A $(1.4 + \epsilon)$-Approximation Algorithm for the 2-Max-Duo Problem[*][†][‡]

Yao Xu[1], Yong Chen[§][2], Guohui Lin[¶][3], Tian Liu[‖][4], Taibo Luo[**][5], and Peng Zhang[††][6]

1 **Department of Computing Science, University of Alberta, Edmonton, Canada**
`xu2@ualberta.ca`

2 **Department of Computing Science, University of Alberta, Edmonton, Canada and Department of Mathematics, Hangzhou Dianzi University, Zhejiang, China**
`chenyong@hdu.edu.cn`

3 **Department of Computing Science, University of Alberta, Edmonton, Canada**
`guohui@ualberta.ca`

4 **Key Laboratory of High Confidence Software Technologies (MOE), Institute of Software, School of Electronic Engineering and Computer Science, Peking University, Beijing, China**
`lt@pku.edu.cn`

5 **Business School, Sichuan University, Chengdu, China and Department of Computing Science, University of Alberta, Edmonton, Canada**
`taibo@ualberta.ca`

6 **School of Computer Science and Technology, Shandong University, Shandong, China**
`algzhang@sdu.edu.cn`

Abstract

The *maximum duo-preservation string mapping* (Max-Duo) problem is the complement of the well studied *minimum common string partition* (MCSP) problem, both of which have applications in many fields including text compression and bioinformatics. k-Max-Duo is the restricted version of Max-Duo, where every letter of the alphabet occurs at most k times in each of the strings, which is readily reduced into the well known *maximum independent set* (MIS) problem on a graph of maximum degree $\Delta \leq 6(k-1)$. In particular, 2-Max-Duo can then be approximated arbitrarily close to 1.8 using the state-of-the-art approximation algorithm for the MIS problem. 2-Max-Duo was proved APX-hard and very recently a $(1.6+\epsilon)$-approximation was claimed, for any $\epsilon > 0$. In this paper, we present a vertex-degree reduction technique, based on which, we show that 2-Max-Duo can be approximated arbitrarily close to 1.4.

1998 ACM Subject Classification F.2.2 Pattern matching, G.2.1 Combinatorial algorithms, G.4 Algorithm design and analysis

Keywords and phrases Approximation algorithm, duo-preservation string mapping, string partition, independent set

Digital Object Identifier 10.4230/LIPIcs.ISAAC.2017.66

[*] This work was partially supported by NSERC Canada and NSF China.
[†] A full version of this paper can be found at `https://arxiv.org/abs/1702.0625606256`.
[‡] All authors are supported by NSERC Canada.
[§] Chen is supported by the NSFC Grants No. 11401149, 11571252 and 11571087, and the China Scholarship Council Grant No. 201508330054.
[¶] Correspondence authors. Lin is supported by the NSFC Grant No. 61672323.
[‖] Liu is supported by the NSFC Grant Nos. 61370052 and 61370156.
[**] Luo is supported by the NSFC Grant No. 71371129 and the PSF China Grant No. 2016M592680.
[††] Zhang is supported by the NSFC Grant No. 61672323.

1 Introduction

The *minimum common string partition* (MCSP) problem is a well-studied string comparison problem in computer science, with applications in fields such as text compression and bioinformatics. MCSP was first introduced by Goldstein *et al.* [16], and can be defined as follows: Consider two length-n strings $A = (a_1, a_2, \ldots, a_n)$ and $B = (b_1, b_2, \ldots, b_n)$ over some alphabet Σ, such that B is a permutation of A. Let $\mathcal{P}_A$ be a *partition* of A, which is a multi-set of substrings whose concatenation in a certain order becomes A. The *cardinality* of $\mathcal{P}_A$ is the number of substrings in $\mathcal{P}_A$. The MCSP problem asks to find a minimum cardinality partition $\mathcal{P}_A$ of A which is also a partition of B. k-MCSP denotes the restricted version of MCSP where every letter of the alphabet Σ occurs at most k times in each of the two strings.

Goldstein *et al.* [16] have shown that the MCSP problem is NP-hard and APX-hard, even when $k = 2$. There have been several approximation algorithms [10, 11, 12, 16, 18, 19] presented since 2004, among which the current best result is an $O(\log n \log^* n)$-approximation for the general MCSP and an $O(k)$-approximation for k-MCSP. On the other hand, MCSP is proved to be *fixed parameter tractable* (FPT), with respect to k and/or the cardinality of the optimal partition [13, 17, 7, 8].

An ordered pair of consecutive letters in a string is called a *duo* of the string [16], which is said to be *preserved* by a partition if the pair resides inside a substring of the partition. Therefore, a length-ℓ substring in the partition *preserves* $\ell - 1$ duos of the string. With the complementary objective to that of MCSP, the problem of maximizing the number of duos preserved in the common partition is referred to as the *maximum duo-preservation string mapping* problem by Chen *et al.* [9], denoted as Max-Duo. Analogously, k-Max-Duo is the restricted version of Max-Duo where every letter of the alphabet Σ occurs at most k times in each string. In this paper, we focus on 2-Max-Duo, to design an improved approximation algorithm.

Along with Max-Duo, Chen *et al.* [9] introduced the *constrained maximum induced subgraph* (CMIS) problem, in which one is given an m-partite graph $G = (V_1, V_2, \ldots, V_m, E)$ with each V_i having n_i^2 vertices arranged in an $n_i \times n_i$ matrix, and the goal is to find n_i vertices in each V_i from different rows and different columns such that the number of edges in the induced subgraph is maximized. k-CMIS is the restricted version of CMIS where $n_i \le k$ for all i. Given an instance of Max-Duo, we may construct an instance of CMIS by setting m to be the number of distinct letters in the string A, and n_i to be the number of occurrences of the i-th distinct letter; the vertex in the (s,t)-entry of the $n_i \times n_i$ matrix "means" mapping the s-th occurrence of the i-th distinct letter in the string A to its t-th occurrence in the string B; and there is an edge between a vertex of V_i and a vertex of V_j if the two corresponding mappings together preserve a duo. Therefore, Max-Duo is a special case of CMIS, and furthermore k-Max-Duo is a special case of k-CMIS. Chen *et al.* [9] presented a k^2-approximation for k-CMIS and a 2-approximation for 2-CMIS, based on a linear programming and randomized rounding techniques. These imply that k-Max-Duo can also be approximated within a ratio of k^2 and 2-Max-Duo can be approximated within a ratio of 2.

Alternatively, an instance of the k-Max-Duo problem with the two strings $A = (a_1, a_2, \ldots, a_n)$ and $B = (b_1, b_2, \ldots, b_n)$ can be viewed as a bipartite graph $H = (A, B, F)$, constructed as follows: The vertices in A and B are $a_1, a_2, \ldots, a_n$ in order and $b_1, b_2, \ldots, b_n$ in order, respectively, and there is an edge between a_i and b_j if they are the same letter. The two edges $(a_i, b_j), (a_{i+1}, b_{j+1}) \in F$ are called a pair of *parallel* edges. This way, a common partition of the strings A and B corresponds one-to-one to a perfect matching in H, and the number of duos preserved by the partition is exactly the number of pairs of parallel edges in the matching.

Moreover, from the bipartite graph $H = (A, B, F)$, we can construct another graph $G = (V, E)$ in which every vertex of V corresponds to a pair of parallel edges of F, and there is an edge between two vertices of V if the two corresponding pairs of parallel edges of F *cannot* co-exist in any perfect matching of H (called *conflicting*, which can be determined in constant time; see Section 2 for more details). This way, one easily sees that a set of duos that can be preserved together, by a perfect matching of H, corresponds one-to-one to an independent set of G [16, 5]. Therefore, the Max-Duo problem can be cast as a special case of the well-known *maximum independent set* (MIS) problem [15]; furthermore, Boria *et al.* [5] showed that in such a reduction, an instance of k-Max-Duo gives rise to a graph with a maximum degree $\Delta \le 6(k-1)$. It follows that the state-of-the-art $\big((\Delta+3)/5+\epsilon\big)$-approximation algorithm for MIS [2], for any $\epsilon > 0$, is a $\big((6k-3)/5+\epsilon\big)$-approximation algorithm for k-Max-Duo. Especially, 2-Max-Duo can now be better approximated within a ratio of $1.8+\epsilon$. Boria *et al.* [5] proved that 2-Max-Duo is APX-hard, similar to 2-MCSP [16], via a linear reduction from MIS on cubic graphs. For MIS on cubic graphs, it is NP-hard to approximate within 1.00719 [3]. Besides, Boria *et al.* [5] claimed that 2-Max-Duo can be approximated within $1.6+\epsilon$, for any $\epsilon > 0$.

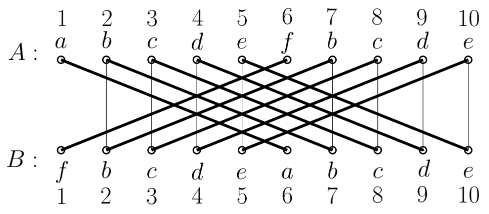
Recently, Boria *et al.* [4] presented a local search 3.5-approximation for the general Max-Duo problem. In the meantime, Brubach [6] presented a 3.25-approximation using a novel *combinatorial triplet matching*. Max-Duo has also been proved to be FPT by Beretta *et al.* [1], with respect to the number of preserved duos in the optimal partition. Most recently, two local search algorithms were independently designed for the general Max-Duo problem at the same time, achieving approximation ratios of 2.917 [20] and $2+\epsilon$ [14] for any $\epsilon > 0$, respectively. They both exceed the previously the best $\big((6k-3)/5+\epsilon\big)$-approximation algorithm for k-Max-Duo, when $k \ge 3$. In this paper, we focus on the 2-Max-Duo problem; using the above reduction to the MIS problem, we present a vertex-degree reduction scheme and design an improved $(1.4+\epsilon)$-approximation, for any $\epsilon > 0$.

The rest of the paper is organized as follows. We provide some preliminaries in Section 2, including several important structural properties of the graph constructed from the two given strings. The vertex-degree reduction scheme is also presented as a separate subsection in Section 2. The new approximation algorithm, denoted as Approx, is presented in Section 3, where we show that it is a $(1.4+\epsilon)$-approximation for 2-Max-Duo. We conclude the paper in Section 4.

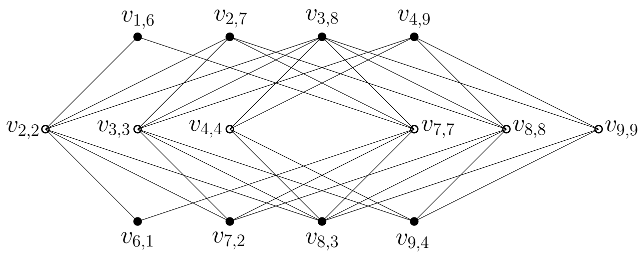
2 Preliminaries

Consider an instance of the k-Max-Duo problem with two length-n strings $A = (a_1, a_2, \ldots, a_n)$ and $B = (b_1, b_2, \ldots, b_n)$ such that B is a permutation of A. Recall that we can view the instance as a bipartite graph $H = (A, B, F)$, where the vertices in A and B are $a_1, a_2, \ldots, a_n$ in order and $b_1, b_2, \ldots, b_n$ in order, respectively, and there is an edge between $a_i \in A$ and $b_j \in B$ if they are the same letter, denoted as $e_{i,j}$. See Figure 2.1a for an example, where $A = (a, b, c, d, e, f, b, c, d, e)$ and $B = (f, b, c, d, e, a, b, c, d, e)$. Note that $|F| \le kn$, and so H can be constructed in $O(n^2)$ time.

The two edges $e_{i,j}, e_{i+1,j+1} \in F$ are called a pair of *parallel* edges (and they are said to be parallel to each other); when both are included in a perfect matching of H, the corresponding duo (a_i, a_{i+1}) of A is preserved. Two pairs of parallel edges are *conflicting* if they cannot co-exist in any perfect matching of H. This motivates the following reduction from the



(a) The bipartite graph $H = (A, B, F)$, where the ten edges in bold form a perfect matching.



(b) The instance graph $G = (V, E)$ of MIS, where the eight filled vertices form an independent set.

Figure 2.1 An instance of the k-Max-Duo problem with $A = (a, b, c, d, e, f, b, c, d, e)$ and $B = (f, b, c, d, e, a, b, c, d, e)$. Figure 2.1a is the graphical view as a bipartite graph $H = (A, B, F)$, where a perfect matching consisting of the ten bold edges form into eight pairs of parallel edges, corresponding to the eight preserved duos $(a, b), (b, c), (c, d), (d, e), (f, b), (b, c), (c, d)$ and (d, e). Figure 2.1b shows the instance graph $G = (V, E)$ of MIS constructed from H, where the independent set $\{v_{1,6}, v_{2,7}, v_{3,8}, v_{4,9}, v_{6,1}, v_{7,2}, v_{8,3}, v_{9,4}\}$ corresponds to the eight pairs of parallel edges shown in Figure 2.1a, and consequently also corresponds to the eight preserved duos. In this instance, we have $k = 2$. Any maximum independent set of G must contain some of the degree-6 vertices, invalidating the $(1.6 + \epsilon)$-approximation for 2-Max-Duo proposed in [5].

k-Max-Duo problem to the MIS problem: From the bipartite graph $H = (A, B, F)$, we construct another graph $G = (V, E)$ in which a vertex $v_{i,j}$ of V corresponds to the pair of parallel edges $(e_{i,j}, e_{i+1,j+1})$ of F; two vertices of V are *conflicting* if and only if the two corresponding pairs of parallel edges are conflicting, and two conflicting vertices of V are adjacent in G. One can see that a set of duos of A that can be preserved all together, a set of pairwise non-conflicting pairs of parallel edges of F, and an independent set in G, are equivalent to each other. See Figure 2.1b for an example of the graph $G = (V, E)$ constructed from the bipartite graph H shown in Figure 2.1a. We note that $|V| \le k(n-1)$ and thus G can be constructed in $O(k^2n^2)$ time from the instance of the k-Max-Duo problem.

In the graph G, for any $v \in V$, we use $N(v)$ to denote the set of its neighbors, that is, the vertices adjacent to v. The two ordered letters in the duo corresponding to the vertex v is referred to as the *letter content* of v. For example, in Figure 2.1b, the letter content of $v_{1,6}$ is "ab" and the letter content of $v_{6,1}$ is "fb".

Recall from the construction that there is an edge $e_{i,j}$ in the graph $H = (A, B, F)$ if $a_i = b_j$, and there is a vertex $v_{i,j}$ in the graph $G = (V, E)$ if the parallel edges $e_{i,j}$ and $e_{i+1,j+1}$ are in $H = (A, B, F)$.

▶ **Lemma 2.1.** *The graph $G = (V, E)$ has the following properties.*

1. *If $v_{i,j}$, $v_{i+2,j+2} \in V$, then $v_{i+1,j+1} \in V$.*
2. *Given any subset of vertices $V' \subset V$, let $F' = \{e_{i,j} | v_{i,j} \in V'\}$, $A' = \{a_i | e_{i,j} \in F'\}$,*

and $B' = \{b_j | e_{i,j} \in F'\}$. If the subgraph $H' = (A', B', F')$ in H is connected, then all the vertices of V' have the same letter content; and consequently for any two vertices $v_{i,j}, v_{h,\ell} \in V'$, we have both $v_{h,j}, v_{i,\ell} \in V$.

3. *For any $v_{i,j} \in V$, we have*

$$N(v_{i,j}) = \bigcup_{p=-1,0,1} \{v_{i'+p,j+p} \in V \mid i' \neq i\} \cup \bigcup_{p=-1,0,1} \{v_{i+p,j'+p} \in V \mid j' \neq j\}. \quad (1)$$

Proof. The proof is mostly based on the definitions, or how the graphs are constructed from the instance of the 2-Max-Duo problem. Due to the space limit, the interested reader can find the detailed proofs of several earlier lemmas and corollaries in the full version. ◀

From Lemma 2.1 and its proof (in the full version), we see that for any vertex of V there are at most $k - 1$ conflicting vertices of each kind (corresponding to a set in Equation (1)). We thus have the following corollary.

▶ **Corollary 2.2.** *The maximum degree of the vertices in $G = (V, E)$ is $\Delta \leq 6(k - 1)$.*

2.1 When $k = 2$

We examine more properties for the graph $G = (V, E)$ when $k = 2$. First, from Corollary 2.2 we have $\Delta \leq 6$.

Berman and Fujito [2] have presented an approximation algorithm with a performance ratio arbitrarily close to $(\Delta + 3)/5$ for the MIS problem, on graphs with maximum degree Δ. This immediately implies a $(1.8 + \epsilon)$-approximation for 2-Max-Duo. Our goal is to reduce the maximum degree of the graph $G = (V, E)$ to achieve a better approximation algorithm. To this purpose, we examine all the degree-6 and degree-5 vertices in the graph G, and show a scheme to safely remove them from consideration when computing an independent set. This gives rise to a new graph G_2 with maximum degree at most 4, leading to a desired $(1.4 + \epsilon)$-approximation for 2-Max-Duo.

We remark that, in our scheme we first remove the degree-6 vertices from G to compute an independent set, and later we add half of these degree-6 vertices to the computed independent set to become the final solution. Contrary to the claim that there always exists a maximum independent set in G containing no degree-6 vertices [5, Lemma 1], the instance in Figure 2.1 shows that any maximum independent set for the instance must contain some degree-6 vertices, thus invalidating the $(1.6 + \epsilon)$-approximation for 2-Max-Duo proposed in [5].

In more details, the instance of 2-Max-Duo, illustrated in Figure 2.1, consists of two length-10 strings $A = (a, b, c, d, e, f, b, c, d, e)$ and $B = (f, b, c, d, e, a, b, c, d, e)$. The bipartite graph $H = (A, B, F)$ is shown in Figure 2.1a and the instance graph $G = (V, E)$ of the MIS problem is shown in Figure 2.1b. In the graph G, we have six degree-6 vertices: $v_{2,2}, v_{7,7}, v_{3,3}, v_{3,8}, v_{8,3}$ and $v_{8,8}$. One can check that $\{v_{1,6}, v_{2,7}, v_{3,8}, v_{4,9}, v_{6,1}, v_{7,2}, v_{8,3}, v_{9,4}\}$ is an independent set in G, of size 8. On the other hand, if none of these degree-6 vertices is included in an independent set, then because the four vertices $v_{4,4}, v_{4,9}, v_{9,4}, v_{9,9}$ form a square implying that at most two of them can be included in the independent set, the independent set would be of size at most 6, and thus can never be a maximum independent set in G.

Consider a duo (a_i, a_{i+1}) of the string A and for ease of presentation assume its letter content is "ab". If no duo of the string B has the same letter content "ab", then this duo of the string A can never be preserved; in fact this duo does not even become (a part of) a vertex of V of the graph G. If there is exactly one duo (b_j, b_{j+1}) of the string B having the same letter content "ab", then these two duos make up a vertex $v_{i,j} \in V$, and from Lemma 2.1 we know that the degree of the vertex $v_{i,j} \in V$ is at most 5, since there is no such vertex $v_{i,j'}$ with $j' \neq j$ sharing exactly the two letters a_i and a_{i+1} with $v_{i,j}$. Therefore, if the degree of the vertex $v_{i,j} \in V$ is six, then there must be two duos of the string A and two duos of the string B having the same letter content "*ab*". Assume the other duo of the string A and the other duo of the string B having the same letter content "*ab*" are $(a_{i'}, a_{i'+1})$ and $(b_{j'}, b_{j'+1})$, respectively. Then all four vertices $v_{i,j}, v_{i,j'}, v_{i',j}, v_{i',j'}$ exist in V. We call the subgraph of G induced on these four vertices a *square*, and denote it as $S(i, i'; j, j') = (V(i, i'; j, j'), E(i, i'; j, j'))$, where $V(i, i'; j, j') = \{v_{i,j}, v_{i,j'}, v_{i',j}, v_{i',j'}\}$ and $E(i, i'; j, j') = \{(v_{i,j}, v_{i,j'}), (v_{i,j}, v_{i',j}), (v_{i',j'}, v_{i,j'}), (v_{i',j'}, v_{i',j})\}$ due to their conflicting relationships. One clearly sees that every square has a unique letter content, which is the letter content of its four member vertices.

In Figure 2.1b, there are three squares $S(2, 7; 2, 7)$, $S(3, 8; 3, 8)$ and $S(4, 9; 4, 9)$, with their letter contents "*bc*", "*cd*" and "*de*", respectively. The above argument says that every degree-6 vertex of V must belong to a square, but the converse is not necessarily true, for example, all vertices of the square $S(4, 9; 4, 9)$ have degree 4. We next characterize several properties of a square.

The following lemma is a direct consequence of how the graph G is constructed and the fact that $k = 2$.

▶ **Lemma 2.3.** *In the graph $G = (V, E)$ constructed from an instance of* 2-Max-Duo*,*

1. *for each index i, there are at most two distinct j and j' such that $v_{i,j}, v_{i,j'} \in V$;*
2. *if $v_{i,j}, v_{i,j'} \in V$ where $j' \neq j$, and $v_{i+1,j''+1} \in V$ (or symmetrically, $v_{i-1,j''-1} \in V$), then either $j'' = j$ or $j'' = j'$.*

▶ **Lemma 2.4.** *For any square $S(i, i'; j, j')$ in the graph $G = (V, E)$, $N(v_{i,j}) = N(v_{i',j'})$, $N(v_{i,j'}) = N(v_{i',j})$, and $N(v_{i,j}) \cap N(v_{i,j'}) = \emptyset$. (Together, these imply that every vertex of V is adjacent to either none or exactly two of the four member vertices of a square.)*

▶ **Corollary 2.5.** *In the graph $G = (V, E)$, the degree-6 vertices can be partitioned into pairs, where each pair of degree-6 vertices belong to a square in G and they are adjacent to the same six other vertices, two inside the square and four outside of the square.*

Proof. We have seen that every degree-6 vertex in the graph G must be in a square. The above Lemma 2.4 states that the four vertices of a square $S(i, i'; j, j')$ can be partitioned into two pairs, $\{v_{i,j}, v_{i',j'}\}$ and $\{v_{i,j'}, v_{i',j}\}$, and the two vertices inside each pair are non-adjacent to each other and have the same neighbors. In particular, if the vertex $v_{i,j}$ in the square $S(i, i'; j, j')$ has degree 6, then Lemma 2.1 states that it is adjacent to the six vertices $v_{i-1,j'-1}, v_{i,j'}, v_{i+1,j'+1}, v_{i'-1,j-1}, v_{i',j}, v_{i'+1,j+1}$ (see an illustration in Figure 2.2). ◀

▶ **Corollary 2.6.** *If there is no square in the graph $G = (V, E)$, then every degree-5 vertex is adjacent to a degree-1 vertex.*

We say the two vertices $v_{i,j}$ and $v_{i+1,j+1}$ of V are *consecutive*; and we say the two squares $S(i, i'; j, j')$ and $S(i + 1, i' + 1; j + 1, j' + 1)$ in G are *consecutive*. Clearly, two consecutive squares contain four pairs of consecutive vertices. The following Lemma 2.7 summarizes the fact that when two consecutive vertices belong to two different squares, then these two squares are also consecutive (and thus contain the other three pairs of consecutive vertices).

▶ **Lemma 2.7.** *In the graph G, if there are two consecutive vertices $v_{i,j}$ and $v_{i+1,j+1}$ belonging to two different squares $S(i_1, i'_1; j_1, j'_1)$ and $S(i_2, i'_2; j_2, j'_2)$ respectively, then $i_2 = i_1 + 1, i'_2 = i'_1 + 1, j_2 = j_1 + 1, j'_2 = j'_1 + 1$, i.e., these two squares are consecutive.*

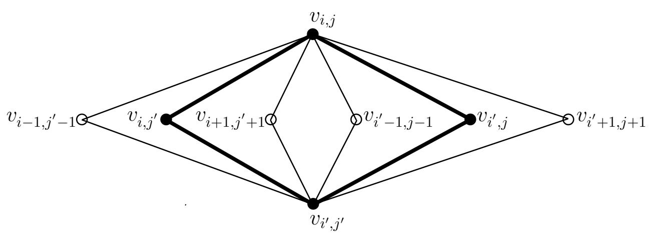


Figure 2.2 The square $S(i, i'; j, j')$ shown in bold lines. The two non-adjacent vertices $v_{i,j}$ and $v_{i',j'}$ of the square form a pair stated in Corollary 2.5; they have 6 common neighbors, of which two inside the square and four outside of the square.

A series of p consecutive squares $\{S(i+q, i'+q; j+q, j'+q), q = 0, 1, \ldots, p-1\}$ in the graph G, where $p \geq 1$, is *maximal* if none of the square $S(i-1, i'-1; j-1, j'-1)$ and the square $S(i+p, i'+p; j+p, j'+p)$ exists in the graph G. Note that the non-existence of the square $S(i-1, i'-1; j-1, j'-1)$ in G does not rule out the existence of some of the four vertices $v_{i-1,j-1}, v_{i'-1,j'-1}, v_{i-1,j'-1}, v_{i'-1,j-1}$ in V; in fact by Lemma 2.1 there can be as many as two of these four vertices existing in V (however, more than two would imply the existence of the square). Similarly, there can be as many as two of the four vertices $v_{i+p,j+p}, v_{i'+p,j'+p}, v_{i+p,j'+p}, v_{i'+p,j+p}$ existing in V. In the sequel, a maximal series of p consecutive squares starting with $S(i, i'; j, j')$ is denoted as $\mathcal{S}^p(i, i'; j, j')$, where $p \geq 1$. See for an example in Figure 2.3b where there is a maximal series of 2 consecutive squares $\mathcal{S}^2(2, 8; 2, 8)$, where the instance of the 2-Max-Duo is expanded slightly from the instance shown in Figure 2.1.

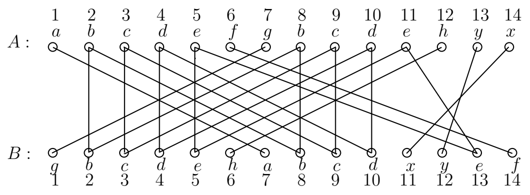
▶ **Lemma 2.8.** *Suppose $\mathcal{S}^p(i, i'; j, j')$, where $p \geq 1$, exists in the graph G. Then,*

1. *the two substrings $(a_i, a_{i+1}, \ldots, a_{i+p})$ and $(a_{i'}, a_{i'+1}, \ldots, a_{i'+p})$ of the string A and the two substrings $(b_j, b_{j+1}, \ldots, b_{j+p})$ and $(b_{j'}, b_{j'+1}, \ldots, b_{j'+p})$ of the string B are identical and do not overlap;*
2. *if a maximum independent set of G contains less than $2p$ vertices from $\mathcal{S}^p(i, i'; j, j')$, then it must contain either the four vertices $v_{i-1,j-1}, v_{i'-1,j'-1}, v_{i'+p,j+p}, v_{i+p,j'+p}$ or the four vertices $v_{i'-1,j-1}, v_{i-1,j'-1}, v_{i+p,j+p}, v_{i'+p,j'+p}$.*

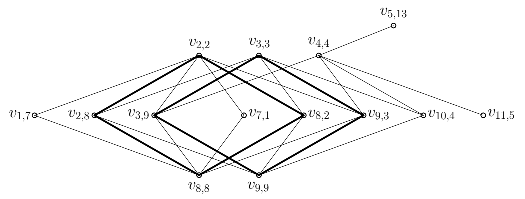
Proof. By the definition of the square $S(i+q, i'+q; j+q, j'+q)$, we have $a_{i+q} = a_{i'+q}$ and $a_{i+q+1} = a_{i'+q+1}$; we thus conclude that the two substrings $(a_i, a_{i+1}, \ldots, a_{i+p})$ and $(a_{i'}, a_{i'+1}, \ldots, a_{i'+p})$ are identical. In Figure 2.3b, for $\mathcal{S}^2(2, 8; 2, 8)$ the two substrings are "*bcd*". If these two substrings overlapped, then there would be three occurrences of at least one letter, contradicting the fact that $k = 2$. This proves the first item.

Note that the square $S(i-1, i'-1; j-1, j'-1)$ does not exist in the graph G, and thus at most two of its four vertices (which are $v_{i-1,j-1}, v_{i'-1,j-1}, v_{i-1,j'-1}$ and $v_{i'-1,j'-1}$) exist in V. We claim that if no vertex of the square $S(i, i'; j, j')$ is in I^*, then there are exactly two of the four vertices $v_{i-1,j-1}, v_{i'-1,j-1}, v_{i-1,j'-1}$ and $v_{i'-1,j'-1}$ exist in V and they both are in I^*. Suppose otherwise there is at most one of the four vertices in I^*, say $v_{i-1,j-1}$; we may increase the size of I^* by removing $v_{i-1,j-1}$ while adding either the two vertices $v_{i,j}$ and $v_{i',j'}$ or the two vertices $v_{i',j}$ and $v_{i,j'}$ (depending on which vertices of the square $S(i+1, i'+1; j+1, j'+1)$ are in I^*), a contradiction.

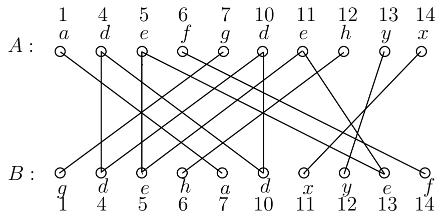
Assume next that a vertex of the square $S(i, i'; j, j')$ is in I^*, say $v_{i,j}$; then due to maximality of I^* and Lemma 2.4 both $v_{i,j}$ and $v_{i',j'}$ are in I^*. We claim and prove similarly as in the last paragraph that if no vertex of the square $S(i+1, i'+1; j+1, j'+1)$ is in I^*, then there are exactly two of the four vertices $v_{i-1,j-1}, v_{i'-1,j-1}, v_{i-1,j'-1}$ and $v_{i'-1,j'-1}$ exist in V



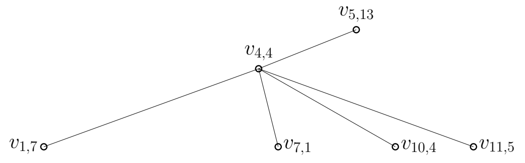
(a) The bipartite graph $H = (A, B, F)$.



(b) The instance graph $G = (V, E)$.



(c) The bipartite graph $H' = (A', B', F')$ after removal of $\mathcal{S}^2(2, 8; 2, 8)$.



(d) The updated instance graph $G' = (V', E')$ after removal of $\mathcal{S}^2(2, 8; 2, 8)$.

Figure 2.3 An instance of the 2-Max-Duo problem with $A = (a, b, c, d, e, f, g, b, c, d, e, h, y, x)$ and $B = (g, b, c, d, e, h, a, b, c, d, x, y, e, f)$. The bipartite graph $H = (A, B, F)$ is shown in Figure 2.3a and the instance graph $G = (V, E)$ of the MIS problem is shown in Figure 2.3b. There is a maximal series of 2 squares $\mathcal{S}^2(2, 8; 2, 8)$ in the graph G, with the four substrings "bcd". The bipartite graph $H' = (A', B', F')$ is shown in Figure 2.3c and the graph $G' = (V', E')$ is shown in Figure 2.3d, on $A' = (a, d, e, f, g, d, e, h, y, x)$ and $B' = (g, d, e, h, a, d, x, y, e, f)$. Applying the vertex contracting process on G also gives the graph G'.

and they both are in I^*. If there is a vertex of the square $S(i+1, i'+1; j+1, j'+1)$ in I^*, then it must be one of $v_{i+1,j+1}$ and $v_{i'+1,j'+1}$; and due to maximality and Lemma 2.4 both $v_{i+1,j+1}$ and $v_{i'+1,j'+1}$ are in I^*. And so on; repeatedly applying this argument, we claim and prove similarly that if no vertex of the square $S(i+p-1, i'+p-1; j+p-1, j'+p-1)$ is in I^*, then there are exactly two of the four vertices $v_{i-1,j-1}, v_{i'-1,j-1}, v_{i-1,j'-1}$ and $v_{i'-1,j'-1}$ exist in V and they both are in I^*. If there is a vertex of the square $S(i+p-1, i'+p-1; j+p-1, j'+p-1)$ in I^*, then it must be one of $v_{i+p-1,j+p-1}$ and $v_{i'+p-1,j'+p-1}$; and due to maximality and Lemma 2.4 both $v_{i+p-1,j+p-1}$ and $v_{i'+p-1,j'+p-1}$ are in I^*.

To summarize, we proved in the above two paragraphs that if I^* contains less than $2p$ vertices from $\mathcal{S}^p(i, i'; j, j')$, then there are exactly two of the four vertices $v_{i-1,j-1}, v_{i'-1,j-1}$, $v_{i-1,j'-1}$ and $v_{i'-1,j'-1}$ exist in V and they both are in I^*; and these two vertices are either $v_{i-1,j-1}$ and $v_{i'-1,j'-1}$ or $v_{i'-1,j-1}$ and $v_{i-1,j'-1}$. Symmetrically, there are exactly two of the four vertices $v_{i+p,j+p}, v_{i'+p,j+p}, v_{i+p,j'+p}$ and $v_{i'+p,j'+p}$ exist in V and they both are in I^*; and these two vertices are either $v_{i+p,j+p}$ and $v_{i'+p,j'+p}$ or $v_{i'+p,j+p}$ and $v_{i+p,j'+p}$. Clearly from the above, when the combination is $v_{i-1,j-1}$ and $v_{i'-1,j'-1}$ versus $v_{i+p,j+p}$ and $v_{i'+p,j'+p}$, we may increase the size of I^* to contain exactly $2p$ vertices from $\mathcal{S}^p(i, i'; j, j')$ without affecting any vertex outside of $\mathcal{S}^p(i, i'; j, j')$, a contradiction. Therefore, the only possible combinations are $v_{i-1,j-1}$ and $v_{i'-1,j'-1}$ versus $v_{i'+p,j+p}$ and $v_{i+p,j'+p}$, and $v_{i'-1,j-1}$ and $v_{i-1,j'-1}$ versus $v_{i+p,j+p}$ and $v_{i'+p,j'+p}$. This proves the second item of the lemma. ◀

Suppose $\mathcal{S}^p(i, i'; j, j')$, where $p \geq 1$, exists in the graph G. Let A' denote the string obtained from A by removing the two substrings $(a_i, a_{i+1}, \ldots, a_{i+p-1})$ and $(a_{i'}, a_{i'+1}, \ldots, a_{i'+p-1})$ and concatenating the remainder together, and B' denote the string obtained from B by removing the two substrings $(b_j, b_{j+1}, \ldots, b_{j+p-1})$ and

Algorithm 3.1 Approx – A high-level description of the approximation algorithm for 2-Max-Duo.

```
1: Construct the graph $G = (V, E)$ from two input strings $A$ and $B$;
2: while (there is a square in the graph) do
3:     find a maximal series of squares;
4:     locate the four identical substrings of $A$ and $B$ as in Lemma 2.8;
5:     remove the corresponding substrings and accordingly update the graph;
6: end while
7: denote the resultant graph as $G_1 = (V_1, E_1)$;
8: set $L_1$ to contain all degree-0 and degree-1 vertices of $G_1$;
9: set $N[L_1]$ to be the closed neighborhood of $L_1$ in $G_1$, i.e. $N[L_1] = L_1 \cup N(L_1)$;
10: set $G_2 = G_1[V_1 - N[L_1]]$, the subgraph of $G_1$ induced on $V_1 - N[L_1]$;
11: compute an independent set $I_2$ in $G_2$ by the $((\Delta + 3)/5 + \epsilon)$-approximation in [2];
12: set $I_1 = I_2 \cup L_1$, an independent set in $G_1$;
13: return an independent set $I$ in $G$ using $I_1$ and Corollary 2.9.
```

$(b_{j'}, b_{j'+1}, \ldots, b_{j'+p-1})$ and concatenating the remainder. Let the graph $G' = (V', E')$ denote the instance graph of the MIS problem constructed from the two strings A' and B'. See for an example G' in Figure 2.3d, where there is a maximal series of 2 consecutive squares $\mathcal{S}^2(2, 8; 2, 8)$ in the graph G.

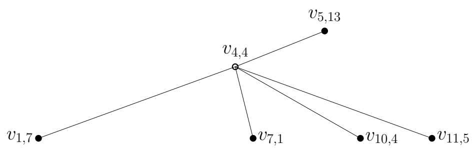
▶ **Corollary 2.9.** *Suppose $\mathcal{S}^p(i, i'; j, j')$, where $p \geq 1$, exists in the graph G. Then, the union of a maximum independent set in the graph $G' = (V', E')$ and certain $2p$ vertices from $\mathcal{S}^p(i, i'; j, j')$ becomes a maximum independent set in the graph $G = (V, E)$, where these certain $2p$ vertices are $v_{i,j}, v_{i+1,j+1}, \ldots, v_{i+p-1,j+p-1}$ and $v_{i',j'}, v_{i'+1,j'+1}, \ldots, v_{i'+p-1,j'+p-1}$ if $v_{i-1,j-1}$ or $v_{i+p,j+p}$ is in the maximum independent set in G', or they are $v_{i',j}, v_{i'+1,j+1}, \ldots,$ $v_{i'+p-1,j+p-1}$ and $v_{i,j'}, v_{i+1,j'+1}, \ldots, v_{i+p-1,j'+p-1}$ if $v_{i'-1,j-1}$ or $v_{i'+p,j+p}$ is in the maximum independent set in G'.*

Iteratively applying the above string shrinkage process, or equivalently the vertex contracting process, associated with the elimination of a maximal series of consecutive squares. In $O(n)$ iterations, we achieve the final graph containing no squares, which we denote as $G_1 = (V_1, E_1)$.

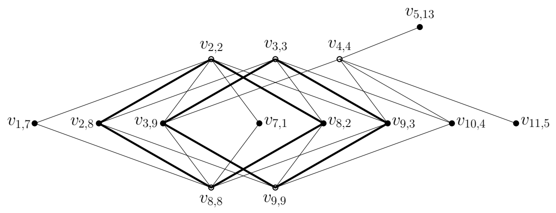
3 An approximation algorithm for 2-Max-Duo

A high-level description of the approximation algorithm, denoted as Approx, for the 2-Max-Duo problem is depicted in Algorithm 3.1.

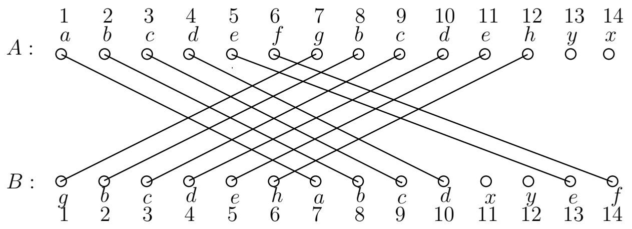
In more details, given an instance of the 2-Max-Duo problem with two length-n strings A and B, the first step of our algorithm is to construct the graph $G = (V, E)$, which is done in $O(n^2)$ time. In the second step (Lines 2–7 in Algorithm 3.1), it iteratively applies the vertex contracting process presented in Section 2 at the existence of a maximal series of consecutive squares, and at the end it achieves the final graph $G_1 = (V_1, E_1)$ which does not contain any square. This second step can be done in $O(n^2)$ time too since each iteration of vertex contracting process is done in $O(n)$ time and there are $O(n)$ iterations. In the third step (Lines 8–10 in Algorithm 3.1), let L_1 denote the set of singletons (degree-0 vertices) and leaves (degree-1 vertices) in the graph G_1; our algorithm removes all the vertices of L_1 and their neighbors from the graph G_1 to obtain the remainder graph $G_2 = (V_2, E_2)$. This step can be done in $O(n^2)$ time too due to $|V_1| \leq |V| \leq 2n$, and the resultant graph G_2 has maximum degree $\Delta \leq 4$ by Corollaries 2.5 and 2.6. (See for an example illustrated in Figure 3.1a.) In the fourth step (Lines 11–12 in Algorithm 3.1), our algorithm calls the



(a) The independent set $I_1 = \{v_{1,7}, v_{7,1}, v_{10,4}, v_{11,5}, v_{5,13}\}$ in G_1, consisting of all the five leaves of $G_1 = G'$ shown in Figure 2.3d.



(b) Using I_1, since $v_{10,4} \in I_1$, the four vertices $v_{2,8}, v_{3,9}, v_{8,2}, v_{9,3}$ are added to form an independent set I in the original graph G shown in Figure 2.3b.



(c) The parallel edges of H corresponding to the independent set I shown in Figure 3.1b, also correspond to the 9 preserved duos $(a, b), (b, c), (c, d), (e, f)$, $(g, b), (b, c), (c, d), (d, e), (e, h)$ for the instance shown in Figure 2.3a.

Figure 3.1 Illustration of the execution of our algorithm Approx on the instance shown in Figure 2.3. The independent set I_1 in the graph G_1 is shown in Figure 3.1a in filled circles, for which we did not apply the state-of-the-art approximation algorithm for the MIS problem. The independent set I in the graph G is shown in Figure 3.1b in filled circles, according to Corollary 2.9 the four vertices $v_{2,8}, v_{3,9}$, $v_{8,2}, v_{9,3}$ are added due to $v_{10,4} \in I_1$. The parallel edges of H corresponding to the vertices of I are shown in Figure 3.1c, representing a feasible solution to the 2-Max-Duo instance shown in Figure 2.3.

state-of-the-art approximation algorithm for the MIS problem [2] on the graph G_2 to obtain an independent set I_2 in G_2; and returns $I_1 = L_1 \cup I_2$ as an independent set in the graph G_1. The running time of this step is dominated by the running time of the state-of-the-art approximation algorithm for the MIS problem, which is a high polynomial in n and $1/\epsilon$. In the last step (Line 13 in Algorithm 3.1), using the independent set I_1 in G_1, our algorithm adds $2p$ vertices from each maximal series of p consecutive squares according to Corollary 2.9, to produce an independent set I in the graph G. (For an illustrated example see Figure 3.1b.) The last step can be done in $O(n)$ time.

The state-of-the-art approximation algorithm for the MIS problem on a graph with maximum degree Δ has a performance ratio of $(\Delta + 3)/5 + \epsilon$, for any $\epsilon > 0$ [2].

▶ **Lemma 3.1.** *In the graph $G_1 = (V_1, E_1)$, let OPT_1 denote the cardinality of a maximum independent set in G_1, and let SOL_1 denote the cardinality of the independent set I_1 returned by the algorithm* Approx*. Then, $\text{OPT}_1 \le (1.4 + \epsilon)\text{SOL}_1$, for any $\epsilon > 0$.*

▶ **Theorem 3.2.** *The* 2-Max-Duo *problem can be approximated within a ratio arbitrarily close to* 1.4*, by a linear reduction to the* MIS *problem.*

Proof. We prove by induction. At the presence of maximal series of p consecutive squares, we perform the vertex contracting process iteratively. In each iteration to handle one maximal series of p consecutive squares, let G and G' denote the graph before and after the contracting step, respectively. Let OPT' denote the cardinality of a maximum independent set in G', and let SOL' denote the cardinality of the independent set I' returned by the algorithm Approx. Given any $\epsilon > 0$, from Lemma 3.1, we may assume that $\mathrm{OPT}' \le (1.4 + \epsilon)\mathrm{SOL}'$.

Let OPT denote the cardinality of a maximum independent set in G, and let SOL denote the cardinality of the independent set returned by the algorithm Approx, which adds $2p$ vertices from the maximal series of p consecutive squares to the independent set I' in G', according to Corollary 2.9, to produce an independent set I in the graph G. Lemma 2.8 states that $\mathrm{OPT} = \mathrm{OPT}' + 2p$. Therefore,

$$\mathrm{OPT} = \mathrm{OPT}' + 2p \le (1.4 + \epsilon)\mathrm{SOL}' + 2p \le (1.4 + \epsilon)(\mathrm{SOL}' + 2p) = (1.4 + \epsilon)\mathrm{SOL}.$$

This proves that for the original graph $G = (V, E)$ we also have $\mathrm{OPT} \le (1.4 + \epsilon)\mathrm{SOL}$ accordingly. That is, the worst-case performance ratio of our algorithm Approx is $1.4 + \epsilon$, for any $\epsilon > 0$. The time complexity of the algorithm Approx has been determined to be polynomial at the beginning of the section, and it is dominated by the time complexity of the state-of-the-art approximation algorithm for the MIS problem. The theorem is thus proved. ◀

4 Conclusion

In this paper, we examined the Max-Duo problem, the complement of the well studied *minimum common string partition* problem. Based on an existing linear reduction to the *maximum independent set* (MIS) problem [16, 5], we presented a vertex-degree reduction technique for the 2-Max-Duo to reduce the maximum degree of the constructed instance graph to 4. Along the way, we uncovered many interesting structural properties of the constructed instance graph. This degree reduction enables us to adopt the state-of-the-art approximation algorithm for the MIS problem on low degree graphs [2] to achieve a $(1.4 + \epsilon)$-approximation for 2-Max-Duo, for any $\epsilon > 0$.

It is worth mentioning that our vertex-degree reduction technique can be applied for k-Max-Duo with $k \ge 3$. In fact, we had worked out the details for $k = 3$, to reduce the maximum degree of the constructed instance graph from 12 to 10, leading to a $(2.6 + \epsilon)$-approximation for 3-Max-Duo, for any $\epsilon > 0$. Nevertheless, the $(2.6 + \epsilon)$-approximation is superseded by the $(2 + \epsilon)$-approximation for the general Max-Duo [14].

It would be worthwhile to investigate whether the maximum degree can be further reduced to 3, by examining the structural properties associated with the degree-4 vertices. On the other hand, it is also interesting to examine whether a better-than-1.4 approximation algorithm can be designed directly for the MIS problem on those degree-4 graphs obtained at the end of the vertex contracting process.

References

1 S. Beretta, M. Castelli, and R. Dondi. Parameterized tractability of the maximum-duo preservation string mapping problem. *Theoretical Computer Science*, 646:16–25, 2016.

2 P. Berman and T. Fujito. On approximation properties of the independent set problem for low degree graphs. *Theory of Computing Systems*, 32:115–132, 1999.

3 P. Berman and M. Karpinski. On some tighter inapproximability results. In *Proceedings of the of 26th International Colloquium on Automata, Languages and Programming (ICALP'99)*, pages 200–209, 1999.

4 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 2016), volume 54 of *LIPIcs*, pages 11:1–11:8, 2016.

5 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 2014)*, volume 8701 of *LNBI*, pages 14–25, 2014.

6 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 2016)*, volume 9838 of *LNBI*, pages 52–64, 2016.

7 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 2013)*, volume 8126 of *LNBI*, pages 244–258, 2013.

8 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)*, pages 102–121, 2014.

9 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. *Theoretical Computer Science*, 530:1–11, 2014.

10 X. Chen, J. Zheng, Z. Fu, P. Nan, Y. Zhong, S. Lonardi, and T. Jiang. Assignment of orthologous genes via genome rearrangement. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 2:302–315, 2005.

11 M. Chrobak, P. Kolman, and J. Sgall. The greedy algorithm for the minimum common string partition problem. In *Proceedings of the 7th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX 2004) and the 8th International Workshop on Randomization and Computation (RANDOM 2004)*, volume 3122 of *LNCS*, pages 84–95, 2004.

12 G. Cormode and S. Muthukrishnan. The string edit distance matching problem with moves. *ACM Transactions on Algorithms*, 3:2:1–2:19, 2007.

13 P. Damaschke. Minimum common string partition parameterized. In *Proceedings of the 8th International Workshop on Algorithms in Bioinformatics (WABI 2008)*, volume 5251 of *LNBI*, pages 87–98, 2008.

14 B. Dudek, P. Gawrychowski, and P. Ostropolski-Nalewaja. A family of approximation algorithms for the maximum duo-preservation string mapping problem. *arXiv*, 1702.02405, 2017.

15 M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-completeness.* W. H. Freeman and Company, San Francisco, 1979.

16 A. Goldstein, P. Kolman, and J. Zheng. Minimum common string partition problem: Hardness and approximations. In *Proceedings of the 15th International Symposium on Algorithms and Computation (ISAAC 2004)*, volume 3341 of *LNCS*, pages 484–495, 2004.

17 H. Jiang, B. Zhu, D. Zhu, and H. Zhu. Minimum common string partition revisited. *Journal of Combinatorial Optimization*, 23:519–527, 2012.

18 P. Kolman and T. Waleń. Reversal distance for strings with duplicates: Linear time approximation using hitting set. In *Proceedings of the 4th International Workshop on Approximation and Online Algorithms (WAOA 2006)*, volume 4368 of *LNCS*, pages 279–289, 2006.

19 P. Kolman and T. Waleń. Approximating reversal distance for strings with bounded number of duplicates. *Discrete Applied Mathematics*, 155:327–336, 2007.

20 Y. Xu, Y. Chen, T. Luo, and G. Lin. A local search 2.917-approximation algorithm for duo-preservation string mapping. *arXiv*, 1702.01877, 2017.