Journal of Combinatorial Optimization (2020) 40:806–824
https://doi.org/10.1007/s10878-020-00621-0

A $(1.4 + \epsilon)$-approximation algorithm for the 2-MAX-DUO problem

Yong Chen[1] · Guohui Lin[2] · Tian Liu[3] · Taibo Luo[4] · Bing Su[5] · Yao Xu[6] · Peng Zhang[7]

Published online: 25 July 2020

Abstract

The *maximum duo-preservation string mapping* (MAX- DUO) problem is the complement of the well studied *minimum common string partition* 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 $\Delta \le 6(k-1)$. 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 on bounded-degree graphs. 2-MAX- DUO was proved APX-hard and very recently a $(1.6 + \epsilon)$-approximation algorithm 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.

Keywords Approximation algorithm · Duo-preservation string mapping · String partition · Independent set

Mathematics Subject Classification F.2.2 Pattern matching · G.2.1 Combinatorial algorithms · G.4 Algorithm design and analysis

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. In both text compression and bioinformatics, string (or sequence) comparison is a routine work. For the similarity between two strings, a

An extended abstract appears in Proceedings of the 28th International Symposium on Algorithms and Computation (ISAAC 2017). LIPICS 92, Article No. 66, pp. 66:1–66:12.

Extended author information available on the last page of the article commonly used measure is the *edit distance*, which is the minimum number of edit operations required to transform one string into the other. At the finest scale, the edit operations involve a single character of a string, including insertion, deletion, and substitution. When comparing two long strings such as the whole genomes of multiple species, long range operations become more interesting, leading to the genome rearrangement problems (Chen et al. 2005; Swenson et al. 2008). In particular, a transportation operation is to cut out a substring and insert it back at another position in the string. The problem of partitioning one string into a minimum number of substrings such that a reshuffle of them becomes the other string is referred to as the *minimum common string partition* (MCSP) problem.

MCSP was first introduced by Goldstein et al. (2004), 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 re-ordering 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 given strings.

Goldstein et al. (2004) have shown that the MCSP problem is NP-hard and APX-hard, even when $k = 2$. There have been several approximation algorithms (Chen et al. 2005; Chrobak et al. 2004; Cormode and Muthukrishnan 2007; Goldstein et al. 2004; Kolman and Waleń 2006, 2007) proposed since 2004, among which the current best result is an $O(\log n \log^* n)$-approximation algorithm for the general MCSP and an $O(k)$-approximation algorithm for k-MCSP. On the other hand, MCSP is proved to be *fixed parameter tractable* (FPT), with respect to the cardinalities of the parts in an optimal partition and/or a combination of the cardinalities and k (Damaschke 2008; Jiang et al. 2012; Bulteau et al. 2013; Bulteau and Komusiewicz 2014).

In a given string A, a pair of adjacent letters in A is called a *duo* of the string A (Goldstein et al. 2004); a length-ℓ substring in a partition of A *preserves* $\ell - 1$ duos of A. The complementary objective to that of MCSP is to maximize the number of duos preserved in a common partition of A and B, and such an optimization problem is referred to as the *maximum duo-preservation string mapping* (MPSM) problem by Chen et al. (2014). In this paper, we call this MPSM problem as Max- Duo, mostly because the acronym MPSM looks too similar to the other acronyms. 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 given string. Max- Duo was proved to be FPT by Beretta et al. (2016a, b), with respect to the number of preserved duos in the optimal partition. In this paper, we focus on 2-Max- Duo, to design an improved approximation algorithm.

Along with Max- Duo, Chen et al. (2014) 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 $n_i \leq k$ 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. This way, Max- Duo becomes a special case of CMIS, and furthermore k-Max- Duo is a special case of k-CMIS. Chen et al. (2014) presented a k^2-approximation algorithm for k-CMIS and a 2-approximation algorithm 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. (See for an illustration in Fig. 1a, and more details in Sect. 2.)

Moreover, from the bipartite graph $H = (A, B, F)$, we can construct another graph $G = (V, E)$ in which a vertex of V one-to-one 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 (such two pairs of parallel edges are called *conflicting*, which can be determined in constant time; see Sect. 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, one-to-one corresponds to an independent set of G (Goldstein et al. 2004; Boria et al. 2014). Therefore, the Max-Duo problem can be cast as a special case of the well-known *maximum independent set* (MIS) problem (Garey and Johnson 1979); furthermore, Boria et al. (2014) 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 (Berman and Fujito 1999), 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. (2014) proved that 2-Max- Duo is APX-hard, similar to 2-MCSP (Goldstein et al. 2004), via a linear reduction from MIS on cubic graphs (denoted as 3- MIS), for which it is NP-hard to approximate within 1.00719 (Berman and Karpinski 1999). Besides, Boria et al. (2014) claimed that 2-Max- Duo can be approximated within $1.6+\epsilon$, for any $\epsilon > 0$, but their proof is flawed (see Sect. 2 for a counterexample).

Recently, Boria et al. (2016) presented a local search 3.5-approximation algorithm for the general Max- Duo problem. In the meantime, Brubach (2016) presented a 3.25-approximation algorithm using a novel *combinatorial triplet matching*. Most recently, two new local search algorithms were independently designed for the general Max- Duo problem at the same time, achieving approximation ratios of 2.917 (Xu et al. 2017) and $2+\epsilon$ (Dudek et al. 2017) for any $\epsilon > 0$, respectively. They both beat the previously 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 algorithm, for any $\epsilon > 0$. The following chart summarizes the approximation results for the 2-Max- Duo problem:

$$2\,\text{(Chen et al. 2014)} \longrightarrow 1.8 + \epsilon\ \text{(Boria et al. 2014)}\ (\rightarrow \text{falsely claimed } 1.6 + \epsilon\ \text{(Boria et al. 2014)}) \longrightarrow 1.4 + \epsilon\ \text{(Theorem 3.2)}.$$

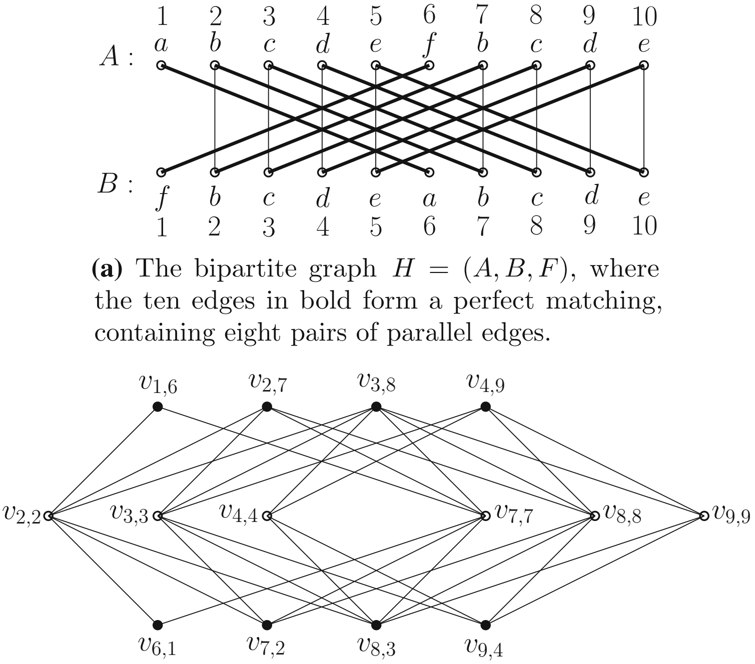
The rest of the paper is organized as follows. We provide some preliminaries in Sect. 2, including several important structural properties of the graph constructed from the two given strings. The vertex-degree reduction scheme is also presented as a separate subsection in Sect. 2. The new approximation algorithm, denoted as Approx, is presented in Sect. 3, where we show that it is a $(1.4 + \epsilon)$-approximation algorithm for 2-Max- Duo. In Sect. 4, we review the APX-hardness reduction from 3- MIS to 2-Max- Duo and point out a direction for better approximating 2-Max- Duo. We conclude the paper in Sect. 5.

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 re-ordering 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 Fig. 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 H can be constructed in $O(n^2)$ time, and $|F| \le kn$.

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 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. We remark that in general the graph G is not bipartite. 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 Fig. 1b for an example of the graph $G = (V, E)$ constructed from the bipartite graph H shown in Fig. 1a. We note that $|V| \le k(n - 1)$ and thus G can be constructed in $O(k^2 n^2)$ 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 are referred to as the *letter content* of v. For example, in Fig. 1b, the letter content of $v_{1,6}$ is "ab" and the letter content of $v_{6,1}$ is "fb".



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

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

Fig. 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 1a is the graphical view as a bipartite graph $H = (A, B, F)$, where a perfect matching consisting of the ten bold edges; these ten 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 1b shows the instance graph $G = (V, E)$ of the MIS problem 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 Fig. 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 algorithm for 2-Max- Duo proposed in Boria et al. (2014)

Recall from the construction that there is an edge $e_{i,j}$ in the graph $H = (A, B, F)$ if $a_i = b_j$, (therefore the induced subgraph by a specific letter in H is a complete bipartite graph), 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*

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

Proof By definition, $v_{i,j} \in V$ if and only if $e_{i,j}, e_{i+1,j+1} \in F$.

1. If also $v_{i+2,j+2} \in V$, that is, $e_{i+2,j+2}, e_{i+3,j+3} \in F$, then $e_{i+1,j+1}, e_{i+2,j+2} \in F$ leading to $v_{i+1,j+1} \in V$.
2. Note that an edge $e_{i,j} \in F$ if and only if the two vertices a_i and b_j are the same letter, and clearly each connected component in H is complete bipartite and all the vertices are the same letter. It follows that if the induced subgraph $H' = (A', B', F')$ in H is connected, then all its vertices are the same letter. Let $F'' = \{e_{i+1,j+1} | v_{i,j} \in V'\}$, $A'' = \{a_i | e_{i,j} \in F''\}$, and $B'' = \{b_j | e_{i,j} \in F''\}$. The subgraph $H'' = (A'', B'', F'')$ in H has exactly the same topology as H', and thus it is also connected and all its vertices are the same letter. Therefore, all the vertices of V' have the same letter content; and consequently for any two vertices $v_{i,j}, v_{h,\ell} \in V'$, both $v_{h,j}, v_{i,\ell} \in V$.
3. For any vertex $v_{i,j}$, or equivalently the pair of parallel edges $(e_{i,j}, e_{i+1,j+1})$ in F, which are incident at four vertices $a_i, a_{i+1}, b_j, b_{j+1}$, a conflicting pair of parallel edges can be one of the following six kinds: to share exactly one of a_i and a_{i+1}, to share both a_i and a_{i+1}, to share exactly one of b_j and b_{j+1}, and to share both b_j and b_{j+1}. The sets of these six kinds of conflicting pairs are as described in Eq. (1), for example, $\{v_{i'-1,j-1} \in V \mid i' \neq i\}$ is the set of conflicting pairs each sharing b_j (but not b_{j+1}) with the pair $v_{i,j}$.

This proves the lemma. □

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

Corollary 2.2 (Boria et al. 2014) *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 (1999) 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 algorithm 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 algorithm 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 (Boria et al. 2014, Lemma 1), the instance in Fig. 1 shows that any maximum independent set for the instance must contain some degree-6 vertices, thus invalidating the $(1.6+\epsilon)$-approximation algorithm for 2-Max- Duo proposed in Boria et al. (2014).

In more details, the instance of 2-Max- Duo, illustrated in Fig. 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 Fig. 1a and the instance graph $G = (V, E)$ of the MIS problem is shown in Fig. 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 8-set in G. 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 maximum in G.

Consider a duo (a_i, a_{i+1}) of the string A. If its letter content is "aa", then either there is no vertex of V with letter content "aa" or there is exactly one vertex of V with letter content "aa" which is an isolated vertex in G. We thus assume without loss of generality that the letter content is "ab", where $a \neq b$.

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, no vertex of V would have its letter content "ab"). 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 both 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 Fig. 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 $k = 2$.

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.)*

Proof Consider the two vertices $v_{i,j}$ and $v_{i',j'}$, which have common neighbors $v_{i,j'}$ and $v_{i',j}$ in the square.

Note that $v_{i,j'}$ and $v_{i,j}$ share both a_i and a_{i+1}. If there is a vertex adjacent to $v_{i,j}$ by sharing a_{i+1} but not a_i, then this vertex is $v_{i+1,j''+1}$ with $j'' \neq j$, and thus it has to be $v_{i+1,j'+1}$ (by Lemma 2.3). We consider two subcases: If $i + 1 = i' - 1$ (*i.e.*, $(a_i, a_{i+1}, a_{i'}, a_{i'+1}) =$ "*abab*" is a substring of A), then $j' + 1 = j - 1$ (*i.e.*, $(b_{j'}, b_{j'+1}, b_j, b_{j+1}) =$ "*abab*" is a substring of B) due to $k = 2$. Thus, this vertex $v_{i+1,j'+1}$ is adjacent to $v_{i',j'}$ too, but not adjacent to $v_{i,j'}$ or $v_{i',j}$.

If $i + 1 \neq i' - 1$, then we conclude that $a_{i+2} \notin \{a, b\}$ and thus this vertex $v_{i+1,j'+1}$ shares only a_{j+1} with the vertex $v_{i,j}$, shares only $b_{j'+1}$ with $v_{i',j'}$, shares exactly a_{i+1} and $b_{j'+1}$ with $v_{i,j'}$, and shares none of $\{a_{i'}, a_{i'+1}, b_j, b_{j+1}\}$ with $v_{i',j}$. That is, $v_{i+1,j'+1}$ is adjacent to $v_{i',j'}$ too, but not adjacent to $v_{i,j'}$ or $v_{i',j}$.

The other three symmetric cases can be discussed exactly the same and the lemma is proved. □

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 Fig. 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.*

Proof Assume the vertex $v_{i,j}$ has degree 5. Due to the non-existence of any square in the graph G and Lemma 2.1, either there is no vertex sharing both a_i and a_{i+1} with $v_{i,j}$, or there is no vertex sharing both b_j and b_{j+1} with $v_{i,j}$, but not both. We assume without loss of generality that there is no vertex sharing both a_i and a_{i+1} with $v_{i,j}$, and furthermore assume $v_{i',j}$, $i' \neq i$, is the vertex sharing both b_j and b_{j+1} with $v_{i,j}$.

It follows that $N(v_{i,j}) = \{v_{i-1,j''-1}, v_{i+1,j'''+1}, v_{i'-1,j-1}, v_{i',j}, v_{i'+1,j+1}\}$, for some $j'' \neq j$ and $j''' \neq j$. Due to $k = 2$, this implies that $a_{i-1} \neq b_{j-1} = a_{i'-1}$ and $a_{i+2} \neq b_{j+2} = a_{i'+2}$. Therefore, if a vertex of V shares $a_{i'}$ but not $a_{i'+1}$ ($a_{i'+1}$ but

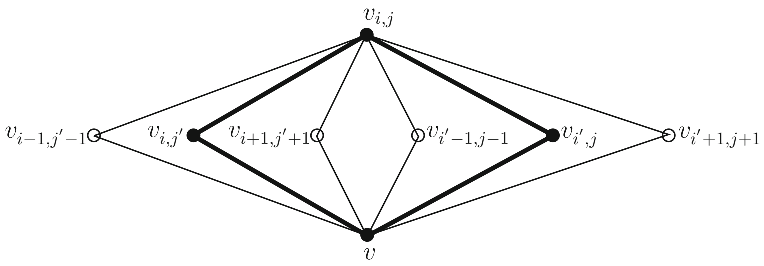


Fig. 2 The square $S(i, i'; j, j')$ shown in bold lines. The two non-adjacent vertices $v_{i,j}$ and $v_{i',j'}$ of the square form a pair stated in Corollary 2.5; they have 6 common neighbors, of which two are inside the square and four are outside of the square

not $a_{i'}$, b_j but not b_{j+1}, b_{j+1} but not b_j, respectively) with the vertex $v_{i',j}$, then it also shares b_j (b_{j+1}, $a_{i'}$, $a_{i'+1}$, respectively) with the vertex $v_{i',j}$; no vertex of V shares both $a_{i'}$ and $a_{i'+1}$ with the vertex $v_{i',j}$. That is, the vertex $v_{i',j}$ is adjacent to only $v_{i,j}$ in the graph G. □

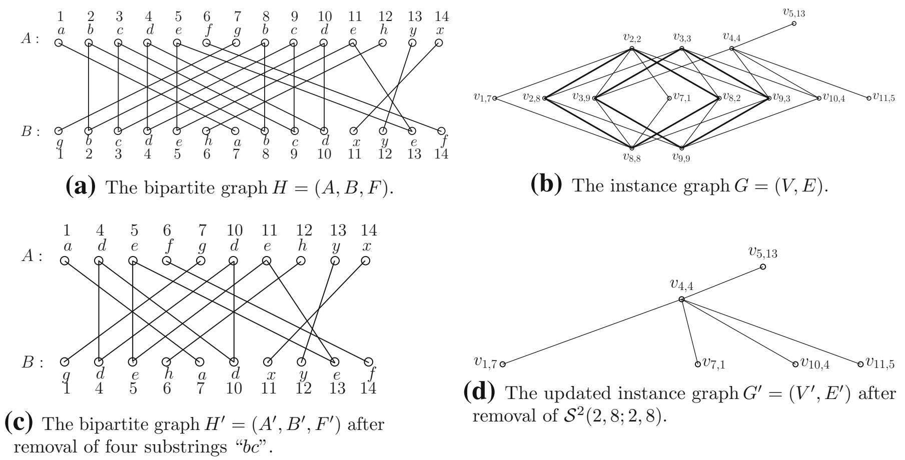
We say the two vertices $v_{i,j}$ and $v_{i+1,j+1}$ of V are *consecutive*; and we say the two squares $S(i, i'; j, j')$ and $S(i+1, i'+1; j+1, j'+1)$ in G are *consecutive*. Clearly, two consecutive squares contain four pairs of consecutive vertices. The following Lemma 2.7 summarizes the fact that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

Proof This is a direct result of the fact that no two distinct squares have any member vertex in common, due to each square having its unique letter content. □

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 Fig. 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 Fig. 1.

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



(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 four substrings "bc".

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

Fig. 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 Fig. 3a and the constructed instance graph $G = (V, E)$ of the MIS problem is shown in Fig. 3b. There is a maximal series of 2 squares $\mathcal{S}^2(2, 8; 2, 8)$ in G, associated with the four substrings "bcd". After the removal of the four substrings "bc", we achieve $A' = (a, d, e, f, g, d, e, h, y, x)$ and $B' = (g, d, e, h, a, d, x, y, e, f)$, for which the bipartite graph $H' = (A', B', F')$ is shown in Fig. 3c and the corresponding graph $G' = (V', E')$ is shown in Fig. 3d

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 are identical and do not overlap; 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 I^* 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 $S(i + q, i' + q; j + q, j' + q)$, for each $q = 0, 1, \ldots, p - 1$, 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 Fig. 3b, for $\mathcal{S}^2(2, 8; 2, 8)$ the two substrings are "bcd". If these two substrings overlap, then there would be three occurrences of at least one letter, contradicting the fact that $k = 2$. This proves the first item.

Next, for the square $S(i + q, i' + q; j + q, j' + q)$, if one vertex, say $v_{i+q,j+q}$, is in the maximum independent set I^*, then due to maximality of I^* and Lemma 2.4 another vertex, $v_{i'+q,j'+q}$ in this case, is also in I^*. That is, either none or exactly two vertices of the square $S(i + q, i' + q; j + q, j' + q)$ are in I^*. It follows that if I^* contains less than $2p$ vertices from $\mathcal{S}^p(i, i'; j, j')$, then there is at least one square of which no vertex is in I^*. Assume r is the least index such that no vertex of the square $S(i + r, i' + r; j + r, j' + r)$ is in I^*, and assume without loss of generality that at least one of the vertices $v_{i+r+1,j+r+1}$ and $v_{i'+r+1,j'+r+1}$ is in I^*.

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 $v_{i-1,j-1}$, $v_{i'-1,j-1}$, $v_{i-1,j'-1}$, $v_{i'-1,j'-1}$) exist in V. We claim that there are exactly two of these four vertices

$v_{i-1,j-1}, v_{i'-1,j-1}, v_{i-1,j'-1}, v_{i'-1,j'-1}$ exist in V and they both are in I^*. Suppose otherwise there is at most one of these four vertices in I^*, say $v_{i-1,j-1}$; we may increase the size of I^* by removing $v_{i-1,j-1}$ together with the two vertices of the square $S(i+q, i'+q; j+q, j'+q)$, for each $q = 0, 1, \ldots, r-1$, while adding the two vertices $v_{i+q,j+q}$ and $v_{i'+q,j'+q}$ of the square $S(i+q, i'+q; j+q, j'+q)$, for each $q = 0, 1, \ldots, r$, that is, removing $2r+1$ while adding $2r+2$ vertices, a contradiction. Again from Lemma 2.4, these two vertices existing in V are either $v_{i-1,j-1}, v_{i'-1,j'-1}$, or $v_{i'-1,j-1}, v_{i-1,j'-1}$.

A symmetric argument shows that either $v_{i+p,j+p}, v_{i'+p,j'+p} \in I^*$, or $v_{i'+p,j+p}, v_{i+p,j'+p} \in I^*$.

Lastly, if $v_{i-1,j-1}, v_{i'-1,j'-1} \in I^*$ and $v_{i+p,j+p}, v_{i'+p,j'+p} \in I^*$ ($v_{i'-1,j-1}, v_{i-1,j'-1} \in I^*$ and $v_{i'+p,j+p}, v_{i+p,j'+p} \in I^*$, respectively), then I^* can be expanded to include the two vertices $v_{i+q,j+q}$ and $v_{i'+q,j'+q}$ ($v_{i'+q,j+q}$ and $v_{i+q,j'+q}$, respectively) of the square $S(i+q, i'+q; j+q, j'+q)$, for each $q = 0, 1, \ldots, p-1$, a contradiction to the maximality of I^*. 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 $(b_{j'}, b_{j'+1}, \ldots, b_{j'+p-1})$ 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 Fig. 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 I' 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 $\{v_{i+q,j+q}, v_{i'+q,j'+q} \mid q = 0, 1, \ldots, p-1\}$ if $v_{i-1,j-1} \in I'$ or $v_{i+p,j+p} \in I'$, or they are $\{v_{i'+q,j+q}, v_{i+q,j'+q} \mid q = 0, 1, \ldots, p-1\}$ if $v_{i'-1,j-1} \in I'$ or $v_{i'+p,j+p} \in I'$.*

Proof Consider the construction of the graph $G' = (V', E')$ from the two strings A' and B'. Equivalently, starting with the graph $G = (V, E)$, if we

- Contract the p vertices $\{v_{i+q,j+q} \mid q = 0, 1, \ldots, p-1\}$ into the vertex $v_{i+p,j+p}$ if it exists or otherwise into a void vertex,
- Contract the p vertices $\{v_{i'+q,j'+q} \mid q = 0, 1, \ldots, p-1\}$ into the vertex $v_{i'+p,j'+p}$ if it exists or otherwise into a void vertex,
- Contract the p vertices $\{v_{i'+q,j+q} \mid q = 0, 1, \ldots, p-1\}$ into the vertex $v_{i'+p,j+p}$ if it exists or otherwise into a void vertex, and
- Contract the p vertices $\{v_{i+q,j'+q} \mid q = 0, 1, \ldots, p-1\}$ into the vertex $v_{i+p,j'+p}$ if it exists or otherwise into a void vertex,

then we obtain a graph that is exactly G'. In the graph G', the vertices $v_{i-1,j-1}$ and $v_{i'+p,j+p}$, if both exist in V, become adjacent to each other; so are the vertices $v_{i'-1,j-1}$ and $v_{i+p,j+p}$, if both exist in V. It follows that the maximum independent set I' in the graph $G' = (V', E')$ does not contain both vertices $v_{i-1,j-1}$ and $v_{i'+p,j+p}$, or both

```
Algorithm APPROX
 1: Construct the graph $G = (V, E)$ from the 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 stated in Lemma 2.8;
 5:     remove these four 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 algorithm in [3];
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.
```

Fig. 4 A high-level description of the approximation algorithm for 2-MAX- DUO

vertices $v_{i'-1,j-1}$ and $v_{i+p,j+p}$. Therefore, starting with I', we can add exactly $2p$ vertices from $\mathcal{S}^p(i, i'; j, j')$ to form an independent set in G, of which the maximality can be proved by a simple contradiction.

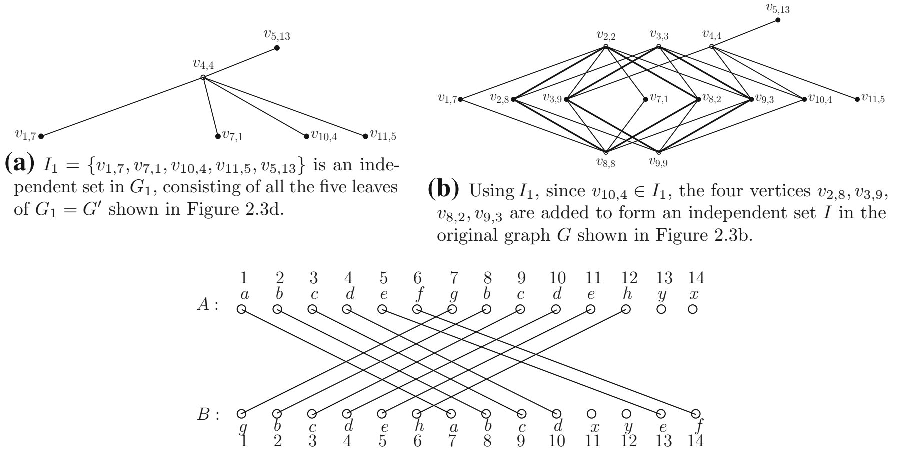
We remark that in the extreme case where none of the vertices of $S(i-1, i'-1; j-1, j'-1)$ and none of the vertices of $S(i+p, i'+p; j+p, j'+p)$ are in I', we may add either of the two sets of $2p$ vertices from $\mathcal{S}^p(i, i'; j, j')$ to form a maximum independent set in G. □

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 Fig. 4.

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 Fig. 4), it iteratively applies the vertex contracting process presented in Sect. 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 the vertex contracting process is done in $O(n)$ time and there are $O(n)$ iterations. In the third step (Lines 8–10 in Fig. 4), 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 $|V_1| \le |V| \le 2n$, and the resultant graph G_2 has maximum degree $\Delta \le 4$ by Corollaries 2.5 and 2.6 . (See for an example illustrated in Fig. 5a.) In the fourth step (Lines 11–12 in Fig. 4), our algorithm calls the state-of-the-art approximation algorithm for the MIS problem (Berman and Fujito 1999) 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



(a) $I_1 = \{v_{1,7}, v_{7,1}, v_{10,4}, v_{11,5}, v_{5,13}\}$ is an independent set 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 3.2b, 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.

Fig. 5 Illustration of the execution of our algorithm Approx on the instance shown in Fig. 3. The independent set I_1 in the graph G_1 is shown in Fig. 5a 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 Fig. 5b 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 Fig. 5c, representing a feasible solution to the 2-Max- Duo instance shown in Fig. 3

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 Fig. 4), 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 Fig. 5b.) 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 $\epsilon > 0$ (Berman and Fujito 1999).

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 *returned by the algorithm* Approx*. Then,* $\mathrm{OPT}_1 \le (1.4 + \epsilon)\mathrm{SOL}_1$*, for any* $\epsilon > 0$.

Proof Recall that L_1 denotes the set of singletons (degree-0 vertices) and leaves (degree-1 vertices) in the graph G_1; our algorithm Approx removes all the vertices of L_1 and their neighbors from the graph G_1 to obtain the remainder graph $G_2 = (V_2, E_2)$. The graph G_2 has maximum degree $\Delta \le 4$ by Corollaries 2.5 and 2.6 . Let OPT_2 denote the cardinality of a maximum independent set in G_2, and let SOL_2 denote the cardinality of the independent set I_2 returned by the state-of-the-art approximation algorithm for the MIS problem. We have $\mathrm{OPT}_1 = |L_1| + \mathrm{OPT}_2$ and

$\mathrm{OPT}_2 \le (1.4+\epsilon)\mathrm{SOL}_2$, for any $\epsilon > 0$. Therefore,

$$\mathrm{OPT}_1 \le |L_1| + (1.4+\epsilon)\mathrm{SOL}_2 \le (1.4+\epsilon)(|L_1| + \mathrm{SOL}_2) = (1.4+\epsilon)\mathrm{SOL}_1.$$

This proves the lemma. □

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 on degree*-4 *graphs.*

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 on degree-4 graphs. The theorem is thus proved. □

4 The APX-hardness reduction from 3- MIS to 2- Max- Duo

In the above approximation algorithm Approx for 2- Max- Duo, we apply a vertex-degree reduction scheme on the constructed instance graph of the MIS problem, to remove all the degree-6 vertices and all the degree-5 vertices. This scheme essentially reduces the 2- Max- Duo problem to computing a maximum independent set in a graph of maximum degree $\Delta \le 4$. One might wonder whether all the degree-4 vertices can be similarly removed.

Goldstein et al. (2004) proved that the 2-MCSP problem is APX-hard via a linear reduction from the MIS problem on cubic graphs (3- MIS); Boria et al. (2014) showed that the same reduction could also be applied to prove that 2-Max- Duo is APX-hard. In this section, we review this APX-hardness reduction from 3-MIS to 2-Max- Duo, to point out that it is unlikely possible to further reduce the maximum degree Δ from 4 to 3 by removing all the degree-4 vertices.

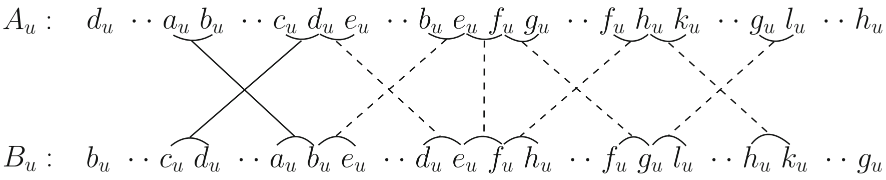

A_u : d_u · · a_u b_u · · c_u d_u e_u · · b_u e_u f_u g_u · · f_u h_u k_u · · g_u l_u · · h_u
B_u : b_u · · c_u d_u · · a_u b_u e_u · · d_u e_u f_u h_u · · f_u g_u l_u · · h_u k_u · · g_u

Fig. 6 The instance $I_u = (A_u, B_u)$ defined for each vertex $u \in V'$. The two dots between a pair of two consecutive main substrings represent $x_u^i y_u^i$ in A_u and $y_u^i x_u^i$ in B_u, respectively, for $i = 1, 2, \ldots, 6$. Each solid or dashed line connects a pair of common duos between A_u and B_u. The set of five duos connecting by solid lines is a unique optimal solution to I_u

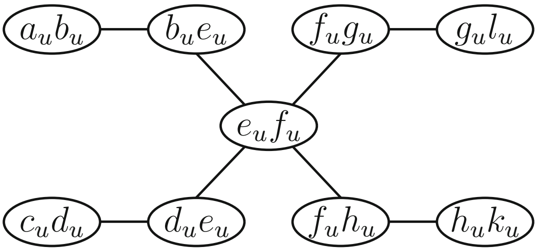
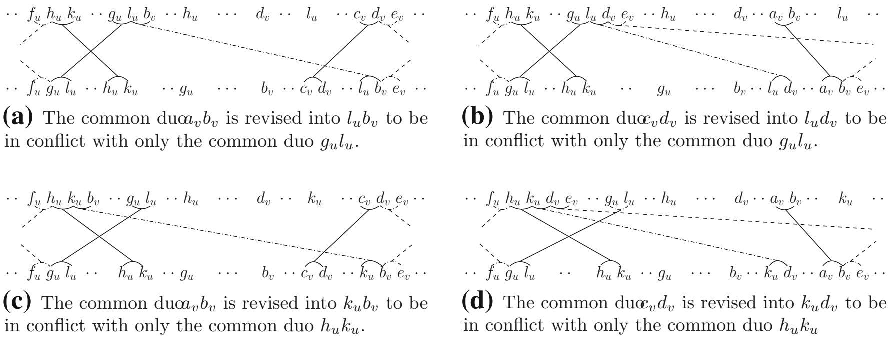


Fig. 7 The gadget subgraph associated with the instance $I_u = (A_u, B_u)$, in which there are nine vertices corresponding to the nine common duos between A_u and B_u

Given a cubic graph $G' = (V', E')$, an instance of 2-Max- Duo can be constructed in the following three steps.

1. For each vertex $u \in V'$, define a small 2-Max- Duo instance $I_u = (A_u, B_u)$ as shown in Fig. 6, where both A_u and B_u are length-28 strings with seven main substrings, and each pair of two consecutive main substrings are separated by a substring of two letters $x_u^i y_u^i$ in A_u and by $y_u^i x_u^i$ in B_u, respectively, for $i = 1, 2, \ldots, 6$. These 12 letters x_u^i's and y_u^i's are distinct, each appears only once in A_u and is represented by a dot in Fig. 6.
   One can easily check that there are nine common duos between A_u and B_u, and the set of five duos connected by solid lines in Fig. 6 is the unique optimal solution to the instance I_u. Equivalently, this constructs a gadget subgraph of the MIS problem, as shown in Fig. 6a, in which there are nine vertices one-to-one corresponding to the nine common duos and two vertices are adjacent if and only if they are conflicting. The vertex subset $\{a_u b_u, c_u d_u, e_u f_u, g_u l_u, h_u k_u\}$ is the unique maximum independent set in this subgraph.
2. Orient each edge in E' such that every vertex of V' has at most two incoming edges and at most two outgoing edges. This can be done by partitioning G' into a set of edge-disjoint cycles and a forest, followed by orienting the edges of a cycle to form a directed cycle, and rooting a tree at a leaf and then orienting the edges away from the root.
3. Let $A = \bigcup_{u \in V'} A_u$, $B = \bigcup_{u \in V'} B_u$, and the whole instance $I = (A, B)$. For each directed edge $(u, v) \in E'$, modify the instances I_u and I_v such that an optimal solution to I coincides with at most one of the optimal solutions to I_u and I_v. To this purpose, either the common duo $a_v b_v$ is revised into $l_u b_v$ ($k_u b_v$, respectively) to be in conflict with only the common duo $g_u l_u$ ($h_u k_u$, respectively), or the common duo $c_v d_v$ is revised into $l_u d_v$ ($k_u d_v$, respectively) to be in conflict with only the common


$\cdot\cdot\ f_u\ h_u\ k_u\ \cdot\cdot\ g_u\ l_u\ b_v\ \cdot\cdot\ h_u\ \cdots\ d_v\ \cdot\cdot\ l_u\ \cdot\cdot\ c_v\ d_v\ e_v\ \cdot\cdot$
$\cdot\cdot\ f_u\ g_u\ l_u\ \cdot\cdot\ h_u\ k_u\ \cdot\cdot\ g_u\ \cdots\ b_v\ \cdot\cdot\ c_v\ d_v\ \cdot\cdot\ l_u\ b_v\ e_v\ \cdot\cdot$
$\cdot\cdot\ f_u\ h_u\ k_u\ \cdot\cdot\ g_u\ l_u\ d_v\ e_v\ \cdot\cdot\ h_u\ \cdots\ d_v\ \cdot\cdot\ a_v\ b_v\ \cdot\cdot\ l_u\ \cdot\cdot$
$\cdot\cdot\ f_u\ g_u\ l_u\ \cdot\cdot\ h_u\ k_u\ \cdot\cdot\ g_u\ \cdots\ b_v\ \cdot\cdot\ l_u\ d_v\ \cdot\cdot\ a_v\ b_v\ e_v\ \cdot\cdot$
$\cdot\cdot\ f_u\ h_u\ k_u\ b_v\ \cdot\cdot\ g_u\ l_u\ \cdot\cdot\ h_u\ \cdots\ d_v\ \cdot\cdot\ k_u\ \cdot\cdot\ c_v\ d_v\ e_v\ \cdot\cdot$
$\cdot\cdot\ f_u\ g_u\ l_u\ \cdot\cdot\ h_u\ k_u\ \cdot\cdot\ g_u\ \cdots\ b_v\ \cdot\cdot\ c_v\ d_v\ \cdot\cdot\ k_u\ b_v\ e_v\ \cdot\cdot$
$\cdot\cdot\ f_u\ h_u\ k_u\ d_v\ e_v\ \cdot\cdot\ g_u\ l_u\ \cdot\cdot\ h_u\ \cdots\ d_v\ \cdot\cdot\ a_v\ b_v\ \cdot\cdot\ k_u\ \cdot\cdot$
$\cdot\cdot\ f_u\ g_u\ l_u\ \cdot\cdot\ h_u\ k_u\ \cdot\cdot\ g_u\ \cdots\ b_v\ \cdot\cdot\ k_u\ d_v\ \cdot\cdot\ a_v\ b_v\ e_v\ \cdot\cdot$

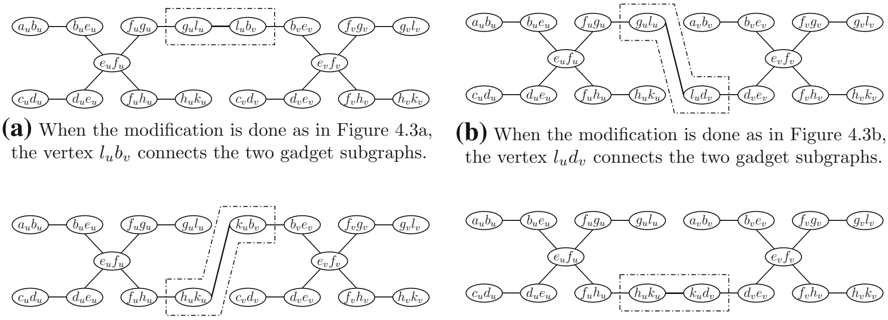
(a) The common duoa_vb_v is revised into l_ub_v to be in conflict with only the common duo g_ul_u.

(b) The common duoc_vd_v is revised into l_ud_v to be in conflict with only the common duo g_ul_u.

(c) The common duoa_vb_v is revised into k_ub_v to be in conflict with only the common duo h_uk_u.

(d) The common duoc_vd_v is revised into k_ud_v to be in conflict with only the common duo h_uk_u

Fig. 8 Four options of modifying the instances I_u and I_v by only modifying the right side of A_u and the left sides of A_v, B_v such that an optimal solution to I coincides with at most one of the optimal solutions to I_u and I_v



(a) When the modification is done as in Figure 4.3a, the vertex l_ub_v connects the two gadget subgraphs.

(b) When the modification is done as in Figure 4.3b, the vertex l_ud_v connects the two gadget subgraphs.

(c) When the modification is done as in Figure 4.3c, the vertex k_ub_v connects the two gadget subgraphs.

(d) When the modification is done as in Figure 4.3d, the vertex k_ud_v connects the two gadget subgraphs.

Fig. 9 Four different configurations for joining the two gadget subgraphs for the vertices $u, v \in V'$, in each of which a common duo is revised for the directed edge $(u, v) \in E'$

duo g_ul_u (h_uk_u, respectively). These four options of modification (Goldstein et al. 2004; Boria et al. 2014) are shown in Fig. 8. Since every vertex of V' has at most two incoming edges and at most two outgoing edges, the revision process for the directed edge $(u, v) \in E'$ can be independently done with respect to all the other edges of E'.

One can check (or refer to the detailed proofs in Goldstein et al. (2004); Boria et al. (2014)) that there exists an independent set of size α in G' if and only if $4n + \alpha$ duos can be preserved in $I_{G'}$, where $n = |V'|$.

The above common duo modification process for each directed edge $(u, v) \in E'$ is equivalent to joining the two gadget subgraphs for the vertices $u, v \in V'$ by connecting one of g_ul_u and h_uk_u to one of a_vb_v and c_vd_v, but additionally revising the letter content of the common duo of I_v. Corresponding to the four options of modification shown in Fig. 8, the two gadget subgraphs are joined as shown in Fig. 9, respectively.

Since each directed edge $(u, v) \in E'$ gives rise to exactly one of the four possible configurations shown in Fig. 9, we conclude from G' being cubic that exactly three of the four degree-1 vertices $\{a_ub_u, c_ud_u, g_ul_u, h_uk_u\}$ in the gadget subgraph for the vertex $u \in V'$ increase their degree to 2. It follows that however the edge orientation scheme is, all the vertices in the final graph G have degrees 1, 2, or 4. Therefore, it is impossible to determine in polynomial time which subset of all the degree-4 vertices is in the maximum independent set of G.

5 Conclusion

In this paper, we examined the 2- Max- Duo problem to design an improved approximation algorithm. Based on an existing linear reduction to the MIS problem (Goldstein et al. 2004; Boria et al. 2014), we presented a vertex-degree reduction scheme to reduce the maximum degree of the constructed instance graph from 6 to 4. Along the way, we uncovered several interesting structural properties of the constructed instance graph. Our main contribution is a $(1.4 + \epsilon)$-approximation algorithm for 2-Max- Duo, for any $\epsilon > 0$.

It is worth mentioning that our vertex-degree reduction technique can also be applied for k-Max- Duo when $k \geq 3$. For example, 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 algorithm. Nevertheless, the $(2.6 + \epsilon)$-approximation algorithm is superseded by the $(2+\epsilon)$-approximation algorithm for the general Max-Duo (Dudek et al. 2017).

For 2- Max- Duo, it would be interesting to investigate whether the maximum degree can be further reduced to 3, but not by determining in polynomial time which subset of all the degree-4 vertices is in the maximum independent set. On the other hand, one could examine whether certain structural properties of the 2- Max- Duo instance support a direct better-than-1.4 approximation algorithm, that is, not by calling the existing $(1.4 + \epsilon)$-approximation algorithm for the MIS problem, or not even by reducing to the MIS problem.

Acknowledgements This research is supported by the NSERC Canada. Additionally, Chen is supported by the National Natural Science Foundation of China Grant No.'s 11971139, 11771114 and 11571252, and the China Scholarship Council Grant No. 201508330054; Liu is supported by the National Natural Science Foundation of China Grant No. 61370052; Luo is supported by the National Natural Science Foundation of China Grant No. 71701162, the Humanities and Social Science Foundation of Ministry of Education of China Grant No. 18YJC630114, and the Youth Innovation Team of Shaanxi Universities; Su is supported partially by the Humanities and Social Science Foundation of Ministry of Education of China Grant No. 18YJAZH080 and Science and Technology Department of Shaanxi Province Grant No. 2020JQ-654; Zhang is supported by the National Natural Science Foundation of China Grant No.'s 61972228 and 61672323.

References

Beretta S, Castelli M, Dondi R (2016a) Corrigendum to "Parameterized tractability of the maximum-duo preservation string mapping problem" [646(2016), 16–25]. Theoret Comput Sci 653:108–110

Beretta S, Castelli M, Dondi R (2016b) Parameterized tractability of the maximum-duo preservation string mapping problem. Theoret Comput Sci 646:16–25

Berman P, Fujito T (1999) On approximation properties of the independent set problem for low degree graphs. Theory Comput Syst 32:115–132

Berman P, Karpinski M (1999) On some tighter inapproximability results. In: Proceedings of the of 26th international colloquium on automata, languages and programming (ICALP'99), pp 200–209

Boria N, Kurpisz A, Leppänen S, Mastrolilli M (2014) 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, pp 14–25

Boria N, Cabodi G, Camurati P, Palena M, Pasini P, Quer S (2016) 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, pp 11:1–11:8

Brubach B (2016) 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, pp 52–64

Bulteau L, Fertin G, Komusiewicz C, Rusu I (2013) 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, pp 244–258

Bulteau L, Komusiewicz C (2014) 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), pp 102–121

Chen X, Zheng J, Fu Z, Nan P, Zhong Y, Lonardi S, Jiang T (2005) Assignment of orthologous genes via genome rearrangement. IEEE/ACM Trans Comput Biol Bioinf 2:302–315

Chen W, Chen Z, Samatova NF, Peng L, Wang J, Tang M (2014) Solving the maximum duo-preservation string mapping problem with linear programming. Theoret Comput Sci 530:1–11

Chrobak M, Kolman P, Sgall J (2004) 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, pp 84–95

Cormode G, Muthukrishnan S (2007) The string edit distance matching problem with moves. ACM Trans Algorithms 3:2:1–2:19

Damaschke P (2008) Minimum common string partition parameterized. In: Proceedings of the 8th international workshop on algorithms in bioinformatics (WABI 2008), volume 5251 of LNBI, pp 87–98

Dudek B, Gawrychowski P, Ostropolski-Nalewaja P (2017) A family of approximation algorithms for the maximum duo-preservation string mapping problem. In: Proceedings of the 28th annual symposium on combinatorial pattern matching (CPM 2017), volume 78 of LIPIcs, pp 10:1–10:14. arXiv:1702.02405

Garey MR, Johnson DS (1979) Computers and intractability: a guide to the theory of NP-completeness. W. H. Freeman and Company, San Francisco

Goldstein A, Kolman P, Zheng J (2004) 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, pp 484–495

Jiang H, Zhu B, Zhu D, Zhu H (2012) Minimum common string partition revisited. J Comb Optim 23:519–527

Kolman P, Waleń T (2007) Approximating reversal distance for strings with bounded number of duplicates. Discrete Appl Math 155:327–336

Kolman P, Waleń T (2006) 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, pp 279–289

Swenson KM, Marron M, Earnest-DeYoung JV, Moret BM (2008) Approximating the true evolutionary distance between two genomes. J Exp Algorithmics 12:3–5

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

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Affiliations

Yong Chen[1] · Guohui Lin[2] · Tian Liu[3] · Taibo Luo[4] · Bing Su[5] · Yao Xu[6] · Peng Zhang[7]

✉ Guohui Lin
guohui@ualberta.ca

Yong Chen
chenyong@hdu.edu.cn

Tian Liu
lt@pku.edu.cn

Taibo Luo
tbluo@xidian.edu.cn

Bing Su
subing684@sohu.com

Yao Xu
yxu@kettering.edu

Peng Zhang
algzhang@sdu.edu.cn

[1] Department of Mathematics, Hangzhou Dianzi University, Hangzhou, Zhejiang, China

[2] Department of Computing Science, University of Alberta, Edmonton AB T6G 2E8, Canada

[3] Key Laboratory of High Confidence Software Technologies (MOE), Department of Computer Science and Technology, Peking University, Beijing, China

[4] School of Economics and Management, Xidian University, Xi'an, Shaanxi, China

[5] School of Economics and Management, Xi'an Technological University, Xi'an, Shaanxi, China

[6] Department of Computer Science, Kettering University, Flint, MI, USA

[7] School of Computer Science and Technology, Shandong University, Jinan, Shandong, China