Accepted for publication at *Theoretical Computer Science*, Elsevier.

Revisiting the Parameterized Complexity of Maximum-Duo Preservation String Mapping*

Christian Komusiewicz† Mateus de Oliveira Oliveira‡ Meirav Zehavi§

Abstract

In the Maximum-Duo Preservation String Mapping (Max-Duo PSM) problem, the input consists of two related strings A and B of length n and a nonnegative integer k. The objective is to determine whether there exists a mapping m from the set of positions of A to the set of positions of B that maps only to positions with the same character and preserves at least k *duos*, which are pairs of adjacent positions. We develop a randomized algorithm that solves Max-Duo PSM in $4^k \cdot n^{O(1)}$ time, and a deterministic algorithm that solves this problem in $6.855^k \cdot n^{O(1)}$ time. The previous best known (deterministic) algorithm for this problem has $(8e)^{2k+o(k)} \cdot n^{O(1)}$ running time [Beretta et al., Theor. Comput. Sci. 2016]. We also show that Max-Duo PSM admits a problem kernel of size $O(k^3)$, improving upon the previous best known problem kernel of size $O(k^6)$.

Keywords: Maximum-Duo Preservation String Mapping, Parameterized Algorithms, Kernelization

1 Introduction

Computing distances between strings is a fundamental task in computer science. For many distance measures, the distance between two strings A and B is defined as the minimum number of local operations that are needed to transform A into B, for example the deletion or insertion of a character. For these measures, the distance between two strings A and B can be usually computed in polynomial time [13, 23]. In some applications, however, it is necessary to consider nonlocal operations that transform one string into the other. In comparative genomics, for example, genomes are modeled as strings with one character corresponding to a complete gene and one is interested in determining the evolutionary distance between two genomes. During biological evolution, genomes may be altered by large-scale mutations such as the reversal or the transposition of larger parts of the genome [19].

One approach to approximate the distance between two strings A and B with respect to many of these operations is to compute a smallest *common string partition* [11, 27]. Informally, a size-ℓ common string partition of two strings A and B is a partition of A and B, each into ℓ nonoverlapping substrings, such that the resulting two multisets of substrings of A and B are the same. The problem to compute a smallest common string partition, known as Minimum Common String Partition, is NP-hard [11, 22].

An alternative way of defining such a partition is to ask for a partition of A into ℓ nonoverlapping substrings such that permuting the order of these substrings and concatenating them subsequently gives the string B. This second view implies a mapping m that (bijectively) maps each position i of A to a position $m(i)$ of B such that $A[i] = B[m(i)]$. The size of the common

*A preliminary version of this paper appeared in the proceedings of CPM 2017.

†Fachbereich Mathematik und Informatik, Philipps-Universität Marburg, Germany. komusiewicz@informatik.uni-marburg.de

‡University of Bergen, Bergen, Norway. mateus.oliveira@uib.no

§Ben-Gurion University of the Negev, Beersheba, Israel. meiravze@bgu.ac.il

string partition is then exactly the number of pairs of consecutive positions i and $i+1$ (called *duos*) such that $m(i)+1 \neq m(i+1)$ plus one since i is the end of one part and $i+1$ is the start of the next part. Therefore, computing a mapping m that maps only positions with the same characters to each other and maximizes the number k of consecutive positions for which $m(i)+1 = m(i+1)$ directly yields a minimum common string partition of A and B. The problem of computing such a mapping is known as Maximum-Duo Preservation String Mapping (Max-Duo PSM). Since Max-Duo PSM is simply a dual of the Minimum Common String Partition problem, it is NP-hard as well. Motivated by this hardness, we study Max-Duo PSM from the viewpoint of parameterized algorithmics. More precisely, our aim is to obtain efficient algorithms when the parameter is k, the number of preserved duos. Before describing previous and our results, we give a formal problem definition.

Formal Problem Definition. Let A and B be two strings over a finite set of symbols Σ. Throughout this work, we assume that $|A| = |B| = n$ and that A and B are *related*, that is, B is a permutation of A. A *mapping of A into B* is a (bijective) function $m : [n] \to [n]$ where for each $i \in [n]$,[1] $A[i] = B[m(i)]$. A *duo* in A is a pair of consecutive positions $(i, i+1)$ of A. We say that a mapping m *preserves a duo* $(i, i+1)$ if $m(i)+1 = m(i+1)$. Accordingly, the Max-Duo PSM problem is defined as follows.

> Maximum-Duo Preservation String Mapping (Max-Duo PSM)
> **Input:** Two related strings, A and B, and a nonnegative integer k.
> **Question:** Does there exist a (bijective) mapping m of A into B such that the number of preserved duos is at least k?

Previous Work. Initially, Max-Duo PSM has been proposed as an alternative possibility of achieving approximation algorithms for Minimum Common String Partition (MCSP) [10], because the best known polynomial-time approximation algorithm has an approximation factor of $O(\log n \log^* n)$ [12]. Consequently, most work on Max-Duo PSM focuses on approximation algorithms with the first constant-factor approximation algorithm achieving an approximation factor of 4 [6]. This was subsequently improved to a factor of 3.5 [5] and then to a factor of 3.25 [7]. Recently further progress concerning the approximation factor has been reported [18, 28].

Beretta et al. [2, 1] initiated the study of Max-Duo PSM from the viewpoint of parameterized algorithmics. They studied both the fixed-parameter tractability and the kernelization complexity of Max-Duo PSM, showing that this problem can be solved in $(8e)^{2k+o(k)} \cdot n^{O(1)}$ time, and that it admits a kernel of size $O(k^6)$. Thus, Beretta et al. [2, 1] were the first to show that Max-Duo PSM is FPT and that it admits a polynomial kernel. The fixed-parameter algorithm of Beretta et al. [2, 1] is based on a combination of color coding and dynamic programming.

In comparison with Max-Duo PSM, MCSP has been investigated more thoroughly from the viewpoint of parameterized algorithms. Damaschke [15] presented the first fixed-parameter algorithms for MCSP, for combined parameters such as "partition size ℓ plus repetition number of the input strings".[2] Subsequently, MCSP was shown to be fixed-parameter tractable with the single parameter partition size ℓ [9]. Jiang et al. [24] considered the combined parameter "partition size ℓ plus maximum occurrence d of any character" and showed that MCSP can be solved in $(d!)^k \cdot n^{O(1)}$ time. Subsequently, this running time was improved to $O(d^{2k} \cdot kn)$ [8].

Our Contribution. We make two main contributions. First, we develop two algorithms for the Max-Duo PSM problem that are substantially faster than the (deterministic) algorithm by Beretta et al. [2, 1], which runs in $(8e)^{2k+o(k)} \cdot n^{O(1)}$ time. Specifically, we develop a randomized

[1] We use $[n]$ as shorthand for $\{1, 2, \ldots, n\}$.

[2] The repetition number of a nonempty string x is defined as the largest i such that $x = uv^i w$ where v is nonempty.

algorithm that solves Max-Duo PSM in $4^k \cdot n^{O(1)}$ time, as well as a deterministic algorithm that solves this problem in $6.855^k \cdot n^{O(1)}$ time. Here, in the context of our randomized algorithm, we mean that if we determine that the input is a yes-instance, then this answer is necessarily correct, and if we determine that the input is a no-instance, then this answer is correct with probability at least 9/10.[3] For the purpose of developing our algorithms, we present a reduction from Max-Duo PSM to a problem of finding paths in an edge-colored graph, which might be of independent interest. This reduction lies at the heart of our algorithms, since by employing advanced tools from the field of parameterized algorithmics, namely, the methods of narrow sieves [4, 3] and representative sets [20], it is possible to quickly solve the resulting graph problem.

Second, we prove that Max-Duo PSM admits a kernel of size $O(k^3)$, improving upon the kernel of size $O(k^6)$ by Beretta et al. [2].

Preliminaries. We use $[i, j]$ to denote the set $\{i, i+1, \ldots, j\}$ of natural numbers between i and j. Moreover, given a string A, we denote the substring starting at position i and ending at position j by $A[i, j]$. For a (directed) graph G, let $V(G)$ denote the vertex set of G and $E(G)$ the edge set of G.

The field of parameterized algorithmics studies *parameterized problems*, where each problem instance is associated with a *parameter* k, usually a nonnegative integer. Given a parameterized problem, the first question is whether the problem is *fixed-parameter tractable (FPT)*, that is, whether it can be solved in $f(k) \cdot |X|^{O(1)}$ time, where f is an arbitrary function that depends *only* on k and $|X|$ is the size of the input instance. In other words, the notion of FPT signifies that the combinatorial explosion can be confined to the parameter k. A second question is whether the problem also admits a *polynomial kernelization*. Here, a problem Π is said to admit a polynomial kernelization if there exists a polynomial-time algorithm that, given an instance (X, k) of Π, outputs an equivalent instance $(\widehat{X}, \widehat{k})$ of Π, called a *kernel*, where $|\widehat{X}| = \widehat{k}^{O(1)}$ and $\widehat{k} \leq k$; kernelization is a mathematical concept that aims to analyze preprocessing procedures in a formal, rigorous manner. For further details, refer to [17, 14, 21].

2 Reduction to a Path Finding Problem

In this section, we present a reduction from Max-Duo PSM to the following graph problem.

> Substantially Blue Path
> **Input:** A directed acyclic graph (DAG) G, an edge-coloring $c : E(G) \to \{R, B\}$, a vertex-labeling $\ell : V(G) \to \mathbb{N}$, and nonnegative integers k and r.
> **Question:** Does G contain a directed path P such that
>
> - $|V(P)| \leq r$,
> - for all $u, v \in V(P)$, $\ell(u) \neq \ell(v)$, and
> - $|\{e \in E(P) : c(e) = B\}| \geq k$.

Construction. Let (A, B, k) be an instance of Max-Duo PSM. We construct an instance (G, c, ℓ, k, r) of Substantially Blue Path as follows (here, the parameter k is the same). First, we initialize G to be an empty graph. Now, for every pair of substrings $A[i, j]$ of A and $B[p, q]$ of B such that $j - i \leq k$ and $A[i, j] = B[p, q]$, we insert a directed path $P_{i,j,p,q}$ on $j - i + 1$ new vertices into G whose edges are colored blue and such that the label of the dth vertex on this path is $(p + d - 1)$. The purpose of this path is to represent the possibility to preserve all duos in $A[i, j]$ by mapping this substring to $B[p, q]$. The labels of the vertices are meant to ensure that every position in B is mapped only once. Now, a complete mapping of A to B can be seen

[3] Clearly, the probability of success can be improved by running the algorithm multiple times and determining that the input is a yes-instance if and only if at least one of the calls determined so.

as a combination of mappings of substrings that are represented by the paths. Thus, we next turn to connect the paths we have just constructed by adding new edges.

For every two paths $P_{i,j,p,q}$ and $P_{i',j',p',q'}$ such that $j < i'$, we add a red edge from the last vertex of the path $P_{i,j,q,p}$ to the first vertex of the path $P_{i',j',q',p'}$. Informally, the manner in which we direct these edges is meant to ensure that every position in A is mapped only once. Clearly, the resulting graph G is a DAG. Finally, we set $r = 2k$.

Correctness. We first note that the construction can be done in $O(|V(G)| + |E(G)|)$ time. Now, observe that the number of paths $P_{i,j,p,q}$ that G contains is bounded by $n^2(k+1)$ (as the index q equals $p + (j - i)$), and that each path $P_{i,j,p,q}$ consists of at most $(k+1)$ vertices. Hence, it holds that $|V(G)| \leq n^2(k+1)^2$ which directly implies $|E(G)| < n^4(k+1)^2$. Thus, we have the following observation.

Observation 1. *The instance (G, c, ℓ, k, r) can be constructed in $O(n^4k^2)$ time.*

We prove the correctness by proving two lemmata that together imply that the instances (A, B, k) and (G, c, ℓ, k, r) are equivalent.

Lemma 1. *If (A, B, k) is a yes-instance of* Max-Duo PSM*, then (G, c, ℓ, k, r) is a yes-instance of* Substantially Blue Path*.*

Proof. Let m be a mapping from A into B preserving at least k duos. Consider the set $\{A_1, \ldots, A_r\}$ of substrings of A containing exactly the first k preserved duos, where we assume that A_i precedes A_{i+1} in A. Consider any A_z and let $[i_z, j_z]$ be the set of positions of A_z in A. Since the mapping preserves the duos in A_z, there is a substring $B[q_z, p_z]$ such that $m(i_z + s) = q_z + s$, $0 \leq s \leq j - i$. This implies that $A[i_z, j_z] = B[q_z, p_z]$. Thus, G contains the path $P_z := P_{i_z,j_z,q_z,p_z}$.

By the above, for each A_z, G contains a path P_z containing $|A_z| - 1$ blue edges. Moreover, for A_i and A_j, the vertices in P_i and P_j have different labels since the mapping m is injective. Finally, there is a red edge from the last vertex of P_i to the first vertex of P_{i+1} since the last position of A_i is strictly smaller than the first position of A_{i+1}. Thus, the concatenation of P_1, P_2 until P_r gives a path in G. The number of blue edges in this path is exactly k, and the number of vertices in this path is at most $2k$, since every P_i contains at least one blue edge. □

Lemma 2. *If (G, c, ℓ, k) is a yes-instance of* Substantially Blue Path*, then (A, B, k) is a yes-instance of* Max-Duo PSM*.*

Proof. Let P be a solution of the Substantially Blue Path instance. That is, P is a path in G on at most $2k$ vertices, all with different labels, containing at least k blue edges. Let $\{P_1, \ldots, P_r\}$ be the set of disjoint paths obtained from P by removing all red edges where we assume that there is a red edge from P_i to P_{i+1} for all $i \in [r-1]$. Consider some P_z. By the construction of G, $P_z = P_{i,j,q,p}$ for some $i < j$ and $q < p$. Hence, there is a substring $A[i, j]$ of A and a substring $B[q, p]$ of B such that $A[i, j] = B[q, p]$. Call these two substrings the substrings of A and B, respectively, that *correspond* to P_z. Observe that for P_i and P_j, $i < j$, the substrings corresponding to P_i and P_j are disjoint: For the substrings in A this is due to the fact that the indices of the corresponding substring for P_i are lower than those of the substring of A corresponding to P_j. For the substrings in B this is due to the fact that the vertices in P_i and P_j have different labels. Thus, there is a mapping from A into B that maps the corresponding strings for each path P_i and maps all other positions arbitrarily. The number of duos preserved by this mapping is at least k. □

Altogether, we arrive at the following.

Lemma 3. *Given an instance (A, B, k) of* Max-Duo PSM*, an equivalent instance (G, c, ℓ, k, r) of* Substantially Blue Path *where $r = 2k$ can be constructed in $O(n^4k^2)$ time.*

3 A Randomized Algorithm based on Narrow Sieves

In this section, we adapt the method of *narrow sieves* that was applied to solve the k-Path problem [4] to solve Substantially Blue Path. More precisely, our objective is to provide a constructive proof for the following result.

Lemma 4. *There exists a randomized algorithm that solves* Substantially Blue Path *in* $2^r \cdot r^{O(1)} \cdot |E(G)|$ *time and polynomial space.*

In light of Lemma 3, once we have Lemma 4 at hand, we immediately obtain the following theorem.

Theorem 1. *There exists a randomized algorithm that solves* Max-Duo PSM *in* $4^k \cdot k^{O(1)} \cdot n^4$ *time and polynomial space.*

In the following, we focus on the proof of Lemma 4. To this end, let (G, c, ℓ, k, r) be an instance of Substantially Blue Path. Clearly, we can assume that $|V(G)| \leq |E(G)|$. To be able to rely on dynamic programming later, we need to define a notion of a partial solution:

Definition 1. *Let P be a directed path in G. Given a vertex $v \in V(G)$, $s \in [r]$ and $b \in [r] \cup \{0\}$, we say that P is a (v, s, b)-*path *if the last vertex of P is v, $|V(P)| = s$ and $|\{e \in E(P) : c(e) = B\}| = b$. If for all $u, w \in V(P)$, it holds that $\ell(u) \neq \ell(v)$, then we say that P is a* good path*.*

To employ the method of narrow sieves, we need to associate labels with entities whose uniqueness should be preserved. For this purpose, we have the following definition:

Definition 2. *Let P be a (v, s, b)-path. Given $f : V(P) \to [r]$, we say that (P, f) is a (v, s, b)-*pair*. If P is good, then we say that (P, f) is a* good pair*, and if f is an injective function, then we say that (P, f) is an* injective pair*. Given $L \subseteq [r]$ such that the image of f is a subset of L, we say that (P, f) is an L-*labeled pair*.*

Now, we define two central sets of labeled partial solutions. The first one, $\mathcal{P}$, is the set of all pairs (P, f) that are injective (v, s, b)-pairs for some $v \in V(G)$ and $s, b \in [r]$ such that $b \geq k$. The second one, $\mathcal{Q}$, is the set of all good pairs (P, f) in $\mathcal{P}$. Note that for every pair $(P, f) \in \mathcal{Q}$, it holds that P is a solution for Substantially Blue Path, and for every solution P for Substantially Blue Path, by letting f be a function that assigns i to the ith vertex on P, we obtain a pair $(P, f) \in \mathcal{Q}$. Thus, we have the following observation.

Observation 2. *The instance (G, c, ℓ, k, r) is a yes-instance if and only if $\mathcal{Q} \neq \emptyset$.*

With these definitions at hand, we may describe the rough idea of the approach. We represent all labeled partial solutions of $\mathcal{P}$ by a polynomial in such a way that each labeled partial solution corresponds to one monomial. We will ensure that the partial solutions of $\mathcal{P} \setminus \mathcal{Q}$ cancel each other out which will imply that the polynomial is not identically 0 if and only if $\mathcal{Q} \neq \emptyset$. To this end, we now describe how we represent labeled partial solutions by monomials. For every label $i \in \text{image}(\ell)$ and integer $j \in [r]$, we introduce the variable $x_{i,j}$, and for every edge $e \in E(G)$, we introduce the variable y_e. This gives the following representation:

Definition 3. *Let (P, f) be a (v, s, b)-pair. Then, the monomial associated with (P, f) is defined as follows.*

$$\mathsf{mon}(P, f) = \prod_{v \in V(P)} x_{\ell(v), f(v)} \cdot \prod_{e \in E(P)} y_e.$$

Accordingly, we define the following polynomial (which will be evaluated over a field of characteristic 2).

Definition 4. $\mathsf{POL} = \sum_{(P,f)\in\mathcal{P}} \mathsf{mon}(P,f)$.

To analyze this polynomial, we first observe that given a monomial associated with a pair $(P,f) \in \mathcal{Q}$, we can uniquely recover the pair (P,f). To see this, consider some monomial M that is associated with a pair $(P,f) \in \mathcal{Q}$. Then, the variables y_e of M specify exactly which edges are used by P, and therefore the path P is recovered. Now, since the pair (P,f) belongs to $\mathcal{Q}$, we have that P is a good path. Hence, the variables $x_{i,j}$ of M specify exactly how f labels the vertices of P. In other words, we have the following observation.

Observation 3. *For all $(P,f) \in \mathcal{Q}$, there does not exist $(P',f') \in \mathcal{P} \setminus \{(P,f)\}$ such that $\mathsf{mon}(P,f) = \mathsf{mon}(P',f')$.*

The following lemma will be used to show that the partial solutions of $\mathcal{P} \setminus \mathcal{Q}$ cancel each other out.

Lemma 5. *There exists a function $g : \mathcal{P} \setminus \mathcal{Q} \to \mathcal{P} \setminus \mathcal{Q}$ such that for all $(P,f) \in \mathcal{P} \setminus \mathcal{Q}$, it holds that $\mathsf{mon}(P,f) = \mathsf{mon}(g(P,f))$, $g(P,f) \neq (P,f)$, and $g(g(P,f)) = (P,f)$.*

Proof. Let $<$ be some order on $\{\{u,v\} : u,v \in V(P)\}$. Given $(P,f) \in \mathcal{P} \setminus \mathcal{Q}$, define $\mathsf{rep}(P,f) = \{\{u,v\} : u,v \in V(P), u \neq v, \ell(u) = \ell(v)\}$. Since P is not a good path, it holds that $\mathsf{rep}(P,f) \neq \emptyset$. Hence, it is well defined to let $\{u,v\}$ be the smallest set in $\mathsf{rep}(P,f)$ according to $<$. We let h be defined as f except that $h(u) = f(v)$ and $h(v) = f(u)$. Now, we set $g(P,f) = (P,h)$. Clearly, $g(P,f) \in \mathcal{P}$. Note that $\mathsf{rep}(P,f) = \mathsf{rep}(P,h)$, and hence $g(P,f) \notin \mathcal{Q}$ and $g(g(P,f)) = (P,f)$. Since $(P,f) \in \mathcal{P}$, it holds that f is an injective function; therefore $f(v) \neq f(u)$, which implies that $g(P,f) \neq (P,f)$. Finally, since $\ell(u) = \ell(v)$, it holds that $\mathsf{mon}(P,f) = \mathsf{mon}(g(P,f))$. □

Let $\mathbb{F}$ be a field of characteristic 2 (whose precise size will be determined later). From now on, we suppose that POL is evaluated over $\mathbb{F}$. Notice that

$$\mathsf{POL} = \sum_{(P,f)\in\mathcal{Q}} \mathsf{mon}(P,f) + \sum_{(P,f)\in\mathcal{P}\setminus\mathcal{Q}} \mathsf{mon}(P,f).$$

By Lemma 5, we have that $\mathsf{POL} = \sum_{(P,f)\in\mathcal{Q}} \mathsf{mon}(P,f)$. Then, by Observation 3, we have that POL is not identically 0 if and only if $\mathcal{Q}$ is not empty. Hence, by Observation 2, we have the following lemma.

Lemma 6. *The instance (G,c,ℓ,k,r) is a yes-instance if and only if POL is not identically 0.*

Due to Lemma 6, our task is to determine whether POL is identically 0. For this purpose, we need the following notation. Given $v \in V(G)$, $s \in [r]$, $b \in [r] \cup \{0\}$ and $L \subseteq [r]$, let $\mathcal{P}_{v,s,b,L}$ denote the set of all L-labeled (v,s,b)-pairs (P,f), and $\mathsf{POL}_{v,s,b,L} = \sum_{(P,f)\in\mathcal{P}_{v,s,b,L}} \mathsf{mon}(P,f)$. Moreover, denote

$$\mathcal{P}_L = \bigcup_{v \in V(G), s,b \in [r], b \geq k} \mathcal{P}_{v,s,b,L},$$

and $\mathsf{POL}_L = \sum_{(P,f)\in\mathcal{P}_L} \mathsf{mon}(P,f)$. By the principle of inclusion-exclusion, we have that $\mathsf{POL} = \sum_{L\subseteq[r]} (-1)^{r-|L|}\mathsf{POL}_L$. Then, since $\mathbb{F}$ is a field of characteristic 2 (refer to [4] for further details) we obtain the following.

Observation 4. $\mathsf{POL} = \sum_{L\subseteq[r]} \mathsf{POL}_L$.

Hence, to determine whether POL is identically 0, it is sufficient to determine whether $\sum_{L\subseteq[r]} \mathsf{POL}_L$ is identically 0. To proceed, we need to recall the following well-known lemma.

Lemma 7 ([25, 29, 16])**.** *Let $p(x_1, x_2, \ldots, x_n)$ be a nonzero polynomial of total degree at most d over a finite field $\mathbb{K}$. Then, for $a_1, a_2, \ldots, a_n \in \mathbb{K}$ selected independently and uniformly at random, $Pr(p(a_1, a_2, \ldots, a_n) \neq 0) \geq 1 - d/|\mathbb{K}|$.*

Notice that POL is a polynomial of total degree at most $2r$. Therefore, by setting $|\mathbb{F}| = 2^{\lceil \log(20r) \rceil}$, from Lemma 6, Observation 4, and Lemma 7, we have that

Lemma 8. *For a random assignment to all variables $x_{i,j}$ and y_e, if (G, c, ℓ, k, r) is a no-instance, then $\sum_{L\subseteq[r]} \mathsf{POL}_L$ evaluates to 0, and otherwise it does* not *evaluate to 0 with probability at least 9/10.*

According to Lemma 8, to conclude that Lemma 4 is correct, it is sufficient to prove the following result.

Lemma 9. *Given $L \subseteq [r]$ and an assignment to all variables $x_{i,j}$ and y_e, the polynomial POL_L can be evaluated in $r^{O(1)} \cdot |E(G)|$ time.*

Proof sketch. The evaluation can be performed by a simple procedure based on dynamic programming. For the sake of completeness, we present the base cases and recursive formula below. For simplicity, we abuse notation by using the symbols $x_{i,j}$ and y_e to refer to the values assigned to the variables $x_{i,j}$ and y_e, respectively.

The procedure uses a table M, which has an entry $M[v, s, b]$ for all $v \in V(G)$, $s \in [r]$ and $b \in [r] \cup \{0\}$. The purpose of this entry is to store the value of $\mathsf{POL}_{v,s,b,L}$. Then, the value of POL_L is given by $\sum_{\substack{v \in V(G),\\ s, b \in [r], b \geq k}} M[v, s, b]$.

The basis consists of the following cases:

- If $b \geq s$, then $M[v, s, b] = 0$.
- Else if $s = 1$, then $M[v, s, b] = \sum_{i\in L} x_{\ell(v),i}$.

Now, consider an entry $M[v, s, b]$ not computed in the basis. We assume that a reference to an undefined entry returns 0. Then,

$$M[v, s, b] = \left(\sum_{\substack{(u,v) \in E(G),\\ c(u,v) = R}} \left((\sum_{i\in L} x_{\ell(v),i}) \cdot y_{(u,v)} \cdot M[u, s-1, b] \right) \right) + \left(\sum_{\substack{(u,v) \in E(G),\\ c(u,v) = B}} \left((\sum_{i\in L} x_{\ell(v),i}) \cdot y_{(u,v)} \cdot M[u, s-1, b-1] \right) \right).$$

Intuitively, the recursive formula is partitioned into two: the first sum corresponds to the case where the last arc is red, and the second sum corresponds to the case where the last arc is

blue. In both cases, we consider all possibilities for the identity of the vertex u appearing before v—these are all the ingoing neighbors u of v—along with multiplication by the corresponding variable $y_{(u,v)}$ and either $M[u, s-1, b]$ or $M[u, s-1, b-1]$, all possibilities for the label i of v (which is represented by the internal sum) along with multiplication by the corresponding variable $x_{\ell(v),i}$. The only difference between the two cases is in the third argument (the number of blue arcs), which clearly only decreases in the second case. □

Finally, we would like to remark that if one is interested in finding a mapping that is a solution rather than just determining whether such a mapping exists, this goal can be achieved by standard means of self-reduction. Briefly, if k is not positive, then we are done. Else, if the algorithm determines that there exists a solution, then we may "guess" (i.e., perform exhaustive search) a longest substring A' of A that is mapped by some solution while preserving all duos in A' as well as the substring B' of B to which it is mapped. If our guess is correct, then the symbol preceding A' in A is not equal to the symbol preceding B' in B and the symbol after A' in A is also not equal to the symbol after B' in B (if such symbols exist). Then, we may replace A' and B' in A and B, respectively, by some new symbol, decrease k by $|A'| - 1$, and call the algorithm recursively. Notice that the length of A' is at least 2, and hence the size of the input has decreased.

4 Deterministic Algorithm: Representative Sets

In this section, we adapt the approach in which the method of *representative sets* is applied to solve the k-Path problem [20]. More precisely, our objective is to provide a constructive proof for the following result.

Lemma 10. *There exists a deterministic algorithm that solves* Substantially Blue Path *in* $O((\frac{1+\sqrt{5}}{2})^{r+o(r)} \cdot |E(G)| \log |E(G)|)$ *time.*

In light of Lemma 3, once we have Lemma 10 at hand, we directly obtain the following theorem.

Theorem 2. *There exists a deterministic algorithm that solves* Max-Duo PSM *in* $O((\frac{1+\sqrt{5}}{2})^{2k+o(k)} \cdot n^4 \log n) = O(6.855^k \cdot n^4 \log n)$ *time.*

Next, we focus on the proof of Lemma 10. To this end, let (G, c, ℓ, k, r) be an instance of Substantially Blue Path. Without loss of generality, we can assume that the image of ℓ is a subset of $[|V(G)|]$ and that $|V(G)| \leq |E(G)|$. Here, a *p-set* is a set of size p. To describe our algorithm, we need to present the definition of a representative family.

Definition 5 ([20])**.** *Given a universe U and a family $\mathcal{S}$ of p-subsets of U, we say that a subfamily $\widehat{\mathcal{S}} \subseteq \mathcal{S}$* t-represents *$\mathcal{S}$ if for every pair of sets $X \in \mathcal{S}$, and $Y \subseteq U \setminus X$ of size $t - p$, there exists a set $\widehat{X} \in \widehat{\mathcal{S}}$ such that $\widehat{X} \cap Y = \emptyset$.*

Intuitively, this definition arises in dynamic programming procedures naturally as follows. Suppose that we seek a solution of size t, and we have a family $\mathcal{S}$ of "partial" solutions of size $p \leq t$. Every set X in $\mathcal{S}$ can be potentially "completed" to solution, where all possibilities to complete it are given by all sets of size $t - p$ disjoint from it (where, possibly, none will actually give a solution). Then, the notion of a representative family assures that if we can complete some set X in $\mathcal{S}$ to a solution, then there is also a set in $\widehat{\mathcal{S}}$ that can be completed to a solution, and therefore it is sufficient to work only with the sets in $\widehat{\mathcal{S}}$.

The papers [20] and [26] present an algorithm, to which we refer as RepAlg, that given a universe U and a family $\mathcal{S}$ of p-subsets of U, computes a subfamily $\widehat{\mathcal{S}} \subseteq \mathcal{S}$ of size $S(|U|, t, p)$

that t-represents $\mathcal{S}$ in $|\mathcal{S}| \cdot T(|U|, t, p)$ time, such that the following condition is satisfied:

$$\sum_{p=1}^{t} |U| \cdot S(|U|, t, p-1) \cdot T(|U|, t, p) = \left(\frac{1+\sqrt{5}}{2}\right)^{t+o(t)} \cdot |U| \log |U|.$$

We proceed by presenting a procedure that is based on a combination of dynamic programming and calls to RepAlg. For this purpose, we use a table M that has an entry $M[v, s, b]$ for all $v \in V(G)$, $s \in [r]$ and $b \in [r] \cup \{0\}$. Let $\mathcal{P}_{v,s,b}$ denote the set of all *good* (v, s, b)-paths (see Definition 1). Given a (v, s, b)-path, define $\ell(P) = \{\ell(v) : v \in V(P)\}$. Moreover, define $\mathcal{S}_{v,s,b} = \{\ell(P) : P \in \mathcal{P}_{v,s,b}\}$. The purpose of the entry $M[v, s, b]$ is to store a subfamily of $\mathcal{S}_{v,s,b}$ that r-represents it. Next, we show how to compute the entries of M. Here, the calls to RepAlg are done with respect to $[|E(G)|]$ as the universe and $t = r$.

The basis consists of the following cases:

- If $s = 1$ but $b \neq 0$, then $M[v, s, b] = \emptyset$.
- Else if $s = 1$, then $M[v, s, b] = \{\{\ell(v)\}\}$.

Now, consider an entry $M[v, s, b]$ not computed in the basis. We assume that a reference to an undefined entry returns an empty set. Then, we first compute the two following families.

- $\mathcal{A}_{v,s,b} = \{X \cup \{\ell(v)\} : (u, v) \in E(G), c(u, v) = R, X \in M[u, s-1, b], \ell(v) \notin X\}$.
- $\mathcal{B}_{v,s,b} = \{X \cup \{\ell(v)\} : (u, v) \in E(G), c(u, v) = B, X \in M[u, s-1, b-1], \ell(v) \notin X\}$.

Accordingly, we compute $M[v, s, b]$ as follows.

$$M[v, s, b] = \mathsf{RepAlg}(\mathcal{A}_{v,s,b} \cup \mathcal{B}_{v,s,b}).$$

First, note that running time for the entire computation is bounded by

$$\begin{aligned} & O\Big(\sum_{v \in V(G)} \sum_{s=1}^{r} \sum_{b=0}^{r} \sum_{(u,v) \in E(G)} S(|E(G)|, r, s) \cdot T(|E(G)|, r, s)\Big) \\ = \; & O\Big(\sum_{s=1}^{r} r|E(G)| \cdot S(|E(G)|, r, s) \cdot T(|E(G)|, r, s)\Big). \end{aligned}$$

Thus, we have the following observation.

Observation 5. *The table M is computed in $O((\frac{1+\sqrt{5}}{2})^{r+o(r)} \cdot |E(G)| \log |E(G)|)$ time.*

Next, we prove that the computation of M is correct.

Lemma 11. *The computation of M ensures that for all $v \in V(G)$, $s \in [r]$ and $b \in [r] \cup \{0\}$, $M[v, s, b]$ r-represents $\mathcal{S}_{v,s,b}$.*

Proof. We prove the statement by induction on s. In the basis, where $s = 1$, it is clear that $M[v, s, b]$ is simply assigned $\mathcal{S}_{v,s,b}$, and therefore it also 1-represents $\mathcal{S}_{v,s,b}$. Now, fix some $s \geq 2$, and suppose that the statement is correct for $s - 1$. To prove that the statement is correct for s, choose some $v \in V(G)$, $b \in [r] \cup \{0\}$, $X \in \mathcal{S}_{v,s,b}$ and $Y \subseteq [|E(G)|] \setminus X$ such that $|Y| = r - s$. We need to show that there exists $\widehat{X} \in M[v, s, b]$ such that $\widehat{X} \cap Y = \emptyset$. Note that $M[v, s, b]$ r-represents $\mathcal{A}_{v,s,b} \cup \mathcal{B}_{v,s,b}$, and therefore is $\mathcal{A}_{v,s,b} \cup \mathcal{B}_{v,s,b}$ contains a set that is disjoint from Y, so does $M[v, s, b]$. Thus, it is sufficient that we show that there exists $\widehat{X} \in \mathcal{A}_{v,s,b} \cup \mathcal{B}_{v,s,b}$ such that $\widehat{X} \cap Y = \emptyset$.

Since $X \in \mathcal{S}_{v,s,b}$, there exists a good (v, s, b)-path P such that $\ell(P) = X$. Let u be the vertex on P that precedes v, and let Q be the path obtained by removing v from P.

Note that $\ell(Q) = X \setminus \{\ell(v)\}$. Thus, if $c(u,v) = R$, then Q is a good $(u, s-1, b)$-path and therefore $X \setminus \{\ell(v)\} \in \mathcal{S}_{u,s-1,b}$, and otherwise Q is a good $(u, s-1, b-1)$-path and therefore $X \setminus \{\ell(v)\} \in \mathcal{S}_{u,s-1,b-1}$. First, let us assume that $X \setminus \{\ell(v)\} \in \mathcal{S}_{u,s-1,b}$. By the inductive hypothesis, $M[u, s-1, b]$ r-represents $\mathcal{S}_{u,s-1,b}$, and therefore $M[u, s-1, b]$ contains a set Z such that $Z \cap (Y \cup \{\ell(v)\}) = \emptyset$. Thus, $Z \cup \{\ell(v)\} \in \mathcal{A}_{v,s,b}$, and we conclude that the statement is correct. Now, let us assume that $X \setminus \{\ell(v)\} \in \mathcal{S}_{u,s-1,b-1}$. By the inductive hypothesis, $M[u, s-1, b-1]$ r-represents $\mathcal{S}_{u,s-1,b-1}$, and therefore $M[u, s-1, b-1]$ contains a set Z such that $Z \cap (Y \cup \{\ell(v)\}) = \emptyset$. Thus, $Z \cup \{\ell(v)\} \in \mathcal{B}_{v,s,b}$, and again we conclude that the statement is correct. □

With these lemmas at hand, we are ready to prove Lemma 10.

Proof. By Observation 5 and Lemma 11, we first compute M, ensuring that the condition in Lemma 11 is satisfied, in $O((\frac{1+\sqrt{5}}{2})^{r+o(r)} \cdot |E(G)| \log |E(G)|)$ time. Then, we conclude that the input instance is a yes-instance if and only if there exist $v \in V(G)$, $s \in [r]$ and $b \in [r]$ such that $b \geq k$ and $M[v, s, b] \neq \emptyset$. On the one hand, since for all $v \in V(G)$, $s \in [r]$ and $b \in [r]$, $M[v, s, b] \subseteq \mathcal{S}_{v,s,b}$, it is clear that if we accept, the input instance is indeed a yes-instance. On the other hand, if the input instance is a yes-instance, then there exist $v \in V(G)$, $s \in [r]$ and $b \in [r]$ such that $b \geq k$ and $\mathcal{S}_{v,s,b} \neq \emptyset$. Then, since $M[v, s, b]$ 0-represents $\mathcal{S}_{v,s,b}$, it holds that $M[v, s, b] \neq \emptyset$, and therefore we accept. □

5 A Cubic Problem Kernel

In this section we will show that Max-Duo PSM admits a kernel of size $O(k^3)$. Let (A, B, k) be an instance of Max-Duo PSM, and let $S \in \{A, B\}$. If $S = A$, then we let $\overline{S} = B$. Analogously, if $S = B$, then we let $\overline{S} = A$.

Let m be a map of S into $\overline{S}$, and let D be a set of duos. We denote by $m(D) = \{(m(i), m(i+1)) \mid (i, i+1) \in D\}$ the image of D under m. We say that m *preserves* D if m preserves each duo in D. Let C_A and C_B be sets of duos. We say that the pair (C_A, C_B) is *complete* for (A, B, k) if whenever there is a map m of A into B that preserves k duos, then there is a subset $D \subseteq C_A$ with $|D| = k$ and a map m' such that m' preserves D and $m'(D) \subseteq C_B$. The size of (C_A, C_B) is defined as $|C_A| + |C_B|$. Let $f : \mathbb{N} \to \mathbb{N}$ be a function. A complete pair (C_A, C_B) of size $f(k)$ for (A, B, k) can be used to construct a kernel (A', B', k) of size $O(f(k))$ for (A, B, k).

Theorem 3 ([2, Section 4.2])**.** *Let (C_A, C_B) be a complete pair of size $f(k)$ for (A, B, k). Then one can construct in $O(f(k))$ time related strings A' and B', each of size $O(f(k))$ such that (A, B, k) is a yes-instance of* Max-Duo PSM *if and only if (A', B', k) is a yes-instance of* Max-Duo PSM*.*

Using Theorem 3, it is sufficient to show that one can obtain in polynomial time a complete pair (C_A, C_B) for (A, B, k) of size $O(k^3)$. To construct this pair, we proceed roughly as follows. First, we use the observation that the instance is a trivial yes-instance if some k consecutive duos can be matched. We call such a sequence of consecutive duos a block. The above observation implies that to add all the duos of a matched block to the complete pair, it is sufficient to identify one duo of each preserved block and then add all duos which are in distance at most k. The main part of the kernelization is thus identify one duo of each block. We first observe that if a certain type of duos, called rare duos, occurs too often, then we have a yes-instance. We then exploit that each block either contains a rare duo or is matched to a block containing a rare duo. The most difficult part of the kernelization is to identify at least one duo in each block that is matched to a rare duo. We identify these blocks by considering, for each rare duo, a sufficient number of duos of large blocks that match this duo. In other words, the duos that are not added to the kernel are those that are not rare, not close to rare duos, and not part of a

long block that matches a block with a rare duo. We now give the formal description of the kernelization algorithm.

A *block* of size s is a set $X = \{(i, i+1), (i+1, i+2), ..., (i+s-1, i+s)\}$ consisting of s consecutive duos. We say that $(i, i+1)$ is the *root* of X. If S is a string of length at least $i+s$, then we let $str(S, X) = S[i, i+s]$ be the substring of S corresponding to the positions that occur in X. The following observation is immediate.

Observation 6. *Let (A, B, k) be an instance of* Max-Duo PSM *and let m be a map of A into B that preserves a block X of size k. Then (A, B, k) is a yes-instance of* Max-Duo PSM*. Moreover, the instance (A', B', k) where $A' = str(A, X)$ and $B' = str(B, m(X))$ is also yes-instance of* Max-Duo PSM*.*

The above observation implies a trivial problem kernel of size $O(k)$ whenever some block of size k can be preserved. In the remainder of this section we thus assume that no map m of A into B preserves a block of size k. Our algorithm is based on the notion of *rare* duo, which we define next. For each two symbols $a, b \in \Sigma$, and each string $S \in \{A, B\}$, we let

$$n(S, a, b) := |\{i : 1 \leq i \leq |S| - 1,\ S[i, i+1] = ab\}|$$

be the number of occurrences of the length-two string ab as a substring of S. We say that a length-two string ab is *rare for* S if ab occurs as a sub-string of both S and $\overline{S}$ and $n(S, a, b) \leq n(\overline{S}, a, b)$. Observe that if ab occurs as many times in S as it occurs in $\overline{S}$, then ab is rare for both S and $\overline{S}$. We say that a duo $(i, i+1)$ is rare for S if $S[i, i+1]$ is rare for S. We let $rare(S)$ be the set of duos that are rare for S.

Lemma 12. *If either $|rare(A)| \geq 4k$ or $|rare(B)| \geq 4k$, then (A, B, k) is a yes-instance.*

Proof. The duo graph associated with A and B is the bipartite graph $G(A, B) = (V_A \dot{\cup} V_B, E)$ defined as follows.

$$V_A = \{(i, i+1) \mid 1 \leq i \leq n-1\} \qquad V_B = \{(j, j+1) \mid 1 \leq j \leq n-1\}$$

$$E = \{[(i, i+1), (j, j+1)] \mid A[i, i+1] = B[j, j+1]\}$$

Intuitively, each of the sets V_A and V_B contains all pairs of consecutive positions from $[n]$. A duo $(i, i+1)$ in V_A is adjacent to a duo $(j, j+1)$ in V_B if and only if the length-two string $A[i, i+1]$ and $B[j, j+1]$ can be matched to each other.

If $e = [(i, i+1), (j, j+1)]$ is an edge of $G(A, B)$, then we say that $(i, i+1)$ is the left endpoint of e and $(j, j+1)$ is the right endpoint of e. If M is a matching in $G(A, B)$, then we let M_A be the set of duos in V_A that are left endpoints of edges in M, and M_B be the set of duos in V_B that are right endpoints of edges in M.

Assume that either $|rare(A)| \geq 4k$ or $|rare(B)| \geq 4k$. Then G contains a matching M of size at least $4k$. Moreover, given a matching M of size at least $4k$ for the graph $G(A, B)$, one can construct a sub-matching M' of M of size at least k such that M' directly gives a map preserving at least k duos [6]. Therefore, the instance is a yes-instance in this case. □

In the remainder of this section we thus assume that there are less than $4k$ duos that are rare for A, and less than $4k$ duos that are rare for B. This implies that we may add all rare duos to the sets C_A and C_B without surpassing the desired size bound of $O(k^3)$.

Let S be a string in $\{A, B\}$. We say that a duo $(j, j+1)$ is a *match for a duo* $(i, i+1)$ *in* $\overline{S}$ if there exists a map m of S into $\overline{S}$ that preserves $(i, i+1)$, and $(m(i), m(i+1)) = (j, j+1)$. If X and Y are blocks, then we say that Y is a *match* for X in $\overline{S}$ if there exists a map m of S into $\overline{S}$ such that m preserves X, and $m(X) = Y$.

Observation 7. *Let $S \in \{A, B\}$ and let $(j, j+1)$ be a match for $(i, i+1)$ in $\overline{S}$. Then if $(i, i+1)$ is not rare for S, $(j, j+1)$ is rare for $\overline{S}$.*

Algorithm 1

```
1: procedure ROOTS(S, i, i+1)
2:     R = ∅
3:     k' ← size of the maximal block which is rooted at (i, i+1), rare for S, and has a
4:          match in S̄. Note that k' ≤ k − 1.
5:     for ℓ = k' to 1 do
6:         X ← unique block of size ℓ rooted at (i, i+1)
7:         for j = 1 to n − 1 do
8:             if |R| < 2k − 1 and (j, j+1) is a root for a match of X in S̄ and
9:                 |j' − j| > k for each j' such that (j', j'+1) ∈ R then
10:                R ← R ∪ {(j, j+1)}
11:    output R
```

Proof. Since $(j, j+1)$ is a match for $(i, i+1)$ in $\overline{S}$, there is some length-two string ab such that $S[i, i+1] = \overline{S}[j, j+1] = ab$. Since $(i, i+1)$ is not rare for S, the string ab occurs strictly more often in S than it occurs in $\overline{S}$. In other words, $n(S, a, b) > n(\overline{S}, a, b)$. This implies that $(j, j+1)$ is rare for $\overline{S}$. □

This observation is useful because it tells us that for each match in a map, one of the two duos is rare, so by adding all the rare duos to C_A and C_B, we essentially pick up one half of each match. We now consider two types of matched blocks that may occur in the solution. First, there may be pairs of matched blocks X and Y that both contain nonrare duos. We can add all duos of these blocks by considering a sufficiently large neighborhood of all rare duos. To this end, for each $i \in \{1, ..., n-1\}$, let

$$\mathcal{B}_k(i) = \{(i', i'+1) \mid i' \in \{1, ..., n-1\},\ i - k \leq i' \leq i + k\}$$

denote the *ball of radius* k around the duo $(i, i+1)$

The following lemma essentially implies that by adding the ball of radius k around each rare duo, we add all pairs of matched blocks that both contain at least one nonrare duo.

Lemma 13. *Let $S \in \{A, B\}$, X be a block of size at most $k-1$ containing a duo $(i, i+1)$ that is not rare for S, and let m be a map of S into $\overline{S}$ such that X is preserved by m. Then $(m(i), m(i+1))$ is rare for $\overline{S}$ and $m(X) \subseteq \mathcal{B}_k(m(i))$.*

Proof. Since $(i, i+1)$ is preserved by m, $(m(i), m(i+1))$ is a match for $(i, i+1)$ in $\overline{S}$. Since $(i, i+1)$ is not rare for S, by Observation 7, $(m(i), m(i+1))$ is rare for $\overline{S}$. Since m preserves X and since $|X| \leq k-1$, $m(X)$ is a block of size at most $k-1$. Therefore, all duos in $m(X)$ must be in the ball of radius k around $(m(i), m(i+1))$, that is, $m(X) \subseteq \mathcal{B}_k(m(i))$. □

We now turn to the second type of matched pairs of blocks, those where one block X of S has only rare duos for S; we call such a block X *rare*. Since X is rare, it is rooted at some rare duo $(i, i+1)$. To obtain the complete set, we need to add duos in $\overline{S}$. This is done by the procedure ROOTS which receives as input a string S and a duo in S and returns a set of duos $\textsc{Roots}(S, i)$.

Intuitively, for each block X that is rare for S and rooted at $(i, i+1)$, the set $\textsc{Roots}(S, i)$ contains a selection of roots of matches for X in the string $\overline{S}$. This selection is made according to two criteria. First, roots of matches for larger blocks are added first. Second, the roots in $\textsc{Roots}(S, i)$ are sufficiently far apart from each other. In particular, no root of $\textsc{Roots}(S, i)$ is contained in the ball of radius k of any other root of $\textsc{Roots}(S, i)$. Now consider the set

$$F(S, i) = \bigcup_{(j, j+1) \in \textsc{Roots}(S, i)} \mathcal{B}_{2k}(j).$$

Intuitively, $F(S,i)$ consists of all duos that are sufficiently close to duos in $\textsc{Roots}(S,i)$. The next lemma states that if some map m of S into $\overline{S}$ preserves some block X that is rooted at $(i,i+1)$ and rare for S, then this map can be transformed into a map m' that preserves X, that sends X to a subset of $F(S,i)$, and that is equal to m on every duo outside X.

Lemma 14. *Let m be a map of S into $\overline{S}$, D be a set of duos such that $|D| = k$, and $X \subseteq D$ be a block that is rooted at $(i,i+1)$, that is rare for S, and that is preserved by m. Then there is a map m' of S into $\overline{S}$ satisfying the following properties.*

1. *X is preserved by m',*
2. *$m'(X) \subseteq F(S,i)$, and*
3. *$(m'(i'), m'(i'+1)) = (m(i'), m(i'+1))$ for each $(i', i'+1) \in D \setminus X$.*

Proof. Assume that $m(X)$ is not contained in $F(S,i)$. We show how to transform m into a map m' that satisfies the properties of the lemma. Let $(j, j+1)$ be the root of $m(X)$ in $\overline{S}$. First, observe that, by the construction of $F(S,i)$, this implies that for every duo $(j', j'+1) \in \textsc{Roots}(S,i)$ we have $(j, j+1) \notin \mathcal{B}_k(j)$. Now this implies $|\textsc{Roots}(S,i)| = 2k-1$, because otherwise the procedure would have added at least one further duo since $(j, j+1)$ fulfills the conditions of Lines 8 and 9 of Algorithm 1. Moreover, since the roots are added in decreasing order of the lengths of the matched blocks, every duo in $\textsc{Roots}(S,i)$ is the root of a block of length $|X|$ in $\overline{S}$ that matches X.

Now, note that each duo $(q, q+1)$ is contained in at most two balls of radius k rooted at duos in $\textsc{Roots}(S,i)$. In other words, there are at most two duos $(j_1, j_1+1) \in \textsc{Roots}(S,i)$ and $(j_2, j_2+1) \in \textsc{Roots}(S,i)$ such that $(q, q+1) \in \mathcal{B}_k(j_1)$ and $(q, q+1) \in \mathcal{B}_k(j_2)$. Therefore, since $|D \setminus X| \leq k-1$, the set $D \setminus X$ intersects at most $2k-2$ balls of radius k rooted at duos in $\textsc{Roots}(S,i)$. This implies that there is at least one duo $(j', j'+1) \in \textsc{Roots}(S,i)$ such that $\mathcal{B}_k(j') \cap (D \setminus X) = \emptyset$. Moreover, $(j', j'+1)$ is the root of a match for X in $\overline{S}$ as argued above. Therefore, we may set m' as the map of S into $\overline{S}$ that preserves X, that sends the root of X to $(j', j'+1)$, and that is equal to m on every duo $(i', i'+1) \in D \setminus X$. □

Now, for each $S \in \{A, B\}$, consider the following set C_S of duos.

$$C_S = \left[\bigcup_{(i,i+1) \in rare(S)} \mathcal{B}_k(i) \right] \cup \left[\bigcup_{(i,i+1) \in rare(\overline{S})} F(\overline{S}, i) \right]. \tag{1}$$

In other words, for each duo $(i, i+1)$ that is rare for S, C_S contains all duos in the ball of radius k around $(i, i+1)$. Moreover, for each duo $(i, i+1)$ that is rare for $\overline{S}$, C_S contains all duos in the set $F(\overline{S}, i)$. The following lemma states that if there is a map of S into $\overline{S}$ preserving k duos, then we may assume that these duos are mapped to $C_{\overline{S}}$.

Lemma 15. *Let D be a set of duos such that $|D| = k$. Let m be a map of S into $\overline{S}$ that preserves all duos in D. Then there is a map m' of S into $\overline{S}$ that preserves D, and such that $m'(D) \subseteq C_{\overline{S}}$.*

Proof. Assume towards a contradiction that this is not the case and let m be the map such that $|m(D) \cap C_{\overline{S}}|$ is maximal among all maps of S into $\overline{S}$ that preserve D. Since $m(D) \not\subseteq C_{\overline{S}}$, there is some maximal block X in S such that X is preserved by m and $m(X) \not\subseteq C_{\overline{S}}$. Note that X is rare since since, otherwise, $m(X)$ would contain a rare duo $(j, j+1)$ and then $m(X)$ would be contained in $\mathcal{B}_k(j) \subseteq C_{\overline{S}}$. Since X is rare, Lemma 14 applies to X and D. Thus, there is a mapping m' such that X is preserved by m', $m'(X) \subseteq F(S,i)$ and m' agrees with m on all duos in $D \setminus X$. By construction, $F(S,i) \subseteq C_{\overline{S}}$, and thus $|m'(D) \cap C_{\overline{S}}| > |m(D) \cap C_{\overline{S}}|$. Since m' preserves D as well, this contradicts the choice of m. □

Let C_A and C_B be sets of duos constructed according to Equation 1. We can show that (C_A, C_B) is complete for (A, B, k) by applying Lemma 15 twice. More precisely, once with respect to maps of A into B, and once with respect to maps of B into A.

Lemma 16. *The pair (C_A, C_B) is complete for (A, B, k).*

Proof. Let D_1 be a set of duos of size k. Let m_1 be a map of A into B which preserves all duos in D_1. Then by Lemma 15 there is a map m_2 of A into B which also preserves all duos in D_1, but with the property that $m_2(D_1) \subseteq C_B$. Now let $D_2 = m_2(D_1)$, and $m_3 = m_2^{-1}$ be the inverse of m_2. In other words, m_3 is a map of B into A such that for each $i \in [n]$, $m_2(i) = j$ if and only if $m_3(j) = i$. Then m_3 preserves all duos in D_2. By Lemma 15 there is a map m_4 of B into A that also preserves all duos in D_2 and, additionally, fulfills the property that $m_4(D_2) \subseteq C_A$.

Let $D_3 = m_4(D_2)$, and let $m_5 = m_4^{-1}$ be the inverse of m_4. Then m_5 is a map of A into B that preserves $D_3 \subseteq C_A$ and such that $m_5(D_3) = D_2 \subseteq C_B$. Since $|D_3| = |D_2| = k$, the pair (C_A, C_B) is complete for (A, B, k). □

Now, we can upper-bound the size of C_S and the time needed to construct C_S, thus arriving at our main theorem.

Theorem 4. *Given an instance $I = (A, B, k)$ of* Max-Duo PSM*, one can construct in $O(|\Sigma|^2 \cdot n + k^3 \cdot n)$ time an instance $I' = (A', B', k)$ of* Max-Duo PSM *with $|A'|$ and $|B'|$ bounded by $O(k^3)$ such that I is a yes-instance if and only if I' is a yes-instance.*

We first show the running time to construct the kernel.

Proposition 1. *For each $S \in \{A, B\}$, $|C_S| = O(k^3)$ and C_S can be constructed in $O(|\Sigma|^2 \cdot n + k^3 n)$ time.*

Proof. By assumption $|rare(S)| \leq 4k$. Additionally, for each i, the ball $\mathcal{B}_k(i)$ has size at most $2k+1$. Finally, for each duo $(i, i+1)$ that is rare for S, the set $F(\overline{S}, i)$ has at most $(2k-1)(4k+1)$ duos. Therefore, $|C_S| \leq 4k(2k+1) + 4k(2k-1)(4k+1) = O(k^3)$.

Now let us analyze the time to construct C_S. First, the construction of the sets $rare(S)$ and $rare(\overline{S})$ takes $O(|\Sigma|^2 \cdot n)$ time, since we just need to count for each length-two string $ab \in \Sigma \times \Sigma$, the number of times $n(S, a, b)$ that ab occurs in S and the number of times $n(\overline{S}, a, b)$ that ab occurs in $\overline{S}$. Now, for each position $i \in \{1, ..., n-1\}$, we add $(i, i+1)$ to $rare(S)$ if $S[i, i+1] = ab$ and $n(S, a, b) \leq n(\overline{S}, a, b)$. Analogously, we add $(i, i+1)$ to $rare(\overline{S})$ if $\overline{S}[i, i+1] = ab$ and $n(\overline{S}, a, b) \leq n(S, a, b)$.

Now, the construction of the set $\textsc{Roots}(S, i)$ according to Algorithm 1 takes $O(k^2 \cdot n)$ time. Since $\textsc{Roots}(S, i) \leq 2k - 1$, and by assumption $|rare(S)| \leq 4k$, the construction of $F(S, i)$ also takes $O(k^2 \cdot n)$ time. Analogously, the construction of $F(\overline{S}, i)$ takes $O(k^2 \cdot n)$ time. Therefore, the construction of C_S takes $O(|\Sigma|^2 \cdot n + k^3 \cdot n)$ time. □

Proof of Theorem 4. First, if some map m of A into B preserves a block X of size k, then (A, B, k) is a yes-instance for Max-Duo PSM and we can output in $O(1)$ time an equivalent instance of constant size. We note that the existence of such a map m can be verified in $O(n)$ time by solving the Longest Common Substring problem for A and B.

Second, if $rare(A) \geq 4k$ or $rare(B) \geq 4k$, then (A, B, k) is a yes-instance for Max-Duo PSM, and we can output in $O(1)$ time an equivalent instance of constant size.

Now since no map preserves a block of size k, and if both $rare(A) < 4k$ and $rare(B) < 4k$, then by Lemma 16, the pair (C_A, C_B) constructed according to Equation 1 is complete for (A, B, k). Additionally, by Proposition 1, $|C_A| = |C_B| = O(k^3)$, and both C_A and C_B can be constructed in $O(|\Sigma| \cdot n + k^3 \cdot n)$ time.

Since the complete pair (C_A, C_B) constructed has size $O(k^3)$, we can apply Theorem 3 to construct in $O(k^3)$ time an instance (A', B', k) for Max-Duo PSM of size $O(k^3)$ such that (A', B', k) is a yes-instance if and only if (A, B, k) is a yes-instance. Therefore, the overall time to construct (A', B', k) is upper-bounded by $O(|\Sigma|^2 \cdot n + k^3 \cdot n)$. □

References

[1] S. Beretta, M. Castelli, and R. Dondi, *Corrigendum to "Parameterized tractability of the maximum-duo preservation string mapping problem" [Theoret. Comput. Sci. 646 (2016) 16–25]*, Theor. Comput. Sci., 653 (2016), pp. 108–10.

[2] ——, *Parameterized tractability of the maximum-duo preservation string mapping problem*, Theor. Comput. Sci., 646 (2016), pp. 16–25.

[3] A. Björklund, *Determinant sums for undirected hamiltonicity*, SIAM J. Comput., 43 (2014), pp. 280–299.

[4] A. Björklund, T. Husfeldt, P. Kaski, and M. Koivisto, *Narrow sieves for parameterized paths and packings*, J. Comput. Syst. Sci., 87 (2017), pp. 119–139.

[5] N. Boria, G. Cabodi, P. Camurati, M. Palena, P. Pasini, and S. Quer, *A 7/2-approximation algorithm for the maximum duo-preservation string mapping problem*, in Proceedings of the 27th Annual Symposium on Combinatorial Pattern Matching (CPM '16), R. Grossi and M. Lewenstein, eds., vol. 54 of LIPIcs, Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2016, pp. 11:1–11:8.

[6] N. Boria, A. Kurpisz, S. Leppänen, and M. Mastrolilli, *Improved approximation for the maximum duo-preservation string mapping problem*, in Proceedings of the 14th International Workshop on Algorithms in Bioinformatics (WABI '14), D. G. Brown and B. Morgenstern, eds., vol. 8701 of Lecture Notes in Computer Science, Springer, 2014, pp. 14–25.

[7] B. Brubach, *Further improvement in approximating the maximum duo-preservation string mapping problem*, in Proceedings of the 16th International Workshop on Algorithms in Bioinformatics (WABI '16), M. C. Frith and C. N. S. Pedersen, eds., vol. 9838 of Lecture Notes in Computer Science, Springer, 2016, pp. 52–64.

[8] L. Bulteau, G. Fertin, C. Komusiewicz, and I. Rusu, *A fixed-parameter algorithm for minimum common string partition with few duplications*, in Proceedings of the 13th International Workshop on Algorithms in Bioinformatics (WABI '13), A. E. Darling and J. Stoye, eds., 2013, pp. 244–258.

[9] L. Bulteau and C. Komusiewicz, *Minimum common string partition parameterized by partition size is fixed-parameter tractable*, in Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, (SODA '14), C. Chekuri, ed., SIAM, 2014, pp. 102–121.

[10] W. Chen, Z. Chen, N. F. Samatova, L. Peng, J. Wang, and M. Tang, *Solving the maximum duo-preservation string mapping problem with linear programming*, Theor. Comput. Sci., 530 (2014), pp. 1–11.

[11] X. Chen, J. Zheng, Z. Fu, P. Nan, Y. Zhong, S. Lonardi, and T. Jiang, *Assignment of orthologous genes via genome rearrangement*, IEEE/ACM T. Comput. Bi., 2 (2005), pp. 302–315.

[12] G. Cormode and S. Muthukrishnan, *The string edit distance matching problem with moves*, ACM T. Alg, 3 (2007), pp. 2:1–2:19.

[13] M. Crochemore, C. Hancart, and T. Lecroq, *Algorithms on strings*, Cambridge University Press, 2007.

[14] M. Cygan, F. V. Fomin, L. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, and S. Saurabh, *Parameterized Algorithms*, Springer, 2015.

[15] P. Damaschke, *Minimum common string partition parameterized*, in Proceedings of the 8th International Workshop on Algorithms in Bioinformatics (WABI '08), K. A. Crandall and J. Lagergren, eds., vol. 5251 of Lecture Notes in Computer Science, Springer, 2008, pp. 87–98.

[16] R. A. DeMillo and R. J. Lipton, *A probabilistic remark on algebraic program testing*, Inf. Process. Lett., 7 (1978), pp. 193–195.

[17] R. G. Downey and M. R. Fellows, *Fundamentals of Parameterized Complexity*, Texts in Computer Science, Springer, 2013.

[18] B. Dudek, P. Gawrychowski, and P. Ostropolski-Nalewaja, *A family of approximation algorithms for the maximum duo-preservation string mapping problem*, in 28th Annual Symposium on Combinatorial Pattern Matching, CPM 2017, July 4-6, 2017, Warsaw, Poland, 2017, pp. 10:1–10:14.

[19] G. Fertin, A. Labarre, I. Rusu, E. Tannier, and S. Vialette, *Combinatorics of Genome Rearrangements*, Computational Molecular Biology, MIT Press, 2009.

[20] F. V. Fomin, D. Lokshtanov, F. Panolan, and S. Saurabh, *Efficient computation of representative families with applications in parameterized and exact algorithms*, J. ACM, 63 (2016), pp. 29:1–29:60.

[21] F. V. Fomin, D. Lokshtanov, S. Saurabh, and M. Zehavi, *Kernelization: Theory of Parameterized Preprocessing.*, Cambridge University Press, 2019.

[22] A. Goldstein, P. Kolman, and J. Zheng, *Minimum common string partition problem: Hardness and approximations*, Electron. J. Comb., 12 (2005).

[23] D. Gusfield, *Algorithms on Strings, Trees, and Sequences - Computer Science and Computational Biology*, Cambridge University Press, 1997.

[24] H. Jiang, B. Zhu, D. Zhu, and H. Zhu, *Minimum common string partition revisited*, J. Comb. Optim., 23 (2012), pp. 519–527.

[25] J. T. Schwartz, *Fast probabilistic algorithms for verification of polynomial identities*, J. ACM, 27 (1980), pp. 701–717.

[26] H. Shachnai and M. Zehavi, *Representative families: A unified tradeoff-based approach*, J. Comput. Syst. Sci., 82 (2016), pp. 488–502.

[27] K. M. Swenson, M. Marron, J. V. Earnest-DeYoung, and B. M. E. Moret, *Approximating the true evolutionary distance between two genomes*, ACM J. Exp. Alg., 12 (2008).

[28] Y. Xu, Y. Chen, T. Luo, and G. Lin, *A local search 2.917-approximation algorithm for duo-preservation string mapping*, CoRR, arXiv:1702.01877 (2017).

[29] R. Zippel, *Probabilistic algorithms for sparse polynomials*, in Proceedings of the International Symposium on Symbolic and Algebraic Computation (EUROSAM '79), E. W. Ng, ed., vol. 72 of Lecture Notes in Computer Science, Springer, 1979, pp. 216–226.