# Parameterized Complexity of Superstring Problems*

Ivan Bliznets† Fedor V. Fomin‡ Petr A. Golovach ‡ Nikolay Karpov †
Alexander S. Kulikov † Saket Saurabh ‡§

May 6, 2016

**Abstract**

In the Shortest Superstring problem we are given a set of strings $S = \{s_1, \ldots, s_n\}$ and integer $\ell$ and the question is to decide whether there is a superstring $s$ of length at most $\ell$ containing all strings of $S$ as substrings. We obtain several parameterized algorithms and complexity results for this problem.

In particular, we give an algorithm which in time $2^{\mathcal{O}(k)}\,\mathrm{poly}(n)$ finds a superstring of length at most $\ell$ containing at least $k$ strings of $S$. We complement this by a lower bound showing that such a parameterization does not admit a polynomial kernel up to some complexity assumption. We also obtain several results about "below guaranteed values" parameterization of the problem. We show that parameterization by compression admits a polynomial kernel while parameterization "below matching" is hard.

## 1 Introduction

We consider the Shortest Superstring problem defined as follows:

| Shortest Superstring |
|---|
| **Input:** A set of $n$ strings $S = \{s_1, \ldots, s_n\}$ over an alphabet $\Sigma$ and a non-negative integer $\ell$. |
| **Question:** Is there a string $s$ of length at most $\ell$ containing all strings from $S$ as substrings? |

This is a well-known NP-complete problem [11] with a range of practical applications from DNA assembly [8] to data compression [10]. Due to this fact approximation algorithms for it are widely studied. The currently best known approximation guarantee $2\frac{11}{23}$ is due to Mucha [17]. At the same time the best known exact algorithms run in roughly $2^n$ steps and are known for more than 50 years already. More precisely, using known algorithms for the Traveling Salesman problem, Shortest Superstring can be solved either in time $\mathcal{O}^*(2^n)$ and the same space by dynamic programming over subsets [2, 14] or in time $\mathcal{O}^*(2^n)$ and only polynomial space by inclusion-exclusion [15, 16] (here, $\mathcal{O}^*(\cdot)$ hides factors that are polynomial in the input length, i.e., $\sum_{i=1}^{n} |s_i|$). Such algorithms can only be used in practice to solve instances of very moderate size. Stronger upper bounds are known

*The research leading to these results has received funding from the Government of the Russian Federation (grant 14.Z50.31.0030) and the grant of the President of Russian Federation (MK-6550.2015.1). A preliminary version of the paper appeared in the proceedings of CPM 2015.

†St. Petersburg Department of Steklov Institute of Mathematics of the Russian Academy of Sciences.
‡Department of Informatics, University of Bergen, Norway.
§Institute of Mathematical Sciences, Chennai, India.

for a special case when input strings have bounded length [12, 13]. There are heuristic methods for solving Traveling Salesman, and hence also Shortest Superstring, they are efficient in practice, however have no efficient provable guarantee on the running time (see, e.g., [1]).

In this paper, we study the Shortest Superstring problem from the parameterized complexity point of view. This field studies the complexity of computational problems with respect not only to input size, but also to some additional parameters and tries to identify parameters of input instances that make the problem tractable. Interestingly, prior to our work, except observations following from the known reductions to Traveling Salesman, not much about the parameterized complexity of Shortest Superstring was known. We refer to the survey of Bulteau et al. [4] for a nice overview of known results on parameterized algorithms and complexity of strings problems. Thus our work can be seen as the first non-trivial step towards the study of this interesting and important problem from the perspective of parameterized complexity.

**Our results** In this paper we study two types of parameterization for Shortest Superstring and present two types of results. The first set of results concerns "natural" parameterizations of the problem. We consider the following generalization of Shortest Superstring:

| Partial Superstring |
| --- |
| **Input:** A collection (multiset) of strings $S$ over an alphabet $\Sigma$, and non-negative integers $k, \ell$. |
| **Question:** Is there a string $s$ of length at most $\ell$ such that $s$ is a superstring of a collection of at least $k$ strings $S' \subseteq S$? |

If $k = |S|$, then this is Shortest Superstring. Notice that $S$ can contain copies of the same string and a string of $S$ can be a substring of another string of the collection. For Shortest Superstring, such cases could be easily avoided, but it is natural to assume that we have such possibilities for Partial Superstring.

Here we show that Partial Superstring is fixed parameter tractable (FPT) when parameterized by $k$ or $\ell$. We complement this result by showing that it is unlikely that the problem admits a polynomial kernel with respect to these parameters.

The second set of results concerns "below guaranteed value" parameterizations. Note that an obvious (non-optimal) superstring of $S = \{s_1, \ldots, s_n\}$ is a string of length $\sum_{i=1}^{n} |s_i|$ formed by concatenating all strings from $S$. For a superstring $s$ of $S$ the value $\sum_{i=1}^{n} |s_i| - |s|$ is called the *compression of $s$ with respect to $S$*. We first show that it is FPT with respect to $r$ to check whether one can achieve a compression at least $r$ by constructing a kernel of size $\mathcal{O}(r^4)$. We complement this result by a hardness result about a "stronger" parameterization. Let us partition $n$ input strings into $n/2$ pairs such that the sum of the $n/2$ resulting overlaps is maximized. Such a partition can be found in polynomial time by constructing a maximum weight matching in an auxiliary graph. Then this total overlap provides a lower bound on the maximum compression (or, equivalently, an upper bound on the length of a shortest superstring). We show that deciding whether at least one additional symbol can be saved beyond the maximum weight matching value is already NP-complete.

## 2 Basic definitions and preliminaries

**Strings.** Let $s$ be a string. By $|s|$ is denoted the *length* of $s$. By $s[i]$, where $1 \leq i \leq |s|$, is denoted the $i$-th symbol of $s$, and $s[i, j] = s[i] \ldots s[j]$ for $1 \leq i \leq j \leq |s|$. We assume that $s[i, j]$ is the empty string if $i > j$. A nonempty string $s'$ is a *substring* of $s$ if $s' = s[i, j]$ for some $1 \leq i \leq j \leq |s|$; the

empty string is a substring of any string. A string $s'$ is a *proper substring* of $s$ if $s'$ is a substring of $s$ and $s' \neq s$. Respectively, if $s'$ is a (proper) substring of $s$, then $s$ is a (proper) *superstring* of $s'$. We write $s' \subseteq s$ ($s' \supseteq s$) to denote that $s'$ is a substring (superstring) of $s$ and we write $s' \subset s$ and $s' \supset s$ to denote proper sub- and superstrings. We denote $\mathrm{prefix}_i(s) = s[1,i]$ and $\mathrm{suffix}_i(s) = s[|s|-i+1,|s|]$ the *$i$-th prefix* and *$i$-th suffix* of $s$ respectively for $i \in \{1,\ldots,|s|\}$; $\mathrm{prefix}_0(s) = \mathrm{suffix}_0(s)$ is the empty string. For a collection of strings $S$, a string $t$ is a superstring of $S$ if $t$ is a superstring of each string in $S$. The *compression measure* of a superstring $t$ of a collection of strings $S$ is $\sum_{x\in S}|x| - |t|$. We denote by $uv$ the *concatenation* of $u$ and $v$. The *overlap* of two strings $u$ and $v$, denoted by $\mathrm{overlap}(u,v)$, is the longest suffix of $u$ which is also a prefix of $v$.

For a sequence of strings $s_1,\ldots,s_n$, we define the function $\sigma$ that maps $s_1,\ldots,s_n$ to a string as follows:

$$\sigma(s_1,\ldots,s_n) = \mathrm{prefix}_{r_1}(s_1)\mathrm{prefix}_{r_2}(s_2)\cdots\mathrm{prefix}_{r_{n-1}}(s_{n-1})s_n$$

where $r_i = |s_i| - |\mathrm{overlap}(s_i,s_{i+1})|$. It is easy to see that $\sigma(s_1,\ldots,s_n)$ is a superstring of $s_1,\ldots,s_n$ of length $\sum_{i=1}^{n}|s_i| - \sum_{i=1}^{n-1}|\mathrm{overlap}(s_i,s_{i+1})|$. We need the following folklore property of superstrings.

**Lemma 1.** *Let $s$ be a superstring of a collection of strings $S = \{s_1,\ldots,s_n\}$ that does not contain a pair of distinct strings $s_i, s_j$ such that $s_i \subset s_j$. Let also $s_i = s[p_i,q_i]$ for $i \in [n]$ and assume that $p_i < p_j$ for $i < j$. Then $\sigma(s_1,\ldots,s_n)$ is a superstring of $S$ of length at most $|s|$.*

This lemma allows to consider the shortest superstring problem for a set $S = \{s_1,\ldots,s_n\}$ with no $s_i \subset s_j$ as finding a permutation $\pi$ of the given $n$ strings minimizing $\sum_{i=1}^{n}|s_i| - \sum_{i=1}^{n-1}|\mathrm{overlap}(s_{\pi_i},s_{\pi_{i+1}})|$ or, equivalently, maximizing $\sum_{i=1}^{n-1}|\mathrm{overlap}(s_{\pi_i},s_{\pi_{i+1}})|$. For the case when $S$ is allowed to contain a string that is a proper substring of another string from $S$, we need a generalized definition. For a sequence of strings $s_1,\ldots,s_n$, *the concatenation with overlaps* $\Delta(s_1,\ldots,s_n)$ is defined as follows: if $s_{i+1} \subset s_i$ and $i$ is the minimal such an index then $\Delta(s_1,\ldots,s_i,s_{i+1},\ldots,s_n) = \Delta(s_1,\ldots,s_i,s_{i+2},\ldots,s_n)$; if there is no such $i$ then $\Delta(s_1,\ldots,s_n) = \sigma(s_1,\ldots,s_n)$. The following lemma is a counterpart of Lemma 1.

**Lemma 2.** *Let $t$ be a superstring of a collection of strings $S = \{s_1,\ldots,s_n\}$ then $|t| \geq \min_{\pi\in S_n}|\,\Delta(s_{\pi_1},\ldots,s_{\pi_n})|$.*

*Proof.* This can be proved by induction on the size of $S$. The base case $|S| = 1$ is clear. If no $s_i \in S$ is a substring of $s_j \in S$ then the statement follows from Lemma 1. Otherwise assume without loss of generality that $s_n \subset s_i$. By the induction hypothesis, there exists a permutation $\pi \in S_{n-1}$ such that $|t| \geq |\,\Delta(s_{\pi_1},\ldots,s_i,\ldots,s_{\pi_{n-1}})|$. Inserting $s_n$ after $s_i$ in $\pi$ gives us a required permutation from $S_n$:

$$|t| \geq |\,\Delta(s_{\pi_1},\ldots,s_i,\ldots,s_{\pi_{n-1}})| = |\,\Delta(s_{\pi_1},\ldots,s_i,s_n,\ldots,s_{\pi_{n-1}})|\,.$$

□

**Graphs.** We consider finite directed and undirected graphs without loops or multiple edges. The vertex set of a (directed) graph $G$ is denoted by $V(G)$, the edge set of an undirected graph and the arc set of a directed graph $G$ is denoted by $E(G)$. To distinguish edges and arcs, the edge with two end-vertices $u,v$ is denoted by $\{u,v\}$, and we write $(u,v)$ for the corresponding arc. For an arc

$e = (u, v)$, $v$ is the *head* of $e$ and $u$ is the tail. Let $G$ be a directed graph. For a vertex $v \in V(G)$, we say that $u$ is an *in-neighbor* of $v$ if $(u, v) \in E(G)$. The set of all in-neighbors of $v$ is denoted by $N_G^-(v)$. The *in-degree* $d_G^-(v) = |N_G^-(v)|$. Respectively, $u$ is an *out-neighbor* of $v$ if $(v, u) \in E(G)$, the set of all out-neighbors of $v$ is denoted by $N_G^+(v)$, and the *out-degree* $d_G^+(v) = |N_G^+(v)|$. For a directed graph $G$, a (directed) *trail* of *length* $k$ is a sequence $v_0, e_1, v_1, e_2, \ldots, e_k, v_k$ of vertices and arcs of $G$ such that $v_0, \ldots, v_k \in V(G)$, $e_1, \ldots, e_k \in E(G)$, the arcs $e_1, \ldots, e_k$ are pairwise distinct, and for $i \in \{1, \ldots, k\}$, $e_i = (v_{i-1}, v_i)$. We omit the word "directed" if it does not create a confusion. Slightly abusing notations we often write a trail as a sequence of its vertices $v_0, \ldots, v_k$ or arcs $e_1, \ldots, e_k$. If $v_0, \ldots, v_k$ are pairwise distinct, then $v_0, \ldots, v_k$ is a (directed) path. Recall that a path of length $|V(G)| - 1$ is a *Hamiltonian* path. For an undirected graph $G$, a set $U \subseteq V(G)$ is a *vertex cover* of $G$ if for any edge $\{u, v\}$ of $G$, $u \in U$ or $v \in U$. A set of edges $M$ with pairwise distinct end-vertices is a *matching*.

We consider the following auxiliary problem:

| Long Trail |
| --- |
| **Input:** A directed graph $G$ and a non-negative integer $\ell$. |
| **Question:** Is there a trail of length at least $\ell$ in $G$? |

**Lemma 3.** Long Trail *is* NP*-complete. In particular, the problem is* NP*-complete if* $\ell = |V(G)| - 1$.

*Proof.* We reduce the Hamiltonian Path problem for directed graphs that is well known to be NP-complete (see, e.g., [11]). Let $G$ be a directed graph with $n$ vertices. We construct the graph $G'$ as follows.

- For each $v \in V(G)$, construct two vertices $v^-, v^+$ and an arc $(v^-, v^+)$.
- For each $(u, v) \in E(G)$, construct an arc $(u^+, v^-)$.
- Construct two vertices $s, t$ and for each $v \in V(G)$, construct arcs $(s, v^-), (v^+, t)$.

We claim that $G'$ has a trail of length at least $2n+1 = |V(G')|-1$ if and only if $G$ has a Hamiltonian path.

Suppose that $G$ has a Hamiltonian path $v_1, \ldots, v_n$. Then the trail $s, v_1^-, v_1^+, v_2^-, v_2^+, \ldots, v_n^-, v_n^+, t$ in $G'$ has length $2n + 1$.

Assume that $G'$ has a trail $P$ of length at least $2n+1$. Without loss of generality we can assume that $s$ is the first vertex of $P$ and $t$ is the last. To see it, suppose that $x \neq s$ is the first vertex of $P$. Notice that $s$ is not in $P$, because $d_{G'}^-(s) = 0$. If $x = v^-$ for $v \in V(G)$, then we can consider the extended trail $s, (s, x), P$. If $x = v^+$ for $v \in V(G)$, then let $u^-$ be the next vertex in $P$ after $x$. We consider the trail $P'$ obtained from $P$ by the replacement of $x$ and $(x, u^-)$ by $s$ and $(s, u^-)$ respectively. Clearly, $P'$ has the same length as $P$. By the symmetric arguments, we obtain that we can assume that $t$ is the last vertex of $P$. We have that any vertex of $G'$ occurs exactly once in $P$, because $d_{G'}^-(s) = d_{G'}^+(t) = 0$ and $d_{G'}^+(v^-) = d_{G'}^-(v^+) = 1$ for $v \in V(G)$. Moreover, for each vertex $v \in V(G)$, $(v^-, v^+)$ in $P$, because $v^-$ is the unique in-neighbor of $v^+$ and $v^+$ is the unique out-neighbor of $v^-$ respectively for $v \in V(G)$. Hence, $P$ can be written as $s, v_1^-, v_1^+, v_2^-, v_2^+, \ldots, v_n^-, v_n^+, t$ for $v_1, \ldots, v_n \in V(G)$. It remains to observe that $v_1, \ldots, v_n$ is a Hamiltonian path in $G$. □

**Circuits.** An arithmetic circuit is a directed acyclic graph whose nodes of in-degree zero are labeled with variables $x_1, \ldots, x_n$ and are called inputs while nodes of non-zero in-degree are labeled with

+ (summation) and $\times$ (multiplication) and are called gates. Each gate of a circuit computes a polynomial of $x_1, \ldots, x_n$. One gate of a circuit is also designated as an output gate and we say that a circuit computes a polynomial of this gate. The size of a circuit is its number of gates.

**Parameterized Complexity.** Parameterized complexity is a two dimensional framework for studying the computational complexity of a problem. One dimension is the input size and another one is a parameter. We refer to the recent books of Cygan et al. [5] and Downey and Fellows [6] for detailed introductions to parameterized complexity.

Formally, a parameterized problem $\mathcal{P} \subseteq \Sigma^* \times \mathbb{N}$, where $\Sigma$ is a finite alphabet, i.e., an instance of $\mathcal{P}$ is a pair $(I, k)$ for $I \in \Sigma^*$ and $k \in \mathbb{N}$, where $I$ is an input and $k$ is a parameter. It is said that a problem is *fixed-parameter tractable* (or FPT), if it can be solved in time $f(k) \cdot |I|^{O(1)}$ for some function $f$. A *kernelization* for a parameterized problem is a polynomial algorithm that maps each instance $(I, k)$ to an instance $(I', k')$ such that

i) $(I, k)$ is a yes-instance if and only if $(I', k')$ is a yes-instance of the problem, and

ii) the size of $I'$ and $k'$ are bounded by $f(k)$ for a computable function $f$.

The output $(I', k')$ is called a *kernel*. The function $f$ is said to be the *size* of a kernel. Respectively, a kernel is *polynomial* if $f$ is polynomial.

While a decidable parameterized problem is FPT if and only if it has a kernel, it is widely believed that not all FPT problems have polynomial kernels (see [5] for details). In particular, Bodlaender, Jansen and Kratsch [3] introduced the cross-composition technique that allow to show that a parameterized problem has no polynomial kernel unless NP $\subseteq$ coNP /poly. To introduce this technique, we need some definitions.

Let $\Sigma$ be a finite alphabet. An equivalence relation $\mathcal{R}$ on the set of strings $\Sigma^*$ is called a *polynomial equivalence relation* if the following two conditions hold:

i) there is an algorithm that given two strings $x, y \in \Sigma^*$ decides whether $x$ and $y$ belong to the same equivalence class in time polynomial in $|x| + |y|$,

ii) for any finite set $S \subseteq \Sigma^*$, the equivalence relation $\mathcal{R}$ partitions the elements of $S$ into a number of classes that is polynomially bounded in the size of the largest element of $S$.

Let $L \subseteq \Sigma^*$ be a language, let $\mathcal{R}$ be a polynomial equivalence relation on $\Sigma^*$, and let $\mathcal{P} \subseteq \Sigma^* \times \mathbb{N}$ be a parameterized problem. An *OR-cross-composition of $L$ into $\mathcal{P}$* (with respect to $\mathcal{R}$) is an algorithm that, given $t$ instances $x_1, x_2, \ldots, x_t \in \Sigma^*$ of $L$ belonging to the same equivalence class of $\mathcal{R}$, takes time polynomial in $\sum_{i=1}^{t} |x_i|$ and outputs an instance $(y, k) \in \Sigma^* \times \mathbb{N}$ such that:

i) the parameter value $k$ is polynomially bounded in $\max\{|x_1|, \ldots, |x_t|\} + \log t$,

ii) the instance $(y, k)$ is a yes-instance for $\mathcal{P}$ if and only if at least one instance $x_i$ is a yes-instance for $L$ and $i \in \{1, \ldots, t\}$.

It is said that *$L$ OR-cross-composes into $\mathcal{P}$* if a cross-composition algorithm exists for a suitable relation $\mathcal{R}$.

The following theorem was proved by Bodlaender, Jansen and Kratsch [3].

**Theorem 1** ([3])**.** *If an* NP*-hard language $L$ OR-cross-composes into the parameterized problem $\mathcal{P}$, then $\mathcal{P}$ does not admit a polynomial kernelization unless* NP $\subseteq$ coNP /poly*.*

## 3 FPT-algorithms for Partial Superstring

In this section we show that Partial Superstring is FPT, when parameterized by $k$ or $\ell$. For technical reasons, we consider the following variant of the problem with weights:

Partial Weighted Superstring
**Input:** A collection of strings $S$ over an alphabet $\Sigma$ with a weight function $w\colon S \to \mathbb{N}_0$, and non-negative integers $k, \ell$ and $W$.
**Question:** Is there a string $s$ of length at most $\ell$ such that $s$ is a superstring of a collection of $k$ strings $S' \subseteq S$ with $w(S') = \sum_{s \in S'} w(s) \geq W$?

Clearly, if $w \equiv 1$ and $W = k$, then we have the Partial Superstring problem.

Below we present an algorithm for Partial Superstring and Partial Weighted Superstring based on the following multilinear monomial detection tests.

**Theorem 2** ([18]). *Let $P(x_1, \ldots, x_n)$ be a polynomial of degree at most $k$, represented by an arithmetic circuit of size $s(n)$ with $+$ gates (of unbounded in-degree), $\times$ gates (of in-degree two), and no scalar multiplications. There exists a randomized algorithm with running time $\mathcal{O}^*(2^k s(n))$ that given $P$ outputs yes with probability at least $\frac{1}{5}$ if there is a multilinear monomial in sum-product expansion of $P$, and always outputs no if there is no such monomial.*

**Theorem 3** ([9]). *Let $P(x_1, \ldots, x_n)$ be a polynomial of degree at most $k$, represented by an arithmetic circuit of size $s(n)$ with $+$ gates (of unbounded in-degree), $\times$ gates (of in-degree two), and no scalar multiplications; $w\colon 2^{\{1,\ldots,n\}} \to \mathbb{N}$ be an additive weight function and $W$ be the maximum value of $w$. There exists a deterministic algorithm with running time $\mathcal{O}(3.8408^k s(n) n \log^2 n \log W)$ that given $P$ outputs minimum possible weight of multilinear monomial is such monomials exist.*

**Theorem 4.** Partial Superstring *can be solved in time $\mathcal{O}^*(2^k)$ by a randomized algorithm that outputs "no" with probability 1 if there is no superstring and outputs a superstring with probability at least $1/5$ if it exists.* Partial Weighted Superstring *can be solved in time $\mathcal{O}^*(3.8408^k)$ by a deterministic algorithm.*

*Proof.* We will assume that $\ell < \sum_{i=1}^{n} |s_i|$ as otherwise the problem is trivial.

We first obtain the algorithmic result for Partial Superstring. Due to Theorem 2, it suffices to construct a polynomial size circuit $C$ that computes a polynomial $P_k$ containing a multilinear monomial if and only if there is a string of length at most $l$ that contains $k$ strings from $S$ as substrings.

The required polynomial contains a monomial for each possible concatenation with overlaps of $k$ strings of total length at most $\ell$:

$$P_k(x_1, \ldots, x_n) = \sum_{(i_1,\ldots,i_k) \in [n]^k} [|\,\Delta(s_{i_1}, \ldots, s_{i_k})| \leq \ell] \cdot x_{i_1} x_{i_2} \cdots x_{i_k}$$

(note that we do not require $i_1, \ldots, i_k$ to be pairwise different). Slightly abusing notation in this definition, we use $[\cdot]$ to denote two different things: for an integer $m$, by $[m]$ we denote the set $\{1, 2, \ldots, m\}$; for a Boolean predicate $P$, by $[P]$ we denote the characteristic function: it is equal to 1 if $P$ is true and is equal to 0 otherwise. By Lemma 2, there is a superstring of length at most $\ell$ of $k$ strings if and only if $P_k(x_1, \ldots, x_n)$ contains a multilinear monomial.

We now describe a polynomial size arithmetic circuit $C$ computing the polynomial $P_k$. For this, we introduce the following auxiliary polynomials. For $1 \le j \le n$, $1 \le t \le n$, and $0 \le w \le \ell$, let $Q(t, j, w)$ be a polynomial of variables $x_1, \ldots, x_n$ containing a monomial for each concatenation with overlaps of $t+1$ strings starting with $s_j$ of total length at most $w$:

$$Q(t, j, w) = \sum_{(i_1,\ldots,i_t)\in[n]^t} [|\,\Delta(s_j, s_{i_1}, \ldots, s_{i_t})| \le w] \cdot x_j x_{i_1} x_{i_2} \cdots x_{i_t} \tag{1}$$

Note that $P_k = \sum\limits_{j\in[n]} Q(k-1, j, \ell)$.

Below we show how to compute the polynomials $Q(\cdot,\cdot,\cdot)$ inductively, that is, we express $Q(t,\cdot,\cdot)$ through $Q(t-1,\cdot,\cdot)$. The base case is easy: $Q(0, j, w)$ is equal to $x_j$ if $|s_j| \le w$ and is equal to 0 otherwise. Assume now that $t > 0$. We show that

$$Q(t, j, w) = \sum_{i:\ s_i\subset s_j} x_i Q(t-1, j, w) + \sum_{i:\ s_i\not\subset s_j} x_j Q(t-1, i, w - (|s_j| - |\mathrm{overlap}(s_j, s_i)|))\,. \tag{2}$$

In this recurrence relation, we treat $Q(\cdot,\cdot,w)$ for $w < 0$ as zero.

To give a formal proof we show that the right-hand sides of equations (1) and (2) contain exactly the same summands. First, we divide the right side of (1) into two parts. There are two cases: either $s_{i_1} \subset s_j$ or $s_{i_1} \not\subset s_j$. If $s_{i_1} \subset s_j$ then $\Delta(s_j, s_{i_1}, s_{i_2}, \ldots, s_{i_t}) = \Delta(s_j, s_{i_2}, \ldots, s_{i_t})$. Then $\sum_{i:\ s_i\subset s_j} x_i Q(t-1, j, w)$ is equal to

$$\sum_{i:\ s_i\subset s_j} x_i \left( \sum_{(i_1,\ldots,i_{t-1})\in[n]^{t-1}} [|\,\Delta(s_j, s_{i_1}, \ldots, s_{i_{t-1}})| \le w] \cdot x_j x_{i_1} \cdots x_{i_{t-1}} \right) =$$
$$= \sum_{(i,i_1,\ldots,i_{t-1})\in[n]^t, s_i\subset s_j} [|\,\Delta(s_j, s_i, s_{i_1} \ldots, s_{i_{t-1}})| \le w] \cdot x_j x_i x_{i_1} \cdots x_{i_{t-1}}.$$

If $s_{i_1} \not\subset s_j$ then $\Delta(s_j, s_{i_1}, s_{i_2}, \ldots, s_{i_t}) = \mathrm{prefix}_{|s_j|-|\mathrm{overlap}(s_j,s_{i_1})|}(s_j)\,\Delta(s_{i_1}, s_{i_2}, \ldots, s_{i_t})$. Then $\sum_{i:\ s_i\not\subset s_j} x_j Q(t-1, i, w - (|s_j| - |\mathrm{overlap}(s_j, s_i)|))$ is equal to

$$\sum_{i:\ s_i\not\subset s_j} x_j \left( \sum_{(i_1,\ldots,i_{t-1})\in[n]^{t-1}} [|\,\Delta(s_i, s_{i_1}, \ldots, s_{i_{t-1}})| \le w - (|s_j| - |\mathrm{overlap}(s_j, s_i)|)] \cdot x_i x_{i_1} \cdots x_{i_{t-1}} \right) =$$
$$= \sum_{(i,i_1,\ldots,i_{t-1})\in[n]^t, s_i\not\subset s_j} [|\,\Delta(s_j, s_i, s_{i_1} \ldots, s_{i_{t-1}})| \le w] \cdot x_j x_i x_{i_1} \cdots x_{i_{t-1}}.$$

Thus,

$$\sum_{(i,i_1,\ldots,i_{t-1})\in[n]^t, s_i\subset s_j} [|\,\Delta(s_j, s_i, s_{i_1} \ldots, s_{i_{t-1}})| \le w] \cdot x_j x_i x_{i_1} \cdots x_{i_{t-1}} +$$
$$\sum_{(i,i_1,\ldots,i_{t-1})\in[n]^t, s_i\not\subset s_j} [|\,\Delta(s_j, s_i, s_{i_1} \ldots, s_{i_{t-1}})| \le w] \cdot x_j x_i x_{i_1} \cdots x_{i_{t-1}} =$$
$$= \sum_{(i,i_1,\ldots,i_{t-1})\in[n]^t} [|\,\Delta(s_j, s_i, s_{i_1}, \ldots, s_{i_{t-1}})| \le w] \cdot x_j x_i x_{i_1} \cdots x_{i_{t-1}}$$

To upper bound the size of the resulting arithmetic circuitm we rewrite 2 as follows:

$$Q(t,j,w) = (\sum_{i\colon s_i \subset s_j} x_i)Q(t-1,j,w) + x_j \left( \sum_{i\colon s_i \not\subset s_j} Q(t-1,i,w-(|s_j|-|\mathrm{overlap}(s_j,s_i)|)) \right)$$

This gives an arithmetic circuit of size $\mathcal{O}(kn^2\ell)$. To see this, note that there are $O(kn^2\ell)$ different $Q$'s and an arithmetic circuit representing $Q(t,j,w)$ can be constructed through the circuits representing $Q(t-1,\cdot,\cdot)$'s by (2) using just two multiplication gates (of in-degree 2) and three summation gates (of in-degree at most $n$).

The result for the Partial Weighted Superstring problem follows from Theorem 3. For this, we assign a variable $x_i$ the weight $w_{\max} - w(s_i)$ where $w_{\max} = \max_{1\leq i\leq n} w(s_i)$. This way, each monomial is assigned a non-negative weight. Searching for a superstring $s$ such that $|s| \leq l$, $s$ is a superstring of $k' \leq k$ strings $S' \subseteq S$ with maximal $w(S)$ corresponds to searching for a minimal weight multilinear monomial in $P_{k'}$. So, we just apply the algorithm from Theorem 3 to all $P_{k'}$, $k \leq k$, and return the best answer. □

**Corollary 1.** Partial Superstring *is* FPT *when parameterized by* $\ell$.

*Proof.* Consider an instance $(S,k,\ell)$ of Partial Superstring. Recall that $S$ can contain several copies of the same string. We construct a set of weighted strings $S'$ by replacing a string $s$ that occurs $r$ times in $S$ by a single copy of $s$ of weight $w(s) = r$. Let $W = k$. Observe that there is a string $s$ of length at most $\ell$ such that $s$ is a superstring of a collection of at least $k$ strings of $S$ if and only if there a string $s$ of length at most $\ell$ such that $s$ is a superstring of a set of strings of $S'$ of total weight at least $W$. A string of length at most $\ell$ has at most $\ell(\ell-1)/2$ distinct substrings. We consider the instances $(S',w,k',\ell,W)$ of Partial Weighted Superstring for $k' \in \{1,\ldots,\ell(\ell-1)/2\}$. For each of these instances, we solve the problem using Theorem 4. It remains to observe that there is a string $s$ of length at most $\ell$ such that $s$ is a superstring of a set of strings of $S'$ of total weight at least $W$ if and only if one of the instances $(S',w,k',\ell,W)$ is a yes-instance of Partial Weighted Superstring. □

We complement the above algorithmic results by showing that we hardly can expect that Partial Superstring has a polynomial kernel when parameterized by $k$ or $\ell$.

**Theorem 5.** Partial Superstring *does not admit a polynomial kernel when parameterized by* $k+m$ *or* $\ell+m$ *for strings of length at most* $m$ *over the alphabet* $\Sigma = \{0,1\}$ *unless* NP $\subseteq$ coNP /poly.

*Theorem 5.* We show that Long Trail OR-cross-composes into Partial Superstring. Recall that Long Trail was shown to be NP-complete in Lemma 3.

We assume that two instances $(G,\ell)$ and $(G',\ell')$ of Long Trail are equivalent if $|V(G)| = |V(G')|$ and $\ell = \ell'$. Consider equivalent instances $(G_i,\ell)$ of Long Trail for $i \in \{1,\ldots,t\}$. Let $V(G_i) = \{v_1^i,\ldots,v_n^i\}$ for $i \in \{1,\ldots,t\}$. Let $r = \max\{\lfloor \log n \rfloor, \lfloor \log t \rfloor\} + 2$. Denote by $x_i$ the string of length $r$ that encodes a positive integer $i$ in binary for $i \leq 2^r - 1$. Let $x^* = x_i$ for $i = 2^r - 1$, i.e., $x^* =$ '$1\ldots1$'. Notice that if $i \leq \max\{n,t\}$, then the first symbol of $x_i$ is '0'. For each arc $e = (v_p^i, v_q^i)$ of $G_i$, we construct a string $s_e = x_i x^* x_i x_p x_i x^* x_i x_q x_i x^* x_i$. Clearly, $|s_e| = 11r$. Notice also that if $e = (v_p^i, v_q^i)$ and $e' = (v_q^i, v_t^i)$, then the suffix $x_i x^* x_i x_q x_i x^* x_i$ of $s_e$ is a prefix of $s_{e'}$. In particular, it means that $\mathrm{overlap}(s_e, s_{e'}) \geq 7r$. We define

$$S = \{s_e \mid e \in E(G_i), 1 \leq i \leq t\}$$

and let $k = \ell$, $\ell' = 4r\ell + 7r$. We claim that there is $i \in \{1, \ldots, t\}$ such that $G_i$ has a trail of length $\ell$ if and only if there is a string $s$ of length at most $\ell'$ that is a superstring of $k$ strings of $S$.

Suppose that there is $i \in \{1, \ldots, t\}$ such that $G_i$ has a trail $e_1, \ldots, e_\ell$. Consider $s = \sigma(s_{e_1}, \ldots, s_{e_\ell})$. Because the length of each $s_{e_i}$ is $11r$ and $|\mathrm{overlap}(s_{e_{i-1}}, s_{e_i})| \geq 7r$, we obtain that $|s| \leq 11r\ell - 7r(\ell - 1) = \ell'$. Hence, $s$ is a string of length at most $\ell'$ that is a superstring of $k = \ell$ strings.

Assume now that there is a string $s$ of length at most $\ell'$ that is a superstring of $k$ strings of $S$. Because no string of $S$ is a substring of another one, we can assume that $s = \sigma(s_{e_1}, \ldots, s_{e_k})$ for some $e_1, \ldots, e_k \in E(G_1) \cup \ldots \cup E(G_t)$ by Lemma 1. We use the following properties of the overlap of two strings $s_e, s_{e'} \in S$. Recall that if $e = (v_p^i, v_q^i)$ of $G_i$, then $s_e = x_i x^* x_i x_p x_i x^* x_i x_q x_i x^* x_i$, $x^* =$ '$1 \ldots 1$' and the first symbol of $x_i$ is '0'. It implies that $|\mathrm{overlap}(s_e, s_{e'})| \leq 7r$ and $|\mathrm{overlap}(s_e, s_{e'})| = 7r$ if and only if $e, e' \in E(G_i)$ for some $i \in \{1, \ldots, t\}$ and $e = (v_p^i, v_q^i)$, $e' = (v_q^i, v_z^i)$ for some $p, q, z \in \{1, \ldots, n\}$. Since $|s| \leq \ell' = 4r\ell + 7r$ and $k = \ell$, $|\mathrm{overlap}(s_{e_{j-1}}, s_{e_j})| = 7r$ for $j \in \{2, \ldots, k\}$. Hence, $e_1, \ldots, e_\ell$ is a trail in some $G_i$.

It remains to observe that $k + m = \mathcal{O}(n + \log t)$ and $\ell' + m = \mathcal{O}((n + \log t)^2)$ to complete the proof. □

## 4 Shortest Superstring below guaranteed values

In this section we discuss Shortest Superstring parameterized by the difference between upper bounds for the length of a shortest superstring and the length of a solution superstring. For a collection of strings $S$, the length of the shortest superstring is trivially upper bounded by $\sum_{x \in S} |x|$. We show that Shortest Superstring admits a polynomial kernel when parameterized by the compression measure of a solution.

**Theorem 6.** Shortest Superstring *admits a kernel of size* $\mathcal{O}(r^4)$ *when parameterized by* $r = \sum_{x \in S} |x| - \ell$.

*Proof.* Let $(S, \ell)$ be an instance of Shortest Superstring, $r = \sum_{x \in S} |x| - \ell$. First, we apply the following reduction rules for the instance.

**Rule 1.** If there are distinct elements $x$ and $y$ of $S$ such that $x \subseteq y$, then delete $x$ and set $r = r - |x|$. If $r \leq 0$, then return a yes-answer and stop.

**Rule 2.** If there is $x \in S$ such that for any $y \in S \setminus \{x\}$, $|\mathrm{overlap}(x, y)| = |\mathrm{overlap}(y, x)| = 0$, then delete $x$ and set $\ell = \ell - |x|$. If $S = \emptyset$ and $\ell \geq 0$, then return a yes-answer and stop. If $\ell < 0$, then return a no-answer and stop.

**Rule 3.** If there are distinct elements $x$ and $y$ of $S$ such that $|\mathrm{overlap}(x, y)| \geq r$, then return a yes-answer and stop.

It is straightforward to verify that these rules are *safe*, i.e., by the application of a rule we either solve the problem or obtain an equivalent instance. We exhaustively apply Rules 1–3. To simplify notations, we assume that $S$ is the obtained set of strings and $\ell$ and $r$ are the obtained values of the parameters. Notice that all strings in $S$ are distinct and no string is a substring of another. Our next aim is to bound the lengths of considered strings.

**Rule 4.** If there is $x \in S$ with $|x| > 2r$, then set $\ell = \ell - |x| + 2r$ and $x = \mathrm{prefix}_r(x)\mathrm{suffix}_r(x)$. If $\ell < 0$, then return a no-answer and stop.

To see that the rule is safe, recall that $x$ is not a sub- or superstring of any other string of $S$, and $|\mathrm{overlap}(x,y)| < r$ and $|\mathrm{overlap}(y,x)| < r$ for any $y \in S$ distinct from $x$ after the applications of Rule 3. As before, we apply Rule 4 exhaustively.

Now we construct an auxiliary graph $G$ with the vertex set $S$ such that two distinct $x, y \in S$ are adjacent in $G$ if and only if $|\mathrm{overlap}(x,y)| > 0$ or $|\mathrm{overlap}(y,x)| > 0$. We greedily select a maximal matching $M$ in $G$ and apply the following rule.

**Rule 5.** If $|M| \geq r$, then return a yes-answer and stop.

To show that the rule is safe, it is sufficient to observe that if $M = \{x_1, y_1\}, \ldots, \{x_h, y_h\}$, $|\mathrm{overlap}(x_i, y_i)| > 0$ for $i \in \{1, \ldots, h\}$ and $h \geq r$, then the string $s$ obtained by the consecutive concatenations with overlaps of $x_1, y_1, \ldots, x_h, y_h$ and then all the other strings of $S$ in arbitrary order, then the compression measure of $s$ is at least $r$.

Assume from now that we do not stop here, i.e., $|M| \leq r-1$. Let $X \subseteq S$ be the set of end-vertices of the edges of $M$ and $Y = S \setminus X$. Let $X = \{x_1, \ldots, x_h\}$. Clearly, $h \leq 2(r-1)$. Observe that $X$ is a vertex cover of $G$ and $Y$ is an independent set of $G$.

For each ordered pair $(i,j)$ of distinct $i, j \in \{1, \ldots, h\}$, find an ordering $y_1, \ldots, y_t$ of the elements of $Y$ sorted by the decrease of $|\mathrm{overlap}(x_i, y_p)| + |\mathrm{overlap}(y_p, x_j)|$ for $p \in \{1, \ldots, t\}$. We construct the set $R_{(i,j)}$ that contains the first $\min\{2h, t\}$ elements of the sequence.

For each $i \in \{1, \ldots, h\}$, find an ordering $y_1, \ldots, y_t$ of the elements of $Y$ sorted by the decrease of $|\mathrm{overlap}(y_p, x_i)|$ for $p \in \{1, \ldots, t\}$. We construct the set $S_i$ that contains the first $\min\{2h, t\}$ elements of the sequence.

For each $i \in \{1, \ldots, h\}$, find an ordering $y_1, \ldots, y_t$ of the elements of $Y$ sorted by the decrease of $|\mathrm{overlap}(x_i, y_p)|$ for $p \in \{1, \ldots, t\}$. We construct the set $T_i$ that contains the first $\min\{2h, t\}$ elements of the sequence.

Let

$$S' = X \cup \Big( \bigcup_{(i,j),\ i,j \in \{1,\ldots,h\}, i \neq j} R_{(i,j)} \Big) \cup \Big( \bigcup_{i \in \{1,\ldots,h\}} S_i \Big) \cup \Big( \bigcup_{i \in \{1,\ldots,h\}} T_i \Big).$$

**Claim** $(*)$**.** *There is a superstring $s$ of $S$ with the compression measure at least $r$ if and only if there is a superstring $s'$ of of $S'$ with the compression measure at least $r$.*

*Proof of Claim* $(*)$. If $s'$ is a superstring of $S'$ with the compression measure at least $r$, then the string $s$ obtained from $s'$ by the concatenation of $s'$ and the strings of $S \setminus S'$ (in any order) is a superstring of $S$ with the same compression measure as $s'$.

Suppose that $s$ is a shortest superstring of $S$ and the compression measure at least $r$. By Lemma 1, $s = \sigma(s_1, \ldots, s_n)$, where $S = \{s_1, \ldots, s_n\}$. Let

$$Z = \{s_i \mid s_i \in Y, |\mathrm{overlap}(s_{i-1}, s_i)| > 0 \text{ or } |\mathrm{overlap}(s_i, s_{i+1})| > 0, 1 \leq i \leq n\};$$

we assume that $s_0, s_{n+1}$ are empty strings.

We show that $|Z| \leq 2h$. Suppose that $s_i \in Z$. If $|\mathrm{overlap}(s_{i-1}, s_i)| > 0$, then $s_{i-1} \in X$, because $s_i \in Y$ and any two strings of $Y$ have the empty overlap. By the same arguments, if $|\mathrm{overlap}(s_i, s_{i+1})| > 0$, then $s_{i+1} \in X$. Because $|X| = h$, we have that $|Z| \leq 2h$.

Suppose that the shortest superstring $s$ is chosen in such a way that $|Z \setminus S'|$ is minimum. We prove that $Z \subseteq S'$ in this case. To obtain a contradiction, assume that there is $s_i \in Z \setminus S'$. We consider three cases.

**Case 1.** $|\mathrm{overlap}(s_{i-1}, s_i)| > 0$ and $|\mathrm{overlap}(s_i, s_{i+1})| > 0$. Recall that $s_{i-1}, s_{i+1} \in X$ in this case. Since $s_i \notin S'$, $s_i \notin R_{(p,q)}$ for $x_p = s_{i-1}$ and $x_q = s_{i+1}$. In particular, it means that $|R_{(p,q)}| = 2h$. As $|Z| \leq 2h$ and $|R_{(p,q)}| = 2h$, there is $s_j \in R_{(p,q)}$ such that $s_j \notin Z$, i.e., $|\mathrm{overlap}(s_{j-1}, s_j)| = |\mathrm{overlap}(s_j, s_{j+1})| = 0$. By the definition of $R_{(p,q)}$, $|\mathrm{overlap}(s_{i-1}, s_j)| + |\mathrm{overlap}(s_j, s_{i+1})| \geq |\mathrm{overlap}(s_{i-1}, s_i)| + |\mathrm{overlap}(s_i, s_{i+1})|$. Consider $s^* = \sigma(s_1, \ldots, s_{i-1}, s_j, s_{i+1}, \ldots, s_{j-1}, s_i, s_{j+1}, \ldots, s_n)$ assuming that $i < j$ (the other case is similar). Because $|\mathrm{overlap}(s_{i-1}, s_j)| + |\mathrm{overlap}(s_j, s_{i+1})| \geq |\mathrm{overlap}(s_{i-1}, s_i)| + |\mathrm{overlap}(s_i, s_{i+1})|$, $|s^*| \leq |s|$. Moreover, since $s$ is a shortest superstring of $S$, $|s| \leq |s^*|$ and, therefore, $|\mathrm{overlap}(s_{j-1}, s_i)| = |\mathrm{overlap}(s_i, s_{j+1})| = 0$. But then for the set $Z^*$ constructed for $s^*$ in the same way as the set $Z$ for $s$, we obtain that $|Z^* \setminus S'| < |Z \setminus S'|$; a contradiction.

**Case 2.** $|\mathrm{overlap}(s_{i-1}, s_i)| = 0$ and $|\mathrm{overlap}(s_i, s_{i+1})| > 0$. Then $s_{i+1} \in X$. Since $s_i \notin S'$, $s_i \notin S_p$ for $x_p = s_{i+1}$ and $|S_p| = 2h$. As $|Z| \leq 2h$ and $|S_p| = 2h$, there is $s_j \in S_p$ such that $s_j \notin Z$, i.e., $|\mathrm{overlap}(s_{j-1}, s_j)| = |\mathrm{overlap}(s_j, s_{j+1})| = 0$. By the definition of $S_p$, $|\mathrm{overlap}(s_j, s_{i+1})| \geq |\mathrm{overlap}(s_i, s_{i+1})|$. As in Case 1, consider $s^*$ obtained by the exchange of $s_i$ and $s_j$ in the sequence of strings that is used for the concatenations with overlaps. In the same way, we obtain a contradiction with the choice of $Z$, because for the set $Z^*$ constructed for $s^*$ in the same way as the set $Z$ for $s$, we obtain that $|Z^* \setminus S'| < |Z \setminus S'|$.

**Case 3.** $|\mathrm{overlap}(s_{i-1}, s_i)| > 0$ and $|\mathrm{overlap}(s_i, s_{i+1})| = 0$. To obtain contradiction in this case, we use the same arguments as in Case 2 using symmetry. Notice that we should consider $T_p$ instead of $S_p$.

Now let $s' = \sigma(s_{i_1}, \ldots, s_{i_p})$, where $s_{i_1}, \ldots, s_{i_p}$ is the sequence of strings of $S'$ obtained from $s_1, \ldots, s_n$ by the deletion of the strings of $S \setminus S'$. Because we have that $Z \subseteq S'$, the overlap of each deleted string with its neighbors is empty and, therefore, $s'$ has the same compression measure as $s$. □

To finish the construction of the kernel, we define $\ell' = \ell - \sum_{x \in S \setminus S'} |x|$ and apply the following rule that is safe by Claim $(*)$.

**Rule 6.** If $\ell' < 0$, then return a no-answer and stop. Otherwise, return the instance $(S', \ell')$ and stop.

Since $|X| = h \leq 2(r-1)$, $|S'| \leq h + h^2 \cdot 2h + h \cdot 2h + h \cdot 2h = 2h^3 + 4h^2 + h = O(h^3) = O(r^3)$. Because each string of $S'$ has length at most $2r$, the kernel has size $O(r^4)$.

It is easy to see that Rules 1-3 can be applied in polynomial time. Then graph $G$ and $M$ can be constructed in polynomial time and, trivially, Rule 5 demands $\mathcal{O}(1)$ time. The sets $X$, $Y$, $R_{(i,j)}$, $S_i$ and $T_i$ can be constructed in polynomial time. Hence, $S'$ and $\ell'$ can be constructed in polynomial time. Because Rule 6 can be applied in time $\mathcal{O}(1)$, we conclude that the kernel is constructed in polynomial time. □

Now we consider another upper bound for the length of the shortest superstring. Let $S$ be a collection of strings. We construct an auxiliary weighted graph $G(S)$ with the vertex set $S$ by assigning the weight $w(\{x, y\}) = \max\{|\mathrm{overlap}(x, y)|, |\mathrm{overlap}(y, x)|\}$ for any two distinct $x, y \in S$. Let $\mu(S)$ be the size of a maximum weighted matching in $G(S)$. Clearly, $G(S)$ can be constructed in polynomial time and the computation of $\mu(S)$ is well known to be polynomial [7]. If $M = \{x_1, y_1\}, \ldots, \{x_h, y_h\}$ and $|\mathrm{overlap}(x_i, y_i)| = w(\{x_i, y_i\})$ for $i \in \{1, \ldots, h\}$, then the string $s$ obtained by the consecutive concatenations with overlaps of $x_1, y_1, \ldots, x_h, y_h$ and then (possibly)

the remaining string of $S$ has the compression measure at least $\mu(S)$. Hence, $\sum_{x\in S}|x|-\mu(S)$ is an upper bound for the length of the shortest superstring of $S$. We show that it is NP-hard to find a superstring that is shorter than this bound.

**Theorem 7.** SHORTEST SUPERSTRING *is* NP*-complete for* $\ell=\sum_{x\in S}|x|-\mu(S)-1$ *even if restricted to the alphabet* $\Sigma=\{0,1\}$.

*Proof.* We reduce LONG TRAIL that was shown to be NP-complete in Lemma 3 for $\ell=|V(G)|-1$. Let $(G,\ell)$ be an instance of the problem, $n=|V(G)|=\ell+1$. We assume that $n\geq 2^6=64$. Let $V(G)=\{v_1,\ldots,v_n\}$ and $E(G)=\{e_1,\ldots,e_m\}$. Let also $p=\lceil(n-1)/3\rceil$ and $q=n-1-2p$. Denote by $z=$'01...1' and $z^*=$'1...1' the strings of length $p$ such that the first symbol of $z$ is '0' and all the other symbols are '1'-s and $z^*$ is a strings of '1'-s. For a positive integer $i\leq 2^{q-1}-1$, denote by $x_i$ the string of length $q-1$ that encodes $i$ in binary and by $y_i$ the string of length $q$ that encodes $2i$. Notice that $q\geq n/3-4$ and $\log n^2\leq q-3$, because $n\geq 2^6$. Hence, the first symbols of $x_i$ and $y_i$ are '0' if $i\leq n^2$. Observe also that the last symbol of each $y_i$ is '0'. For each $h\in\{1,\ldots,m\}$, we consider the arc $e_h=(v_i,v_j)$ of $G$ and construct two strings:

- $s_h=zy_hz^*zx_iz^*zx_jz^*$,
- $s'_h=zx_iz^*zx_jz^*zy_hz^*$.

We define $S=\{s_h,s'_h\mid 1\leq h\leq m\}$.

We need the following properties of the strings of $S$.

i) For $h\in\{1,\ldots,m\}$, $|\mathrm{overlap}(s_h,s'_h)|=2(n-2)$ and $|\mathrm{overlap}(s'_h,s_h)|=n-1$.

ii) For distinct $h,h'\in\{1,\ldots,m\}$, $|\mathrm{overlap}(s_h,s'_{h'})|=n-2$ if the head of $e_h$ coincides with the tail of $e_{h'}$ and $|\mathrm{overlap}(s_h,s'_{h'})|=0$ otherwise.

iii) For distinct $h,h'\in\{1,\ldots,m\}$, $|\mathrm{overlap}(s'_h,s_{h'})|=|\mathrm{overlap}(s_h,s_{h'})|=|\mathrm{overlap}(s'_h,s'_{h'})|=0$.

These properties immediately follow from the definition of $s_h,s'_h$ and the facts that $|z|=|z^*|\geq|y_h|=|x_i|+1=|x_j|+1$, the strings $z,y_h,x_i,x_j$ start with '0', the last symbol of $y_h$ is '0', and $z=$'01...1', $z^*=$'1...1'. It is sufficient to notice that if the overlap of two strings is not empty, then the $p$-th prefix and suffix of the overlap is always $z$ and $z^*$ respectively.

Now we consider the weighted graph $G(S)$ and observe that $M=\{\{s_h,s'_h\}\mid 1\leq h\leq m\}$ is a maximum weight matching in $G(S)$ and $\mu(S)=2(n-2)m$ by (i)–(iii).

We claim that $G$ has a trail of length at least $\ell=n-1$ if and only if $S$ has a superstring of length at most $\ell'=\sum_{x\in S}|x|-\mu(S)-1$.

Suppose that the sequence of arcs $e_{i_1},\ldots,e_{i_\ell}$ composes a trail in $G$. Let $\{e_{j_1},\ldots,e_{j_{m-\ell}}\}=E(G)\setminus\{e_{i_1},\ldots,e_{i_\ell}\}$. Consider

$$s=\sigma(s'_{i_1},s_{i_1},\ldots,s'_{i_\ell},s_{i_\ell},s_{j_1},s'_{j_1},\ldots,s_{j_{m-\ell}},s'_{j_{m-\ell}}).$$

Since $|\mathrm{overlap}(s'_{i_h},s_{i_h})|=n-1$ for $h\in\{1,\ldots,\ell\}$ by (i), $|\mathrm{overlap}(s_{i_{h-1}},s'_{i_h})|=n-2$ for $h\in\{2,\ldots,\ell\}$ by (ii), $\mathrm{overlap}(s_{i_\ell},s_{j_1})=0$ by (iii) and $|\mathrm{overlap}(s_{j_h},s'_{j_h})|=2(n-2)$ by (i), the compression measure of $s$ is $t=(n-1)\ell+(n-2)(\ell-1)+2(n-2)(m-\ell)$. Recall that $\mu(S)=2(n-2)m$ and $\ell=n-1$. Hence,

$$t-\mu(S)=(n-1)\ell+(n-2)(\ell-1)+2(n-2)(m-\ell)-2(n-2)m=\ell-n+2=1.$$

Hence, $s$ is a superstring of $S$ of length at most $\ell'$.

Assume that $s$ is a shortest superstring of $S$ and $|s| \leq \ell'$. By Lemma 1, we can assume that $s$ is obtained from a sequence $\sigma$ of the strings of $S$ by the concatenations with overlaps.

We show that for every $h \in \{1, \ldots, m\}$, either $s_h, s'_h$ or $s'_h, s_h$ are consecutive in $\sigma$. To obtain a contradiction, assume first that for some $h \in \{1, \ldots, m\}$, $s_h$ occurs in $\sigma$ before $s'_h$ but these strings are not consecutive. Let $a$ be the predecessor of $s_h$, $b$ be a predecessor of $s'_h$ and $c$ be a successor of $s'_h$ in $\sigma$; if $s_h$ is the first element of $\sigma$ or $s'_h$ is the last element, we assume that $a$ or $c$ is the empty string respectively. Then $|\mathrm{overlap}(a, s_h)| = |\mathrm{overlap}(s'_h, c)| = 0$ by (iii) and $|\mathrm{overlap}(b, s'_h)| \leq n-2$ by (ii) and (iii). Consider the sequence $\sigma'$ obtained from $\sigma$ by the placement of $s'_h$ between $a$ and $s_h$. Because $|\mathrm{overlap}(s'_h, s_h)| = n-1$ by (1), the string $s'$ obtained from $\sigma'$ by the concatenations with overlaps has length at most $|s|-1$; a contradiction. Suppose now that for some $h \in \{1, \ldots, m\}$, $s'_h$ occurs in $\sigma$ before $s_h$ but these strings are not consecutive. Let $a$ be the successor of $s'_h$, $b$ be a predecessor of $s_h$ and $c$ be a successor of $s_h$ in $\sigma$; if $s_h$ is the last element of $\sigma$, we assume that $c$ is the empty string. We have that $|\mathrm{overlap}(s'_h, a)| = |\mathrm{overlap}(b, s_h)| = 0$ by (iii) and $|\mathrm{overlap}(s_h, c)| \leq n-2$ by (ii) and (iii). Consider the sequence $\sigma'$ obtained from $\sigma$ by the placement of $s_h$ between $s'_h$ and $a$. Because $|\mathrm{overlap}(s'_h, s_h)| = n-1$ by (i), the string $s'$ obtained from $\sigma'$ by the concatenations with overlaps has length at most $|s|-1$; a contradiction.

We decompose $\sigma$ into inclusion maximal subsequences $\sigma_1, \ldots, \sigma_r$ such that the overlap between any two consecutive strings in each subsequence is not empty. Because either $s_h, s'_h$ or $s'_h, s_h$ are consecutive in $\sigma$ for $h \in \{1, \ldots, m\}$ and $|\mathrm{overlap}(s_h, s'_h)| = 2(n-2)$ and $|\mathrm{overlap}(s'_h, s_h)| = n-1$ by (i), each pair $s_h, s'_h$ is in the same subsequence. In particular, it means that the number of elements in each subsequence is even. Let $n_i$ be the size of $\sigma_i$ and let $w_i$ be the string obtained by the concatenation with overlaps from $\sigma_i$ for $i \in \{1, \ldots, r\}$. Because $n_1 + \ldots + n_r = 2m$, $|M| = m$ and the compression measure of $s$ is at least $\mu(S) + 1$, there is $i \in \{1, \ldots, r\}$ such that the compression measure $\alpha$ of $w_i$ is at least $n_i/2 \cdot \mu(S)/m + 1 = n_i(n-2) + 1$.

Suppose that $s_h, s'_h$ are in $\sigma_i$ for some $h \in \{1, \ldots, m\}$. Then they are consecutive. If $s_h$ has a predecessor $a$ in $\sigma$, then $|\mathrm{overlap}(a, s_h)| = 0$, and if $s'_h$ has a successor $b$ in $\sigma$, then $|\mathrm{overlap}(s'_h, b)| = 0$ by (iii). Hence, $\sigma_i = s_h, s'_h$ and $n_i = 2$ in this case, but then by (i), $\alpha = 2(n-2) < n_i(n-1)/2 + 1$; a contradiction. It follows that $w_i = \sigma(s'_{i_1}, s_{i_1}, \ldots, s'_{i_k}, s_{i_k})$, where distinct $i_1, \ldots, i_k \in \{1, \ldots, m\}$ and $k = n_i/2$. Since for $j \in \{2, \ldots, k\}$, the overlap between $s_{i_{j-1}}$ and $s'_{i_j}$ is not empty, $|\mathrm{overlap}(s_{i_{j-1}}, s'_{i_j})| = n-2$ and the head of the arc $e_{i_{j-1}}$ is the tail of $e_{i_j}$. Hence, $e_{i_1}, \ldots, e_{i_k}$ is a trail in $G$. By (i) and (ii), we have that $\alpha = k(n-1) + (k-1)(n-2) \geq 2k(n-2) + 1$. Therefore, $k \geq n-1$, i.e., $G$ has a trail of length at least $\ell = n-1$. □

# 5 Conclusions

In the paper we provide a number of results about the parameterized complexity of the Shortest Superstring problem under different parameterizations. Recall that the well-known Shortest Supersequence problem asks for a set of strings $S$ over an alphabet $\Sigma$, about a string $s$ of minimum length such that every string $x \in S$ is a *subsequence* of $s$, that is, there are indices $1 \leq i_1 < \ldots < i_{|x|} \leq |s|$ such that $s[i_j] = x[j]$ for $j \in \{1, \ldots, |x|\}$. We leave it as an open question whether it is possible to obtain similar results about the parameterized complexity of some variants of Shortest Supersequence.

## References


[1] Concorde TSP Solver. http://www.math.uwaterloo.ca/tsp/concorde.html

[2] Bellman, R.: Dynamic programming treatment of the travelling salesman problem. Journal of the ACM (JACM) 9(1), 61–63 (1962)

[3] Bodlaender, H.L., Jansen, B.M.P., Kratsch, S.: Kernelization lower bounds by cross-composition. SIAM J. Discrete Math. 28(1), 277–305 (2014)

[4] Bulteau, L., Hüffner, F., Komusiewicz, C., Niedermeier, R.: Multivariate algorithmics for NP-hard string problems. Bulletin of the EATCS 114 (2014)

[5] Cygan, M., Fomin, F.V., Kowalik, L., Lokshtanov, D., Marx, D., Pilipczuk, M., Pilipczuk, M., Saurabh, S.: Parameterized Algorithms. Springer (2015)

[6] Downey, R.G., Fellows, M.R.: Fundamentals of Parameterized Complexity. Texts in Computer Science, Springer (2013)

[7] Edmonds, J.: Maximum matching and a polyhedron with $0,1$-vertices. J. Res. Nat. Bur. Standards Sect. B 69B, 125–130 (1965)

[8] Evans, P.A., Wareham, T.: Efficient restricted-case algorithms for problems in computational biology. In: Algorithms in Computational Molecular Biology: Techniques, Approaches and Applications, pp. 27–49. Wiley Series in Bioinformatics, Wiley (2011)

[9] Fomin, F.V., Lokshtanov, D., Panolan, F., Saurabh, S.: Representative sets of product families. In: Algorithms-ESA 2014, pp. 443–454. Springer (2014)

[10] Gallant, J., Maier, D., Storer, J.A.: On finding minimal length superstrings. J. Comput. Syst. Sci. 20(1), 50–58 (1980)

[11] Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W. H. Freeman (1979)

[12] Golovnev, A., Kulikov, A.S., Mihajlin, I.: Solving 3-superstring in $3^{n/3}$ time. In: Mathematical Foundations of Computer Science 2013, pp. 480–491. Springer (2013)

[13] Golovnev, A., Kulikov, A.S., Mihajlin, I.: Solving SCS for bounded length strings in fewer than $2^n$ steps. Information Processing Letters 114(8), 421–425 (2014)

[14] Held, M., Karp, R.M.: A dynamic programming approach to sequencing problems. Journal of the Society for Industrial & Applied Mathematics 10(1), 196–210 (1962)

[15] Karp, R.M.: Dynamic programming meets the principle of inclusion and exclusion. Operations Research Letters 1(2), 49–51 (1982)

[16] Kohn, S., Gottlieb, A., Kohn, M.: A generating function approach to the traveling salesman problem. In: Proceedings of the 1977 annual conference. pp. 294–300. ACM (1977)

[17] Mucha, M.: Lyndon words and short superstrings. In: Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms. pp. 958–972. SIAM (2013)

[18] Williams, R.: Finding paths of length k in $O^*(2^k)$ time. Information Processing Letters 109(6), 315–318 (2009)