Petr Kolman Bernard Lidický
Jean-Sébastien Sereni

ON FAIR EDGE DELETION PROBLEMS

P. KOLMAN, B. LIDICKÝ, AND J.-S. SERENI

Abstract. In edge deletion problems, we are given a graph G and a graph property π and the task is to find a subset of edges the deletion of which results in a subgraph of G satisfying the property π. Typically the objective is to minimize the total number of deleted edges while in less common fair versions the objective is to minimize the maximum number of edges removed from a single vertex. We focus on the minimum fair odd cycle transversal (OCT) problem where the task is to make the graph bipartite; the problem is closely related to improper colorings of graphs. Though the classical version of the problem was diligently studied, the minimum fair version brings new challenges. We describe a $\Theta(\sqrt{n})$ approximation algorithm for general graphs and an exact polynomial time algorithm for graphs of bounded treewidth. Though there are several general frameworks (e.g., MSOL) for dealing with optimization problems on graphs of bounded treewidth, the minimum fair OCT does not seem to fit into any of them. Analogous results are proved for minimum fair cut problem.

1. Introduction

Many problems in combinatorial optimization can be formulated as *edge* (or *node*) *deletion problems*. Given a graph $G = (V, E)$ and a graph property π (e.g., being a tree, a bipartite graph or a series-parallel graph), the problem is to find a subset of edges the deletion of which results in a subgraph of G satisfying the property π [31]. Typically, the objective function is to minimize the total number of deleted edges. (Since many edge deletion problems are NP-complete [5, 31], finding good approximation algorithms is an active and relevant area [2, 3].) We study edge deletion problems under a different objective function: our goal is to minimize the maximum number of edges removed from a single vertex (i.e., we want to minimize the maximum degree in (V, F) where F is the set of deleted edges). As we shall see, such an objective function has both a theoretical appeal and a relevance for practical applications.

Lin and Sahni [26] coined the term *fair edge deletion problems* in a paper dealing with the following problem: given an undirected graph $G = (V, E)$, find a subset F of edges such that $(V, E \setminus F)$ is connected and acyclic, and the maximum degree of the graph (V, F) is as small as possible. They proved that the problem is NP-complete. Surprisingly enough, this is one of the very rare edge deletion problems studied with the aforementioned *fair* objective function.

We focus on the (minimum) fair odd cycle transversal problem. An *odd cycle transversal* (OCT) of a graph $G = (V, E)$ is a subset $F \subseteq E$ of edges such that

Key words and phrases. odd cycle transversal, approximation algorithm, polynomial-time algorithm, tree-width, linear programming, improper coloring, cut, fair objective function, dynamic programming.

This research was partially supported by the grant GA ČR 201/09/0197.

the graph $(V, E \setminus F)$ is bipartite. The *minimum OCT* problem consists in finding an OCT of minimum size. The *minimum fair OCT*, on the other hand, consists in finding an OCT F such that the maximum degree of (V, F) is as small as possible. Informally, instead of minimizing the total cost, we aim at minimizing the maximum cost at a vertex. It is natural to ask whether the fair version substantially differs from the usual version of the problem. To answer this, we first review some known results about the minimum OCT problem and explain some links between the fair OCT problem and graph coloring.

The minimum OCT problem is also known as the odd-cycle (edge) cover [25], as the maximum cut problem and as the minimum uncut problem. The problem is NP-hard for general graphs [17] but polynomial on planar graphs [23] and on graphs with bounded tree-width [6]. Regarding approximations, there is an $O(\sqrt{\log n})$-approximation [1] for the minimum uncut problem (i.e., we minimize the number of removed edges) and a 0.879-approximation [18] for the maximum cut problem (i.e., we maximize the number of edges that remain in the graph). In both cases, the approximations are based on semidefinite relaxations of the problem considered.

Let us now underline the links between OCTs and *improper colorings* (also known as *defective colorings*). An (ℓ, k)*-coloring* of a graph $G = (V, E)$ is a partition of V into ℓ parts such that each part induces a subgraph of G with maximum degree at most k. Note that a graph has a $(2, k)$-coloring if and only if the optimum value of the minimum fair OCT problem is at most k. Improper coloring is a very natural generalization of the usual notion of proper coloring, which is a core topic of graph theory (in particular, an $(\ell, 0)$-coloring is just a proper ℓ-coloring). Improper colorings have been introduced and studied both as a theoretical notion [15, 22, 28, 29, 30] and as a tool to model practical problems, such as the telecommunication problem proposed by the company Alcatel [20, 21, 24]. Among the numerous results about improper colorings, let us point out that even for planar graphs, the $(2, k)$-coloring problem is NP-complete [12, 16] and, thus, the minimum fair OCT is NP-hard on planar graphs.

The substantial difference between the complexity of the fair and the usual versions of the OCT (cf. planar graphs) provides a momentum to study the fair OCT. (For example, it is not clear whether the minimum fair OCT problem for bounded tree-width graphs should be expected to be polynomial or NP-Complete.) Another reason to study the fair OCT is the lack of algorithms for the (ℓ, k)-coloring problems and, in particular, the $(2, k)$-coloring problem. Conditions on the girth of a planar graph to ensure the existence of a $(2, k)$-coloring have been studied [22, 28], as well as approximation algorithms for special classes of graphs [20, 24], however, general approximation algorithms are missing.

New results. Our contributions are as follows. First, we use linear programming to obtain an approximation algorithm of ratio $\Theta(\sqrt{n})$ for graphs on n vertices and we show that the bound on the performance of the algorithm is tight even for planar graphs (Section 2). Remarkably, the integrality gap of the linear program is $\Omega(n)$. Next, we provide a polynomial time algorithm to solve the minimum fair OCT problem for series-parallel graphs (Subsection 3.1). Series-parallel graphs have tree-width at most 2, and it turns out that it is possible to generalize our approach to obtain an exact algorithm for bounded tree-width graphs; this is the main result of the paper (Subsection 3.2).

Since there are dozens or even hundreds of papers describing polynomial time algorithms for different problems on graphs of bounded treewidth, we feel obligated to explain why we did write one more. A classical result of Courcelle [13] states that every graph property that is expressible in Monadic Second Order Logic (MSOL) can be solved in linear-time for graphs of bounded tree-width. Several frameworks generalizing this results have been designed, such as the Extended MSOL [4] and monadic second order evaluations [14]; other approaches yielding similar results include, e.g., the predicate calculus [10], cf. [9]. To some extend, these results make it possible to replace most of the specialized algorithms by a single general algorithm (though, most likely, the performance of the general algorithm will fall short of the performance of the specialized algorithms). It is straightforward to check that the minimum OCT problem (i.e., the maximum cut problem) can be formulated in the Extended MSOL and in the predicate calculus. Similarly, for an arbitrary but fixed integer k, the question whether there exists an OCT F such that the maximum degree of the graph (V, F) is at most k can be formulated in the same frameworks. However, as far as we know, the *fair* problems that we study in this paper fail to fall into any of these frameworks. This rises an entitling question, namely to formally define a class of problems that contains the minimum fair OCT problem, and are solvable in polynomial time for graphs of bounded tree-width. The question is all the more tempting that the methods we use to obtain polynomial-time exact algorithms are not far from standard technics; this may hint at the existence of a formal framework tailored to express *fair* problems.

Minimum fair cut. As we also observe, all our results extend to the *minimum fair cut* problem. Here, given a graph $G = (V, E)$ and two vertices $x, y \in V$, the problem consists in finding an x-y cut $F \subseteq E$ such that the maximum degree of (V, F) is as small as possible. Closely related is the *matching cut* problem. Is there a cut F in G that is also a matching? Chvátal [11] proved that for general graphs, the problem is NP-hard (cf. [27]); he did not use the name matching cut. For planar graphs, Bonsma [8] described an exact polynomial time algorithm for the matching cut. We describe a $\Theta(\sqrt{n})$-approximation algorithm for the minimum fair cut on general graphs, an exact algorithm for series-parallel graphs and outline how to generalize this algorithm for graphs of bounded tree-treewidth.

Notation. Given a graph $G = (V, E)$ and a vertex $v \in V$, we let $\delta(v)$ be the set of edges adjacent to v. If $F \subseteq E$, then $\deg_F(v)$ denotes the degree of v in the graph (V, F) and Δ_F the maximum degree in (V, F); we say that Δ_F is the *value* of the set F. For $X \subseteq V$, we define $G[X]$ to be the subgraph of G induced by X.

2. Approximation Algorithm

Given a graph $G = (V, E)$, let $\mathcal{C}$ be the set of all odd cycles (viewed as edge sets). The minimum fair OCT problem can be formulated using integer linear programming as follows.

$$
\begin{array}{lccl}
(1) & \min & k & \text{subject to} \\
 & \forall C \in \mathcal{C}, & \sum_{e \in C} x_e & \geq 1 \\
 & \forall u \in V, & \sum_{e \in \delta(v)} x_e & \leq k \\
 & \forall e \in E, & x_e & \in \{0, 1\}
\end{array}
$$

In a linear programming relaxation, the last condition is replaced by $x_e \geq 0$.

We pause to note that the number of inequalities in the linear program (LP) may be exponential in the size of G, nevertheless, the LP is solvable in polynomial time using the ellipsoid method (given a vector x, one can check whether there is a violated inequality [19]).

We also notice that the integrality gap of the relaxation by itself is very large, namely n: for odd n, consider a graph composed of a single cycle of length n. Then, the fractional optimum is $2/n$ (for each edge we set $x_e = 1/n$) while the integral optimum is 1. We now combine the relaxation with a few observations to obtain an $O(\sqrt{n})$-approximation.

Solve the LP for the graph $G = (V, E)$, and set $F = \{e \in E : \; x_e \geq 1/(4\sqrt{n})\}$ and $F' = E \setminus F$. Let H be the subset of edges of E that have both end degrees at most $\sqrt{n}+1$ in $(V, E \setminus F)$, that is, $H = \{\{u, v\} \in E \setminus F \mid \deg_{F'}(u) \leq \sqrt{n}+1, \; \deg_{F'}(v) \leq \sqrt{n}+1\}$. Let $G' = (V, E \setminus (F \cup H))$.

Lemma 2.1. *The graph $G' = (V, E \setminus (F \cup H))$ is bipartite.*

Proof. Suppose on the contrary that G' contains an odd cycle, and let C be the shortest of them. First, note that the length of C is at least $4\sqrt{n}+1$: since x is a feasible solution of the LP for G', every odd cycle of length at most $4\sqrt{n}$ in G' contains an edge e with $x_e \geq 1/(4\sqrt{n})$.

Let D be the set of edges with exactly one endpoint in C; note that the only edges with two endpoints in C are edges of the cycle. Since at least one of every two successive nodes in C has at least $\sqrt{n}+2$ neighbors and since at least $\sqrt{n}$ of these neighbors are outside C, the set D contains $\sqrt{n} \cdot 4\sqrt{n}/2 = 2n$ or more edges. On the other hand, since C is the shortest odd cycle, every vertex outside C has at most two neighbors in C. Hence, D contains strictly fewer than $2n$ edges, a contradiction. □

Theorem 2.2. *The above procedure computes an $O(\sqrt{n})$ approximation of the minimum fair odd cycle transversal.*

Proof. Let $\hat{k}$ be the integral optimum and $\bar{k}$ the fractional optimum (so $\bar{k} \leq \hat{k}$). We assume that the input graph G is not bipartite, that is, $\hat{k} \geq 1$. Then, for every vertex $u \in V$ the set H contains at most $\sqrt{n}+1$ edges adjacent to u. Moreover, for every vertex $u \in V$ the rounding procedure guarantees that the number of edges from F adjacent to u is at most $4\sqrt{n}\sum_{e\in\delta(u)} x_e \leq 4\sqrt{n}\bar{k}$. □

A simple example demonstrates that the bound on the approximation ratio of the algorithm is tight even for planar (and series-parallel) graphs. Think about a graph obtained from a cycle of length $2\sqrt{n}+1$ by replacing all edges but one by $\sqrt{n}/2$ internally vertex disjoint paths of length two. The minimum value of an OCT is one while the value of an OCT reported by the algorithm is $\sqrt{n}$.

We conclude this section with an observation that the integrality gap of the LP is large even for planar (and series-parallel) graphs with large minimum degree.

Theorem 2.3. *The integrality gap of the LP (1) for 2-connected planar (and series-parallel) graphs with minimum degree $\sqrt{n}+1$ is $\Omega(\sqrt{n})$.*

Proof. Let n be an integer such that $\sqrt{n}$ is an integer. Consider $\sqrt{n}$ vertices $v_1, v_2, \ldots, v_{\sqrt{n}}$. For each $i \in \{1, 2, \ldots, \sqrt{n}-1\}$, add $\sqrt{n}$ parallel edges between v_i and v_{i+1}, and subdivide each of these edges once. Thus, v_i and v_{i+1} are linked by $\sqrt{n}$ internally disjoint paths $P^i_1, \ldots, P^i_{\sqrt{n}}$ of length 2. Last, add an edge between $v_{\sqrt{n}}$ and v_i for every $i \in \{1, \ldots, \lceil\sqrt{n}/2\rceil\}$. Let $G = (V, E)$ be the obtained graph. Thus, G is a planar graph with n vertices and maximum degree $2\sqrt{n}+1$. Further, every odd cycle of G contains the vertex $v_{\sqrt{n}}$ and has length at least $2 \cdot \lfloor \sqrt{n}/2 \rfloor + 1$. Thus, setting $x_e = 1/\sqrt{n}$ for every edge e of G yields a feasible fractional solution with objective value $2 + \frac{1}{\sqrt{n}}$. However, the integral optimum is $\lceil \sqrt{n}/2 \rceil$. To see this, let F be an OCT of G. If all the edges $\{v_{\sqrt{n}}, v_i\}$ for $i \in \{1, \dots, \lceil \sqrt{n}/2 \rceil\}$ belong to F, then the degree of $v_{\sqrt{n}}$ in (V, F) is at least $\lceil \sqrt{n}/2 \rceil$, as wanted. So, assume that there exists $i \in \{1, \dots, \lceil \sqrt{n/2} \rceil\}$ such that $\{v_{\sqrt{n}}, v_i\} \notin F$. Then, observe that there must exist $j \in \{i, \dots, \sqrt{n}\}$ such that F contains at least one edge of each of the paths P_s^j. (For otherwise there would exist an odd cycle $v_i, y_i, v_{i+1}, y_{i+1}, \dots, v_{\sqrt{n}-1}, y_{\sqrt{n}-1}, v_{\sqrt{n}}$ in $(V, E \setminus F)$, where y_t belongs to P_s^t for some integer s.) Consequently, one of v_j and v_{j+1} has degree at least $\lceil \sqrt{n}/2 \rceil$ in the graph (V, F). This concludes the proof. □

Minimum Fair Cut Problem. The same approach yields a $\Theta(\sqrt{n})$ approximation algorithm for the minimum fair cut problem; the difference is that in the linear program (1) we replace the set $\mathcal{C}$ of all odd cycles by the set of all x-y-paths. For the lack of space we omit further details.

3. Exact Algorithms

Given a graph $G = (V, E)$ and a collection $\mathscr{H}$ of subgraphs of G, an $\mathscr{H}$-*transversal* (or a transversal of $\mathscr{H}$) is a set $F \subseteq E$ such that every subgraph $H \in \mathscr{H}$ contains an edge in F. Thus, an odd-cycle transversal of G is an $\mathscr{H}$-transversal where $\mathscr{H}$ is the collection of all odd cycles of G.

Our goal is to present polynomial-time algorithms to solve the minimum fair OCT and the minimum fair cut problems for bounded tree-width graphs. As an introduction to the method used, we first focus on series-parallel graphs (which form a subclass of the class of graphs with tree-width at most two). Then, we present the algorithm for graphs of tree-width at most τ, for any fixed integer τ.

Before doing so, let us note that the minimum value of an OCT problem is not bounded by any function of the tree-width. To this end, consider again the graph G described in proof of Theorem 2.3 and note that G is a series parallel graph (and thus, has tree-width at most two) and the minimum value of an OCT is $\sqrt{n}/2$. An analogous observation holds for the minimum fair x-y cut problem.

3.1. **Series-Parallel Graphs.**

A graph G with two dedicated and distinct nodes $s, t \in V$, the *source* and the *sink*, is *series-parallel* if and only if one of the following holds:

1 (Base case): G consists only of the nodes s, t and the edge $\{s, t\}$.
2 (Parallel decomposition): G can be obtained from two series-parallel graphs G_1 and G_2, with source-sink pairs s_1, t_1 and s_2, t_2, by taking the disjoint union of G_1 and G_2 and identifying s_1 with s_2 and t_1 with t_2, which gives the source s and sink t of G, respectively.
3 (Series decomposition): G is obtained analogously to the parallel composition from two series-parallel graphs G_1 and G_2, except that in this case t_1 is identified with s_2 and $s = s_1$, $t = t_2$.

When talking about a series-parallel graph G, we always assume that s and t are the source and sink nodes of G; the nodes s and t are also called *terminals* of G.

We define a binary operation $\oplus$ over family of collections of subsets of E by setting $\mathcal{A} \oplus \mathcal{B} = \{F' \cup F'' \mid F' \in \mathcal{A}, F'' \in \mathcal{B}\}$. The precedence of $\oplus$ is higher than the precedence of $\cup$.

A path is *odd* if it has an odd number of edges, and *even* otherwise. Apart from transversals for odd-cycles, we use transversals for odd and even s-t paths. In particular, for a series parallel graph G with terminals s and t, we let

- $\mathcal{T}_C(G)$ be the set of all odd-cycle transversals in G,
- $\mathcal{T}_O(G)$ be the set of all transversals of odd-cycles and odd s-t paths in G,
- $\mathcal{T}_E(G)$ be the set of all transversals of odd-cycles and even s-t paths in G,
- $\mathcal{T}_A(G)$ be the set of all transversals of odd-cycles and all s-t paths in G.

The definitions of the series and parallel compositions of two graphs yield recursive relations on the sets of transversals, as we observe in the next lemma.

Lemma 3.1. *For a graph G obtained by the parallel composition from G_1 and G_2, the following equalities hold.*

$$\begin{aligned}
\mathcal{T}_C(G) &= \mathcal{T}_A(G_1) \oplus \mathcal{T}_C(G_2) \ \cup\ \mathcal{T}_C(G_1) \oplus \mathcal{T}_A(G_2) \ \cup\ \mathcal{T}_O(G_1) \oplus \mathcal{T}_O(G_2) \ \cup\ \mathcal{T}_E(G_1) \oplus \mathcal{T}_E(G_2)\,,\\
\mathcal{T}_O(G) &= \mathcal{T}_O(G_1) \oplus \mathcal{T}_O(G_2)\,,\\
\mathcal{T}_E(G) &= \mathcal{T}_E(G_1) \oplus \mathcal{T}_E(G_2)\,,\\
\mathcal{T}_A(G) &= \mathcal{T}_A(G_1) \oplus \mathcal{T}_A(G_2)\,.
\end{aligned} \tag{2}$$

For a graph G obtained by the series composition from G_1 and G_2, the following equalities hold.

$$\begin{aligned}
\mathcal{T}_C(G) &= \mathcal{T}_C(G_1) \oplus \mathcal{T}_C(G_2)\,,\\
\mathcal{T}_O(G) &= \mathcal{T}_A(G_1) \oplus \mathcal{T}_C(G_2) \ \cup\ \mathcal{T}_C(G_1) \oplus \mathcal{T}_A(G_2) \ \cup\ \mathcal{T}_O(G_1) \oplus \mathcal{T}_O(G_2) \ \cup\ \mathcal{T}_E(G_1) \oplus \mathcal{T}_E(G_2)\,,\\
\mathcal{T}_E(G) &= \mathcal{T}_A(G_1) \oplus \mathcal{T}_C(G_2) \ \cup\ \mathcal{T}_C(G_1) \oplus \mathcal{T}_A(G_2) \ \cup\ \mathcal{T}_O(G_1) \oplus \mathcal{T}_E(G_2) \ \cup\ \mathcal{T}_E(G_1) \oplus \mathcal{T}_O(G_2)\,,\\
\mathcal{T}_A(G) &= \mathcal{T}_A(G_1) \oplus \mathcal{T}_C(G_2) \ \cup\ \mathcal{T}_C(G_1) \oplus \mathcal{T}_A(G_2)\,.
\end{aligned} \tag{3}$$

Proof. The equalities follow from the recursive structure of series-parallel graphs. Think about the first equality in (2). There are three types of odd cycles in G. Cycles contained in G_1, cycles contained in G_2 and cycles going through both G_1 and G_2. To destroy the odd cycles of the third type, we have to do destroy all s-t-paths in G_1, or all s-t-paths in G_2, or all odd s-t-paths in G_1 and all odd s-t-paths in G_2, or all even s-t-paths in G_1 and all even s-t-paths in G_2. The equality follows. An analogous reasoning yields the other equalities. □

We solve the minimum fair odd-cycle transversal using dynamic programming. We heavily exploit the recursive structure of series-parallel graphs which implies the recursive structure of the transversals captured in the previous lemma. To simplify the exposition, we introduce some additional notation. For a graph $G = (V, E)$, we let Δ be the maximum degree of G. Recall that $\deg_F(v)$ is the degree of $v \in V$ in the graph (V, F). Hence, the minimum fair odd-cycle transversal problem amounts to compute $\min\{\Delta_F \ :\ F \in \mathcal{T}_C(G)\}$, for a given graph G. Recall that the *value* of an odd-cycle transversal F is defined as Δ_F.

As tools for the dynamic programming we define, for a series parallel graph G, four matrices C, E, O, A, each of size $(\Delta+1) \times (\Delta+1)$. For $(i,j) \in \{0,1,\ldots,\Delta\}^2$,

- $C(i,j) = \min\{\Delta_F \ :\ F \in \mathcal{T}_C,\ \deg_F(s) \le i,\ \deg_F(t) \le j\}$,
- $O(i,j) = \min\{\Delta_F \ :\ F \in \mathcal{T}_O,\ \deg_F(s) \le i,\ \deg_F(t) \le j\}$,
- $E(i,j) = \min\{\Delta_F \ :\ F \in \mathcal{T}_E,\ \deg_F(s) \le i,\ \deg_F(t) \le j\}$,
- $A(i,j) = \min\{\Delta_F \ :\ F \in \mathcal{T}_A,\ \deg_F(s) \le i,\ \deg_F(t) \le j\}$,

we assume that the minimum over an empty set is ∞. In other words, $C(i,j)$ is the minimum value of an odd-cycle transversal that uses at most i edges adjacent to s and j edges adjacent to t; if no such transversal exists, then $C(i,j) = \infty$. The meaning of entries in other matrices is analogous. For notational convenience, we often use $C(G,i,j)$ to mean the entry $C(i,j)$ of the matrix associated with the graph G; analogous conventions are used for the matrices O, E and A, too.

Lemma 3.2. *Given a series-parallel graph G, the matrices C, O, E and A can be computed in polynomial time.*

Proof. Exploiting Lemma 3.1, we compute the entries recursively. To give an example, let G be a series-parallel graph obtained by the parallel composition of G_1 and G_2, where G_i has source and sink s_i and t_i, respectively. Then,

$$\begin{aligned} C(G,i,j) = \min\{\Delta_{F_1\cup F_2} \;:\; (F_1,F_2) \;&\in \mathcal{T}_A(G_1)\times\mathcal{T}_C(G_2) \;\cup\; \mathcal{T}_C(G_1)\times\mathcal{T}_A(G_2) \\ &\cup \mathcal{T}_O(G_1)\times\mathcal{T}_O(G_2) \;\cup\; \mathcal{T}_E(G_1)\times\mathcal{T}_E(G_2), \\ &\deg_{F_1}(s_1)+\deg_{F_2}(s_2)\le i, \deg_{F_1}(t_1)+\deg_{F_2}(t_2)\le j\}\,. \end{aligned}$$

Note that with matrices C, O, E and A for the graphs G_1 and G_2, the formula (4) provides an efficient way of computing the value $C(G,i,j)$ (in time $O(i^2j^2)$). Since calculating the values for the simplest series-parallel graph (i.e., two nodes connected by an edge) is trivial, it is possible to compute the matrices C, O, E and A for the given series-parallel graph G in polynomial time. □

As $C(G,\Delta,\Delta)$ is the value of a minimum fair odd-cycle transversal in G, we obtain the following theorem.

Theorem 3.3. *The minimum fair odd-cycle transversal in series-parallel graphs can be computed in polynomial time.*

Minimum Fair Cut Problem. An algorithm solving the minimum fair cut problem on series parallel graphs can be obtained by a similar approach. Let $G = (V,E)$ be a series-parallel graph with two terminal nodes s and t and let $x \neq y$ be two nodes from V. For $u, v \in V$, we define $\mathcal{P}_{u,v}$ to be the set of all paths between u and v. We work with a number of different transversals.

- $\mathcal{T}(G,x,y)$ is the set of all transversals of $\mathcal{P}_{x,y}$,
- $\mathcal{T}_a(G,x,y)$ is the set of all transversals of $\mathcal{P}_{x,y}\cup\mathcal{P}_{x,s}\cup\mathcal{P}_{y,s}\cup\mathcal{P}_{x,t}\cup\mathcal{P}_{y,t}$,
- $\mathcal{T}_s(G,x,y)$ is the set of all transversals of $\mathcal{P}_{x,y}\cup\mathcal{P}_{x,s}\cup\mathcal{P}_{y,s}$,
- $\mathcal{T}_t(G,x,y)$ is the set of all transversals of $\mathcal{P}_{x,y}\cup\mathcal{P}_{x,t}\cup\mathcal{P}_{y,t}$,
- $\mathcal{T}_m(G,x,y)$ is the set of all transversals of $\mathcal{P}_{x,y}\cup\mathcal{P}_{x,s}\cup\mathcal{P}_{y,t}$,
- $\mathcal{T}_b(G,x,y)$ is the set of all transversals of $\mathcal{P}_{x,y}\cup\mathcal{P}_{x,s}$,
- $\mathcal{T}_c(G,x,y)$ is the set of all transversals of $\mathcal{P}_{x,y}\cup\mathcal{P}_{x,t}$,
- $\mathcal{T}_d(G,x,y)$ is the set of all transversals of $\mathcal{P}_{x,y}\cup\mathcal{P}_{x,s}\cup\mathcal{P}_{x,t}$,
- $\mathcal{T}_a(G,x)$ is the set of all transversals of $\mathcal{P}_{x,s}\cup\mathcal{P}_{x,t}$,
- $\mathcal{T}_s(G,x)$ is the set of all transversals of $\mathcal{P}_{x,s}$,
- $\mathcal{T}_t(G,x)$ is the set of all transversals of $\mathcal{P}_{x,t}$,
- $\mathcal{T}_b(G,x)$ is the set of all transversals of $\mathcal{P}_{x,s}\cup\mathcal{P}_{x,t}\cup\mathcal{P}_{s,t}$,
- $\mathcal{T}_c(G,x)$ is the set of all transversals of $\mathcal{P}_{x,s}\cup\mathcal{P}_{s,t}$,
- $\mathcal{T}_d(G,x)$ is the set of all transversals of $\mathcal{P}_{x,t}\cup\mathcal{P}_{s,t}$,
- $\mathcal{T}(G)$ is the set of all transversals of $\mathcal{P}_{s,t}$.

For the forthcoming exposition, it is convenient to distinguish four cases, according to the structure of G and the location of x and y in G.

Case 1. The graph $G = (V, E)$ is obtained by the parallel composition of graphs $G_1 = (V_1, E_1)$, $G_2 = (V_2, E_2)$, and $x \in V_1$ and $y \in V_2$.
Case 2. The graph $G = (V, E)$ is obtained by the parallel composition of graphs $G_1 = (V_1, E_1)$, $G_2 = (V_2, E_2)$, and $x, y \in V_1$.
Case 3. The graph $G = (V, E)$ is obtained by the series composition of graphs $G_1 = (V_1, E_1)$, $G_2 = (V_2, E_2)$, and $x \in V_1$ and $y \in V_2$.
Case 4. The graph $G = (V, E)$ is obtained by the series composition of graphs $G_1 = (V_1, E_1)$, $G_2 = (V_2, E_2)$, and $x, y \in V_1$.

Without loss of generality we assume that these four cases cover all possible combinations of the structure of G and the location of x and y in G. In a similar way as for the minimum fair odd-cycles transversal problem, for each of the four cases, a set of equations describes the relations between the transversals in question. The equations follow from a careful case-distinction and elementary arguments; we omit the technical details.

Lemma 3.4. *The following equations hold.*
For cases 1 and 2.

$$\begin{aligned}
\mathcal{T}_a(G,x) &= \mathcal{T}_a(G_1,x),\\
\mathcal{T}_s(G,x) &= \mathcal{T}_a(G_1,x) \ \cup\ \mathcal{T}_s(G_1,x) \oplus \mathcal{T}(G_2),\\
\mathcal{T}_t(G,x) &= \mathcal{T}_a(G_1,x) \ \cup\ \mathcal{T}_t(G_1,x) \oplus \mathcal{T}(G_2),\\
\mathcal{T}_b(G,x) &= \mathcal{T}_b(G_1,x) \oplus \mathcal{T}(G_2),\\
\mathcal{T}_c(G,x) &= \mathcal{T}_c(G_1,x) \oplus \mathcal{T}(G_2),\\
\mathcal{T}_d(G,x) &= \mathcal{T}_d(G_1,x) \oplus \mathcal{T}(G_2),\\
\mathcal{T}(G) &= \mathcal{T}(G_1) \oplus \mathcal{T}(G_2).
\end{aligned}$$

For case 1.

$$\begin{aligned}
\mathcal{T}(G,x,y) &= \mathcal{T}_a(G_1,x) \ \cup\ \mathcal{T}_a(G_2,y) \ \cup\ \mathcal{T}_s(G_1,x) \oplus \mathcal{T}_t(G_2,y) \ \cup\ \mathcal{T}_t(G_1,x) \oplus \mathcal{T}_s(G_2,y),\\
\mathcal{T}_a(G,x,y) &= \mathcal{T}_a(G_1,x) \oplus \mathcal{T}_a(G_2,y),\\
\mathcal{T}_s(G,x,y) &= \mathcal{T}_s(G_1,x) \oplus \mathcal{T}_a(G_2,y) \ \cup\ \mathcal{T}_a(G_1,x) \oplus \mathcal{T}_t(G_2,y),\\
\mathcal{T}_t(G,x,y) &= \mathcal{T}_t(G_1,x) \oplus \mathcal{T}_a(G_2,y) \ \cup\ \mathcal{T}_a(G_1,x) \oplus \mathcal{T}_t(G_2,y),\\
\mathcal{T}_m(G,x,y) &= \mathcal{T}_s(G_1,x) \oplus \mathcal{T}_t(G_2,y),\\
\mathcal{T}_b(G,x,y) &= \mathcal{T}_a(G_1,x) \ \cup\ (\mathcal{T}_s(G_1,x) \oplus \mathcal{T}_t(G_2,y),\\
\mathcal{T}_c(G,x,y) &= \mathcal{T}_a(G_1,x) \ \cup\ (\mathcal{T}_t(G_1,x) \oplus \mathcal{T}_s(G_2,y),\\
\mathcal{T}_d(G,x,y) &= \mathcal{T}_a(G_1,x).
\end{aligned}$$

For case 2.

$$\begin{aligned}
\mathcal{T}(G,x,y) &= \mathcal{T}(G_1,x,y) \oplus \mathcal{T}(G_2) \ \cup\ \mathcal{T}_s(G_1,x,y) \ \cup\ \mathcal{T}_t(G_1,x,y) \ \cup\ \mathcal{T}_m(G_1,x,y) \ \cup\ \mathcal{T}_n(G_1,x,y),\\
\mathcal{T}_a(G,x,y) &= \mathcal{T}_a(G_1,x,y),\\
\mathcal{T}_s(G,x,y) &= \mathcal{T}_s(G_1,x,y) \oplus \mathcal{T}(G_2) \ \cup\ \mathcal{T}_a(G_1,x,y),\\
\mathcal{T}_t(G,x,y) &= \mathcal{T}_t(G_1,x,y) \oplus \mathcal{T}(G_2), \mathcal{T}_a(G_1,x,y),\\
\mathcal{T}_m(G,x,y) &= \mathcal{T}_m(G_1,x,y) \oplus \mathcal{T}(G_2), \mathcal{T}_a(G_1,x,y),\\
\mathcal{T}_b(G,x,y) &= \mathcal{T}_b(G_1,x,y) \oplus \mathcal{T}(G_2), \mathcal{T}_d(G_1,x,y),\\
\mathcal{T}_c(G,x,y) &= \mathcal{T}_c(G_1,x,y) \oplus \mathcal{T}(G_2), \mathcal{T}_d(G_1,y,x),\\
\mathcal{T}_d(G,x,y) &= \mathcal{T}_d(G_1,x,y).
\end{aligned}$$

For cases 3 and 4.

$$
\begin{aligned}
\mathcal{T}_a(G,x) &= \mathcal{T}_a(G_1,x) \ \cup\ \mathcal{T}_s(G_1,x) \oplus \mathcal{T}(G_2),\\
\mathcal{T}_s(G,x) &= \mathcal{T}_s(G_1,x,y),\\
\mathcal{T}_t(G,x) &= \mathcal{T}_t(G_1,x) \ \cup\ \mathcal{T}(G_2),\\
\mathcal{T}_b(G,x) &= \mathcal{T}_b(G_1,x) \ \cup\ \mathcal{T}_s(G_1,x) \oplus \mathcal{T}(G_2),\\
\mathcal{T}_c(G,x) &= \mathcal{T}_c(G_1,x) \ \cup\ \mathcal{T}_s(G_1,x) \oplus \mathcal{T}(G_2),\\
\mathcal{T}_d(G,x) &= \mathcal{T}_d(G_1,x) \ \cup\ \mathcal{T}_t(G_1,x) \oplus \mathcal{T}(G_2),\\
\mathcal{T}(G) &= \mathcal{T}(G_1) \ \cup\ \mathcal{T}(G_2).
\end{aligned}
$$

For case 3.

$$
\begin{aligned}
\mathcal{T}(G,x,y) &= \mathcal{T}_t(G_1,x) \ \cup\ \mathcal{T}_s(G_2,y),\\
\mathcal{T}_a(G,x,y) &= \mathcal{T}_b(G_1,x) \oplus \mathcal{T}_t(G_2,y) \ \cup\ \mathcal{T}_s(G_1,x) \oplus \mathcal{T}_b(G_2,y) \cup\\
&\quad\ \mathcal{T}_c(G_1,x) \oplus \mathcal{T}_d(G_2,y) \ \cup\ \mathcal{T}_a(G_1,x) \oplus \mathcal{T}_a(G_2,y),\\
\mathcal{T}_s(G,x,y) &= \mathcal{T}_s(G_1,x) \oplus \mathcal{T}_s(G_2,y) \ \cup\ \mathcal{T}_c(G_1,x),\\
\mathcal{T}_t(G,x,y) &= \mathcal{T}_t(G_1,x) \oplus \mathcal{T}_t(G_2,y) \ \cup\ \mathcal{T}_d(G_2,y),\\
\\
\mathcal{T}_m(G,x,y) &= \mathcal{T}_a(G_1,x) \oplus \mathcal{T}_t(G_2,y) \ \cup\ \mathcal{T}_s(G_1,x) \oplus \mathcal{T}_a(G_2,y),\\
\mathcal{T}_b(G,x,y) &= \mathcal{T}_a(G_1,x) \ \cup\ \mathcal{T}_s(G_1,x) \oplus \mathcal{T}_s(G_2,y),\\
\mathcal{T}_c(G,x,y) &= \mathcal{T}_t(G_1,x) \ \cup\ \mathcal{T}_c(G_2,y),\\
\mathcal{T}_d(G,x,y) &= \mathcal{T}_a(G_1,x) \ \cup\ \mathcal{T}_s(G_1,x) \oplus \mathcal{T}_c(G_2,y).
\end{aligned}
$$

For case 4.

$$
\begin{aligned}
\mathcal{T}(G,x,y) &= \mathcal{T}(G_1,x,y),\\
\mathcal{T}_a(G,x,y) &= \mathcal{T}_a(G_1,x,y) \ \cup\ \mathcal{T}_s(G_1,x,y) \oplus \mathcal{T}(G_2),\\
\mathcal{T}_s(G,x,y) &= \mathcal{T}_s(G_1,x),\\
\mathcal{T}_t(G,x,y) &= \mathcal{T}_t(G_1,x,y) \ \cup\ \mathcal{T}(G_1,x,y) \oplus \mathcal{T}(G_2),\\
\mathcal{T}_m(G,x,y) &= \mathcal{T}_m(G_1,x,y) \ \cup\ \mathcal{T}_c(G_1,x,y) \oplus \mathcal{T}(G_2),\\
\mathcal{T}_b(G,x,y) &= \mathcal{T}_b(G_1,x,y),\\
\mathcal{T}_c(G,x,y) &= \mathcal{T}_c(G_1,x,y) \ \cup\ \mathcal{T}(G_1,x,y) \oplus \mathcal{T}(G_2),\\
\mathcal{T}_d(G,x,y) &= \mathcal{T}_d(G_1,x,y) \ \cup\ \mathcal{T}_b(G_1,x) \oplus \mathcal{T}(G_2).
\end{aligned}
$$

□

Analogously to the odd-cycle transversal problem, with a series-parallel graph G and each of the set of transversals in question, we associate a square matrix of size $(\Delta+1)\times(\Delta+1)$. For a series-parallel graph G with source s and sink t and set of transversals $\mathcal{T}'$, we define the entry (i,j) of the matrix $\mathcal{M}_{\mathcal{T}'}$ as $\mathcal{M}_{\mathcal{T}'}(G,i,j) = \min\{\Delta_F \ :\ F \in \mathcal{T}',\ \deg_F(s) \le i,\ \deg_F(t) \le j\}$. The proof of the following lemma goes along the same lines as the proof of Lemma 3.2. As the entry $\mathcal{M}_{\mathcal{T}(G)}(G,\Delta,\Delta)$ contains the value of the minimum fair x-y cut in G, we immediately obtain the main theorem.

Lemma 3.5. *Given a series-parallel graph G, the matrices associated with the sets of transversals can be computed in polynomial time.* □

Theorem 3.6. *Given a series-parallel graph with two distinguished nodes x and y, the minimum fair x-y-cut can be computed in polynomial time.* □

3.2. **Graphs with Bounded Tree-Width.** The method described in this section exploits a special type of tree decomposition, namely a nice tree decomposition. Let us just recall that in a nice tree decomposition there are four types of nodes: leaf, join, introduce and forget nodes. The leaf nodes are leaves in the tree, the join nodes are inner nodes with two children and the introduce and forget nodes are inner nodes with one child. We refer to the survey by Bodlaender and Koster [7] for further exposition on this topic.

Given a graph $G = (V, E)$, assume that a subset $S = \{v_1, \dots, v_s\} \subseteq V$ is a vertex separator of G; the bags from a nice tree decomposition of G will be used as the vertex separators. We let (V_L, V_R) be a partition of $V \setminus S$ such that there are no edges between V_L and V_R in G; the set V_L is called the *left part* of G and V_R the *right part* of G. Let $G_L = G[V_L \cup S]$ and $G_R = G[V_R \cup S]$, let $E_S = E \cap S \times S$. For every two distinct vertices $x, y \in S$, we define

- $P^{G,O}_{x,y}$ to be the set of all odd x-y-paths in G,
- $P^{G,E}_{x,y}$ to be the set of all even x-y-paths in G,
- C^G to be the set of all odd cycles in G.

Then we define a collection of sets

$$\mathcal{S}^G = \{C^G\} \cup \bigcup_{x \neq y \in S} \{P^{G,O}_{x,y}, P^{G,E}_{x,y}\} .$$

For every $\mathcal{R} \subseteq \mathcal{S}^G$, we let $\mathcal{T}^G_{\mathcal{R}}$ be the set of all transversals of the paths and cycles in $\bigcup_{P \in \mathcal{R}} P$ in the graph G.

Lemma 3.7. *Let $G = (V, E)$ be a graph and $S = \{v_1, \dots, v_s\} \subseteq V$ a vertex separator of G. For every $\mathcal{R} \subseteq \mathcal{S}^G$, there exist sets $\mathcal{R}_1, \dots, \mathcal{R}_k \subseteq \mathcal{S}^{G_L}$ and sets $\mathcal{R}'_1, \dots, \mathcal{R}'_k \subseteq \mathcal{S}^{G_R}$ such that*

$$\mathcal{T}^G_{\mathcal{R}} = \bigcup_{i=1}^{k} \mathcal{T}_{\mathcal{R}_i} \oplus \mathcal{T}_{\mathcal{R}'_i} .$$

□

The proof is similar to the proofs of Lemma 3.1 and Lemma 3.4 for series-parallel graphs; we omit it.

As in the previous section, a polynomial algorithm does not have time to examine all transversals of C^G or the other sets. Instead, for each set of transversals $\mathcal{T}^G_{\mathcal{R}}$ we store some information about the most relevant transversals. For every subset $\mathcal{R} \subseteq \mathcal{S}^G$, every s-tuple $(d_1, d_2, \dots, d_s) \in \{0, 1, \dots, \Delta\}^s$ and every $E'_S \subseteq E_S$, we set

$$\mathcal{M}^G_{\mathcal{R}}(d_1, d_2, \dots, d_s, E'_S) = \min\{\Delta_F \,:\, F \in \mathcal{T}^G_{\mathcal{R}},\ F \cap S \times S = E'_S,\ \forall 1 \le i \le s,\ \delta_F(v_i) \le d_i\} .$$

In words, $\mathcal{M}^G_{\mathcal{R}}(d_1, d_2, \dots, d_s, E'_S)$ is the minimum value of a transversal of $\bigcup_{P \in \mathcal{R}} P$ with the property that on S it coincides with E'_S, and for each $v_i \in S$, the transversal uses at most d_i edges adjacent to v_i. We let $\mathcal{M}^G_{\mathcal{R}}$ be the whole $(s+1)$-dimensional matrix; in total there are $2^{|\mathcal{S}^G|}$ such matrices associated with the graph G and the separator S, each of size $(\Delta + 1)^s \cdot 2^{|E_S|}$.

The next lemma follows from Lemma 3.7. The key observation is that the size of the set $\mathcal{S}^G$ is bounded by $O(2^\tau)$.

Lemma 3.8. *Assume that we have a nice tree decomposition T of a graph G with tree-width $\tau = O(1)$. Let i be a join node in T with a bag S and with children G_L and G_R, and assume that we have the matrices $\mathcal{M}^{G_L}_{\mathcal{R}'}$ and $\mathcal{M}^{G_R}_{\mathcal{R}''}$ for every subset*

$\mathcal{R}' \subseteq \mathcal{S}^{G_L}$ and $\mathcal{R}'' \subseteq \mathcal{S}^{G_R}$. Then, it is possible to calculate all the matrices $\mathcal{M}^G_{\mathcal{R}}$ for $\mathcal{R} \subseteq \mathcal{S}^G$, in polynomial time. □

Analogous statements hold for all other types of nodes (i.e., leaf node, introduce node and forget node) in the nice tree decomposition of G, yielding the main theorem.

Theorem 3.9. *The minimum fair odd-cycle transversal problem is solvable in polynomial time on graphs with bounded tree-width.*

Proof. By the definition of the matrix $\mathcal{M}^G_{C^G}$, the minimum value of an OCT in G is equal the minimum

$$\min_{E'_S \subseteq E_S} \mathcal{M}^G_{C^G}(\Delta, \ldots, \Delta, E'_S)\,.$$

The time bound follows from Lemma 3.8 and the fact that the nice decomposition tree of G has size $O(|V|)$. □

Applying the same method for the minimum fair cut problem, we obtain the following theorem.

Theorem 3.10. *The minimum fair cut problem is solvable in polynomial time on graphs with bounded tree-width.* □

Open Problems and Future Work

As already explained in the Introduction, a challenging open problem is to formally define a class of polynomial-time problems, as large as possible, that contains the minimum fair OCT. We anticipate that another extension of MSOL (or of the other approaches) will encompass these problems. Note that the problem itself (finding a bipartite subgraph) is expressible in MSOL, and what makes the problem hard is the objective function.

Finding better approximation algorithms on planar and general graphs for the two problems considered in this paper, as well as for other fair edge deletion problems, is also a challenging task.

References

[1] A. Agarwal, M. Charikar, K. Makarychev, and Y. Makarychev. $O(\sqrt{\log n})$ approximation algorithms for min UnCut, min 2CNF deletion, and directed cut problems. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing (STOC)*, pages 573–581, 2005.

[2] N. Alon, A. Shapira, and B. Sudakov. Additive approximation for edge-deletion problems. In *Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 419–428, 2005.

[3] N. Alon, A. Shapira, and B. Sudakov. Additive approximation for edge-deletion problems. *Ann. Math.*, 170:371–411, 2009.

[4] S. Arnborg, J. Lagergren, and D. Seese. Problems easy for tree-decomposable graphs. *Journal of Algorithms*, 12:308–340, 1991.

[5] T. Asano. An application of duality to edge-deletion problems. *SIAM Journal on Computing*, 16(2):312–331, 1987.

[6] H. L. Bodlaender and K. Jansen. On the complexity of the maximum cut problem. In *Nordic Journal of Computing*, volume 7. 2000.

[7] H. L. Bodlaender and A. M. C. A. Koster. Combinatorial optimization on graphs of bounded treewidth. *Computer Journal*, 51(3):255–269, 2008.

[8] P. S. Bonsma. The complexity of the matching-cut problem for various graph classes. *Electronic Notes in Discrete Mathematics*, 13:18–21, 2003.

[9] R. B. Borie. Generation of polynomial-time algorithms for some optimization problems on tree-decomposable graphs. *Algorithmica*, 14(2):123–137, 1995.
[10] R. B. Borie, R. G. Parker, and C. A. Tovey. Automatic generation of linear-time algorithms from predicate calculus descriptions of problems on recursively constructed graph families. *Algorithmica*, 7:555–581, 1992.
[11] V. Chvátal. Recognizing decomposable graphs. *J. Graph Theory*, 8(1):51–53, 1984.
[12] R. Corrêa, F. Havet, and J.-S. Sereni. About a Brooks-type theorem for improper colouring. *Australas. J. Combin.*, 43:219–230, 2009.
[13] B. Courcelle. The monadic second-order theory of graphs. I. recognizable sets of finite graphs. *Information and Computation*, 85(1):12–75, 1990.
[14] B. Courcelle and M. Mosbah. Monadic second-order evaluations on tree-decomposable graphs. *Theoretical Computer Science*, 109(1–2):49–82, 1 Mar. 1993.
[15] L. J. Cowen, R. H. Cowen, and D. R. Woodall. Defective colorings of graphs in surfaces: partitions into subgraphs of bounded valency. *J. Graph Theory*, 10(2):187–195, 1986.
[16] L. J. Cowen, W. Goddard, and C. E. Jesurum. Defective coloring revisited. *J. Graph Theory*, 24(3):205–219, 1997.
[17] M. Garey, D. S. Johnson, and L. J. Stockmeyer. Some simplified NP-complete graph problems. *Theoretical Computer Science*, 1:237–267, 1976.
[18] M. X. Goemans and D. P. Williamson. .879-approximation algorithms for MAX CUT and MAX 2SAT. In *Proceedings of the 26th Annual ACM Symposium on the Theory of Computing (STOC)*, pages 422–431, 1994.
[19] M. Groetschel and W. R. Pulleyblank. Weakly bipartite graphs and the max-cut problem. *Operations Research Letters*, 1(1):23 – 27, 1981.
[20] F. Havet, R. J. Kang, and J.-S. Sereni. Improper colouring of unit disk graphs. *Networks*, 54(3):150–164, 2009.
[21] F. Havet and J.-S. Sereni. Channel assignment and improper choosability of graphs. In *Graph-theoretic concepts in computer science*, volume 3787 of *Lecture Notes in Comput. Sci.*, pages 81–90. Springer, Berlin, 2005.
[22] F. Havet and J.-S. Sereni. Improper choosability of graphs and maximum average degree. *J. Graph Theory*, 52(3):181–199, 2006.
[23] D. S. Johnson. The NP-completeness column: An ongoing guide. *Journal of Algorithms*, 6(1):145–159, 1985.
[24] R. J. Kang, T. Müller, and J.-S. Sereni. Improper colouring of (random) unit disk graphs. *Discrete Math.*, 308(8):1438–1454, 2008.
[25] J. Komlós. Covering odd cycles. *Combinatorica*, 17(3):393–400, 1997.
[26] L.-S. Lin and S. Sahni. Fair edge deletion problems. *IEEE Trans. Computers*, 38(5):756–761, 1989.
[27] M. Patrignani and M. Pizzonia. The complexity of the matching-cut problem. In *Graph-Theoretic Concepts in Computer Science*, volume 2204 of *Lecture Notes in Computer Science*, pages 284–295, 2001.
[28] R. Škrekovski. List improper colourings of planar graphs. *Combin. Probab. Comput.*, 8(3):293–299, 1999.
[29] R. Škrekovski. List improper colorings of planar graphs with prescribed girth. *Discrete Math.*, 214(1-3):221–233, 2000.
[30] D. R. Woodall. Defective choosability of graphs with no edge-plus-independent-set minor. *J. Graph Theory*, 45(1):51–56, 2004.
[31] M. Yannakakis. Edge-deletion problems. *SIAM Journal on Computing*, 10(2):297–309, 1981.

E-mail address: `kolman,lidicky,sereni@kam.mff.cuni.cz`