# Multicut Algorithms via Tree Decompositions*

Reinhard Pichler, Stefan Rümmele, and Stefan Woltran

Vienna University of Technology

**Abstract.** Various forms of multicut problems are of great importance in the area of network design. In general, these problems are intractable. However, several parameters have been identified which lead to fixed-parameter tractability (FPT). Recently, Gottlob and Lee have proposed the treewidth of the structure representing the graph and the set of pairs of terminal vertices as one such parameter. In this work, we show how this theoretical FPT result can be turned into efficient algorithms for optimization, counting, and enumeration problems in this area.

## 1 Introduction

Multicut problems [1] are of great importance in the area of network design – with applications to telecommunications, routing, VLSI design and transportation. An instance of a multicut problem is given by an undirected graph $G = (V, E)$ and a set $H$ of pairs of *terminal* vertices. A *solution* of the Edge Multicut (EMC) problem is a set of edges whose removal disconnects each terminal pair. In case of the Restricted (resp. Unrestricted) Vertex Multicut problem, the *solutions* are sets of non-terminal vertices (resp. of arbitrary vertices) whose removal disconnects each terminal pair. Each notion of solutions naturally gives rise to an optimization problem ("find the cardinality of a minimal solution"). Equipping, in addition, edges (resp. vertices) with some positive integer naturally leads to weighted versions of this optimization problem ("find the total weight of a minimal solution"). All these variants of multicut problems are intractable (the corresponding decision problems whether a solution with a given cardinality resp. total weight exists are NP-complete [2,3]), even for graphs of bounded treewidth [2,4]. For planar graphs with outer terminals and fixed cardinality $|H|$, EMC is solvable in polynomial-time [5].

An important approach in dealing with intractable problems is to search for *fixed-parameter tractability* (FPT), see e.g. [6]. Thereby one tries to confine the combinatorial explosion to certain problem *parameters*. More formally, we speak of FPT w.r.t. a parameter $k$, if a problem is solvable in time $f(k) \cdot n^{O(1)}$, where $n$ denotes the size of the input instance. The function $f$ is usually exponential but only depends on $k$. In case of multicut problems, various such parameters have been studied like solution size [7,8], cardinality $|H|$ plus solution size [9,10], $|H|$ plus the treewidth of the graph $G$ [11,12], or the treewidth of the structure representing both $G$ and $H$ [13]. The result in [13] was proved by showing

* Supported by the Austrian Science Fund (FWF), project P20704-N18, and the Vienna Science and Technology Fund (WWTF), project ICT08-028.

T. Calamoneri and J. Diaz (Eds.): CIAC 2010, LNCS 6078, pp. 167–179, 2010.

that the solutions to any of the above multicut problems can be described by a monadic second-order (MSO) formula over the structure representing $G$ and $H$. The FPT follows by an extension of Courcelle's Theorem [14] proved in [15]: optimization problems expressible by an MSO formula over the input structures are fixed-parameter tractable w.r.t. the treewidth of these structures. The proof of this result in [15] is constructive – yielding algorithms whose running time is non-elementary in terms of the number of quantifier alternations of the MSO formula. For the MSO formula in [13], we would thus end up with time and space requirements which are at least double exponential in the treewidth. The goal of our paper is to construct more efficient algorithms in case of bounded treewidth of the structure representing $G$ and $H$. Our main results are as follows (Due to space constraints, we omit proofs in this paper; full proofs are given in [16]):

- *Optimization Problem.* We present a new dynamic-programming based algorithm for computing the optimal value of the Edge Multicut problem which works by partitioning the vertices into disconnected parts. The resulting algorithm runs in time $O(2^{2w^* \log w^*} \cdot \|(G,H)\|)$, where $w^*$ denotes the treewidth of the input structure and $\|(G,H)\|$ denotes its size. Therefore this algorithm is in fixed-parameter linear time and single exponential in terms of the treewidth.
- *Counting and Enumeration.* We come up with a refinement of the optimization algorithm in order to *count* the number of optimal solutions and also to *enumerate* all optimal solutions. Determining the number of optimal solutions is possible in time $O(2^{2((w^*)^2+w^* \log w^*)} \cdot \|(G,H)\|)$. Moreover, the enumeration of all optimal solutions is feasible with a delay (i.e., the time needed to compute the first respectively the next solution) that also fits into this bound.

## 2 Preliminaries

We briefly recall some basic definitions on treewidth and multicut problems.

**Tree Decomposition and Treewidth.** A *tree decomposition* of a graph $G = (V,E)$ is a pair $(T,\chi)$, where $T$ is a tree and $\chi$ maps each node $n$ of $T$ (we use $n \in T$ as a shorthand below) to a *bag* $\chi(n) \subseteq V$ with the following properties: (1) for each $v \in V$, there is an $n \in T$, s.t. $v \in \chi(n)$; (2) for each $\{v,w\} \in E$, there is an $n \in T$, s.t. $v,w \in \chi(n)$; (3) for each $n_1,n_2,n_3 \in T$, s.t. $n_2$ lies on the path from $n_1$ to $n_3$, $\chi(n_1) \cap \chi(n_3) \subseteq \chi(n_2)$ holds.

A rooted tree decomposition $(T,\chi)$ is called *normalized* (or *nice*) [17], if (1) each $n \in T$ has at most two children; (2) for each $n \in T$ with two children $n_1,n_2$, $\chi(n) = \chi(n_1) = \chi(n_2)$; and (3) for each $n \in T$ with one child $n'$, $\chi(n)$ and $\chi(n')$ differ in exactly one element; in other words the symmetric set-difference of $\chi(n)$ and $\chi(n')$ is a singleton $\{v\}$ with $v \in V$.

The *width* of a tree decomposition is defined as the cardinality of its largest bag minus one. The *treewidth* of a graph $G$, denoted as $tw(G)$, is the minimum width over all tree decompositions of $G$. For arbitrary but fixed $w \geq 1$, it is feasible in linear time (with huge constants though) to decide if a graph has treewidth $\leq w$ and, if so, to compute a tree decomposition of width $w$ [18]. Moreover, there exist

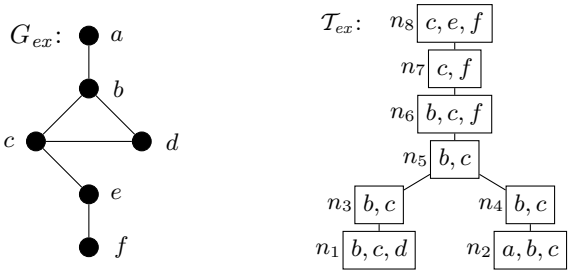


**Fig. 1.** Example graph $G_{ex}$ and normalized tree decomposition $\mathcal{T}_{ex}$ of $G_{ex}$

sophisticated heuristics to efficiently construct tree decompositions of small width [19,20]. W.l.o.g., we may assume that every tree decomposition is normalized, since this normalization can be obtained in linear time without increasing the width [17]. We thus distinguish between four types of nodes: (vertex) introduction (I), (vertex) removal (R), branch (B), and leaf (L) nodes. The first two types will usually be augmented with the vertex $v$ which is removed or added compared to the bag of the child node, i.e., ($v$R) and ($v$I).

*Example 1.* Figure 1 shows an example graph $G_{ex}$ together with a normalized tree decomposition $\mathcal{T}_{ex}$ of width 2. Note that the treewidth of $G_{ex}$ is indeed 2, since only trees (or forests) have treewidth = 1. Examples for node types are $n_1$ as (L) node, $n_3$ as ($d$R) node, $n_5$ as (B) node, and $n_6$, as ($f$I) node.

**Multicut Problems.** Since we are dealing with undirected graphs $G = (V, E)$, we consider terminal pairs $h \in H$ as sets $h \subseteq V$ of cardinality two. In [13], the treewidth $w^*$ of a structure representing both the graph $G$ and the set $H$ of pairs of terminals was introduced as a parameter of the multicut problems. Alternatively, $w^*$ can also be defined as $w^* = tw(G')$ with $G' = (V, E \cup H)$.

In this paper, we focus on the edge multicut problem (EMC) without weights. Given an instance $(G, H)$ of the EMC problem with $G = (V, E)$, we define a cut for $(G, H)$ as a set $F \subseteq E$ of edges, such that for each $\{h_1, h_2\} \in H$, $h_1$ and $h_2$ are disconnected in $(V, E \setminus F)$. $\text{Cuts}(G, H)$ will denote the set of cuts for $(G, H)$ and $\text{MinCuts}(G, H)$ is the set of minimum cuts for $(G, H)$, i.e. $\text{MinCuts}(G, H) = \{F \in \text{Cuts}(G, H) \colon \forall F' \in \text{Cuts}(G, H) : |F| \leq |F'|\}$.

*Example 2.* Consider the instance $(G_{ex}, H_{ex})$ of the EMC problem, where $G_{ex} = (V, E)$ is the graph in Figure 1 and $H_{ex} = \{\{a, b\}, \{c, d\}\}$. As is easily verified, the sets $\{\{a, b\}, \{b, c\}, \{c, d\}\}$ and $\{\{a, b\}, \{b, d\}, \{c, d\}\}$ are the minimum cuts for $(G_{ex}, H_{ex})$. In this simple example, we have $(V, E \cup H_{ex}) = (V, E)$. Hence, the treewidth $w^*$ of $(G_{ex}, H_{ex})$ is equal to the treewidth of $G_{ex}$. We thus can use the tree decomposition of $G_{ex}$ in Figure 1 also as a tree decomposition of $(G_{ex}, H_{ex})$.

For the remainder of the paper, we fix some notation: $G = (V, E)$ and $H$ always refer to the currently considered instance $(G, H)$ of a multicut problem and we denote the size of a reasonable representation by $\|(G, H)\|$. $\mathcal{T} = (T, \chi)$ denotes

a normalized tree decomposition of $(G, H)$ and $n_{root}$ refers to the root node of $T$. For any node $n \in T$, we denote by $T_n$ the subtree of $T$ rooted at $n$ and we use $\chi(T_n) := \bigcup_{m \in T_n} \chi(m)$. Hence, we always have $V = \chi(T_{n_{root}})$.

## 3 Dynamic Programming for Edge Multicut

In this section, we give a dynamic programming algorithm for the EMC problem to compute the number of edges a minimum cut requires. As well, we will sketch at the end of the section, how this algorithm can be used to compute one solution of a given EMC problem. However, we will also observe that the algorithm is not powerful enough to count or enumerate (without repetitions) all solutions of the given EMC problem. We will overcome this problem later in Section 4.

Our central objects in this section are so-called *colorings* (i.e., a special form of partitions), which partition subsets of $V$ into several sets (colors), such that no terminal pair $(h_1, h_2) \in H$ appears in the same color. The intended meaning of a coloring is thus a separation of the vertices into components. Indeed, such a separation induces a cut in a natural way. For our considerations in Section 4, note that vertices of the same color are not necessarily connected in such a graph.

**Definition 1.** *A* coloring *over a set $W \subseteq V$ of vertices is a partition $\mathcal{C} = \{C_1, \ldots, C_k\}$ of $W$, such that $u, v \in C_i$ implies $\{u, v\} \notin H$. Moreover, we denote the set of all colorings over $W$ by $\mathbf{C}(W)$. Finally, we define $\Gamma(\mathcal{C}) = E \cap \{\{u, v\} \colon u \in C_i, v \in C_j, i \neq j\}$ as the* cut induced by $\mathcal{C}$.

**Proposition 1.** $\mathrm{Cuts}(G, H) = \bigcup_{\mathcal{C} \in \mathbf{C}(V)} \{F \colon \Gamma(\mathcal{C}) \subseteq F \subseteq E\}$.

For our purposes, colorings related to the bags of the tree decomposition are of particular interest. For a coloring $\mathcal{C}$ over $\chi(n)$, it is natural to extend $\mathcal{C}$ to a coloring over $\chi(T_n)$. We will thus later be able to extend colorings over $\chi(n_{root})$ to colorings over $V$. The necessary concepts are introduced below.

**Definition 2.** *Given sets $W \subseteq W' \subseteq V$, $\mathcal{C} \in \mathbf{C}(W)$, and $\mathcal{C}' \in \mathbf{C}(W')$. We say that $\mathcal{C}'$* extends *$\mathcal{C}$, in symbols $\mathcal{C} \leq \mathcal{C}'$, in case $\mathcal{C} = \{C' \cap W \colon C' \in \mathcal{C}'\} \setminus \{\emptyset\}$.*

**Definition 3.** *Given a node $n \in T$, we call a coloring over $\chi(n)$ also a coloring of $n$ and we write $\mathbf{C}(n)$ instead of $\mathbf{C}(\chi(n))$. For $\mathcal{C} \in \mathbf{C}(n)$, we define*

$$\begin{aligned} \mathrm{ext}_n(\mathcal{C}) &= \{\mathcal{C}' \in \mathbf{C}(\chi(T_n)) \colon \mathcal{C} \leq \mathcal{C}'\} \\ \sigma_n(\mathcal{C}) &= \min\{|\Gamma(\mathcal{C}')| : \mathcal{C}' \in \mathrm{ext}_n(\mathcal{C})\} \\ \mathcal{E}_n(\mathcal{C}) &= \{\Gamma(\mathcal{C}') \colon \mathcal{C}' \in \mathrm{ext}_n(\mathcal{C}) \text{ s.t. } |\Gamma(\mathcal{C}')| = \sigma_n(\mathcal{C})\}. \end{aligned}$$

Intuitively, $\mathrm{ext}_n(\mathcal{C})$ yields all colorings over $\chi(T_n)$ which extend the coloring $\mathcal{C}$. Note that such an extension always exists: Indeed, for nodes $\{u, v\} \in H$ contained in $\chi(n)$ this is assured by definition of colorings. Moreover, for nodes $\{u, v\} \in H$ which appear below $n$, we just have to assign different colors to $u$ and $v$. Furthermore, $\sigma_n(\mathcal{C})$ gives the cardinality of the smallest induced cut among these extended colorings; and $\mathcal{E}_n(\mathcal{C})$ actually yields all these minimum cuts.

*Example 3.* The first three columns of the table in Figure 2 show the colorings $\mathcal{C} = \{C_1, C_2, \ldots\}$ for all nodes $n$ of our example tree decomposition, together with the value $\sigma_n(\mathcal{C})$. Due to space limitations, we will reuse this table in Section 4. Until then, the other columns can be ignored. Note that for the root node, $n_8$, coloring $\mathcal{C} = \{c, e, f\}$ with minimal value $\sigma_{n_8}(\mathcal{C}) = 3$ yields the minimum cuts of our example. In fact, $\mathcal{E}_{n_8}(\mathcal{C}) = \{\{\{a, b\}, \{b, c\}, \{c, d\}\}, \{\{a, b\}, \{b, d\}, \{c, d\}\}\}$.

The last observation in the above example leads us to the following lemma.

**Lemma 1.** *Let $\mathbf{C}_{min}$ be the set of all $\mathcal{C} \in \mathbf{C}(n_{root})$ such that no other coloring $\mathcal{C}'$ of $n_{root}$ satisfies $\sigma_{n_{root}}(\mathcal{C}') < \sigma_{n_{root}}(\mathcal{C})$. Then,* $\mathrm{MinCuts}(G, H) = \bigcup_{\mathcal{C} \in \mathbf{C}_{min}} \mathcal{E}_{n_{root}}(\mathcal{C})$.

In other words, to compute the size of a minimum cut, it is sufficient to derive the values $\sigma_{n_{root}}(\mathcal{C})$ of all colorings $\mathcal{C} \in \mathbf{C}(n_{root})$. However, we do not want to compute $\sigma_{n_{root}}(\mathcal{C})$ directly with the help of Definition 3, but establish $\sigma_n(\mathcal{C})$ for all nodes $n \in T$ in a bottom-up fashion. We use the following concept.

**Definition 4.** *For $\mathcal{C} \in \mathbf{C}(n)$ with $n \in T$, define $\pi_n(\mathcal{C})$ recursively as follows:*

- *If $n$ is an* (L) *node, $\pi_n(\mathcal{C}) = |\Gamma(\mathcal{C})|$;*
- *If $n$ is a* ($v$R) *node with child $n'$, $\pi_n(\mathcal{C}) = \min\{\pi_{n'}(\mathcal{C}') \colon \mathcal{C}' \in \mathbf{C}(n'), \mathcal{C} \leq \mathcal{C}'\}$;*
- *If $n$ is a* ($v$I) *node with child $n'$, $\pi_n(\mathcal{C}) = \pi_{n'}(\mathcal{C}') + |\{u \colon \{u, v\} \in \Gamma(\mathcal{C})\}|$, where $\mathcal{C}' \in \mathbf{C}(n')$ is the unique coloring with $\mathcal{C}' \leq \mathcal{C}$;*
- *If $n$ is a* (B) *node with children $n', n''$, $\pi_n(\mathcal{C}) = \pi_{n'}(\mathcal{C}) + \pi_{n''}(\mathcal{C}) - |\Gamma(\mathcal{C})|$.*

**Lemma 2.** *For any node $n \in T$ and any $\mathcal{C} \in \mathbf{C}(n)$, $\sigma_n(\mathcal{C}) = \pi_n(\mathcal{C})$.*

This lemma (together with Definition 4) naturally yields an algorithm to determine the size of an optimal solution by computing $\pi_n(\mathcal{C})$ for all colorings $\mathcal{C}$ of all nodes $n$ in the tree decomposition. We start by determining $\pi_n(\mathcal{C})$ for all colorings $\mathcal{C}$ of leaf nodes $n$. For all remaining nodes $n \in T$, the values $\pi_n(\mathcal{C})$ of all colorings $\mathcal{C}$ are computed in a bottom-up fashion along the tree decomposition. Finally, we obtain the cardinality of the minimum cuts as the minimal value $\pi_{n_{root}}(\mathcal{C})$ among the colorings $\mathcal{C}$ at the root node $n_{root}$.

*Example 4.* We illustrate some aspects of this algorithm on our example problem using Figure 2. Consider, for instance, coloring $\mathcal{C} = \{\{b\}, \{c\}\}$ for node $n_3$. We have two colorings for $n_1$ (the child node of $n_3$) which extend $\mathcal{C}$, namely, $\mathcal{C}' = \{\{b, d\}, \{c\}\}$ and $\mathcal{C}'' = \{\{b\}, \{c\}, \{d\}\}$. We know $\sigma_{n_1}(\mathcal{C}') = \pi_{n_1}(\mathcal{C}') = |\Gamma(\mathcal{C}')| = 2$, $\sigma_{n_1}(\mathcal{C}'') = \pi_{n_1}(\mathcal{C}'') = |\Gamma(\mathcal{C}'')| = 3$, and thus we select the smaller number and assign $\pi_{n_3}(\mathcal{C}) = 2$. For the branch node $n_5$, consider coloring $\mathcal{C} = \{\{b\}, \{c\}\}$. In a bottom-up traversal, we already know $\pi_{n_3}(\mathcal{C}) = \pi_{n_4}(\mathcal{C}) = 2$, and we clearly have $|\Gamma(\mathcal{C})| = 1$. Thus, $\pi_{n_5}(\mathcal{C}) = 2 + 2 - 1 = 3$, which equals $\sigma_{n_5}(\mathcal{C})$ as expected.

**Theorem 1.** *Given an instance $(G, H)$ of EMC and a tree decomposition of $(G, H)$ of width $w^*$, the cardinality $|F|$ of the solutions $F \in \mathrm{MinCuts}(G, H)$ can be computed in time $O(2^{2w^* \log w^*} \cdot \|(G, H)\|)$, assuming unit cost for arithmetic operations.*

| | $C_1$ | $C_2$ | $C_3$ | $\sigma$ | $\Delta_1 : \sigma : \#$ | $\Delta_2 : \sigma : \#$ |
|---|---|---|---|---|---|---|
| $n_1$ | $\{b, c\}$ | $\{d\}$ | | 2 | $\{\{C_1, C_2\}\} : 2 : 1$ | |
| | $\{b, d\}$ | $\{c\}$ | | 2 | $\{\{C_1, C_2\}\} : 2 : 1$ | |
| | $\{b\}$ | $\{c\}$ | $\{d\}$ | 3 | $\{\{C_2, C_3\}\} : 3 : 1$ | |
| $n_2$ | $\{a\}$ | $\{b, c\}$ | | 1 | $\{\{C_1, C_2\}\} : 1 : 1$ | |
| | $\{a, c\}$ | $\{b\}$ | | 2 | | |
| | $\{a\}$ | $\{b\}$ | $\{c\}$ | 2 | $\{\{C_1, C_2\}\} : 2 : 1$ | |
| $n_3$ | $\{b, c\}$ | | | 2 | $\emptyset : 2 : 1$ | |
| | $\{b\}$ | $\{c\}$ | | 2 | $\{\{C_1, C_2\}\} : 2 : 1$ | $\emptyset : 3 : 1$ |
| $n_4$ | $\{b, c\}$ | | | 1 | $\emptyset : 1 : 1$ | |
| | $\{b\}$ | $\{c\}$ | | 2 | $\emptyset : 2 : 1$ | |
| $n_5$ | $\{b, c\}$ | | | 3 | $\emptyset : 3 : 1$ | |
| | $\{b\}$ | $\{c\}$ | | 3 | $\{\{C_1, C_2\}\} : 3 : 1$ | $\emptyset : 4 : 1$ |
| $n_6$ | $\{b, c, f\}$ | | | 3 | | |
| | $\{b\}$ | $\{c, f\}$ | | 3 | | |
| | $\{b, c\}$ | $\{f\}$ | | 3 | $\emptyset : 3 : 1$ | |
| | $\{b, f\}$ | $\{c\}$ | | 3 | | |
| | $\{b\}$ | $\{c\}$ | $\{f\}$ | 3 | $\{\{C_1, C_2\}\} : 3 : 1$ | $\emptyset : 4 : 1$ |
| $n_7$ | $\{c, f\}$ | | | 3 | | |
| | $\{c\}$ | $\{f\}$ | | 3 | $\emptyset : 3 : 2$ | |
| $n_8$ | $\{c, e, f\}$ | | | 3 | $\emptyset : 3 : 2$ | |
| | $\{c\}$ | $\{e, f\}$ | | 4 | $\emptyset : 4 : 2$ | |
| | $\{c, e\}$ | $\{f\}$ | | 4 | $\emptyset : 4 : 2$ | |
| | $\{c, f\}$ | $\{e\}$ | | 5 | | |
| | $\{c\}$ | $\{e\}$ | $\{f\}$ | 5 | $\emptyset : 5 : 2$ | |

**Fig. 2.** Colorings of the tree decomposition of Example 2

The algorithm presented above allows us to compute a minimum cut $F$ in a simple postprocessing step. This can be done by a recursive procedure which proceeds by a top-down traversal of the tree decomposition. This procedure takes as input a node $n$ and a coloring $\mathcal{C}$ at $n$ and returns a set of edges (which are part of the cut $F$). Initially, we choose an arbitrary coloring $\mathcal{C}$ with minimal value $\sigma_{n_{root}}(\mathcal{C})$ among the colorings for $n_{root}$. The recursion starts with the root node $n_{root}$ and this coloring $\mathcal{C}$. This initial call to the recursion returns a minimum cut. Depending on the type of each node $n$, the following action is performed.

(a) in case $n$ is a ($v$I)-node, there is a unique coloring $\mathcal{C}'$ for the child node $n'$ of $n$, such that $\mathcal{C}' \leq \mathcal{C}$. We continue the recursion with $n'$ and $\mathcal{C}'$. Let the return value for $(n', \mathcal{C}')$ be $\Gamma'$. Then, the return value for $(n, \mathcal{C})$ is $\Gamma' \cup \Gamma(\mathcal{C})$;
(b) in case $n$ is a ($v$R)-node with child node $n'$, we choose a coloring $\mathcal{C}'$ for $n'$ with $\mathcal{C} \leq \mathcal{C}'$, such that $\pi_{n'}(\mathcal{C}') = \pi_n(\mathcal{C})$. We continue the recursion with $n'$ and $\mathcal{C}'$ and the return value for $(n', \mathcal{C}')$ is also the return value for $(n, \mathcal{C})$;
(c) in case $n$ is a (B)-node with children $n'$ and $n''$, we continue the recursion for both children with $n'$ and $\mathcal{C}$, resp. with $n''$ and $\mathcal{C}$. The return value for $(n, \mathcal{C})$ is the union of the return values of these two calls.
(d) for a leaf node $n$, we just return $\Gamma(\mathcal{C})$.

*Example 5.* For our example problem, we thus have to start with the coloring $\mathcal{C} = \{\{c, e, f\}\}$ of $n_8$. $\mathcal{C}$ extends coloring $\{\{c, f\}\}$ of $n_7$, which is an (R)-node. We have to choose a coloring of $n_6$ which extends $\{\{c, f\}\}$. Two such colorings exist, $\{\{b, c, f\}\}$ and $\{\{b\}, \{c, f\}\}$, and both have the same $\pi$-value as $\{\{c, f\}\}$. Let us select $\mathcal{C}' = \{\{b\}, \{c, f\}\}$, We then have to proceed with $\mathcal{C}'' = \{\{b\}, \{c\}\}$ at $n_5$, which is a (B)-node. At the (B) node, we proceed as follows:

We first continue with $\mathcal{C}''$ in $n_3$ which is a ($d$R)-node. There are two colorings of $n_1$ which extend $\mathcal{C}''$, namely $\{\{b, d\}, \{c\}\}$ and $\{\{b\}, \{c\}, \{d\}\}$. However, only the former has the same $\pi$-value as $\mathcal{C}''$ and is thus selected. Since we are now at a leaf node, we return the induced cut $\Gamma_1 = \{\{b, c\}, \{c, d\}\}$ of $\{\{b, d\}, \{c\}\}$.

Now we go into the second recursion with $\mathcal{C}''$ in $n_4$ which is an ($a$R)-node. There are two colorings for the child $n_2$ which extend $\mathcal{C}''$, namely $\{\{a, c\}, \{b\}\}$ and $\{\{a\}, \{b\}, \{c\}\}$. Both have the same $\pi$-value, so we can use either of them. Let us select $\{\{a, c\}, \{b\}\}$; we return its induced cut $\Gamma_2 = \{\{a, b\}, \{b, c\}\}$.

Back at the branch node, we have to combine the return values and obtain $\Gamma = \Gamma_1 \cup \Gamma_2 = \{\{a, b\}, \{b, c\}, \{c, d\}\}$ which is the return value at $n_6$. For the return value here, we observe $\Gamma \cup \Gamma(\mathcal{C}') = \Gamma$ and thus return $\Gamma$ which is passed on unchanged by (R)-node $n_7$. We compute the final return value at $n_8$ as $\Gamma \cup \Gamma(\mathcal{C}) = \Gamma \cup \emptyset = \Gamma$. Indeed, $\Gamma$ is a minimum cut of our problem.

As this example shows, our postprocessing step would have yielded the same solution, if we had selected $\{\{a\}, \{b\}, \{c\}\}$ instead of $\{\{a, c\}, \{b\}\}$ at node $n_2$. This is due to the fact that our data model does not guarantee that two distinct colorings $\mathcal{C}, \mathcal{C}'$ of some node $n$, satisfy $\mathcal{E}_n(\mathcal{C}) \cap \mathcal{E}_n(\mathcal{C}') = \emptyset$. This, in particular, causes problems for giving a general enumeration algorithm (it would yield duplicate results) and is also a serious obstacle for counting the number of solutions.

## 4 Counting and Enumeration

In the previous section, the key objects to maintain during the bottom-up traversal of the tree decompositions were the colorings $\mathcal{C} \in \mathbf{C}(n)$ for each node $n \in T$. As we have seen, it is possible that two distinct colorings $\mathcal{C}_1, \mathcal{C}_2 \in \mathbf{C}(n)$ share some solutions, i.e. $\mathcal{E}_n(\mathcal{C}_1) \cap \mathcal{E}_n(\mathcal{C}_2) = \emptyset$ does not necessarily hold. The reason is, that we guarantee for two *distinct colors* $C_i, C_j \in \mathcal{C}$, only that the cut *disconnects* all vertices $v \in C_i$ from the vertices $w \in C_j$. No statement was made about the *connectedness of vertices contained in the same partition* $C_i$. We now modify our approach so as to guarantee this connectedness.

**Definition 5.** *A* parsimonious coloring *(*pcoloring*, for short) over a set $W \subseteq V$ of vertices is a tuple $(\mathcal{C}, \Delta)$, where $\mathcal{C} \in \mathbf{C}(W)$ and $\Delta$ is a set of unordered pairs $\{C_i, C_j\}$ of distinct colors $C_i, C_j \in \mathcal{C}$, such that for each $\{u, v\} \in H$ it holds that $\{C_i, C_j\} \in \Delta$, whenever $u \in C_i$ and $v \in C_j$. $\Delta$ is called the* disconnection relation *of the pcoloring $\mathcal{P} = (\mathcal{C}, \Delta)$. We shall use $\mathcal{P}_c$ to identify the coloring $\mathcal{C}$ and $\mathcal{P}_d$ to identify the disconnection relation $\Delta$. Moreover, we denote by $\mathbf{P}(W)$ the set of all pcolorings over $W$ and write $\mathbf{P}(n)$ as a shorthand for $\mathbf{P}(\chi(n))$.*

Given $\mathcal{P} \in \mathbf{P}(n)$, $\mathcal{P}_c$ is simply a coloring of $n$. Ultimately, we are only interested in colorings $\mathcal{P}_c$, s.t. (in some extension of $\mathcal{P}_c$) the vertices in each color are indeed connected in the subgraph of $G$ induced by the vertices $\chi(T_n)$. In contrast to the previous section, we now allow that two different colors $C_i, C_j$ get *melted* to a single color during the bottom-up traversal of $T$; this happens, e.g., at a ($v$I)-node if $v$ has edges to some vertices $v_i \in C_i$ and $v_j \in C_j$ and if none of these edges is added to the cut. However, if for some terminal pair $\{h_1, h_2\} \in H$, we have $h_1 \in C_i$ and $h_2 \in C_j$, then these colors $C_i, C_j$ must never get melted as the bottom-up traversal continues. The very purpose of the second component $\mathcal{P}_d$ of $\mathcal{P}$ is to keep track of such pairs which must not get melted. Clearly, for each coloring $\mathcal{C}$, there exists a relation $\Delta$, s.t. $(\mathcal{C}, \Delta)$ is a pcoloring. Below, we overload the concepts from Definitions 2 and 3 for pcolorings.

**Definition 6.** *Let $W \subseteq W' \subseteq V$, $\mathcal{P} \in \mathbf{P}(W)$ and $\mathcal{P}' \in \mathbf{P}(W')$. Then, $\mathcal{P} \leq \mathcal{P}'$, if $\mathcal{P}_c \leq \mathcal{P}'_c$ and $\mathcal{P}_d = \{\{C_i \cap W, C_j \cap W\} \colon \{C_i, C_j\} \in \mathcal{P}'_d\} \setminus \{\{\emptyset, \cdot\}, \{\cdot, \emptyset\}\}$.*

For a node $n \in T$, let us denote by $G_n$ the subgraph induced by the vertices $\chi(T_n)$, i.e. $G_n = (\chi(T_n), E_n)$ where $E_n = E \cap \{\{v, u\} \colon v, u \in \chi(T_n)\}$. Moreover, for a graph $G_n$, let $\kappa(G_n)$ denote the set of connected components of $G_n$, and, for a set of edges $F$, let $G_n \setminus F = (\chi(T_n), E_n \setminus F)$ as expected.

**Definition 7.** *Let $\Delta(\mathcal{C}) = \{\{C_i, C_j\} \subseteq \mathcal{C} \colon \{u, v\} \in H, u \in C_i, v \in C_j\}$ for coloring $\mathcal{C}$. Given a node $n \in T$ and $\mathcal{P} \in \mathbf{P}(n)$, we define*

$$\mathrm{ext}_n(\mathcal{P}) = \{\mathcal{P}' \in \mathbf{P}(\chi(T_n)) \colon \mathcal{P} \leq \mathcal{P}', \kappa(G_n \setminus \Gamma(\mathcal{P}'_c)) = \mathcal{P}'_c, \mathcal{P}'_d = \Delta(\mathcal{P}'_c)\}.$$

*$\mathcal{P}$ is called* valid (in $n$) *if* $\mathrm{ext}_n(\mathcal{P}) \neq \emptyset$.

*The remaining two concepts are now only defined for valid pcolorings.*

$$\sigma_n(\mathcal{P}) = \min\{|\Gamma(\mathcal{P}'_c)| : \mathcal{P}' \in \mathrm{ext}_n(\mathcal{P})\}$$
$$\mathcal{E}_n(\mathcal{P}) = \{\Gamma(\mathcal{P}'_c) \colon \mathcal{P}' \in \mathrm{ext}_n(\mathcal{P}) \text{ s.t. } |\Gamma(\mathcal{P}'_c)| = \sigma_n(\mathcal{P})\}.$$

The modified definition of extensions guarantees that the vertices in each color are indeed connected in the subgraph of $G$ induced by the vertices $\chi(T_n)$ and that this connectedness is still fulfilled after removing the edges contained in the cut. This property is ensured by the condition $\kappa(G_n \setminus \Gamma(\mathcal{P}'_c)) = \mathcal{P}'_c$ in the definition of $\mathrm{ext}_n(\mathcal{P})$. The condition $\mathcal{P}'_d = \Delta(\mathcal{P}'_c)$ ensures that $\mathcal{P}'_d$ contains only disconnection pairs that are justified by some terminal pair in $H$. As a consequence, $\mathcal{P}'_d$ is uniquely determined by $\mathcal{P}'_c$, which will be crucial for Lemma 4 below. Of course, $\mathrm{ext}_n(\mathcal{P})$ may be empty. For instance, consider the coloring $\mathcal{C} = \{\{a, c\}, \{b\}\}$ at node $n_2$ of our running example. Since $b$ is in a different color than $a$ and $c$, both edges $(a, b)$ and $(b, c)$ are contained in the induced cut $\Gamma(\mathcal{C})$. But then $\{a, c\}$ is disconnected in the corresponding subgraph $G_{n_2} \setminus \Gamma(\mathcal{C})$. Hence, there exists no $\Delta$ to turn $(\mathcal{C}, \Delta)$ into a valid pcoloring. Actually, all rows in Figure 2 where the last two columns are empty contain colorings that cannot be part of a valid pcoloring. Analogously to Lemma 1, particular pcolorings for $n_{root}$ characterize the minimum cuts of a given problem instance.

**Lemma 3.** *Let* $\mathbf{P}_{min}$ *be the set of all valid pcolorings* $\mathcal{P}$ *of* $n_{root}$*, for which there exists no other valid pcoloring* $\mathcal{P}'$ *of* $n_{root}$*, such that* $\sigma_{n_{root}}(\mathcal{P}') < \sigma_{n_{root}}(\mathcal{P})$*. Then,* $\mathrm{MinCuts}(G,H) = \bigcup_{\mathcal{P}\in\mathbf{P}_{min}} \mathcal{E}_{n_{root}}(\mathcal{P})$.

**Lemma 4.** *Given a node* $n \in T$ *and distinct valid pcolorings* $\mathcal{P}_1$ *and* $\mathcal{P}_2$ *of* $n$*, it holds that* $\mathcal{E}_n(\mathcal{P}_1) \cap \mathcal{E}_n(\mathcal{P}_2) = \emptyset$.

Next, we define two operations on pcolorings which we need for the bottom-up traversal of $T$. The *melting* operation is denoted as $\mathcal{P}^S$. It takes a pcoloring $\mathcal{P}$ and some set $S$ of vertices and melts all colors containing at least one vertex from $S$ to a single color. This operation will be needed at ($v$I)-nodes where all colors containing some vertex $v_i$ adjacent to $v$ get melted into a single color (unless we add the edge connecting $v$ and $v_i$ to the cut). The second operation is the *composition* of two pcolorings $\mathcal{P}, \mathcal{P}'$, denoted as $\mathcal{P} \cup \mathcal{P}'$. Intuitively, the first component of $\mathcal{P} \cup \mathcal{P}'$ is obtained as the connected components of a graph of which we know that all vertices jointly occurring in some color of $\mathcal{P}$ or $\mathcal{P}'$ are connected. The composition operation will be needed at (B)-nodes. For both operations, the disconnection relation has to be adapted correspondingly. We first introduce the following notation for a pcoloring $\mathcal{P} \in \mathbf{P}(W)$ and a set $S \subseteq W \subseteq V$:

$$\mathcal{P}|_S = \{C \in \mathcal{P}_c \colon C \cap S \neq \emptyset\} \quad \text{and} \quad [\mathcal{P}, S] = \textstyle\bigcup_{C\in\mathcal{P}|_S} C.$$

Moreover, for a vertex $u$ we write $[\mathcal{P}, u]$ instead of $[\mathcal{P}, \{u\}]$ which simply denotes the color $C \in \mathcal{P}_c$ with $u \in C$.

**Definition 8.** *Let* $W \subseteq V$*,* $\mathcal{P} \in \mathbf{P}(W)$ *and* $S \subseteq W$*. Then* $\mathcal{P}^S = (\mathcal{C}, \Delta)$ *where* $\mathcal{C} = \{[\mathcal{P}, S]\} \cup (\mathcal{P}_c \setminus \mathcal{P}|_S)$ *and*

$$\Delta = \big\{\{[\mathcal{P}, S], C'\} \colon C' \in \mathcal{P}_c \setminus \mathcal{P}|_S,\, C \in \mathcal{P}|_S,\, \{C, C'\} \in \mathcal{P}_d\big\} \cup \big\{\{C, C'\} \colon C, C' \in \mathcal{C}, \{C, C'\} \in \mathcal{P}_d\big\}.$$

**Definition 9.** *Let* $W \subseteq V$*,* $\mathcal{P}, \mathcal{P}' \in \mathbf{P}(W)$*, and for each* $w \in W$*, let* $C_w$ *be the smallest set such that* $w \in C_w$ *and, for each* $u \in C_w$*,* $[\mathcal{P}, u] \cup [\mathcal{P}', u] \subseteq C_w$*. We define* $\mathcal{P} \cup \mathcal{P}' = (\mathcal{C}, \Delta)$ *where* $\mathcal{C} = \{C_w \colon w \in W\}$ *and*

$$\Delta = \{\{C_1, C_2\} \colon C_1, C_2 \in \mathcal{C}, C_1 \neq C_2, C_1' \subseteq C_1, C_2' \subseteq C_2, \{C_1', C_2'\} \in \mathcal{P}_d \cup \mathcal{P}_d'\}.$$

For $S = \emptyset$, or a singleton $S$, we get $\mathcal{P}^S = \mathcal{P}$ as expected. Note that $\mathcal{P}^S$ is possibly *not* a pcoloring. This is the case if $h \subseteq [\mathcal{P}, S]$ for some terminal pair $h \in H$. Moreover, for a *valid* pcoloring $\mathcal{P}$, the pcoloring $\mathcal{P}^S$ may be *invalid*, if $(C_1 \cup C_2) \subseteq [\mathcal{P}, S]$ for some pair $\{C_1, C_2\} \in \mathcal{P}_d$. Likewise, $\mathcal{P} \cup \mathcal{P}'$ may be not a pcoloring. This is the case if a pair $\{h_1, h_2\}$ of terminal vertices ends up in a joint color. Just consider the simple situation that $\mathcal{P}_c$ contains a color $\{h_1, v\}$ and $\mathcal{P}_c'$ contains a color $\{h_2, v\}$ for some vertex $v$. It may also happen that two valid pcolorings $\mathcal{P}$ and $\mathcal{P}'$ lead to an invalid pcoloring $\mathcal{P} \cup \mathcal{P}'$ if $\mathcal{P} \cup \mathcal{P}'$ contains a color $C$, s.t. $C_1 \cup C_2 \subseteq C$ for some pair $\{C_1, C_2\}$ in $\mathcal{P}_d$ or $\mathcal{P}_d'$.

Next, we define relations between pcolorings over adjacent nodes in the tree decomposition. These relations will be used to traverse the tree decomposition.

**Definition 10.** *Let $n \in T$, $\mathcal{P} \in \mathbf{P}(n)$, $\mathcal{P}' \in \mathbf{P}(n')$, and $\mathcal{P}'' \in \mathbf{P}(n'')$. Then, the following pairs are contained in relation $\prec_n$.*

1. *$\mathcal{P}' \prec_n \mathcal{P}$, if $n$ is a (vI)-node with child $n'$, and there exists a set $S \subseteq \chi(n') \cap \{u \colon \{v,u\} \in E\}$, such that $(\mathcal{P}')^S \leq \mathcal{P}$, $[\mathcal{P}', S] \cup \{v\} = [\mathcal{P}, v]$, and for each $\{C', D'\} \in (\mathcal{P}')_d$, there is a $\{C, D\} \in \mathcal{P}_d$ with $C' \subseteq C$ and $D' \subseteq D$.*
2. *$\mathcal{P}' \prec_n \mathcal{P}$, if $n$ is an (R)-node with child $n'$, and $\mathcal{P} \leq \mathcal{P}'$.*
3. *$(\mathcal{P}', \mathcal{P}'') \prec_n \mathcal{P}$, if $n$ has two children $n', n''$, $\mathcal{P} = \mathcal{P}' \cup \mathcal{P}''$, and for each $\{C', D'\} \in (\mathcal{P}')_d \cup (\mathcal{P}'')_d$, there is a $\{C, D\} \in \mathcal{P}_d$ with $C' \subseteq C$ and $D' \subseteq D$.*

As already mentioned above, some colorings at a child node may get melted at (I)-nodes or (B)-nodes. For a ($v$I)-node, $S$ in the above definition denotes an arbitrarily chosen subset of all vertices adjacent to $v$, s.t. all colors containing a vertex in $S$ are then melted into a single color $C_i$ on the transition from $\mathcal{P}'$ to $\mathcal{P}$. The neighbors of $v$ not selected in $S$ may end up in a different color $C_j$. Implicitly, the edges connecting $v$ with these neighbors are thus added to the cut. For (B)-nodes, the condition $\mathcal{P} = \mathcal{P}' \cup \mathcal{P}''$ forces $\mathcal{P}' \cup \mathcal{P}''$ to be a pcoloring. Moreover, the existence of $\{C', D'\} \in \mathcal{P}_d$ with $C \subseteq C'$ and $D \subseteq D'$ makes sure that no "disconnected pair" in either $\mathcal{P}'_d$ or $\mathcal{P}''_d$ gets melted into a single color.

**Lemma 5.** *If $\mathcal{P}' \in \mathbf{P}(n')$ is valid in $n'$ (and $\mathcal{P}'' \in \mathbf{P}(n'')$ is valid in $n''$), then each $\mathcal{P} \in \mathbf{P}(n)$ with $\mathcal{P}' \prec_n \mathcal{P}$ (resp. $(\mathcal{P}', \mathcal{P}'') \prec_n \mathcal{P}$) is valid in $n$. If $\mathcal{P} \in \mathbf{P}(n)$ is valid in $n$, then there exists $\mathcal{P}' \in \mathbf{P}(n')$ (and $\mathcal{P}'' \in \mathbf{P}(n'')$) with $\mathcal{P}' \prec_n \mathcal{P}$ (resp. $(\mathcal{P}', \mathcal{P}'') \prec_n \mathcal{P}$), s.t. $\mathcal{P}'$ is valid in $n'$ (and $\mathcal{P}''$ is valid in $n''$).*

Similarly to Section 3, we can compute $\sigma_n(\mathcal{P})$ for valid pcolorings $\mathcal{P}$ by a bottom-up traversal of $T$. The cases of ($v$I) and (B) nodes are slightly more complicated, since different pcolorings might collapse into the same pcoloring.

**Definition 11.** *For a valid pcoloring $\mathcal{P} \in \mathbf{P}(n)$ and $n \in T$ (possibly with children $n'$,$n''$), we define $\pi_n(\mathcal{P})$ recursively, depending on the node type of $n$.*

- *(L)-node: $\pi_n(\mathcal{P}) = |\Gamma(\mathcal{P}_c)|$;*
- *(vR)-node: $\pi_n(\mathcal{P}) = \min\{\pi_{n'}(\mathcal{P}') \colon \mathcal{P}' \prec_n \mathcal{P}\}$;*
- *(vI)-node: $\pi_n(\mathcal{P}) = \min\{\pi_{n'}(\mathcal{P}') \colon \mathcal{P}' \prec_n \mathcal{P}\} + |\{u \colon \{u,v\} \in \Gamma(\mathcal{P}_c)|$;*
- *(B)-node: $\pi_n(\mathcal{P}) = \min\{\pi_{n'}(\mathcal{P}') + \pi_{n''}(\mathcal{P}'') \colon (\mathcal{P}', \mathcal{P}'') \prec_n \mathcal{P}\} - |\Gamma(\mathcal{P}_c)|$.*

**Lemma 6.** *For any node $n \in T$ and any valid pcoloring $\mathcal{P}$ of $n$, $\sigma_n(\mathcal{P}) = \pi_n(\mathcal{P})$.*

We are now ready to construct an algorithm to count the solutions of an EMC instance. One further notation is required: We restrict the relation $\prec_n$ on pcolorings to a relation $\prec_n^{\min}$, s.t. $\mathcal{P}' \prec_n^{\min} \mathcal{P}$ (resp. $(\mathcal{P}', \mathcal{P}'') \prec_n^{\min} \mathcal{P}$) only holds if $\pi_n(\mathcal{P})$ is determined by $\pi_{n'}(\mathcal{P}')$ (resp. by $\pi_{n'}(\mathcal{P}')$ and $\pi_{n''}(\mathcal{P}'')$) in Definition 11. I.e., $\mathcal{P}' \prec_n^{\min} \mathcal{P}$ holds iff $\mathcal{P}' \prec_n \mathcal{P}$ and either $\pi_n(\mathcal{P}) = \min\{\pi_{n'}(\mathcal{P}') \colon \mathcal{P}' \prec_n \mathcal{P}\}$ (for (R)-nodes) or $\pi_n(\mathcal{P}) = \min\{\pi_{n'}(\mathcal{P}') \colon \mathcal{P}' \prec_n \mathcal{P}\} + |\{u \colon \{u,v\} \in \Gamma(\mathcal{P}_c)|$ (for (I)-nodes) holds. Likewise, $(\mathcal{P}', \mathcal{P}'') \prec_n^{\min} \mathcal{P}$ holds iff $(\mathcal{P}', \mathcal{P}'') \prec_n \mathcal{P}$ and $\pi_n(\mathcal{P}) = \min\{\pi_{n'}(\mathcal{P}') + \pi_{n''}(\mathcal{P}'') \colon (\mathcal{P}', \mathcal{P}'') \prec_n \mathcal{P}\} - |\Gamma(\mathcal{P}_c)|$.

**Definition 12.** *For $\mathcal{P} \in \mathbf{P}(n)$ and $n \in T$ (possibly with children $n'$,$n''$), let*

$$\#_n(\mathcal{P}) = \begin{cases} 1 & n \text{ is (L)-node} \\ \sum_{\mathcal{P}' \prec_n^{\min} \mathcal{P}} \#_{n'}(\mathcal{P}') & n \text{ is (I)- or (R)-node} \\ \sum_{(\mathcal{P}',\mathcal{P}'') \prec_n^{\min} \mathcal{P}} \#_{n'}(\mathcal{P}') \cdot \#_{n''}(\mathcal{P}'') & n \text{ is (B)-node} \end{cases}$$

**Lemma 7.** *For each $n \in T$ and each valid pcoloring $\mathcal{P}$ of $n$, $|\mathcal{E}_n(\mathcal{P})| = \#_n(\mathcal{P})$.*

By combining Lemmas 3, 4, and 7, we immediately get the following theorem.

**Theorem 2.** *Let $\mathbf{P}_{min}$ be the set of all valid pcolorings $\mathcal{P}$ of $n_{root}$, for which there exists no other pcoloring $\mathcal{P}'$ of $n_{root}$, such that $\sigma_{n_{root}}(\mathcal{P}') < \sigma_{n_{root}}(\mathcal{P})$. Then, $|\text{MinCuts}(G,H)| = \sum_{\mathcal{P} \in \mathbf{P}_{min}} \#_{n_{root}}(\mathcal{P})$.*

*Example 6.* We revisit the instance of the EMC problem from Example 2. In Figure 2, the result of computing $\sigma_n(\mathcal{P})$ and $\#_n(\mathcal{P})$ for all valid pcolorings $\mathcal{P}$ at all nodes $n$ is shown. Only the rows with an entry in one of the last two columns correspond to a valid pcoloring $\mathcal{P}$. In this case, the sets in the second column (with heading $C_1, C_2, C_3$) define the coloring $\mathcal{P}_c$ and the column labeled $\Delta_1 : \sigma : \#$ contains the set of "disconnected pairs" $\mathcal{P}_d$ together with the size of the corresponding cuts in $\mathcal{E}_n(\mathcal{P})$ and the number $\#_n(\mathcal{P})$ of such cuts. Some colorings $\mathcal{P}_c$ admit two possible sets of "disconnected pairs" in order to form a valid pcoloring. In these cases, the last column contains the second possibility – again together with the corresponding size of the cuts and the counter $\#_n(\mathcal{P})$. The minimum cuts correspond to the valid pcolorings at the root node $n_8$ with minimal value of $\sigma_{n_8}(\mathcal{P})$. In our example, the only such pcoloring is $\mathcal{P} = (\{\{c, e, f\}\}, \emptyset)$. Thereby the cuts corresponding to $\mathcal{P}$ consist of three edges and there are, in total, two minimum cuts (see Example 2).

We conclude this section by outlining an algorithm for enumerating all solutions with fixed-parameter linear delay by extending the computation of a single solution in the previous section. The basic idea is to traverse the tree decomposition in top-down direction several times – namely once for each solution. In the recursive algorithm described in Section 3, we made arbitrary choices at the root node (for selecting one coloring with optimal value $\sigma_{n_{root}}(\mathcal{C})$) and at internal nodes (e.g., at an (R)-node $n$, for selecting some coloring $\mathcal{C}'$ at the child node $n'$ with $\mathcal{C} \leq \mathcal{C}'$). For enumerating all solutions, we have to maintain lists to keep track of all possible choices at each node in the tree decomposition. These lists allow us to iterate through all possible choices. Hence, at the root, we iterate through all *valid pcolorings* $\mathcal{P}$ of $n_{root}$ with minimal value $\sigma_{n_{root}}(\mathcal{P})$. Likewise, for a valid pcoloring $\mathcal{P}$ at some (R)- or (I)-node $n$, we eventually have to process all pcolorings $\mathcal{P}'$ at the child node $n'$ of $n$, s.t. $\mathcal{P}' \prec_n^{\min} \mathcal{P}$. Finally, also for a valid pcoloring $\mathcal{P}$ at some (B)-node $n$, we eventually have to process all pairs $(\mathcal{P}', \mathcal{P}'')$ of pcolorings at the child nodes $n'$ and $n''$ of $n$, s.t. $(\mathcal{P}', \mathcal{P}'') \prec_n^{\min} \mathcal{P}$. Only at (L)-nodes, no iterating of several possible pcolorings is required. In [16], we state the full algorithm, which crucially depends on Lemma 4, guaranteeing that no duplicates are produced. Note that this algorithm could also be used for the optimization problem, but this would result in a loss of efficiency.

**Theorem 3.** *Given an instance* $(G, H)$ *of the EMC problem together with a tree decomposition* $\mathcal{T}$ *of* $(G, H)$ *having width* $w^*$*. Assuming unit cost for arithmetic operations,* $O(2^{2^{((w^*)^2+w^* \log w^*)}} \cdot \|(G, H)\|)$ *bounds the time to compute* $|\text{MinCuts}(G, H)|$ *as well as the delay to enumerate* $\text{MinCuts}(G, H)$*.*

## 5 Conclusion

We have presented novel algorithms for the optimization and counting problem of Edge Multicut. Moreover, we have outlined how the counting algorithm can be extended to an enumeration algorithm. It is now straightforward to adapt these algorithms to the (Restricted or Unrestricted) Vertex Multicut problem. Indeed, for the optimization problem, we can proceed as in Section 3. The main modification required is to allow colorings where some vertices possibly do not get a color assigned. The intended meaning is that colorless vertices are exactly those which are part of the cut. The pcolorings introduced in Section 4 for the counting and enumeration problem have to be adapted analogously. The details of these algorithms are left for future work. Further extending these algorithms to the weighted multicut problems is even simpler: Rather than maintaining the *cardinality* $\pi_n$ as in Definitions 4 and 11, we would now have to keep track of the *total weight* of a (p)coloring.

Note that the upper bounds on the constants of our algorithms (in Theorems 1 and 3) are obtained by very coarse estimates – assuming straightforward methods for storing and manipulating partitions, etc. For future work, we plan to implement our multicut algorithms – using more sophisticated data structures and algorithms – which should also improve the upper bounds.

## References

1. Costa, M.C., Létocart, L., Roupin, F.: Minimal multicut and maximal integer multiflow: A survey. European Journal of Operational Research 162, 55–69 (2005)
2. Călinescu, G., Fernandes, C.G., Reed, B.A.: Multicuts in unweighted graphs and digraphs with bounded degree and bounded tree-width. J. Alg. 48, 333–359 (2003)
3. Dahlhaus, E., Johnson, D.S., Papadimitriou, C.H., Seymour, P.D., Yannakakis, M.: The complexity of multiterminal cuts. SIAM J. Comput. 23, 864–894 (1994)
4. Garg, N., Vazirani, V.V., Yannakakis, M.: Primal-dual approximation algorithms for integral flow and multicut in trees. Algorithmica 18, 3–20 (1997)
5. Bentz, C.: A simple algorithm for multicuts in planar graphs with outer terminals. Discrete Applied Mathematics 157, 1959–1964 (2009)
6. Flum, J., Grohe, M.: Parameterized Complexity Theory. Springer, Heidelberg (2006)
7. Bousquet, N., Daligault, J., Thomassé, S., Yeo, A.: A polynomial kernel for multicut in trees. In: Proc. STACS 2009. LIPIcs, vol. 3, pp. 183–194 (2009)
8. Marx, D., Razgon, I.: Constant ratio fixed-parameter approximation of the edge multicut problem. In: Fiat, A., Sanders, P. (eds.) ESA 2009. LNCS, vol. 5757, pp. 647–658. Springer, Heidelberg (2009)
9. Marx, D.: Parameterized graph separation problems. Theor. Comput. Sci. 351, 394–406 (2006)

10. Xiao, M.: Simple and improved parameterized algorithms for multiterminal cuts. Theory Comput. Syst. (to appear, 2010)
11. Guo, J., Hüffner, F., Kenar, E., Niedermeier, R., Uhlmann, J.: Complexity and exact algorithms for vertex multicut in interval and bounded treewidth graphs. European Journal of Operational Research 186, 542–553 (2008)
12. Bentz, C.: On the complexity of the multicut problem in bounded tree-width graphs and digraphs. Discrete Applied Mathematics 156, 1908–1917 (2008)
13. Gottlob, G., Lee, S.T.: A logical approach to multicut problems. Inf. Process. Lett. 103, 136–141 (2007)
14. Courcelle, B.: Graph rewriting: An algebraic and logic approach. In: Handbook of Theor. Comp. Sci., vol. B, pp. 193–242. Elsevier Science Publishers, Amsterdam (1990)
15. Arnborg, S., Lagergren, J., Seese, D.: Easy problems for tree-decomposable graphs. J. Algorithms 12, 308–340 (1991)
16. Pichler, R., Rümmele, S., Woltran, S.: Multicut algorithms via tree decompositions. Technical Report DBAI-TR-2010-67, Technische Universität Wien (2010)
17. Kloks, T.: Treewidth: Computations and Approximations. Springer, Berlin (1994)
18. Bodlaender, H.L.: A linear-time algorithm for finding tree-decompositions of small treewidth. SIAM J. Comput. 25, 1305–1317 (1996)
19. van den Eijkhof, F., Bodlaender, H.L., Koster, A.M.C.A.: Safe reduction rules for weighted treewidth. Algorithmica 47, 139–158 (2007)
20. Bodlaender, H.L., Koster, A.M.C.A.: Combinatorial optimization on graphs of bounded treewidth. Comput. J. 51, 255–269 (2008)