Multi-Multiway Cut Problem on Graphs of Bounded Branch Width

Xiaojie Deng, Bingkai Lin, and Chihao Zhang

Department of Computer Science, Shanghai Jiao Tong University
Email: {lvchaxj, kai314159, chihao.zhang}@gmail.com

Abstract. The Multi-Multiway Cut problem proposed by Avidor and Langberg[2] is a natural generalization of Multicut and Multiway Cut problems. That is, given a simple graph G and c sets of vertices $S_1, \cdots, S_c$, the problem asks for a minimum set of edges whose removal disconnects every pair of vertices in S_i for all $1 \leq i \leq c$. In [13], the authors asked whether the problem is polynomial time solvable for fixed c on trees. In this paper, we give both a logical approach and a dynamic programming approach to the Multi-Multiway Cut problem on graphs of bounded branch width, which is exactly the class of graphs with bounded treewidth. In fact, for fixed c and branch width k, we show that the Multi-Multiway Cut problem can be solved in linear time, thus affirmatively answer the question in [13].

1 Introduction

The Multi-Multiway Cut problem is defined as follows[2]:

Multi-Multiway Cut($\mathcal{G}$)
Input: A simple graph $G = (V, E) \in \mathcal{G}$ and c sets of vertices $S_1, S_2, \ldots, S_c$.
Problem: Find a set of edges $C \subseteq E$ with minimum cardinality whose removal disconnects every pair of vertices in each set S_i

The well-studied Multiway Cut problem is a special case of Multi-Multiway Cut when $c = 1$ and Multicut problem is a special case of Multi-Multiway Cut when every S_i contains exactly two vertices. Multiway Cut is NP-hard on general graphs [7] and Multicut is NP-hard even on trees (by a simple reduction from vertex cover to stars). Based on these two hardness results, [13] asked whether Multi-Multiway Cut is polynomial-time solvable on trees when c is a constant.

In this paper, we show that when $\mathcal{G}$ is the class of graphs of branch width at most k, Multi-Multiway Cut($\mathcal{G}$) can be solved in $O(k^{2k+2} \cdot 2^{2kc} \cdot |G|)$. The notion of branch width is equivalent to treewidth up to constant, which roughly measures how similar a graph is to a tree. We will present two algorithms.

The first one is using logical approach based on Courcelle's theorem, i.e., we will give a monadic second order formula characterizing Multi-Multiway Cut and Courcelle's theorem directly implies a linear algorithm for fixed c and k. However, the canonical algorithm behind Courcelle's theorem contains huge hidden constant and thus it is impractical. Our second algorithm is based on dynamic programming and can be viewed as a subtle way of applying Courcelle's theorem to specific problem, thus it is more efficient.

Related Work The Multicut problem is known to be APX-hard for $c \geq 3$ [7]. An $O(\log c)$-approximation algorithm based on the well-known region growing rounding technique was presented in [8]. From the parameterized complexity point of view, various parameters have been studied like solution size [4], cardinality k and solution size [14] [17], or treewidth of the input structure [9],[15].

The Multiway Cut problem is also known to be APX-hard for $\ell \geq 3$ [7], where $\ell = |S_1|$. A $(3/2 - 1/\ell)$-approximation algorithm was presented in [5]. Later Karger et al. [12] improved the approximation ratio to $(1.3438 - \varepsilon_m)$. In terms of exact algorithms, [17] gave an algorithm in $O(2^{(\frac{\ell-2}{\ell-1}k)}\ell T(n,m))$ time, where k is the solution size, $T(n,m) = O(\min(n^{2/3}, m^{1/2})m)$, n is the number of vertices and m is the number of edges in graph.

For Multi-Multiway Cut problem, Avidor and Langberg [2] presented an $O(\log k)$-approximation algorithm using the idea of region growing. [18] studied some variant of Multi-Multiway Cut on trees including the prize-collecting version.

In [13], Liu and Zhang gave a fixed-parameter tractable algorithm for Multi-Multiway Cut on trees in which the parameter is the size of solution and c is constant.

Independently, Kanj et al. proved a similar results in [11].

Our Result We present two linear algorithms for Multi-Multiway Cut with constant c and k, where k is the branch width of input graph. Our algorithms can be further extended to graphs with weight on edges and directed graph whose underlying undirected graph is of bounded branch width.

Organization of the Paper In Section 2, we introduce some necessary background and notations. In Section 3, we use Courcelle's Theorem to give an algorithm for Multi-Multiway Cut on graphs of bounded branch width. Next, in Section 4, we present the dynamic programming algorithm and finally conclude the paper in Secion 5.

2 Preliminaries

$\mathbb{N}$ denote the set of natural numbers. For a set S, the set of all k-element subsets of S is $[S]^k$. The power set of S is denoted by $\mathcal{P}(S) = \{X | X \subseteq S\}$. $|S|$ is the cardinality of S.

2.1 Graph

A graph is a pair $G = (V, E)$, where V is a finite set of vertices and $E \subseteq V^2$. The size of G is $|G| = |V| + |E|$. An edge $\{u, v\}$ is also written as uv. We also denote the edges set and vertices set of G as $E(G)$ and $V(G)$. A graph with labels is $G = (V, E, L_1, L_2, \cdots, L_l)$, where each $L_i \subseteq V$ is a label set. The union of two graph G and H is $G \cup H = (V(G) \cup V(H), E(G) \cup E(H))$.

2.2 Branch Decomposition

Definition 1 (Branch Decomposition). *Given a graph $G = (V, E)$, a branch decomposition is a pair (T, β), such that*

1 T is a binary tree with $|E|$ leaves and every inner node of T has two children.
2 β is a mapping from $V(T)$ to $\mathcal{P}(E)$ satisfying the following conditions:
  2.1 For each leaf $v \in V(T)$, there exists $e \in E(G)$ with $\beta(v) = \{e\}$, and there are no $v, u \in V(T), v \neq u$ such that $\beta(v) = \beta(u)$.
  2.2 For every inner node $v \in V(T)$ with children v_l, v_r, $\beta(v) = \beta(v_l) \cup \beta(v_r)$;

Definition 2 (Boundary). *Given a graph $G = (V, E)$, for every set $F \subseteq E$, the boundary $\partial F = \{v \mid v$ is incident to edges in F and $E \setminus F\}$.*

Definition 3 (Width of a Branch Decomposition). *Given a branch decomposition (T, β) of $G = (V, E)$, the width of this decomposition is* $\max\{|\partial\beta(v)| \mid v \in V(T)\}$.

The branch width $bw(G)$ of G is defined as the minimum width of all branch decompositions (T, β) for G.

Proposition 1. *[3] For any fixed k, there is a linear time algorithm that checks if a graph has branch width k and, if so, outputs a branch decomposition of minimum width.*

Branch width is related to another well-known graph parameter, treewidth. Indeed, they are equivalent up to constant. Let $tw(G)$ be the treewidth of graph G, then

Proposition 2. *[16]* $bw(G) \leq tw(G) + 1 \leq \max\{\frac{3}{2} \cdot bw(G), 2\}$.

Definition 4. *Given a branch decomposition (T, β) of $G = (V, E)$, for any $t \in V(T)$, let $\mathcal{V}_t = \{v \mid \exists u \in V(G), vu \in \beta(t)\}$, the subgraph $\mathcal{G}_t$ is*

$$\mathcal{G}_t = (\mathcal{V}_t, \beta(t))$$

.

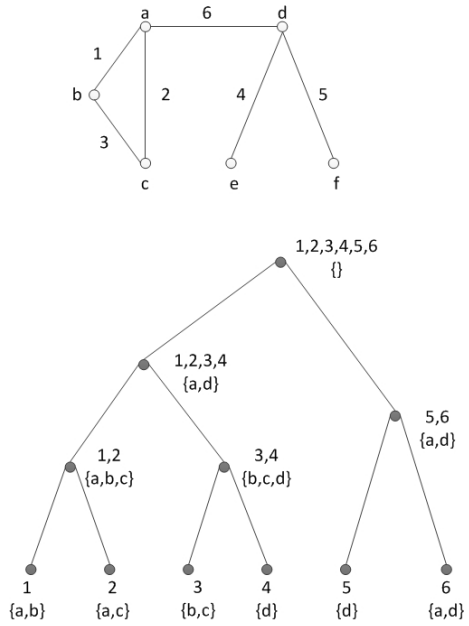

Fig. 1. A graph with a branch decomposition of width 3.

2.3 Logic

We use FO and MSO to denote First Order Logic and Monadic Second Order Logic respectively. The difference between these two logic is that FO only allows quantification over individual variables while MSO allows quantification over set variables. Furthermore, MSO can be extended to MSO_2. In this paper, MSO_2 is the logic that allows quantification over subset of edges in graph.

3 Logical Approach

We consider a Multi-Multiway Cut instance $(G, S_1, \cdots, S_c)$ where $G = (V, E)$ is a simple graph of branch width k. Many NP-hard problems on graphs have efficient solutions when the input graphs have bounded treewidth by the following theorem:

Theorem 1 (Courcelle's Theorem, Optimization Version). *[6,1] Given an* MSO *formula $\phi(U)$, there is an algorithm A satisfies that for any labeled graph $G = (V, E, L_1, \cdots, L_l)$ of treewidth k, A computes the set $U \subseteq V$ with minimum cardinality, such that $G \vDash \phi(U)$, with running time bounded by $f(k, |\phi(U)|)|G|$ for some computable function f.*

By Proposition 2, the branch width of a graph is equivalent to its treewidth up to some constant. Therefore Theorem 1 holds for graph with bounded branch

width. Furthermore, the optimization version of Courcelle's Theorem for MSO can be extend to MSO_2 formula by turning the original graph G into a labeled graph $I(G) = (V_I, E_I, L_V, L_E)$, where $V_I = V(G) \cup E(G)$, $L_V = V(G), L_E = E(G)$ and $E_I = \{\{v, e\} \mid v \in e\}$. Any MSO_2 formula about the original graph G can be translated into an MSO formula about $I(G)$. Since the graph $I(G)$ is also of bounded branch width, we have an optimization version of Courcelle's theorem for MSO_2. Readers can refer [10] for more detail.

Thus it is sufficient to write down an MSO_2 formula capturing Multi-Multiway Cut. We begin with a MSO_2 formula saying two vertices x, y are connected without edges in C:

$$\mathsf{conn}(x, y, C) = \forall X[X(x) \wedge (\forall u \forall v X(u) \wedge E(u, v) \wedge \neg C(u, v) \rightarrow X(v)) \rightarrow X(y)]$$

The Multi-Multiway Cut problem can be captured by an MSO_2 formula

$$\mathsf{mmcut}(C) = \bigwedge_{i=1}^{c} \forall x \forall y (S_i(x) \wedge S_i(y) \rightarrow \neg \mathsf{conn}(x, y, C))$$

Therefore for all $C \subseteq E(G), G \vDash \mathsf{mmcut}(C) \Leftrightarrow C$ is a multi-multiway cut of $(G, S_1, \cdots, S_c)$, thus we can solve Multi-Multiway Cut on bounded branch width graph via Courcelle's Theorem in time $f(k, c) \cdot |G|$ for some computable function f.

4 Dynamic Programming Approach

Let $(G, S_1, \ldots, S_c)$ be an instance of Multi-Multiway Cut and (T, β) be a branch decomposition of G of width k. It is convenient to view each $i \in [c]$ as a color, i.e., $v \in S_i$ means v is assigned color i. Note that a vertex v may be assigned with multiple colors (or no color) since we do not require $S_i \cap S_j = \varnothing$ for $i \neq j$. We use $col(v)$ to denote the set of colors v assigned. Thus a multi-multiway cut is an edge set whose removal disconnects all pairs of vertices with common color.

In fact, we will compute a table $C(t, Z)$ for every $t \in V(T)$ and Z. Here $Z = \{(X_i, Y_i) \mid i \in I\}$ for some index set I. $\{X_i \mid i \in I\}$ is a partition of $\partial\beta(t)$ and each Y_i is a set of colors. We say a set of edges $C \subseteq \beta(t)$ is *consistent with* Z if and only if

(1) C is a multi-multiway cut of $\mathcal{G}_t$.
(2) Let $H = (\mathcal{V}_t, \beta(t) \setminus C)$, i.e., the subgraph of $\mathcal{G}_t$ after removing C. Let $\{P_i \mid i \in I\}$ be the family of connected components in H such that $P_i \cap \partial\beta(t) \neq \varnothing$ for all $i \in I$. Then $X_i = P_i \cap \partial\beta(t)$ and $Y_i = \bigcup_{v \in P_i} col(v)$ for all $i \in I$.

It is easy to see that, for every multi-multiway cut C of $\mathcal{G}_t$, there is only one consistent Z. Intuitively, Z encodes colors exposed to external when removing C from $\mathcal{G}_t$.

The value of $C(t, Z)$ is a minimum edge set $C \subseteq \beta(t)$ that is consistent with Z, if there are more than one C consistent with Z with minimum cardinality,

then $C(t, Z)$ is arbitrary one of them. If no such C exists, then the value of $C(t, Z)$ is "Impossible". Indeed, then minimum multi-multiway cut of G is the one in $\{C(r, Z) \mid \text{ all possible } Z\}$ with minimum cardinality, where r is the root of T.

In the following, we will show how to compute $C(t, Z)$ recursively.

4.1 Computing $C(t, Z)$

If t is a leaf in T, the computation of $C(t, Z)$ is easy, otherwise let t_ℓ and t_r be its two children in T. $C(t, Z)$ is computed from some $C(t_\ell, Z_\ell)$ and $C(t_r, Z_r)$. We will use the following algorithm as a subroutine:

Merge Two Cuts

Input: (C_ℓ, Z_ℓ) and (C_r, Z_r), where Z_ℓ(resp. Z_r) is consistent with C_ℓ(resp. C_r)

Output: If $C_\ell \cup C_r$ is a multi-multiway cut of $\mathcal{G}_t$, compute the set Z consistent with $C_\ell \cup C_r$ on $\mathcal{G}_t$, otherwise return "Not Mergeable"

1 Let $Z_\ell = \{(X_i, Y_i) \mid i \in I\}, Z_r = \{(X_j, Y_j) \mid j \in J\}$ where I and J are two index sets.
2 Construct a bipartite graph $B = (I, J, E)$, for every $i \in I$ and $j \in J$, $ij \in E$ if and only if $X_i \cap X_j \neq \varnothing$.
3 Let $\{P_s \mid s \in S\}$ be family of connected components in B where S is an index set.
4 For every $s \in S$ and $p, q \in P_s$, if $Y_p \cap Y_q \neq \bigcup_{v \in X_p \cap X_q} col(v)$, return "Not Mergeable".
5 For $s \in S$:
  5.1 Let $X_s = \left(\bigcup_{p \in P_s} X_p\right) \cap \beta(t)$
  5.2 Let $Y_s = \left(\bigcup_{p \in P_s} Y_p\right)$
6 return $Z = \{(X_s, Y_s) \mid s \in S \text{ and } X_s \neq \varnothing\}$.

Algorithm 1: Merge Two Cuts

Step 4 in Algorithm 1 checks whether there are two vertices sharing some common color being connected after merging. If it is the case, the algorithm outputs "Not Mergeable", which means $C_\ell \cup C_r$ is not a multi-multiway cut of G. The size of bipartite graph B is $O(k^2)$ and all other operations are linear to the size of B, so this algorithm is in $O(k^2)$.

Fig 2 shows an example where $X_{u_1} \cap X_{v_1} = \{1\}$, $X_{u_3} \cap X_{v_2} = \{2\}$ and $X_{u_3} \cap X_{v_4} = \{3\}$; with the corresponding color sets $Y_{u_1} = \{white, black\}$, $Y_{u_3} = \{white, black, gray\}$, $Y_{v_1} = \{black, gray\}$ and $Y_{v_2} = \{white, black, gray\}$. Applying Algorithm 1 to this example, in the Step 4 of Algorithm 1, we can find u_3, v_2

in the same component of the bipartite graph, and $Y_{u_3} \cap Y_{v_2} = \{gray, white\}$, while $\bigcup_{v \in X_{u_3} \cap X_{v_2}} col(v) = \{gray\}$. The algorithm will return "Not Mergeable", since there are two vertices with the common color white being connected after merging.

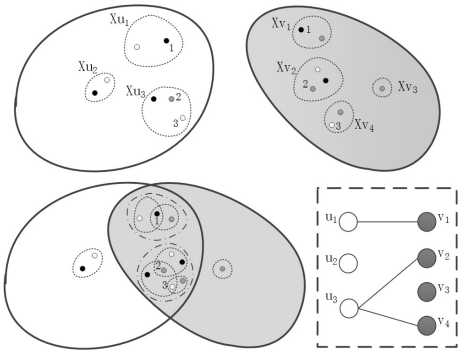

Fig. 2. Merge Two Pairs

To ease the presentation, we define a binary operation $\oplus$:

Definition 5. *$Z_\ell \oplus Z_r = Z$ if and only if Algorithm 1 output Z on inputs Z_ℓ and Z_r.*

Compute $C(t, Z)$

Input: $(G, S_1, \cdots, S_c)$ along with a branch decomposition (T, β) of width k

Output: Compute the table $C(t, Z)$

1 If t is a leaf in T. Let uv be the unique edge in $\beta(t)$. There are two cases:

(1.1) $Z = \{(X_1, Y_1)\}$ where $X_1 = \{u, v\}$. In this case $C(t, Z) = \varnothing$ if $col(u) \cap col(v) = \varnothing$ and $Y_1 = col(u) \cup col(v)$. Otherwise, $C(t, Z) =$"Impossible";

(1.2) $Z = \{(X_1, Y_1), (X_2, Y_2)\}$ where $X_1 = \{u\}$ and $X_2 = \{v\}$. In this case $C(t, Z) = \{uv\}$ if $Y_1 = col(u), Y_2 = col(v)$. Otherwise, $C(t, Z) =$"Impossible".

2 If t is not a leaf in T, let t_ℓ and t_r be its two children.

$$C(t, Z) = \arg\min_{|C|}\{C \mid C = C(t_\ell, Z_\ell) \cup C(t_r, Z_r) \text{ and } Z = Z_\ell \oplus Z_r\}$$

Algorithm 2: Compute $C(t, Z)$

We can now compute $C(t, Z)$ as follows: $C(t, Z)$ is the minimum edge set $C \subseteq \beta(t)$ such that $C = C(t_\ell, Z_\ell) \cup C(t_r, Z_r)$ for some Z_ℓ, Z_r satisfying $Z_\ell \oplus Z_r = Z$. We enumerate all such Z_ℓ and Z_r to compute $C(t, Z)$. Note that if one side of the union is "Impossible", then the result is also "Impossible". In summary, we use Algorithm 2 to compute $C(t, Z)$.

Putting all together, we have:

Theorem 2. *Given a* Multi-Multiway Cut *instance* $(G, S_1, \cdots, S_c)$ *along with a branch decomposition* (T, β) *of width* k*, the minimum multi-multiway cut can be computed in time* $O(k^{2k+2} \cdot 2^{2kc} \cdot |G|)$.

Proof. First note that, for a fixed k, the number of distinct Z is at most $k^k 2^{ck}$. This is because

(1) $|\beta(t)| \leq k$ and the number of partitions of a k-size set is upper bounded by k^k, and
(2) There are at most 2^c distinct sets of colors and each Y_i is one of them.

In Algorithm 2, $C(t, Z)$ is computed in a bottom-up style: once we know all $C(t_\ell, Z_\ell)$ and $C(t_r, Z_r)$, we can enumerate all of them to compute the corresponding $C(t, Z)$. We need to deal with at most $(k^k 2^{ck})^2$ pairs of Z_ℓ and Z_r and for each pair, we use Algorithm 1 to compute their merging results, and then take the minimum $C(t, Z)$ for each Z. After computing the table $C(t, Z)$, we choose the cut in $\{C(r, Z) \mid \text{ all possible } Z\}$ with minimum cardinality as the final answer, where r is the root of T. In all, we use $O(k^{2k+2} \cdot 2^{2kc} \cdot |G|)$ time. □

4.2 Proof of the Correctness

The correctness of Algorithm 1

Lemma 1. *If* C_ℓ *is a multi-multiway cut consistent with* Z_ℓ *on* $\mathcal{G}_{t_\ell}$*,* C_r *is a multi-multiway cut consistent with* Z_r *on* $\mathcal{G}_{t_r}$ *and* $Z_\ell \oplus Z_r = Z$*, then* $C := C_\ell \cup C_r$ *is a multi-multiway cut consistent with* Z *on* $\mathcal{G}_t$.

Proof. Assume $Z = \{(X_k, Y_k) \mid k \in K\}$ for index set K.

First, we need to show that if $Z_\ell \oplus Z_r = Z$, then C is a multi-multiway cut of $\mathcal{G}_t$. That is, there are no two vertices with common color connected in $\mathcal{G}_t$ after removing C. Let $\mathcal{P}_t = \{P_s \mid s \in S\}$ be the family of connected components in $\mathcal{G}_t$ after removing C, $\mathcal{P}_\ell = \{P_i \mid i \in I\}$ (resp. $\mathcal{P}_r = \{P_j \mid j \in J\}$) be the set of connected components in $\mathcal{G}_{t_\ell}$ (resp. $\mathcal{G}_{t_r}$) after removing C_ℓ (resp. C_r). It follows from the definition of branch decomposition that

$$P_s = \bigcup_{i \in I'} P_i \cup \bigcup_{j \in J'} P_j$$

for some index sets $I' \subseteq I$ and $J' \subseteq J$.

Thus if P_s contains two vertices with common color, Step 4 of Algorithm 1 will return "Not Mergeable", but this is impossible since $Z_\ell \oplus Z_r = Z$ means Algorithm 1 outputs Z on inputs Z_ℓ, Z_r.

It remains to verify $\{X_k \mid k \in K\}$ is exactly $\{P_s \cap \partial\beta(t) \mid s \in S \text{ and } P_s \cap \partial\beta(t) \neq \varnothing\}$ and Y_k is the set of colors appeared in the component containing X_k.

Let P_s be a component in $\mathcal{P}_t$ such that $P_s \cap \partial\beta(t) \neq \varnothing$. It follows from the definition of branch decomposition that

$$P_s = \bigcup_{i \in I'} P_i \cup \bigcup_{j \in J'} P_j$$

for some index sets $I' \subseteq I$ and $J' \subseteq J$.

Then $I' \cup J'$ is a connected component in the bipartite graph B constructed in Step 2. The corresponding $X_k := \left(\bigcup_{s \in I' \cup J'} X_s\right) \cap \beta(t)$. It is easy to verify that the correspondence is a bijection between two sets.

The consistency of $\{Y_k \mid k \in K\}$ follows from Step 5.2 of Algorithm 1 and consistency of $\{X_k \mid k \in K\}$ directly. □

Now we prove the correctness of the Algorithm 2.

Lemma 2. *$C(t, Z)$ computed in Algorithm 2 is the minimum cut that consistent with Z on $\mathcal{G}_t$.*

Proof. We apply induction on the branch decomposition tree.

If t is leaf, the correctness is obvious. Otherwise, t has two children t_ℓ and t_r in T. Suppose C is the minimum cut that consistent with Z on $\mathcal{G}_t$, Let $C_\ell = C \cap \beta(t_\ell)$ and $C_r = C \cap \beta(t_r)$. There is only one Z_ℓ (resp. Z_r) that C_ℓ (resp. Z_r) is consistent with. Since $C = C_\ell \cup C_r$, we have $Z_r \oplus Z_\ell = Z$ by the Algorithm 1.

Let $C'_\ell = C(t_\ell, Z_\ell)$, we have $|C'_\ell| \leq |C_\ell|$ by the induction; similarly let $C'_r = C(t_r, Z_r)$, we have $|C'_r| \leq |C_r|$. Thus $C'_\ell \cup C'_r$ is a cut consistent with $Z_\ell \oplus Z_r$. On the other hand, $C_\ell \cup C_r$ is also a cut consistent with $Z_\ell \oplus Z_r$. We have $|C'_\ell \cup C'_r| \leq |C_\ell \cup C_r| = |C| \leq |C'_\ell \cup C'_r|$. According to our algorithm, $C(t, Z)$ is a cut of minimum cardinality over all $C(t_\ell, Z_\ell) \cup C(t_r, Z_r)$ for $Z_\ell \oplus Z_r = Z$, therefore $|C(t, Z)| = |C|$. □

5 Conclusion

In this paper, we presented two linear algorithms for Multi-Multiway Cut problem with constant number of terminal sets on graphs of bounded branch width. The logical approach is straightforward and very easy to design, however, it relies on a canonical algorithm and thus not practical. Our second approach is somehow a refinement of the canonical algorithm to specific problem, the more subtle design of subproblem gains much efficiency.

6 Acknowledgements

The work is supported by NSF of China (61033002) and by Science and Technology Commission of Shanghai Municipality (11XD1402800).

References

1. S. Arnborg, J. Lagergren, and D. Seese. Easy problems for tree-decomposable graphs. *Journal of Algorithms*, 12(2):308–340, 1991.
2. A. Avidor and M. Langberg. The multi-multiway cut problem. *Theoretical Computer Science*, 377(1-3):35–42, 2007.
3. Hans L Bodlaender and Dimitrios M Thilikos. Constructive linear time algorithms for branchwidth. In *Automata, Languages and Programming*, pages 627–637. Springer, 1997.
4. Nicolas Bousquet, Jean Daligault, Stephan Thomassé, Anders Yeo, et al. A polynomial kernel for multicut in trees. In *26th International Symposium on Theoretical Aspects of Computer Science STACS 2009*, pages 183–194, 2009.
5. Gruia Călinescu, Howard Karloff, and Yuval Rabani. An improved approximation algorithm for multiway cut. In *Proceedings of the thirtieth annual ACM symposium on theory of computing*, pages 48–52. ACM, 1998.
6. Bruno Courcelle. Graph rewriting: An algebraic and logic approach. In *Handbook of Theoretical Computer Science, Volume B: Formal Models and Sematics (B)*, pages 193–242. Elsevier and MIT Press, 1990.
7. Elias Dahlhaus, David S. Johnson, Christos H. Papadimitriou, Paul D. Seymour, and Mihalis Yannakakis. The complexity of multiterminal cuts. *SIAM Journal on Computing*, 23(4):864–894, 1994.
8. Naveen Garg, Vijay V Vazirani, and Mihalis Yannakakis. Approximate max-flow min-(multi) cut theorems and their applications. *SIAM Journal on Computing*, 25(2):235–251, 1996.
9. Georg Gottlob and Stephanie Tien Lee. A logical approach to multicut problems. *Information Processing Letters*, 103(4):136–141, 2007.
10. Martin Grohe. Logic, graphs, and algorithms. *Logic and Automata–History and Perspectives*, pages 357–422, 2007.
11. Iyad Kanj, Guohui Lin, Tian Liu, Weitian Tong, Ge Xia, Jinhui Xu, Boting Yang, Fenghui Zhang, Peng Zhang, and Binhai Zhu. Algorithms for cut problems on trees. *Manuscript*, 2013.
12. David R Karger, Philip Klein, Cliff Stein, Mikkel Thorup, and Neal E Young. Rounding algorithms for a geometric embedding of minimum multiway cut. *Mathematics of Operations Research*, 29(3):436–461, 2004.
13. Hong Liu and Peng Zhang. On the generalized multiway cut in trees problem. *Journal of Combinatorial Optimization*, pages 1–13, 2012.
14. Dániel Marx. Parameterized graph separation problems. *Theoretical Computer Science*, 351(3):394–406, 2006.
15. Reinhard Pichler, Stefan Rümmele, and Stefan Woltran. Multicut algorithms via tree decompositions. In *Algorithms and Complexity*, pages 167–179. Springer, 2010.
16. Neil Robertson and Paul D Seymour. Graph minors. x. obstructions to tree-decomposition. *Journal of Combinatorial Theory, Series B*, 52(2):153–190, 1991.
17. Mingyu Xiao. Simple and improved parameterized algorithms for multiterminal cuts. *Theory of Computing Systems*, 46(4):723–736, 2010.
18. P. Zhang. Approximating generalized multicut on trees. *Computation and Logic in the Real World*, pages 799–808, 2007.