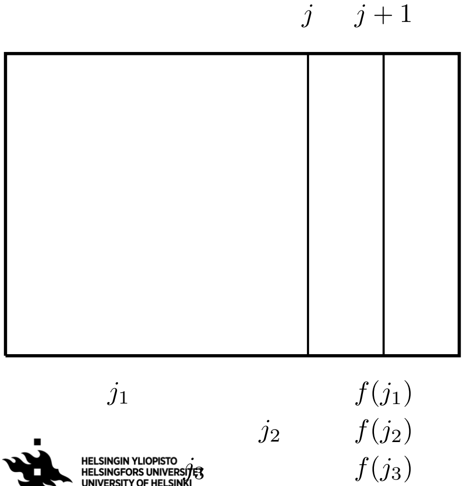
- $O\left(d^{O(1)}\left(N^{\alpha-1}n + N^{\delta-1}M^{\beta-1}n^2\right)\right)$

There exist α, β, δ that lead to time $O(n^{2-\epsilon})$ for OV

$$\begin{array}{c} X \\ X_1 \left\{ \begin{array}{c} x_1 \\ \vdots \\ x_N \end{array} \right. \\ X_2 \left\{ \begin{array}{c} x_{N+1} \\ \vdots \\ x_{2N} \end{array} \right. \\ \vdots \\ X_{\lceil \frac{n}{N} \rceil} \left\{ \begin{array}{c} x_{n-N+1} \\ \vdots \\ x_n \end{array} \right. \end{array} \qquad \begin{array}{c} Y \\ \left. \begin{array}{c} y_1 \\ \vdots \\ y_M \end{array} \right\} Y_1 \\ \left. \begin{array}{c} y_{M+1} \\ \vdots \\ y_{2M} \end{array} \right\} Y_2 \\ \vdots \\ \left. \begin{array}{c} y_{n-M+1} \\ \vdots \\ y_n \end{array} \right\} Y_{\lceil \frac{n}{M} \rceil} \end{array}$$

HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

Construction of EFG from MSA



`for every` j:
  `while` $j = f(j_x)$:
    `score` $\leftarrow$ max(`score`, $\#blocks(j_x) + 1$);
    $x \leftarrow x + 1$;
  $\#blocks(j) \leftarrow$`score`;

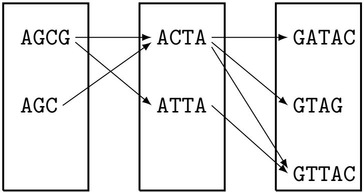
$\#block(j_1) + 1$

$\#block(j_2) + 1$

$\#block(j_3) + 1$

Indexing Elastic Founder Graphs

Construct string $D = \Pi_{i \in \{1,2,\ldots,b\}} \Pi_{v \in V^i, (v,w) \in E}\, \ell^{-1}(w)\ell^{-1}(v)$



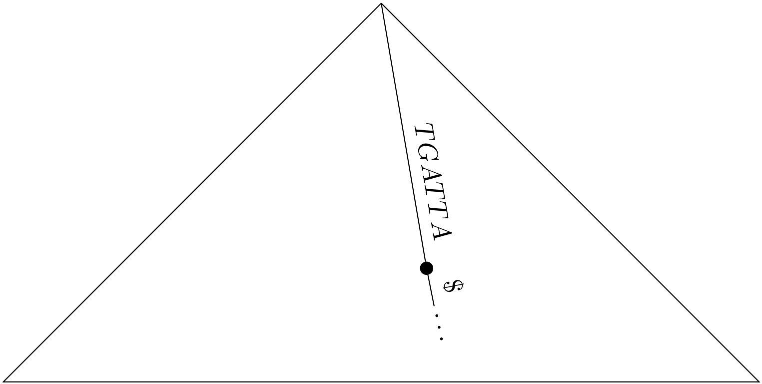
$D = ATCAGCGA\$$
$ATTAGCGA\$$
$ATCACGA\$$
$CATAGATCA\$$
$GATGATCA\$$
$CATTGATCA\$$
$CATTGATTA\$$

Build the suffix tree of D

Polynomial time in $|D|$, $O(|D| \log |D|)$ bits of space

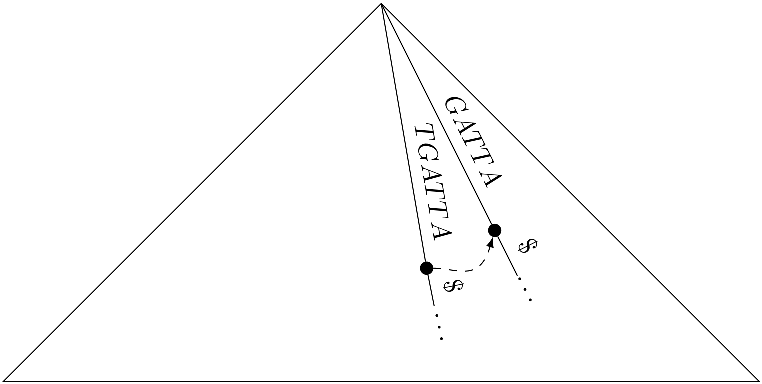
Index Elastic Founder Graphs

Query string $Q = CGATTAGT$



Index Elastic Founder Graphs

Query string $Q = CGATTAGT$



Index Elastic Founder Graphs

Query string $Q = CGATTAGT$