

Triangle-Free 2-Matchings Revisited

Maxim Babenko ^{*}, Alexey Gusakov ^{**}, Ilya Razenshteyn ^{***}

Moscow State University

Abstract. A *2-matching* in an undirected graph $G = (VG, EG)$ is a function $x: EG \rightarrow \{0, 1, 2\}$ such that for each node $v \in VG$ the sum of values $x(e)$ on all edges e incident to v does not exceed 2. The *size* of x is the sum $\sum_e x(e)$. If $\{e \in EG \mid x(e) \neq 0\}$ contains no triangles then x is called *triangle-free*.

Cornuéjols and Pulleyblank devised a combinatorial $O(mn)$ -algorithm that finds a triangle free 2-matching of maximum size (hereinafter $n := |VG|$, $m := |EG|$) and also established a min-max theorem.

We claim that this approach is, in fact, superfluous by demonstrating how their results may be obtained directly from the Edmonds–Gallai decomposition. Applying the algorithm of Micali and Vazirani we are able to find a maximum triangle-free 2-matching in $O(m\sqrt{n})$ -time. Also we give a short self-contained algorithmic proof of the min-max theorem.

Next, we consider the case of regular graphs. It is well-known that every regular graph admits a perfect 2-matching. One can easily strengthen this result and prove that every d -regular graph (for $d \geq 3$) contains a perfect triangle-free 2-matching. We give the following algorithms for finding a perfect triangle-free 2-matching in a d -regular graph: an $O(n)$ -algorithm for $d = 3$, an $O(m + n^{3/2})$ -algorithm for $d = 2k$ ($k \geq 2$), and an $O(n^2)$ -algorithm for $d = 2k + 1$ ($k \geq 2$).

1 Introduction

1.1 Basic Notation and Definitions

We shall use some standard graph-theoretic notation throughout the paper. For an undirected graph G we denote its sets of nodes and edges by VG and EG , respectively. For a directed graph we speak of arcs rather than edges and denote the arc set of G by AG . A similar notation is used for paths, trees, and etc. Unless stated otherwise, we do not allow loops and parallel edges or arcs in graphs. An undirected graph is called *d-regular* (or just *regular* if the value of d is unimportant) if all degrees of its nodes are equal to d . A subgraph of G induced by a subset $U \subseteq VG$ is denoted by $G[U]$.

^{*} Email: max@adde.math.msu.su. Supported by RFBR grant 09-01-00709-a.

^{**} Email: agusakov@gmail.com

^{***} Email: ilyaraz@gmail.com

1.2 Triangle-Free 2-Matchings

Definition 1. Given an undirected graph G , a 2-matching in G is a function $x: EG \rightarrow \{0, 1, 2\}$ such that for each node $v \in VG$ the sum of values $x(e)$ on all edges e incident to v does not exceed 2.

A natural optimization problem is to find, given a graph G , a *maximum* 2-matching x in G , that is, a 2-matching of maximum *size* $\|x\| := \sum_e x(e)$. When $\|x\| = |VG|$ we call x *perfect*.

If $\{e \mid x(e) = 1\}$ partitions into a collection of node-disjoint circuits of odd length then x is called *basic*. Applying a straightforward reduction one can easily see that for each 2-matching there exists a basic 2-matching of the same or larger size (see [CP80, Theorem 1.1]). From now on we shall only consider basic 2-matchings x .

One may think of a basic 2-matching x as a collection of node disjoint *double edges* (each contributing 2 to $\|x\|$) and *odd length circuits* (where each edge of the latter contributes 1 to $\|x\|$). See Fig. 1.2(a) for an example.

Computing the maximum size $\nu_2(G)$ of a 2-matching in G reduces to finding a maximum matching in an auxiliary bipartite graph obtained by splitting the nodes of G . Therefore, the problem is solvable in $O(m\sqrt{n})$ -time with the help of Hopcroft–Karp’s algorithm [HK73] (hereinafter $n := |VG|$, $m := |EG|$). A simple min-max relation is known (see [Sch03, Th. 6.1.4] for an equivalent statement):

Theorem 1. $\nu_2(G) := \min_{U \subseteq VG} (|VG| + |U| - \text{iso}(G - U))$.

Here $\nu_2(G)$ is the maximum size of a 2-matching in G , $G - U$ denotes the graph obtained from G by removing nodes U (i.e. $G[VG - U]$) and $\text{iso}(H)$ stands for the number of isolated nodes in H . The reader may refer to [Sch03, Ch. 30] and [LP86, Ch. 6] for a survey.

Let $\text{supp}(x)$ denote $\{e \in EG \mid x(e) \neq 0\}$. The following refinement of 2-matchings was studied by Cornuéjols and Pulleyblank [CP80] in connection with the Hamilton cycle problem:

Definition 2. Call a 2-matching x *triangle-free* if $\text{supp}(x)$ contains no triangle.

They investigated the problem of finding a maximum size triangle-free 2-matching, devised a combinatorial algorithm, and gave an $O(n^3)$ estimate for its running time. Their algorithm initially starts with $x := 0$ and then performs a sequence of *augmentation steps* each aiming to increase $\|x\|$. Totally, there are $O(n)$ steps and a more careful analysis easily shows that the step can be implemented to run in $O(m)$ time. Hence, in fact the running time of their algorithm is $O(mn)$.

The above algorithm also yields a min-max relation as a by-product. Denote the maximum size of a triangle-free 2-matching in G by $\nu_2^3(G)$.

Definition 3. A triangle cluster is a connected graph whose edges partition into disjoint triangles such that any two triangles have at most one node in common and if such a node exists, it is an articulation point of the cluster. (See Fig. 1.2(b) for an example.)

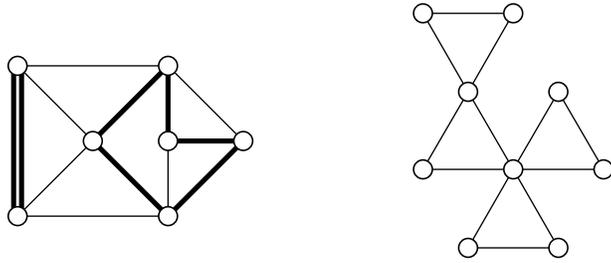


Fig. 1. (a) A perfect basic 2-matching. (b) A triangle cluster.

Let $\text{cluster}(H)$ be the number of the connected components of H that are triangle clusters.

Theorem 2. $\nu_2^3(G) := \min_{U \subseteq V_G} (|VG| + |U| - \text{cluster}(G - U))$.

One may notice a close similarity between Theorem 2 and Theorem 1.

1.3 Our Contribution

The goal of the present paper is to devise a faster algorithm for constructing a maximum triangle-free 2-matching. We give a number of results that improve the above-mentioned $O(mn)$ time bound.

Firstly, let G be an arbitrary undirected graph. We claim that the direct augmenting approach of Cornuéjols and Pulleyblank is, in fact, superfluous. In Section 2 we show how one can compute a maximum triangle-free 2-matching with the help of the Edmonds–Gallai decomposition [LP86, Sec. 3.2]. The resulting algorithm runs in $O(m\sqrt{n})$ time (assuming that the maximum matching in G is computed by the algorithm of Micali and Vazirani [MV80]). Also, this approach directly yields Theorem 2.

Secondly, there are some well-known results on matchings in regular graphs.

Theorem 3. *Every 3-regular bridgeless graph has a perfect matching.*

Theorem 4. *Every regular bipartite graph has a perfect matching.*

The former theorem is usually credited to Petersen while the second one is an easy consequence of Hall’s condition.

Theorem 5 (Cole, Ost, Schirra [COS01]). *There exists a linear time algorithm that finds a perfect matching in a regular bipartite graph.*

Theorem 4 and Theorem 5 imply the following:

Corollary 1. *Every regular graph has a perfect 2-matching. The latter 2-matching can be found in linear time.*

In Section 3 we consider the analogues of Corollary 1 with 2-matchings replaced by triangle-free 2-matchings. We prove that every d -regular graph ($d \geq 3$) has a perfect triangle-free 2-matching. This result gives a simple and natural strengthening to the non-algorithmic part of Corollary 1.

As for the complexity of finding a perfect 2-matching in a d -regular graph it turns out heavily depending on d . The ultimate goal is a linear time algorithm but we are only able to fulfill this task for $d = 3$. The case of even d ($d \geq 4$) turns out reducible to $d = 4$, so the problem is solvable in $O(m + n^{3/2})$ time by the use of the general algorithm (since $m = O(n)$ for 4-regular graphs). The case of odd d ($d \geq 5$) is harder, we give an $O(n^2)$ -time algorithm, which improves the general time bound of $O(m\sqrt{n})$ when $m = \omega(n^{3/2})$.

2 General Graphs

2.1 Factor-Critical Graphs, Matchings, and Decompositions

We need several standard facts concerning maximum matchings (see [LP86, Ch. 3] for a survey). For a graph G , let $\nu(G)$ denote the maximum size of a matching in G and $\text{odd}(H)$ be the number of connected components of H with an odd number of vertices.

Theorem 6 (Tutte–Berge). $\nu(G) = \min_{U \subseteq VG} \frac{1}{2} (|VG| + |U| - \text{odd}(G - U))$.

Definition 4. A graph G is factor-critical if for any $v \in VG$, $G - v$ admits a perfect matching.

Theorem 7 (Edmonds–Gallai). Consider a graph G and put

$$\begin{aligned} D &:= \{v \in VG \mid \text{there exists a maximum size matching missing } v\}, \\ A &:= \{v \in VG \mid v \text{ is a neighbor of } D\}, \\ C &:= VG - (A \cup D). \end{aligned}$$

Then $U := A$ achieves the minimum in the Tutte–Berge formula, and D is the union of the odd connected components of $G[VG - A]$. Every connected component of $G[D]$ is factor-critical. Any maximum matching in G induces a perfect matching in $G[C]$ and a matching in $G[VG - C]$ that matches all nodes of A to distinct connected components of $G[D]$.

We note that once a maximum matching M in G is found, an Edmonds–Gallai decomposition of G can be constructed in linear time by running a search for an M -augmenting path. Most algorithms that find M yield this decomposition as a by-product. Also, the above augmenting path search may be adapted to produce an *odd ear decomposition* of every odd connected component of $G[VG - A]$:

Definition 5. An ear decomposition $G_0, G_1, \dots, G_k = G$ of a graph G is a sequence of graphs where G_0 consists of a single node, and for each $i = 0, \dots, k - 1$, G_{i+1} obtained from G_i by adding the edges and the intermediate nodes of an ear. An ear of G_i is a path P_i in G_{i+1} such that the only nodes of P_i belonging to G_i are its (possibly coinciding) endpoints. An ear decomposition with all ears having an odd number of edges is called *odd*.

The next statement is widely-known and, in fact, comprises a part of the blossom-shrinking approach to constructing a maximum matching.

Lemma 1. *Given an odd ear decomposition of a factor-critical graph G and a node $v \in VG$ one can construct in linear time a matching M in G that misses exactly the node v .*

Finally, we classify factor-critical graphs depending on the existence of a perfect triangle-free 2-matching. The proof of the next lemma is implicit in [CP83] and one can easily turn it into an algorithm:

Lemma 2. *Each factor-critical graph G is either a triangle cluster or has a perfect triangle-free 2-matching x . Moreover, if an odd ear decomposition of G is known then these cases can be distinguished and x (if exists) can be constructed in linear time.*

2.2 The Algorithm

For the sake of completeness, we first establish an upper bound on the size of a triangle-free 2-matching.

Lemma 3. *For each $U \subseteq VG$, $\nu_2^3(G) \leq |VG| + |U| - \text{cluster}(G - U)$.*

Proof.

Removing a single node from a graph G may decrease $\nu_2^3(G)$ by at most 2. Hence, $\nu_2^3(G) \leq \nu_2^3(G - U) + 2|U|$. Also, $\nu_2^3(G - U) \leq (|VG| - |U|) - \text{cluster}(G - U)$ since every connected component of $G - U$ that is a triangle cluster lacks a perfect triangle-free 2-matching. Combining these inequalities, one gets the desired result. \square

The next theorem both gives an efficient algorithm a self-contained proof of the min-max formula.

Theorem 8. *A maximum triangle-free 2-matching can be found in $O(m\sqrt{n})$ time.*

Proof.

Construct an Edmonds–Gallai decomposition of G , call it (D, A, C) , and consider odd ear decompositions of the connected components of $G[D]$. As indicated earlier, the complexity of this step is dominated by finding a maximum matching M in G . The latter can be done in $O(m\sqrt{n})$ time (see [MV80]).

The matching M induces a perfect matching M_C in $G[C]$. We turn M_C into double edges in the desired triangle-free 2-matching x by putting $x(e) := 2$ for each $e \in M_C$.

Next, we build a bipartite graph H . The nodes in the upper part of H correspond to the components of $G[D]$, the nodes in the lower part of H are just the nodes of A . There is an edge between a component C and a node v in H if and only if there is at least one edge between C and v in G . Let us call the components that are triangle clusters *bad* and the others *good*. Consider another

bipartite graph H' formed from H by dropping all nodes (in the upper part) corresponding to good components.

The algorithm finds a maximum matching $M_{H'}$ in H' and then augments it to a maximum matching M_H in H . This is done in $O(m\sqrt{n})$ time using Hopcroft–Karp algorithm [HK73]. It is well-known that an augmentation can only increase the set of matched nodes, hence every bad component matched by $M_{H'}$ is also matched by M_H and vice versa. From the properties of Edmonds–Gallai decomposition it follows that M_H matches all nodes in A .

Each edge $e \in M_H$ corresponds to an edge $\tilde{e} \in EG$, we put $x(\tilde{e}) := 2$.

Finally, we deal with the components of $G[D]$. Let C be a component that is matched (in M_H) by, say, an edge $e_C \in M_H$. As earlier, let \tilde{e}_C be the preimage of e_C in G . Since C is factor-critical, there exists a matching M_C in C that misses exactly the node in C covered by \tilde{e}_C . We find M_C in linear time (see Lemma 1) and put $x(e) := 2$ for each $e \in M_C$.

As for the unmatched components, we consider good and bad ones separately. If an unmatched component C is good, we apply Lemma 2 to find (in linear time) and add to x a perfect triangle-free 2-matching in C . If C is bad, we employ Lemma 1 and find (in linear time) a matching M_C in C that covers all the nodes except for an arbitrary chosen one and set $x(e) := 2$ for each $e \in M_C$.

The running time of the above procedure is dominated by constructing the Edmonds–Gallai decomposition of G and finding matchings $M_{H'}$ and M_H . Clearly, it is $O(m\sqrt{n})$.

It remains to prove that x is a maximum triangle-free 2-matching. Let n_{bad} be the number of bad components in $G[D]$. Among these components, let k_{bad} be matched by $M_{H'}$ (and, hence, by M_H). Then $\|x\| = |VG| - (n_{\text{bad}} - k_{\text{bad}})$. From König–Egervary theorem (see, e.g., [LP86]) there exists a vertex cover L in H' of cardinality k_{bad} (i.e. a subset $L \subseteq VH'$ such that each edge in H' is incident to at least one node in L). Put $L = L_A \cup L_D$, where L_A are the nodes of L belonging to the lower part of H and L_D are the nodes from the upper part. The graph $G - L_A$ contains at least $n_{\text{bad}} - |L_D|$ components that are triangle clusters. (They correspond to the uncovered nodes in the upper part of H' . Indeed, these components are only connected to L_A in the lower part.) Hence, putting $U := L_A$ in Lemma 3 one gets $\nu_2^3(G) \leq |VG| + |L_A| - (n_{\text{bad}} - |L_D|) = |VG| + |L| - n_{\text{bad}} = |VG| - (n_{\text{bad}} - k_{\text{bad}}) = \|x\|$. Therefore, x is a maximum triangle-free 2-matching, as claimed. \square

3 Regular Graphs

3.1 Existence of a Perfect Triangle-Free 2-Matching

Theorem 9. *Let G be a graph with $n - q$ nodes of degree d and q nodes of degree $d - 1$ ($d \geq 3$). Then, there exists a triangle-free 2-matching in G of size at least $n - q/d$.*

Proof.

Consider an arbitrary subset $U \subseteq VG$. Put $t := \text{cluster}(G - U)$ and let C_1, \dots, C_t be the triangle cluster components of $G - U$. Fix an arbitrary component $H := C_i$ and let k be the number of triangles in H . One has $|VH| = 2k + 1$. Each node of H is incident to either d or $d - 1$ edges. Let q_i denote the number of nodes of degree $d - 1$ in H . Since $|EH| = 3k$ it follows that $(2k + 1)d - 6k - q_i = d + (2d - 6)k - q_i \geq d - q_i$ edges of G connect H to U . Totally, the nodes in U have at least $\sum_{i=1}^t (d - q_i) \geq td - q$ incident edges. On the other hand, each node of U has the degree of at most d , hence $td - q \leq |U|d$ therefore $t - |U| \leq q/d$. By the min-max formula (see Theorem 2) this implies the desired bound. \square

Corollary 2. *Every d -regular graph ($d \geq 3$) has a perfect triangle-free 2-matching.*

3.2 Cubic graphs

For $d = 3$ we speed up the general algorithm ultimately as follows:

Theorem 10. *A perfect triangle-free 2-matching in a 3-regular graph can be found in linear time.*

Proof.

Consider a 3-regular graph G . First, we find an arbitrary inclusion-wise maximal collection of node-disjoint triangles $\Delta_1, \dots, \Delta_k$ in G . This is done in linear time by performing a local search at each node $v \in VG$. Next, we contract $\Delta_1, \dots, \Delta_k$ into *composite* nodes z_1, \dots, z_k and obtain another 3-regular graph G' (note that G' may contain multiple parallel edges).

Construct a bipartite graph H' from G' as follows. Every node $v \in VG'$ is split into a pair of nodes v^1 and v^2 . Every edge $\{u, v\} \in EG'$ generates edges $\{u^1, v^2\}$ and $\{v^1, u^2\}$ in H' . There is a natural surjective many-to-one correspondence between perfect matchings in H' and perfect 2-matchings in G' . Applying the algorithm of Cole, Ost and Schirra [COS01] to H' we construct a perfect 2-matching x' in G' in linear time. As usual, we assume that x' is basic, in particular x' contains no circuit of length 2 (i.e. $\text{supp}(x')$ contains no pair of parallel edges).

Our final goal is to expand x' into a perfect triangle-free 2-matching x in G . The latter is done as follows. Consider an arbitrary composite node z_i obtained by contracting Δ_i in G . Suppose that a double edge e of x' is incident to z_i in G' . We keep the preimage of e as a double edge of x and add another double edge connecting the remaining pair of nodes in Δ_i . See Fig. 3.2(a).

Next, suppose that x' contains an odd-length circuit C' passing through z_i . Then, we expand z_i to Δ_i and insert an additional pair of edges to C' . Note that the length of the resulting circuit C is odd and is no less than 5. See Fig. 3.2(b).

Clearly, the resulting 2-matching x is perfect. But why is it triangle-free? For sake of contradiction, suppose that Δ is a triangle in $\text{supp}(x)$. Then, Δ is an

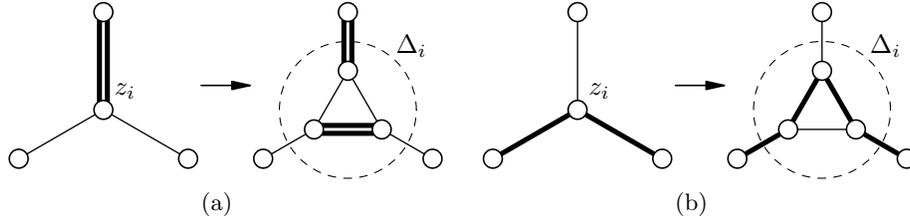


Fig. 2. Uncontraction of z_i .

odd circuit in x' and no node of Δ is composite. Hence, Δ is a triangle disjoint from $\Delta_1, \dots, \Delta_k$ — a contradiction. \square

Combining the above connection between triangle-free 2-matchings in G and 2-matchings in G' with the result of Voorhoeve [Voo79] one can prove the following:

Theorem 11. *There exists a constant $c > 1$ such that every 3-regular graph G contains at least c^n perfect triangle-free 2-matchings.*

3.3 Even-degree graphs

To find a perfect triangle-free 2-matching in a $2k$ -regular graph G ($k \geq 2$) we replace it by a 4-regular spanning subgraph and then apply the general algorithm.

Lemma 4. *For each $2k$ -regular ($k \geq 1$) graph G there exists and can be found in linear time a 2-regular spanning subgraph.*

Proof.

Since the degrees of all nodes in G are even, EG decomposes into a collection of edge-disjoint circuits. This decomposition takes linear time. For each circuit C from the above decomposition we choose an arbitrary direction and traverse C in this direction turning undirected edges into directed arcs. Let \vec{G} denote the resulting digraph. For each node v exactly k arcs of \vec{G} enter v and exactly k arcs leave v .

Next, we construct a bipartite graph H from \vec{G} as follows: each node $v \in \vec{G}$ generates a pair of nodes $v^1, v^2 \in VH$, each arc $(u, v) \in A\vec{G}$ generates an edge $\{u^1, v^2\} \in EH$. The graph H is k -regular and, hence, contains a perfect matching M (which, by Theorem 5, can be found in linear time). Each edge of M corresponds to an arc of \vec{G} and, therefore, to an edge of G . Clearly, the set of the latter edges forms a 2-regular spanning subgraph of G . \square

Theorem 12. *A perfect triangle-free 2-matching in a d -regular graph ($d = 2k$, $k \geq 2$) can be found in $O(m + n^{3/2})$ time.*

Proof.

Consider an undirected $2k$ -regular graph G . Apply Lemma 4 and construct find a 2-regular spanning subgraph H_1 of G . Next, discard the edges of H_1 and reapply Lemma 4 thus obtaining another 2-regular spanning subgraph H_2 (here we use that $k \geq 2$). Their union $H := (VG, EH_1 \cup EH_2)$ is a 4-regular spanning subgraph of G . By Corollary 2 graph H still contains a perfect triangle-free 2-matching x , which can be found by the algorithm from Theorem 8. It takes $O(m)$ time to construct H and $O(n^{3/2})$ time, totally $O(m + n^{3/2})$ time, as claimed. \square

3.4 Odd-degree graphs

The case $d = 2k + 1$ ($k \geq 2$) is more involved. We extract a spanning subgraph H of G whose node degrees are 3 and 4. A careful choice of H allows us to ensure that the number of nodes of degree 3 is $O(n/d)$. Then, by Theorem 9 subgraph H contains a nearly-perfect triangle-free 2-matching. The latter is found and then augmented to a perfect one with the help of the algorithm from [CP80]. More details follow.

Lemma 5. *There exists and can be found in linear time a spanning subgraph H of graph G with nodes degrees equal to 3 and 4. Moreover, at most $O(n/d)$ nodes in H are of degree 3.*

Proof.

Let us partition the nodes of G into pairs (in an arbitrary way) and add $n/2$ virtual edges connecting these pairs. The resulting graph G' is $2k + 2$ -regular. (Note that G' may contain multiple parallel edges.)

Our task is find a 4-regular spanning subgraph H' of G' containing at most $O(n/d)$ virtual edges. Once this subgraph is found, the auxiliary edges are dropped creating $O(n/d)$ nodes of degree 3 (recall that each node of G is incident to exactly one virtual edge).

Subgraph H' is constructed by repeatedly pruning graph G' . During this process graph G' remains d' -regular for some even d' (initially $d' := d + 1$).

At each pruning step we first examine d' . Two cases are possible. Suppose d' is divisible by 4, then a *large* step is executed. The graph G' is decomposed into a collection of edge-disjoint circuits. In each circuit, every second edge is marked as *red* while others are marked as *blue*. This red-blue coloring partitions G' into a pair of spanning $d'/2$ -regular subgraphs. We replace G' by the one containing the smallest number of virtual edges. The second case (which leads to a *small* step) applies if d' is not divisible by 4. Then, with the help of Lemma 4 a 2-regular spanning subgraph is found in G' . The edges of this subgraph are removed from G' , so d' decreases by 2.

The process stops when d' reaches 4 yielding the desired subgraph H' . Totally, there are $O(\log d)$ large (and hence also small) steps each taking time proportional to the number of remaining edges. The latter decreases exponentially, hence the total time to construct H' is linear.

It remains to bound the number of virtual edges in H' . There are exactly $t := \lceil \log_2(d+1)/4 \rceil$ large steps performed by the algorithm. Each of the latter decreases the number of virtual edges in the current subgraph by at least a factor of 2. Hence, at the end there are $O(n/2^t) = O(n/d)$ virtual edges in H' , as required. \square

Theorem 13. *A perfect triangle-free 2-matching in a d -regular graph ($d = 2k + 1$, $k \geq 2$) can be found in $O(n^2)$ time.*

Proof.

We apply Lemma 5 and construct a subgraph H in $O(m)$ time. Next, a maximum triangle-free 2-matching x is found in H , which takes $O(|EH| \cdot |VH|^{1/2}) = O(n^{3/2})$ time. By Theorem 9 the latter 2-matching obeys $n - \|x\| = O(n/d)$. To turn x into a perfect triangle-free 2-matching in G we apply the algorithm from [CP80] and perform $O(n/d)$ augmentation steps. Each step takes $O(m)$ time, so totally the desired perfect triangle-free 2-matching is constructed in $O(m + n^{3/2} + mn/d) = O(n^2)$ time. \square

Acknowledgements

The authors are thankful to Andrew Stankevich for fruitful suggestions and helpful discussions.

References

COS01. R. Cole, K. Ost, and S. Schirra. Edge-coloring bipartite multigraphs in $O(E \log D)$ time. *Combinatorica*, Vol. 21:5–12, 2001.

CP80. G. Cornuéjols and W. R. Pulleyblank. Perfect triangle-free 2-matchings. *Mathematical Programming Studies*, Volume 13:1–7, 1980.

CP83. G. Cornuéjols and W. R. Pulleyblank. Critical graphs, matchings and tours or a hierarchy of relaxations for the travelling salesman problem. *Combinatorica*, 3(1):35–52, 1983.

HK73. J. E. Hopcroft and R. M. Karp. An $n^{5/2}$ algorithm for maximum matchings in bipartite graphs. *SIAM Journal on Computing*, 2(4):225–231, 1973.

LP86. L. Lovász and M. D. Plummer. *Matching Theory*. Akadémiai Kiadó - North Holland, Budapest, 1986.

MV80. S. Micali and V. Vazirani. An $O(\sqrt{|V|} \cdot |E|)$ algorithm for finding maximum matching in general graphs. *Proc. 21st IEEE Symp. Foundations of Computer Science*, pages 248–255, 1980.

Sch03. A. Schrijver. *Combinatorial Optimization*. Springer, Berlin, 2003.

Voo79. M. Voorhoeve. A lower bound for the permanents of certain $(0, 1)$ -matrices. *Nederl. Akad. Wetensch. Indag. Math.*, 41(1):83–86, 1979.