A Parameterized Algorithm for Mixed-Cut

Ashutosh Rai[1(✉)], M.S. Ramanujan[2], and Saket Saurabh[1,3]

[1] The Institute of Mathematical Sciences, Chennai, India
{ashutosh,saket}@imsc.res.in
[2] Vienna Institute of Technology, Vienna, Austria
ramanujan@ac.tuwien.ac.at
[3] University of Bergen, Bergen, Norway

Abstract. The classical Menger's theorem states that in any undirected (or directed) graph G, given a pair of vertices s and t, the maximum number of vertex (edge) disjoint paths is equal to the minimum number of vertices (edges) needed to disconnect s from t. This min-max result can be turned into a polynomial time algorithm to find the maximum number of vertex (edge) disjoint paths as well as the minimum number of vertices (edges) needed to disconnect s from t. In this paper we study a mixed version of this problem, called Mixed-Cut, where we are given an undirected graph G, vertices s and t, positive integers k and l and the objective is to test whether there exist a k sized vertex set $S \subseteq V(G)$ and an l sized edge set $F \subseteq E(G)$ such that deletion of S and F from G disconnects from s and t. Apart from studying a generalization of classical problem, one of our main motivations for studying this problem comes from the fact that this problem naturally arises as a subproblem in the study of several graph editing (modification) problems. We start with a small observation that this problem is NP-complete and then study this problem, in fact a much stronger generalization of this, in the realm of parameterized complexity. In particular we study the Mixed Multiway Cut-Uncut problem where along with a set of terminals T, we are also given an equivalence relation $\mathcal{R}$ on T, and the question is whether we can delete at most k vertices and at most l edges such that connectivity of the terminals in the resulting graph respects $\mathcal{R}$. Our main result is a fixed parameter algorithm for Mixed Multiway Cut-Uncut using the method of recursive understanding introduced by Chitnis et al. (FOCS 2012).

1 Introduction

Given a graph, a typical *cut problem* asks for finding a set of vertices or edges such that their removal from the graph makes the graph satisfy some separation property. The most fundamental version of the cut problems is Minimum Cut, where given a graph and two vertices, called *terminals*, we are asked to find

The research leading to these results has received funding from the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013)/ERC grant agreement no. 306992.

E. Kranakis et al. (Eds.): LATIN 2016, LNCS 9644, pp. 672–685, 2016.
DOI: 10.1007/978-3-662-49529-2_50

the minimum sized subset of vertices (or edges) of the graph such that deleting them separates the terminals. The Minimum Cut problem is known to be polynomial time solvable for both edge and vertex versions and both in undirected and directed graphs. The core of the polynomial time solvability of the Minimum Cut problem is one of the classical min-max results in graph theory – the Menger's theorem. Menger's theorem states that in any undirected (or directed) graph G, given a pair of vertices s and t, the maximum number of vertex (edge) disjoint paths is equal to the minimum number of vertices (edges) needed to be deleted to disconnect s from t.

While Minimum Cut is polynomial time solvable; even a slight generalization becomes NP-hard. Two of the most studied generalizations of Minimum Cut problem which are NP-hard are Multiway Cut and Multicut. In the Multiway Cut problem, we are given a set of terminals, and we are asked to delete minimum number of vertices (or edges) to separate the terminals from each other. This problem is known to be NP-hard even when the number of terminals is at least three. In the Multicut problem, given pairs of terminals, we are asked to delete minimum number of vertices (or edges) so that it separates all the given terminal pairs. The Multicut problem is known to be NP-hard when the number of pairs of terminals is at least three. The mixed version of the problem, which is the central topic of this paper, namely Mixed Cut is also NP-hard. In this problem we are given an undirected graph G, vertices s and t, positive integers k and l and the objective is to test whether there exist a k sized vertex set $S \subseteq V(G)$ and an l sized edge set $F \subseteq E(G)$ such that deletion of S and F from G disconnects from s and t. In this paper we study Mixed Cut, in fact a stronger generalization of it, in the realm of parameterized complexity.

The field of parameterized complexity tries to provide efficient algorithms for NP-complete problems by going from the classical view of single-variate measure of the running time to a multi-variate one. It aims at getting algorithms of running time $f(k)n^{\mathcal{O}(1)}$, where k is an integer measuring some aspect of the problem. These algorithms are called fixed parameter tractable (FPT) algorithms and the integer k is called the *parameter*. In most of the cases, the solution size is taken to be the parameter, which means that this approach gives faster algorithms when the solution is of small size. For more background on parameterized complexity, the reader is referred to the monographs [9,10,18]. In this paper we study the problem called Mixed Multiway Cut-Uncut (MMCU) where given a graph G, $T \subseteq V(G)$, and equivalence relation $\mathcal{R}$ on T and integers k and l, we are asked whether there exist $X \subseteq (V(G) \setminus T)$ and $F \subseteq E(G)$ such that $|X| \leq k$, $|F| \leq l$ and for all $u, v \in T$, u and v belong to the same connected component of $G - (X, F)$ if and only if $(u, v) \in \mathcal{R}$. We start by giving a brief overview of related work and then give our results and methods.

Related Works. Multiway Cut was one of the first cut problems to be explored under the realm of parameterized complexity. It was known to be FPT using graph minors, but Marx [15] was the first one to give a constructive algorithm to show that Multiway Cut is FPT when parameterized by the solution size. He also showed that Multicut is FPT when parameterized by the solution

size plus the number of terminals. Subsequently, a lot of work has been done on cut problems in the field of parameterized complexity [3,5,8,13,14,16,17]. Recently, Chitnis et al. [7] introduced the technique of *randomized contractions* and used that to solve the Unique Label Cover problem. They also show that the same techniques can be applied to solve a generalization of Multiway Cut problem, namely Multiway Cut-Uncut, where an equivalence relation $\mathcal{R}$ is also supplied along with the set of terminals and we are to delete minimum number of vertices (or edges) such that the terminals lie in the same connected of the resulting graph if and only if they lie in the same equivalence class of $\mathcal{R}$. The Multiway Cut-Uncut problem was first shown to FPT by Marx et al. [16]. It is easy to see that MMCU not only generalized Mixed Cut and Mixed Multiway Cut, but also both edge and vertex versions of Multiway Cut and Multiway Cut-Uncut problems. Mixed Cut is studied and mentioned in the books [2,11] and is also a useful subroutine in parameterized graph editing problems [4]. Cao and Marx [4] studied this problem during their study on Chordal Editing problem and gave an algorithm with running time $2^{\mathcal{O}(k+l)}n^{\mathcal{O}(1)}$ on chordal graphs. Algorithms for cut-problems can be applied to several problems, which at first do not look like cut problems. Examples include well studied problems such as Feedback Vertex Set [6] and Odd Cycle Transversal [19]. Thus, MMCU is not only an interesting combinatorial problem in itself but it is also useful in designing other parameterized algorithms (for example in editing problems, *cf.* [4]). Hence, it is natural and timely to obtain a parameterized algorithms for MMCU.

Our Results and Methods. Even though the vertex and edge versions of Minimum Cut problem are polynomial time solvable, we show that allowing deletion of both, the vertices and the edges, makes the Mixed Cut problem NP-hard. To show that, we use a simple reduction from the Bipartite Partial Vertex Cover problem which was recently shown to be NP-hard [1,12]. Then we show that MMCU is FPT when parameterized by $k + l$. In particular we prove the following theorem.

Theorem 1. Mixed Multiway Cut-Uncut *is FPT with an algorithm running in time* $2^{(k+l)^{\mathcal{O}(1)}} \cdot n^{\mathcal{O}(1)}$.

There are two ways to approach our problem – one is via treewidth reduction technique of Marx et al. [16] and the second is via the method of recursive understanding introduced by Chitnis et al. [7]. However, the method of treewidth reduction technique would lead to an algorithm for MMCU that has double exponential dependence on $k + l$ and thus we did not pursue this method. We use recursive understanding introduced by Chitnis et al. [7] to solve the problem. The main observation is that if there is a small vertex separation which divides the graph into big parts, then we can recursively reduce the size of one of the big parts. Otherwise, the graph is highly connected, and the structure of the graph can be exploited to obtain a solution. We follow the framework given in [7] and design our algorithm. In particular we utilise the recursive understanding technique to first find a small separator in the graph which separates the graph

into two parts, each of sufficiently large size and then recursively solve a 'border' version of the same problem on one of the two sides. The border version of the problem is a generalization which also incorporates a special bounded set of vertices, called *border terminals*. During the course of our algorithm, we will attempt to solve the border problem on various subgraphs of the input graph. The objective in the border problem is to find a bounded set of vertices contained within a particular subgraph such that any vertex in this subgraph *not* in the computed set is not required in *any* solution for the given instance irrespective of the vertices chosen outside this subgraph. The algorithm in [7] returns the minimum solutions in the recursive steps. Since we allow both edge and vertex deletion, there is no clear ordering on the solutions, and hence we need to look for solutions of all possible sizes while making the recursive call.

This leaves us with the base case of the recursion, that is when we are unable to find a separator of the required kind. This is called high connectivity phase and this is the place where one needs a problem specific algorithm in the framework given in [7]. Since the solution we are looking for contains both edges and vertices, we need some additional work, as the good node separation framework gives bound only for vertices that can be part of the solution. Once we have done that, the frameworks lends itself for our use, and we can use a separating set family to get to the solution. This results in an extra factor of $k + l$ in the exponent as compared to the algorithm in [7], as the separating family needs to take care of both vertices and edges.

2 Preliminaries

In this section, we first give the notations and definitions which are used in the paper. Then we state some basic properties of mixed-cuts and some known results which will be used later in the paper.

Notations and Definitions: A tuple $G = (V, E)$ is a multigraph if V is a set (called *vertices*) and E is a multiset of 2-element subsets of V (called *edges*). For a multigraph G, we denote the set of vertices of the multigraph by $V(G)$ and the set of edges of the multigraph by $E(G)$. In slight abuse of terminology, we will be calling multigraphs also as graphs in the rest of the paper. We denote $|V(G)|$ and $|E(G)|$ by n and m respectively, where the graph is clear from context. For a set $S \subseteq V(G)$, the *subgraph of G induced by S* is denoted by $G[S]$ and it is defined as the subgraph of G with vertex set S and edge set $\{(u, v) \in E(G) : u, v \in S\}$ and the subgraph obtained after deleting S is denoted as $G - S$. For $F \subseteq E(G)$, by $V(F)$ we denote the set $\{v \mid \exists u$ such that $uv \in F\}$. For a set $Z = V' \cup E'$ where $V' \subseteq V(G)$ and $E' \subseteq E(G)$, by $G(Z)$ we denote the subgraph $G' = (V' \cup V(E'), E')$. For a tuple $\mathcal{X} = (X, F)$ such that $X \subseteq V(G)$ and $F \subseteq E(G)$, by $G - \mathcal{X}$ we denote the graph $G' = (V(G) \setminus X, E(G) \setminus F)$ and by $V(\mathcal{X})$ we denote the vertex set $X \cup V(F)$. All vertices adjacent to a vertex v are called neighbours of v and the set of all such vertices is called *open* neighbourhood of v, denoted by $N_G(v)$. For a set of vertices $S \subseteq V(G)$, we define $N_G(S) = (\cup_{v \in S} N(v)) \setminus S$. We drop the subscript G when the graph is clear from the context.

We define the Mixed Multiway Cut-Uncut problem as follows.

Mixed Multiway Cut-Uncut (MMCU)	
Input:	A multigraph G, a set of terminals $T \subseteq V(G)$, an equivalence relation $\mathcal{R}$ on the set T and integers k and l.
Parameters:	k, l
Question:	Does there exist $X \subseteq (V(G) \setminus T)$ and $F \subseteq E(G)$ such that $\lvert X\rvert \leq k$, $\lvert F\rvert \leq l$ and for all $u, v \in T$, u and v belong to the same connected component of $G - (X, F)$ if and only if $(u, v) \in \mathcal{R}$?

We say that a tuple $\mathcal{X} = (X, F)$, where $X \subseteq V(G) \setminus T$ and $F \subseteq E(G)$, is a solution to an MMCU instance $\mathcal{I} = (G, T, \mathcal{R}, k, l)$ if $|X| \leq k$, $|F| \leq l$ and for all $u, v \in T$, u and v belong to the same connected component of $G - (X, F)$ if and only if $(u, v) \in \mathcal{R}$. We define a partial order on the solutions of the instance $\mathcal{I}$. For two solutions $\mathcal{X} = (X, F)$ and $\mathcal{X}' = (X', F')$ of an MMCU instance $\mathcal{I}$, we say that $\mathcal{X}' \leq \mathcal{X}$ if $X' \subseteq X$ and $F' \subseteq F$. We say that a solution $\mathcal{X}$ to an MMCU instance $\mathcal{I}$ is *minimal* if there does not exist another solution $\mathcal{X}'$ to $\mathcal{I}$ such that $\mathcal{X}' \neq \mathcal{X}$ and $\mathcal{X}' \leq \mathcal{X}$. For a solution $\mathcal{X} = (X, F)$ of an MMCU instance $\mathcal{I} = (G, T, \mathcal{R}, k, l)$ and $v \subseteq V(G)$, we say that $\mathcal{X}$ *affects* v if either $v \in X$ or there exists $u \in V(G)$ such that $uv \in F$.

Observation 1. *If $\mathcal{X} = (X, F)$ is a minimal solution to an* MMCU* *instance $\mathcal{I} = (G, T, \mathcal{R}, k, l)$, then none of the edges in F are incident to X.*

Now we state the definitions of good node separations and flower separations from [7]. Then we state the lemmas that state the running time to find such separations and the properties of the graph if such separations do not exist.

Lemma 2 (1.1 in [7]). *Given a set U of size n together with integers $0 \leq a, b \leq n$, one can in $\mathcal{O}(2^{\mathcal{O}(\min(a,b)\log(a+b))} n \log n)$ time construct a family $\mathcal{F}$ of at most $\mathcal{O}(2^{\mathcal{O}(\min(a,b)\log(a+b))} \log n)$ subsets of U, such that the following holds: for any sets $A, B \subseteq U$, $A \cap B = \emptyset$, $|A| \leq a$, $|B| \leq b$, there exists a set $S \in \mathcal{F}$ with $A \subseteq S$ and $B \cap S = \emptyset$.*

Definition 3 (C.1 in [7]). *Let G be a connected graph and $V^\infty \subseteq V(G)$ a set of undeletable vertices. A triple (Z, V_1, V_2) of subsets of $V(G)$ is called a (q, k)-good node separation, if $|Z| \leq k$, $Z \cap V^\infty = \emptyset$, V_1 and V_2 are vertex sets of two different connected components of $G - Z$ and $|V_1 \setminus V^\infty|, |V_2 \setminus V^\infty| > q$.*

Definition 4 (C.2 in [7]). *Let G be a connected graph, $V^\infty \subseteq V(G)$ a set of undeletable vertices, and $T_b \subseteq V(G)$ a set of border terminals in G. A pair $(Z, (V_i)_{i=1}^r)$ is called a (q, k)-flower separation in G (with regard to border terminals T_b), if the following holds:*

- *$1 \leq |Z| \leq k$ and $Z \cap V^\infty = \emptyset$; the set Z is the core of the flower separation $(Z, (V_i)_{i=1}^r)$;*
- *V_i are vertex sets of pairwise different connected components of $G - Z$, each set V_i is a petal of the flower separation $(Z, (V_i)_{i=1}^r)$;*

– $V(G) \setminus (Z \cup \bigcup_{i=1}^{r} V_i)$, *called a stalk, contains more than* q *vertices of* $V \setminus V^\infty$*;*
– *for each petal* V_i *we have* $V_i \cap T_b = \emptyset$, $|V_i \setminus V^\infty| \leq q$ *and* $N_G(V_i) = Z$*;*
– $|(\bigcup_{i=1}^{r} V_i) \setminus V^\infty| > q$.

Lemma 5 (C.3 in [7]). *Given a connected graph* G *with undeletable vertices* $V^\infty \subseteq V(G)$ *and integers* q *and* k*, one may find in* $\mathcal{O}(2^{\mathcal{O}(\min(q,k)\log(q+k))}n^3 \log n)$ *time a* (q,k)*-good node separation of* G*, or correctly conclude that no such separation exists.*

Lemma 6 (C.4 in [7]). *Given a connected graph* G *with undeletable vertices* $V^\infty \subseteq V(G)$ *and border terminals* $T_b \subseteq V(G)$ *and integers* q *and* k*, one may find in* $\mathcal{O}(2^{\mathcal{O}(\min(q,k)\log(q+k))}n^3 \log n)$ *time a* (q,k)*-flower separation in* G *w.r.t.* T_b*, or correctly conclude that no such flower separation exists.*

Lemma 7 (C.5 in [7]). *If a connected graph* G *with undeletable vertices* $V^\infty \subseteq V(G)$ *and border terminals* $T_b \subseteq V(G)$ *does not contain a* (q,k)*-good node separation or a* (q,k)*-flower separation w.r.t.* T_b *then, for any* $Z \subseteq V(G) \setminus V^\infty$ *of size at most* k*, the graph* $G - Z$ *contains at most* $(2q+2)(2^k - 1) + |T_b| + 1$ *connected components containing a vertex of* $V(G) \setminus V^\infty$*, out of which at most one has more than* q *vertices not in* V^∞.

3 NP-Completeness of Mixed Cut

We prove that Mixed Cut in NP-complete by giving a reduction from the Bipartite Partial Vertex Cover problem which is defines as follows.

Bipartite Partial Vertex Cover (BPVC)	
Input:	A bipartite graph $G = (X \uplus Y, E)$, integers p and q.
Output:	Does there exist $S \subseteq V(G)$ such that $\|S\| \leq p$ and at least q edges in E are incident on X?

Theorem 8 ([1,12]). BPVC *is NP-complete.*

For an instance of BPVC, we assume that the given bipartite graph does not have any isolated vertices, as a reduction rule can be applied in polynomial time which takes care of isolated vertices and produces an equivalent instance. Given an instance (G,p,q) of BPVC where $G = (X \uplus Y, E)$ is a bipartite graph, we get an instance (G',s,t,k,l) of Mixed Cut as follows. To get the graph G', we introduce two new vertices s and t and add all edges from s to X and t to Y. More formally, $G' = (V', E')$ where $V' = V(G) \cup \{s,t\}$ and $E' = E \cup \{sx \mid x \in X\} \cup \{ty \mid y \in Y\}$. Then we put $k = p$ and $l = m - q$, where $m = |E|$. It is easy to see that (G,p,q) is a Yes instance of BPVC if and only if (G',s,t,k,l) is a Yes instance of Mixed Cut, and hence we get the following theorem.

Theorem 9. Mixed Cut *is NP-complete even on bipartite graphs.*

4 An Algorithm for MMCU

In this section, we describe the FPT algorithm for MMCU. In fact, we will give an algorithm, which when provided with an instance $(G, T, \mathcal{R}, k, l)$ of MMCU, not only decides whether there exists a solution (X, F) such that $|X| \leq k$ and $|F| \leq l$, but also outputs such a solution that is also minimal. To that end, we assume that the graph is connected and that the number of equivalence classes is bounded by $(k + l)(k + l + 1)$.

4.1 Operations on the Graph

Definition 10. *Let $\mathcal{I} = (G, T, \mathcal{R}, k, l)$ be an* MMCU *instance and let $v \in V(G) \setminus T$. By bypassing a vertex v we mean the following operation: we delete the vertex v from the graph and, for any $u_1, u_2 \in N_G(v)$, we add an edge (u_1, u_2) if it is not already present in G.*

Definition 11. *Let $\mathcal{I} = (G, T, \mathcal{R}, k, l)$ be an* MMCU *instance and let $u, v \in T$ such that $(u, v) \in \mathcal{R}$. By identifying vertices u and v in T, we mean the following operation: we make a new set $T' = (T \setminus \{u, v\}) \cup \{x_{uv}\}$, for each edge of the form $uw \in E(G)$ or $vw \in E(G)$, we add an edge $x_{uv}w$ to $E(G)$ and we put the new vertex x_{uv} in the same equivalence class as vertices u and v. Observe that the operation might add parallel edges.*

Lemma 12. *Let $\mathcal{I} = (G, T, \mathcal{R}, k, l)$ be an* MMCU *instance, let $v \in V(G) \setminus T$ and let $\mathcal{I}' = (G', T, \mathcal{R}, k, l)$ be the instance $\mathcal{I}$ with v bypassed. Then:*

- *if $\mathcal{X} = (X, F)$ is a solution to $\mathcal{I}'$, then $\mathcal{X}$ is a solution to $\mathcal{I}$ as well;*
- *if $\mathcal{X} = (X, F)$ is a solution to $\mathcal{I}$ and $v \notin X$ and for all $u \in N(v)$ $vu \notin F$ then $\mathcal{X}$ is a solution to $\mathcal{I}'$ as well.*

Lemma 13. *Let $\mathcal{I} = (G, T, \mathcal{R}, k, l)$ be an* MMCU *instance and let $u, v \in T$ be two different terminals with $(u, v) \notin \mathcal{R}$, such that $uv \in E(G)$, then for any solution $\mathcal{X} = (X, F)$ of $\mathcal{I}$, we have $uv \in F$.*

The proof of the Lemma 13 follows from the fact that any solution must delete the edge uv to disconnect u from v. The proof of the next lemma follows by simple observation that u and v have at least $k+l+1$ internally vertex disjoint paths or from the fact that $(u, v) \in \mathcal{R}$ and thus after deleting the solution they must belong to the same connected component and thus every minimal solution does not use the edge $uv \in E(G)$.

Lemma 14. *Let $\mathcal{I} = (G, T, \mathcal{R}, k, l)$ be an* MMCU *instance and let $u, v \in T$ be two different terminals with $(u, v) \in \mathcal{R}$, such that $uv \in E(G)$ or $|N_G(u) \cap N_G(v)| > k + l$. Let $\mathcal{I}'$ be instance $\mathcal{I}$ with terminals u and v identified. Then the set of minimal solution of $\mathcal{I}$ and $\mathcal{I}'$ is the same.*

Lemma 15. *Let $\mathcal{I} = (G, T, \mathcal{R}, k, l)$ be an* MMCU *instance and let $U = \{v_1, v_2, \ldots, v_t\} \subseteq T$ be different terminals of the same equivalence class of $\mathcal{R}$, pairwise nonadjacent and such that $N_G(u_1) = N_G(u_2) = \cdots = N_G(u_t) \subseteq V(G) \setminus T$ and $t > l + 2$. Let $\mathcal{I}'$ be obtained from $\mathcal{I}$ by deleting all but $l + 2$ terminals in U (and all pairs that contain the deleted terminals in $\mathcal{R}$). Then the sets of minimal solutions to $\mathcal{I}$ and $\mathcal{I}'$ are equal.*

Lemma 16. *Let $\mathcal{I} = (G, T, \mathcal{R}, k, l)$ be an* MMCU *instance and let $uv \in E(G)$ be an edge with multiplicity more than $l + 1$. Then for any minimal solution $\mathcal{X} = (X, F)$ of $\mathcal{I}$, F does not contain any copies of uv.*

Proof. If $\{u, v\} \cap X \neq \emptyset$, then by Observation 1, we have that none of the copies of uv are in F. Otherwise, F contains at most l copies of edge uv. Let $\mathcal{X}' = (X, F \setminus \{uv\})$. Then we have that for any two $x, y \in V(G)$, x and y are adjacent in $G - \mathcal{X}$ if and only if they are adjacent in $G - \mathcal{X}'$, contradicting the minimality of $\mathcal{X}$.

4.2 Borders and Recursive Understanding

In this section, we define the bordered problem and describe the recursive phase of the algorithm. Let $\mathcal{I} = (G, T, \mathcal{R}, k, l)$ be an MMCU instance and let $T_b \subseteq V(G) \setminus T$ be a set of border terminals, where $|T_b| \leq 2(k + l)$. Define $\mathcal{I}_b = (G, T, \mathcal{R}, k, l, T_b)$ to be an instance of the bordered problem. By $\mathbb{P}(\mathcal{I}_b)$ we define the set of all tuples $\mathcal{P} = (X_b, E_b, \mathcal{R}_b, k', l')$, such that $X_b \subseteq T_b$, E_b is an equivalence relation on $T_b \setminus X_b$, $\mathcal{R}_b$ is an equivalence relation on $T \cup (T_b \setminus X_b)$ such that $E_b \subseteq \mathcal{R}_b$ and $\mathcal{R}_b|_T = \mathcal{R}$, $k' \leq k$ and $l' \leq l$. For a tuple $\mathcal{P} = (X_b, E_b, \mathcal{R}_b, k', l')$, by $G_{\mathcal{P}}$ we denote the graph $G \cup E_b$, that is the graph G with additional edges E_b.

The intuition behind defining the tuple $\mathcal{P}$ is as following. The set X_b denotes the intersection of the solution with the border terminals. The equivalence relation E_b tells which of the border terminals can be connected from outside the graph considered. This can be looked at as analogous to *torso* operation on the graph. The equivalence relation $\mathcal{R}_b$ tells how the terminals and border terminals are going to get partitioned in different connected components after deleting the solution. Since deletion of any solution respects the relation $\mathcal{R}$, we have that $\mathcal{R}_b|_T = \mathcal{R}$. The numbers k' and l' are guesses for how much the smaller graph is going to contribute to the solution.

We say that a tuple $\mathcal{X} = (X, F)$ is a solution to $(\mathcal{I}_b, \mathcal{P})$ where $\mathcal{P} = (X_b, E_b, \mathcal{R}_b, k', l')$ if $|X| \leq k'$, $|F| \leq l'$ and for all $u, v \in T \cup (T_b \setminus X_b)$, u and v belong to the same connected component of $G_{\mathcal{P}} - (X, F)$ if and only if $(u, v) \in \mathcal{R}_b$. We also say that $\mathcal{X}$ is a solution to $\mathcal{I}_b = (G, T, \mathcal{R}, k, l, T_b)$ whenever $\mathcal{X}$ is a solution to $\mathcal{I} = (G, T, \mathcal{R}, k, l)$. Now we define the bordered problem as follows.

Border-Mixed Multiway Cut-Uncut(B-MMCU)	
Input:	An MMCU instance $\mathcal{I} = (G, T, \mathcal{R}, k, l)$ with G being connected and a set $T_b \subseteq V(G) \setminus T$ such that $\lvert T_b \rvert \leq 2(k+l)$; denote $\mathcal{I}_b = (G, T, \mathcal{R}, k, l, T_b)$.
Output:	For each $\mathcal{P} = (X_b, E_b, \mathcal{R}_b, k', l') \in \mathbb{P}(\mathcal{I}_b)$, output a $\mathsf{sol}_{\mathcal{P}} = \mathcal{X}_{\mathcal{P}}$ being a minimal solution to $(\mathcal{I}_b, \mathcal{P})$, or $\mathsf{sol}_{\mathcal{P}} = \perp$ if no solution exists.

It is easy to see that MMCU reduces to B-MMCU, by putting $T_b = \emptyset$. Also, in this case, any answer to B-MMCU for $\mathcal{P} = (\emptyset, \emptyset, \mathcal{R}, k, l)$ returns a solution for MMCU instance. To bound the size of the solutions returned for an instance of B-MMCU we observe the following.

$$\begin{aligned}
\lvert \mathbb{P}(\mathcal{I}_b) \rvert &\leq (k+1)(l+1)(1+\lvert T_b \rvert(\lvert T_b \rvert + (k+l)(k+l+1)))^{\lvert T_b \rvert} \\
&\leq (k+1)(l+1)(1+2(k+l)^2(k+l+3))^{2(k+l)} \\
&= 2^{\mathcal{O}((k+l)\log(k+l))}
\end{aligned}$$

This is true because $\mathcal{R}_b$ has at most $(k+l)(k+l+1)+\lvert T_b \rvert$ equivalence classes, E_b has at most T_b equivalence classes, each $v \in T_b$ can either go to X_b or choose an equivalence class in $\mathcal{R}_b$ and E_b, and k' and l' have $k+1$ and $l+1$ possible values respectively. Let $q = (k+2l)(k+1)(l+1)(1+2(k+l)^2(k+l+3))^{2(k+l)}+k+l$, then all output solutions to a B-MMCU instance $\mathcal{I}_b$ affect at most $q-(k+l)$ vertices in total. Now we are ready to state the lemma which is central for the recursive understanding step.

Lemma 17. *Assume we are given a* B-MMCU *instance* $\mathcal{I}_b = (G, T, \mathcal{R}, k, l, T_b)$ *and two disjoint sets of vertices* $Z, V^* \subseteq V(G)$*, such that* $\lvert Z \rvert \leq k+l$*,* $Z \cap T = \emptyset$*,* $Z_W := N_G(V^*) \subseteq Z$*,* $\lvert V^* \cap T_b \rvert \leq k+l$ *and the subgraph of* G *induced by* $W := V^* \cup Z_W$ *is connected. Denote* $G^* = G[W]$*,* $T_b^* = (T_b \cup Z_W) \cap W$*,* $T^* = T \cap W$*,* $\mathcal{R}^* = \mathcal{R}|_{T \cap W}$ *and* $\mathcal{I}^* = (G^*, T^*, R^*, k, l, T_b^*)$*. Then* $\mathcal{I}^*$ *is a proper* B-MMCU *instance. Moreover, if we denote by* $(\mathsf{sol}^*_{\mathcal{P}^*})_{\mathcal{P}^* \in \mathbb{P}(\mathcal{I}_b^*)}$ *an arbitrary output to the* B-MMCU *instance* $\mathcal{I}_b^*$ *and*

$$U(\mathcal{I}_b^*) = T_b^* \cup \{v \in V(G) \mid \mathcal{P}^* \in \mathbb{P}(\mathcal{I}_b^*), \mathsf{sol}^*_{\mathcal{P}^*} = \mathcal{X}^*_{\mathcal{P}^*} \neq \perp \text{ and } \mathcal{X}^*_{\mathcal{P}^*} \text{ affects } v\},$$

then there exists a correct output $(\mathsf{sol}_{\mathcal{P}})_{\mathcal{P} \in \mathbb{P}(\mathcal{I}_b)}$ *to the* B-MMCU *instance* $\mathcal{I}_b$ *such that whenever* $\mathsf{sol}_{\mathcal{P}} = \mathcal{X}_{\mathcal{P}} \neq \perp$ *and* $\mathcal{X}_{\mathcal{P}}$ *is a minimal solution to* $(\mathcal{I}_b, \mathcal{P})$ *then* $V(\mathcal{X}_{\mathcal{P}}) \cap V^* \subseteq U(\mathcal{I}_b^*)$.

The proof of lemma basically says that for any solution (X, F) of $\mathcal{I}_b, \mathcal{P})$ we can replace its intersection with the graph G^* with one of the solutions of the recursive calls and it still remains a solution. Now we describe the recursive step of the algorithm.

Step 1. *Assume we are given a* B-MMCU *instance* $\mathcal{I}_b = (G, T, \mathcal{R}, k, l, T_b)$. *Invoke first the algorithm of Lemma 5 in a search for* $(q, k+l)$*-good node separation (with* $V^\infty = T$*). If it returns a good node separation* (Z, V_1, V_2)*, let* $j \in \{1, 2\}$ *be such that* $|V_j \cap T_b| \leq k+l$ *and denote* $Z^* = Z$, $V^* = V_j$. *Otherwise, if it returns that no such good node separation exists in* G*, invoke the algorithm of Lemma 6 in a search for* $(q, k+l)$*-flower separation w.r.t.* T_b *(with* $V^\infty = T$ *again). If it returns that no such flower separation exists in* G*, pass the instance* $\mathcal{I}_b$ *to the next step. Otherwise, if it returns a flower separation* $(Z, (V_i)_{i=1}^r)$*, denote* $Z^* = Z$ *and* $V^* = \bigcup_{i=1}^r V_i$.

In the case we have obtained Z^* *and* V^* *(either from Lemma 5 or Lemma 6), invoke the algorithm recursively for the* B-MMCU *instance* $\mathcal{I}_b^*$ *defined as in the statement of Lemma 17 for separator* Z^* *and set* V^**, obtaining an output* $(\mathsf{sol}^*_{\mathcal{P}^*})\mathcal{P}^* \in \mathbb{P}(\mathcal{I}_b^*)$. *Compute the set* $U(\mathcal{I}_b^*)$. *Bypass (in an arbitrary order) all vertices of* $V^* \setminus (T \cup U(\mathcal{I}_b^*))$. *Recall that* $T_b^* \subseteq U(\mathcal{I}_b^*)$*, so no border terminal gets bypassed. After all vertices of* $V^* \setminus U(\mathcal{I}_b^*)$ *are bypassed, perform the following operations on terminals of* $V^* \cap T$*:*

1. *As long as there exist two different* $u, v \in V^* \cap T$ *such that* $(u, v) \notin \mathcal{R}$*, and* $uv \in E(G)$*, then delete the edge* uv *and decrease* l *by* 1*; if* l *becomes negative by this operation, return* $\perp$ *for all* $\mathcal{P} \in \mathbb{P}(\mathcal{I}_b)$.
2. *As long as there exist two different* $u, v \in V^* \cap T$ *such that* $(u, v) \in \mathcal{R}$ *and either* $uv \in E(G)$ *or* $|N_G(u) \cap N_G(v)| > k + l$*, identify* u *and* v.
3. *If the above two rules are not applicable, then, as long as there exist pairwise distinct terminals* $u_1, u_2, \ldots, u_t \in T$ *of the same equivalence class of* $\mathcal{R}$ *that have the same neighbourhood and* $t > l + 2$*, delete* u_i *for* $i > l + 2$ *from the graph (and delete all pairs containing* u_i *from* $\mathcal{R}$*).*

Let $\mathcal{I}_b'$ *be the outcome instance.*

Finally, restart this step on the new instance $\mathcal{I}_b'$ *and obtain a family of solutions* $(\mathsf{sol}_{\mathcal{P}})_{\mathcal{P} \in \mathbb{P}(\mathcal{I}_b')}$ *and return this family as an output to the instance* $\mathcal{I}_b$.

After the bypassing operations, we have that V^* contains at most q vertices that are not terminals (at most $k+l$ border terminals and at most $q-(k+l)$ vertices which are neither terminals nor border terminals). Let us now bound the number of terminal vertices once Step 1 is applied. Note that, after Step 1 is applied, for any $v \in T \cap V^*$, we have $N_G(v) \subseteq (V^* \setminus T) \cup Z$ and $|(V^* \setminus T) \cup Z| \leq (q+k+l)$. Due to the first and second rule in Step 1, for any set $A \subseteq (V^* \setminus T) \cup Z$ of size $k+l+1$, at most one terminal of $T \cap V^*$ is adjacent to all vertices of A. Due to the third rule in Step 1, for any set $B \subseteq (V^* \setminus T) \cup Z$ of size at most $k+l$ and for each equivalence class of $\mathcal{R}$, there are at most $l+2$ terminals of this equivalence class with neighbourhood exactly B. Let $q' := |T \cup V^*|$, then we have the following.

$$q' \leq (q+k+l)^{k+l+1} + (l+2)(k+l)(k+l+1)\sum_{i=1}^{k+l}(q+k+l)^i = 2^{\mathcal{O}((k+l)^2 \log(k+l))}$$

Lemma 18. *Assume that we are given a* B-MMCU *instance* $\mathcal{I}_b = (G, T, \mathcal{R}, k, l, T_b)$ *on which Step 1 is applied, and let* $\mathcal{I}'_b$ *be an instance after Step 1 is applied. Then any correct output to the instance* $\mathcal{I}'_b$ *is a correct output to the instance* $\mathcal{I}_b$ *as well. Moreover, if Step 1 outputs* $\perp$ *for all* $\mathcal{P} \in \mathbb{P}(\mathcal{I}'_b)$*, then this is a correct output to* $\mathcal{I}_b$*.*

Proof. We first note that by Lemma 17, for all $\mathcal{P} \in \mathbb{P}(\mathcal{I}_b)$, for all the vertices $v \notin U(\mathcal{I}^*_b)$, there exists a minimal solution to $(\mathcal{I}_b, \mathcal{P})$ that does not affect v, hence by Lemma 12, the bypassing operation is justified. The second and third rules are justified by Lemmas 14 and 15 respectively. The first rule is justified by Lemma 13, and if application of this rule makes l negative then for any $\mathcal{P} \in \mathbb{P}(\mathcal{I}_b)$, there is no solution to $(\mathcal{I}_b, \mathcal{P})$.

A careful running time analysis of Step 1 gives us the following recurrence.

$$T(n) \leq \max_{q+1 \leq n' \leq n-q-1} \big(\mathcal{O}(2^{\mathcal{O}((k+l)^2 \log(k+l))} n^3 \log n) + T(n' + k + l) + T(\min(n-1, n - n' + q + q'))\big)$$

The base case for the recursive calls is the high connectivity phase, which takes time $\mathcal{O}(2^{\mathcal{O}((k+l)^3 \log(k+l))} n^3 \log n)$ as we will argue later. Solving the recurrence for the worst case gives $T(n) = \mathcal{O}(2^{\mathcal{O}((k+l)^3 \log(k+l))} n^4 \log n)$, which is the desired upper bound for the running time of the algorithm.

4.3 High Connectivity Phase

In this section we describe the high connectivity phase for the algorithm. Assume we have a B-MMCU instance $\mathcal{I}_b = (G, T, \mathcal{R}, k', l', T_b)$ where Step 1 is not applicable. Let us fix $\mathcal{P} = (X_b, E_b, \mathcal{R}_b, k, l) \in \mathbb{P}(\mathcal{I}_b)$. We iterate through all possible values of $\mathcal{P}$ and try to find a minimal solution to $(\mathcal{I}_b, \mathcal{P})$. Since $|\mathbb{P}(\mathcal{I}_b)| = 2^{\mathcal{O}((k+l)\log(k+l))}$ it results in a factor of $2^{\mathcal{O}((k+l)\log(k+l))}$ in the running time. For a graph G, by $\mathcal{L}(G)$ we denote the set $V(G) \cup E(G)$. Similarly, for a tuple $\mathcal{X} = (X, F)$, by $\mathcal{L}(\mathcal{X})$ we denote the set $X \cup F$. We once again need to use Lemmas 13–15 to bound number of terminals. We also need to apply Lemma 16 to bound the number of edges.

Step 2. *Apply Lemmas 13, 14 and 15 exhaustively on the set* T *of terminals in the graph (as done in rules 1–3 of Step 1, but doing it for all of* T *instead of just* $T \cap V^*$*). Apply Lemma 16 to reduce multiplicity of all edges in the graph to at most* $l + 1$*.*

The running time analysis of applying Lemmas 13–15 in this step is exactly the same as the one done in Step 1. Also, Lemma 16 can be applied in $\mathcal{O}(n^2 l)$ time. Hence, the step takes $\mathcal{O}(n^3(k + l + \log n))$ time. After applying Step 2 exhaustively, we know that no two terminals are adjacent, and hence for any solution $\mathcal{X} = (X, F)$, we have that $F \cap E(G[T]) = \emptyset$.

Now we look at what can happen after deleting a set $\mathcal{X} = (X, F)$ from the graph G such that $X \subseteq V(G) \setminus T$, $F \subseteq E(G)$, $|X| \leq k$ and $|F| \leq l$. Since we have assumed that Step 1 is not applicable, for any $\mathcal{X} = (X, F)$ where $X \subseteq V(G) \setminus T$, $F \subseteq E(G)$, $|X| \leq k$ and $|F| \leq l$, Lemma 7 implies that the graph $G - \mathcal{X}$ contains at most $t := (2q+2)(2(k+l)1) + 2(k+l) + 1$ connected components containing a non-terminal out of which at most one can contain more than q vertices outside T. Let us denote its vertex set by $\mathsf{big}(\mathcal{X})$ (observe that this can possibly be the empty set, in case such a component does not exist). Now we define the notion of interrogating a solution, which will help us in highlighting the solution.

Definition 19. *Let $\mathcal{Z} = (Z, F')$ where $Z \subseteq V(G) \setminus T$, $F' \subseteq E(G) \setminus E(G[T])$, $|Z| \leq k$ and $|F'| \leq l$ and let $S \subseteq \mathcal{L}(G) \setminus T$. We say that S interrogates $\mathcal{Z}$ if the following holds:*

- *$S \cap \mathcal{L}(\mathcal{Z}) = \emptyset$;*
- *for any connected component C of $G - \mathcal{Z}$ with at most q vertices outside T, all vertices and edges of C belong to $S \cup T$.*

Lemma 20. *Let $q'' = (qt+k+l)^{k+l+1} + (l+2)(k+l)(k+l+1)\sum_{i=1}^{k+l}(qt+k+l)^i$. Let $\mathcal{F}$ be a family obtained by the algorithm of Lemma 7 for universe $U = \mathcal{L}(G) \setminus T$ and constants $a = qt + (l+1)\binom{q''+qt}{2}$ and $b = k + l$, Then, for any $\mathcal{Z} = (Z, F')$ where $Z \subseteq V(G) \setminus T$, $F' \subseteq E(G) \setminus E(G[T])$, $|Z| \leq k$ and $|F'| \leq l$, there exists a set $S \in \mathcal{F}$ that interrogates $\mathcal{Z}$.*

The proof of this lemma follows from the observation that since we can bound the number of vertices and edges in the small components, there exists a set family of desired size.

Step 3. *Compute the family $\mathcal{F}$ from Lemma 20 and branch into $|\mathcal{F}|$ subcases, indexed by sets $S \in \mathcal{F}$. In a branch S we seek for a minimal solution $\mathcal{X}_\mathcal{P}$ to $(\mathcal{I}_b, \mathcal{P})$, which is interrogated by S.*

Note that since we have $q'' = 2^{\mathcal{O}((k+l)^2 \log(k+l))}$ and $q, t = 2^{\mathcal{O}((k+l)\log(k+l))}$, the family $\mathcal{F}$ of Lemma 2 is of size $\mathcal{O}(2^{\mathcal{O}((k+l)^3 \log(k+l))} \log n)$ and can be computed in $\mathcal{O}(2^{\mathcal{O}((k+l)^3 \log(k+l))} n \log n)$ time. The correctness of Step 3 is obvious from Lemma 20. As discussed, it can be applied in $\mathcal{O}(2^{\mathcal{O}((k+l)^3 \log(k+l))} n \log n)$ time and gives rise to $\mathcal{O}(2^{\mathcal{O}((k+l)^3 \log(k+l))} \log n)$ subcases. The size of the separating family here is the reason for the extra factor of $k + l$ in the exponent, as compared to the algorithm in [7]. There, it was needed to only consider the vertices of the small connected components while looking for a solution, while for our problem, we needed to find a separating family for both the vertices and edges of small components, which has a larger size, and hence a larger separating family is needed.

Lemma 21. *Let $\mathcal{X}_\mathcal{P} = (X, F)$ be a solution to $(\mathcal{I}_b, \mathcal{P})$ interrogated by S. Then there exists a set $T^{\mathsf{big}} \subseteq T \cup (T_b \setminus X_b)$ that is empty or contains all vertices of exactly one equivalence class of $\mathcal{R}_b$, such that $X \subseteq (X_b \cup N_G(S(T^{\mathsf{big}}))$ and*

$F = A_{G,X}(S(T^{\mathsf{big}}))$, where $S(T^{\mathsf{big}})$ is the union of vertex sets of all connected components of $G(S \cup T \cup (T_b \setminus X_b))$ that contain a vertex of $(T \cup (T_b \setminus X_b)) \setminus T^{\mathsf{big}}$ and $A_{G,X}(S(T^{\mathsf{big}}))$ is set of edges in G which have at least one end point in $S(T^{\mathsf{big}})$ but do not belong to any of the connected components of $G[S(T^{\mathsf{big}})]$ and are not incident on X.

Step 4. *For each branch, where S is the corresponding guess, we do the following. For each set T^{big} that is empty or contains all vertices of one equivalence class of $\mathcal{R}_b$, if $|N_G(S(T^{\mathsf{big}}))| \leq k + l$, then for each $X \subseteq X_b \cup N_G(S(T^{\mathsf{big}}))$ such that $|X| \leq k$, and $F = A_{G,X}(S(T^{\mathsf{big}}))$, check whether (X, F) is a solution to $(\mathcal{I}_b, \mathcal{P})$ interrogated by S. For each $\mathcal{P}$, output a minimal solution to $(\mathcal{I}_b, \mathcal{P})$ that is interrogated by S. Output $\perp$ if no solution is found for any choice of S, T^{big} and X.*

The correctness of the step follows from Lemma 21 and the fact that if S interrogates a solution $\mathcal{X}$ to $(\mathcal{I}_b, \mathcal{P})$, then $|N_G(S(T^{\mathsf{big}}))| \leq k + l$. Note that $\mathcal{R}$ has at most $(k+l)(k+l+1)$ equivalence classes. As $|T_b| \leq 2(k+l)$, we have $\mathcal{R}_b$ has at most $(k+l)(k+l+3)$ equivalence classes, and hence there are at most $(k+l)(k+l+3)+1$ choices of the set T^{big}. For each T^{big}, computing $N_G(S(T^{\mathsf{big}}))$ and checking whether $|N_G(S(T^{\mathsf{big}}))| \leq k + l$ takes $\mathcal{O}(n^2)$ time. Since $X_b \leq k$, there are at most $(k+1)(2k+l)^k$ choices for X, and then computing $F = A_{G,X}(S(T^{\mathsf{big}}))$ and checking whether (X, F) is a solution to $(\mathcal{I}_b, \mathcal{P})$ interrogated by S take $\mathcal{O}(n^2)$ time each. Finally, checking whether the solution is minimal or not and computing a minimal solution takes additional $\mathcal{O}((k+l)n^2)$ time. Therefore Step 4 takes $\mathcal{O}(2^{\mathcal{O}((k+l)^3 \log(k+l))} n^2 \log n)$ time for all subcases.

This finishes the description of fixed-parameter algorithm for MMCU and we get the following theorem.

Theorem 22. MMCU *can be solved in $\mathcal{O}(2^{\mathcal{O}((k+l)^3 \log(k+l))} n^4 \log n)$ time.*

References

1. Apollonio, N., Simeone, B.: The maximum vertex coverage problem on bipartite graphs. Discrete Appl. Math. **165**, 37–48 (2014)
2. Beineke, L.W., Wilson, R.J. (eds.): Topics in Structural Graph Theory. Cambridge University Press, Cambridge (2013)
3. Bousquet, N., Daligault, J., Thomassé, S.: Multicut is FPT. In: Proceedings of the 43rd ACM Symposium on Theory of Computing, STOC 2011, San Jose, CA, USA, 6–8 June 2011, pp. 459–468 (2011)
4. Cao, Y., Marx, D.: Chordal editing is fixed-parameter tractable. In: 31st International Symposium on Theoretical Aspects of Computer Science, STACS 2014, Lyon, France, 5–8 March 2014, pp. 214–225 (2014)
5. Chen, J., Liu, Y., Lu, S.: An improved parameterized algorithm for the minimum node multiway cut problem. Algorithmica **55**(1), 1–13 (2009)
6. Chen, J., Liu, Y., Lu, S., O'Sullivan, B., Razgon, I.: A fixed-parameter algorithm for the directed feedback vertex set problem. J. ACM **55**(5), 1–19 (2008)

7. Chitnis, R.H., Cygan, M., Hajiaghayi, M., Pilipczuk, M., Pilipczuk, M.: Designing FPT algorithms for cut problems using randomized contractions. In: 53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2012, New Brunswick, NJ, USA, 20–23 October 2012, pp. 460–469 (2012)
8. Chitnis, R.H., Hajiaghayi, M., Marx, D.: Fixed-parameter tractability of directed multiway cut parameterized by the size of the cutset. SIAM J. Comput. **42**(4), 1674–1696 (2013)
9. Downey, R.G., Fellows, M.R.: Fundamentals of Parameterized Complexity. Texts in Computer Science. Springer, London (2013)
10. Flum, J., Grohe, M.: Parameterized Complexity Theory. Texts in Theoretical Computer Science. An EATCS Series. Springer, Berlin (2006)
11. Frank, A.: Connections in combinatorial optimization. Discrete Appl. Math. **160**(12), 1875 (2012)
12. Joret, G., Vetta, A.: Reducing the rank of a matroid. CoRR, abs/1211.4853 (2012)
13. Kawarabayashi, K., Thorup, M.: The minimum k-way cut of bounded size is fixed-parameter tractable. In: IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011, Palm Springs, CA, USA, 22–25 October 2011, pp. 160–169 (2011)
14. Kratsch, S., Pilipczuk, M., Pilipczuk, M., Wahlström, M.: Fixed-parameter tractability of multicut in directed acyclic graphs. In: Czumaj, A., Mehlhorn, K., Pitts, A., Wattenhofer, R. (eds.) ICALP 2012, Part I. LNCS, vol. 7391, pp. 581–593. Springer, Heidelberg (2012)
15. Marx, D.: Parameterized graph separation problems. Theor. Comput. Sci. **351**(3), 394–406 (2006)
16. Marx, D., O'Sullivan, B., Razgon, I.: Finding small separators in linear time via treewidth reduction. ACM Trans. Algorithms **9**(4), 30 (2013)
17. Marx, D., Razgon, I.: Fixed-parameter tractability of multicut parameterized by the size of the cutset. SIAM J. Comput. **43**(2), 355–388 (2014)
18. Niedermeier, R.: Invitation to Fixed Parameter Algorithms. Oxford Lecture Series in Mathematics and Its Applications. Oxford University Press, Oxford (2006)
19. Reed, B.A., Smith, K., Vetta, A.: Finding odd cycle transversals. Oper. Res. Lett. **32**(4), 299–301 (2004)