Restricted t-Matchings via Half-Edges

Katarzyna Paluch
Institute of Computer Science, University of Wrocław, Poland

Mateusz Wasylkiewicz
Institute of Computer Science, University of Wrocław, Poland

Abstract

For a bipartite graph G we consider the problem of finding a maximum size/weight square-free 2-matching and its generalization - the problem of finding a maximum size/weight $K_{t,t}$-free t-matching, where t is an integer greater than two and $K_{t,t}$ denotes a bipartite clique with t vertices on each of the two sides. Since the weighted versions of these problems are $\mathcal{NP}$-hard in general, we assume that the weights are vertex-induced on any subgraph isomorphic to $K_{t,t}$. We present simple combinatorial algorithms for these problems. Our algorithms are significantly simpler and faster than those previously known. We dispense with the need to shrink squares and, more generally subgraphs isomorphic to $K_{t,t}$, the operation which occurred in all previous algorithms for such t-matchings and instead use so-called *half-edges*. A *half-edge* of edge e is, informally speaking, a half of e containing exactly one of its endpoints.

Additionally, we consider another problem concerning restricted matchings. Given a (not necessarily bipartite) graph $G = (V, E)$, a set of k subsets of edges $E_1, E_2, \ldots, E_k$ and k natural numbers $r_1, r_2, \ldots, r_k$, the Restricted Matching Problem asks to find a maximum size matching of G among such ones that for each $1 \leq i \leq k$, M contains at most r_i edges of E_i. This problem is $\mathcal{NP}$-hard even when G is bipartite. We show that it is solvable in polynomial time if (i) for each i the graph G contains a clique or a bipartite clique on all endpoints of E_i; in the case of a bipartite clique it is required to contain E_i and (ii) the sets $E_1, \ldots, E_k$ are almost vertex-disjoint - the endpoints of any two different sets have at most one vertex in common.

2012 ACM Subject Classification Theory of computation → Graph algorithms analysis

Keywords and phrases restricted 2-matchings

Digital Object Identifier 10.4230/LIPIcs.ESA.2021.73

Funding Partially supported by Polish National Science Center grant 2018/29/B/ST6/02633.

Acknowledgements The authors thank Pratik Ghosal and Prajakta Nimbhorkar for discussions at the early stage of research on this paper.

1 Introduction

Given a positive integer t, a subset M of edges of an undirected simple graph G is called a *t-matching* if every vertex is incident to at most t edges of M. A t-matching of maximum size can be found in polynomial time by a reduction to the classical matching problem. A 2-matching is called *square-free* if it does not contain any cycle of length 4. A *C_k-free* 2-matching is one without any cycle of length at most k. The *C_k-free 2-matching problem* consists in finding a C_k-free 2-matching of maximum size. Observe that the C_k-free 2-matching problem for $n/2 \leq k < n$, where n is the number of vertices in the graph, is equivalent to finding a Hamiltonian cycle, and thus $\mathcal{NP}$-hard. Hartvigsen [13] gave a complicated algorithm for the case of $k = 3$. Papadimitriou [4] showed that this problem is $\mathcal{NP}$-hard when $k \geq 5$. The complexity of the C_4-free 2-matching problem is unknown.

When the graph is bipartite the smallest length of a cycle contained in it is at least 4. We refer to cycles of length four as *squares*. Polynomial time algorithms for the C_4-free 2-matching problem in bipartite graphs were shown by Hartvigsen [14], Pap [30], Babenko [1] and analyzed by Király [18]. A generalization of a square-free 2-matching in a bipartite graph is a $K_{t,t}$*-free* t-matching - a t-matching without any $K_{t,t}$ - a bipartite clique with t vertices on each of the two sides. $K_{t,t}$-free t-matchings were first considered by Frank [7], who provided a min-max formula for $K_{t,t}$-free t-matchings based on a result in [8] on crossing bi-supermodular functions. Using this formula, it is possible to compute the size of a maximum $K_{t,t}$-free t-matching by the ellipsoid method or a combinatorial method by Fleiner [6]. Moreover, one can compute a maximum $K_{t,t}$-free t-matching through Végh and Benczúr's algorithm [36] for covering pairs of sets and directly using Pap's algorithm [29].

In the weighted version of the $K_{t,t}$-free t-matching problem, each edge e is associated with a nonnegative weight $w(e)$ and we are interested in finding a $K_{t,t}$-free t-matching of maximum weight, where the weight of a t-matching M is defined as the sum of weights of edges belonging to M. The weighted square-free 2-matching problem in bipartite graphs was proven to be $\mathcal{NP}$-hard [12, 19] even if the weight of every edge is either 0 or 1. Bérczi and Kobayashi [2] sharpened the result and showed that the problem is $\mathcal{NP}$-hard even if the given graph is cubic, bipartite and planar. The weighted $K_{t,t}$-free t-matching problem in bipartite graphs is solvable in polynomial time when the weights of edges are vertex-induced on every subgraph isomorphic to $K_{t,t}$, which was shown by Makai [24] and Takazawa [33].

Apart from $K_{t,t}$-free t-matchings, we consider another problem concerning restricted matchings. Given a (not necessarily bipartite) graph $G = (V, E)$, a set of k subsets of edges $E_1, E_2, \ldots, E_k$ and k natural numbers $r_1, r_2, \ldots, r_k$, the Restricted Matching Problem asks to find a maximum size classical matching of G among such ones that for each $1 \leq i \leq k$, M contains at most r_i edges of E_i. This problem was first studied in [17] by Itai, Rodeh and Tanimoto for bipartite graphs and shown to be $\mathcal{NP}$-hard for the general case and solvable in polynomial time for the variant when there is only one set E_1, i.e., when $k = 1$. The version, in which G is bipartite and each E_i contains two edges (and hence each $r_i = 1$) was proven to be $\mathcal{NP}$-hard by Garey and Johnson [11]. The problem was also considered in [25] and [31].

Our results We present simple combinatorial algorithms for the weighted and unweighted version of the $K_{t,t}$-free t-matching problem in bipartite graphs. In the weighted version we assume that the weights of edges are vertex-induced on every subgraph isomorphic to $K_{t,t}$. In these algorithms we successively find and apply a minimum length/weight augmenting path until it is no longer possible. The search for an augmenting path is conducted in a specially modified graph G, called G'. Graph G' is obtained from G by replacing some subgraphs with so-called gadgets that contain *half-edges.* A *half-edge* of edge e is, informally speaking, a half of e containing exactly one of its endpoints. Half-edges have been first introduced in [27]. Each subgraph that is replaced with a gadget in a given step is isomorphic to $K_{t,t}$ and $t^2 - 1$ of its edges belong to the current t-matching M. In previous algorithms for these problems such or similar subgraphs were *shrunk.* One could say that we take an opposite approach and *expand* such subgraphs. However, in our case these expansions do not build on each other and in each step G' is constructed only from the original graph G and a current t-matching M. We give a detailed description of these algorithms for square-free 2-matchings and their analyses and only an outline for the $K_{t,t}$-free t-matching problem. In addition to being significantly simpler our algorithms are also faster than those known previously. For the unweighted square-free 2-matching problem our algorithm has running time $\mathcal{O}(nm)$, where n denotes the number of vertices in the graph and m the number of edges. Both algorithms by Hartvigsen and Babenko run in $\mathcal{O}(n^3)$ time and the one by Pap in $\mathcal{O}(n^4)$. For the weighted/unweighted version of the $K_{t,t}$-free t-matching problem we give an algorithm with running time, respectively, $\mathcal{O}(tnm + t^2n^2 \log n)$ and $\mathcal{O}(nm + tn^2 + \sqrt{tn}m)$. For the weighted variant the algorithm by Takazawa has runtime $\mathcal{O}(tn^2m + tn^3 \log n)$ for the t-factor problem (each vertex has to be incident with exactly t edges) and $\mathcal{O}(t^5n^6 + t^4n^6 \log n)$ for the t-matching problem because of the costly reduction to the t-factor problem. The algorithm by Makai has polynomial time but it uses the ellipsoid method.

Regarding classical matchings we devise a polynomial time algorithm for the variant of the Restricted Matching Problem when (i) for each i the graph G contains a clique or a bipartite clique on all endpoints of E_i; in the case of a bipartite clique it is required to contain E_i and (ii) the sets $E_1, \ldots, E_k$ are almost vertex-disjoint - the endpoints of any two different sets have at most one vertex in common.

Motivation C_k-free 2-matchings and $K_{t,t}$-free t-matchings are classical problems of combinatorial optimization. They have applications in traveling salesman problems, problems related to finding a smallest 2-edge-connected spanning subgraph as well as in increasing the vertex-connectivity (see [5, 2, 3, 34] for more details). A good survey of these applications has been given by Takazawa [35].

Related work Some generalizations of the C_k-free 2-matching problem were investigated. Recently, Kobayashi [21] gave a polynomial algorithm for finding a maximum weight 2-matching that does not contain any triangle from a given set of forbidden edge-disjoint triangles. Polynomial algorithms for square-free and/or triangle-free 2-matchings in subcubic graphs were presented in [15, 16, 2, 3, 20, 28, 22]. An algorithm by Paluch and Wasylkiewicz uses a similar approach as the one presented in this paper but requires only one computation of a b-matching. When it comes to the square-free 2-matching problem in general graphs, Nam [26] constructed a complex algorithm for it for graphs, in which all squares are vertex-disjoint.

2 Preliminaries

Let $G = (V, E)$ be an undirected graph with vertex set V and edge set E. We denote the number of vertices of G by n and the number of edges of G by m. We denote a vertex set of G by $V(G)$ and an edge set by $E(G)$. We assume that all graphs are ***simple***, i.e., they contain neither loops nor parallel edges. We denote an edge connecting vertices v and u by (v, u). A ***path*** of graph G is a sequence $P = (v_0, \ldots, v_l)$ for some $l \geq 1$ such that $(v_i, v_{i+1}) \in E$ for every $i \in \{0, 1, \ldots, l-1\}$. We refer to l as the ***length*** of P. A ***cycle*** of graph G is a sequence $c = (v_0, \ldots, v_{l-1})$ for some $l \geq 3$ of pairwise distinct vertices of G such that $(v_i, v_{(i+1) \bmod l}) \in E$ for every $i \in \{0, 1, \ldots, l-1\}$. We refer to l as the ***length*** of c. We will sometimes treat a path or a cycle as an edge set and sometimes as a sequence of edges. For an edge set $F \subseteq E$ and $v \in V$, we denote by $\deg_F(v)$ the number of edges of F incident to v. For any two edge sets $F_1, F_2 \subseteq E$, the symmetric difference $F_1 \oplus F_2$ denotes $(F_1 \setminus F_2) \cup (F_2 \setminus F_1)$.

For a natural number t, we say that an edge set $F \subseteq E$ is a ***t-matching*** if $\deg_F(v) \leq t$ for every $v \in V$. t-matchings belong to a wider class of b-matchings, where for every vertex v of G, we are given a natural number $b(v)$ and a subset of edges is a ***b-matching*** if every vertex v is incident to at most $b(v)$ of its edges. A b-matching of G of maximum weight can be computed in polynomial time. We refer to Lovász and Plummer [23] for further background on b-matchings.

Let M be a b-matching. We say that an edge e is ***matched*** (in M) if $e \in M$ and ***unmatched*** (in M) otherwise. Additionally, an edge belonging to M will be referred to as a ***M-edge*** and an edge not belonging to M as a ***non-M-edge***. We call a vertex v ***deficient (in M)*** if $\deg_M(v) < b(v)$. ***An M-alternating path*** P is any sequence of vertices $(v_1, v_2, \ldots, v_k)$ such that edges on P are alternately M-edges and non-M-edges and no edge occurs on P

more than once and $v_1 \neq v_k$. ***An M-alternating cycle*** C has the same definition as an M-alternating path except that $v_1 = v_k$ and additionally $(v_{k-1}, v_k) \in M$ iff $(v_1, v_2) \notin M$. Note that an M-alternating path or cycle may go through some vertices more than once but via different edges. An M-alternating path is called ***M-augmenting*** if it begins and ends with a non-M-edge and if it begins and ends with a deficient vertex. We say that M is a ***maximum*** b-matching if there is no b-matching of G with more edges than M. Given a weight function $w : E \to \mathbb{R}$ we define ***weight*** of M as $w(M) = \sum_{e \in M} w(e)$. We say that M has ***maximum weight*** if there is no b-matching of G of weight greater than $w(M)$.

Given a weight function w, the ***alternating weight*** of an M-alternating path or cycle P is defined as $\tilde{w}(P) = \sum_{e \in P \cap M} w(e) - \sum_{e \in P \setminus M} w(e)$. We say that an M-augmenting path P is ***minimum*** if it has minimum alternating weight among all M-augmenting paths and cannot be shortened without increasing its alternating weight. An M-alternating cycle is a ***negative cycle*** if its alternating weight is negative. An ***application*** of an M-alternating path or cycle P to M is an operation whose result is $M \oplus P$. Note that $w(M \oplus P) = w(M) - \tilde{w}(P)$.

We are interested in computing a b-matching of a graph G where we are given vectors $l, u \in \mathbb{N}^V$ and a weight function $w : E \to \mathbb{R}$. For a vertex $v \in V$, $[l(v), u(v)]$ is said to be a ***capacity interval*** of v. An edge set $M \subseteq E$ is said to be an (l, u)-matching if $l(v) \leq \deg_M(v) \leq u(v)$ for every $v \in V$. A maximum weight (l, u)-matching can be computed efficiently.

For a weight function $w : E \to \mathbb{R}$ and a subgraph H of G, we say that w is ***vertex-induced on*** H if there exists a function $r : V(H) \to \mathbb{R}$ such that $w(u, v) = r(u) + r(v)$ for every edge (u, v) of H. We call r ***a potential function*** of H.

An instance of the square-free 2-matching problem consists of an undirected bipartite graph $G = (V, E)$ and the goal is to find a maximum square-free 2-matching of G. A generalization of a square-free 2-matching in a bipartite graph is a $K_{t,t}$-free t-matching - a t-matching without any $K_{t,t}$ - a bipartite clique with t vertices on each of the two sides. An instance of the $K_{t,t}$-free t-matching problem consists of an undirected bipartite graph $G = (V, E)$ and a natural number $t \geq 2$. The aim is to compute a maximum $K_{t,t}$-free t-matching. We also consider weighted versions of these problems, in which we are additionally given a weight function $w : E \to \mathbb{R}_{\geq 0}$ that is vertex-induced on each subgraph of G isomorphic to $K_{t,t}$ and the task consists in finding a maximum weight $K_{t,t}$-free t-matching.

3 Outline of the Algorithm for Square-Free 2-Matchings

The general scheme of the algorithm for each variant of the square-free 2-matching problem is the same - we give it below.

Algorithm 1 Computing a maximum (weight) square-free 2-matching of a bipartite graph G.

1: Let M be an empty 2-matching of G.
2: Construct an auxiliary bipartite graph $G' = (V', E')$ with $\mathcal{O}(n)$ vertices and $\mathcal{O}(m)$ edges, and its 2-matching M' of size $\mathcal{O}(n)$ by replacing some squares of G with gadgets containing half-edges. (Both gadgets and half-edges are defined later.)
3: Compute a shortest (resp. minimum) M'-augmenting path P of G'. If G' contains no M'-augmenting path (resp. no M'-augmenting path with negative alternating weight), stop the algorithm and return M.
4: Apply P to M' obtaining M'' and extract a square-free 2-matching M_1 of G from M'' such that $|M_1| = |M| + 1$ (resp. $w(M_1) > w(M)$). Set M as M_1 and go to 2.

▷ Claim 3.1. Algorithm 1 runs in time $O(nm + n^2 \log n)$.

Proof. It will be easy to implement all steps of an Algorithm 1 except 3 in linear time. 3 can be implemented to run in time $O(|E'|+|V'|\log|V'|)$ by Fredman and Tarjan's implementation of Dijkstra's algorithm [9] similarly as in the Hungarian method described by Schrijver [32]. Every step is executed $O(n)$ times since $|M|$ increases by one every time 4 is executed. ◁

Let us also remark that in the unweighted version of the problem Algorithm 1 runs in $O(nm)$ since 3 can be implemented to run in linear time.

4 Maximum square-free 2-matchings in bipartite graphs

In this section we show how to find a maximum square-free 2-matching in a bipartite graph G. When computing a maximum 2-matching N of G, which is not required to be square-free, we can proceed as follows. As long as G contains some N-augmenting path P, apply it to N and repeat. When the goal is to compute a maximum square-free 2-matching of G, this approach is not applicable for two reasons. Firstly, by applying an augmenting path to a square-free 2-matching we may obtain a 2-matching which is not square-free. Secondly, it may happen that a square-free 2-matching M is not maximum but G does not contain any M-augmenting path P such that $M \oplus P$ is square-free. An example of such a 2-matching is shown in Figure 1. Nevertheless, it turns out as we demonstrate below, that we may still use this method if the search for an augmenting path is conducted in an appropriately modified graph G, called G'.

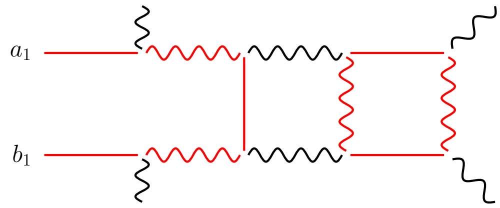

Figure 1 Edges of a square-free 2-matching M are waved. M is not maximum, G contains an M-augmenting path P (with endpoints a_1, b_1) but $M \oplus P$ is not square-free. If we apply both P and an M-alternating cycle C (indicated by red edges), we obtain a larger square-free 2-matching.

First let us check what types of squares of G are in danger of appearing in a 2-matching after the application of a shortest augmenting path. (In fact P does not have to be shortest - it suffices if P has no shortcuts.)

▶ **Fact 4.1.** *Any shortest M-augmenting path P' has the property that for any vertex v, it contains at most two edges incident to v: at most one matched edge and at most one unmatched one.*

Proof. Otherwise P' would contain an alternating cycle C and hence could be shortened. (This property does not hold in non-bipartite graphs.) ◀

▶ **Lemma 4.2** (Proposition 2.1. in [33]). *Let M be a square-free 2-matching of G and P any shortest M-augmenting path. If $M \oplus P$ contains a square s, then exactly three edges of s belong to M.*

Proof. Suppose that $M \oplus P$ contains a square $s = (a, b, c, d)$ such that at least two of its edges do not belong to M, one of which is (a, b). Therefore $(a, b) \in P$ and by the fact above we get that both (b, c) and (a, d) belong to M. Hence, $(c, d) \notin M$. Thus, the edges $(a, b), (c, d)$ belong to P and are connected in P by an odd-length alternating M-path R. R's endpoints are either a and d or b and c. Also, R begins and ends with an M-edge. This means that P can be shortened by replacing R either with (a, d) or (b, c) - a contradiction. ◀

Suppose that M is a (possibly empty) square-free 2-matching of G. We say that a square s of G is ***saturated*** (in M) if exactly three edges of s are contained in M. The graph G' in which we are going to search for an augmenting path is obtained from the original graph G by replacing a subset of saturated squares with specially constructed subgraphs called *gadgets*. More details are given below.

One can observe that, since G is bipartite, any edge e of G belongs to at most two different saturated squares.

▶ **Definition 4.3.** *A saturated square s, which has exactly one common edge with some other saturated square is said to be* ***unproblematic****. Otherwise, s is said to be* ***problematic****.*

Unproblematic squares can be easily got rid of by replacing some edges with other ones as explained in more detail in the proof of Lemma 4.8.

▶ **Fact 4.4.** *If two problematic squares have a common edge, then they share exactly two edges, both of which are in M. Any problematic square is non-edge-disjoint with at most one other problematic square.*

Proof. Let s_1 and s_2 be two problematic squares of G with a common edge. Since they are problematic, they have at least two common edges. Note that s_1 and s_2 cannot share more than two edges because otherwise they would share all four vertices. However, two different squares cannot share all vertices because G is simple and bipartite. Let e_1, e_2 be the common edges of s_1 and s_2. They cannot be vertex-disjoint because then again s_1 and s_2 would have four common vertices. Hence, e_1 and e_2 have a common vertex v.

Next we argue that both e_1 and e_2 must belong to M. Suppose to the contrary that $e_1 \notin M$. This implies that the endpoint $v'$of e_2 different from v is incident to three egdes of M - a contradiction.

To see that s_1 cannot share an edge with a problematic square $s_3 \neq s_2$, observe that in such a case s_3 would have to share with s_1 exactly one of the edges of $\{e_1, e_2\}$ and additionally some edge e_3. The edge e_3 cannot belong to s_2 because it is incident to a vertex of s_1 not contained in s_2. Hence, s_3 and s_2 share exactly one edge, which implies that s_2 is not problematic - a contradiction. ◀

If G contains at least one problematic square, we build a graph $G' = (V', E')$ together with its 2-matching M', in which each problematic square s is replaced with a subgraph, called a ***gadget for*** s. The precise construction of G' and M' is the following. We start off with $G' = G$. We initialize M' as the set of edges of M which are not part of any problematic square.

Let $s = (a, b, c, d)$ be any problematic square of G such that $(a, b) \notin M$. For each edge (p, q) of s we add two new vertices v_q^p and v_p^q, called ***subdivision vertices*** (of s), and we replace (p, q) with three new edges: $(p, v_q^p), (v_q^p, v_p^q), (v_p^q, q)$. Each of the edges $(p, v_q^p), (v_p^q, q)$ is called a ***half-edge*** (of (p, q) and also of s). The edge (v_q^p, v_p^q) is called an ***eliminator*** (of (p, q)). We remove the eliminator of the edge (a, b) from E'. Additionally, we introduce two new vertices u_s^1 and u_s^2, called ***global vertices***. We connect u_s^1 with every subdivision vertex connected to a or c. Similarly, we connect u_s^2 with every subdivision vertex connected to b or d. We define vectors $l, u \in \mathbb{N}^{V'}$ as follows. We set a capacity interval of every vertex of the original graph G to $[0, 2]$ and we set a capacity interval of every other vertex of G' to $[1, 1]$, i.e., every vertex of $V' \setminus V$ is matched to exactly one vertex of G' in any (l, u)-matching of G'. For every edge $e \in M$ of s, we add half-edges of e to M'. Additionally, we add (u_s^1, v_b^a) and (u_s^2, v_a^b) to M'. If two problematic squares share two edges, then their gadgets overlap, i.e., we build a gadget for each one of them in the way described above.

The main ideas behind the gadget for a problematic square $s = (a, b, c, d)$ are the following. An (l, u)-matching M' of G' is to represent roughly a square-free 2-matching M of G. If M' contains both half-edges of some edge e, then e is included in M. If M' contains an eliminator of e, then e does not belong to M (is excluded from M). We want to ensure that at least one edge of s does not belong to M. This is done by requiring that the two global vertices u_s^1 and u_s^2 are matched to two subdivision vertices. In this way two half-edges of s are guaranteed not to belong to M' and hence to M.

Additionally, we can observe that for a 2-matching M depicted in Figure 1, there exists one M'-augmenting path in G' comprising all red edges.

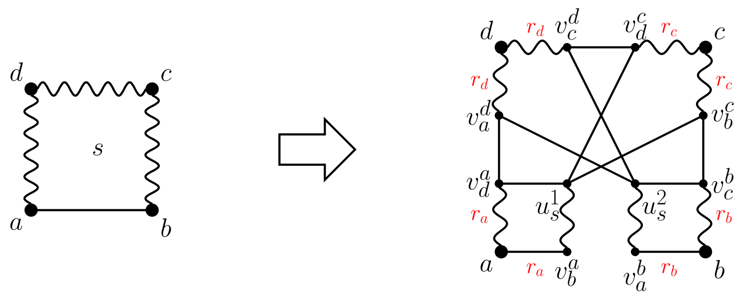

Figure 2 A gadget for a problematic square $s = (a, b, c, d)$ such that $(a, b) \notin M$. Weights of the edges for the weighted version are given in red.

It turns out that if G' contains an M'-augmenting path, then we can apply a shortest one to M', obtaining a larger 2-matching M'' of G'. From M'' we can in turn obtain a square-free 2-matching M_2 of G of size $|M| + 1$. This is achieved by first changing around the half-edges of M'' so that for each edge $e \in E$ belonging to a problematic square we have that either both half-edges of e are contained in M'' or none. Next, if needed, we get rid of unproblematic squares.

▶ **Lemma 4.5.** *Let P be a shortest M'-augmenting path of G' and let $M'' = M' \oplus P$. Then there exists exactly one 2-matching M_1 of G, denoted as $img(M'')$, such that:*

1. *for each vertex v of G it holds that $deg_{M''}(v) = deg_{M_1}(v)$,*
2. *for each edge $e \in E$ not belonging to any problematic square, we have that $e \in M_1 \Leftrightarrow e \in M''$.*

Proof. We obtain M_1 from M'' as follows. First for each edge $e \in E$ not belonging to any problematic square, we include e into M_1 if and only if $e \in M''$. Next we remove all edges of M'' incident to global vertices and flip the half-edges so that for each edge $e \in E$ belonging to a problematic square we have that either both half-edges of e are contained in M'' or none. To see that the half-edges can indeed be changed around in such a way we use the following observation.

▶ **Observation 4.6.** *Let $s = (a, b, c, d)$ be a problematic square such that $(a, b) \notin M$. If P goes through any unmatched half-edge of (a, b), then it does not go through any matched half-edge of s incident to a or b.*

Proof. Assume that P goes through (v_a^b, b). Then P must also go through the global vertex u_s^2, which in turn means that P must also go through d. (Otherwise P would contain an alternating cycle going through vertices u_s^2, v_c^b, b, v_a^b.) Suppose now that P also contains the matched half-edge of s incident to b. Then P must also contain (v_c^b, v_b^c) and (v_b^c, c). (If s shares edges (b, c) and (c, d) with another problematic square s', then from v_c^b P may go to a global vertex of s', but then it also has to go to d and the case is as below.) This way we obtain a contradiction, because we could shorten P: instead of going through d and u_s^2, b, c using seven edges P could use three edges $(d, v_c^d), (v_c^d, v_d^c), (v_d^c, c)$ instead.

If, on the other hand, P also contained the matched half-edge incident to a, then it would have to contain also the eliminator of (a, d) and the second half-edge of (a, d), which would mean that P contains four edges incident to d, using both matched edges incident to it - a contradiction. ◀

This means the following.

1. If P goes through u_s^1 but not u_s^2, then P also goes through c and thus P contains exactly two half-edges of s: one matched half-edge incident to c and one unmatched half-edge incident to a. As a result M'' contains exactly one matched half-edge incident to c and two matched half-edges incident to a and thus the degrees of vertices a, b, c, d with respect to half-edges of s contained in M'' are equal to, respectively, $2, 1, 1, 2$. In this case we set M_1 so that it contains edges $(b, a), (a, d), (d, c)$ but not (b, c). The case when P goes through u_s^2 but not u_s^1 is symmetrical.
2. If P goes through both u_s^1 and u_s^2, then P contains exactly four half-edges of s: one matched half-edge incident to c, one matched half-edge incident to d and both half-edges of (a, b). As a result the degrees of vertices a, b, c, d with respect to half-edges of s contained in M'' are equal to, respectively, $2, 2, 1, 1$. In this case we set M_1 to contain edges $(b, a), (a, d), (b, c)$ but not (d, c).
3. If P goes neither through u_s^1 nor through u_s^2, then P either does not go through any half-edge of s at all or goes through two matched half-edges of exactly one of the edges $(b, c), (c, d), (a, d)$. As a result M'' has the property that for each edge e of s either both half-edges of e are in M'' or none. In this case we include an edge e of s into M only if its both half-edges are present in M''.

This finishes the proof. ◀

We have the analogue of Lemma 4.2.

▶ **Lemma 4.7.** *Let P be any shortest M'-augmenting path in G'. If $M_1 = img(M' \oplus P)$ contains a square s, then s is unproblematic (in M).*

Proof. Let $s = (a, b, c, d)$ be some square of G that is contained in M_1. First, we can notice that if each of the unmatched edges of s is also present in G', then by Lemma 4.2, we know that s can appear in M_1 only if it is saturated and hence only if it is unproblematic (because otherwise s is replaced with a gadget). Second, we observe that s cannot be problematic, because the gadget for s ensures that at least two half-edges of s do not belong to $M' \oplus P$ and hence at least one edge of s does not belong to M_1.

Suppose then now that s is not saturated and at least one of its unmatched edges, say (a, b), is not present in G'. It means that there exists a problematic square s' that contains (a, b). The edge (a, b) appears in M_1 only if at least one of the half-edges of (a, b) is contained in P. Suppose it is (a, v_b^a). It means that P contains also some matched (half-)edge e' incident to a. By Observation 4.6 the edge e' cannot be contained in s'. Neither can it be contained in s because then s could not appear in M_1. This means that the edge (a, d) of s is unmatched. We notice however, that (a, d) cannot belong to M_1 because neither (a, d) can belong to P (as it would mean that P contains four (half-)edges incident to a and thus could be shortened) nor any half-edge of (a, d) can belong to P (if (a, d) belongs to a problematic square s'' then by Observation 4.6, if P contains a half-edge of (a, d), then it does not contain a matched half-edge incident to a). ◀

▶ **Lemma 4.8.** *Let P be any shortest M'-augmenting path in G' and $M_1 = img(M' \oplus P)$. If M_1 is not square-free then it can be transformed into a square-free 2-matching M_2 such that $|M_1| = |M_2|$.*

Proof. We consider every square $s = (a, b, c, d)$ of M_1. By Lemma 4.7, s is unproblematic. Hence it shares exactly one edge with another unproblematic square s'. Assume that (a, b) is unmatched in M. Observe that (c, d) cannot be a common edge of s and s' because any vertex of G can be incident to at most two edges of M. Neither can (a, b) be a common edge of s and s', because then P could be shortened and not go through (a, b) at all. Suppose then that (b, c) is a common edge of s and $s' = (b, c, e, f)$. It means that $(c, e) \notin M$ and $(b, f) \in P \cap M$. Since apart from (a, b) none of the edges of s belongs to P, the edge (e, c) cannot belong to P either. Therefore, s' is an M_1-alternating cycle. We apply s' to M_1. As a result s does not occur in M_1 any more. Also, this operation does not introduce any new square into M_1 because the edges $(f, b), (b, a), (a, d), (d, c), (c, e)$ form a path of length five and are guaranteed to belong to M_1; therefore, none of them can be part of a square. ◀

▶ **Lemma 4.9.** *If there is no M'-augmenting path in G', then M is a maximum square-free 2-matching of G.*

Proof. It is a special case of Lemma 5.9. ◀

5 Maximum weight square-free 2-matchings in bipartite graphs

In this section we extend the results from the previous section to the weighted setting. Recall that this problem is $\mathcal{NP}$-hard for general weights, therefore we assume that the weight funcion w is vertex-induced on every square. Some proofs are omitted due to space constraints.

▶ **Lemma 5.1.** *Let s and s' be two problematic squares that have exactly two common edges. Then w is vertex-induced on $s \cup s'$.*

Proof. Let $s = (a, b, c, d)$ and $s' = (a, b, c, e)$. Let r and r' be potential functions of, respectively, s and s'. Observe that $r(a) + r(b) = w(a, b) = r'(a) + r'(b)$ and $r(b) + r(c) = w(b, c) = r'(b) + r'(c)$. Let $\Delta = r(a) - r'(a)$. We increase both $r'(a)$ and $r'(c)$ by Δ and decrease both $r'(b)$ and $r'(e)$ by Δ. Notice that r' is still a valid potential function of s' after this operation. Additionally, now r and r' agree on the common vertices. Therefore, $r \cup r'$ is a potential function of $s \cup s'$. ◀

To the construction of G' from Section 4 we add a weight function $w' : E(G') \to \mathbb{R}$ defined as follows. The half-edges incident to a, b, c and d get weight $r(a)$, $r(b)$, $r(c)$ and $r(d)$, respectively, where $r(a), \ldots, r(d)$ are potentials of s. All the other edges of the gadget get weight 0.

Define $k : E(G') \to \{0, 1/2, 1\}$ such that

$$k(e) = \begin{cases} 1 & \text{if } e \in E(G), \\ 1/2 & \text{if } e \text{ is a half-edge}, \\ 0 & \text{otherwise}. \end{cases}$$

For $e \in E(G')$ we say that $k(e)$ is the ***size*** of e.

▶ **Definition 5.2.** *Consider any (l,u)-matching N of G'. We define the **size** of N as $k(N) = \sum_{e \in N} k(e)$. We say that N is **extreme** if it has maximum weight among all (l,u)-matchings of size $k(N)$ in G'. A matching M of G is said to be **extreme** if it has maximum weight among all matchingss of size $|M|$ in G'.*

Algorithm 1 for computing a maximum weight square-free 2-matching differs from the variant for computing for computing a maximum (size) 2-matching only in the fact that we compute a minimum M'-augmenting path instead of a shortest M'-augmenting one. Finding a minimum M'-augmenting path requires computing an M'-augmenting path P with minimum alternating weight. To be able to do this, we need to know that there are no negative cycles in G'. We prove the absence of negative cycles in G' as well as the optimality of the 2-matching computed by Algorithm 1 by using linear programming.

The weighted square-free 2-matching problem can be formulated as an integer program as follows. We assign a variable $x(e)$ for each edge $e \in E$. Any such variable can take on only two values: 0 or 1, where setting a variable $x(e)$ to 1 denotes including e in the 2-matching. To ensure that variables $x(e)$ encode a 2-matching we add constraint 2b. Constraint 2c means that for any square s of the graph at most three of its edges can belong to the 2-matching.

Let S denote the set of all squares of G and $x \in \mathbb{R}^{E(G)}$. The weighted square-free 2-matching problem can be formulated as an integer program, whose linear programming relaxation is the following:

$$(P) \quad \text{maximize} \quad \sum_{e \in E(G)} w(e)x(e) \tag{1a}$$

$$\text{subject to} \quad \sum_{e \in \delta(v)} x(e) \leq 2 \quad (\forall v \in V(G)), \tag{1b}$$

$$\sum_{e \in E(s)} x(e) \leq 3 \quad (\forall s \in S), \tag{1c}$$

$$\sum_{e \in E(G)} x(e) = k. \tag{1d}$$

Let x be any feasible solution of (P) and $M = \{e \in E : x(e) = 1\}$. We can check that M is a square-free 2-matching of G. Namely, the first constraint ensures that for any vertex v at most two edges of M are incident to v and the second constraint implies that for any square s of the graph at most three of its edges belong to M. The linear program (P) has been shown to have an integral optimal solution by Makai [24]. For our purposes we need linear programs, which are relaxations of integer programs for, correspondingly, the extreme size k square-free 2-matching problem and the extreme size k (l,u)-matching problem. These linear programs (P_k) and (P'_k) and their duals $(D_k), (D'_k)$ are given below, where $x' \in \mathbb{R}^{E'(G)}$.

$$
\begin{aligned}
(P_k) \quad & \text{maximize} && \sum_{e \in E(G)} w(e)x(e) && \text{(2a)}\\
& \text{subject to} && \sum_{e \in \delta(v)} x(e) \le 2 \quad (\forall v \in V(G)), && \text{(2b)}\\
& && \sum_{e \in E(s)} x(e) \le 3 \quad (\forall s \in S), && \text{(2c)}\\
& && \sum_{e \in E(G)} x(e) = k, && \text{(2d)}\\
& && 0 \le x(e) \le 1 \quad (\forall e \in E(G)). && \text{(2e)}
\end{aligned}
$$

$$
\begin{aligned}
(D_k) \quad & \text{minimize} && 2\sum_{v \in V(G)} p(v) + \sum_{e \in E(G)} q(e) + 3\sum_{s \in S} \alpha(s) + \beta k && \text{(3a)}\\
& \text{subject to} && p(u) + p(v) + q(e) + \sum_{s \in S : e \in E(s)} \alpha(s) + \beta \ge w(e) (\forall e = (u,v) \in E(G)), && \text{(3b)}\\
& && p, q, \alpha \ge 0. && \text{(3c)}
\end{aligned}
$$

$$
\begin{aligned}
(P'_k) \quad & \text{maximize} && \sum_{e \in E(G)} w'(e)x(e) && \text{(4a)}\\
& \text{subject to} && \sum_{e \in \delta(v)} x(e) \le 2 \qquad (\forall v \in V(G)), && \text{(4b)}\\
& && \sum_{e \in \delta(v)} x(e) = 1 \qquad (\forall v \in V(G') \setminus V(G)), && \text{(4c)}\\
& && \sum_{e \in E(G')} k(e)x(e) = k, && \text{(4d)}\\
& && 0 \le x(e) \le 1 \qquad (\forall e \in E(G')). && \text{(4e)}
\end{aligned}
$$

$$
\begin{aligned}
(D'_k) \quad & \text{minimize} && 2\sum_{v \in V(G)} p(v) + \sum_{v \in V(G') \setminus V(G)} p(v) + \sum_{e \in E(G')} q(e) + \beta k && \text{(5a)}\\
& \text{subject to} && p(u) + p(v) + q(e) + \beta k(e) \ge w'(e) \quad (\forall e = (u,v) \in E(G')), && \text{(5b)}\\
& && p(v) \ge 0 \qquad (\forall v \in V(G)), && \text{(5c)}\\
& && q \ge 0. && \text{(5d)}
\end{aligned}
$$

We define the linear program (P') as (P'_k) without the inequality 4d. We denote the dual programs of (P) and (P'), respectively, as (D) and (D'), correspondingly. These dual programs differ from (D_k) and (D'_k) in that they do not contain the variable β.

▶ **Fact 5.3.** *Consider an optimal integral primal solution x^* of (P_k) and an optimal dual solution p^*, q^*, α^*, β^* of (D_k). Define $M^* = \{e \in E : x^*(e) = 1\}$. From complementarity slackness we have the following:*

$$e \in M^* \implies p^*(u) + p^*(v) + q^*(e) + \sum_{s \in S: e \in E(s)} \alpha^*(s) + \beta^* = w(e) \qquad (\forall e = (u,v) \in E(G))$$

$$\begin{aligned} v \text{ is deficient in } M^* &\implies p^*(v) = 0 \qquad (\forall v \in V), \\ e \notin M^* &\implies q^*(e) = 0 \qquad (\forall e \in E), \\ s \text{ is not saturated in } M^* &\implies \alpha^*(s) = 0 \qquad (\forall s \in S). \end{aligned}$$

Similar constraints hold for the other linear programs. We identify a 2-matching with its incidence vector x.

Let us now explain how we use these linear programs. Observe that to show that G' contains no negative cycles, it suffices to demonstrate that M' is extreme in G', or in other words, that M' is an optimal solution of (P'_k). Below in Lemma 5.6 we prove that M' is extreme in G' if M is extreme in G. This means that we need to show that for every k, $0 \leq k \leq n$ in iteration k of Algorithm 1, the computed 2-matching M of size k is an optimal solution of (P_k). Of course, the empty 2-matching is an optimal solution of (P_0). Assuming that we have an extreme 2-matching of size $k-1$ in M, we build M' and G' and find a minimum M'-augmenting path in G'. Next we show that by applying P to M' we obtain an (l,u)-matching N of size k, which is extreme in G'. This (l,u)-matching N corresponds to a 2-matching M_1 of size k. We prove that the optimality of the solution N of (P'_k) implies the optimality of the solution M_1 of (P_k).

▶ **Lemma 5.4.** *Consider any bipartite graph H and $l, u : V(H) \to \mathbb{N}_{\geq 0}$. Then an (l,u)-matching polytope is defined by the following inequalities:*

$$\begin{aligned} l(v) \leq x(\delta(v)) \leq u(v) &\quad (\forall v \in V(H)), \\ 0 \leq x(e) \leq 1 &\quad (\forall e \in E(H)). \end{aligned}$$

Proof. It is known that an incidence matrix of a bipartite graph is totally unimodular, hence incidence matrix A_H of H is totally unimodular. Observe that $P = \{x \in \mathbb{R}^{E(H)} : l \leq A_H x \leq u \wedge 0 \leq x \leq 1\}$, hence P is integral from theory of totally unimodular matrices. ◀

▶ **Lemma 5.5.** *Linear programs (P'_k) and (P') have integral optimal solutions.*

To compute a minimum M'-augmenting path in G', we first find an M'-augmenting path P with minimum alternating weight. To be able to do this, we need to know that there are no negative cycles in G'. In the following lemma we prove the absence of negative cycles in G'. Next, if needed, we shorten P.

▶ **Lemma 5.6.** *Assume that M is an optimal solution to (P_k). Then M' is an optimal solution to (P'_k), and thus M' is extreme in G'. Hence, there are no negative cycles in G'.*

▶ **Lemma 5.7.** *Let M' be an extreme (l,u)-matching in G' and P a minimum M'-augmenting path. Then $N' = M' \oplus P$ is extreme in G'.*

The proof is almost identical to that of Theorem 17.2 in [32].

▶ **Lemma 5.8.** *Let P be a minimum M'-augmenting path and let $M'' = M' \oplus P$ and $N = \mathrm{img}(M'')$.*

If $M' \oplus P$ is extreme in G', then N is an optimal solution of (P_k).

▶ **Lemma 5.9.** *Assume that M' is a maximum-weight (l,u)-matching of G'. Then M is a maximum-weight square-free 2-matching of G.*

▶ **Lemma 5.10.** *Let P be a minimum M'-augmenting path and let $M'' = M' \oplus P$ and $M_1 = \text{img}(M'')$. Then $w(M_1) = w'(M') - \tilde{w}'(P)$ and M_1 can contain only unproblematic squares. M_1 can be transformed into a square-free 2-matching M_2 such that $w(M_2) = w(M_1)$.*

Proof. At the beginning we show that $w(M_1) = w'(M'')$. We have that $w'(M'') = w'(M') - \tilde{w}'(P)$. Observe that the flipping of half-edges does not change the weight of M_1. Hence $w(M_1) = w'(M'')$.

We observe that Fact 4.1 is still valid in the weighted case, because G' contains no negative alternating cycles. The same is true for Observation 4.6 because of Fact 4.1 and the following. We can shorten P going through d and u_s^2, b, c so that it uses three edges $(d, v_c^d), (v_c^d, v_d^c), (v_d^c, c)$ instead, because the weight of each of these two subpaths is the same and equal to $r(c) + r(d)$.

Next we notice that the proof of Lemma 4.7 goes through for the weighted setting as long as Lemma 4.2 is still valid. We now argue that it indeed is. It suffices to prove that if $s = (a,b,c,d)$ is such that $(a,b),(c,d) \in P \setminus M$, $(b,c),(a,d) \in M \setminus P$, then P can be shortened. Suppose that R is a subpath of P that consists of edges strictly between (a,b) and (c,d) and that a and d are its endpoints. (The case that b and c are the endpoints of R is symmetrical.) Let P_R be a path obtained from P by replacing its subpath R by an edge (a,d). We show that $w'(P_R) \leq w'(P)$, contradicting the choice of P. w is vertex-induced on s, therefore, $w'(P) - w'(P_R) = w'(R) - w'(a,d) = w'(R) + w(a,d) = w'(R) + w(a,b) + w(c,d) - w(b,c) = w'(R) + w'(a,b) + w'(b,c) + w'(c,d) = w'(C)$ where C is an alternating cycle of G consisting of R and three edges of s. Recall that $w'(C) \geq 0$ because M is extreme. ◀

The corollary of Lemmas 4.8, 5.9 and 5.10 is:

▶ **Theorem 5.11.** *Algorithm 1 computes a maximum (resp. maximum weight) square-free 2-matching of G.*

6 Maximum-weight $K_{t,t}$-free t-matchings in bipartite graphs

In this section we solve the weighted $K_{t,t}$-free t-matching problem in bipartite graphs. Since the case of $t = 2$ has already been addressed in the previous section, we assume that $t \geq 3$. Also, similarly as in Section 5, we assume that the weight function w is vertex-induced on every $K_{t,t}$ of G. The general scheme of the algorithm for the weighted $K_{t,t}$-free t-matching problem is similar to Algorithm 1- we give it below.

Algorithm 2 Computing a maximum weight $K_{t,t}$-free t-matching of a bipartite graph G.

1: Let M be an empty t-matching of G.
2: Construct an auxiliary bipartite graph $G' = (V', E')$ with $\mathcal{O}(tn)$ vertices and $\mathcal{O}(m)$ edges, and its t-matching M' of size $\mathcal{O}(tn)$ by replacing some $K_{t,t}$'s of G with gadgets containing half-edges.
3: Compute a minimum M'-augmenting path P of G'. If G' contains no M'-augmenting path with negative alternating weight, stop the algorithm and return M.
4: Apply P to M' obtaining M'' and extract a $K_{t,t}$-free t-matching M_1 of G from M'' such that $|M_1| = |M| + 1$ and $w(M_1) > w(M)$. Set M as M_1 and go to 2.

▷ Claim 6.1. Algorithm 2 runs in time $\mathcal{O}(tnm + t^2 n^2 \log n)$.

Proof. It is possible to implement all steps of an Algorithm 2 except 3 in linear time. Every step is executed $\mathcal{O}(tn)$ times since $|M|$ increases by one every time 4 is executed. ◁

Let us also remark that in the unweighted version of the problem the runtime of Algorithm 2 is $\mathcal{O}(nm + tn^2 + \sqrt{tnm})$ because 3 can be implemented to run in linear time and the algorithm can start not from an empty matching but from a maximum $(t-1)$-matching, whose computation takes $\mathcal{O}(\sqrt{tnm})$ time. Additionally, we may want to forbid only some subgraphs of G isomorphic to $K_{t,t}$. Then we proceed analogously, but replace only forbidden subgraphs with gadgets.

▶ **Definition 6.2.** *A subgraph H of G isomorphic to $K_{t,t}$ is **saturated** if it contains exactly one non-M-edge.*

▶ **Lemma 6.3.** *Any two different saturated $K_{t,t}$'s of G are vertex-disjoint.*

Proof. Let H be any saturated $K_{t,t}$ of G. We say that a vertex v of H is **basic** in H if v is an endpoint of the only non-M-edge of H. Otherwise, we say that v is **nonbasic** in H. Let $V_1(G) \cup V_2(G)$ denote the bipartition of $V(G)$. Thus H has exactly two basic vertices: one in $V_1(H)$ and the other in $V_2(H)$.

Let H_1 and H_2 be any two different saturated $K_{t,t}$'s of G with a common vertex $v \in V_1(G_1)$. We first show that it cannot happen that v is nonbasic both in H_1 and H_2. Suppose to the contrary that v is nonbasic both in H_1 and H_2. Then all t edges of M incident to v_1 belong both to H_1 and H_2. Hence, we get that $V_2(H_1) = V_2(H_2)$. Since $t \geq 3$, at least one of the vertices of $V_2(H_1)$ is nonbasic both in H_1 and H_2, which implies that $V_1(H_1) = V_1(H_2)$, but this contradicts the fact that H_1 and H_2 are different.

Suppose next that v is basic both in H_1 and H_2. Then v has $t-1$ incident edges of M in H_1 and $t-1$ incident edges of M in H_2. Since M is a t-matching, at least one of these edges, say (v, v'), belongs both to H_1 and H_2. This however means that v' is nonbasic both in H_1 and H_2 (because the endpoints of an M-edge cannot be both basic in the same saturated $K_{t,t}$).

Finally, consider the case when v is basic in H_1 and nonbasic in H_2. It means that at least two M-edges incident to v, say (v, v') and (v, v''), belong to both H_1 and H_2. Vertices v', v'' belong to $V_2(H_2)$, none of them is basic in H_1 and at most one is basic in H_2. Therefore, at least one of them is nonbasic both in H_1 and H_2 - a contradiction. ◀

▶ **Observation 6.4.** *All saturated $K_{t,t}$'s of G can be found in linear time.*

Proof. We can use a linear time algorithm by Galil and Italiano [10]. ◀

We replace every saturated $K_{t,t}$ of G with a gadget described below. By Lemma 6.3, all saturated $K_{t,t}$'s of G are vertex-disjoint.

The construction of the gadget for a saturated $K_{t,t}$ is the following. Let H by any $K_{t,t}$ of G, $A_H = \{a_1, a_2, \ldots, a_t\}$ be a set of vertices of one side of H and $B_H = \{b_1, b_2, \ldots, b_t\}$ - of the other side. Let $r : V(H) \to \mathbb{R}$ be a potential function of H. Assume that $(a_1, b_1) \notin M$. For every edge (a_i, b_j) of H, we introduce two subdivision vertices, $v^{a_i}_{b_j}$, $v^{b_j}_{a_i}$, two half-edges and one eliminator. We remove an eliminator of (a_1, b_1). We set the weight of every half-edge incident to $v \in V(H)$ to $r(v)$. Additionally, we add u^1_s and u^2_s to G'. We connect u^1_s to every subdivision vertex adjacent to some vertex of A_H and we connect u^2_s to every subdivision vertex adjacent to some vertex of B_H. We add all half-edges of this gadget except for $(a_1, v^{a_1}_{b_1})$ and $(b_1, v^{b_1}_{a_1})$ to M'. Additionally, we add $(u^1_s, v^{a_1}_{b_1})$ and $(u^2_s, v^{b_1}_{a_1})$ to M'.

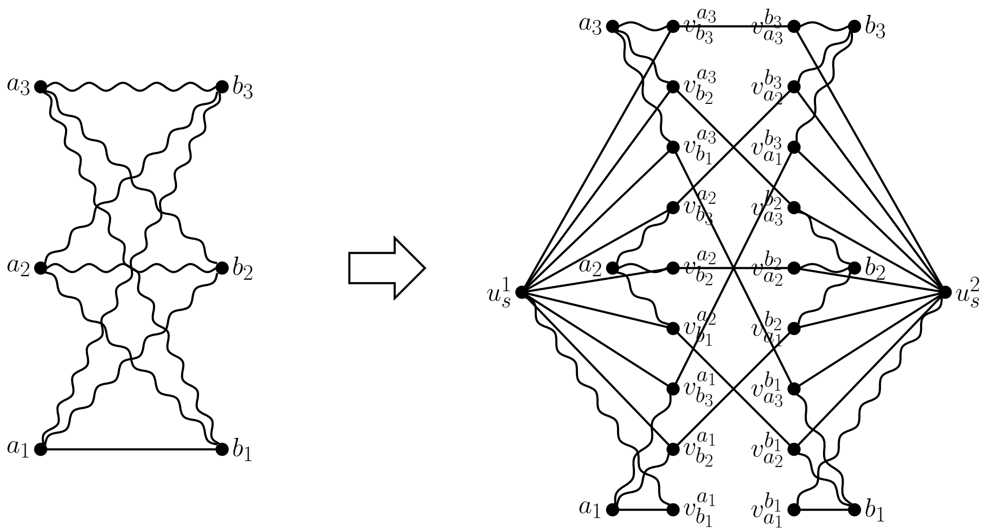

Figure 3 A gadget for a problematic $K_{3,3}$.

7 Restricted matchings

In this section we consider the following variant of the Restricted Matching Problem. We are given (1) a (not necessarily bipartite) graph $G = (V, E)$, (2) a natural number k, (3) a set of k subsets of edges $E_1, E_2, \ldots, E_k$ such that for each $1 \leq i \leq k$ (i) the graph G contains a clique or a bipartite clique on all endpoints of E_i; in the case of a bipartite clique, it contains the whole set E_i and (ii) the sets $E_1, \ldots, E_k$ are almost vertex-disjoint - the endpoints of any two different cliques have at most one vertex in common and (4) k natural numbers $r_1, r_2, \ldots, r_k$. The task is to find a maximum size matching M of G among ones that satisfy the condition: for each $1 \leq i \leq k$ it holds that M contains at most r_i edges of E_i. Any matching M of G that satisfies: for each $1 \leq i \leq k$ $|M \cap E_i| \leq r_i$ is called a ***restricted matching***.

To solve this problem we construct a graph G' with gadgets for each of the sets E_i. The construction of G' is similar to the one used for square-free 2-matchings. The precise construction of G' is the following. We start off with $G' = G$. For each $1 \leq i \leq k$ we build a subgraph, called a ***gadget for*** E_i. Let $n_i = |E_i|$. Each edge (p, q) of E_i is replaced with three new edges $(p, v_q^p), (v_q^p, v_p^q), (v_p^q, q)$, two of which are half-edges of (p, q) and the third one the eliminator of (p, q). If G contains a clique on all endpoints of E_i we introduce one new global vertex u_i and connect it with every subdivision vertex of an edge belonging to E_i. We set the interval of u_i as $[2n_i - 2r_i, 2n_i - 2r_i]$. If G contains a bipartite clique $K_i = (A_i \cup B_i, E_i')$ on all endpoints of E_i we introduce two new global vertices u_i^1 and u_i^2. We connect u_s^1 with every subdivision vertex of an edge of E_i, which is a neighbour of a vertex of A_i and similarly, we connect u_s^2 with every subdivision vertex of an edge of E_i, which is a neighbour of a vertex of B_i. We set the interval of both $u_i^1$and u_i^2 as $[n_i - r_i, n_i - r_i]$. Let $N = \sum_{i=1}^k n_i$, $R = \sum_{i=1}^k r_i$ and $E_R = \bigcup_{i=1}^k E_i$.

An (l, u)-matching M' of G' is to represent roughly a restricted matching M of G. If M' contains both half-edges of some edge $e \in E_R$, then e is included in M. If M' contains an eliminator of e, then e does not belong to M (is excluded from M). The intuition behind the gadget for the set E_i is that the global vertex or vertices in it are required to be matched to $2n_i - 2r_i$ subdivision vertices of edges of E_i. In this way they block $2n_i - 2r_i$ half-edges, which means that at most $2n_i - (2n_i - 2r_i) = 2r_i$ half-edges of edges of E_i can be present in M'. This implies that at most r_i edges of E_i can appear in the matching M.

▶ **Theorem 7.1.** *Any maximum (l, u)-matching M' of G' yields a maximum restricted matching of G.*

Proof. Any restricted matching M of G corresponds to an (l, u)-matching M_1 of G' such that $|M_1| = |M| + 2N - R$. To construct such M_1 we proceed as follows. We set M_1 to an empty (l, u)-matching. For each $e \in M \cap E_R$, we add both half-edges of e to M_1. For each edge $e \in M \setminus E_R$, we add e to M_1. Next for each $1 \leq i \leq k$ there exist at least $n_i - r_i$ edges of E_i that do not belong to M. We choose any such $(n_i - r_i)$-element subset $F_i \subseteq E_i$ and for each $e \in F_i$ we connect in M_1 the global vertex/the global vertices $u_i/u_i^1, u_i^2$ to the two subdivision vertices of e. For every edge e of $E_i \setminus (M \cup F_i)$ we add the eliminator of e to M_1. Let us note that the size of any (l, u)-matching N' of G' satisfies: $2|N'| = \sum_{v \in V} deg_{N'}(v) + \sum_{v \in V' \setminus V} deg_{N'}(v) = \sum_{v \in V} deg_{N'}(v) + 4N - 2R$. This means that the thus constructed M_1 has size $|M| + 2N - R$.

Consider now any (l, u)-matching M' of G'. If for every edge $e \in E_r$ it holds that either both half-edges of e are contained in M' or none, then we say that M' is ***integral***. Every integral M' yields a restricted matching M of G such that $|M| = |M'| - 2N + R$. We next show that even when M' is not integral, we are able to build a restricted matching M such that $|M| = |M'| - 2N + R$. We only need to say what to do with half-edges, i.e., with those edges of E_r for which M' contains only one of their half-edges. We deal with each set E_i separately. Suppose first that E_i is such that the graph G contains a clique on all endpoints of E_i. Let us notice that the number of edges of E_i with exactly one of its half-edges contained in M' is even. Let F_i denote such edges and V_i' denote those endpoints of edges of F_i that are incident to some half-edge of an edge of F_i. Then $|V_i'|$ is even. We pair vertices of V_i' in an arbitrary way and for two vertices u, u' belonging to one pair we replace two half-edge of M' incident with u and u' with one edge (u, u') of M. The case when G contains a bipartite clique on all endpoints of E_i is analogous. ◀

References

1 Maxim Babenko. Improved algorithms for even factors and square-free simple b-matchings. *Algorithmica*, 64(3):362–383, 2012.

2 Kristóf Bérczi and Yusuke Kobayashi. An algorithm for $(n-3)$-connectivity augmentation problem: Jump system approach. *Journal of Combinatorial Theory, Series B*, 102(3):565–587, 2012.

3 Kristóf Bérczi and László Végh. Restricted b-matchings in degree-bounded graphs. In *Integer Programming and Combinatorial Optimization*, pages 43–56, 2010.

4 Gérard Cornuéjols and William Pulleyblank. A matching problem with side conditions. *Discrete Mathematics*, 29(2):135–159, 1980.

5 Marshall Fisher, George Nemhauser, and Laurence Wolsey. An analysis of approximations for finding a maximum weight hamiltonian circuit. *Operations Research*, 27(4):799–809, 1979.

6 Tamás Fleiner. Uncrossing a family of set-pairs. *Combinatorica*, 21:145–150, 2001.

7 András Frank. Restricted t-matchings in bipartite graphs. *Discrete Applied Mathematics*, 131(2):337–346, 2003.

8 András Frank and Tibor Jordán. Minimal edge-coverings of pairs of sets. *Journal of Combinatorial Theory, Series B*, 65(1):73–110, 1995.

9 Michael Fredman and Robert Endre Tarjan. Fibonacci heaps and their uses in improved network optimization algorithms. *Journal of the ACM*, 34(3):596–615, 1987.

10 Zvi Galil and Giuseppe Italiano. Reducing edge connectivity to vertex connectivity. *SIGACT News*, 22:57–61, 1991.

11 Michael Garey and David Johnson. *Computers and Intractability.* Freeman, 1979.

12 Jim Geelen. The C_6-free 2-factor problem in bipartite graphs is NP-complete. Unpublished, 1999.

13 David Hartvigsen. *Extensions of Matching Theory*. PhD thesis, Carnegie-Mellon University, 1984.

14 David Hartvigsen. Finding maximum square-free 2-matchings in bipartite graphs. *Journal of Combinatorial Theory, Series B*, 96(5):693–705, 2006.

15 David Hartvigsen and Yanjun Li. Maximum cardinality simple 2-matchings in subcubic graphs. *SIAM Journal on Optimization*, 21(3):1027–1045, 2011.

16 David Hartvigsen and Yanjun Li. Polyhedron of triangle-free simple 2-matchings in subcubic graphs. *Mathematical Programming*, 138:43–82, 2013.

17 Alon Itai, Michael Rodeh, and Steven Tanimoto. Some matching problems for bipartite graphs. *Journal of the ACM*, 25(4):517–525, 1978.

18 Zoltán Király. C_4-free 2-factors in bipartite graphs. Technical report, Egerváry Research Group, 1999.

19 Zoltán Király. Restricted t-matchings in bipartite graphs. Technical report, Egerváry Research Group, 2009.

20 Yusuke Kobayashi. A simple algorithm for finding a maximum triangle-free 2-matching in subcubic graphs. *Discrete Optimization*, 7:197–202, 2010.

21 Yusuke Kobayashi. Weighted triangle-free 2-matching problem with edge-disjoint forbidden triangles. In *Integer Programming and Combinatorial Optimization*, pages 280–293, 2020.

22 Yusuke Kobayashi and Xin Yin. An algorithm for finding a maximum t-matching excluding complete partite subgraphs. *Discrete Optimization*, 9(2):98–108, 2012.

23 László Lovász and Michael Plummer. *Matching theory*. AMS Chelsea Publishing, corrected reprint of the 1986 original edition, 2009.

24 Márton Makai. On maximum cost $K_{t,t}$-free t-matchings of bipartite graphs. *SIAM Journal on Discrete Mathematics*, 21:349–360, 2007.

25 Monaldo Mastrolilli and Georgios Stamoulis. Constrained matching problems in bipartite graphs. In *Combinatorial Optimization*, pages 344–355, 2012.

26 Yunsun Nam. *Matching Theory: Subgraphs with Degree Constraints and other Properties*. PhD thesis, University of British Columbia, 1994.

27 Katarzyna Paluch, Khaled Elbassioni, and Anke van Zuylen. Simpler approximation of the maximum asymmetric traveling salesman problem. In *29th International Symposium on Theoretical Aspects of Computer Science*, pages 501–506, 2012.

28 Katarzyna Paluch and Mateusz Wasylkiewicz. A simple combinatorial algorithm for restricted 2-matchings in subcubic graphs - via half-edges. *Information Processing Letters*, 171, 2021.

29 Gyula Pap. Alternating paths revisited ii: restricted b-matchings in bipartite graphs. Technical report, Egerváry Research Group, 2005.

30 Gyula Pap. Combinatorial algorithms for matchings, even factors and square-free 2-factors. *Mathematical Programming*, 110:57–69, 2007.

31 Irena Rusu. Maximum weight edge-constrained matchings. *Discrete Applied Mathematics*, 156(5):662–672, 2008.

32 Alexander Schrijver. *Combinatorial optimization: polyhedra and efficiency*, volume 24. Springer Science & Business Media, 2003.

33 Kenjiro Takazawa. A weighted $K_{t,t}$-free t-factor algorithm for bipartite graphs. *Mathematics of Operations Research*, 34(2):351–362, 2009.

34 Kenjiro Takazawa. Decomposition theorems for square-free 2-matchings in bipartite graphs. *Discrete Applied Mathematics*, 233:215–223, 2017.

35 Kenjiro Takazawa. Finding a maximum 2-matching excluding prescribed cycles in bipartite graphs. *Discrete Optimization*, 26:26–40, 2017.

36 László Végh and András Benczúr. Primal-dual approach for directed vertex connectivity augmentation and generalizations. In *SODA 2005. Proceedings of the sixteenth annual ACM-SIAM symposium on discrete algorithms.*, pages 186–194, 2005.