Algorithmica (2009) 55: 1–13
DOI 10.1007/s00453-007-9130-6

An Improved Parameterized Algorithm for the Minimum Node Multiway Cut Problem

Jianer Chen · Yang Liu · Songjian Lu

Received: 1 August 2007 / Accepted: 6 November 2007 / Published online: 21 November 2007

Abstract The PARAMETERIZED NODE MULTIWAY CUT problem is for a given graph to find a separator of size bounded by k whose removal separates a collection of terminal sets in the graph. In this paper, we develop an $O(k4^k n^3)$ time algorithm for this problem, significantly improving the previous algorithm of time $O(4^{k^3} n^5)$ for the problem. Our result gives the first polynomial time algorithm for the MINIMUM NODE MULTIWAY CUT problem when the separator size is bounded by $O(\log n)$.

Keywords Multiway cut problem · Parameterized algorithm · Fixed-parameter tractability · Minimum cut · Network flow

1 Introduction

The MULTITERMINAL CUT problem is a well-known problem, and has been extensively studied [2, 12, 14]. Applications of this problem are found in distributed computing [15], VLSI [5], computer vision [1], and many other fields. The problem is defined as follows: given an undirected graph $G = (V, E)$ and a set of l vertices $\{t_1, \ldots, t_l\}$ in G (the vertices t_i are called *terminals*), find an edge set E' of minimum

A preliminary version of this paper was presented at The 10th Workshop on Algorithms and Data Structures (WADS 2007).

This work was supported in part by the National Science Foundation under the Grants CCR-0311590 and CCF-0430683.

J. Chen · Y. Liu · S. Lu (✉)
Department of Computer Science, Texas A&M University, College Station, TX 77843, USA
e-mail: sjlu@cs.tamu.edu

J. Chen
e-mail: chen@cs.tamu.edu

Y. Liu
e-mail: yangliu@cs.tamu.edu

size in G such that after the deletion of E', no two terminals are in the same connected component. This problem is NP-hard for general graphs for any fixed integer $l \geq 3$, and is also NP-hard for planar graphs when l is not fixed [7].

A generalization of the MULTITERMINAL CUT problem is the MINIMUM NODE MULTIWAY CUT problem, which, for a given graph and a given set of terminals, is to find a vertex set S of minimum size such that after the deletion of S, no two terminals are in the same connected component. The MINIMUM NODE MULTIWAY CUT problem is at least as hard as the MULTITERMINAL CUT problem, as the latter can be reduced to the former in time $O(|V| + |E|)$, if we require that no terminal be in the separator S [6]. Therefore, the MINIMUM NODE MULTIWAY CUT problem is also NP-hard if the number l of terminals is at least 3.

When there are only two terminals s and t, the MULTITERMINAL CUT problem and the MINIMUM NODE MULTIWAY CUT problem become the edge version and the vertex version of the MINIMUM s-t CUT problem, respectively. According to the max-flow min-cut theorem [10], the MINIMUM s-t CUT problem, for both the edge version and the vertex version, can be solved via algorithms for the MAXIMUM s-t FLOW problem. For an undirected graph G of n vertices and m edges, the MAXIMUM s-t FLOW problem can be solved in time $O(n^{7/6}m^{2/3})$ [11]. In consequence, when there are only two terminals, the MULTITERMINAL CUT problem and the MINIMUM NODE MULTIWAY CUT problem can also be solved in time $O(n^{7/6}m^{2/3})$.

A natural extension of the MINIMUM NODE MULTIWAY CUT problem is to have a collection of terminal sets, instead of a collection of individual terminals. Formally, let $G = (V, E)$ be an undirected graph, and let $\{T_1, \ldots, T_l\}$ be a collection of *terminal sets* where each T_i is a subset of vertices in G. A *separator* S for $\{T_1, T_2, \ldots, T_l\}$ is a subset of vertices in G such that no vertex in S is in any terminal set, and after deleting S from the graph G, no connected component in the resulting graph contains vertices from more than one terminal set.

In certain real world applications, one may expect that the size of the separator be small. For example, suppose that we are given a network (i.e., a graph) $G = (V, E)$ and a collection of network node groups $\{T_1, \ldots, T_l\}$ in G, and we want to monitor the message communication among the node groups. A separator for $\{T_1, \ldots, T_l\}$ in the network G will well serve for this purpose: any communication path between any two node groups must pass through at least one node in the separator. Therefore, if we set up a monitor process in each of the nodes in the separator, then we can monitor all communications among the node groups. Naturally, we may want to limit the cost of this monitoring system by using only a small number of "monitor nodes" in the network G.

This motivates a parameterized version of the MINIMUM NODE MULTIWAY CUT problem, which will be called the PARAMETERIZED NODE MULTIWAY CUT problem and is defined as follows: given an undirected graph $G = (V, E)$, a collection of pairwise disjoint terminal sets $\{T_1, \ldots, T_l\}$ (where each T_i is a subset of vertices in G), and a parameter k, either construct a separator of at most k vertices in G, or report that no such a separator exists. Our goal is, for the PARAMETERIZED NODE MULTIWAY CUT problem, to develop a *fixed-parameter tractable algorithm* [9], i.e., an algorithm whose running time is of the form $f(k)n^c$ with a function f independent of the input size n and a constant c. In particular, when the parameter value k is small, such

a fixed-parameter tractable algorithm will be practically effective. In fact, the study of fixed-parameter tractable algorithms for a variety of parameterized problems has drawn considerable attention recently and has direct impact on real word applications where the selected parameter varies in a small range [9].

It can be derived from the graph minor theory of Robertson and Seymour [9] that there is a fixed-parameter tractable algorithm for the PARAMETERIZED NODE MULTIWAY CUT problem. However, the proof is not constructive. An explicit constructive algorithm for the problem was given by Marx [13], who developed an algorithm of running time $O(n^5 4^{k^3})$ for the PARAMETERIZED NODE MULTIWAY CUT problem for its original version (i.e., in which each terminal set is restricted to contain a single terminal). To our knowledge, this is the only known constructive fixed-parameter tractable algorithm for the problem.

In this paper, we present an algorithm of running time $O(n^3 k 4^k)$ for the PARAMETERIZED NODE MULTIWAY CUT problem, which significantly improves the algorithm given in [13]. In the real world of computing, this improvement makes it become possible to practically solve the problem for some reasonable values of the parameter k. For example, for the case of $k = 10$, our algorithm has running time $O(n^3 4^{10})$, which is practically feasible using the currently available computation power. On the other hand, the algorithm in [13] in this case has running time $O(n^5 4^{1000})$, which is totally infeasible from the practical point of view. Theoretically, our result gives the first polynomial time algorithm for the MINIMUM NODE MULTIWAY CUT problem when the size of the optimal separator is of order $O(\log n)$.

Finally, we remark that the techniques we developed in this paper seem to be very powerful for solving various kinds of multiway cut problems. In particular, very recently the techniques have been extended to directed graphs, and led to a fixed parameter tractable algorithm for the FEEDBACK VERTEX SET problem on directed graphs [4], thus resolving an outstanding open problem in the area of parameterized computation and complexity [8, 9].

2 On Minimum V-cuts between Two Terminal Sets

We start with some terminology. All graphs in our discussion are supposed to be undirected.

Let $G = (V, E)$ be a graph and let u and v be two vertices in G. A *path between u and v* is a simple path in G whose two ends are u and v, respectively. We say that there is a *path between a vertex u and a vertex subset V'* if there is a path between the vertex u and a vertex v in the subset V'. For two vertex subsets V_1 and V_2, we say that there is a *path between V_1 and V_2* if there exist a vertex u in V_1 and a vertex v in V_2 such that there is a path between u and v. Two paths are *internally disjoint* if there is no vertex that is an internal vertex for both the paths.

Let G be a graph, and let $\{T_1, \ldots, T_l\}$ be a collection of pairwise disjoint terminal sets (each terminal set is a subset of vertices in G). A subset S of vertices in G is a *separator* for $\{T_1, \ldots, T_l\}$ if S contains no vertex in any of the sets $T_1, \ldots, T_l$, and if after deleting all vertices in S from G, there is no path between any two different subsets T_i and T_j in the resulting graph. In particular, a separator S for two terminal sets T_1 and T_2 is also called a *V-cut* between the two sets T_1 and T_2.

Finally, for a subset V' of vertices in the graph G, we will denote by $G(V')$ the subgraph of G that is induced by the vertex subset V'.

We start with the following well-known theorem.

Proposition 2.1 (Menger's Theorem–Vertex Version [3]) *Let u and v be two distinct and nonadjacent vertices in a graph G. Then the maximum number of internally disjoint paths between u and v in G is equal to the size of a minimum V-cut between u and v in G.*

Proposition 2.1 can be generalized from the case for two vertices to the case for two vertex subsets, as follows.

Let T be a subset of vertices in the graph $G = (V, E)$. By *merging T (into a single vertex)*, we mean the operation that first deletes all vertices in T then creates a new vertex w adjacent to each v of the vertices in $V - T$ where v is a neighbor of a vertex in T in the original graph G.

By the merging operation, and Proposition 2.1, we directly derive the following version for Menger's Theorem (the second part of the lemma follows from the fact that a minimum V-cut of size h between T_1 and T_2 must contain at least one vertex in each of the h internally disjoint paths between T_1 and T_2).

Lemma 2.2 *Let T_1 and T_2 be two disjoint vertex subsets in a graph G such that no vertex in T_1 is adjacent to a vertex in T_2. Then the maximum number h of internally disjoint paths between T_1 and T_2 in G is equal to the size of a minimum V-cut between T_1 and T_2 in G. Moreover, for any set π of h internally disjoint paths between T_1 and T_2 in G, every minimum V-cut between T_1 and T_2 in G contains exact one vertex in each of the paths in π.*

Lemma 2.2 provides an efficient algorithm that constructs the maximum number of internally disjoint paths and a minimum V-cut between two given vertex subsets in a graph.

Lemma 2.3 *Let T_1 and T_2 be two disjoint vertex subsets in a graph $G = (V, E)$ such that no vertex in T_1 is adjacent to a vertex in T_2. Then in time $O((|V| + |E|)k)$, we can decide if the size h of a minimum V-cut between T_1 and T_2 is bounded by k, and in case $h \leq k$, construct h internally disjoint paths between T_1 and T_2.*

Proof By merging the vertex subsets T_1 and T_2 into two vertices t_1 and t_2, respectively, we reduce the problem to the problem of determining if the minimum V-cut between the two vertices t_1 and t_2 has a size bounded by k, which can be further reduced, by standard techniques, to the problem of deciding if the maximum flow from t_1 to t_2 has a value bounded by k. To solve the maximum flow problem, we simply run the standard Ford-Fulkerson's augmenting path procedure [10] at most k times. Each execution of the procedure takes time $O(|V| + |E|)$ and increases the flow value by at least 1. Thus, after executing the procedure at most k times, either we end up with a flow of value larger than k—in this case the V-cut between T_1 and T_2 has a size larger than k; or we obtain a maximum flow from t_1 to t_2 whose value

h is bounded by k—in this case we can easily construct h internally disjoint paths between T_1 and T_2. □

3 The Main Algorithm

Now we return back to the PARAMETERIZED NODE MULTIWAY CUT problem. Formally, an instance $(G, \{T_1, \ldots, T_l\}, k)$ of the PARAMETERIZED NODE MULTIWAY CUT problem consists of an undirected graph G, a collection $\{T_1, \ldots, T_l\}$ of pairwise disjoint *terminal sets* (each terminal set is a vertex subset in G), and a parameter k. The objective is to either construct a separator of at most k vertices for $\{T_1, \ldots, T_l\}$, or conclude that no such a separator exists.

Before we formally present our algorithm, we give a less formal but intuitive explanation on the basic idea of the algorithm. Let the size of a minimum V-cut between T_1 and $\bigcup_{j\neq 1} T_j$ be m.

Pick a vertex u that is not in any terminal set and has a neighbor in T_1. If u also has a neighbor in another terminal set T_i, $i \neq 1$, then we can directly include u in the separator (this is necessary because the separator must separate T_1 and T_i), and recursively find a separator of size $k-1$ in the remaining graph. On the other hand, if u has no neighbor in other terminal sets, then we compute the size m' of a minimum V-cut between the sets $T_1' = T_1 \cup \{u\}$ and $\bigcup_{i\neq 1} T_i$. It can be proved that we must have $m \leq m'$. Note that by Lemma 2.3, the values m and m' can be computed in polynomial time.

In the case $m = m'$, we will show that the instance $(G, \{T_1, T_2, \ldots, T_l\}, k)$ has a separator of size bounded by k if and only if the instance $(G, \{T_1', T_2, \ldots, T_l\}, k)$ has a separator of size bounded by k. Then we recursively work on the new instance $(G, \{T_1', T_2, \ldots, T_l\}, k)$. Thus, in the case of $m = m'$, we can reduce the number of vertices that are not in the separator by 1.

On the other hand, suppose $m < m'$. Then we branch on the vertex u in two cases, one includes u in the separator and the other excludes u from the separator. In the case of including the vertex u in the separator, we recursively work on the instance $(G - \{u\}, \{T_1, T_2, \ldots, T_l\}, k-1)$, in which the parameter value is decreased by 1; and in the case of excluding the vertex u from the separator, we recursively work on the instance $(G, \{T_1', T_2, \ldots, T_l\}, k)$, in which the size of the minimum V-cut between T_1' and $\bigcup_{i\neq 1} T_i$ is increased by at least 1.

Therefore, for the given instance $(G, \{T_1, T_2, \ldots, T_l\}, k)$, we can either (1) apply a polynomial time process that either decreases the parameter value by 1 or reduces the number of vertices not in the separator by 1, or (2) branch into two cases, of which one decreases the parameter value by 1 and the other increases the value m by at least 1 (see the definition of m given in the second paragraph in this section). Note that all these generated new instances will be "simpler" than the original given instance: (i) reducing the number of vertices not in the separator will narrow down our search space for the separator; (ii) an instance of parameter value bounded by 1 can be solved in polynomial time; and (iii) an instance in which the value m is larger than the parameter value k obviously has no separator of size bounded by k.

To present our formal discussions, we fix an instance $(G, \{T_1, \ldots, T_l\}, k)$ of the PARAMETERIZED NODE MULTIWAY CUT problem, where $G = (V, E)$ is a graph,

and $\{T_1, \ldots, T_l\}$ is a collection of terminal sets in G. Let the size of a minimum V-cut between T_1 and $\bigcup_{j\neq 1} T_j$ be m. Moreover, fix a vertex u that is not in any of the terminal sets but has a neighbor in the terminal set T_1. Let $T_1' = T_1 \cup \{u\}$.

Lemma 3.1 *Let m be the size of a minimum V-cut between the two sets T_1 and $\bigcup_{j\neq 1} T_j$, and let m' be the size of a minimum V-cut between the two sets T_1' and $\bigcup_{j\neq 1} T_j$. Then $m' \geq m$.*

Proof The lemma follows from the observation that every V-cut between the sets T_1' and $\bigcup_{j\neq 1} T_j$ is also a V-cut between the sets T_1 and $\bigcup_{j\neq 1} T_j$. □

The following theorem is the most crucial observation for our algorithm.

Theorem 3.2 *If the minimum V-cuts between the sets T_1 and $\bigcup_{j\neq 1} T_j$ and the minimum V-cuts between the sets T_1' and $\bigcup_{j\neq 1} T_j$ have the same size, then the instance $(G, \{T_1, T_2, \ldots, T_l\}, k)$ has a separator of size bounded by k if and only if the instance $(G, \{T_1', T_2, \ldots, T_l\}, k)$ has a separator of size bounded by k.*

Proof If the instance $(G, \{T_1', T_2, \ldots, T_l\}, k)$ has a separator S of size bounded by k, then it is obvious that S is also a separator for the instance $(G, \{T_1, T_2, \ldots, T_l\}, k)$. In consequence, the instance $(G, \{T_1, T_2, \ldots, T_l\}, k)$ also has a separator of size bounded by k.

Now we consider the other direction. Suppose that the instance $(G, \{T_1, T_2, \ldots, T_l\}, k)$ has a separator S_k of size bounded by k.

To simplify the discussion, denote by T_{other} the set $\bigcup_{j\neq 1} T_j$. Let S_m be a minimum V-cut between T_1' and T_{other} (note that S_m does not contain u). Then S_m is also a V-cut between T_1 and T_{other}. In fact, by the assumption of the theorem, S_m is also a minimum V-cut between T_1 and T_{other}. Let $C(T_1)$ be the set of vertices x such that either $x \in T_1$ or there is a path between x and T_1 in the subgraph $G - S_m$. In particular, since u is not in S_m and u is adjacent to T_1, we have $u \in C(T_1)$. Moreover, let $C(T_{\text{other}}) = V - C(T_1) - S_m$.

By Lemma 2.2, there exist $|S_m|$ internally disjoint paths between T_1 and T_{other}, each contains exactly one vertex in the set S_m. Therefore, each of these $|S_m|$ paths is cut into two subpaths by a vertex in S_m, such that one subpath is in the induced subgraph $G(C(T_1))$ and the another subpath is in the induced subgraph $G(C(T_{\text{other}}))$. From this, we derive that there are $|S_m|$ internally disjoint paths between T_1 and S_m in the induced subgraph $G(C(T_1) \cup S_m)$, each contains a distinct vertex in the set S_m.

Define $A = S_k \cap C(T_1)$, $B = S_k \cap S_m$, and $C = S_k \cap C(T_{\text{other}})$. Finally, let S_m' be the set of vertices x in S_m such that there is a path between x and T_{other} in the induced subgraph $G(C(T_{\text{other}}) \cup S_m - S_k)$ (see Fig. 1 for an intuitive illustration of these sets).

We first prove that $|A| \geq |S_m'|$.

From the fact that there are $|S_m|$ internally disjoint paths between T_1 and S_m in the induced subgraph $G(C(T_1) \cup S_m)$ in which each path contains a distinct vertex in the set S_m, we derive that there are $|S_m'|$ internally disjoint paths between T_1 and S_m' in the induced subgraph $G(C(T_1) \cup S_m')$. If $|A| < |S_m'|$, then there must be a path P_1 between T_1 and a vertex v' in S_m' in the subgraph $G(C(T_1) \cup S_m' - A) = G(C(T_1) \cup S_m' - S_k)$.

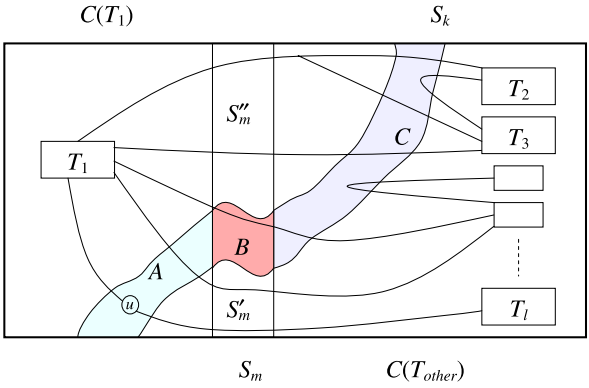

Fig. 1 Decomposition of separators

Moreover, by the definition of the set S'_m, there is also a path P_2 between v' and T_{other} in the induced subgraph $G(C(T_{\text{other}}) \cup S_m - S_k)$. The concatenation of the paths P_1 and P_2 would give a path between T_1 and T_{other} in the induced subgraph $G(V - S_k)$, which contradicts the assumption that S_k is a separator of the instance $(G, \{T_1, T_2, \ldots, T_l\}, k)$. Therefore, we must have $|A| \geq |S'_m|$.

Define a set $S'_k = S'_m \cup B \cup C$. We now prove that the set S'_k is a separator for the instance $(G, \{T'_1, T_2, \ldots, T_l\}, k)$. Suppose that the set S'_k is not a separator for the instance $(G, \{T'_1, T_2, \ldots, T_l\}, k)$, then there are two vertices v_1 and v_2 that are in two different terminal sets in $\{T'_1, T_2, \ldots, T_l\}$ and there exists a path P between v_1 and v_2 in the induced subgraph $G(V - S'_k)$. We discuss this in two possible cases.

Case 1: There is a vertex w in the path P such that $w \in C(T_1)$. Because (1) at least one of the vertices v_1 and v_2 is in the set T_{other}, (2) there is a path between T_1 and w in the induced subgraph $G(C(T_1))$, and (3) S_m is a V-cut between T_1 and T_{other}, we conclude that there must be a vertex $s \in S_m$ that is also on the path P. Without loss of generality, we can suppose that the vertex v_1 is in the set T_{other}, and that the subpath P' of P that begins from v_1 and ends at s has no vertices from $C(T_1)$—for this we only have to pick the first vertex s in S_m when we traverse on the path P from v_1 to v_2. Then the path P' is in the induced subgraph $G(C(T_{\text{other}}) \cup S_m - S'_k)$, which is a subgraph of the induced subgraph $G(C(T_{\text{other}}) \cup S_m - S_k)$. Now by the definition of the set S'_m, the vertex s is in the set S'_m, thus in the set S'_k. But this is impossible because we assumed that the path P is in the induced subgraph $G(V - S'_k)$.

Case 2: All vertices of the path P come from the induced subgraph $G(V - S'_k - C(T_1))$. Then neither of the vertices v_1 and v_2 can be from the set T_1. Moreover, since $G(V - S'_k - C(T_1))$ is a subgraph of the induced subgraph $G(V - S_k)$, the path P, which is between two different terminal sets in $\{T_2, \ldots, T_l\}$, would contain no vertex in S_k. But this again contradicts the assumption that S_k is a separator for the instance $(G, \{T_1, T_2, \ldots, T_l\}, k)$.

Combining the discussions in Case 1 and Case 2, we conclude that the set S'_k is a separator for the instance $(G, \{T'_1, T_2, \ldots, T_l\}, k)$.

Since $|A| \geq |S'_m|$, $S_k = A \cup B \cup C$, and $S'_k = S'_m \cup B \cup C$, and A does not intersect $B \cup C$, we conclude that $|S_k| \geq |S'_k|$. In particular, if the instance

$(G, \{T_1, T_2, \ldots, T_l\}, k)$ has the separator S_k of size bounded by k, then the instance $(G, \{T_1', T_2, \ldots, T_l\}, k)$ has the separator S_k' of size also bounded by k.

This completes the proof of the theorem. □

The proof of Theorem 3.2 becomes complicated partially because the vertex u may be included in a separator for the instance $(G, \{T_1, T_2, \ldots, T_l\}, k)$. If we restrict that the vertex u is not in the separators for the instance $(G, \{T_1, T_2, \ldots, T_l\}, k)$, then a result similar to Theorem 3.2 can be obtained much more easily, even without the need of the condition that the minimum V-cuts between T_1 and $\bigcup_{j\neq 1} T_j$ and the minimum V-cuts between T_1' and $\bigcup_{j\neq 1} T_j$ have the same size. This is given in the following lemma. This result will also be needed in our algorithm.

Lemma 3.3 *Let S be a vertex subset in the graph G such that S does not include the vertex u. Then S is a separator for the instance $(G, \{T_1, T_2, \ldots, T_l\}, k)$ if and only if S is a separator for the instance $(G, \{T_1', T_2, \ldots, T_l\}, k)$.*

Proof If S is a separator for the instance $(G, \{T_1', T_2, \ldots, T_l\}, k)$, then as explained in Theorem 3.2, S is also a separator for the instance $(G, \{T_1, T_2, \ldots, T_l\}, k)$.

For the other direction, suppose that S is a separator for the instance $(G, \{T_1, T_2, \ldots, T_l\}, k)$. We show that S is also a separator for the instance $(G, \{T_1', T_2, \ldots, T_l\}, k)$. Suppose that S is not a separator for the instance $(G, \{T_1', T_2, \ldots, T_l\}, k)$. Then there is a path P in $G - S$ between two different terminal sets in $\{T_1', T_2, \ldots, T_l\}$. Let one of these two terminal sets in $\{T_1', T_2, \ldots, T_l\}$ be T_i, where $i \neq 1$. The path P must contain the vertex u (recall that S does not contain u)—otherwise the path P in $G - S$ would be between two different terminal sets in $\{T_1, T_2, \ldots, T_l\}$, contradicting the assumption that S is a separator for $(G, \{T_1, T_2, \ldots, T_l\}, k)$. However, this would imply that the path from T_1 to u (recall that u has a neighbor in T_1) then following the path P to the terminal set T_i would give a path in $G - S$ between T_1 and T_i, again contradicting the assumption that S is a separator for $(G, \{T_1, T_2, \ldots, T_l\}, k)$. This contradiction shows that the set S must be also a separator for the instance $(G, \{T_1', T_2, \ldots, T_l\}, k)$. □

Now, we are ready to present our algorithm. For an instance $(G, \{T_1, \ldots, T_l\}, k)$ of the PARAMETERIZED NODE MULTIWAY CUT problem, a vertex in the graph G that does not belong to any terminal sets will be called a "non-terminal".

The algorithm is given in Fig. 2.

Theorem 3.4 *The algorithm* **NMC**$(G, \{T_1, T_2, \ldots, T_l\}, k)$ *solves the* PARAMETERIZED NODE MULTIWAY CUT *problem in time* $O(n^3 k 4^k)$.

Proof We first prove the correctness of the algorithm. Let $(G, \{T_1, T_2, \ldots, T_l\}, k)$ be an input to the algorithm, which is an instance of the PARAMETERIZED NODE MULTIWAY CUT problem, where $G = (V, E)$ is a graph, $\{T_1, T_2, \ldots, T_l\}$ is a collection of terminal sets, and k is the upper bound of the size of the separator we are looking for.

If there is an edge whose two ends are in two different terminal sets, then we have no way to separate these two terminal sets since all vertices in a separator are supposed to be non-terminals. Step 1 handles this case correctly.

Algorithm NMC$(G, \{T_1, T_2, \ldots, T_l\}, k)$
input: an instance $(G, \{T_1, T_2, \ldots, T_l\}, k)$ of the PARAMETERIZED NODE MULTIWAY CUT problem ($l \geq 2$)
output: a separator of size bounded by k for $(G, \{T_1, T_2, \ldots, T_l\}, k)$, or report "No" (i.e., no such a separator)

1. **if** an edge has its two ends in two different terminal sets
   then return "No";
2. **if** a non-terminal w has two neighbors in two different terminal sets
   then return $w + \mathbf{NMC}(G - w, \{T_1, \ldots, T_l\}, k-1)$; ‡
3. find the size m_1 of a minimum V-cut between T_1 and $\bigcup_{j=2}^{l} T_j$;
4. **if** $m_1 > k$ **then** return "No";
5. **if** ($m_1 = 0$ and $l = 2$) **then** return $\emptyset$;
5.1 **if** ($m_1 = 0$ and $l > 2$) **then** return $\mathbf{NMC}(G, \{T_2, \ldots, T_l\}, k)$;
6. **else** pick a non-terminal u that has a neighbor in T_1; let $T_1' = T_1 + u$;
6.1 **if** the size of a minimum V-cut between T_1' and $\bigcup_{j=2}^{l} T_j$ is equal to m_1
   then return $\mathbf{NMC}(G, \{T_1', T_2, \ldots, T_l\}, k)$;
6.2 **else** $S = u + \mathbf{NMC}(G - u, \{T_1, T_2, \ldots, T_l\}, k-1)$;
   if S is not "No" **then** return S;
6.3 **else** return $\mathbf{NMC}(G, \{T_1', T_2, \ldots, T_l\}, k)$.

‡To simplify the expression, we suppose that "No" plus any vertex set gives a "No".

Fig. 2 An algorithm for the PARAMETERIZED NODE MULTIWAY CUT problem

If a non-terminal w has two neighbors that are in two different terminal sets, then w must be in the separator because otherwise the two terminal sets will not be separated. Thus, we can simply include the vertex w in the separator, and recursively find a separator of size bounded by $k-1$ for the same collection of terminal sets $\{T_1, T_2, \ldots, T_l\}$ in the remaining graph $G - w$. This case is correctly handled by step 2.

Step 3 computes the size m_1 of a minimum V-cut between the sets T_1 and $\bigcup_{j=2}^{l} T_j$. By Lemma 2.3, the value m_1 can be computed in time $O((|V| + |E|)k)$.

If $m_1 > k$, then the size of a minimum V-cut between T_1 and $\bigcup_{j=2}^{l} T_j$ is larger than k, which means that even separating the set T_1 from the other sets $\bigcup_{j=2}^{l} T_j$ requires more than k vertices. Thus, no separator of size bounded by k can exist for the terminal sets $T_1, T_2, \ldots, T_l$. This is handled by step 4.

In step 5 we handle the case $m_1 = 0$ and $l = 2$, which we do not need to remove any vertex to separate T_1 and T_2, i.e. the problem is solved. So we return an empty set $\emptyset$ as a separator of size 0 (note that because of step 4, here we must have $k \geq 0$). In step 5.1, $m_1 = 0$ and $l > 2$, which means that T_1 is already separated from $T_2, \ldots, T_l$. Hence we only need to find a separator to separate $T_2, \ldots, T_l$. Therefore, step 5.1 handles this case correctly.

When the algorithm reaches step 6, the following conditions hold true: (1) no edge has its two ends in two different terminal sets (because of step 1); (2) no non-terminal has two neighbors in two different terminal sets (because of step 2); (3) $0 < m_1 \leq k$ (because of steps 4–5). In particular, by condition (3), there must be a non-terminal u that has a neighbor in T_1.

Let $T_1' = T_1 + u$ and let m' be the size of a minimum V-cut between the sets T_1' and $\bigcup_{j=2}^{l} T_j$. If $m' = m_1$, then by Theorem 3.2, the instance $(G, \{T_1, T_2, \ldots, T_l\}, k)$ has a

separator of size bounded by k if and only if the instance $(G, \{T_1', T_2, \ldots, T_l\}, k)$ has a separator of size bounded by k. In particular, as shown in the proof of Theorem 3.2, a separator of size bounded by k for the instance $(G, \{T_1', T_2, \ldots, T_l\}, k)$ is actually also a separator for the instance $(G, \{T_1, T_2, \ldots, T_l\}, k)$. Therefore, in this case, we can recursively work on the instance $(G, \{T_1', T_2, \ldots, T_l\}, k)$, as given in step 6.1. On the other hand, if $m' \neq m_1$, which means $m' > m_1$, then we simply branch on the vertex u in two cases: (1) including u in the separator and recursively working on the remaining graph for a separator of size bounded by $k-1$, as given by step 6.2; and (2) excluding u from the separator thus looking for a separator that does not include u and is of size bounded by k for the instance $(G, \{T_1, T_2, \ldots, T_l\}, k)$. By Lemma 3.3, the second case is equivalent to finding a separator of size bounded by k for the instance $(G, \{T_1', T_2, \ldots, T_l\}, k)$. This case is thus handled by step 6.3.

This completes the proof of the correctness of the algorithm. Now we analyze the complexity of the algorithm.

The recursive execution of the algorithm can be described as a search tree $\mathcal{T}$. We first count the number of leaves in the search tree $\mathcal{T}$. Note that only steps 6.2–6.3 of the algorithm correspond to branches in the search tree $\mathcal{T}$. Let $D(k, m_1)$ be the total number of leaves in the search tree $\mathcal{T}$ for the algorithm $\mathbf{NMC}(G, \{T_1, T_2, \ldots, T_l\}, k)$, where m_1 is the size of a minimum V-cut between the sets T_1 and $\bigcup_{j=2}^{l} T_j$. Then steps 6.2–6.3 induce the following recurrence relation:

$$D(k, m_1) \leq D(k-1, m'') + D(k, m'''), \tag{1}$$

where m'' is the size of a minimum V-cut between T_1 and $\bigcup_{j=2}^{l} T_j$ *in the graph* $G-u$ as given in step 6.2, and m''' is the size of a minimum V-cut between T_1' and $\bigcup_{j=2}^{l} T_j$ *in the graph* G as given in step 6.3. Note that $m_1 - 1 \leq m'' \leq m_1$ because removing the vertex u from G cannot increase the size of a minimum V-cut between two sets, and can decrease the size of a minimum V-cut between the two sets by at most 1. Moreover, by Lemma 3.1 and because of step 6.1, the size m''' of a minimum V-cut between T_1' and $\bigcup_{j=2}^{l} T_j$ in step 6.3 is at least $m_1 + 1$. Summarizing these, we have

$$m_1 - 1 \leq m'' \leq m_1 \quad \text{and} \quad m''' \geq m_1 + 1. \tag{2}$$

Introduce a new function D' such that $D'(2k - m_1) = D(k, m_1)$, and let $t = 2k - m_1$. Then by Inequalities (1) and (2), the branch in step 6.2–6.3 in the algorithm becomes

$$D'(t) \leq D'(t_1) + D'(t_2),$$

where when $t = 2k - m_1$ then $t_1 = 2(k-1) - m'' \leq t - 1$, and $t_2 = 2k - m''' \leq t - 1$.

We also point out that certain non-branching steps (i.e., steps 2, 5.1, and 6.1) may also change the values of k and m_1, thus changing the value $t = 2k - m_1$. However, none of these steps increases the value $t = 2k - m_1$: (1) step 2 decreases the value k by 1 and the value m_1 by at most 1, which as a total will decrease the value $t = 2k - m_1$ by at least 1; (2) step 5.1 keeps the value k unchanged and, since we have $m_1 = 0$ before the execution of this step, the new value m_1 is at least as large as the old value m_1. As a consequence, the value $t = 2k - m_1$ is not increased; (3) finally, step 6.1 does not change the values k and m_1, thus neither changes the value

$t = 2k - m_1$. In summary, the value $t = 2k - m_1$ after a branching step to the next branching step can never be increased.

Our initial instance starts with $t = 2k - m_1 \leq 2k$. In the case $t = 2k - m_1 = 0$, because we also have the conditions $k \geq m_1 \geq 0$, we must have $m_1 = 0$ and $k = 0$, in this case the algorithm can solve the instance without further branching. Therefore, we have $D'(0) = 1$. Combining all these, we derive

$$D(k, m_1) = D'(2k - m_1) \leq 2^{2k},$$

and the search tree $\mathcal{T}$ has at most 2^{2k} leaves.

Finally, it is easy to verify that along each root-leaf path in the search tree $\mathcal{T}$, the running time of the algorithm is bounded by $O(n^3k)$, where n is the number of vertices in the graph. In conclusion, the running time of the algorithm **NMC**$(G, \{T_1, T_2, \ldots, T_l\}, k)$ is bounded by $O(n^3k4^k)$.

This completes the proof of the theorem. □

4 Extensions and Further Research

We developed new and powerful techniques that lead to an algorithm of running time $O(n^3k4^k)$ for the PARAMETERIZED NODE MULTIWAY CUT problem. The algorithm significantly improves previous algorithms for the problem. More recently, our techniques have been extended to directed graphs that lead to a fixed parameter tractable algorithm for the FEEDBACK VERTEX SET problem on directed graphs [4], thus resolving an outstanding open problem in parameterized computation and complexity.

In our definitions, we require that the terminal sets be pairwise disjoint and that a separator do not include any vertex in the terminal sets. This version of the problem may be properly called the *strict* PARAMETERIZED NODE MULTIWAY CUT problem. A natural question is whether our algorithm will still work if the constraints are removed. For this, we consider the following more general version of the problem. Let G be a graph, let $\{T_1, \ldots, T_l\}$ be a collection of terminal sets (which are not required to be pairwise disjoint), and let k be a parameter. We are asked to either construct a subset S of at most k vertices in G (where the set S is not required to be disjoint from the terminal sets) such that for any i and j, $1 \leq i, j \leq l$, there is no path from $T_i - S$ to $T_j - S$ in the graph $G - S$, or conclude that no such a subset S exists in the graph G. Let this problem be called the *general* PARAMETERIZED NODE MULTIWAY CUT problem.

Theorem 4.1 *The general* PARAMETERIZED NODE MULTIWAY CUT *problem can be solved in time* $O(n^3k4^k)$.

Proof Let $(G, \{T_1, \ldots, T_l\}, k)$ be an instance of the general PARAMETERIZED NODE MULTIWAY CUT problem, where G is a graph, $\{T_1, \ldots, T_l\}$ is a collection of terminal sets, and k is the parameter. If a vertex v is contained in more than one terminal set, then v should be directly included in the separator. Therefore, we can assume, without loss of generality, that the terminal sets are pairwise disjoint. In particular, $l \leq n$.

Introduce l new vertices $t_1, \ldots, t_l$, such that for each i, $1 \leq i \leq l$, the vertex t_i is adjacent to all vertices in T_i. Let G_+ be the new graph after adding these l new vertices. The graph G_+ has $n_+ = n + l \leq 2n$ vertices.

It is straightforward to verify that for any subset S of vertices in the graph G, there is no path between two subsets $T_i - S$ and $T_j - S$ in the graph $G - S$ *if and only if* there is no path between the vertices t_i and t_j in the graph $G_+ - S$, for all i and j, $1 \leq i$, $j \leq l$. Note that a subset S of vertices in G does not include any of the vertices $t_1, \ldots, t_l$ in the graph G_+, and that the subsets $\{t_1\}, \ldots, \{t_l\}$ are pairwise disjoint. Therefore, a subset S in the graph G is a solution to the instance $(G, \{T_1, \ldots, T_l\}, k)$ of the general PARAMETERIZED NODE MULTIWAY CUT problem *if and only if* S is a solution to the instance $(G_+, \{\{t_1\}, \ldots, \{t_l\}\}, k)$ for the strict PARAMETERIZED NODE MULTIWAY CUT problem. By Theorem 3.4, a solution to the instance $(G_+, \{\{t_1\}, \ldots, \{t_l\}\}, k)$ of the strict PARAMETERIZED NODE MULTIWAY CUT problem can be constructed in time $O(n_+^3 k 4^k) = O(n^3 k 4^k)$. We conclude that the general PARAMETERIZED NODE MULTIWAY CUT problem can also be solved in time $O(n^3 k 4^k)$. □

Another version of the problem is the PARAMETERIZED NODE MULTICUT problem [13], where we look for a separator of size k to separate each of the l given pairs of terminals. When both k and l are used as parameters, based on the techniques developed in the current paper, the fixed parameter tractable algorithm presented in [13] for the PARAMETERIZED NODE MULTICUT problem can be improved. On the other hand, if only k is used as the parameter, or if the graph G is a directed graph (or even just a directed acyclic graph), it is currently unknown whether the PARAMETERIZED NODE MULTICUT problem has fixed parameter tractable algorithms, which seems a very interesting topic for further research.

References

1. Boykov, Y., Veksler, O., Zabih, R.: Markov random fields with efficient approximations. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition, pp. 648–655, 1998
2. Calinescu, G., Karloff, H., Rabani, Y.: An improved approximation algorithm for multiway cut. J. Comput. Syst. Sci. **60**, 564–574 (2000)
3. Chartrand, G., Lesniak, L.: Graphs & Digraphs, 2nd edn. Wadsworth/Cole Mathematics Series, Belmont (1986)
4. Chen, J., Liu, Y., Lu, S., O'Sullivan, B., Razgon, I.: A fixed-parameter algorithm for the directed feedback vertex set problem. Manuscript (2007)
5. Cong, J., Labio, W., Shivakumar, N.: Multi-way VLSI circuit partitioning based on dual net representation. In: Proc. IEEE International Conference on Computer-Aided Design, pp. 56–62, 1994
6. Cunningham, W.: The optimal multiterminal cut problem. DIMACS Ser. Discrete Math. Theor. Comput. Sci. **5**, 105–120 (1991)
7. Dahlhaus, E., Johnson, D., Papadimitriou, C., Seymour, P., Yannakakis, M.: The complexity of multiterminal cuts. SIAM J. Comput. **23**, 864–894 (1994)
8. Downey, R., Fellows, M.: Fixed-parameter tractability and completeness I: basic results. SIAM J. Comput. **24**, 873–921 (1995)
9. Downey, R., Fellows, M.: Parameterized Complexity. Monograph in Computer Science. Springer, New York (1999)
10. Ford, L. Jr., Fulkerson, D.: Flows in Networks. Princeton University Press, Princeton (1962)
11. Karger, D., Levine, M.: Finding maximum flows in undirected graphs seems easier than bipartite matching. In: Proc. 30th Annual ACM Symposium on Theory of Computing, pp. 69–78, 1998

12. Karger, D., Klein, P., Stein, C., Thorup, M., Young, N.: Rounding algorithms for a geometric embedding of minimum multiway cut. In: Proc. 31th Annual ACM Symposium on Theory of Computing, pp. 668–678, 1999
13. Marx, D.: Parameterized graph separation problems. Theor. Comput. Sci. **351**, 394–406 (2006)
14. Naor, J., Zosin, L.: A 2-approximation algorithm for the directed multiway cut problem. SIAM J. Comput. **31**, 477–482 (2001)
15. Stone, H.: Multiprocessor scheduling with the aid of network flow algorithms. IEEE Trans. Softw. Eng. **3**, 85–93 (1977)