A polynomial time algorithm to compute the connected treewidth of a series–parallel graph

Guillaume Mescoff, Christophe Paul, Dimitrios M. Thilikos

A polynomial time algorithm to compute the connected treewidth of a series-parallel graph*

Guillaume Mescoff† Christophe Paul‡ Dimitrios M. Thilikos†

Abstract

It is well known that the *treewidth* of a graph G corresponds to the *node search number* where a team of searchers is pursuing a fugitive that is *lazy* and *invisible* (or alternatively is *agile* and *visible*) and has the ability to move with infinite speed via unguarded paths. Recently, *monotone and connected node search strategies* have been considered. A search strategy is *monotone* if it prevents the fugitive from pervading again areas from where he had been expelled and is *connected* if, at each step, the set of vertices that is or *has been* occupied by the searchers induces a connected subgraph of G. It has been shown that the corresponding connected and monotone search number of a graph G can be expressed as the *connected treewidth*, denoted $\mathbf{ctw}(G)$, that is defined as the minimum width of a rooted tree-decomposition $(\mathcal{X}, T, r)$, where the union of the bags corresponding to the nodes of a path of T containing the root r is connected in G. In this paper, we initiate the algorithmic study of connected treewidth. We design a $O(n^2 \cdot \log n)$-time dynamic programming algorithm to compute the connected treewidth of biconnected series-parallel graphs. At the price of an extra n factor in the running time, our algorithm generalizes to graphs of treewidth at most two.

Keywords: Graph decompositions, Graph Classes, Width parameters, Combinatorial algorithms, Series-parallel graphs, Treewidth, Connected treewidth, Dynamic programming.

1 Introduction

Since its introduction [8, 24, 35], the concept of *treewidth* of a graph has led to major advancements in graph theory. The treewidth parameter, denoted by tw, is central to the design of efficient graph algorithms (see [2] for a survey on early works in this direction and [10] for a recent survey). According to the theorem of Courcelle [13], properties expressible in MSO_2 logic can be tested in parameterized linear time on graph of bounded treewidth. This result established treewidth as an important structural parameter in the context of parameterized complexity [18]. Treewidth can be defined in several equivalent ways, while the standard definition is by means of a tree-decomposition (see Section 2). During the last two decades, a number of width-parameters have been introduced as combinatorial variants or alternatives to treewidth (see [26] for a survey on width parameters). This paper deals with the *connected*

*Research supported by ANR projects DEMOGRAPH (ANR-16-CE40-0028) and ESIGMA (ANR-17-CE23-0010) and the French-German Collaboration ANR/DFG Project UTMA ANR-20-CE92-0027.

†École Normale Supérieure de Rennes (ENS Rennes).

‡LIRMM, Univ Montpellier, CNRS, Montpellier, France.

treewidth parameter, a new variant of treewidth that is motivated by the study of connected search games in graphs.

A *node search* game is a game opposing a group of searchers and a fugitive, occupying the vertices of a graph. In a search strategy, one searcher may either be placed to or removed from a vertex and a winning search strategy is a sequence of moves of the searchers that eventually leads to the capture of the fugitive. The capture of the fugitive happens when a searcher lands on the vertex occupied by the fugitive while the fugitive cannot escape along a searcher-free path. The cost of a search strategy is the maximum number of searchers simultaneously occupying vertices of the graph. The *clean territory* during some step of the game is the set of vertices from which the fugitive has been, so far, expelled by the searchers' strategy. A strategy is *monotone* if it guarantees that the fugitive will not be able to visit an already cleaned territory. Also, a strategy is *connected* if it guarantees, that at any step, the clean territory induces a connected graph. The fugitive can be *lazy* or *agile*. Being lazy means that the fugitive is staying at his position, as long as a searcher is not moving on that position. An agile fugitive may move at any time regardless of the move of the searchers. Also a fugitive can be *visible* or *invisible* in the sense that the searcher's strategy *may* or *may not* take into account the current position of the fugitive. The node search number of a graph (for some of the above variants) is the minimum cost of a winning search strategy.

It is well-known that the *search number against an invisible and lazy fugitive* is equal to treewidth plus one, while the same quantity is also equal to the *search number against a visible and agile fugitive* [14, 36]. On the other hand, if the fugitive is *invisible and agile*, then the corresponding search number is equal to the parameter of pathwidth plus one [30, 32, 29, 28]. Moreover all the aforementioned versions of the game are *monotone* in the sense that, for every search strategy, there is a monotone one with the same cost [9, 36]. In this paper, we deal only with monotone version node search.

Interestingly, the motivating story of one of the seminal papers [33] on graph searching was inspired by an earlier article of the Breisch in Southwestern Cavers Journal [12]. This paper concerned speleotopological explorations, which are, by essence, connected explorations as the searchers cannot "teleport" to a vertex that is away from the current clean territory. The connectivity constraint was considered for the first time in [5, 4], where the price of connectivity, defined as the ratio between monotone connected node search number (for some of the considered strategy variants) and the node search number, was first considered. In [22, 23], it was proven that the price of connectivity for monotone node search against a visible and agile fugitive (or, equivalently, an invisible and lazy one) is $\Theta(\log n)$. Dereniowski [15] introduced the notion of *connected pathwidth* of a graph, denoted $\mathsf{cpw}(G)$ and showed its equivalence with the monotone node search number against an invisible and agile fugitive. He proved that, for every graph G, $\mathsf{cpw}(G) \leqslant 2 \cdot \mathsf{pw}(G) + O(1)$, henceforth resolving the question of the price of connectivity for a monotone node search against an invisible and agile fugitive. Extending the work of [15], Adler et al. [1] recently introduced a definition of *connected treewidth*, denoted hereafter $\mathsf{ctw}(G)$. They proved that, as for treewidth, the connected treewidth parameter is equivalent to the connected monotone node search number against an invisible and lazy fugitive and, as in the non-connected setting, the same holds for the visible and agile case. Also, they proved that connected treewidth is equivalent to a connected variant of the tree vertex separation number (see Section 2).

In this paper, we are interested in the problem of computing the connected treewidth of a graph. So far, very little is known on the algorithmic aspects of the connected treewidth

and connected pathwidth parameters. First of all, checking, given a graph G and an integer k, whether the connected treewidth (or the connected pathwidth) is at most k is an NP-complete problem (see Theorem 4). This means that, in general, we may not expect any polynomial algorithm for computing connected treewidth. Very recently, the problem of deciding $\mathsf{cpw}(G) \leq k$ was shown to be fixed parameterized tractable [27], improving an $n^{O(k^2)}$-time algorithm [16]. An explanation for this lack of algorithmic results partially relies on the fact that, unlike pathwidth and treewidth, connected treewidth and connected pathwidth parameters do not enjoy nice combinatorial properties such as closeness under taking of minors. Interestingly, both parameters are closed under contractions [1]. However, for graph contractions, there is no analogue of the algorithmic machinery developed in the context of graph minors (see Section 4 for a more developed discussion).

We stress that connected treewidth may be arbitrarily bigger than treewidth. For instance, consider a complete binary tree of height k, introduce a new vertex and make it adjacent to all the leaves of the tree. As observed in [1], this graph has connected treewidth k while it is a series-parallel graph that has treewidth 2.

The above motivated us to initiate the study of computing the connected treewidth on the class of series-parallel graphs. First introduced by Macmahon in 1982 [31] (see also [34]), series-parallel graphs are essentially graphs of treewidth at most two [19]. More precisely, a graph has treewidth at most two if and only if each of its biconnected components induces a series-parallel graph. The recursive construction, by means of series and parallel composition (see Section 2), of series-parallel graphs allows to solve a large number of NP-hard problems in polynomial (or even linear) time, see for example [6, 7, 25]. It follows that the class of series-parallel graphs, among others, forms a natural test bed for the existence of efficient graph algorithms [11].

In this paper, we design a $O(n^2 \cdot \log n)$-time algorithm to compute the connected treewidth of a biconnected series-parallel graph. The algorithm is extended to a $O(n^3 \cdot \log n)$-algorithm for graphs of treewidth at most 2. This result constitutes a first step toward the computation of connected treewidth parameterized by treewidth (see Section 4 for a discussion).

2 Preliminaries

We use standard notations for graphs, as for example in [17]. We consider undirected and simple graphs. We let $G = (V, E)$ denote a graph on n vertices, with $V = V(G)$ its vertex set and $E = E(G)$ its edge set. A vertex x is a *neighbor* of y if xy is an edge of E. If S is a subset of V, then $G[S]$ is the *subgraph of G induced by S*. A path P between vertex x and y is called an *(x, y)-path* and the vertices of P distinct from x and y are the *internal vertices* of P. A vertex $x \in V$ is a *cut vertex* if the removal of x strictly increases the number of connected components of the graph. A graph is *biconnected* if it is connected and does not contain a cut vertex. Given an integer q, we use $[q]$ as a shortcut for the set $\{1, \ldots, q\}$.

A *layout* σ of a graph $G = (V, E)$ is a total ordering of its vertices, in other words σ is a bijection from V to $[n]$. For two vertices x and y, we write $x <_\sigma y$ if $\sigma(x) < \sigma(y)$. We define $\sigma_{<i} = \{x \in V \mid \sigma(x) < i\}$ (the sets $\sigma_{>i}$, $\sigma_{\leq i}$ and $\sigma_{\geq i}$ are similarly defined). If S is a subset of V, then $\sigma[S]$ is the layout of $G[S]$ such that for every $x, y \in S$, $\sigma(x) < \sigma(y)$ if and only if $\sigma[S](x) < \sigma[S](y)$. Let σ_1 and σ_2 be two layouts on disjoint vertex sets V_1 and V_2. Then $\sigma = \sigma_1 \odot \sigma_2$, the *concatenation* of σ_1 and σ_2, is the layout on $V_1 \cup V_2$ such that for every $x_1 \in V_1$ and every $x_2 \in V_2$, $\sigma(x_1) < \sigma(x_2)$, $\sigma[V_1] = \sigma_1$ and $\sigma[V_2] = \sigma_2$.

2.1 Series-parallel graphs

A 2-*terminal graph* is a pair $\mathbf{G} = (G, (x, y))$ where $G = (V, E)$ is a graph and (x, y) is a pair of distinguished vertices, hereafter called the *terminals*. Consider two 2-terminal graphs $(G_1, (x_1, y_1))$ and $(G_2, (x_2, y_2))$, where $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$. Then:

- the *series-composition*, denoted $(G_1, (x_1, y_1)) \otimes (G_2, (x_2, y_2))$, yields the 2-terminal graph $(G, (x_1, y_2))$ with G being the graph obtained by identifying the terminal y_1 with x_2;
- the *parallel-composition*, denoted $(G_1, (x_1, y_1)) \oplus (G_2, (x_2, y_2))$, yields the 2-terminal graph $(G, (x_1, y_1))$ with G being the graph obtained by identifying the terminal x_1 with x_2 and the terminal y_1 with y_2.

Definition 1. *A 2-terminal graph $(G, (x, y))$ is* series-parallel *if either it is the single edge graph with $\{x, y\}$ as vertex set, or if it results from the series or the parallel composition of two 2-terminal series-parallel graphs. A graph $G = (V, E)$ is a* series-parallel graph *if for some pair of vertices $x, y \in V$, $(G, (x, y))$ is a 2-terminal series-parallel graph.*

Observe that from the definition above, we may generate multi-graphs. However, in this paper we only consider simple graphs. Our results extend easily to multi-graphs: observe that if G^* is the graph obtained from G after suppressing multiple edges to simple ones, then $\mathsf{ctw}(G) = \mathsf{ctw}(G^*)$. When it is clear from the context, $\mathbf{G}$ will denote the 2-terminal series-parallel graph $(G, (x, y))$. Observe that a series-parallel graph $\mathbf{G}$ can be represented by a so-called *SP-tree* $T(\mathbf{G})$. The leaves of $T(\mathbf{G})$ are labelled by the edges of G. Every internal node of $T(\mathbf{G})$ is labelled by a composition operation ($\oplus$ or $\otimes$) and a pair of terminal vertices. For an internal node t of $T(\mathbf{G})$, we let T_t denote the subtree of $T(\mathbf{G})$ rooted at t and V_t the subset of vertices incident to an edge labelling a leaf of T_t. Suppose that t is labelled $(\otimes, (x_t, y_t))$, then the node t represents the 2-terminal graph $(G[V_t], (x_t, y_t))$.

Theorem 1. *[20] If a graph $G = (V, E)$ is a biconnected series-parallel graph and $xy \in E$, then $(G, (x, y))$ is a 2-terminal series-parallel graph.*

Theorem 1 can be rephrased as follows: if xy is an edge of a biconnected series-parallel graph, then $\mathbf{G} = (G, (x, y))$ is a 2-terminal series-parallel graph such that $\mathbf{G} = (G_1, (x, y)) \oplus (G_2, (x, y))$ where $G_1 = (\{x, y\}, \{xy\})$ and $G_2 = (V, E \setminus \{xy\})$.

Theorem 2. *[37] The problem of testing whether a graph G is series-parallel can be solved in linear time. Moreover if G is a biconnected series-parallel graph, then a SP-tree of $(G, (x, y))$, where xy is an edge, can be build in linear time.*

2.2 Connected tree-decomposition and connected layouts

A *tree-decomposition* of a graph $G = (V, E)$ is pair $(T, \mathcal{F})$ where $T = (V_T, E_T)$ is a tree and $\mathcal{F} = \{X_t \subseteq V \mid t \in V_T\}$ such that

1. $\bigcup_{t \in V_T} X_t = V$;
2. for every edge $xy \in E$, there exists a node $t \in V_T$ such that $\{x, y\} \subseteq X_t$;
3. for every vertex $x \in V$, the set $\{t \in V_T \mid x \in X_t\}$ induces a connected subgraph of T.

We refer to V_T as the set of nodes of T and the sets of $\mathcal{F}$ as the bags of $(T, \mathcal{F})$. The width of a tree-decomposition $(T, \mathcal{F})$ is $\mathsf{width}(T, \mathcal{F}) = \max\{|X| - 1 \mid X \in \mathcal{F}\}$ and the treewidth of a graph G is

$$\mathsf{tw}(G) = \min\{\mathsf{width}(T, \mathcal{F}) \mid (T, \mathcal{F}) \text{ is a tree-decomposition of } G\}$$

For two nodes u and v of V_T, we define $P_{u,v}$ as the unique path between u and v in T and the set $V_{u,v} \subseteq V$ as $\bigcup_{t \in P_{uv}} X_t$. A tree-decomposition $(T, \mathcal{F})$ is *connected* if there exists a node $r \in V_T$ such that for every node $u \in V_T$, the subgraph $G[V_{r,u}]$ is connected. Then the connected treewidth of a graph G is

$$\mathsf{ctw}(G) = \min\{\mathsf{width}(T, \mathcal{F}) \mid (T, \mathcal{F}) \text{ is a connected tree-decomposition of } G\}$$

A layout σ of G is *connected* if for every $i \in [n]$, the subgraph $G[\sigma_{\leq i}]$ is connected. We let $\mathcal{L}^c(G)$ be the set of connected layouts of a graph G. For every $v \in V$, we define the *supporting set* of v as

$$S_\sigma(v) = \big\{x \in V(G) \mid \sigma(x) < \sigma(v) \text{ and there exists a } (x,v)\text{-path } P \text{ such that every internal vertex } y \text{ of } P \text{ satisfies } \sigma(y) > \sigma(v)\big\}.$$

We set $\mathsf{cost}(G, \sigma) = \max\{|S_\sigma(v)| \mid v \in V\}$. The *tree vertex separation number* of a graph is defined as

$$\mathsf{vs}(G) = \min\{\mathsf{cost}(G, \sigma) \mid \sigma \in \mathcal{L}(G)\}.$$

When restricting to the set of connected layouts, we obtain the *connected tree vertex separation number*

$$\mathsf{cvs}(G) = \min\{\mathsf{cost}(G, \sigma) \mid \sigma \in \mathcal{L}^c(G)\}.$$

Theorem 3 ([1])**.** *For every graph $G = (V, E)$, we have $\mathsf{ctw}(G) = \mathsf{ctvs}(G)$.*

Notice that if in the above definitions we drop the connectivity demand from the considered layouts, we have that $\mathsf{tw}(G) = \mathsf{vs}(G)$ providing an alternative layout-definition of the parameter of treewidth, as observed in [2, 14]. It is known that deciding whether $\mathsf{tw}(G) \leq k$ is an NP-complete problem [3]. An easy reduction of treewidth to connected treewidth is the following: consider a graph G, add a vertex v_{new}, and make v_{new} adjacent to all the vertices of G. We call the new graph G^+. It follows, as a direct consequence of the layout definitions, that $\mathsf{ctw}(G^+) = \mathsf{tw}(G) + 1$. We conclude to the following.

Theorem 4. *The problem of deciding, given a graph G and an integer k, whether $\mathsf{ctw}(G) \leq k$ is NP-complete.*

2.3 Rooted graphs and extended graphs

Rooted graphs. A *rooted graph* is a pair (G, R) where $G = (V, E)$ is a graph and $\mathsf{R} \subseteq V$ is a subset of vertices, hereafter called *roots*. The definition of a rooted graph naturally extends to two-terminal graphs. If $\mathbf{G} = (G, (x, y))$ is a series-parallel graph and $\mathsf{R} \subseteq V$ a set of roots, then the corresponding rooted two-terminal graph will be denoted by $(\mathbf{G}, \mathsf{R})$. Observe that the set of roots R may be different from the terminal pair (x, y).

A rooted graph (G, R) is connected if and only if every connected component of G contains a root from R. A *rooted layout* of (G, R) is a layout σ of G such that $\sigma_{\leq |\mathsf{R}|} = \mathsf{R}$. Based on this, the notion of connected layout naturally extends to rooted graphs and rooted layouts as follows. A rooted layout σ of (G, R) is connected if for every i, $|\mathsf{R}| < i \leq n$, $(G[\sigma_{\leq i}], \mathsf{R})$ is a connected rooted graph.

Extended graphs. Let $\mathbf{G} = (G, (x, y))$ be a 2-terminal graph where $G = (V, E)$. Suppose that $F \subseteq \binom{V}{2} \setminus E(G)$, i.e., F is a subset of non-edges of G. We define the *extended* graph $\mathbf{G}^{+F}$ as the 2-terminal graph $(G^{+F}, (x, y))$ where $G^{+F} = (V, E \cup F)$. The edges in E are called *solid* edges, while the edges of F are called *fictive* edges. Hereafter the 2-terminal graph $\mathbf{G}$ (and the graph G respectively) is called the *solid graph* of $\mathbf{G}^{+F}$ (and G^{+F} respectively).

We define the connected components of the extended graph $\mathbf{G}^{+F}$ as the connected components of its solid graph $\mathbf{G}$. In particular, $\mathbf{G}^{+F}$ is connected if and only if $\mathbf{G}$ is connected. Thereby, we say a vertex is isolated in $\mathbf{G}^{+F}$ if it is isolated in $\mathbf{G}$. Likewise, we say that two vertices are adjacent in $\mathbf{G}^{+F}$ if they are in $\mathbf{G}$. Therefore the neighbourhood $N(v)$ of a vertex v in $\mathbf{G}^{+F}$ is its neighbourhood in $\mathbf{G}$, while its extended neighbourhood $N^{+F}(v)$ also comprises every vertex u such that $uv \in F$ (we say that u is an *extended-neighbour* of v). The purpose of introducing fictive edges is *not* to augment the connectivity of the solid graph but to keep track of cumulative cost in the recursive calls of the dynamic programming algorithm while computing the connected treewidth of a series-parallel graph.

This connectivity definition of an extended graph also transfers to (rooted) layouts. More precisely, if $\mathsf{R} \subseteq V$ is a set of roots and $\mathbf{G}^{+F}$ an extended graph, then $(\mathbf{G}^{+F}, \mathsf{R})$ is a *rooted extended graph*. A layout σ of $(\mathbf{G}^{+F}, \mathsf{R})$ is connected if and only if it is a connected layout of $(\mathbf{G}, \mathsf{R})$. Observe that if $\mathbf{G}^{+F}$ is not connected, then a connected layout of $\mathbf{G}^{+F}$ exists if and only if every connected component of $\mathbf{G}^{+F}$ contains a root from R.

An *extended path* of $\mathbf{G}^{+F}$ is a path that may contain a fictive edge of F, while a *solid path* in $\mathbf{G}^{+F}$ is a path of $\mathbf{G}$, that is, a path that only contains solid edges. In a layout σ of the extended graph $\mathbf{G}^{+F}$, the *extended supporting set* of vertex v is:

$$S_\sigma^{(e)}(v) = \big\{x \in V \mid \sigma(x) < \sigma(v) \text{ and there exists an extended } (x,v)\text{-path } P \text{ such that every internal vertex } y \text{ of } P \text{ belongs to } \sigma_{>i}\big\}.$$

The definitions of the extended cost $\mathsf{ecost}(\mathbf{G}^{+F}, \sigma)$ and of the extended connected tree vertex separation number $\mathsf{ectvs}(\mathbf{G}^{+F})$ follow accordingly:

$$\mathsf{ecvs}(\mathbf{G}^{+F}) = \min\{\mathsf{ecost}(\mathbf{G}^{+F}, \sigma) \mid \sigma \in \mathcal{L}^c(\mathbf{G}^{+F})\},$$

where $\mathsf{ecost}(\mathbf{G}^{+F}, \sigma) = \max\{|S_\sigma^{(e)}(v)| \mid v \in V\}$.

3 A dynamic programming algorithm

Before describing the dynamic programming algorithm to compute the connected treewidth of treewidth at most 2 graphs, we examine the case of biconnected series-parallel graphs. More precisely, we show how to derive the connected treewidth of a series-parallel graph depending on whether it results from a series or from a parallel composition. To handle these recursion steps, we have to manipulate extended series-parallel graphs.

3.1 Biconnected series-parallel graphs

In this section, we let $\mathbf{G} = (G, (x, y))$ be a series-parallel graph such that $G = (V, E)$. We suppose that G results from the series or the parallel composition of $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$. When dealing with the 2-terminal graphs $\mathbf{G}_1$ and $\mathbf{G}_2$, the terminal pairs will be clear from the context.

3.1.1 Parallel composition

Lemma 1. *Let* $(\mathbf{G}^{+\emptyset}, \mathsf{R})$ *be a rooted extended graph such that* $\mathsf{R} = \{x, y\}$ *and* $\mathbf{G} = \mathbf{G}_1 \oplus \mathbf{G}_2$ *with* $\mathbf{G}_1 = (G_1, (x, y))$ *and* $\mathbf{G}_2 = (G_2, (x, y))$ *(see Figure 2). Then*

$$\mathsf{ectvs}(\mathbf{G}^{+\emptyset}, \mathsf{R}) = \max\left\{\mathsf{ectvs}(\mathbf{G}_1^{+\emptyset}, \mathsf{R}), \mathsf{ectvs}(\mathbf{G}_2^{+\emptyset}, \mathsf{R})\right\}.$$

Proof. Let $\sigma^* \in \mathcal{L}^c(\mathbf{G}^{+\emptyset}, \mathsf{R})$ be a connected layout of minimum cost. From σ^*, we define a layout σ of $(\mathbf{G}^{+\emptyset}, \mathsf{R})$ as follows: $\sigma = \sigma^*[V_1] \odot \sigma^*[V_2 \setminus \{x, y\}]$ (see Figure 1).

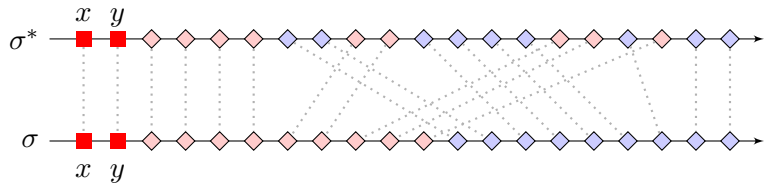

Figure 1: Rearranging a layout of minimum cost of an extended graph $\mathbf{G} = \mathbf{G}_1 \oplus \mathbf{G}_2$ without a fictive edge. Red vertices belongs to $V_1 \setminus \{x, y\}$ and blue vertices belong to $V_2 \setminus \{x, y\}$.

Observe that as $\{x, y\}$ separates V_1 from V_2 and as $\sigma^* \in \mathcal{L}^c(\mathbf{G}^{+\emptyset}, \mathsf{R})$, it follows that $\sigma \in \mathcal{L}^c(\mathbf{G}^{+\emptyset}, \mathsf{R})$, $\sigma_1 = \sigma[V_1] \in \mathcal{L}^c(\mathbf{G}_1^{+\emptyset}, \mathsf{R})$ and $\sigma_2 = \sigma[V_2] \in \mathcal{L}^c(\mathbf{G}_2^{+\emptyset}, \mathsf{R})$. Moreover, $\{x, y\}$ separating V_1 from V_2 implies that for every vertex $v_1 \in V_1 \setminus \{x, y\}$, we have $S^{(e)}_{\sigma^*}(v_1) = S^{(e)}_{\sigma_1}(v_1) \subseteq V_1$ and that for every vertex $v_2 \in V_2 \setminus \{x, y\}$, we have $S^{(e)}_{\sigma^*}(v_2) = S^{(e)}_{\sigma_2}(v_2) \subseteq V_2$. It follows that for every vertex $v \in V \setminus \{x, y\}$, $S^{(e)}_{\sigma^*}(v) = S^{(e)}_{\sigma}(v)$, implying that $\mathsf{ecost}(\mathbf{G}^{+\emptyset}, \sigma) = \mathsf{ecost}(\mathbf{G}^{+\emptyset}, \sigma^*)$.

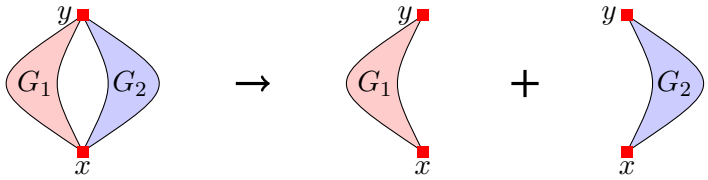

Figure 2: Parallel decomposition of an extended graph without fictive edges. If $\mathbf{G} = \mathbf{G}_1 \oplus \mathbf{G}_2$, then we have $\mathsf{ectvs}(\mathbf{G}^{+\emptyset}, \mathsf{R}) = \max\left\{\mathsf{ectvs}(\mathbf{G}_1^{+\emptyset}, \mathsf{R}), \mathsf{ectvs}(\mathbf{G}_2^{+\emptyset}, \mathsf{R})\right\}$.

To conclude, we observe that if $\tau_1 \in \mathcal{L}^c(\mathbf{G}_1^{+\emptyset}, \mathsf{R})$ and $\tau_2 \in \mathcal{L}^c(\mathbf{G}_2^{+\emptyset}, \mathsf{R})$, then $\tau = \tau_1 \odot \tau_2[V_2 \setminus \{x, y\}]$ belongs to $\mathcal{L}^c(\mathbf{G}^{+\emptyset}, \mathsf{R})$ and that $\mathsf{ecost}(\mathbf{G}^{+\emptyset}, \tau) = \max\{\mathsf{ecost}(\mathbf{G}_1^{+\emptyset}, \tau_1), \mathsf{ecost}(\mathbf{G}_2^{+\emptyset}, \tau_2)\}$. So the optimality of σ^* and σ imply that $\mathsf{ectvs}(\mathbf{G}^{+\emptyset}, \mathsf{R}) = \max\left\{\mathsf{ectvs}(\mathbf{G}_1^{+\emptyset}, \mathsf{R}), \mathsf{ectvs}(\mathbf{G}_2^{+\emptyset}, \mathsf{R})\right\}$. □

Lemma 2. *Let* $(\mathbf{G}^{+F}, \mathsf{R})$ *be a rooted extended graph such that* $\mathsf{R} = \{x, r_1 \ldots, r_k\}$ *(with* $k > 0$*),* $r_1, \ldots r_k$ *are isolated vertices (in* $\mathbf{G}$*),* $F = \{yr_i \mid i \in [k]\}$ *and* $(G[V \setminus \{r_1, \ldots, r_k\}], (x, y)) = \mathbf{G}_1 \oplus \mathbf{G}_2$ *with* $\mathbf{G}_1 = (G_1, (x, y))$ *and* $\mathbf{G}_2 = (G_2, (x, y))$ *(see Figure 3). Then*

$$\mathsf{ectvs}(\mathbf{G}^{+F}, \mathsf{R}) = \min\left\{\begin{array}{l} \max\left\{\mathsf{ectvs}(\mathbf{G}[V_1 \cup \mathsf{R}]^{+F\cup\{xy\}}, \mathsf{R}), \mathsf{ectvs}(\mathbf{G}_2^{+\emptyset}, \{x, y\})\right\} \\ \\ \max\left\{\mathsf{ectvs}(\mathbf{G}[V_2 \cup \mathsf{R}]^{+F\cup\{xy\}}, \mathsf{R}), \mathsf{ectvs}(\mathbf{G}_1^{+\emptyset}, \{x, y\})\right\} \end{array}\right\}.$$

Proof. We observe that $\mathbf{G}^{+F}$ contains k isolated vertices, the root vertices $r_1, \ldots, r_k$, and a connected component resulting from a parallel composition. Let $\sigma^* \in \mathcal{L}^c(\mathbf{G}^{+F}, \mathsf{R})$ be a connected layout of minimum cost. Consider the neighbor v of y so that $\sigma^*(v)$ is minimized. By the connectivity of σ^*, $\sigma^*(v) < \sigma^*(y)$. Suppose without loss of generality that $v \in V_1$. The case $v \in V_2$ is symmetric. Observe that v can be the vertex x, in which case it can arbitrarily be considered as a vertex of V_1 and V_2). From σ^*, we define the layout $\sigma_1 = \sigma[V_1 \cup \mathsf{R}]$ of $(\mathbf{G}[V_1 \cup \mathsf{R}]^{+F \cup \{xy\}}, \mathsf{R})$ and the layout $\sigma_2 = \sigma[V_2]$ of $(\mathbf{G}_2, \{x, y\})$, where $\sigma = \langle r_1, \ldots, r_k \rangle \odot \sigma^*[V_1] \odot \sigma^*[V_2 \setminus \{x, y\}]$ is a layout of $(\mathbf{G}^{+F}, \mathsf{R})$ (see Figure 1).

Claim 1. *$\sigma_1 \in \mathcal{L}^c(\mathbf{G}[V_1 \cup \mathsf{R}]^{+F \cup \{xy\}}, \mathsf{R})$, $\sigma_2 \in \mathcal{L}^c(\mathbf{G}_2, \{x, y\})$ and $\sigma \in \mathcal{L}^c(\mathbf{G}^{+F}, \mathsf{R})$.*

Proof of claim. Let v_1 be a vertex of V_1 distinct from x and y. Observe that every neighbor of v_1 belongs to V_1. By the assumption above, we know that y has a neighbor $v \in V_1$ prior to it in σ^*. As the relative ordering between vertices of V_1 is left unchanged, every vertex of $V_1 \setminus \{x\}$ has a neighbor prior to it in σ as well. This implies that $\sigma_1 \in \mathcal{L}^c(\mathbf{G}[V_1 \cup \mathsf{R}]^{+F \cup \{xy\}}, \mathsf{R})$. Consider now a vertex $v_2 \in V_2$ distinct from x and y. As in the previous case, every neighbor of v_2 belongs to V_2. As the relative ordering between vertices of V_2 has only been modified by moving y ahead, v_2 has a neighbor prior to it in σ. This implies that $\sigma_2 \in \mathcal{L}^c(\mathbf{G}_2, \{x, y\})$ and that $\sigma \in \mathcal{L}^c(\mathbf{G}^{+F}, \mathsf{R})$. ◇

Claim 2. $\mathsf{ecost}((\mathbf{G}^{+F}, \mathsf{R}), \sigma) = \mathsf{ecost}((\mathbf{G}^{+F}, \mathsf{R}), \sigma^*)$.

Proof of claim. We first consider a vertex $v_1 \in V_1$. By construction, we have $\sigma^*[V_1] = \sigma[V_1]$ and for every vertex $v_2 \in V_2 \setminus \{x, y\}$, $\sigma(v_1) \leqslant \sigma(v_2)$. It follows that $S_\sigma^{(e)}(v_1) \subseteq V_1 \cup \mathsf{R}$. Suppose that $\sigma(v_1) > \sigma(y)$. As $\{x, y\}$ separates v_1 from the vertices of $V_2 \cup \{r_1, \ldots, r_k\}$, we have $S_\sigma^{(e)}(v_1) = S_{\sigma^*}^{(e)}(v_1)$. Suppose that $\sigma(v_1) \leq \sigma(y)$. We observe that if for $i \in [k]$, $r_i \in S_\sigma^{(e)}(v_1)$, then there exists a (v_1, y)-path P in G_1 such that every vertex $v \in P$ satisfies $\sigma(v_1) \leq \sigma(v)$. Similarly, if $u \in V_1$ belongs to $S_\sigma^{(e)}(v_1)$, then there exists a (v_1, u)-path P' in G_1 such that every vertex $v' \in P'$ distinct from u satisfies $\sigma(v_1) \leq \sigma(v')$. The existence of the paths P and P' is a consequence of the fact that $\{x, y\}$ separates the vertices of $V_1 \setminus \{x, y\}$ from the rest of the graph. As $\sigma^*[V_1 \cup \mathsf{R}] = \sigma[V_1 \cup \mathsf{R}]$, we have $\sigma^*(v_1) \leq \sigma^*(v)$ and $\sigma^*(v_1) \leq \sigma^*(v')$. It follows that $r_i, u \in S_{\sigma^*}^{(e)}(v_1)$, implying in turn that $|S_\sigma^{(e)}(v_1)| \leq |S_{\sigma^*}^{(e)}(v_1)|$.

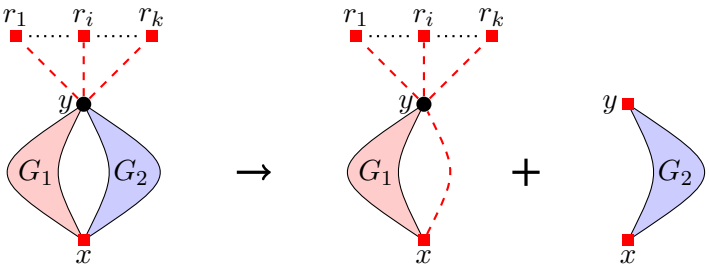

Figure 3: Decomposition of an extended graph with several isolated roots and resulting from a parallel composition.

Let v_2 be a vertex of $V_2 \setminus \{x, y\}$. Observe that, as $\{x, y\}$ separates the vertices of $V_2 \setminus \{x, y\}$ from the vertices of $\mathsf{R} \cup V_1 \setminus \{x, y\}$ and as $\sigma(x) < \sigma(y) < \sigma(v_2)$, we have $S_\sigma^{(e)}(v_2) \subseteq V_2$. As $\sigma[V_2 \setminus \{y\}] = \sigma^*[V_2 \setminus \{y\}]$, every vertex $u \in V_2 \setminus \{y\}$ that belongs to $S_\sigma^{(e)}(v_2)$ also belongs to

$S^{(e)}_{\sigma^*}(v_2)$. Suppose that $y \in S^{(e)}_{\sigma}(v_2)$. Then there exists a (y, v_2)-path P such that every internal vertex u of P satisfies $\sigma(v_2) < \sigma(u)$. As $\sigma[V_2 \setminus \{y\}] = \sigma^*[V_2 \setminus \{y\}]$, we also have $\sigma^*(v_2) < \sigma^*(v)$. Let us distinguish two cases:

- If $\sigma^*(y) < \sigma^*(v_2)$, then $y \in S^{(e)}_{\sigma}(v_2)$ implying that $S^{(e)}_{\sigma}(v_2) \subseteq S^{(e)}_{\sigma^*}(v_2)$.
- Otherwise, $\sigma^*(y) > \sigma^*(v_2)$ and then $y \notin S^{(e)}_{\sigma^*}(v_2)$. But in that case, let us recall that by assumption the first neighbor v of y in σ^* belongs to V_1. It follows that $v \in S^{(e)}_{\sigma^*}(v_2)$. As we argue that $S^{(e)}_{\sigma}(v_2) \subseteq V_2$, $v \in S^{(e)}_{\sigma^*}(v_2) \setminus S^{(e)}_{\sigma}(v_2)$. So v is a replacement vertex for y in $S^{(e)}_{\sigma^*}(v_2)$, implying that $|S^{(e)}_{\sigma}(v_2)| \leq |S^{(e)}_{\sigma^*}(v_2)|$.

So we proved that $\mathsf{ecost}((\mathbf{G}^{+F}, \mathsf{R}), \sigma) \leq \mathsf{ecost}((\mathbf{G}^{+F}, \mathsf{R}), \sigma^*)$. As by Claim 1, $\sigma \in \mathcal{L}^c(\mathbf{G}^{+F}, \mathsf{R})$, the optimality of σ^* implies that $\mathsf{ecost}((\mathbf{G}^{+F}, \mathsf{R}), \sigma) = \mathsf{ecost}((\mathbf{G}^{+F}, \mathsf{R}), \sigma^*)$. ◇

Let us now conclude the proof. Claim 1, Claim 2 and the optimality of σ^* imply that σ is a connected layout of $\mathbf{G}^{+F}$ of minimum cost. By Claim 1, we have $\sigma_1 = \sigma[V_1 \cup \mathsf{R}] \in \mathcal{L}^c(\mathbf{G}[V_1 \cup \mathsf{R}]^{+F \cup \{xy\}}, \mathsf{R})$. Observe that in the extended graph $\mathbf{G}[V_1 \cup \mathsf{R}]^{+F \cup \{xy\}}$, the edge xy simulates every (x, y)-path of G whose internal vertices belong to V_2. It follows that for every vertex $v_1 \in V_1 \cup \mathsf{R}$, $S^{(e)}_{\sigma}(v_1) = S^{(e)}_{\sigma_1}(v_1)$. Likewise, by Claim 1, we know that $\sigma_2 = \sigma[V_2] \in \mathcal{L}^c(\mathbf{G}_2, \{x, y\})$. As $\{x, y\}$ separates the vertices of V_2 from the other vertices, for every $v_2 \in V_2 \setminus \{x, y\}$, we have $S^{(e)}_{\sigma}(v_2) = S^{(e)}_{\sigma_2}(v_2)$. It follows that $\mathsf{ectvs}(\mathbf{G}^{+F}, \mathsf{R}) = \max\left\{\mathsf{ectvs}(\mathbf{G}[V_1 \cup \mathsf{R}]^{+F \cup \{xy\}}, \mathsf{R}), \mathsf{ectvs}(\mathbf{G}_2^{+\emptyset}, \{x, y\})\right\}$. (see Figure 3) □

The next lemma shows that, in the case of a parallel composition, a fictive edge between the two terminal vertices is irrelevant. This is because the graph is already biconnected.

Lemma 3. *Let $(\mathbf{G}^{+F \cup \{xy\}}, \mathsf{R})$ be a rooted extended graph such that $\mathsf{R} = \{x, r_1 \ldots, r_k\}$ (with $k > 0$), $r_1, \ldots r_k$ are isolated vertices (in $\mathbf{G}$), $F = \{yr_i \mid i \in [k]\}$ and $(G[V \setminus \{r_1, \ldots, r_k\}], (x, y)) = \mathbf{G}_1 \oplus \mathbf{G}_2$ with $\mathbf{G}_1 = (G_1, (x, y))$ and $\mathbf{G}_2 = (G_2, (x, y))$. Then*

$$\mathsf{ectvs}(\mathbf{G}^{+F \cup \{xy\}}, \mathsf{R}) = \mathsf{ectvs}(\mathbf{G}^{+F}, \mathsf{R}).$$

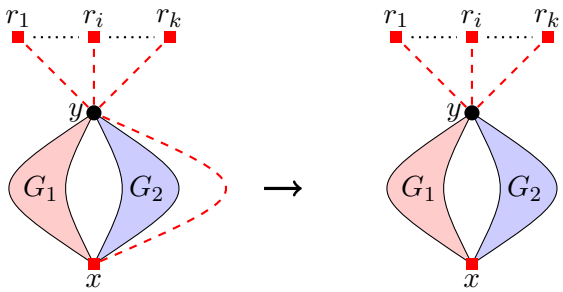

Figure 4: The fictive edge xy is irrelevant: $\mathsf{ectvs}(\mathbf{G}^{+F \cup \{xy\}}, \mathsf{R}) = \mathsf{ectvs}(\mathbf{G}^{+F}, \mathsf{R})$.

Proof. Let $\sigma^* \in \mathcal{L}^c(\mathbf{G}^{+F \cup \{xy\}}, \mathsf{R})$ be a connected layout of minimum cost. Consider the neighbor v of y such that $\sigma^*(v)$ is minimum. By the connectivity of σ^*, $\sigma^*(v) < \sigma^*(y)$. Suppose without loss of generality that $v \in V_1$. The case $v \in V_2$ is symmetric.

As in the proof of Lemma 2, we define the layout $\sigma_1 = \sigma[V_1 \cup \mathsf{R}]$ of $(\mathbf{G}[V_1 \cup \mathsf{R}]^{+F \cup \{xy\}}, \mathsf{R})$ and the layout $\sigma_2 = \sigma[V_2]$ of $(\mathbf{G}_2, \{x, y\})$, where $\sigma = \langle r_1, \ldots, r_k \rangle \odot \sigma^*[V_1] \odot \sigma^*[V_2 \setminus \{x, y\}]$

is a layout of $(\mathbf{G}^{+F\cup\{xy\}}, \mathsf{R})$. We first observe that Claim 1 applies to σ_1, σ_2 and σ which are thereby connected layouts. We also remark that the proof of Claim 2 applies to σ even in the presence of the fictive edge xy. So we have $\sigma \in \mathcal{L}^c(\mathbf{G}^{+F}, \mathsf{R})$ and $\mathsf{ecost}((\mathbf{G}^{+F}, \mathsf{R}), \sigma) = \mathsf{ecost}((\mathbf{G}^{+F}, \mathsf{R}), \sigma^*)$. Following the proof of Lemma 2, we show that for every $v_1 \in V_1 \cup \mathsf{R}$, $S^{(e)}_{\sigma}(v_1) = S^{(e)}_{\sigma_1}(v_1)$ and that for every $v_2 \in V_2 \setminus \{x, y\}$, $S^{(e)}_{\sigma}(v_2) = S^{(e)}_{\sigma_2}(v_2)$. So we proved that $\mathsf{ectvs}(\mathbf{G}^{+F\cup\{xy\}}, \mathsf{R}) = \max\left\{\mathsf{ectvs}(\mathbf{G}[V_1 \cup \mathsf{R}]^{+F\cup\{xy\}}, \mathsf{R}), \mathsf{ectvs}(\mathbf{G}_2^{+\emptyset}, \{x, y\})\right\}$. Lemma 2 allows us to conclude that $\mathsf{ectvs}(\mathbf{G}^{+F\cup\{x,y\}}, \mathsf{R}) = \mathsf{ectvs}(\mathbf{G}^{+F}, \mathsf{R})$ as claimed. In other words the fictive edge xy is irrelevant (see Figure 4). □

3.1.2 Series composition

Lemma 4. *Let $(\mathbf{G}^{+\emptyset}, \mathsf{R})$ be a rooted extended graph with $\mathsf{R} = \{x, y\}$ and such that $\mathbf{G} = \mathbf{G}_1 \otimes \mathbf{G}_2$ with $\mathbf{G}_1 = (G_1, (x, z))$ and $\mathbf{G}_2 = (G_2, (z, y))$ (see Figure 6). Then*

$$\mathsf{ectvs}(\mathbf{G}^{+\emptyset}, \mathsf{R}) = \min \left\{ \begin{array}{l} \max\left\{\mathsf{ectvs}(\tilde{\mathbf{G}}_1^{+\{zy\}}, \mathsf{R}), \mathsf{ectvs}(\mathbf{G}_2^{+\emptyset}, \mathsf{R}_2)\right\} \\ \\ \max\left\{\mathsf{ectvs}(\tilde{\mathbf{G}}_2^{+\{zx\}}, \mathsf{R}), \mathsf{ectvs}(\mathbf{G}_1^{+\emptyset}, \mathsf{R}_1)\right\} \end{array} \right\},$$

where $\tilde{\mathbf{G}}_1$ (resp. $\tilde{\mathbf{G}}_2$) is obtained from $\mathbf{G}_1$ (resp. $\mathbf{G}_2$) by adding y (resp. x) as an isolated vertex, and where $\mathsf{R}_1 = \{z, x\}$, $\mathsf{R}_2 = \{z, y\}$.

Proof. Let $\sigma^* \in \mathcal{L}^c(\mathbf{G}, \mathsf{R})$ be a connected layout of minimum cost. Consider the neighbor v of z so that $\sigma^*(v)$ is minimized. By the connectivity of σ^*, $\sigma^*(v) < \sigma^*(z)$. Suppose without loss of generality that $v \in V_1$. The case $v \in V_2$ is symmetric. From σ^*, we define $\sigma_1 = \sigma[V_1 \cup \mathsf{R}]$ a layout of $(\mathsf{ectvs}(\tilde{\mathbf{G}}_1^{+\{zy\}}, \mathsf{R})$, $\sigma_2 = \langle z, y\rangle \odot \sigma[V_2 \setminus \{y, z\}]$ a layout of $(\mathbf{G}_2^{+\emptyset}, \mathsf{R}_2)$ and $\sigma = \langle x, y\rangle \odot \sigma^*[V_1 \setminus \{x\}] \odot \sigma^*[V_2 \setminus \{y, z\}]$ a layout of $(\mathbf{G}^{+\emptyset}, \mathsf{R})$ (see Figure 5).

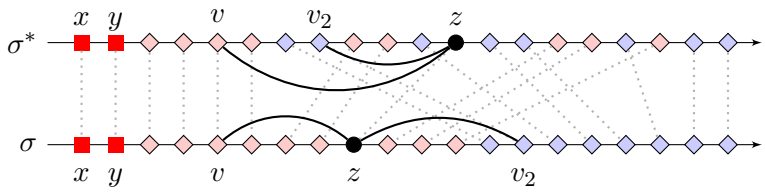

Figure 5: Rearranging a layout σ^* of $\mathbf{G} = \mathbf{G}_1 \otimes \mathbf{G}_2$ of minimum cost into $\sigma = \langle x, y\rangle \odot \sigma^*[V_1 \setminus \{x\}] \odot \sigma^*[V_2 \setminus \{y, z\}]$. Red diamond vertices belong to $V_1 \setminus \{x\}$, blue diamond vertices belong to $V_2 \setminus \{y, z\}$ and red square vertices are the roots. Observe that the path v, z, v_2 certifies that $v \in S^{(e)}_{\sigma^*}(v_2)$. But as $\sigma(z) < \sigma(v_2)$, $v \notin S^{(e)}_{\sigma}(v_2)$. Instead we have that $z \in S^{(e)}_{\sigma}(v_2)$.

Claim 3. *$\sigma_1 \in \mathcal{L}^c(\mathsf{ectvs}(\tilde{\mathbf{G}}_1^{+\{zy\}}, \mathsf{R}))$, $\sigma_2 \in \mathcal{L}^c(\mathbf{G}_2^{+\emptyset}, \mathsf{R}_2)$ and $\sigma \in \mathcal{L}^c(\mathbf{G}^{+\emptyset}, \mathsf{R})$.*

Proof of claim. Let v_1 be a vertex of V_1 distinct from x and z. Every neighbor of v_1 belongs to V_1. As the relative ordering between vertices of V_1 is left unchanged, vertex v_1 has a neighbor prior to it in σ. It follows that $\sigma_1 \in \mathcal{L}^c(\mathsf{ectvs}(\tilde{\mathbf{G}}_1^{+\{zy\}}, \mathsf{R}))$. Suppose that v_2 is a vertex of V_2 distinct from y and z. Then as in the previous case, every neighbor of v_2 belongs to V_2. As the relative ordering in σ_2 and σ between vertices of V_2 has only been modified by moving z ahead, v_2 has a neighbor prior to it in σ. It follows that $\sigma_2 \in \mathcal{L}^c(\mathbf{G}_2^{+\emptyset}, \mathsf{R}_2)$. To prove that σ is connected, we are left with vertex z. By assumption, z has a neighbor $v \in V_1$ such that

$\sigma^*(v) < \sigma^*(z)$, implying that $\sigma(v) < \sigma(z)$. As every vertex has a neighbor prior to it in σ, the layout σ is connected. $\diamond$

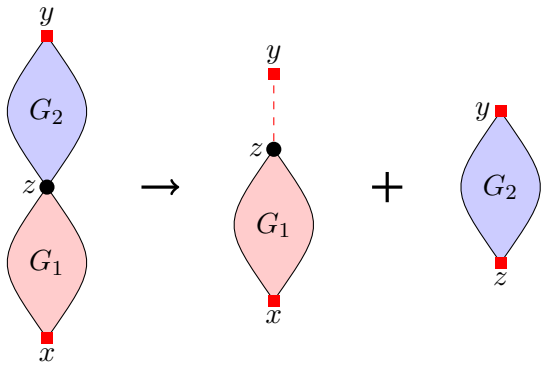

Figure 6: Decomposition of an extended graph resulting from a series composition.

Claim 4. $\mathsf{ecost}((\mathbf{G}^{+\emptyset}, \mathsf{R}), \sigma) = \mathsf{ecost}((\mathbf{G}^{+\emptyset}, \mathsf{R}), \sigma^*)$.

Proof of claim. We first consider a vertex $v_1 \in V_1$. By construction, we have $\sigma^*[V_1] = \sigma[V_1]$ and for every vertex $v_2 \in V_2 \setminus \{x, y\}$, $\sigma(v_1) \leqslant \sigma(v_2)$. It follows that $S^{(e)}_{\sigma}(v_1) \subseteq V_1 \cup \mathsf{R}$. Suppose that $\sigma(v_1) > \sigma(z)$. As z separates v_1 from the vertices of V_2, we have that $S^{(e)}_{\sigma}(v_1) = S^{(e)}_{\sigma^*}(v_1)$. Suppose now that $\sigma(v_1) \leq \sigma(z)$ and let $v \in V_1$ a vertex that belongs to $S^{(e)}_{\sigma}(v_1)$. Then, as z is a cut vertex of $\mathbf{G}^{+\emptyset}$, there exists a (v, v_1)-path P in G_1 such that every vertex $v' \in P$ distinct from v satisfies $\sigma(v_1) \leq \sigma(v')$. As $\sigma^*[V_1] = \sigma[V_1]$, the path P certifies that $v \in S^{(e)}_{\sigma^*}(v_1)$, implying that $|S^{(e)}_{\sigma}(v_1)| \leq |S^{(e)}_{\sigma^*}(v_1)|$.

Let us now consider a vertex $v_2 \in V_2 \setminus \{y, z\}$. Observe that, as z is a cut vertex, $\sigma^*(z) < \sigma^*(v_2)$ implies that $S^{(e)}_{\sigma^*}(v_2) \subseteq V_2$. As $\sigma[V_2 \setminus \{z\}] = \sigma^*[V_2 \setminus \{z\}]$, every vertex $u \in V_2 \setminus \{z\}$ that belongs to $S^{(e)}_{\sigma}(v_2)$ also belongs to $S^{(e)}_{\sigma^*}(v_2)$. Suppose that $z \in S^{(e)}_{\sigma}(v_2)$. Then there exists a (z, v_2)-path P such that every internal vertex u of P satisfies $\sigma(v_2) < \sigma(u)$. As $\sigma[V_2 \setminus \{z\}] = \sigma^*[V_2 \setminus \{z\}]$, we also have $\sigma^*(v_2) < \sigma^*(v)$. Let us distinguish two cases:

- If $\sigma^*(z) < \sigma^*(v_2)$, then $z \in S^{(e)}_{\sigma}(v_2)$ implying that $S^{(e)}_{\sigma}(v_2) \subseteq S^{(e)}_{\sigma^*}(v_2)$.
- Otherwise, $\sigma^*(z) > \sigma^*(v_2)$ and then $z \notin S^{(e)}_{\sigma^*}(v_2)$. But in that case, let us recall that by assumption the first neighbor v of z in σ^* belongs to V_1. It follows that $v \in S^{(e)}_{\sigma^*}(v_2)$. As we argue that $S^{(e)}_{\sigma}(v_2) \subseteq V_2$, $v \in S^{(e)}_{\sigma^*}(v_2) \setminus S^{(e)}_{\sigma}(v_2)$. So v is a replacement vertex for z in $S^{(e)}_{\sigma^*}(v_2)$, implying that $|S^{(e)}_{\sigma}(v_2)| \leq |S^{(e)}_{\sigma^*}(v_2)|$.

So we proved that $\mathsf{ecost}((\mathbf{G}^{+\emptyset}, \mathsf{R}), \sigma) \leq \mathsf{ecost}((\mathbf{G}^{+\emptyset}, \mathsf{R}), \sigma^*)$. As by Claim 3, $\sigma \in \mathcal{L}^c(\mathbf{G}^{+F}, \mathsf{R})$, the optimality of σ^* implies that $\mathsf{ecost}((\mathbf{G}^{+\emptyset}, \mathsf{R}), \sigma) = \mathsf{ecost}((\mathbf{G}^{+\emptyset}, \mathsf{R}), \sigma^*)$. $\diamond$

Let us now conclude the proof. Claim 3, Claim 4 and the optimality of σ^* imply that σ is a connected layout of $\mathbf{G}^{+F}$ of minimum cost. By Claim 3, we have $\sigma_1 = \sigma[V_1 \cup \mathsf{R}] \in \mathcal{L}^c(\tilde{\mathbf{G}}_1^{+\{xy\}}, \mathsf{R})$. Observe that in the extended graph $\tilde{\mathbf{G}}_1^{+\{xy\}}$, the fictive edge xy simulates every (z, y)-path of $\mathbf{G}$ whose internal vertices belong to V_2. It follows that for every vertex $v_1 \in V_1 \cup \mathsf{R}$, $S^{(e)}_{\sigma}(v_1) = S^{(e)}_{\sigma_1}(v_1)$. Likewise, by Claim 3, we know that $\sigma_2 = \sigma[V_2] \in \mathcal{L}^c(\mathbf{G}_2^{+\emptyset}, \{z, y\})$. As z separates the vertices of V_2 from the other vertices, for every $v_2 \in V_2 \setminus \{z\}$, we have $S^{(e)}_{\sigma}(v_2) = S^{(e)}_{\sigma_2}(v_2)$. It follows that $\mathsf{ectvs}(\mathbf{G}^{+\emptyset}, \mathsf{R}) = \max\left\{\mathsf{ectvs}(\tilde{\mathbf{G}}_1^{+\{zy\}}, \mathsf{R}), \mathsf{ectvs}(\mathbf{G}_2^{+\emptyset}, \mathsf{R}_2)\right\}$. $\square$

Let us now consider a rooted extended graph $(\mathbf{G}^{+F}, \mathsf{R})$ with $\mathsf{R} = \{x, r_1 \dots, r_k\}$ and $(G[V \setminus \{r_1, \dots, r_k\}], (x, y)) = \mathbf{G}_1 \otimes \mathbf{G}_2$ (see Figure 8). The fact that y is not a root vertex forces that in a connected layout starting from the root x forces the vertex z to be the first vertex of V_2. Using this observation, one can apply the same arguments than in the proof of Lemma 4.

Lemma 5. *Let $(\mathbf{G}^{+F}, \mathsf{R})$ be a rooted extended graph such that $\mathsf{R} = \{x, r_1 \dots, r_k\}$ (with $k > 0$), $r_1, \dots r_k$ are isolated vertices (in $\mathbf{G}$), $F = \{yr_i \mid i \in [k]\}$ and $(G[V \setminus \{r_1, \dots, r_k\}], (x, y)) = \mathbf{G}_1 \otimes \mathbf{G}_2$ with $\mathbf{G}_1 = (G_1, (x, z))$ and $\mathbf{G}_2 = (G_2, (z, y))$. Then*

$$\mathsf{ectvs}(\mathbf{G}^{+F}, \mathsf{R}) = \max\left\{\mathsf{ectvs}(\mathbf{G}[V_1 \cup \mathsf{R}]^{+F'}, \mathsf{R}), \mathsf{ectvs}(\mathbf{G}[V_2 \cup \mathsf{R}']^{+F}, \mathsf{R}')\right\},$$

where $F' = \{zr_i \mid i \in [k]\}$ and $\mathsf{R}' = \{z, r_1, \dots, r_k\}$.

Proof. Let $\sigma^* \in \mathcal{L}^c(\mathbf{G}^{+F}, \mathsf{R})$ be a connected layout of minimum cost. From σ^*, we define: $\sigma_1 = \sigma[V_1 \cup \mathsf{R}]$ a layout of $(\mathbf{G}[V_1 \cup \mathsf{R}]^{+F'}, \mathsf{R})$, $\sigma_2 = \langle z, y \rangle \odot \sigma[V_2 \setminus \{y, z\}]$ a layout of $(\mathbf{G}[V_2 \cup \mathsf{R}']^{+F}, \mathsf{R}')$ and $\sigma = \langle r_1, \dots, r_k \rangle \odot \sigma^*[V_1] \odot \sigma^*[V_2 \setminus \{z\}]$ (see Figure 7) a layout of $(\mathbf{G}^{+F}, \mathsf{R})$. We observe that in this case, as y is not a root, $\sigma^*(z) \leq \sigma^*(v_2)$ for every vertex $v_2 \in V_2$.

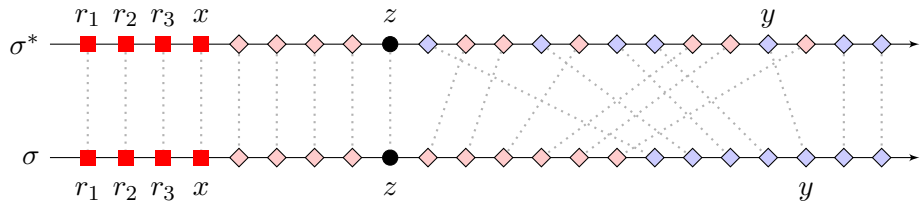

Figure 7: Rearranging a layout σ^* of minimum cost into $\sigma = \langle r_1, r_2, r_3, x \rangle \odot \sigma^*[V_1 \setminus \{x\}] \odot \sigma^*[V_2 \setminus \{z\}]$. Red diamond vertices belong to $V_1 \setminus \{x\}$, blue diamond vertices belong to $V_2 \setminus \{z\}$, red square vertices are the roots.

Claim 5. *$\sigma_1 \in \mathcal{L}^c(\mathbf{G}[V_1 \cup \mathsf{R}]^{+F'}, \mathsf{R})$, $\sigma_2 \in \mathcal{L}^c(\mathbf{G}[V_2 \cup \mathsf{R}']^{+F}, \mathsf{R}')$ and $\sigma \in \mathcal{L}^c(\mathbf{G}^{+F}, \mathsf{R})$.*

Proof of claim. Let v_1 be a vertex of V_1 distinct from x. Every neighbor of v_1 belongs to V_1. As the relative ordering between vertices of V_1 is left unchanged, vertex v_1 has a neighbor prior to it in σ. It follows that $\sigma_1 \in \mathcal{L}^c(\mathbf{G}[V_1 \cup \mathsf{R}]^{+F'}, \mathsf{R})$. Suppose that v_2 is a vertex of V_2 distinct from z. Then as in the previous case, every neighbor of v_2 belongs to V_2. As the relative ordering between vertices of V_2 is left unchanged in σ_2 and σ, vertex v_2 has a neighbor prior to it in σ. It follows that $\sigma_2 \in \mathcal{L}^c(\mathbf{G}[V_2 \cup \mathsf{R}']^{+F}, \mathsf{R}')$ and $\sigma \in \mathcal{L}^c(\mathbf{G}^{+F}, \mathsf{R})$. ◇

Claim 6. $\mathsf{ecost}((\mathbf{G}^{+F}, \mathsf{R}), \sigma) = \mathsf{ecost}((\mathbf{G}^{+F}, \mathsf{R}), \sigma^*)$.

Proof of claim. We first consider a vertex $v_1 \in V_1$. Observe that $\sigma^*(z) \leq \sigma^*(v_2)$ for every vertex $v_2 \in V_2$. It follows that $S^{(e)}_{\sigma^*}(v_1) \cap (V_2 \setminus \{z\}) = \emptyset$. The fact that $\sigma^*[V_1 \cup \mathsf{R}] = \sigma[V_1 \cup \mathsf{R}]$ implies $S^{(e)}_{\sigma}(v_1) = S^{(e)}_{\sigma^*}(v_1)$. Similar arguments hold for every vertex $v_2 \in V_2 \setminus \{z\}$. As z separates vertices of V_2 from vertices of V_1 and as $\sigma^*(x) < \sigma^*(z) < \sigma^*(v_2)$, we have $S^{(e)}_{\sigma^*}(v_2) \cap (V_1 \setminus \{x, z\}) = \emptyset$. The fact that $\sigma^*[V_2 \cup \mathsf{R}] = \sigma[V_2 \cup \mathsf{R}]$ implies $S^{(e)}_{\sigma}(v_2) = S^{(e)}_{\sigma^*}(v_2)$. Thereby $\mathsf{ecost}((\mathbf{G}^{+F}, \mathsf{R}), \sigma) = \mathsf{ecost}((\mathbf{G}^{+F}, \mathsf{R}), \sigma^*)$. ◇

Let us now conclude the proof of the lemma. Claim 5, Claim 6 and the optimality of σ^* imply that σ is a connected layout of G^{+F} of minimum cost. From Claim 5, we have that

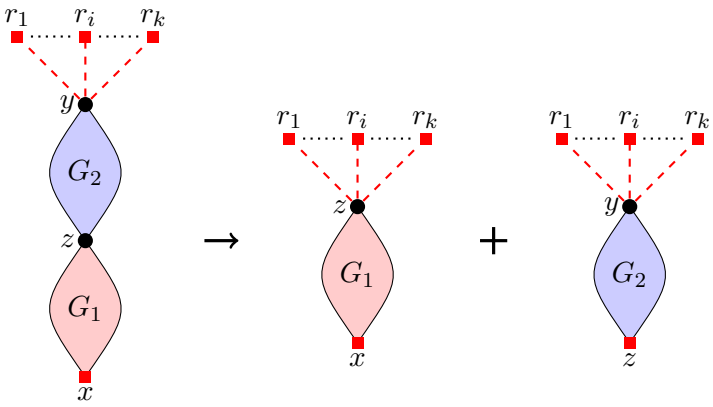

Figure 8: Decomposition of an extended graph resulting from a series composition. A connected layout starting at x places z as the first vertex of V_2.

$\sigma_1 = \sigma[V_1 \cup \mathsf{R}] \in \mathcal{L}^c(\mathbf{G}[V_1 \cup \mathsf{R}]^{+F'}, \mathsf{R})$. Observe that in the extended graph $\mathbf{G}[V_1 \cup \mathsf{R}]^{+F'}$, for $i \in [k]$, the fictive edge $zr_i \in F'$ simulates every simple extended (z, r_i)-path in $\mathbf{G}^{+F}$. It follows that for every vertex $v_1 \in V_1 \cup \mathsf{R}$, $S^{(e)}_{\sigma}(v_1) = S^{(e)}_{\sigma_1}(v_1)$. Likewise, from Claim 5, we know that $\sigma_2 = \sigma[V_2 \cup \mathsf{R}'] \in \mathcal{L}^c(\mathbf{G}[V_2 \cup \mathsf{R}']^{+F}, \mathsf{R}')$. As noticed before, z separates the vertices of V_2 from vertices of V_1. It follows for every $v_2 \in V_2 \setminus \{z\}$, we have $S^{(e)}_{\sigma}(v_2) = S^{(e)}_{\sigma_2}(v_2)$, completing the proof. □

If there is a fictive edge between the two terminal vertices x and y of $\mathbf{G}$, then , as in Lemma 5, a connected layout starting at x may visit y before z. In this case, we obtain the following lemma.

Lemma 6. *Let $(\mathbf{G}^{+F\cup\{xy\}}, \mathsf{R})$ be a rooted extended graph such that $\mathsf{R} = \{x, r_1 \dots, r_k\}$ (with $k > 0$), $r_1, \dots r_k$ are isolated vertices (in $\mathbf{G}$), $F = \{yr_i \mid i \in [k]\}$ and $(G[V \setminus \{r_1, \dots, r_k\}], (x, y)) = \mathbf{G}_1 \otimes \mathbf{G}_2$ with $\mathbf{G}_1 = (G_1, (x, z))$ and $\mathbf{G}_2 = (G_2, (z, y))$. Then*

$$\mathsf{ectvs}(\mathbf{G}^{+F\cup\{xy\}}, \mathsf{R}) = \max\left\{\mathsf{ectvs}(\mathbf{G}[V_1 \cup \mathsf{R}]^{+F'\cup\{xz\}}, \mathsf{R}), \mathsf{ectvs}(\mathbf{G}[V_2 \cup \mathsf{R}']^{+F\cup\{xy\}}, \mathsf{R}')\right\},$$

where $\mathsf{R}' = \{z, r_1, \dots, r_k, x\}$ and $F' = \{zr_i \mid i \in [k]\}$.

Proof. We proceed as in the proof of Lemma 5. We transform a layout $\sigma^* \in \mathcal{L}^c(\mathbf{G}^{+F\cup\{xy\}}), \mathsf{R})$ of minimum cost into the layout $\sigma = \langle r_1, \dots, r_k\rangle \odot \sigma^*[V_1] \odot \sigma^*[V_2 \setminus \{z\}]$ (see Figure 7). Observe that since the solid graph G is the same as in Lemma 5, Claim 5 applies and thereby $\sigma \in \mathcal{L}^c(\mathbf{G}^{+F\cup\{xy\}}, \mathsf{R})$.

Claim 7. $\mathsf{ecost}((\mathbf{G}^{+F\cup\{xy\}}, \mathsf{R}), \sigma) = \mathsf{ecost}((\mathbf{G}^{+F\cup\{xy\}}, \mathsf{R}), \sigma^*)$.

Proof of claim. The existence of the fictive edge xy does not change the arguments used in the proof of Claim 6. Let us consider $v_1 \in V_1$ and $v_2 \in V_2 \setminus \{z\}$. First as $\sigma^*(x) < \sigma^*(z) < \sigma^*(v_2)$ and z separates vertices of V_2 from vertices of V_1, we obtain that $S^{(e)}_{\sigma^*}(v_1) \cap (V_2 \setminus \{z\}) = \emptyset$ and $S^{(e)}_{\sigma^*}(v_2) \cap (V_1 \setminus \{x, z\}) = \emptyset$. Moreover $\sigma^*[V_1 \cup \mathsf{R}] = \sigma[V_1 \cup \mathsf{R}]$ implies $S^{(e)}_{\sigma}(v_1) = S^{(e)}_{\sigma^*}(v_1)$ and $\sigma^*[V_2 \cup \mathsf{R}] = \sigma[V_2 \cup \mathsf{R}]$ implies $S^{(e)}_{\sigma}(v_2) = S^{(e)}_{\sigma^*}(v_2)$, proving the claim. In other words, σ is a connected layout of $\mathbf{G}^{+F\cup\{xy\}}$ of minimum cost. ◇

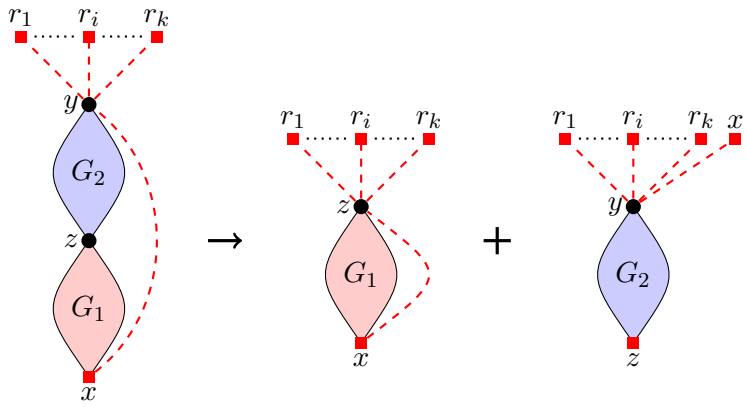

Figure 9: Decomposition of an extended graph resulting from a series composition.

Let us now conclude the proof of the lemma. Claim 7, the fact that $\sigma \in \mathcal{L}^c(\mathbf{G}^{+F\cup\{xy\}}, \mathsf{R})$ and the optimality of σ^* imply that σ is a connected layout of $\mathbf{G}^{+F}$ of minimum cost. From Claim 5, we have that $\sigma_1 = \sigma[V_1 \cup \mathsf{R}] \in \mathcal{L}^c(\mathbf{G}[V_1 \cup \mathsf{R}]^{+F'\cup\{xy\}}, \mathsf{R})$. Observe that in the extended graph $\mathbf{G}[V_1 \cup \mathsf{R}]^{+F'}$, for $i \in [k]$, the fictive edge $zr_i \in F'$ (for $i \in [k]$) simulates every simple extended (z, r_i)-path in $\mathbf{G}^{+F\cup\{xy\}}$. Similarly the fictive edge zx aims at representing extended (z, x)-paths avoiding $V_1 \setminus \{x\}$ in $\mathbf{G}^{+F\cup\{xy\}}$. It follows that for every vertex $v_1 \in V_1 \cup \mathsf{R}$, $S^{(e)}_\sigma(v_1) = S^{(e)}_{\sigma_1}(v_1)$. Likewise, from Claim 5, we know that $\sigma_2 = \sigma[V_2 \cup \mathsf{R}'] \in \mathcal{L}^c(\mathbf{G}[V_2 \cup \mathsf{R}']^{+F\cup\{zx\}}, \mathsf{R}')$. As noticed before, z separates the vertices of V_2 from vertices of V_1, and thereby for every $v_2 \in V_2$, $V_1 \setminus \{z\} \cap S^{(e)}_\sigma(v_2) = \emptyset$. Observe that despite the fact that $x \notin V_2$, it is preserved as a (pendant) root in R'. It follows for every $v_2 \in V_2 \setminus \{z\}$, we have $S^{(e)}_\sigma(v_2) = S^{(e)}_{\sigma_2}(v_2)$, completing the proof. □

3.2 The dynamic programming algorithm

The following upper bound on connected treewidth enables us to optimize the size of the DP tables.

Theorem 5 ([21, 22, 23])**.** *Every graph G on n vertices satisfies* $\mathsf{ctw}(G) \leq \mathsf{tw}(G) \cdot (\log n + 1)$.

As series-parallel graphs have treewidth at most two, Theorem 5 implies that the connected treewidth of a series-parallel graph on n vertices is at most $c_{\mathrm{sp}} = \lceil 2(\log n + 1) \rceil$. This bound allows us to optimize the size of the table in our dynamic programming algorithm. Moreover this bound is tight [22, 23], even on series-parallel graphs as certified by a construction of contraction obstructions for connected treewidth at most k, for every $k \geq 2$ [1]. Let us first focus on the case of biconnected series-parallel graph.

Proposition 1. *Let G be a biconnected series-parallel graph on n vertices. Then computing* $\mathsf{ctw}(G)$ *can be done in $O(n^2 \cdot \log n)$-time.*

Proof. Let $\mathbf{G} = (G, (x, y))$, with $G = (V, E)$, be a biconnected 2-terminal graph such that $xy \in E$. By Theorem 1, we have $\mathbf{G} = \mathbf{G}_1 \oplus \mathbf{G}_2$ where $\mathbf{G}_1 = (G_1, (x, y))$ with $G_1 = (\{x, y\}, \{xy\})$ and $\mathbf{G}_2 = (G_2, (x, y))$ with $G_2 = (V, E \setminus \{xy\})$. By Theorem 2, in linear time, we can compute $T(\mathbf{G})$, the SP-tree of $\mathbf{G}$. Recall that the root of $T(\mathbf{G})$ corresponds to the parallel composition $\mathbf{G}_1 \oplus \mathbf{G}_2$. We let $\mathbf{G}_t = (G_t, (x_t, y_t))$ with $G_t = (V_t, E_t)$ denote the subgraph represented by node

t of $T(\mathbf{G})$. We let $\tilde{G}_{t,k}$ be the graph G_t augmented with k isolated vertices $r_1, \ldots, r_k$ where $k \leq c_{\mathrm{sp}}$ and denote $\tilde{\mathbf{G}}_{t,k} = (\tilde{G}_{t,k}, (x_t, y_t))$. In order to apply the rules described in Lemmas 1 – 6, the table $\mathsf{DP}_t[\,\cdot\,]$ stored at every node t contains the following values:

- $\mathsf{DP}_t[0] = \mathsf{ectvs}(\mathbf{G}_t^{+\emptyset}, \{x_t, y_t\})$;
- for $k \in [c_{\mathrm{sp}}]$, $\mathsf{DP}_t[k, x_t] = \mathsf{ectvs}(\tilde{\mathbf{G}}_{t,k}[V_t \cup \mathsf{R}]^{+F}, \mathsf{R})$ where $\mathsf{R} = \{x_t, r_1, \ldots, r_k\}$ and $F = \{y_t r_i \mid i \in [k]\}$;
- for $k \in [c_{\mathrm{sp}}]$, $\mathsf{DP}_t[k, y_t] = \mathsf{ectvs}(\tilde{\mathbf{G}}_{t,k}[V_t \cup \mathsf{R}]^{+F}, \mathsf{R})$ where $\mathsf{R} = \{y_t, r_1, \ldots, r_k\}$ and $F = \{x_t r_i \mid i \in [k]\}$;
- for $k \in [c_{\mathrm{sp}} - 1]$, $\mathsf{DP}_t[k, x_t, x_t y_t] = \mathsf{ectvs}(\tilde{\mathbf{G}}_{t,k}[V_t \cup \mathsf{R}]^{+F}, \mathsf{R})$ where $\mathsf{R} = \{x_t, r_1, \ldots, r_k\}\}$ and $F = \{x_t y_t\} \cup \{y_t r_i \mid i \in [k]\}$;
- for $k \in [c_{\mathrm{sp}} - 1]$, $\mathsf{DP}_t[k, y_t, x_t y_t] = \mathsf{ectvs}(\tilde{\mathbf{G}}_{t,k}[V_t \cup \mathsf{R}]^{+F}, \mathsf{R})$ where $\mathsf{R} = \{y_t, r_1, \ldots, r_k\}\}$ and $F = \{x_t y_t\} \cup \{x_t r_i \mid i \in [k]\}$;

The bounds on the integer k, determining the number of entries in the table $\mathsf{DP}_t[\,\cdot\,]$ of a node t is, are delimited by the upper-bound c_{sp}, as asserted by Theorem 5. The initialization of the table for leaf nodes (see below) guarantees that this bound is respected.

We observe that for every node t, every entry of $\mathsf{DP}_t[\,\cdot\,]$ corresponds to an extended rooted two-terminal graph $(\mathbf{H}^{+F}, \mathsf{R})$ such that: R contains at least two vertices; at least one vertex of R is a terminal vertex; and every root vertex that is not a terminal vertex is an isolated vertex. These properties implies that every connected component of $\mathbf{H}^{+F}$ contains a root vertex, and thereby it guarantees the existence of a connected layout of $(\mathbf{H}^{+F}, \mathsf{R})$.

Suppose that t represents a parallel composition $\mathbf{G}_t = \mathbf{G}'_1 \oplus \mathbf{G}'_2$. The children t_1 and t_2 of t respectively represent the 2-terminal graphs $\mathbf{G}'_1 = (G'_1, (x_t, y_t))$ and $\mathbf{G}'_2 = (G'_2, (x_t, y_t))$. Then $\mathsf{DP}_t[\,\cdot\,]$ is computed as follows:

- By Lemma 1, $\mathsf{DP}_t[0] = \max\{\mathsf{DP}_{t_1}[0], \mathsf{DP}_{t_2}[0]\}$.
- By Lemma 2, we have for $k \in [c_{\mathrm{sp}} - 1]$:

$$\mathsf{DP}_t[k, x_t] = \min \left\{ \begin{array}{l} \max\{\mathsf{DP}_{t_1}[k, x_t, x_t y_t], \mathsf{DP}_{t_2}[0]\} \\ \\ \max\{\mathsf{DP}_{t_2}[k, x_t, x_t y_t], \mathsf{DP}_{t_1}[0]\} \end{array} \right\} \text{ and}$$

$$\mathsf{DP}_t[k, y_t] = \min \left\{ \begin{array}{l} \max\{\mathsf{DP}_{t_1}[k, y_t, x_t y_t], \mathsf{DP}_{t_2}[0]\} \\ \\ \max\{\mathsf{DP}_{t_2}[k, y_t, x_t y_t], \mathsf{DP}_{t_1}[0]\} \end{array} \right\}.$$

- By Lemma 3, we have for $k \in [c_{\mathrm{sp}} - 1]$: $\mathsf{DP}_t[k, x_t, x_t y_t] = \mathsf{DP}_t[k, x_t]$ and $\mathsf{DP}_t[k, y_t, x_t y_t] = \mathsf{DP}_t[k, y_t]$.

Suppose that t represents a series composition $\mathbf{G}_t = \mathbf{G}'_1 \otimes \mathbf{G}'_2$. The children t_1 and t_2 of t respectively represent the 2-terminal graphs $\mathbf{G}'_1 = (G'_1, (x_t, z))$ and $\mathbf{G}'_2 = (G'_2, (z, y_t))$. Then $\mathsf{DP}_t[\,\cdot\,]$ is computed as follows:

- By Lemma 4, we have for $k \in [c_{\mathrm{sp}} - 1]$:

$$\mathsf{DP}_t[0] = \min \left\{ \begin{array}{l} \max\{\mathsf{DP}_{t_1}[x_t], \mathsf{DP}_{t_2}[0]\} \\ \\ \max\{\mathsf{DP}_{t_2}[y_t], \mathsf{DP}_{t_1}[0]\} \end{array} \right\}.$$

- By Lemma 5, we have for $k \in [c_{\mathrm{sp}} - 1]$:

$$\mathsf{DP}_t[k, x_t] = \max\{\mathsf{DP}_{t_1}[k, x_t], \mathsf{DP}_{t_2}[k, z]\} \text{ and}$$

$$\mathsf{DP}_t[k, y_t] = \max\{\mathsf{DP}_{t_1}[k, y_t], \mathsf{DP}_{t_2}[k, z]\}.$$

- By Lemma 6, we have for $k \in [c_{\mathrm{sp}} - 2]$:

$$\mathsf{DP}_t[k, x_t, x_t y_t] = \max\{\mathsf{DP}_{t_1}[k, x_t, x_t z], \mathsf{DP}_{t_2}[k+1, z]\} \text{ and}$$

$$\mathsf{DP}_t[k, y_t, x_t y_t] = \max\{\mathsf{DP}_{t_1}[k, y_t, y_t z], \mathsf{DP}_{t_2}[k+1, z]\}.$$

For every non-leaf node t of $T(\mathbf{G})$, the entries of $\mathsf{DP}_t[\cdot]$ are initialized to some dummy value $\perp$. Every leaf node t of $T(\mathbf{G})$ represents the single edge graph, that is $G_t = (V_t, E_t)$ with $V = \{x_t, y_t\}$ and $E = \{x_t y_t\}$. Then we can initialize the values associated to a leaf node t as follows:

- $\mathsf{DP}_t[0] = \mathsf{ectvs}(\mathbf{G}_t^{+\emptyset}, \{x_t, y_t\}) = 1$
- for $k \in [c_{\mathrm{sp}} - 1]$, $\mathsf{DP}_t[k, x_t] = \mathsf{ectvs}(\tilde{\mathbf{G}}_{t,k}^{+F}, \{x_t, r_1, \ldots, r_k\}) = k + 1$ where $F = \{y_t r_i \mid i \in [k]\}$.
- for $k \in [c_{\mathrm{sp}} - 1]$, $\mathsf{DP}_t[k, y_t] = \mathsf{ectvs}(\tilde{\mathbf{G}}_{t,k}^{+F}, \{y_t, r_1, \ldots, r_k\}) = k + 1$ where $F = \{x_t r_i \mid i \in [k]\}$.
- for $k \in [c_{\mathrm{sp}} - 2]$, $\mathsf{DP}_t[k, x_t, x_t y_t] = \mathsf{ectvs}(\tilde{\mathbf{G}}_{t,k}^{+F}, \{x_t, r_1, \ldots, r_k\}) = k + 1$ where $F = \{y_t r_i \mid i \in [k]\} \cup \{x_t y_t\}$.
- for $k \in [c_{\mathrm{sp}} - 2]$, $\mathsf{DP}_t[k, y_t, x_t y_t] = \mathsf{ectvs}(\tilde{\mathbf{G}}_{t,k}^{+F}, \{y_t, r_1, \ldots, r_k\}) = k + 1$ where $F = \{x_t r_i \mid i \in [k]\} \cup \{x_t y_t\}$.

As the SP-tree $T(\mathbf{G})$ contains $O(n)$ nodes, filling the table $\mathsf{DP}_t[\cdot]$, for every node t, is achieved in $O(n \cdot \log n)$-time. Theorem 3 states that $\mathsf{ctw}(G) = \mathsf{ctvs}(G) = \min\{\mathsf{ectvs}(\mathbf{G}^{+\emptyset}, \{x, y\}) \mid xy \in E\}$. This implies that the whole algorithm runs in $O(n^2 \cdot \log n)$-time. □

3.3 Generalization to graph of treewidth at most two

Recall that a graph G has treewidth at most two if and only if every biconnected component of G induces a series-parallel graph. So we need a lemma to deal with cut vertices.

Lemma 7. *Let $G = (V, E)$ be a graph containing a cut vertex x and let $G_1 = [C_1 \cup \{x\}]$, …, $G_k = G[C_k \cup \{x\}]$ be the induced subgraphs where $C_1, \ldots, C_k$ denote the connected components of $G - x$. Then*

$$\mathsf{ctvs}(G) = \min_{i \in [k]} \Big\{ \max \Big\{ \mathsf{ctvs}(G_i), \max\{\mathsf{ctvs}(G_j, \{x\}) \mid j \in [k], j \neq i\} \Big\} \Big\} \quad \textit{(see Figure 10).}$$

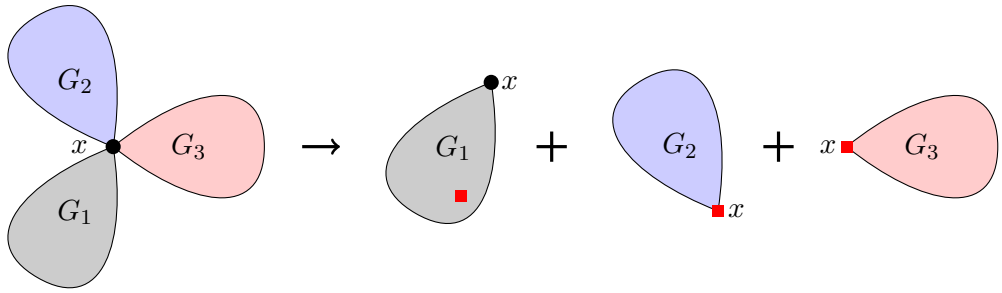

Figure 10: Decomposition of a graph with a cut vertex. If an optimal connected layout σ^* starts at an arbitrary vertex of $G_1 = G[C_1 \cup \{x\}]$, then $\sigma^*[C_2]$ and $\sigma^*[C_3]$ start at x, which becomes a root of $G_2 = G[C_2 \cup \{x\}]$ and of $G_3 = G[C_3 \cup \{x\}]$.

Proof. Let us consider $\sigma \in \mathcal{L}^{(c)}(G)$ and suppose that the first vertex of σ belongs to C_1. Then observe that σ can be rearranged into $\tau = \sigma[C_1] \odot \sigma[C_2 \setminus \{x\}] \ldots, \odot\sigma[C_k \setminus \{x\}]$ and that since x is a cut vertex then $\tau \in \mathcal{L}^{(c)}(G)$ as well. The statement follows from the observation that $\mathsf{cost}(G, \tau) = \max\{\mathsf{cost}(G_i, \sigma[C_i]) \mid i \in [k]\}$. □

Theorem 6. *Computing the connected treewidth of a graph of treewidth at most* 2 *requires* $O(n^3 \cdot \log n)$*-time.*

Proof. Let G be a graph of treewidth at most 2. The algorithm first computes the biconnected tree decomposition of G. This can be done in linear time. Following Lemma 7, we guess a biconnected component C_1 in which the connected layout will start. This generates for every biconnected component C_k distinct from C_1 a root vertex r_k. Then using Proposition 1, in $O(n^2 \cdot \log n)$ we can compute $\mathsf{ctvs}(G[C_1])$ and $\mathsf{ctvs}(G[C_k], \{r_k\})$ for each $k \neq 1$. This leads to an $O(n^3 \cdot \log n)$-time algorithm. □

4 Discussion and open problems

We obtained a polynomial time algorithm to compute the connected treewidth for the class of graphs of treewidth at most two. This result naturally leads to the problem of determining the algorithmic complexity of computing the connected treewidth for the class of bounded treewidth graphs. To discuss this, we present the problem as a decision problem:

Connected Treewidth
Input: A graph G and an integer k.
Question: $\mathsf{ctw}(G) \leq k$?

Our result implies that Connected Treewidth can be solved in $O(n^3 \cdot k)$-time for graphs of treewidth at most 2. Let us discuss the following three conjectures.

Conjecture 1. Connected Treewidth *can be solved by an* $O(n^{f(\mathsf{tw}(G))})$*-time algorithm.*

Conjecture 2. Connected Treewidth *can be solved by an* $O(f(k, \mathsf{tw}(G)) \cdot n)$*-time algorithm.*

Conjecture 3. Connected Treewidth *can be solved by an* $O(n^{f(k)})$*-time algorithm.*

Our result can be seen as a special case of Conjecture 1 (when $\mathsf{tw}(G) \leq 2$). A general resolution of Conjecture 1 would require a vast extension of our dynamic programming approach. In our approach (for treewidth at most two) we essentially solve a slightly modified problem where the input is a pair (G, e), where $e \in E(G)$, and we return the minimum cost of a layout that *begins* with the endpoints of e. Then we reduce the computation of connected treewidth to this problem by paying an overhead of $O(n^2)$. An interesting question is whether and how a similar approach might work for the general case. Of course, one may try to reduce the $O(n^3 \cdot \log n)$-time complexity of our algorithm by avoiding such reductions and directly build a dynamic programming scheme for Connected Treewidth on graphs of treewidth $\leq k$. We believe that this is possible and can reduce the time complexity to $O(n^c)$ for some $1 < c \leq 3$.

For Conjecture 2 one may attempt to use tools related to Courcelle's theorem. This would require to express the question $\mathsf{ctw}(G) \leq k$ in using a formula ϕ_k in Monadic Second Order Logic (MSOL) which is far from being obvious. A possible direction would be to consider the contraction-obstruction set $\mathcal{Z}_k$ of the class $\mathcal{G}_k = \{G \mid \mathsf{ctw}(G) \leq k\}$, i.e., the contraction minimal graphs not in $\mathcal{G}_k$. Indeed contraction testing is MSOL expressible. However, it turns out that, unlike the case for treewidth and pathwidth with respect to minors, $\mathcal{Z}_k$ is infinite for every $k \geq 2$, as observed in [1]. A possible way to overcome this obstacle is to consider some other partial ordering relation, alternative to contractions, that maintains closeness, MSOL expressibility and gives rise to bounded size obstructions. Such a step can be done using the results of [1] for graphs of connected treewidth at most two. However, it is not clear whether this can be extended for bigger values of connected treewidth. Let us mention that recently, Kanté et al. [27] obtained a $O(f(k) \cdot n)$-time algorithm to compute the connected pathwidth of a graph (that is the analogue of Conjecture 2 for pathwidth). Finally, it is also natural to ask if one can compute in FPT-time the connected treewidth of a graph when parameterized by the pathwidth of the input graph.

A proof of Conjecture 3 would follow if we devise an algorithm to check whether $\mathsf{ctw}(G) \leq k$ in $O(n^{f(k,\mathsf{tw}(G))})$ time. This follows directly from the fact that yes-instances of Connected Treewidth have always treewidth at most k. Such a result would be analogous to the one of [16] for connected pathwidth and is perhaps the first (and easier) to be attacked among the three above conjectures.

References

[1] I. Adler, C. Paul, and D. Thilikos. Connected search for a lazy robber. In *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, volume 150 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 7:1–7:14, 2019.

[2] S. Arnborg. Efficient algorithms for combinatorial problems on graphs with bounded decomposability — a survey. *BIT Numerical Mathematics*, 25:1–23, 1985.

[3] S. Arnborg, D. Corneil, and A. Proskurowski. Complexity of finding embeddings in a k-tree. *SIAM Journal on Algebraic Discrete Methods*, 8:277–284, 1987.

[4] L. Barrière, P. Flocchini, F. Fomin, P. Fraigniaud, N. Nisse, N. Santoro, and D. Thilikos. Connected graph searching. *Information and Computation*, 219:1–16, 2012.

[5] L. Barrière, P. Fraigniaud, N. Santoro, and D. Thilikos. Searching is not jumping. In *Internation Workshop on Graph Theoretical Concepts in Computer Science (WG)*, volume 2880 of *Lecture Notes in Computer Science*, pages 34–45, 2003.

[6] M. Bern, E. Lawler, and A. Wong. Why certain subgraph computations require only linear time. In *Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 117–125, 1985.

[7] M. Bern, E. Lawler, and A. Wong. Linear time computation of optimal subgraphs of decomposable graphs. *Journal of Algorithms*, 8:216–235, 1987.

[8] U. Bertelé and F. Brioschi. *Nonserial dynamic programming*. Academic Press, 1972.

[9] D. Bienstock and P. Seymour. Monotonicity in graph searching. *Journal of Algorithms*, 12(2):239–245, 1991.

[10] H. Bodlaender and A. Koster. Combinatorial optimization on graphs of bounded treewidth. *The Computer Journal*, 51(3):255–269, 2008.

[11] R. Borie, G. Parker, and C. Tovey. Automatic generation of linear-time algorithms from predicate calculus descriptions of problems on recursively constructed graph families. *Algorithmica*, 7:555–581, 1992.

[12] R. Breisch. An intuitive approach to speleotopology. *Southwestern Cavers (A publication of the Southwestern Region of the National Speleological Society)*, 6(5):72–78, 1967.

[13] B. Courcelle. The monadic second-order logic of graphs III: tree-decompositions, minor and complexity issues. *Informatique Théorique et Applications*, 26:257–286, 1992.

[14] N. Dendris, L. Kirousis, and D. Thilikos. Fugitive-search games on graphs and related parameters. *Theoretical Computer Science*, 172(1-2):233–254, 1997.

[15] D. Dereniowski. From pathwidth to connected pathwidth. *SIAM Journal on Discrete Mathematics*, 26(4):1709–1732, 2012.

[16] D. Dereniowski, D. Osula, and P. Rzazewski. Finding small-width connected path decomposition in polynomial time. *Theoretical Computer Science*, 794:85–100, 2019.

[17] R. Diestel. *Graph theory*. Springer-Verlag, 2005.

[18] R. Downey and M. Fellows. *Parameterized complexity*. Springer, 1999.

[19] R. Duffin. Topology of series-paralle networks. *Journal of Mathematical Analysis and Applications*, 10:303–318, 1965.

[20] D. Eppstein. Parallel recognition of series-parallel graphs. *Information and Computation*, 98(1):41–55, 1992.

[21] P. Fraigniaud and N. Nisse. Connected treewidth and connected graph searching. In *Latin American Theoretical Informatics Symposium (LATIN)*, number 3887 in Lecture Notes in Computer Science, pages 479–490, 2006.

[22] P. Fraigniaud and N. Nisse. Monotony properties of connected visible graph searching. In *Internation Workshop on Graph Theoretical Concepts in Computer Science (WG)*, volume 4271 of *Lecture Notes in Computer Science*, pages 229–240, 2006.

[23] P. Fraigniaud and N. Nisse. Monotony properties of connected visible graph searching. *Information and Computation*, 206:1383–1393, 2008.

[24] R. Halin. s-functions for graphs. *Journal of Geometry*, 8(1-2):171–186, 1976.

[25] R. Hassin and A. Tamir. Efficient algorithms for optimization and selection on series-parallel graphs. *SIAM Journal of Algebraic and Discrete Methods*, 7(3):379–389, 1986.

[26] P. Hlineny, S. Oum, D. Seese, and G. Gotlob. Width parameters beyond tree-width and their applications. *The Computer Journal*, 51(3):326–362, 2008.

[27] M. M. Kanté, C. Paul, and D. M. Thilikos. A linear fixed parameter tractable algorithm for connected pathwidth, 2020. arXiv 2004.11937.

[28] N. Kinnersley. The vertex separation number of a graph equals its path-width. *Information Processing Letters*, 42(6):345–350, 1992.

[29] L. Kirousis and C. Papadimitriou. Interval graphs and searching. 55(2):181–184, 1985. *Discrete Mathematics*, 55(2):181–184, 1985.

[30] L. Kirousis and C. Papadimitriou. Searching and pebbling. *Theoretical Computer Science*, 47(2):205–212, 1986.

[31] P. Macmahon. The combinations of resistances. *The Electrician*, 1892. re-edited in *Discrete Applied Mathematics*, 54:225-228, 1994.

[32] R. Möhring. Graph problems related to gate matrix layout and PLA folding. In *Computational graph theory*, volume 7 of *Comput. Suppl.*, pages 17–51. Springer, 1990.

[33] T. D. Parsons. Pursuit-evasion in a graph. In *Theory and applications of graphs*, volume 642 of *Lecture Notes in Mathematics*, pages 426–441, 1978.

[34] J. Riordan and C. Shannon. The number of two-terminal series-parallel networks. *Journal of Mathematics and Physics*, 21:83–93, 1942.

[35] N. Robertson and P. Seymour. Graph minors III. planar tree-width. *Journal of Combinatorial Theory Series B*, 36:49–64, 1984.

[36] P. Seymour and R. Thomas. Graph searching and min-max theorem for tree-width. *Journal of Combinatorial Theory Series B*, 58:22–33, 1993.

[37] J. Valdes, R. Tarjan, and E. Lawler. The recognition of series-parallel digraphs. *SIAM Journal on Computing*, 11:298–313, 1982.