Fast exact algorithms for some connectivity problems parametrized by clique-width

Benjamin Bergougnoux, Mamadou Moustapha Kanté

FAST EXACT ALGORITHMS FOR SOME CONNECTIVITY PROBLEMS PARAMETRIZED BY CLIQUE-WIDTH

BENJAMIN BERGOUGNOUX AND MAMADOU MOUSTAPHA KANTÉ

Abstract. Given a clique-width expression of a graph G of clique-width k, we provide $2^{O(k)} \cdot n^{O(1)}$ time algorithms for connectivity constraints on locally checkable properties such as Connected Dominating Set, Connected Perfect Dominating Set or Node-Weighted Steiner Tree. We also propose an $2^{O(k)} \cdot n^{O(1)}$ time algorithm for Feedback Vertex Set. The best running times for all the considered cases were either $2^{O(k \cdot \log(k))} \cdot n^{O(1)}$.

1. Introduction

Tree-width [17] is probably the most well-studied graph parameter in the graph algorithm community, and particularly by people working in *Fixed Parameter Tractable* (FPT for short) algorithms, due partly to its numerous structural and algorithmic properties [4, 8]. For a while people used to think that for many connectivity constraints problems, *e.g.*, Hamiltonian Cycle, Steiner Tree, the naive $k^{O(k)} \cdot n^{O(1)}$ time algorithm, parametrized by tree-width, cannot be improved because it seems necessary to know the connected components of the partial solutions in order to be able to extend them, and certify that the given solution is really connected. But, quite surprisingly Bodlaender et al. showed in [1] that for some of these connectivity constraints problems, one can indeed use the naive algorithm, and instead of keeping all the possible partitions, keep a set of representatives that is single exponential in the tree-width.

Nevertheless, despite the broad interest on tree-width, only sparse graphs can have bounded tree-width. But, on many dense graph classes, some NP-hard problems admit polynomial time algorithms, and many of these algorithms can be explained by the boundedness of their *clique-width*, a graph parameter introduced by Courcelle and al. (see the book [4]) and that emerges from the theory of graph grammars.

Clique-width is defined in terms of the following graph operations: (1) addition of a single labeled vertex, (2) addition of edges between vertices labeled i and those labeled j, (3) renaming of labels, (4) disjoint union; and the clique-width of a graph is the minimum number of labels needed to construct it, and the expression constructing it is called *clique-width expression*. Clique-width generalises tree-width in the sense that if a graph class has bounded tree-width, then it has bounded clique-width [6], but the converse is false as cliques have clique-witdh at most 2 and an unbounded tree-width. Furthermore, clique-width appears also to be of big importance in FPT algorithms [4]. While it is still open whether there exists an FPT algorithm to compute an optimal clique-width expression of a given graph, one can ask when clique-width behaves similarly as tree-width on important problems.

1991 *Mathematics Subject Classification.* F.2.2, G.2.1, G.2.2.

Key words and phrases. clique-width, single exponential algorithm, feedback vertex set, connected σ, ρ-domination.

This work is supported by French Agency for Research under the GraphEN project (ANR-15-CE-0009).

Even though clique-width is far from behaving similarly as tree-width on some well-studied and well-known difficult problems such as HAMILTONICITY [9], Bui-Xuan et al. [2, 3], and Ganian et al. [10, 11] managed to prove, after more substantial work than for tree-width, that for locally checkable properties, and some sparse problems, one can get single exponential time algorithms parametrized by clique-width, some are proved to have a linear dependence on the clique-width, while for others only a polynomial dependence is known. However, on connectivity constraints problems nothing is known, except for some special cases such as FEEDBACK VERTEX SET[1] for which a $2^{O(k \log(k))} \cdot n^{O(1)}$ time algorithm was given in [2].

Our Contributions. We investigate connectivity constraints on locally checkable properties, such as CONNECTED DOMINATING SET,CONNECTED VERTEX COVER or STEINER TREE. All of these problems are some special case of the connected version of the so-called (σ, ρ)- DOMINATING SET problem. This problem was introduced in [18] and studied in graphs of bounded clique-width in [3, 15]. (Definitions are given at the end of Section 2.) We propose $2^{O(k)} \cdot n$ time algorithms, with k the clique-width of the given graph, for the FEEDBACK VERTEX SET problem, and the connected version of the (σ, ρ)- DOMINATING SET problem. While our approach is the same as the rank-based one used in [1] and follows usual dynamic programming, dealing with clique-width operations is known to be harder than manipulating tree-decompositions. One of the main reasons is that, on a particular node of a tree-decomposition, the number of vertices, from the already processed vertices, which have a neighbor in the rest of the graph is bounded, but for clique-width, they can be only classified into a bounded number of equivalence classes (with respect to having the same neighborhood in the rest of the graph). These equivalence classes are the *labels* of a *clique-with expression*.

For both problems, we develop new "naive" dynamic programming algorithms, each keeping tracking of the possible partitions (into connected components of the partial solutions), resulting in $2^{k \log(k)} \cdot n$ time algorithms. We then show how to reduce the time complexity into $2^{O(k)} \cdot n$ by adapting the rank-based approach from [1] to our purposes. Our approach can be summarised as follows.

(1) For each problem, we present a dynamic programming whose steps are the different operations of a given *clique-width expression* of width k. We encode the partial solutions as partitions of labels. At each step of the dynamic programming, we keep track of which labels are intersected by the partial solutions and among them, which are expected to have a future neighbor in the next step of the dynamic programing, *i.e.*, to be part of an operation adding edges with another intersected label. This last information is the key of our dynamic programing, as the vertices in a label expecting a future neighbor behave like a single vertex, in terms of connectivity. Moreover, they are the only vertices that affect the connectivity of the partial solution in the future steps of the dynamic programming, since the other vertices are expected to have no future neighbor.
(2) For (σ, ρ)- DOMINATING SET, we need to store some additional information to ensure the domination. For doing so, we use the notions of *d-neighbor equivalence* introduced in [3].
(3) For FEEDBACK VERTEX SET, to ensure the acyclicity of the partial solutions, we differentiate the labels intersected only once and those that are intersected at least twice. We, moreover, define a notion of *acyclicity* between two partitions that is crucial for the fusion of two partial solutions.

[1] A FEEDBACK VERTEX SET in a graph is a subset of its vertex set which deletion induces a forest.

(4) For both problems, we more or less keep as indexes of the dynamic programming tables the set of labels that are intersected. Each table has at most $2^{O(k)}$ indexes, but the number of possible partitions is $2^{O(k \log(k))}$, giving a running time of $2^{O(k \log(k))} \cdot n$. We reduce this to $2^{O(k)} \cdot n$ by adapting the toolkit introduced in [1] to our needs. This toolkit allows us to improve the running time of our dynamic programming to $2^{O(k)} \cdot n^{O(1)}$, by reducing the number of partial solutions we need to keep at each step. Our main contribution here was to incorporate the notion of acyclicity in the rank-based approach. This contribution was necessary to solve Feedback Vertex Set.

2. Preliminaries

The size of a set V is denoted by $|V|$ and its power set is denoted by 2^V. We write $A \setminus B$ for the set difference of A from B, and we write $A \uplus B$ for the disjoint union of A and B. We often write x to denote the singleton set $\{x\}$. For a mapping $f : A \to B$, and $X \subseteq B$, we let $f^{-1}(X) := \{a \in A \mid f(a) \in X \neq \emptyset\}$. We let $\min(\emptyset) := +\infty$ and $\max(\emptyset) := -\infty$. We let $[k] := \{1, \ldots, k\}$. We denote by $\mathbb{N}$ the set of non-negative integers and by $\mathbb{F}_2$ the binary field.

Partitions. A partition p of a set V is a collection of non-empty subsets of V that are pairwise non-intersecting and such that $\cup_{p_i \in p} p_i = V$; each set p_i in p is called a *block* of p. The set of partitions of a finite set V is denoted by $\Pi(V)$, and $(\Pi(V), \sqsubseteq)$ forms a lattice where $p \sqsubseteq q$ if for each block p_i of p there is a block q_j of q with $p_i \subseteq q_j$. The join operation of this lattice is denoted by $\sqcup$. Let $\sharp\text{block}(p)$ denote the number of blocks of a partition p. Observe that $\emptyset$ is the only partition of the empty set. We denote by $V[X]$, for $X \subseteq V$, the partition of V where one block is X and the other blocks are all singletons. For $p \in \Pi(V)$ and $X \subseteq V$, let $p_{\downarrow X} \in \Pi(X)$ be the partition $\{p_i \cap X \mid p_i \in p\} \setminus \{\emptyset\}$, and for $Y \supseteq V$, let $p_{\uparrow Y} \in \Pi(Y)$ be the partition $p \cup \bigcup_{y \in Y \setminus V} \{\{y\}\}$.

Graphs. Our graph terminology is standard, and we refer to [7]. The vertex set of a graph G is denoted by $V(G)$ and its edge set by $E(G)$. An edge between two vertices x and y is denoted by xy (respectively yx). The subgraph of G induced by a subset X of its vertex set is denoted by $G[X]$, and we write $G \setminus X$ to denote the induced subgraph $G[V(G) \setminus X]$. The set of vertices that is adjacent to x is denoted by $N_G(x)$, and for $U \subseteq V(G)$, $N_G(U) := \left(\bigcup_{v \in U} N_G(v)\right) \setminus U$. For a graph G, we denote by $CC(G)$ the partition $\{V(C) \mid C \text{ is a component of } G\}$ of $V(G)$.

Clique-Width. A *k-labeled graph* is a pair (G, lab_G) with G a graph and lab_G a function from V_G to $[k]$, called the *labeling function*; each set $lab_G^{-1}(i)$ is called a *label class*. The notion of clique-width is defined by Courcelle et al. [5] and is based on the following operations.

(1) Create a graph, denoted by $\mathbf{1}(x)$, with a single vertex x labeled with 1.
(2) For a labeled graph G and distinct labels $i, j \in [k]$, relabel the vertices of G with label i to j (denoted by $ren_{i \to j}(G)$). Notice that there is no more vertices labeled i in $ren_{i \to j}(G)$.
(3) for a labeled graph G and distinct labels $i, j \in [k]$, add all the non-existent edges between vertices with label i and vertices with label j (denoted by $add_{i,j}(G)$).
(4) Take the disjoint union of two labeled graphs G and H, denoted by $G \oplus H$. The labeling function of $G \oplus H$ is $lab_G \uplus lab_H$.

A *k-expression* is a finite well-formed term built with the four operations above. Each k-expression t evaluates into a k-labeled graph $(val(t), lab_t)$. The *clique-width* of a graph G, denoted by $\text{cwd}(G)$, is the minimum k such that G is isomorphic to

$val(t)$ for some k-expression t. We can assume without loss of generality that any k-expression defining a graph G uses $O(n)$ disjoint union operations and $O(nk^2)$ unary operations [6].

It is worth noticing from the recursive definition of k-expressions, one can compute in linear time in $|t|$ the labeling function lab_t of $val(t)$, and hence we will always assume that it is given.

Considered Connectivity Problems. For all the problems in this article, we consider the weight function to be on the vertices. Considering the weight to be on the edges would make all the considered problems NP-hard even on graphs of clique-width 2. A subset $X \subseteq V(G)$ of the vertex set of a graph G is a *feedback vertex set* if $G \setminus X$ is a forest. The problem Feedback Vertex Set (FVS for short) consists in finding a minimum feedback vertex set[2]. It is not hard to verify that X is a minimum feedback vertex set of G if and only if $G \setminus X$ is a maximum forest.

The problem Steiner Tree asks, given a subset of vertices $K \subseteq V(G)$ called *terminals*, a subset T of minimum weight such that $K \subseteq T \subseteq V(G)$ and $G[T]$ is connected.

Let σ and ρ be (non-empty) finite or co-finite subsets of $\mathbb{N}$. We say that a subset D of $V(G)$ (σ, ρ)-*dominates* a subset U if for every vertex $u \in U$, $|N_G(u) \cap D| \in \sigma$ if $u \in D$, otherwise $|N_G(u) \cap D| \in \rho$. A (σ, ρ)-dominating set D of $V(G)$ is called a *connected* (resp. *out-connected*) (σ, ρ)-*dominating set of* G if D (resp. $V \setminus D$) is connected. The problem (Out-)Connected (σ, ρ) Dominating Set ask for a minimum or a maximum (out-)connected dominating set. Examples of some vertex subset properties expressible as (σ, ρ)-dominating set are shown on Table 1.

σ	ρ	Version	Standard name
$\mathbb{N}$	$\mathbb{N}^+$	*Connected*	Connected Dominating Set
$\mathbb{N}^+$	$\mathbb{N}^+$	*Connected*	Connected Total Dominating Set
$\{d\}$	$\mathbb{N}$	*Connected*	Connected Induced d-Regular Subgraph
$\mathbb{N}$	$\{1\}$	*Connected*	Connected Perfect Dominating Set
$\{0\}$	$\mathbb{N}$	*Out-connected*	Connected Vertex Cover

Figure 1. Examples of (out)-connected (σ, ρ)-dominating sets, $\mathbb{N} = \{0, 1, \dots, \}$ and $\mathbb{N}^+ = \{1, 2, \dots\}$.

3. Representing Sets of Weighted Partitions by Matrices

As with usual dynamic programming algorithms dealing with connectivity constraints, the partial solutions of our algorithms are *weighted partitions*. We modify in this section the operators on weighted partitions defined in [1] in order to express our dynamic programming algorithm for FVS in terms of these operations.

Let first define formally our notion of partial solution. Let V and S be two disjoint finite sets. We denote by $\Pi(V, S)$ the set $\bigcup_{X \subseteq S} \{X\} \times \Pi(V)$. A set of *weighted partitions* on (V, S) is a subset of $\Pi(V, S) \times \mathbb{N}$. For $\mathcal{A} \subseteq \Pi(V, S) \times \mathbb{N}$ and $X \subseteq V \cup S$, we let $\mathcal{A}_{\restriction X}$ be $\{(p_0, p, w) \in \mathcal{A} \mid p_0 = X\}$.

We will always denote the elements of $\Pi(V, S)$ as the ordered pairs (p_0, p) with $p_0 \subseteq S$ and $p \in \Pi(V)$ assuming that the blocks of p are always denoted by $p_1, p_2, \dots$. Indeed, elements of $\Pi(V, S)$ can be seen as partitions of $\Pi(V \cup S)$ where we have identified two blocks, *i.e.*, X and $S \setminus X$. Intuitively, V represents the intersected

[2]By a minimum (or maximum) set, we always mean a set of minimum (or maximum) weight.

labels that are expected to have a future and S represents the labels which do no matter in the connectivity anymore. Each weighted partition $(p_0, p, w) \in \Pi(V, S) \times \mathbb{N}$ is intended to mean: there is a solution F of weight w that does not intersect the sets $lab_G^{-1}(p_0)$, the intersected elements in $lab_G^{-1}(S \setminus p_0)$ are expected to have no future neighbor and p is the transitive closure of the following relation on V : $i \sim j$ if there exists $x \in lab_G^{-1}(i)$ and $y \in lab_G^{-1}(j)$ connected in $G[F]$.

Definition 1. We let $\mathsf{acyclic}$ be the relation on $\Pi(V, S) \times \Pi(V, S)$ where $\mathsf{acyclic}(p, q)$ holds exactly when $|V| + \sharp\mathrm{block}(p \sqcup q) - (\sharp\mathrm{block}(p) + \sharp\mathrm{block}(q)) = 0$. In other words, if $F_p := (V\ , E_p)$ and $F_q := (V, E_q)$ are forests with components $p = CC(F_p)$ and $q = CC(F_q)$ respectively, then $\mathsf{acyclic}(p, q)$ holds if and only if $E_p \cap E_q \neq \emptyset$ and $(V, E_p \uplus E_q)$ is a forest.

Let us now redefine some of the operators introduced in [1]. Let V and S be two disjoint finite sets.

Remove non-maximal copies. For $\mathcal{A} \subseteq \Pi(V, S) \times \mathbb{N}$, we define $\mathsf{rmc}(\mathcal{A}) \subseteq \mathcal{A}$ as

$$\mathsf{rmc}(\mathcal{A}) := \{(p_0, p, w) \in \mathcal{A} \mid \forall (p_0, p, w') \in \mathcal{A}, w' \leq w)\}.$$

Ac-Join. Let V' and S' be two disjoint finite sets. For $\mathcal{A} \subseteq \Pi(V, S) \times \mathbb{N}$ and $\mathcal{B} \subseteq \Pi(V', S') \times \mathbb{N}$, we define $\mathsf{acjoin}(\mathcal{A}, \mathcal{B}) \subseteq \Pi(V \cup V', (S \setminus V') \cup (S' \setminus V)) \times \mathbb{N}$ as

$$\mathsf{acjoin}(\mathcal{A}, \mathcal{B}) := \mathsf{rmc}\left(\{(p_0 \setminus (V \cup S')) \cup (q_0 \setminus (V' \cup S)) \cup (p_0 \cap q_0), p_{\uparrow V'} \sqcup q_{\uparrow V}, w_1 + w_2) \mid (p_0, p, w_1) \in \mathcal{A}, (q_0, q, w_2) \in \mathcal{B} \text{ and } \mathsf{acyclic}(p_{\uparrow V'}, q_{\uparrow V})\}\right).$$

This operator is more or less the same as the one in [1], except that we incorporate the acyclicity condition. It is used to construct partial solutions while guaranteeing the acyclicity. Observe that if p_0 and q_0 are the labels not intersected by the partial solution represented respectively by (p_0, p, w_1) and (q_0, q, w_2), then $(p_0 \setminus (V \cup S')) \cup (q_0 \setminus (V' \cup S)) \cup (p_0 \cap q_0)$ represents the labels not intersected by the union of these two partial solutions.

Project. For $X \subseteq (V \cup S)$ and $\mathcal{A} \subseteq \Pi(V, S) \times \mathbb{N}$, let $\mathsf{proj}(\mathcal{A}, X) \subseteq \Pi(V \setminus X, S \setminus X) \times \mathbb{N}$ be

$$\mathsf{proj}(\mathcal{A}, X) := \mathsf{rmc}\left(\{(p_0 \setminus X, p_{\downarrow(V \setminus X)}, w) \mid (p_0, p, w) \in \mathcal{A} \text{ and } \forall p_i \in p,\ (p_i \setminus X) \neq \emptyset\}\right).$$

In other words, take the partitions for which each block is not completely contained in X, and then remove X from those partitions. This operator is used to remove from the partitions the label classes unused after a renaming or that are required to not have future neighbors, *i.e.*, they will not matter in the connectivity of the partial solutions. If a partition has a block fully contained in X, it means that this block will remain disconnected in the future steps of our dynamic programming algorithm, and that is why we remove such partitions.

One needs to perform the above operations efficiently, and this is guaranteed by the following, which assumes that $\log(|\mathcal{A}|) \leq |V \cup S|^{O(1)}$ for each $\mathcal{A} \subseteq \Pi(V, S) \times \mathbb{N}$.

Proposition 3.1 (Folklore)**.** *The operator* acjoin *can be performed in time* $|\mathcal{A}| \cdot |\mathcal{B}| \cdot |V \cup S \cup V' \cup S'|^{O(1)}$ *and the sizes of their outputs are upper-bounded by* $|\mathcal{A}| \cdot |\mathcal{B}|$*. The operators* rmc *and* proj *can be performed in time* $|\mathcal{A}| \cdot |V \cup S|^{O(1)}$*, and the size of their outputs are upper-bounded by* $|\mathcal{A}|$*.*

We now define the notion of representative sets of weighted partitions which is the same as the one in [1], except that we need to incorporate the acyclicity condition as for the acjoin operator above.

Definition 2. Let V and S be two disjoint finite sets and let $\mathcal{A} \subseteq \Pi(V,S) \times \mathbb{N}$. For $(q_0, q) \in \Pi(V,S)$, let

$$\textbf{ac-opt}(\mathcal{A}, (q_0, q)) := \max\{w \mid (q_0, p, w) \in \mathcal{A},\ p \sqcup q = \{V\} \text{ and } \mathsf{acyclic}(p, q)\}.$$

A set of weighted partitions $\mathcal{A}' \subseteq \Pi(V,S) \times \mathbb{N}$ *ac-represents* $\mathcal{A}$ if for each $(q_0, q) \in \Pi(V,S)$, it holds that $\textbf{ac-opt}(\mathcal{A}, (q_0, q)) = \textbf{ac-opt}(\mathcal{A}', (q_0, q))$.

Let Z be a finite set and V', S' be two disjoint finite sets. A function $f : 2^{\Pi(V,S)\times\mathbb{N}} \times Z \to 2^{\Pi(V',S')\times\mathbb{N}}$ is said to *preserve ac-representation* if for each $\mathcal{A}, \mathcal{A}' \subseteq \Pi(V,S) \times \mathbb{N}$ and $z \in Z$, it holds that $f(\mathcal{A}', z)$ ac-represents $f(\mathcal{A}, z)$ whenever $\mathcal{A}'$ ac-represents $\mathcal{A}$.

The following lemma is not difficult to prove. A proof is given in the appendix.

Lemma 3.2. *The operators* rmc, proj, acjoin *and* $\cup$ *preserve ac-representation.*

In the remaining we will prove that we can find for every set $\mathcal{A} \subseteq \Pi(V,S) \times \mathbb{N}$ a set $\mathcal{A}' \subseteq \mathcal{A}$ of size at most $(|V|+1) \cdot 2^{|V \cup S|}$ that ac-represents $\mathcal{A}$, in polynomial time in $\mathcal{A}$. We will encode the ac-representativity by a matrix over $\mathbb{F}_2$ and show that this one has rank at most the desired bound. Next, we show that a basis can be found using matrix multiplications, using the following lemma from [1]. The constant ω denotes the matrix multiplication exponent.

Lemma 3.3 ([1])**.** *Let M be an $n \times m$-matrix over $\mathbb{F}_2$ with $m \leq n$ and let $w : \{1, \ldots, n\} \to \mathbb{N}$ be a weight function. Then, one can find a basis of maximum (or minimum) weight of the row space of M in time $O(nm^{\omega-1})$.*

Let $v_0 \in V$ and let $\mathsf{tricuts}(V,S) := \{(U, V_1, V_2) \mid U \subseteq S \wedge V_1 \uplus V_2 = V \wedge v_0 \in V_1\}$. Let α be a variable, that will be used to encode the edges induced by partitions. Recall that for integers i and j, $\alpha^i \cdot \alpha^j = \alpha^{i+j}$ in any field.

Let M and C be, respectively, a $(\Pi(V,S), \Pi(V,S))$-matrix and a $(\Pi(V,S), \mathsf{tricuts}(V,S))$-matrix, both over $\mathbb{F}_2[\alpha]$, such that

$$M[(p_0,p),(q_0,q)] := \begin{cases} 0 & \text{if } p_0 \neq q_0 \text{ or } p \sqcup q \neq \{V\}, \\ \alpha^{2|V|-(\sharp\mathrm{block}(p)+\sharp\mathrm{block}(q))} & \text{otherwise.} \end{cases}$$

$$C[(p_0,p),(U,V_1,V_2)] := \begin{cases} 0 & \text{if } p_0 \neq U \text{ or } p \not\sqsubseteq (V_1,V_2), \\ \alpha^{|V|-\sharp\mathrm{block}(p)} & \text{otherwise.} \end{cases}$$

The following guarantees an efficient algorithm to reduce the number of partial solutions we need to keep at each step of our dynamic programing.

Theorem 3.4. *There exists an algorithm* $\mathsf{ac\text{-}reduce}$ *that given a set of weighted partitions $\mathcal{A} \subseteq \Pi(V,S) \times \mathbb{N}$, outputs in time $|\mathcal{A}| \cdot 2^{(\omega-1)|V\cup S|-1} \cdot |V|^{O(1)}$ a subset $\mathcal{A}'$ of $\mathcal{A}$ that ac-represents $\mathcal{A}$ and such that $|\mathcal{A}'| \leq (|V|+1) \cdot 2^{|V\cup S|-1}$.*

Proof. We first prove that $M = C \cdot C^t$. Notice that for each $(p_0,p) \in \Pi(V,S)$ and $(U,V_1,V_2) \in \mathsf{tricuts}(V,S)$, $C[(p_0,p),(U,V_1,V_2)] \in \{0,1,\alpha^1,\ldots,\alpha^{|V|}\}$. Let $(p_0,p),(q_0,q) \in \Pi(V,S)$. If $p_0 \neq q_0$, then for each (U,V_1,V_2) either $C[(p_0,p),(U,V_1,V_2)] = 0$ or $C[(q_0,q),(U,V_1,V_2)] = 0$, hence $(C \cdot C^t)[(p_0,p),(q_0,q)] = M[(p_0,p),(q_0,q)] = 0$.

So, suppose that $p_0 = q_0 = U$. Then,

$$\begin{aligned}(C \cdot C^t)[(U,p),(U,q)] &= \sum_{\substack{(U,V_1,V_2)\in \mathsf{tricuts}(V,S)\\ p\sqsubseteq(V_1,V_2)\\ q\sqsubseteq(V_1,V_2)}} \alpha^{|V|-\sharp\mathrm{block}(p)} \cdot \alpha^{|V|-\sharp\mathrm{block}(q)}\\ &= \sum_{\substack{(U,V_1,V_2)\in \mathsf{tricuts}(V,S)\\ p\sqsubseteq(V_1,V_2)\\ q\sqsubseteq(V_1,V_2)}} \alpha^{2|V|-(\sharp\mathrm{block}(p)+\sharp\mathrm{block}(q))}\\ &= \sum_{\substack{(U,V_1,V_2)\in \mathsf{tricuts}(V,S)\\ p\sqcup q\sqsubseteq(V_1,V_2)}} \alpha^{2|V|-(\sharp\mathrm{block}(p)+\sharp\mathrm{block}(q))}.\end{aligned}$$

We are counting the triplets $(U, V_1, V_2) \in \mathsf{tricuts}(V, S)$ such that (V_1, V_2) is coarser than both p and q, hence we are counting the triplets (U, V_1, V_2) such that (V_1, V_2) is coarser than $p \sqcup q$. Since, this number is equal to the number of bipartitions (V_1, V_2) of the blocks of $p \sqcup q$ with $v_0 \in V_1$, we can conclude from the last equality that

$$(C \cdot C^t))[(U,p),(U,q)] = \underbrace{\alpha^{2|V|-(\sharp\mathrm{block}(p)+\sharp\mathrm{block}(q))} + \cdots + \alpha^{2|V|-(\sharp\mathrm{block}(p)+\sharp\mathrm{block}(q))}}_{2^{\sharp\mathrm{block}(p\sqcup q)-1}}$$

and because the characteristic of the field is 2 and $2^{\sharp\mathrm{block}(p\sqcup q)-1}$ is odd iff $\sharp\mathrm{block}(p\sqcup q) = 1$

$$(C \cdot C^t))[(U,p),(U,q)] = \begin{cases}\alpha^{2|V|-(\sharp\mathrm{block}(p)+\sharp\mathrm{block}(q))} & \text{if } p \sqcup q = \{V\}\\ 0 & \text{otherwise.}\end{cases}$$

We can therefore conclude that $M = C \cdot C^t$.

Let $\mathcal{A}$ be a set of weighted partitions. We can suppose *w.l.o.g.* that $\mathcal{A} = \mathsf{rmc}(\mathcal{A})$. For each $0 \leq i \leq |V|$, let $\mathcal{A}_i$ be the set $\{(p_0, p, w) \in \mathcal{A} \mid |V| - \sharp\mathrm{block}(p) = i\}$, and let $C^i_{\mathcal{A}}$ be the restriction of C to the rows in $\{(p_0, p) \mid (p_0, p, w) \in \mathcal{A}_i\}$. Observe that $\{\mathcal{A}_0, \ldots, \mathcal{A}_{|V|}\}$ is a partition of $\mathcal{A}$ and each entry of $C^i_{\mathcal{A}}$ is either 0 or α^i, and hence one can see it as a matrix over $\mathbb{F}_2$ by transforming each α^i by 1. The rank of $C^i_{\mathcal{A}}$ is bounded by $2^{|V\cup S|-1}$ as the number of triplets in $\mathsf{tricuts}(V, S)$ is less than $2^{|V\cup S|-1}$, and by Lemma 3.3 one can find a basis $\mathcal{B}_i$ of $C^i_{\mathcal{A}}$ of maximum weight, for all i, in time bounded by $|\mathcal{A}| \cdot 2^{(\omega-1)\cdot(|V\cup S|-1)} \cdot |V|^{O(1)}$, where the weights are the weights of the considered weighted partitions in $\mathcal{A}$. Let $\mathcal{A}'_i = \{\, (p_0, p, w) \in \mathcal{A}_i \mid (p_0, p) \in \mathcal{B}_i\}$, and let $\mathcal{A}' := \mathcal{A}'_0 \uplus \mathcal{A}'_1 \uplus \cdots \uplus \mathcal{A}'_{|V|}$. Observe that $|\mathcal{A}'| \leq (|V| + 1) \cdot 2^{|V\cup S|-1}$. Moreover, for each $(p_0, p, w) \in \mathcal{A}_i$, there is $B(p_0, p) \subseteq B_i$ such that, for each $(U, V_1, V_2) \in \mathsf{tricuts}(V, S)$,

$$C[(p_0,p),(U,V_1,V_2)] = \sum_{(q_0,q)\in B(p_0,p)} C[(q_0,q),(U,V_1,V_2)].$$

We claim that $\mathcal{A}'$ ac-represents $\mathcal{A}$. Since $\mathcal{A}' \subseteq \mathcal{A}$, it is sufficient to prove that for all $(p_0, p, w) \in \mathcal{A}$ and $(r_0, r) \in \Pi(V, S)$ such that $p_0 = r_0$, $p \sqcup r = \{V\}$ and $\mathsf{acyclic}(p, r)$, there exists $(q_0, q, w') \in \mathcal{A}'$ with $w \leq w'$ and such that $q_0 = r_0$, $q \sqcup r = \{V\}$ and $\mathsf{acyclic}(q, r)$.

By definition of $\mathsf{acyclic}$ and of M, we have $p_0 = r_0$, $p \sqcup r = \{V\}$ and $\mathsf{acyclic}(p, r)$ if and only if $M[(p_0, p), (r_0, r)] = \alpha^{|V|-1}$. From the equality $M = C \cdot C^t$, we have

$$\begin{aligned}
M[(p_0,p),(r_0,r)] &= \sum_{(U,V_1,V_2)\in \mathsf{tricuts}(V,S)} C[(p_0,p),(U,V_1,V_2)] \cdot C^t[(U,V_1,V_2),(r_0,r)] \\
&= \sum_{(U,V_1,V_2)\in \mathsf{tricuts}(V,S)} \left(\sum_{(q_0,q)\in B(p_0,p)} C[(q_0,q),(U,V_1,V_2)] \right) \cdot C^t[(U,V_1,V_2),(r_0,r)] \\
&= \sum_{(q_0,q)\in B(p_0,p)} \left(\sum_{(U,V_1,V_2)\in \mathsf{tricuts}(V,S)} C[(q_0,q),(U,V_1,V_2)] \cdot C^t[(U,V_1,V_2),(r_0,r)] \right) \\
&= \sum_{(q_0,q)\in B(p_0,p)} M[(q_0,q),(r_0,r)].
\end{aligned}$$

Since the characteristic of the field is 2, $M[(p_0, p), (r_0, r)] = \alpha^{|V|-1}$ if and only if there is an odd number of $(q_0, q) \in B(p_0, p)$ such that $M[(q_0, q), (r_0, r)] = \alpha^{|V|-1}$. Let i such that $(p_0, p, w) \in \mathcal{A}_i$. Thus, there exists $(q_0, q) \in B(p_0, p)$ such that $M[(q_0, q), (r_0, r)] = \alpha^{|V|-1}$, *i.e.*, there exists $(q_0, q, w') \in \mathcal{A}'_i$ such that $q_0 = r_0$, $q \sqcup r = \{V\}$ and $\mathsf{acyclic}(q, r)$. Let $(q_0, q, w') \in \mathcal{A}'_i$ such that w' is maximum and $q_0 = r_0$, $q \sqcup r = \{V\}$ and $\mathsf{acyclic}(q, r)$. Assume towards a contradiction that $w' < w$. Thus $(p_0, p) \notin \mathcal{B}_i$. But, $(\mathcal{B}_i \setminus \{(q_0, q)\}) \cup \{(p_0, p)\}$ is also a basis of $C^i_{\mathcal{A}}$ since the set of independent row sets of a matrix forms a matroid. Since $w > w'$, the weight of $(\mathcal{B}_i \setminus \{(q_0, p')\}) \cup \{(q_0, p)\}$ is strictly greater than the weight of $\mathcal{B}_i$, yielding a contradiction. Thus $\mathcal{A}'$ ac-represents $\mathcal{A}$. □

4. Feedback Vertex Set

Instead of computing a minimum feedback vertex set, we will compute a maximum forest. Contrary to the standard technique in which the label classes that are intersected by the solutions is guessed, and for each such guess we store weighted partitions which tell us how the partial solution is connected, we will encode the non intersected label classes in the weighted partitions. As in [1] we express the steps of our algorithm by using the operators on weighted partitions defined in Section 3. But, because it is much easier with the framework to deal with connected solutions, we will compute a maximum tree instead of a maximum forest. For doing so, we introduce an hypothetical new vertex, denoted by v_0, that is universal and we compute a pair (F, E_0) so that F is a maximum forest of G, $E_0 \subseteq \{v_0\} \times V(G)$ and $(V(F) \cup \{v_0\}, E(F) \cup E_0)$ is a tree. For a weight function w on the vertices of G and a subset $S \subseteq V(G)$, we denote by $\mathsf{w}(S) := \sum_{v \in S} \mathsf{w}(v)$.

Definition 3. Let G be a k-labeled graph and let $s : [k] \to \{1, 2, -2\}$ be a total function. If $s^{-1}(2) = \{i_1, \ldots, i_p\}$, we let V_s^+ be the set $\{x_{i_1}^+, \ldots, x_{i_p}^+\}$ disjoint from $V(G) \cup \{v_0\}$. The entries of $\mathcal{A}_G[s]$ are all weighted partitions $(p_0, p, w) \in \Pi(s^{-1}(\{1, 2\}) \cup \{v_0\}, s^{-1}(-2)) \times \mathbb{N}$ such that there exist an induced subgraph F of G and $E_s^0 \subseteq \{v_0\} \times V(F)$ so that $\mathsf{w}(V(F)) = w$,

1. The set $s^{-1}(1) = \{i \in [k] \mid |V(F) \cap lab_G^{-1}(i)| = 1\}$ and $p_0 = \{i \in [k] \mid |V(F) \cap lab_G^{-1}(i)| = 0\}$.
2. The graph $F^+ := (V(F) \cup \{v_0\} \cup V_s^+, E(F) \cup E_s^0 \cup E_s^+)$ is a forest where $E_s^+ := \bigcup_{1 \le j \le p} x_{i_j} \times (V(F) \cap lab_G^{-1}(i_j))$.
3. Each component C of $(V(F) \cup \{v_0\}, E(F) \cup E_s^0)$ intersects $lab_G^{-1}(s^{-1}(\{1, 2\})) \cup \{v_0\}$.
4. The partition p equals $(s^{-1}(\{1, 2\}) \cup \{v_0\}) / \sim_{F^+}$ where $i \sim j$ if a vertex in $lab_G^{-1}(i) \cap V(F)$ is connected in the graph F^+ to a vertex in $lab_G^{-1}(j) \cap V(F)$; we consider $lab_G^{-1}(v_0) = \{v_0\}$.

Our algorithm will store for each labeled graph G a table $tab_G[s]$ that accepts $\mathcal{A}_G[s]$.

The first condition says that the label classes that are not intersected by the solution are stored in p_0, those that are intersected once are in $s^{-1}(1)$ and those that are intersected at least twice are in $s^{-1}(\{-2,2\})\backslash p_0$. We expect the intersected vertices that belong to a label class in $s^{-1}(2)$ to get exactly one future neighbor in the possible extensions of the partial solutions. On the other hand, those that belong to a label class in $s^{-1}(-2)$ are preventing from having a neighbor in any extension of the corresponding partial solution. Condition (2) guarantees that $(V(F)\cup\{v_0\}, E(F)\cup E_s^0)$ is acyclic and can be extended into a forest. Each vertex $x_{i_\ell}^+ \in V_s^+$ is intended to represent the expected future neighbor of the intersected vertices in $lab^{-1}(\ell)$. It is worth noticing that the acyclicity of F^+ implies that two vertices in $V(F)\cap lab_G^{-1}(i)$ with $s(i)=2$ must be in different components in F, otherwise F^+ would contain a cycle. Condition (3) implies that each component of the solution $(V(F)\cup\{v_0\}, E(F)\cup E_s^0)$ contains a vertex that will play a role in the connectivity of possible extensions. Condition (4) tells us that p is a partition of $s^{-1}(\{1,2\})\cup\{v_0\}$ and that i and j are in the same block of p if there are vertices x and y of $V(F)$ labeled respectively i and j and that are in the same component of F^+. Observe that this describes an equivalence relation since, for each label $i\in s^{-1}(2)$, the vertices labeled i are connected in F^+ through x_i^+.

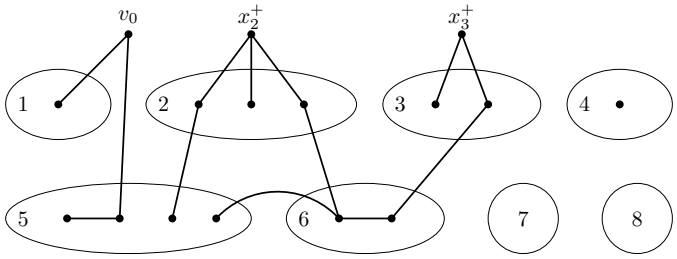

FIGURE 2. Example of graph F^+, here $p=\{\{v_0,1\},\{2,3\},\{4\}\}$, $s^{-1}(1)=\{1,4\}$, $s^{-1}(2)=\{2,3\}$, $s^{-1}(-2)=\{5,6,7,8\}$ and $p_0=\{7,8\}$.

Given a graph G and a k-expression t such that $val(t)=G$, we can find the size of a maximum induced forest of G by checking whether $(\emptyset,\{\{v_0\}\},w)$ is the entry of $\mathcal{A}_{G'}[s_{G'}]$ with $G'=ren_{k\to1}(ren_{(k-1)\to1}(\dots(ren_{2\to1}(t))\dots))$ and $s_{G'}(1):=-2$. By definition, if $(\emptyset,\{\{v_0\}\},w)$ belongs to $\mathcal{A}_{G'}[s_{G'}]$, then there exist a set $F\subseteq V(G)$ and a set of edges $E_0\subseteq\{v_0\}\times V(G)$ such that $(F\cup\{v_0\}, E(G[F])\cup E_0)$ is an induced tree and $\mathsf{w}(F)=w$ is maximum.

Irredundant k-expressions. To simplify the algorithm, we will use *irredundant k-expressions*. A k-expression is *irredundant* if whenever the operation $add_{i,j}$ is applied on G, there is no edge between an i-vertex and an j-vertex in G. It is proved in [6] that any k-expression can be transformed in linear time into an irredundant k-expression.

Computing tab_G for $G=\mathbf{1}(x)$. For $s:\{1\}\to\{1,2,-2\}$, let

$$tab_G[s]:=\begin{cases}\{(\emptyset,\{\{1,v_0\}\},\mathsf{w}(x)),(\emptyset,\{\{1\},\{v_0\}\},\mathsf{w}(x))\} & \text{if } s(1)=1,\\ \{(\{1\},\{\{v_0\}\},0)\} & \text{if } s(1)=-2,\\ \emptyset & \text{if } s(1)=2.\end{cases}$$

Since $|V(G)|=1$, there is no solution intersecting the label class 1 on at least two vertices, and so the set of weighted partitions satisfying Definition 3 equals

the empty set for $s(1) = 2$. For the same reason, if $s(1) = -2$ then x is not in the partial solution. If $s(1) = 1$, there are two possibilities, depending on whether $S_0 = \emptyset$ or $S_0 = \{xv_0\}$. We can thus conclude that $tab_G[s] = \mathcal{A}_G[s]$ is correctly computed.

Computing tab_G for $G = ren_{i\to j}(H)$. We can suppose that H is k-labeled. Let $s : [k] \setminus \{i\} \to \{1, 2, -2\}$, and let $s_1, s_2, s_3 : [k] \to \{1, 2, -2\}$ with $s_1(\ell) := s_2(\ell) := s_3(\ell) := s(\ell)$ for $\ell \notin \{i, j\}$, and

- $s_1(j) = s(j)$, $s_1(i) = -2$,
- If $s(j) = -2$, then $s_2(i) := s_2(j) = -2$, otherwise s_2 is not defined,
- If $s(j) \in \{1, 2\}$, then $s_3(i) = s(j)$ and $s_3(j) = -2$.

We let $tab_G[s] := \mathsf{ac\text{-}reduce}(\mathsf{rmc}(\mathcal{A}))$ where

$$\mathcal{A} := \bigcup_{X\subseteq s_1^{-1}(-2)} \mathsf{proj}(\{i\}, (tab_H[s_1])_{\restriction X\cup\{i\}}) \cup \bigcup_{X\subseteq s_2^{-1}(-2)\setminus\{i\}} \mathsf{proj}(\{j\}, (tab_H[s_2])_{\restriction X\cup\{j\}}) \cup$$
$$\bigcup_{X\subseteq s_3^{-1}(-2)\setminus\{i,j\}} \mathsf{proj}(\{i\}, \mathsf{acjoin}((tab_H[s_3])_{\restriction X\cup\{j\}}, \{([k]\setminus\{i,j\}, \{\{i,j\}\}, 0)\})) \cup$$
$$\bigcup_{\substack{\forall \ell\notin\{i,j\},\ s_H(\ell)=s(\ell)\\ s(j)=-2, s_H(i), s_H(j)\in\{1,-2\}}} \bigcup_{X\subseteq s_H^{-1}(-2)\setminus\{i,j\}} \mathsf{proj}(\{i,j\}, (tab_H[s_H])_{\restriction X}) \cup$$
$$\bigcup_{\substack{\forall \ell\notin\{i,j\},\ s_H(\ell)=s(\ell)\\ s(j)=2, s_H(i), s_H(j)\in\{1,2\}}} \mathsf{proj}(\{i\}, \mathsf{acjoin}((tab_H[s_H]), \{([k]\setminus\{i,j\}, \{\{i,j\}\}, 0)\})).$$

The definition is explained in the next proof, but essentially for each partial solution from tab_H, we either add it to tab_G by forgetting the label i or we check that no vertex labeled i is in a same connected component as a vertex labeled j (with the acjoin operator).

Lemma 4.1. *Let $G = ren_{i\to j}(H)$ be a k-labeled graph. For each $s : [k] \setminus \{i\} \to \{1, -2, 2\}$, the table $tab_G[s]$ ac-represents $\mathcal{A}_G[s]$ assuming that $tab_H[s']$ ac-represents $\mathcal{A}_H[s']$ for each $s' : [k] \to \{1, -2, 2\}$.*

Proof. Since the operators preserve the ac-representativity, it is enough to prove that by substituting $\mathcal{A}_H$ to tab_H in the definition of $\mathcal{A}$ we have $\mathcal{A} = \mathcal{A}_G[s]$. (This will be also the case in all the subsequent correctness proofs.)

First, we prove that $\mathcal{A}_G[s] \subseteq \mathcal{A}$. Let $(p_0, p, w) \in \mathcal{A}_G[s]$ satisfying Conditions (1)-(4) of Definition 3 with F and E_s^0 the induced subgraph of G and the set of edges associated. We claim that (p_0, p, w) is added to $\mathcal{A}$.

If $V(F) \cap lab_H^{-1}(i) = \emptyset$, then $(p_0 \cup \{i\}, p, w)$ associated with (F, E_s^0) satisfies the conditions of the Definition 3 to be in $\mathcal{A}_H[s_1]$. Therefore, $(p_0, p, w) \in \mathsf{proj}(\{i\}, \mathcal{A}_H[s_1]_{\restriction p_0\cup\{i\}})$.

We may assume now that $V(F)\cap lab_H^{-1}(i) \neq \emptyset$. Suppose that $V(F)\cap lab_H^{-1}(j) = \emptyset$. If $s(j) = -2$ then $(p_0\cup\{j\}, p, w)$ associated with (F, E_s^0) satisfies the conditions to be in $\mathcal{A}_H[s_2]$ and $(p_0, p, w) \in \mathsf{proj}(\{j\}, \mathcal{A}_H[s_2]_{\restriction p_0\cup\{j\}})$. Otherwise, if $s(j) > 0$ then let p' be the partition obtained from p where j as been substitute by i. It is easy to see $(p_0\cup\{j\}, p', w)$ associated with (F, E_s^0) satisfies the conditions to be in $\mathcal{A}_H[s_3]$. One then easily checks that $(p_0, p, w) \in \mathsf{proj}(\{i\}, \mathsf{acjoin}(\mathcal{A}_H[s_3]_{\restriction p_0\cup\{j\}}, \{([k]\setminus\{i,j\}, \{(i,j)\}, 0)\}))$.

From now, we assume that both $V(F) \cap lab_H^{-1}(j)$ and $V(F) \cap lab_H^{-1}(j)$ are non-empty. Therefore, $|V(F) \cap lab_G^{-1}(j)| \geq 2$ and $s(j) \in \{-2, 2\}$. Let $s_H : [k] \to$

$\{1,2,-2\}$ with $s_H(\ell) = s(\ell)$ for $\ell \in [k] \setminus \{i,j\}$, and for $t \in \{i,j\}$,

$$s_H(t) = \begin{cases} 1 & \text{if } |V(F) \cap lab_H^{-1}(t)| = 1, \\ s(j) & \text{if } |V(F) \cap lab_H^{-1}(t)| \geq 2. \end{cases}$$

Let p' the partition on $s_H^{-1}(\{1,2\}) \cup \{v_0\}$ satisfying Condition (4) with F and E_s^0. Trivially, (p_0, p', w) associated with (F, E_s^0) satisfies the conditions to be in $\mathcal{A}_H[s_H]$. If $s(j) = -2$, then $s_H(i), s_H(j) \in \{1,-2\}$ and (p_0, p, w) is clearly added by $\mathsf{proj}(\{i,j\}, (\mathcal{A}_H[s_H])_{\restriction p_0})$. Otherwise, if $s(j) = 2$ then $s_H(i), s_H(j) \in \{1,2\}$. We claim that $\mathsf{acjoin}(\{(p_0, p', w)\}, \{([k] \setminus \{i,j\}, \{\{i,j\}\}, 0)\}) \neq \emptyset$. By definition of acjoin and of $\mathsf{acyclic}$, it is enough to prove that i and j are not in the same block of p'. Assume towards a contradiction that i and j are in the same block of p'. It means that a vertex $x \in V(F) \cap lab_H^{-1}(i)$ is connected to a vertex $y \in V(F) \cap lab_H^{-1}(j)$. It is easy to see that this implies the existence of a cycle in F^+ since x_j^+ is adjacent to both x and y in F^+. This is contradiction since F^+ is supposed to be acyclic. One then easily checks, from the definition of proj and acjoin, that (p_0, p, w) belongs to $\mathsf{proj}(\{i\}, \mathsf{acjoin}(\{(p_0, p', w)\}, \{([k] \setminus \{i,j\}, \{\{i,j\}\}, 0)\})$ and is added to $\mathcal{A}$.

Let us now prove that $\mathcal{A} \subseteq \mathcal{A}_G[s]$. Assume that $(p_0, p, w) \in tab_H[s_H]$ and let (F, E_s^0) be the corresponding forest and set of edges which satisfy Conditions (1)-(4) of the Definition 3. Let (p_0', p', w) be the produced weighted partition added to $\mathcal{A}$ from (p_0, p, w). If $j \in p_0'$, $i \in p_0$ or $j \in p_0$ then (p_0', p', w) associated with (F, E_s^0) clearly satisfies the conditions to be in $\mathcal{A}_G[s]$.

We may then assume that $i \notin p_0$ and $j \notin p_0$, *i.e.*, both $V(F) \cap lab_H^{-1}(i)$ and $V(F) \cap lab_H^{-1}(j)$ are non-empty, *i.e.*, $p_0' = p_0$. If $s(j) = -2$, we have done a projection on $\{i,j\}$ after ensuring that $|V(F) \cap lab_H^{-1}(\{i,j\})| \geq 2$. By the definition of projection and of (F, E_s^0), we can therefore conclude that (p_0', p', w) associated with (F, E_s^0) satisfies the conditions to be in $\mathcal{A}_G[s]$. Observe in particular, that if $s_H(i) = s_H(j) = 1$ then Condition (3) is satisfied because by definition of proj, $\{i\}$ and $\{j\}$ cannot be blocks of p, *i.e.*, the vertex in $lab^{-1}(i)$ and the vertex in $lab^{-1}(j)$ must be connected to some vertex which label belongs to $s^{-1}(\{1,2\})$. Otherwise, if $s(j) = 2$, we first ensure that $|V(F) \cap lab_H^{-1}(\{i,j\})| \geq 2$, and we check that two vertices in $V(F) \cap lab_H^{-1}(\{i,j\})$ lie in different components (with the acjoin), and remove i from the blocks of (p_0, p, w) (with the operator proj). Now, since the operator acjoin allows (p_0, p', w) to satisfy Conditions (1)-(2) and (4), and the projection removes i from p_0, we can thus conclude that (p_0', p', w) associated with (F, E_s^0) also satisfies Conditions (1)-(4) to be in $\mathcal{A}_G[s]$. □

Computing tab_G for $G = add_{i,j}(H)$. We can suppose that H is k-labeled. Let $s : [k] \to \{1,2,-2\}$. Let $s_H : [k] \to \{1,2,-2\}$ such that $s_H(\ell) := s(\ell)$, for $\ell \notin \{i,j\}$ and

$$(s_H(i), s_H(j)) := \begin{cases} (1,1) & \text{if } s(i) = s(j) = 1, \\ (2,1) & \text{if } (s(i), s(j)) = (-2, 1), \\ (1,2) & \text{if } (s(i), s(j)) = (1, -2). \end{cases}$$

Observe that s_H may not exist if $(s(i), s(j)) \notin \{(1,1), (1,-2), (-2,1)\}$. We let $tab_G[s] := \mathsf{ac\text{-}reduce}(\mathsf{rmc}(\mathcal{A}))$ where

$$\mathcal{A} := \bigcup_{X \subseteq [k], X \cap \{i,j\} \neq \emptyset} (tab_H[s])_{\restriction X} \bigcup \bigcup_{X \subseteq [k] \setminus \{i,j\}} \mathsf{proj}(s^{-1}(-2) \cap \{i,j\}, \mathsf{acjoin}((tab_H[s_H])_{\restriction X}, \{([k] \setminus \{i,j\}, \{\{i,j\}\}, 0)\}).$$

Lemma 4.2. *Let $G = add_{i,j}(H)$ be a k-labeled graph. For each $s : [k] \to \{1, -2, 2\}$, the table $tab_G[s]$ ac-represents $\mathcal{A}_G[s]$ assuming that $tab_H[s']$ ac-represents $\mathcal{A}_H[s']$ for each $s' : [k] \to \{1, -2, 2\}$.*

Proof. We first recall that $lab_G(x) = lab_H(x)$ for all $x \in V(G) = V(H)$.

First, we prove that $\mathcal{A}_G[s] \subseteq \mathcal{A}$. Let $(p_0, p, w) \in \mathcal{A}_G[s]$ satisfying Conditions (1)-(4) of Definition 3 in G with F and E^0_s the induced subgraph of G and the set of edges associated. If $\{i, j\} \cap p_0 \neq \emptyset$, then F is an induced forest of H and clearly (p_0, p, w) associated with (F, E^0_s) satisfies the conditions to be in $\mathcal{A}_H[s]$. We can thus conclude that (p_0, p, w) is added in $\mathcal{A}$ from $(\mathcal{A}_H)_{\restriction p_0}$.

So, assume that $\{i, j\} \cap p_0 = \emptyset$. Let $X_i := V(F) \cap lab^{-1}_G(i)$, $X_j := V(F) \cap lab^{-1}_G(j)$, and let $F_H := (V(F), E(F) \setminus (X_i \times X_j))$. Observe that $(s(i), s(j))$ must be in $\{(1, 1), (1, -2), (-2, 1)\}$, otherwise, F^+ would not be acyclic. For example, if $s(i) = 2$, X_i contains at least two vertices which are adjacent to x^+_i and the vertices in X_j, since $G = add_{i,j}(H)$, we can find a C_4 in F^+. Thus, s_H is well defined and by definition, the Condition (1) is satisfied by s_H and F_H. Because we consider only irredundant k-expressions, we know that $(X_i \times X_j) \cap E(H) = \emptyset$, *i.e.*, F_H is an induced subgraph of H. Now, it is easy to see that F^+_H is acyclic because F^+ is acyclic. Let p' the partition on $s^{-1}_H(\{1, 2\}) \cup \{v_0\}$ satisfying Condition (4) with F_H. By the definition of s_H and of F_H it is straightforward to check then that (p_0, p', w) associated with (F_H, E^0_s) satisfies all the conditions to be in $\mathcal{A}_H[s_H]$. We claim that $\mathsf{acjoin}(\{(p_0, p', w)\}, \{([k] \setminus \{i, j\}, \{\{i, j\}\}, 0)\}) \neq \emptyset$. It is sufficient to prove that $\{i, j\}$ is not a block of p'. Since there is no edge between X_i and X_j, if $\{i, j\}$ is a subset of a block in p', then there is a path in F^+_H between a vertex x of X_i and a vertex y of X_j. Let us choose a shortest one P_{xy} among all such paths. Because the set of neighbors, in F^+_H, of x^+_i if $s_H(i) = 2$ (resp. x^+_j if $s_H(j) = 2$) is X_i (resp. X_j), the path P_{xy} does not go through x^+_i nor x^+_j, if they exists in F^+_H. Since $V(F^+_H) \setminus \{x^+_i, x^+_j\} = V(F^+)$, P is also a path of F^+. Moreover, P does not use the edges from $X_i \times X_j$, since they do not exists in F^+_H. Thus, there exists a cycle in F^+. This is a contradiction. It is easy to see that p is the partition obtained from p' by joining the blocks containing i and those containing j and by removing $s^{-1}(-2) \cap \{i, j\}$ from this new block. One then easily checks that $(p_0, p, w) \in \mathsf{proj}(s^{-1}(-2) \cap \{i, j\}, \mathsf{acjoin}(\mathcal{A}_H[s_H]_{\restriction p_0}, \{([k] \setminus \{i, j\}, \{\{i, j\}\}, 0)\}))$.

It remains to prove that $\mathcal{A} \subseteq \mathcal{A}_G[s]$. Let $(p_0, p, w) \in tab_H[s_H]$ and $(F_H, E^0_{s_H})$ its associated induced subgraph of H and set of edges. If $p_0 \cap \{i, j\} \neq \emptyset$, then $(p_0, p, w) \in \mathcal{A}$ and clearly (p_0, p, w) associated with $(F_H, E^0_{s_H})$ satisfies the conditions of the Definition 3 to be in $\mathcal{A}_G[s]$. So, assume that $\{i, j\} \cap p_0 = \emptyset$ and let $(p'_0, p', w) \in \mathsf{proj}(s^{-1}(-2) \cap \{i, j\}, \mathsf{acjoin}(\{(p_0, p, w)\}, \{([k] \setminus \{i, j\}, \{\{i, j\}\}, 0)\}))$. Let $F = (V(F_H), E(F_H) \cup (X_i \times X_j)$ where $X_t = V(F_H) \cap lab^{-1}_H(t)$ for $t \in \{1, 2\}$. Clearly F is an induced subgraph of G. We claim that F^+ is a forest. For the same reasons evoked earlier, $\{i, j\}$ cannot be a subset of a block of p thus the vertices in X_i and those in X_j are in different components of F^+_H. If $|X_i| = |X_j| = 1$ then $F^+ = (V(F^+_H), E(F_H) \cup (X_i \times X_j))$ and it is clearly a forest. Otherwise, by definition of s_H, either $|X_i| = 1$ and $|X_j| \geq 2$ or the inverse. Suppose *w.l.o.g.* that we are in the first case. Then, F^+ can be obtained from F^+_H by identifying the vertex x^+_j and the vertex in X_i, and this operation clearly keeps the graph acyclic since the vertex in X_i and x^+_j are not connected in F^+_H. Thus $(F, E^0_{s_H})$ satisfies Condition (2). Moreover, $(F, E^0_{s_H})$ clearly satisfies Conditions (1) and (3). Condition (4) is also satisfied since it is easy to see that p' is obtained from p by joining the block containing i and the one containing j and by removing i or j if respectively $s(i) = -2$ or $s(j) = -2$. We can therefore conclude that $tab_G[s]$ is correctly updated. □

Computing tab_G for $G = G_1 \oplus G_2$. We can suppose that G_1 and G_2 are both k-labeled. Let $s : [k] \to \{1, 2, -2\}$. For $Y_1, Y_2 \subseteq s^{-1}(-2)$, we say that s, $s_1 : [k] \to \{1, 2, -2\}$ and $s_2 : [k] \to \{1, 2, -2\}$ *u-agree on* (Y_1, Y_2) if for each $i \in [k]$,

$$s(i) = \begin{cases} s_1(i) = s_2(i) = -2 & \text{if } i \in Y_1 \cap Y_2, \\ s_2(i) & \text{if } i \in Y_1, \\ s_1(i) & \text{if } i \in Y_2, \\ 2 & \text{if } i \in [k] \setminus (Y_1 \cup Y_2) \text{ and } s_1(i), s_2(i) \in \{1, 2\}, \\ -2 & \text{if } i \in [k] \setminus (Y_1 \cup Y_2) \text{ and } (s_1(i), s_2(i)) \in \{(1, -2), (-2, 1), (1, 1), (-2, -2)\}. \end{cases}$$

Typically, for $i \in \{1, 2\}$, Y_i will represent the set of non-intersected labels of a partial solution of G_i. Observe that if $(s_1(i), s_2(i)) = (1, 1)$ then $s(i)$ can take as values 2 or -2.

We let $tab_G[s] := \mathsf{ac\text{-}reduce}(\mathsf{rmc}(\mathcal{A}))$ where,

$$\mathcal{A} := \bigcup_{Y_1, Y_2 \subseteq s^{-1}(-2)} \bigcup_{\substack{s, s_1, s_2 \\ \text{u-agree on} \\ (Y_1, Y_2)}} \mathsf{acjoin}(\mathsf{proj}(s^{-1}(-2) \setminus Y_1, (tab_{G_1}[s_1])_{\restriction Y_1}), \mathsf{proj}(s^{-1}(-2) \setminus Y_2, (tab_{G_2}[s_2])_{\restriction Y_2})).$$

Lemma 4.3. *Let $G = G_1 \oplus G_2$ be a k-labeled graph. For each $s : [k] \to \{1, -2, 2\}$, the table $tab_G[s]$ ac-represents $\mathcal{A}_G[s]$ assuming that $tab_{G_1}[s']$ and $tab_{G_2}[s']$ ac-represents $\mathcal{A}_{G_1}[s']$ and of $\mathcal{A}_{G_2}[s']$, respectively, for each $s' : [k] \to \{1, -2, 2\}$.*

Proof. First, we prove that $\mathcal{A}_G[s] \subseteq \mathcal{A}$. Let $(p_0, p, w) \in \mathcal{A}_G[s]$ and (F, E^0_s) its associated induced subgraph and set of edges according to Definition 3. For $t \in \{1, 2\}$, let $F_t := G_t[V(F) \cap V(G_t)]$ and $E^0_{s_t} := \{v_0\} \times V(F_t) \cap E^0_s$, $w_t := \mathsf{w}(V(F_t))$, $p^t_0 := \{i \in [k] \mid V(F_t) \cap lab^{-1}_{G_t}(i) = \emptyset\}$, and let $s_t : [k] \to \{1, 2, -2\}$ such that

$$s_t(i) := \begin{cases} -2 & \text{if } i \in p^t_0, \\ 1 & \text{if } |V(F_t) \cap lab^{-1}_{G_t}(i)| = 1, \\ s(i) & \text{if } |V(F_t) \cap lab^{-1}_{G_t}(i)| \geq 2. \end{cases}$$

It is a straightforward check to verify that (s, s_1, s_2) u-agree on (p^1_0, p^2_0). Let p_t the partition on $s^{-1}_t(\{1, 2\}) \cup \{v_0\}$ satisfying Condition (4) with F^+_t. Let $t \in \{1, 2\}$. We claim that (p^t_0, p_t, w_t) associated with $(F_t, E^0_{s_t})$ satisfies the conditions to be in $\mathcal{A}_{G_t}[s_t]$. By definition of p^t_0, p_t and s_t, and of $(F_t, E^0_{s_t})$ it is straightforward to check that Conditions (1) and (4) are satisfied. Because $s^{-1}_t(2) \subseteq s^{-1}(2)$, and $F = F_1 \oplus F_2$, we can conclude that F^+_t is an induced subgraph of F^+. Because F^+ is acyclic, we can conclude that F^+_t is acyclic, *i.e.*, Condition (2) is satisfied. Because $F = F_1 \oplus F_2$, each component C of F is either a component of F_1 or of F_2. Now, if a component C does not intersect $s^{-1}_t(\{1, 2\}) \cup \{v_0\}$, then it is entirely contained in $\bigcup_{j \in s^{-1}_t(-2)} V(F_t) \cap lab^{-1}_{G_t}(j)$. But, this yields a contradiction with (F, E^0_s) satisfying Condition (3) because $s^{-1}_t(-2) \subseteq s^{-1}(-2)$, and C is a component of F. Therefore Condition (3) is also satisfied. We can thus conclude that Conditions (1)-(4) are satisfied by $(F_t, E^0_{s_t})$ and (p^t_0, p_t, w_t). It remains to prove that

$$(p_0, p, w) \in \mathsf{acjoin}(\mathsf{proj}(s^{-1}(-2) \setminus Y_1, \{(p^1_0, p_1, w_1)\}), \mathsf{proj}(s^{-1}(-2) \setminus Y_2, \{(p^2_0, p_2, w_2)\})).$$

First, observe that, for $t \in \{1, 2\}$, we have $\mathsf{proj}(s^{-1}(-2) \setminus Y_t, \{(p^t_0, p_t, w_t)\})) \neq \emptyset$ since F_t satisfies Condition (3). Let $V = s^{-1}(\{1, 2\}) \cup \{v_0\}$ and let $p'_t = ((p_t)_{\downarrow V})_{\uparrow V}$. We claim that $\mathsf{acyclic}(p'_1, p'_2)$ holds. Assume towards a contradiction that it is not the case. By the graphical definition of $\mathsf{acyclic}$, we can easily see that it implies the existence of a sequence $i_1, \ldots, i_{2r}$ of $s^{-1}(\{1, 2\}) \cup \{v_0\}$ such that

$$i_{2r} \sim_{F^+_1} i_1 \text{ and for all } 1 \leq \alpha < r, \text{ we have } i_{2\alpha-1} \sim_{F^+_2} i_{2\alpha} \text{ and } i_{2\alpha} \sim_{F^+_1} i_{2\alpha+1}.$$

Now observe that for all $1 \leq \alpha \leq r$, if $i_\alpha \sim_{F_t^+} i_{\alpha+1}$ then there exists a vertex in $lab_G^{-1}(i_\alpha) \cap V(F_t)$ connected to a vertex in $lab_G^{-1}(i_{\alpha+1}) \cap V(F_t)$. We can then construct a cycle in F^+ from this sequence since $V(F_1) \cap V(F_2) = \emptyset$, F_1^+ and F_2^+ are subgraphs of F^+ and for all $i \mathsf{ins}^{-1}(2)$, the vertices labeled i are connected in F^+ through v_i^+. This is a contradiction since F^+ verifies Condition (2), thus $\mathsf{acyclic}(p_1', p_2')$ holds. One then easily checks that

$$(p_0^1 \cap p_0^2, p_1' \sqcup p_2', w_1+w_2) \in \mathsf{acjoin}(\mathsf{proj}(s^{-1}(-2)\backslash Y_1, \{(p_0^1, p_1, w_1)\}), \mathsf{proj}(s^{-1}(-2)\backslash Y_2, \{(p_0^2, p_2, w_2)\})).$$

Now, we can claim that $p = p_1' \sqcup p_2'$. For doing so, it is sufficient to look the definition of $\sqcup$ and to observe that $\sim_{F^+}$ is the transitive closure of the relation $\mathcal{R}$ where $i\mathcal{R}j$ if $i \sim_{F_1^+} j$ or $i \sim_{F_2^+} j$. One then easily checks that $w = w_1 + w_2$ and $p_0 = p_0^1 \cap p_0^2$ and concludes then that (p_0, p, w) is added to $\mathcal{A}$ through (p_0^1, p_1, w_1) and (p_0^2, p_2, w_2).

We now prove that $\mathcal{A} \subseteq \mathcal{A}_G[s]$, *i.e.*, if $(p_0, p, w) \in \mathsf{acjoin}(\mathsf{proj}(s^{-1}(-2)\backslash p_0^1, \{(p_0^1, p_1, w_1)\}), \mathsf{proj}(s^{-1}(-2) \setminus p_0^2, \{(p_0^2, p_2, w_2)\}))$ is added to $tab_G[s]$, then $(p_0, p, w) \in \mathcal{A}_G[s]$. Let $(F_1, E^0_{s_1})$ and $(F_2, E^0_{s_2})$ be associated with $(p_0^1, p_1, w_1) \in tab_{G_1}[s_1]$ and $(p_0^2, p_2, w_2) \in tab_{G_2}[s_2]$, respectively, with s, s_1, s_2 u-agreeing in (p_0^1, p_0^2). We claim that (p_0, p, w) associated with $(F, E^0_{s_1} \cup E^0_{s_2})$ where $F = (V(F_1) \cup V(F_2), E(F_1) \cup E(F_2))$, satisfies Conditions (1)-(4), *i.e.*, $(p_0, p, w) \in \mathcal{A}_G[s]$. Because s, s_1, s_2 u-agree on (p_0^1, p_0^2), then $p_0 = p_0^1 \cap p_0^2$ and by the definition of the entries of the tables, and of (p_0, p, w), we can conclude that $p \in \Pi(s^{-1}(\{1, 2\}) \cup \{v_0\})$. Also, because s, s_1, s_2 u-agree on (p_0^1, p_0^2) it is easy to prove that Condition (1) is also satisfied. Assume towards a contradiction that Condition (2) is not satisfied, *i.e.*, there exists a cycle C in F^+. Since both F_1^+ and F_2^+ are acyclic, C must be a cycle alternating between paths in F_1^+ and paths in F_2^+. One can easily check that this implies the existence of a sequence $i_1, \ldots, i_{2r}$ of $s^{-1}(\{1, 2\}) \cup \{v_0\}$ such that

$$i_{2r} \sim_{F_1^+} i_1 \text{ and for all } 1 \leq \alpha < r, \text{ we have } i_{2\alpha-1} \sim_{F_2^+} i_{2\alpha} \text{ and } i_{2\alpha} \sim_{F_1^+} i_{2\alpha+1}.$$

Moreover, it is easy to infer, from this sequence and the graphical definition of $\mathsf{acyclic}$, that $\mathsf{acyclic}(((p_1)_{\downarrow V})_{\uparrow V}, ((p_2)_{\downarrow V})_{\uparrow V})$ doesn't hold, where $V = s^{-1}(\{1, 2\}) \cup \{v_0\}$. This is a contradiction since (p_0, p_t) is supposed to be added in $\mathcal{A}$ from (p_0^1, p_1, w_1) and (p_0^2, p_2, w_2). If we suppose that Condition (3) is not satisfied, then there is a connected component C of F that does not intersect $s^{-1}(\{1, 2\}) \cup \{v_0\}$, *i.e.*, C is fully contained in $lab_G^{-1}(s^{-1}(-2))$. Since $F = F_1 \oplus F_2$, C is either a connected component of F_1 or of F_2. Suppose *w.l.o.g.* that C is a connected component of F_1. Observe that C intersects $lab_{G_1}^{-1}(s_1^{-1}(\{1, 2\}))$ because F_1 satisfies Condition (3) in G_1. Moreover, C does not intersect $lab_{G_1}^{-1}(s_1^{-1}(2))$, otherwise C would intersect $lab_G^{-1}(s^{-1}(2))$ since $s_1(i) = 2$ implies $s(i) = 2$, for all $i \in [k]$. Thus C is a connected component of F_1^+ and $b_C := \{i \in s_1^{-1}(1) \mid C \cap lab_{G_1}^{-1}(i) \neq \emptyset\}$ is, by definition, a block of p_1. Moreover, we have $b_C \subseteq s^{-1}(\{-2\}) \setminus p_0^1$ since C does not intersect any vertex in $s^{-1}(\{1, 2\}) \cup \{v_0\}$. Thus, by definition of proj, we have $\mathsf{proj}(s^{-1}(-2) \setminus p_0^1, \{(p_0^1, p_1, w_1)\}) = \emptyset$ which contradicts the fact that (p_0, p, w) is produced from (p_0^1, p_1, w_1) and (p_0^2, p_2, w_2). Condition (2) is then satisfied.

For all $i \in [k]$, let $X_i := F \cap lab_G^{-1}(i)$, and $X_{v_0} := \{v_0\}$. It remains to prove that Condition (4) is satisfied, *i.e.*, for all labels i, j, we have i and j are in the same block of p if and only if a vertex in X_i is connected, in F^+, to a vertex in X_j. By definition of acjoin and proj, we have $p = ((p_1)_{\downarrow V})_{\uparrow V} \sqcup ((p_2)_{\downarrow V})_{\uparrow V}$ where $V = s^{-1}(\{1, 2\}) \cup \{v_0\}$. Thus two labels i and j are in the same block of p if and only if there exists a sequence $b_1, b_2, \ldots, b_r$ of blocks that alternatively belong to p_1 and p_2 and such that $i \in b_1$, $j \in b_r$ and for all $l \in [r-1]$, $b_l \cap b_{l+1} \neq \emptyset$. Hence, it is sufficient to prove that for all $b_1' \in p_1$ and $b_2' \in p_2$, we have $b_1' \cap b_2' \neq \emptyset$ if and only if for all $i \in b_1'$ and $j \in b_2'$, a vertex of X_i is connected to a vertex of X_j in

F^+. Let $(i,j) \in b'_1 \times b'_2$ and $l \in b'_1 \cap b'_2$. Since p_1 satisfies Condition (4), a vertex of X_i is connected, in F_1^+, to a vertex from X_l and since F_1^+ is a subgraph of F^+, we can also conclude that a vertex of X_i is connected, in F^+, to a vertex from X_l. Symmetrically, a vertex of X_l is connected, in F^+, to a vertex from X_j. If $|X_l| = 1$ (*i.e.*, $l = v_0$) then we are done, otherwise $s(l) = 2$ and by definition of F^+ all the vertices in X_l are connected to x_l^+. Hence, Condition (4) is also satisfied.

We can therefore conclude that $tab_G[s]$ is correctly updated. □

Theorem 4.4. *There is an algorithm that, given an n-vertex graph G and an irredundant clique-width k-expression of G, computes a minimum feedback vertex set in time $3^{3k} \cdot 2^{(\omega+1)k} \cdot n \cdot k^{O(1)}$.*

Proof. We do a bottom-up traversal of the clique-width expression and at each step we update the tables as indicated above. The correctness of the algorithm follows from Lemmas 4.1-4.3. Let us discuss the time complexity now. If $G = add_{i,j}(H)$ or $G = ren_{i\to j}(H)$, and $s : [k] \to \{1, 2, -2\}$, then we update $tab_G[s]$ from a constant number of tables from tab_H, each identified in $O(k)$ time from s. Since each table contains at most $(k+1)\cdot 2^k$ entries, we can thus update tab_G in time $2^{(\omega-1)k} \cdot k^{O(1)}$. If $G = G_1 \oplus G_2$ and $s : [k] \to \{1, 2, -2\}$, then one can easily bound the size of $\mathcal{A}$ by the sum over all functions $s_1, s_2 : [k] \to \{1, 2, -2\}$ of $|tab_{G_1}[s_1]| \cdot |tab_{G_2}[s_2]|$. Since there is at most 3^{2k} such functions, we can conclude that $|\mathcal{A}| \leq 3^{2k} \cdot 2^{2k} \cdot k^{O(1)}$. Therefore, $tab_G[s]$ is computed in time $3^{2k} \cdot 2^{(\omega+1)k} \cdot k^{O(1)}$, *i.e.*, we update tab_G in time $3^{3k} \cdot 2^{(\omega+1)k} \cdot k^{O(1)}$. Because the size of a clique-width expression is linear in n, we can conclude that a minimum feedback vertex set can be computed in the given time. □

5. (Out)-Connected (σ, ρ)-Dominating Sets

We will show here how to use the operators defined in [1] in order to obtain singly-exponential time algorithms for computing minimum or maximum connected (σ, ρ)-dominating sets in graphs of bounded clique-width. We let $opt \in \{\min, \max\}$, *i.e.*, we are interested in computing a (out)-connected (σ, ρ)-dominating set of maximum (or minimum) weight if $opt = max$ (or $opt = min$). Let us first give some definitions.

Let V and S two disjoint finite sets. As defined in Section 3, rmc works only for the case $opt = min$, we redefine it as follows in order to take into account maximisation problems. For $\mathcal{A} \subseteq \Pi(V, S) \times \mathbb{N}$, we define $\mathsf{rmc}(\mathcal{A}) \subseteq \mathcal{A}$ as

$$\mathsf{rmc}(\mathcal{A}) := \{(p_0, p, w) \in \mathcal{A} \mid \forall (p_0, p, w') \in \mathcal{A}, opt(w, w') = w\}.$$

Join. Let V' and S' be two disjoint finite sets. For $\mathcal{A} \subseteq \Pi(V, S) \times \mathbb{N}$ and $\mathcal{B} \subseteq \Pi(V', S') \times \mathbb{N}$, we define $\mathsf{join}(\mathcal{A}, \mathcal{B}) \subseteq \Pi(V \cup V', (S \setminus V') \cup (S' \setminus V)) \times \mathbb{N}$ as

$$\mathsf{join}(\mathcal{A}, \mathcal{B}) := \mathsf{rmc}\left(\{(p_0 \setminus (V' \cup S')) \cup (q_0 \setminus (V \cup S)) \cup (p_0 \cap q_0), p_{\uparrow(V')} \sqcup q_{\uparrow(V)}, w_1 + w_2) \mid (p_0, p, w_1) \in \mathcal{A}, (q_0, q, w_2) \in \mathcal{B}\}\right).$$

This operator is the one from [1]. It is used mainly to construct partial solutions of $G \oplus H$ from partial solutions of G and H.

Proposition 5.1 (Folklore)**.** *The operator* join *can be performed in time $|\mathcal{A}| \cdot |\mathcal{B}| \cdot |V \cup S \cup V' \cup S'|^{O(1)}$ and the size of its output is upper-bounded by $|\mathcal{A}| \cdot |\mathcal{B}|$.*

The following is the same as Definition 2, but does not require acyclicity.

Definition 4 ([1])**.** For $\mathcal{A} \subseteq \Pi(V, S) \times \mathbb{N}$ and $(q_0, q) \in \Pi(V, S)$, let

$$\mathbf{opt}(\mathcal{A}, (q_0, q)) := opt\{w \mid (q_0, p, w) \in \mathcal{A},\ p \sqcup q = \{V \setminus q_0\}\}.$$

A set of weighted partitions $\mathcal{A}' \subseteq \Pi(V,S) \times \mathbb{N}$ *represents* $\mathcal{A}$ if for each $(q_0, q) \in \Pi(V,S)$, it holds that $\mathbf{opt}(\mathcal{A}, (q_0, q)) = \mathbf{opt}(\mathcal{A}', (q_0, q))$.

Let Z be a finite set and V', S' be two disjoint finite sets. A function $f : 2^{\Pi(V,S)\times\mathbb{N}} \times Z \to 2^{\Pi(V',S')\times\mathbb{N}}$ is said to *preserve representation* if for each $\mathcal{A}, \mathcal{A}' \subseteq \Pi(V,S) \times \mathbb{N}$ and $z \in Z$, it holds that $f(\mathcal{A}', z)$ represents $f(\mathcal{A}, z)$ whenever $\mathcal{A}'$ represents $\mathcal{A}$.

Lemma 5.2 ([1]). *The operators* rmc, proj, join *and* $\cup$ *preserve representation.*

Proof. Since, for all $(q_0, q) \in \Pi(V,S)$, we have, by definition, $\mathbf{opt}(\mathcal{A}, (q_0, q)) = \mathbf{opt}(\mathcal{A}_{\restriction q_0}, (q_0, q))$, we can conclude that $\mathcal{A}'$ represents $\mathcal{A}$ if and only if ${\mathcal{A}'}_{\restriction X}$ represents $\mathcal{A}_{\restriction X}$ for all $X \subseteq S$. Also,

$$\mathbf{opt}(\mathsf{join}(\mathcal{A}, \mathcal{A}'), (r_0, r)) = opt_{X \subseteq V}\{\mathbf{opt}(\mathsf{join}(\mathcal{A}_{\restriction X}, \mathcal{A}'), (r_0, r))\}.$$

Since it is proved in [1] that the operators rmc, proj, join and $\cup$ preserve the representativeness on $(\Pi(V,S) \times \mathbb{N})_{\restriction X}$ for all $X \subseteq S$, we are done. □

Theorem 5.3 ([1]). *Let X be a subset of S. There exists an algorithm* reduce *that given a set of weighted partitions $\mathcal{A} \subseteq X \times \Pi(V \setminus X) \times \mathbb{N}$, outputs in time $|\mathcal{A}| \cdot 2^{(\omega-1)|V|} \cdot |V|^{O(1)}$ a subset $\mathcal{A}'$ of $\mathcal{A}$ that represents $\mathcal{A}$, and such that $|\mathcal{A}'| \leq 2^{|V|-1}$ and $\sum_{(p_0,p,w)\in\mathcal{A}'} w$ is optimum.*

We now concentrate on connected (σ, ρ)-dominating sets (the modifications for computing out-connected (σ, ρ)-dominating sets are then after straightforward). We also let (σ, ρ) be a fixed pair of non-empty finite or co-finite subsets of $\mathbb{N}$. Moreover, let $d := \max\{d(\sigma), d(\rho)\}$, where $d(\mathbb{N}) := 0$, and for a (non-empty finite or co-finite) subset $\mu \subset \mathbb{N}$, $d(\mu) := \max(\mu)$ if μ is finite, otherwise, $d(\mu) := \max(\mathbb{N} \setminus \mu) + 1$. We will use the symbol $+\infty$ in order to decide if a set of vertices of size greater than d is included in a $(\sigma,\ \rho)$-dominating set. For each $D \subseteq V(G)$ and $i \in [k]$, let

$$r^i_G(D) := \begin{cases} +\infty & \text{if } lab_G^{-1}(i) \subseteq D \text{ and } |lab_G^{-1}(i)| > d, \\ \min(d, |lab_G^{-1}(i) \cap D|) & \text{otherwise,} \end{cases}$$

and let $r_G(D) := (r^1_G(D), \ldots, r^k_G(D))$. If $D \subseteq V(G)$ is a partial solution then the sequence $r_G(D)$ will be the information we keep to describe how D intersects the different label classes. For all $i \in [k]$, observe that $lab_G^{-1}(i) \subseteq D$ if and only if $r_G(D) = r_G(lab^{-1}(i))$. Moreover, notice that $|\{r_G(D) \mid D \subseteq V(G)\}| \leq (d+2)^k$.

We now define a graph from G that describes future extensions of G using clique-width operations. Let $V^+ := \{v^1_1, \ldots, v^1_d, v^2_1, \ldots, v^2_d, \ldots, v^k_1, \ldots, v^k_d\}$ be a set disjoint from $V(G)$ and of size $d \cdot k$. For $R' := (R_1, \ldots, R_k) \subseteq \{0, 1, \ldots, d\}^k$, let $V^+(R') := V^+(R'_1) \cup \cdots \cup V^+(R'_k)$ with

$$V^+(R'_i) := \begin{cases} \emptyset & \text{if } R'_i = 0, \\ \{v^i_1, \ldots, v^i_{R'_i}\} & \text{otherwise,} \end{cases}$$

and let $\mathsf{f}_{R'}(G)$ be the graph with vertex set $V(G) \cup V^+(R')$ and edge set $E(G) \cup E_1 \cup \cdots \cup E_k$ with $E_i = lab_G^{-1}(i) \times V^+(R'_i)$. Observe that E_i is empty if $lab_G^{-1}(i) = \emptyset$ or $V^+(R'_i) = \emptyset$. The following describes the tables manipulated by our algorithm.

Definition 5. For a k-labeled graph G and $R \in \{0,\dots,d,+\infty\}^k, R' \in \{0,\dots,d\}^k$, let $p_0 = \{i \in [k] \mid R_i = 0 \text{ or } R'_i = 0\}$,

$$\begin{aligned}\mathcal{D}_G[R,R'] := \big\{(p_0,p,w) \in \Pi([k]\setminus p_0, p_0) \mid \exists D \subseteq V(G), r_G(D) = R, \mathsf{w}(D) = w,\\ D \cup V^+(R') \ (\sigma,\rho)\text{-dominates } V(G) \text{ in } \mathsf{f}_{R'}(G),\\ p = ([k]\setminus p_0)/\sim_{\mathsf{f}_{R'}(G)}, \text{ and}\\ G[D] \text{ is connected if } R' = \{0\}^k, \text{ otherwise}\\ \forall C \in CC(G[D]), \exists x \in C \text{ with } lab_G(x) \notin p_0\big\}.\end{aligned}$$

Where $\sim_{\mathsf{f}_{R'}(G)}$ is the equivalence relation such that $i \sim_{\mathsf{f}_{R'}(G)} j$ if there exists a vertex in $lab_G^{-1}(i)$ connected to a vertex in $lab_G^{-1}(j)$ in $\mathsf{f}_{R'}(G)$.

It is straightforward to check that the optimum value is the optimum over all $R \in \{0,1,\dots,d,+\infty\}^k$ of $opt\{w \mid ([k],\emptyset,w) \in \mathcal{D}_G[R,\{0\}^k]\}$ for a k-labeled graph G. As in Section 4 our dynamic programming algorithm will store a subset of $\mathcal{D}_G[R,R']$ of size 2^{k-1} that represents it. Recall that we suppose that any graph is given with an irredundant clique-width k-expression. Also, in order to simplify the steps of the algorithm, we will always assume that the given graph is k-connected, at the cost of adding the non-necessary labels in p_0 for each considered weighted partition (p_0,p,w).

Computing tab_G for $G = \mathbf{1}(x)$. For $R \in \{0,\dots,d,+\infty\}^k, R' \in \{0,\dots,d\}^k$, let

$$tab_G[R,R'] := \begin{cases} \emptyset & \text{if } R_1 \notin \{0,1\} \text{ or } R_i \neq 0 \text{ for } i \neq 1,\\ \{\,([k],\emptyset,0)\,\} & \text{if } R_1 = 0 \text{ and } R'_1 \in \rho,\\ \{\,([k]\setminus\{1\},\emptyset,\mathsf{w}(x))\,\} & \text{if } R_1 = 1,\ R' = \{0\}^k \text{ and } 0 \in \sigma,\\ \{\,([k]\setminus\{1\},\{\{1\}\},\mathsf{w}(x))\,\} & \text{if } R_1 = 1,\ R'_1 \neq 0 \text{ and } R'_1 \in \sigma,\\ \emptyset & \text{otherwise.}\end{cases}$$

Since there is only one vertex in G, which is labeled 1, only R_1 may be different from 0 and cannot exceed 1. Also, the possible solutions are either to put x in the solution ($R_1 = 1$) or to discard it ($R_1 = 0$); in both cases we should check that it is (σ,ρ)-dominated. Also, when we include it in a solution, whenever $R' \neq \{0\}^k$, R'_1 should be different from 0. Thus, the fact that $tab_G[R,R']$ represents $\mathcal{D}_G[R,R']$ is clear from the construction.

Computing tab_G for $G = ren_{i\to j}(H)$. We can suppose that H is k-labeled. Let $R \in \{0,\dots,d,+\infty\}^k$ and $R' \in \{0,\dots,d\}^k$. If $R_i \neq 0$, we let $tab_G[R,R'] := \emptyset$. Otherwise, we let $S' \in \{0,\dots,d\}^k$ such that $S'_\ell = R'_\ell$, $\forall \ell \neq i$ and $S'_i = R'_j$ and we set $tab_G[R,R'] := \mathsf{reduce}(\mathsf{rmc}(\mathcal{A}))$ with

$$\mathcal{A} := \begin{cases} \bigcup\limits_{S\in\mathcal{S}} tab_H[S,S'] & \text{if } R_j = 0 \text{ or } R'_j = 0,\\ \mathsf{join}(\ \{([k],\emptyset,0)\},\ \mathsf{proj}(\{i\},\ \mathcal{A}')) & \text{otherwise,}\end{cases}$$

where

$$\mathcal{A}' := \bigcup_{S\in\mathcal{S}} \mathsf{join}(tab_H[S,S'],\ \{([k]\setminus\{i,j\},\{\{i,j\}\},0)\}\ ).$$

and

$$\begin{aligned}\mathcal{S} :=& \big\{S \in \{0,\dots,d,+\infty\}^k \mid (\forall \ell \notin \{i,j\},\ S_\ell\ = R_\ell) \wedge \big[(R_j = \min(d, S_i + S_j))\\ &\vee \big((R_j = +\infty) \wedge (d < S_i + S_j) \wedge (\forall t \in \{i,j\},\ S_t = r_H^t(lab_H^{-1}(t))\,)\,\big)\big]\big\}.\end{aligned}$$

Lemma 5.4. *Let $G = ren_{i\to j}(H)$ be a k-labeled graph. For $R,R' \in \{0,1,\dots,d,+\infty\}^k$, the table $tab_G[R,R']$ represents $\mathcal{D}_G[R,R']$ assuming that $tab_H[S,S']$ represents $\mathcal{D}_H[S,S']$, for all $S,S' \in \{0,1,\dots,d,+\infty\}^k$.*

Proof. Notice that as in the proofs of Lemmas 4.1-4.3, it is enough to prove that by substituting $\mathcal{D}_H$ to tab_H in the definition of $\mathcal{A}$ we have $\mathcal{A} = \mathcal{D}_G[R, R']$ as the operators preserve the representativity (and this is the case for the next two lemmas).

First we prove that $\mathcal{A} \subseteq \mathcal{D}_G[R, R']$. Observe that in G no vertex is labeled by i, and thus $\mathcal{D}_G[R, R']$ is empty if $R_i \neq 0$. Assume now that $R_i = 0$ and let $S \in \mathcal{S}$. Let $(p_0, p, w) \in \mathcal{D}_H[S, S']$ and D its associated set from Definition 5. One can check from the definition of $\mathcal{S}$ that $r_G(D) = R$. Suppose that $R_j = 0$ or $R'_j = 0$. We claim that (p_0, p, w) is a valid entry in $\mathcal{D}_G[R, R']$. It is clear that $p_0 = \{\ell \in [k] \mid R_\ell = 0 \text{ or } R'_\ell = 0\}$ since either $R_j = 0$ and then $S_i = S_j = 0$ or $R'_j = 0$ and then $S'_i = S'_j = 0$. Moreover, $D \cup V^+(R')$ (σ, ρ)-dominates $V(G)$ in $\mathsf{f}_{R'}(G)$ because $D \cup V^+(S')$ (σ, ρ)-dominates $V(G) = V(H)$ in $\mathsf{f}_{S'}(H)$ and $R'_j = S'_i = S'_j$. We can easily check that (p_0, p, w) satisfies all the other conditions to be in $\mathcal{D}_G[R, R']$. Now suppose that $R_j \neq 0$ and $R'_j \neq 0$. Let $\{(p'_0, p', w)\}$ be the result of

$$\mathsf{join}(\{([k], \emptyset, 0)\},\ \mathsf{proj}(\{i\},\ \mathsf{join}(\{(p_0, p, w)\},\ \{([k] \setminus \{i, j\}, \{\{i, j\}\}, 0)\}))).$$

We claim that (p'_0, p', w) is a valid entry of $\mathcal{D}_G[R, R']$ when associated with D. First, observe that applying proj and join with respectively i and $\{([k], \emptyset, 0)\}$ as arguments guarantees that $p'_0 = p_0 \cup \{i\}$ whether $i \in p_0$ or not. Thus $p'_0 = \{\ell \in [k] \mid R_\ell = 0 \text{ or } R'_\ell = 0\}$. Moreover, it is easy to see that the neighborhood of each vertex in $V(G)$ does not change from $\mathsf{f}_{R'}(G)$ to $\mathsf{f}_{S'}(H)$, except for the vertices in $lab_H^{-1}(i)$. Furthermore, for all vertices $v \in lab_H^{-1}(i)$, we have $N_{\mathsf{f}_{R'}(G)}(v) = (N_{\mathsf{f}_{S'}(H)}(v) \setminus V^+(S'_i)) \cup V^+(R'_j)$. Thus $D \cup V^+(R')$ (σ, ρ)-dominates $V(G)$ in $\mathsf{f}_{R'}(G)$ since $S'_i = R'_j$. One can easily check that p' is equal to p if $S_i = 0$ or p' is obtained from p by merging the block containing i and the block containing j and by removing i. From these observations, one can infer that p' corresponds to $([k] \setminus p'_0)/\sim_{\mathsf{f}_{R'}(G)}$ since, if $S_i \neq 0$ then the vertices in $lab_H^{-1}(j)$ are connected to the vertices in $lab_H^{-1}(i)$ in $\mathsf{f}_{R'}(G)$ through the vertices in $V^+(R'_j)$. Finally, all the connected components of $G[D]$ intersect $V(lab_G^{-1}([k] \setminus p_0))$ since $V(lab_H^{-1}(p_0)) = V(lab_G^{-1}(p'_0))$. We can conclude that each weighted partition added to $\mathcal{A}$ is correct.

Assume now that (p_0, p, w) belongs to $\mathcal{D}_G[R, R']$ and let D be its associated set from Definition 5. Let $S := r_H(D)$ and $p'_0 := \{\ell \in [k] \mid S_\ell = 0 \text{ or } S'_\ell = 0\}$. By definition of $\mathcal{S}$, it is clear that $S \in \mathcal{S}$. Moreover, with the same arguments evoked above, one can can easily check that $(p'_0,\ ([k] \setminus p'_0)/\sim_{\mathsf{f}_{S'}(H)},\ w) \in \mathcal{D}_H[S, S']$ and that (p_0, p, w) is added in $\mathcal{A}$. Thus $\mathcal{D}_G[R, R'] \subseteq \mathcal{A}$. □

Computing tab_G for $G = add_{i,j}(H)$. We can suppose that H is k-labeled. Let $R \in \{0, \ldots, d, +\infty\}^k$ and $R' \in \{0, \ldots, d\}^k$. For $t \in \{i, j\}$, we let $\mathcal{P}$ be the following predicate

$$\mathcal{P}(t) := \big(\sigma, \rho \text{ are co-finite}\big) \vee \big(\rho \text{ is co-finite} \wedge R_t = 0\big) \vee \big(\sigma \text{ is co-finite} \wedge R_t = r_G^t(lab_G^{-1}(t))\big).$$

Let, if it exists, $S' \in \{0, \ldots, d\}^k$ such that $S'_\ell = R'_\ell$, for all $\ell \in [k] \setminus \{i, j\}$ and

$$S'_i := \begin{cases} R_j + R'_i & \text{if } R_j + R'_i \leq d, \\ d & \text{if } \mathcal{P}(i), \end{cases} \quad \text{and,} \quad S'_j := \begin{cases} R_i + R'_j & \text{if } R_i + R'_j \leq d, \\ d & \text{if } \mathcal{P}(j). \end{cases}$$

We set $tab_G[R, R'] := \mathsf{reduce}(\mathsf{rmc}(\mathcal{A}))$ with

$$\mathcal{A} := \begin{cases} \{([k],\ \emptyset,\ w) \mid (p_0, p, w) \in tab_H[R, S']\} & \text{if } R' = \{0\}^k, \\ tab_H[R, S'] & \text{if } R_i = 0 \text{ or } R_j = 0, \\ \mathsf{join}(\{([k],\ \emptyset,\ 0)\},\ \mathsf{proj}(\{\ell \in \{i, j\} \mid R'_\ell = 0\},\ \mathcal{A}')) & \text{otherwise,} \end{cases}$$

where

$$\mathcal{A}' := \mathsf{join}(\ tab_H[R, S'],\ \{([k] \setminus \{i,j\}, \{\{i,j\}\}, 0)\})$$

We essentially merge the connected components containing vertices labeled i and those labeled j, and check at the same time whether the resulting set is still a (σ,ρ)-dominating set (by the definition of S'_i and S'_j)

Lemma 5.5. *Let $G = add_{i,j}(H)$ be a k-labeled graph. For $R, R' \in \{0,1,\dots,d\}^k$, the table $tab_G[R,R']$ represents $\mathcal{D}_G[R,R']$ assuming that $tab_H[S,S']$ represents $\mathcal{D}_H[S,S']$ for all $S, S' \in \{0,1,\dots,d\}^k$.*

Proof. Recall that $lab_G^{-1}(\ell) = lab_H^{-1}(\ell)$ for all $\ell \in [k]$. First, we prove that all tuples added to $\mathcal{A}$ belong to $\mathcal{D}_G[R,R']$, *i.e.*, $\mathcal{A} \subseteq \mathcal{D}_G[R,R']$. Let $(p_0,p,w) \in tab_H[R,S']$ with D, its associated set from Definition 5. Independently from the different cases, we claim that $D \cup V^+(R')$ (σ,ρ)-dominates $V(G)$ in $\mathsf{f}_{R'}(G)$. In order to prove this claim, observe that the neighborhood of a vertex in $V(G) \setminus lab_G^{-1}(\{i,j\})$ is the same in $\mathsf{f}_{S'}(H)$ and in $\mathsf{f}_{R'}(G)$. Thus, it is sufficient to prove that $lab_G^{-1}(\{i,j\})$ is (σ,ρ)-dominated by $D \cup V^+(R')$. It is easy to see that for all $v \in lab_G^{-1}(i)$

$$|N_{\mathsf{f}_{R'}(G)}(v) \cap (D \cup V^+(R'))| = |N_H(v) \cap D| + |lab_G^{-1}(j) \cap D| + R'_i.$$

As $D \cup V^+(S')$ (σ,ρ)-dominates $lab_G^{-1}(i)$ in $\mathsf{f}_{S'}(H)$, we can conclude that, if $\mathcal{P}(i)$ is true, then for all vertices $v \in lab_G^{-1}(i)$ and for all $k \in \mathbb{N}$, $|N_{\mathsf{f}_{S'}(H)}(v) \cap D| + S'_i + k$ belongs respectively to σ if $v \in D$ and to ρ if $v \notin D$. Moreover, if $S'_i = R'_i + R_j$, then, for all $v \in lab_G^{-1}(i)$, $|N_{\mathsf{f}_{S'}(H)}(v) \cap D| + S'_i = |N_{\mathsf{f}_{R'}(G)}(v) \cap D| + R'_i$. In both cases, $lab_G^{-1}(i)$ is (σ,ρ)-dominated by $D \cup V^+(R')$. Symmetrically, we can prove that $lab_G^{-1}(j)$ is also (σ,ρ)-dominated. Thus, we can conclude that $D \cup V^+(R')$ (σ,ρ)-dominates $V(G)$ in $\mathsf{f}_{R'}(G)$.

If $R' = \{0\}^k$ then one can easily check that, by definition of S', the two cases $p_0 = [k] \setminus \{i\}$ or $[k] \setminus \{j\}$ are impossible. Thus, we have either $p_0 = [k]$ or $p_0 = [k] \setminus \{i,j\}$. Since $(p_0,p,w) \in \mathcal{D}_H[S,S']$, if $p_0 = [k]$ then $S' = \{0\}^k$ and $H[D] = G[D]$ is connected. Otherwise, if $p_0 = [k] \setminus \{i,j\}$, every connected component of $H[D]$ contains a vertex in $lab_G^{-1}(\{i,j\})$ and both labels are intersected by D, thus $G[D] = add_{i,j}(H[D])$ is connected. In both cases, $G[D]$ is connected and thus $([k], \emptyset, w)$ is a valid entry of $\mathcal{D}_G[R,R']$.

Assume now that $R' \neq \{0\}^k$. If $R_i = 0$ or $R_j = 0$, then we have $H[D] = G[D]$. Moreover, by the definition of S', it is easy to check that $p_0 = \{\ell \mid R_\ell = 0 \text{ or } R'_\ell = 0\}$. Also, the last property of Definition 5, is satisfied since $lab_G^{-1}(p_0) = lab_H^{-1}(p_0)$. We can thus conclude that (p_0,p,w) is a valid entry of $\mathcal{D}_G[R,R']$.

Now assume that $R_i \neq 0$ and $R_j \neq 0$ and let (p'_0,p',w) be the tuple added to $\mathcal{A}$ from (p_0,p,w). Observe that the join and the proj, we successively apply on $\mathcal{A}'$, guarantee that $p'_0 = p_0 \cup \{\ell \in \{i,j\} \mid R'_\ell = 0\}$ and then that $p'_0 = \{\ell \mid R_\ell = 0 \text{ or } R'_\ell = 0\}$. Notice that p' is the partition obtained from p by merging the blocks containing i and j and by removing $\{\ell \in \{i,j\} \mid R'_\ell = 0\}$ from this new bloc. It is straightforward to check that $p' = ([k] \setminus p'_0)/\sim_{\mathsf{f}_{R'}(G)}$ since $G[D] = add_{i,j}(H[D])$. It remains to prove the last condition of Definition 5. Assume towards a contradiction that $G[D]$ admits a connected component C fully included in $lab_G^{-1}(p'_0)$. As every connected component of $H[D]$ intersects $lab_G^{-1}([k] \setminus p_0)$ and $p'_0 \setminus p_0 \subseteq \{i,j\}$, one can check that $p'_0 = p_0 \cup \{i,j\}$ and that C intersects both $lab_G^{-1}(i)$ and $lab_G^{-1}(j)$. Moreover, C must be the only connected component of $G[D]$ which intersects $lab_G^{-1}(\{i,j\})$, otherwise, C would not be maximal. Hence, either $\{i,j\}$ is a bloc of p or $\{i\},\{j\}$ are blocks of p. In both cases, the proj with $\{\ell \in \{i,j\} \mid R'_\ell = 0\} = \{i,j\}$ as argument will

return the empty set. This contradicts the fact that (p'_0, p', w) is added to $\mathcal{A}$ from (p_0, p, w). We can therefore conclude that all tuples added to $\mathcal{A}$ are in $\mathcal{D}_G[R, R']$.

Now, we prove that $\mathcal{D}_G[R, R'] \subseteq \mathcal{A}$. Let $(p_0, p, w) \in \mathcal{D}_G[R, R']$ and D its associated set from Definition 5. Moreover, let $p'_0 = \{\ell \in [k] \mid R_\ell = 0 \text{ or } S'_\ell = 0\}$ and $p' = ([k] \setminus p'_0)/\sim_{\mathsf{f}_{S'}(H)}$. By definition of d and since $D \cup V^+(R')$ (σ, ρ)-dominates $V(G)$ in $\mathsf{f}_{R'}(G)$, it is easy to see that $R_j + R'_i > d$ implies $\mathcal{P}(i)$ and that $R_i + R'_j > d$ implies $\mathcal{P}(j)$. From this observation, one can check that $D \cup V^+(R')$ (σ, ρ)-dominates $V(H)$ in $\mathsf{f}_{S'}(H)$. The last condition of Definition 5 is satisfied since $lab_G^{-1}(p'_0) \subseteq lab_H^{-1}(p_0)$. We can conclude that $(p'_0, p', w) \in \mathcal{D}_H[R, S']$. We claim that (p_0, p, w) is added to $\mathcal{A}$ from (p'_0, p', w). This is trivial if $R' = \{0\}^k$ or $(R_i = 0) \wedge (R_j = 0)$. Assume from now that $R' \neq \{0\}^k$ and $(R_i \neq 0) \vee (R_j \neq 0)$. We prove that the proj, with $\{\ell \in \{i, j\} \mid R'_\ell = 0\}$ as argument, does not return an empty set. Assume towards, a contradiction, that it is the case. It means that $\{\ell \in \{i, j\} \mid R'_\ell = 0\} = \{i, j\}$ and either $\{i, j\}$ is a bloc of p' or $\{i\}$ and $\{j\}$ are blocks of p'. In both cases, it implies that the connected component containing $D \cap lab_G^{-1}(\{i, j\})$ is fully contained in $lab_G^{-1}(p_0)$. This is a contradiction with the last condition of Definition 5. Hence, (p_0, p, w) is added to $\mathcal{A}$. □

Computing tab_G for $G = G_1 \oplus G_2$. We can suppose that G_1 and G_2 are k-labeled. For $R, R' \in \{0, 1, \ldots, d, = \infty\}^k$, we let $tab_G[R, R'] := \mathsf{reduce}(\mathsf{rmc}(\mathcal{A}))$ where

$$\mathcal{A} := \begin{cases} \mathcal{A}_1 \cup \mathcal{A}_2 & \text{if } R' = \{0\}^k, \\ \bigcup\limits_{(S^1, S^2) \in \mathcal{S}} \mathsf{join}(tab_{G_1}[S^1, R'], tab_{G_2}[S^2, R']) & \text{otherwise,} \end{cases}$$

with

$$\mathcal{A}_1 := \begin{cases} tab_{G_1}[R, \{0\}^k] & \text{if } tab_{G_2}[\{0\}^k, \{0\}^k] \neq \emptyset \\ \emptyset & \text{otherwise.} \end{cases}$$

$$\mathcal{A}_2 := \begin{cases} tab_{G_2}[R, \{0\}^k] & \text{if } tab_{G_1}[\{0\}^k, \{0\}^k] \neq \emptyset \\ \emptyset & \text{otherwise.} \end{cases}$$

and, $\mathcal{S}$ is the set of all the pairs $(S^1, S^2) \in \{0, \ldots, d, +\infty\} \times \{0, \ldots, d, +\infty\}$ such that, for each $\ell \in [k]$, either $R_\ell = \min(S^1_\ell + S^2_\ell, d)$ or $(R_\ell = +\infty$ and, $S^1_\ell = r^\ell_{G_1}(lab^{-1}_{G_1}(\ell))$, $S^2_\ell = r^\ell_{G_2}(lab^{-1}_{G_2}(\ell)))$.

Lemma 5.6. *Let $G = G_1 \oplus G_2$ be a k-labeled graph. For $R, R' \in \{0, 1, \ldots, d\}^k$, the table $tab_G[R, R']$ represents $\mathcal{D}_G[R, R']$ assuming that $tab_{G_1}[S, S']$ and $tab_{G_2}[S, S']$ represents $\mathcal{D}_{G_1}[S, S']$ and $\mathcal{D}_{G_2}[S, S']$, respectively, for all $S, S' \in \{0, 1, \ldots, d, +\infty\}^k \times \{0, 1, \ldots, d\}^k$.*

Proof. Assume first that $R' = \{0\}^k$. Therefore, it is not possible to construct a connected set with two connected sets from G_1 and G_2. It is clear by definition that if (p_0, p, w) is added to $\mathcal{A}$, then there is $D \subseteq V(G)$ so that all the conditions of Definition 5 are satisfied by D. Now, if $([k], \emptyset, w) \in \mathcal{D}_G[R, \{0\}^k]$ and D is its associated set, then either $D \subseteq V(G_1)$ or $D \subseteq V(G_2)$. Suppose *w.l.o.g.* that $D \subseteq V(G_1)$. Since no vertex in G_1 has a neighbor in G_2, for each vertex x in $V(G_2)$, $N_G(x) \cap D = \emptyset$, *i.e.* $tab_{G_2}[\{0\}^k, \{0\}^k] \neq \emptyset$ and by induction we should have $([k], \emptyset, w) \in \mathcal{D}_{G_1}[R, \{0\}^k]$. By definition, (p_0, p, w) is added to $\mathcal{A}$. One can conversely prove that any such set belonging to $\mathcal{D}_G[R, \{0\}^k]$ belongs to $\mathcal{A}_1 \cup \mathcal{A}_2$.

Let us now suppose that $R' \neq \{0\}^k$. Let $(p_0, p, w) \in \mathcal{D}_G[R, R']$ satisfying all the conditions of Definition 5 and let $D \subseteq V(G)$ its associated set. Let $D_i := D \cap V(G_i)$ for $i \in \{1, 2, \}$ and let $S^1 := r_{G_1}(D_1)$ and $S^2 := r_{G_2}(D_2)$. It is easy to check that $(S^1, S^2) \in \mathcal{S}$. Because no vertex in G_1 has a neighbor in G_2, it is clear that $D \cup V^+(R')$ (σ, ρ)-dominates $V(G)$ if and only if $D_i \cup V^+(R')$ (σ, ρ)-dominates

$V(G_i)$ for $i \in \{1,2\}$. Also, since each component of G is either a component of G_1 or a component of G_2, for each component C of G_1 (resp. G_2) there is an j-vertex x in C with $R'_j \neq 0$. For $t \in [2]$, let $p_0^t := \{\ell \in [k] \mid S_j^t = 0 \text{ or } R'_\ell = 0\}$ and $p_t := ([k] \setminus p_0^t)/\sim_{f_{R'}(G_t)}$. Therefore, $(p_0^1, p_1, \mathsf{w}(D_1)) \in \mathcal{D}_{G_1}[S^1, R']$ and $(p_0^2, p_2, \mathsf{w}(D_2)) \in \mathcal{D}_{G_2}[S^2, R']$. By definition, we have $p = (p_1)_{\uparrow[k]\setminus p_0} \sqcup (p_2)_{\uparrow[k]\setminus p_0}$, $p_0 = p_0^1 \cap p_0^2$ and $w = \mathsf{w}(D_1) + \mathsf{w}(D_2)$. Therefore, (p_0, p, w) is added to $\mathcal{A}$.

In the other direction, let $(S^1, S^2) \in \mathcal{S}$ and let (p_0^1, p_1, w_1) and (p_0^2, p_2, w_2) in $tab_{G_1}[S^1, R']$ and $tab_{G_2}[S^2, R']$, respectively, so that

$$\{(p_0, p, w)\} = \mathsf{join}(\{(p_0^1, p_1, w_1)\}, \{(p_0^2, p_2, w_2)\}).$$

By definition, there are $D_1 \subseteq V(G_1)$ and $D_2 \subseteq V(G_2)$ associated with (p_0^1, p_1, w_1) and (p_0^2, p_2, w_2) respectively. Let $D := D_1 \cup D_2$. By definition of $\mathcal{S}$, $r(D_1 \cup D_2) = R$ and $\mathsf{w}(D) = \mathsf{w}(D_1) + \mathsf{w}(D_2) = w$. One also checks that $p_0 = \{\ell \in [k] \mid R_\ell \text{ or } R'_\ell = 0\}$ as $p_0 = p_0^1 \cap p_0^2$. Because $E(G) \cap (V(G_1) \times V(G_2)) = \emptyset$, each component of $G[D]$ is either a component of D_1 or of D_2, *i.e.*, each component of $G[D]$ contains an j-vertex with $R'_j \neq 0$. Also, $D \cup V^+(R')$ (σ, ρ)-dominates $V(G)$ if and only $D_1 \cup V^+(R')$ and $D_2 \cup V^+(R')$ (σ, ρ)-dominate $V(G_1)$ and $V(G_2)$ respectively. One can easily check that $p = (p_1)_{\uparrow[k]\setminus p_0} \sqcup (p_2)_{\uparrow[k]\setminus p_0}$ by definition of $\sqcup$ and $\sim_{f_{R'}(G)}$. We can therefore conclude that adding $(p_0, p, w) \in \mathcal{A}$ is correct. □

Theorem 5.7. *There is an algorithm that, given an n-vertex graph G and a clique-width k-expression of G, computes an optimum connected (σ, ρ)-dominating set in time $(d+2)^{2k} \cdot (d+1)^k \cdot 2^{(1+\omega)k} \cdot n \cdot k^{O(1)}$.*

Proof. We do a bottom-up traversal of the clique-width expression and at each step we update the tables as indicated above. The correctness of the algorithm follows from Lemmas 5.4-5.6. Let us discuss the time complexity now. First, notice that there are at most $(d+2)^k \cdot (d+1)^k$ pairs (R, R') so that $tab_G[R, R']$ is well-defined. So, it is enough to show that each table $tab_G[R, R']$ is computed in at most $(d+2)^k \cdot 2^{(1+\omega)k}$. If $G = add_{i,j}(H)$, we update $tab_G[R, R']$ from one entry $tab_H[R, S']$ for some fixed S' computable in $O(k)$ time. Since the used join operation runs in time $2^{k-1} \cdot k^{O(1)}$ and we use reduce on a set of size 2^{k-1}, we are done. Now, if $G = ren_{i \to j}(H)$, then we update $tab_G[R, R']$ from $|\mathcal{S}| = (d+2)^2$ tables from tab_H, each identified in $O(k)$ time from (R, R'). Since each table contains at most 2^{k-1} entries, we can thus update $tab_G[R, R']$ in time $(d+2)^2 \cdot 2^{(\omega-1)k} \cdot k^{O(1)}$. If $G = G_1 \oplus G_2$, the bottleneck is when $R' \neq \{0\}^k$, in this case, we compute $tab_G[R, R']$ from the union of $|\mathcal{S}|$ join. Observe that $|\mathcal{S}| \leq (d+2)^k$ since for R fixed and $S^1 \in \{0, \ldots, d, +\infty\}$, there is at most one $S^2 \in \{0, \ldots, d, +\infty\}$ such that $(S^1, S^2) \in \mathcal{S}$. Each join is computed in $2^{2k-2} \cdot k^{O(1)}$ time and adds at most 2^{k-2} element in $\mathcal{A}$. Therefore, $tab_G[R, R']$ is computed in time $(d+2)^k \cdot 2^{(\omega+1)\cdot k}$.

Because the size of a clique-width expression is linear in n, we can conclude that an optimum (σ, ρ)-dominating set can be computed in the given time. □

As a corollary, we can prove the following.

Theorem 5.8. *There is an algorithm that, given an n-vertex graph G, a subset $K \subseteq V(G)$ and a clique-width k-expression of G, computes a minimum node-weighted steiner tree for (G, K) in time $3^{2k} \cdot 2^{(2+\omega)k} \cdot n \cdot k^{O(1)}$.*

Proof. We can reduce the problem Node-weighted Steiner Tree to a variant of Connected-(σ, ρ)-Dominating Set where $\sigma = \mathbb{N}^+$ and $\rho = \mathbb{N}$. This variant require K to be included in the (σ, ρ)-dominating set. We can add this constraint, by modifying how we compute the table tab_G, when $G = \mathbf{1}(x)$ and $x \in K$. For

$R \in \{0,1,+\infty\}^k, R' \in \{0,1\}^k$, we set

$$tab_G[R,R'] := \begin{cases} \emptyset & \text{if } R_1 \neq 1 \text{ or } R'_1 \neq 1 \text{ or } R_i \neq 0 \text{ for } i \neq 1\ , \\ \{\,(\,[k] \setminus \{1\},\ \{\{1\}\},\ \mathsf{w}(x)\,)\,\} & \text{otherwise.} \end{cases}$$

It is straightforward to check that this modification implements this constraint and our algorithm with this modification computes a minimum node-weighted Steiner Tree. □

For the case of out-connected (σ,ρ)-dominating sets, let us define the following table entries.

Definition 6. For a k-labeled graph G and $(R,R') \in \{0,1,\ldots,d,+\infty\}^k \times \{0,1,\ldots,d\}$, let $p_0 = \{\ell \in [k] \mid R_\ell = r^\ell_G(lab_G^{-1}(\ell)) \text{ or } R'_\ell = 0\}$,

$$\begin{aligned} \overline{\mathcal{D}}_G[R,R'] := \big\{ & (p_0,p,w) \in \Pi([k] \setminus p_0, p_0) \mid \exists D \subseteq V(G), r_G(D) = R, \mathsf{w}(D) = w, \\ & D \cup V^+(R')\ (\sigma,\rho)\text{-dominates } V(G) \text{ in } \mathsf{f}_{R'}(G), \\ & p = ([k] \setminus p_0) \sim_{f_{R'}(G)}, \text{ and } G[V(G) \setminus D] \text{ is connected if } R' = \{0\}^k, \\ & \text{otherwise, } \forall C \in CC(G[V(G) \setminus D]), \exists x \in C \text{ with } lab_G(x) = j \text{ and } R'_j \neq 0\big\}. \end{aligned}$$

Where $\sim_{f_{R'}(G)}$ is the same equivalence relation as the one defined in Definition 5.

Observe that $R_\ell = r^\ell_G(lab_G^{-1}(\ell))$ implies $r^\ell_G(V(G) \setminus D) = 0$. It is straightforward to check that the optimum value is the optimum over all $R \in \{0,1,\ldots,d,+\infty\}^k$ of $opt\{w \mid ([k],\emptyset,w) \in \overline{\mathcal{D}}_G[R,\{0\}^k]\}$ for a k-labeled graph G. Definition 6 is similar to Definition 5, except that p_0 and p records the label classes intersected by the complement of the (σ,ρ)-dominating set. It is now an exercise to modify the algorithm for connected (σ,ρ)-dominating set so that we compute tables for $\overline{\mathcal{D}}_G$.

6. Concluding Remarks

We combine the techniques introduced in [3] and the rank-based approach from [1] to obtain $2^{O(k)} \cdot n^{O(1)}$ time algorithms for several connectivity constraints problems such as Connected Dominating Set, Connected Vertex Cover, Feedback Vertex Set, Connected Induced d-Regular Subgraph, etc. While we did not consider connectivity constraints on locally vertex partitioning problems [3], it seems clear that we can adapt the algorithms from the paper to consider connectivity constraints such as, if the solution is $\{D_1,\ldots,D_q\}$, each block D_i is connected or a proper subset of the blocks form a connected graph. We did not consider counting versions and it would be interesting to know whether we can adapt the approach in [1] based on the determinant to the clique-width.

The main drawback of clique-width is that, for fixed k, there is no known FPT polynomial time algorithm that produces a clique-width expression that even approximates within a constant factor the clique-width. We can avoid this major open question, by using the equivalent notion of *rank-width* and its associated *rank-decomposition* for which an FPT cubic time algorithm is known [12, 14, 16]. But, if we use our approach with a rank-decomposition, the number of labels is bounded by 2^k, and so we will get a doubly exponential time algorithm. We can circle it and obtain $2^{O(k^2)} \cdot n^{O(1)}$ as done for example in [10] for Feedback Vertex Set by using Myhill-Nerode congruences, and it is not yet clear how we can combine this with rank-based approach. Even though we overcome this difficulty, the case of connected locally checkable properties will not be settled as the best upper-bound on the number of neighbor-equivalences for rank-width is $2^{O(k \cdot \log(k))}$ [15] provided through the notion of $\mathbb{Q}$-rank-width [13, 15].

References

[1] Hans L. Bodlaender, Marek Cygan, Stefan Kratsch, and Jesper Nederlof. Deterministic single exponential time algorithms for connectivity problems parameterized by treewidth. *Inform. and Comput.*, 243:86–111, 2015.

[2] Binh-Minh Bui-Xuan, Ondřej Suchý, Jan Arne Telle, and Martin Vatshelle. Feedback vertex set on graphs of low clique-width. *European J. Combin.*, 34(3):666–679, 2013.

[3] Binh-Minh Bui-Xuan, Jan Arne Telle, and Martin Vatshelle. Fast dynamic programming for locally checkable vertex subset and vertex partitioning problems. *Theoret. Comput. Sci.*, 511:66–76, 2013.

[4] Bruno Courcelle and Joost Engelfriet. *Graph structure and monadic second-order logic*, volume 138 of *Encyclopedia of Mathematics and its Applications*. Cambridge University Press, Cambridge, 2012. A language-theoretic approach, With a foreword by Maurice Nivat.

[5] Bruno Courcelle, Joost Engelfriet, and Grzegorz Rozenberg. Handle-rewriting hypergraph grammars. *Journal of Computer and System Sciences*, 46(2):218–270, 1993.

[6] Bruno Courcelle and Stephan Olariu. Upper bounds to the clique width of graphs. *Discrete Applied Mathematics*, 101(1-3):77–114, 2000.

[7] Reinhard Diestel. *Graph Theory*. Number 173 in Graduate Texts in Mathematics. Springer, third edition, 2005.

[8] Rodney G. Downey and Michael R. Fellows. *Fundamentals of parameterized complexity*. Texts in Computer Science. Springer, London, 2013.

[9] Fedor V. Fomin, Petr A. Golovach, Daniel Lokshtanov, and Saket Saurabh. Almost optimal lower bounds for problems parameterized by clique-width. *SIAM J. Comput.*, 43(5):1541–1563, 2014.

[10] Robert Ganian and Petr Hliněný. On parse trees and Myhill-Nerode-type tools for handling graphs of bounded rank-width. *Discrete Appl. Math.*, 158(7):851–867, 2010.

[11] Robert Ganian, Petr Hliněný, and Jan Obdržálek. Clique-width: when hard does not mean impossible. In *28th International Symposium on Theoretical Aspects of Computer Science*, volume 9 of *LIPIcs. Leibniz Int. Proc. Inform.*, pages 404–415. Schloss Dagstuhl. Leibniz-Zent. Inform., Wadern, 2011.

[12] Petr Hliněný and Sang-il Oum. Finding branch-decompositions and rank-decompositions. *SIAM J. Comput.*, 38(3):1012–1032, 2008.

[13] Mamadou Moustapha Kanté and Michael Rao. The rank-width of edge-coloured graphs. *Theory Comput. Syst.*, 52(4):599–644, 2013.

[14] Sang-Il Oum. Approximating rank-width and clique-width quickly. *ACM Trans. Algorithms*, 5(1):Art. 10, 20, 2009.

[15] Sang-il Oum, Sigve Hortemo Sæther, and Martin Vatshelle. Faster algorithms for vertex partitioning problems parameterized by clique-width. *Theoret. Comput. Sci.*, 535:16–24, 2014.

[16] Sang-il Oum and Paul Seymour. Approximating clique-width and branch-width. *J. Combin. Theory Ser. B*, 96(4):514–528, 2006.

[17] Neil Robertson and Paul D. Seymour. Graph minors. II. algorithmic aspects of tree-width. *Journal of Algorithms*, 7(3):309–322, 1986.

[18] Jan Arne Telle and Andrzej Proskurowski. Algorithms for vertex partitioning problems on partial k-trees. *SIAM J. Discrete Math.*, 10(4):529–550, 1997.

Appendix A. Proof of Lemma 3.2

We start by giving some useful facts. The following is quite easy to observe if we take the graphical definition of $\mathsf{acyclic}$.

Fact 1. For all partitions $p, q, r \in \Pi(V)$

$$\mathsf{acyclic}(p, q) \wedge \mathsf{acyclic}(p \sqcup q, r) \Leftrightarrow \mathsf{acyclic}(q, r) \wedge \mathsf{acyclic}(p, q \sqcup r).$$

Proof. For a partition $p \in \Pi(V)$, let $f(p) := |V| - \sharp\mathrm{block}(p)$. One easily checks that $\mathsf{acyclic}(p, q)$ holds if and only if $f(p \sqcup q)$ equals $f(p) + f(q)$. One can therefore deduce, by an easy calculation from this equivalence, that $\mathsf{acyclic}(p \sqcup q, r) \wedge \mathsf{acyclic}(p, q)$ is equivalent to saying that $f(p \sqcup q \sqcup r)$ equals $f(p) + f(p) + f(r)$. Similarly, for $\mathsf{acyclic}(q, r) \wedge \mathsf{acyclic}(p, q \sqcup r)$. □

By definition of $\mathsf{acyclic}$ and of $\sqcup$, we can also observe the following.

Fact 2. Let $q \in \Pi(V)$ and let $X \subseteq V$ such that no subset of X is a block of q. Then, for each $p \in \Pi(V \setminus X)$, we can observe the following equivalences

$$p_{\uparrow X} \sqcup q = \{V\} \iff p \sqcup q_{\downarrow(V\setminus X)} = \{V \setminus X\} \qquad \text{and} \tag{1}$$

$$\mathsf{acyclic}(p_{\uparrow X}, q) \iff \mathsf{acyclic}(p, q_{\downarrow(V\setminus X)}). \tag{2}$$

The following is quite easy to prove since, for all $\mathcal{A} \subseteq \mathbb{H}(V,S) \times \mathbb{N}$ and $(q_0, q) \in \mathbb{H}(V,S)$, by definition of **ac-opt**, we have

$$\textbf{ac-opt}(\mathcal{A}, (q_0, q)) = \textbf{ac-opt}(\mathcal{A}_{\uparrow q_0}, (q_0, q)).$$

Fact 3. Let $\mathcal{A}$ and $\mathcal{A}'$ be subsets of $\mathbb{H}(V,S) \times \mathbb{N}$. $\mathcal{A}'$ ac-represents $\mathcal{A}$ if and only if ${\mathcal{A}'}_{\uparrow X}$ ac-represents $\mathcal{A}_{\uparrow X}$ for all $X \subseteq S$.

Proof of Lemma 3.2. Let V and S be two disjoint finite sets and let $\mathcal{A}$ and $\mathcal{A}'$ be weighted partitions in $\mathbb{H}(V,S) \times \mathbb{N}$.

Remove non-maximal copies. Let $(q_0, q) \in \mathbb{H}(V,S)$. By the definition of rmc, whenever $(p_0, p, w) \in \mathcal{A}$ is such that $p_0 = q_0$, $p \sqcup q = \{V\}$, $\mathsf{acyclic}(p, q)$ and $\textbf{ac-opt}(\mathcal{A}, (q_0, q)) = w$, then $(p_0, p, w) \in \mathsf{rmc}(\mathcal{A})$, otherwise there would exist $(p_0, p, w') \in \mathcal{A}$ with $w' > w$ which would contradict $w = \textbf{ac-opt}(\mathcal{A}, (q_0, q))$. Therefore, $\textbf{ac-opt}(\mathsf{rmc}(\mathcal{A}), (q_0, q)) = \textbf{ac-opt}(\mathcal{A}, (q_0, q))$. We can then conclude that if $\mathcal{A}'$ ac-represents $\mathcal{A}$, it holds that $\mathsf{rmc}(\mathcal{A}')$ ac-represents $\mathsf{rmc}(\mathcal{A})$.

Project. Because $\mathsf{proj}(\mathcal{A}, X) = \mathsf{proj}(\mathsf{proj}(\mathcal{A}, x), X \setminus \{x\})$ for all $x \in X \subseteq V \cup S$, we can assume that $X = \{x\}$. Let $(q_0, q) \in \mathbb{H}(V \setminus x, S \setminus x)$. Let $(p_0, p, w) \in \mathcal{A}$ such that $(p_0 \setminus x, p_{\downarrow V\setminus x}, w) \in \mathsf{proj}(\{(p_0, p, w)\}, X)$ with $p_0 \setminus x = q_0$. Since $\mathsf{proj}(\{(p_0, p, w)\}, X) \neq \emptyset$, $\{x\}$ is not a block of p and by Fact 2, if $x \in V$ then

$$p_{\downarrow V\setminus x} \sqcup q = \{V \setminus x\} \iff p \sqcup q_{\uparrow x} = \{V\}, \qquad \text{and}$$
$$\mathsf{acyclic}(p_{\downarrow V\setminus x}, q) \iff \mathsf{acyclic}(p, q_{\uparrow x}).$$

Thus, if $x \in V$ then $\textbf{ac-opt}(\mathsf{proj}(\mathcal{A}, \{x\}), (q_0, q)) = \textbf{ac-opt}(\mathcal{A}, (q_0, q_{\uparrow x}))$. Otherwise, if $x \in S$ then $p_{\downarrow V\setminus x} = p$ and either $p_0 = q_0$ or $p_0 = q_0 \cup x$. In this case, $\textbf{ac-opt}(\mathsf{proj}(\mathcal{A}, \{x\}), (q_0, q))$ is the maximum between $\textbf{ac-opt}(\mathcal{A}, (q_0 \cup x, q))$ and $\textbf{ac-opt}(\mathcal{A}, (q_0, q))$. In both case, we can then conclude that if $\mathcal{A}'$ ac-represents $\mathcal{A}$, it holds that $\mathsf{proj}(\mathcal{A}', \{x\})$ ac-represents $\mathsf{proj}(\mathcal{A}, \{x\})$.

Ac-Join. Let V' and S' be two disjoint finite sets and let $\mathcal{B} \subseteq \mathbb{H}(V', S') \times \mathbb{N}$. By Fact 3, it is sufficient to prove that, for all $X \subseteq (S \setminus V') \cup (S' \setminus V)$, we have $\mathsf{acjoin}(\mathcal{A}', \mathcal{B})_{\uparrow X}$ ac-represents $\mathsf{acjoin}(\mathcal{A}, \mathcal{B})_{\uparrow X}$. Let $X \subseteq (S \setminus V') \cup (S' \setminus V)$ and $(r_0, r) \in \pi(V \cup V', (S \setminus V') \cup (S' \setminus V))$. If $r_0 \neq X$ then $\textbf{ac-opt}(\mathsf{acjoin}(\mathcal{A}, \mathcal{B})_{\uparrow X}, (r_0, r)) = \textbf{ac-opt}(\mathsf{acjoin}(\mathcal{A}', \mathcal{B})_{\uparrow X}, (r_0, r)) = -\infty$.

Assume now that $r_0 = X$, by definition, $\textbf{ac-opt}(\mathsf{acjoin}(\mathcal{A}, \mathcal{B})_{\uparrow X}, (X, r))$ equals

$$\begin{aligned}\max\{w_1 + w_2 \mid (p_0, p, w_1) \in \mathcal{A} \wedge (q_0, q, w_2) \in \mathcal{B} \wedge X = (p_0 \setminus V') \cup (q_0 \setminus V) \cup (p_0 \cap q_0)\\ \wedge\, \mathsf{acyclic}(p_{\uparrow V'}, q_{\uparrow V}) \wedge \mathsf{acyclic}(p_{\uparrow V'} \sqcup q_{\uparrow V}, r)\\ \wedge\, p_{\uparrow V'} \sqcup q_{\uparrow V} \sqcup r = \{V \cup V'\}\}.\end{aligned}$$

By fact 1, $\textbf{ac-opt}(\mathsf{acjoin}(\mathcal{A}, \mathcal{B})_{\uparrow X}, (X, r))$ equals

$$\begin{aligned}\max\{w_1 + w_2 \mid (p_0, p, w_1) \in \mathcal{A} \wedge (q_0, q, w_2) \in \mathcal{B} \wedge X = (p_0 \setminus V') \cup (q_0 \setminus V) \cup (p_0 \cap q_0)\\ \wedge\, \mathsf{acyclic}(q_{\uparrow V}, r) \wedge \mathsf{acyclic}(p_{\uparrow V'}, q_{\uparrow V} \sqcup r)\\ \wedge\, p_{\uparrow V'} \sqcup q_{\uparrow V} \sqcup r = \{V \cup V'\}\}.\end{aligned}$$

If for some $(q_0, q, w_2) \in \mathcal{B}$ there exists a subset Y of $V' \setminus V$ such that Y is a block of $q_{\uparrow V} \sqcup r$ then there is no $(p_0, p, w_1) \in \mathbb{H}(V,S)$ such that $p_{\uparrow V'} \sqcup q_{\uparrow V} \sqcup r = \{V \cup V'\}$,

i.e., $\mathbf{ac\text{-}opt}(\mathsf{acjoin}(\mathrm{H}(V,S),\{(q_0,q,w_2)\}),(r_0,r))_{\restriction X} = -\infty$. Thus, we can suppose that for all $(q_0,q,w_2) \in \mathcal{B}$ there is no subset Y of $V' \setminus V$ such that Y is a block of $q_{\uparrow V} \sqcup r$ and we can apply the Fact 2, *i.e.*, $\mathbf{ac\text{-}opt}(\mathsf{acjoin}(\mathcal{A},\mathcal{B})_{\restriction X},(X,r))$ equals

$$\begin{aligned}\max\{w_1+w_2 \mid (p_0,p,w_1) \in \mathcal{A} \wedge (q_0,q,w_2) \in \mathcal{B} \wedge X = (p_0 \setminus V') \cup (q_0 \setminus V) \cup (p_0 \cap q_0)\\ \wedge \mathsf{acyclic}(q_{\uparrow V}, r) \wedge \mathsf{acyclic}(p,(q_{\uparrow V} \sqcup r)_{\downarrow V})\\ \wedge p \sqcup (q_{\uparrow V} \sqcup r)_{\downarrow V} = \{V\}\}.\end{aligned}$$

This is equal to

$$\begin{aligned}\max\{\mathbf{ac\text{-}opt}(\mathcal{A},(Y,(q_{\uparrow V} \sqcup r)_{\downarrow V})) + w_2 \mid Y \subseteq S \wedge (q_0,q,w_2) \in \mathcal{B}\\ \wedge X = (Y \setminus V') \cup (q_0 \setminus V) \cup (Y \cap q_0)\\ \wedge \mathsf{acyclic}(q_{\uparrow V}, r)\}.\end{aligned}$$

If $\mathcal{A}'$ ac-represents $\mathcal{A}$ then $\mathbf{ac\text{-}opt}(\mathcal{A},(Y,(q_{\uparrow V} \sqcup r)_{\downarrow V})) = \mathbf{ac\text{-}opt}(\mathcal{A}',(Y,(q_{\uparrow V} \sqcup r)_{\downarrow V}))$ for all $Y \subseteq S$ and $(q_0,q,w_2) \in \mathcal{B}$. Thus, we can conclude that

$$\mathbf{ac\text{-}opt}(\mathsf{acjoin}(\mathcal{A},\mathcal{B})_{\restriction X},(X,r)) = \mathbf{ac\text{-}opt}(\mathsf{acjoin}(\mathcal{A}',\mathcal{B})_{\restriction X},(X,r))$$

and then $\mathsf{acjoin}(\mathcal{A}',\mathcal{B})$ ac-represents $\mathsf{acjoin}(\mathcal{A},\mathcal{B})$ whenever $\mathcal{A}'$ ac-represents $\mathcal{A}$.

Union. It is quite trivial for the operator $\cup$ since, for all $\mathcal{B} \subseteq \mathrm{H}(V,S) \times \mathbb{N}$ and $(q_0,q) \in \mathrm{H}(V,S) \times \mathbb{N}$, we have $\mathbf{ac\text{-}opt}(\mathcal{A} \cup \mathcal{B},(q_0,q)) = \max(\mathbf{ac\text{-}opt}(\mathcal{A},(q_0,q)),\mathbf{ac\text{-}opt}(\mathcal{B},(q_0,q)))$. □

Université Clermont Auvergne, LIMOS, CNRS, Aubière, France.
E-mail address: {mamadou.kante,benjamin.bergougnoux}@uca.fr