Intervalizing k-Colored Graphs*

Hans L. Bodlaender† Babette de Fluiter‡
Department of Computer Science, Utrecht University,
P.O. Box 80.089, 3508 TB Utrecht, the Netherlands

Abstract

The problem to determine whether a given k-colored graph is a subgraph of a properly k-colored interval graph is shown to be solvable in $O(n)$ time when $k = 2$, solvable in $O(n^2)$ time when $k = 3$, and to be NP-complete for any fixed $k \geq 4$. This problem has an application in DNA physical mapping. Our algorithm for $k = 3$ is based on an extensive analysis of the precise structure of graphs of pathwidth two, dynamic programming on certain parts of the input graph, and a careful combination of the results for the different parts.

*This research was partially supported by the Foundation for Computer Science (S.I.O.N) of the Netherlands Organization for Scientific Research (N.W.O.) and partially by the ESPRIT Basic Research Actions of the EC under contract 7141 (project ALCOM II).

†Email: hansb@cs.ruu.nl

‡Email: babette@cs.ruu.nl

1 Introduction

In this paper, we consider the following problem.

> INTERVALIZING COLORED GRAPHS [ICG]
> **Instance:** A graph $G = (V, E)$, a coloring $c : V \to \{1,..., k\}$
> **Question:** Is there a properly colored supergraph $G' = (V, E')$ of G which is an interval graph?

The problem models a problem arising in sequence reconstruction, which appears in some investigations in molecular biology (such as protein sequencing, nucleotide sequencing and gene sequencing (see [FHW93]). A sequence X (usually a large piece of DNA) is fragmented (or k copies of the sequence X are fragmented). For each fragment, a set of characteristics (its 'fingerprint' or 'signature') is determined, and based on respective fingerprints, an 'overlap' measure is computed. Using this overlap information, the fragments are assembled into islands of contiguous fragments (contigs). Instances of ICG model the situation where k copies of X are fragmented, and some fragments (clones) are known to overlap. Fragments of the same copy of X will not overlap. Now each vertex in V represents one fragment; the color of a vertex represents to which copy of X the fragment belongs. It can be seen that ICG helps here to predict other overlaps and to work towards reconstruction of the sequence X.

It is known that ICG for an arbitrary number of colors is NP-complete [FHW93]. However, from the application it appears that the cases where the number of colors k (= the number of copies of X that are fragmented) is some small given constant are of interest. In this paper, we resolve the complexity of this problem for all constant values k. We observe that the case $k = 2$ is easy to resolve in linear time. We show that the case $k = 3$ is solvable in $O(n^2)$ time. Finally, we show that ICG is NP-complete for four colors (and hence, for any fixed number of colors ≥ 4.)

In [FHW93], Fellows et al. consider ICG with a bounded number of colors. They show that, although for fixed $k \geq 3$, yes-instances have bounded pathwidth (and hence bounded treewidth), standard methods for graphs with bounded treewidth will be insufficient to solve ICG, as the problem is 'not finite state'. Also, they show ICG to be hard for the complexity class $W[1]$, (which was strengthened in [BFH94] to hardness for all classes $W[t]$, $t \in \mathbf{N}$). This result implies that it is unlikely that there exists a c, such that for any fixed number of colors k, ICG is solvable in time $O(f(k)n^c)$. Clearly, our NP-completeness result implies the fixed parameter intractability results, but is much stronger.

ICG is closely related to TRIANGULATING COLORED GRAPHS (TCG) where we look for a properly colored *triangulated* supergraph G' of a k-colored input graph G (i.e., G' does not contain a chordless cycle of length at least four). This problem is known to be NP-complete [BFW92], solvable in $O(n^{k+1})$ time for fixed k [MWW94], and solvable in linear time for the cases $k = 2$ and $k = 3$ [BK93, IS93, KW92, NON94]. Despite the close relationship between ICG and TCG, it appears that ICG poses some additional difficulties which require more complex and time consuming algorithms. For instance, while there is an easy characterization which assures that three-colored simple cycles can be triangulated without adding edges between vertices of the same color, for ICG on three-colored simple cycles, such a simple characterization does not exist, and even this case seems to require an $O(n^2)$ algorithm, based on dynamic programming. Additionally, TCG with three colors is 'finite state', while ICG with three colors is not.

Another closely related problem is COLORED PROPER INTERVAL GRAPH COMPLETION, which asks whether a given colored graph is a subgraph of a properly colored unit interval graph. In [KS93, KST94], it is shown that this problem is NP-complete, polynomial for a fixed number of colors, and hard for $W[1]$.

A necessary condition for a three-colored graph G to be 'intervalizable' is that the pathwidth of G is at most two [FHW93]. Our algorithm exploits the precise structure of graphs of pathwidth two (partial two-paths). For parts of the input graphs, a dynamic programming approach is used to compute whether these parts can be intervalized, and some more information. Then, a careful case analysis is necessary to see whether all the different parts can be put together to an intervalization of the entire input graph. In Section 3 we analyze the structure of partial two-paths. We do this first for biconnected partial two-paths, after that for trees of pathwidth two, and finally for general partial two-paths. In Section 4 we consider the algorithms, again first for biconnected graphs, then for trees, and finally, we discuss how information for biconnected and tree-parts of the graph can be pieced together. In Section 5 we discuss our NP-completeness result.

2 Preliminaries

A graph G is a pair (V, E), where V is the set of vertices, and E is the set of edges. An edge is a set of two distinct vertices. The vertices and edges of a graph G are also denoted by $V(G)$ and $E(G)$, respectively.

Let G be a graph, $V' \subseteq V(G)$. The subgraph of G *induced* by V' is denoted by $G[V']$ and is defined as follows. $V(G[V']) = V'$ and $E(G[V']) = \{ e \in E(G) \mid e \subseteq V' \}$.

A *path* P in G is a sequence $(v_1,\ldots, v_s)$ of distinct vertices of G, such that there exists an edge between each pair of consecutive vertices. Vertices v_1 and v_s are the *end points* of P, vertices v_i, $1 < i < s$, are the *inner vertices* of P.

A *cycle* is a graph C which consists of a path P containing all vertices of C, and an edge between the first and the last vertex of the path.

A *chordless* cycle C in G is a subgraph of G which is a cycle in which each two vertices which are not adjacent in C are also not adjacent in G.

A *biconnected* graph is a graph which remains connected if an arbitrary vertex is removed. A *biconnected component* B of a graph G is an induced subgraph of G which is biconnected and which is not a proper subgraph of another induced subgraph of G for which this holds. We only consider biconnected graphs and biconnected components which are non-trivial, i.e. which have at least three vertices.

A *tree* is a connected graph which contains no cycles. We usually denote trees by H instead of G.

An *interval graph* is a graph $G = (V, E)$ for which there is a function $\Phi : V \to \mathcal{I}$, where $\mathcal{I}$ is the set of all intervals on the real line, such that for each pair $v, w \in V$, $\Phi(v) \cap \Phi(w) \neq \emptyset \Leftrightarrow \{v, w\} \in E$. A k-coloring of a graph $G = (V, E)$ is a surjection $c : V \to \{1,\ldots, k\}$. A *proper* k-coloring is a k-coloring c such that for each edge $\{v, w\} \in E$, $c(v) \neq c(w)$. An *intervalization* of a graph $G = (V, E)$ with a k-coloring c, is a supergraph $G' = (V, E')$ $(E \subseteq E')$ of G which is an interval graph and is properly colored by c.

A *path decomposition* PD of a graph $G = (V, E)$ is a sequence $(V_1,\ldots, V_t)$, in which for all i, $V_i \subseteq V$ and V_i is non-empty, and the following conditions are satisfied:
1. For each $v \in V$, there is an i such that $v \in V_i$.
2. For each $e \in E$, there is an i such that $e \subseteq V_i$.
3. For each $i \leq j \leq l$, $V_i \cap V_l \subseteq V_j$.

The sets V_i are called the *nodes* of the path decomposition. The *width* of PD is $\max_i |V_i| - 1$. A graph G has *pathwidth* k if there is path decomposition of width k of G, but there is no path decomposition of width $k - 1$ of G. A graph G is called a *partial k-path* if it has pathwidth at most k.

Let G be a graph, $PD = (V_1,\ldots, V_t)$ a path decomposition of G. Let G' be a subgraph of G. The *occurrence* of G' in PD is a subsequence $(V_j,\ldots, V_{j'})$ of PD in which V_j and $V_{j'}$ contain an edge of G', and no node V_i, with $i < j$ or $i > j'$ contains an edge of G', i.e. $(V_j,\ldots, V_{j'})$ is the shortest subsequence of PD that contains all nodes of PD which contain an edge of G'. We say that G' *occurs* in $(V_j,\ldots, V_{j'})$. The vertices of G' occur in $(V_l,\ldots, V_{l'})$ if these are the only nodes in PD containing vertices of G'. A vertex v (edge e) is an *end vertex* (*end edge*) of G' if in each path decomposition of width two of G, v (e) occurs in the left or right end node of the occurrence of G'. A vertex v is a *double end vertex* of G' if in each path decomposition of width two of G, v occurs in both end nodes of the occurrence of G'. Similar for edges. A vertex v is a *middle vertex* of G' if in each path decomposition of G in which G' occurs in $(V_j, \ldots, V_{j'})$, either $v \in V_j$ or $v \in V_{j'}$ or there is an i, $j \leq i \leq j'$, such that $V_i \cap V(G') = \{v\}$. An edge $e \in E'$ is a *middle edge* of G' if in each path decomposition $PD = (V_1, \ldots, V_t)$ of width two of G in which G' occurs in $(V_j, \ldots, V_{j'})$, either $e \subseteq V_j$ or $e \subseteq V_{j'}$ or there is an i, $j \leq i \leq j'$, such that either $V_i \cap V(G') = e$ or $PD' = (V_1, \ldots, V_i, V_{i'}, V_{i+1}, \ldots, V_t)$ is a path decomposition of G and $V_{i'} \cap V(G') = e$.

Let G be a graph, $PD = (V_1, \ldots, V_t)$ a path decomposition of G. Let $1 \leq j \leq t$. We say that a node V_i is on the *left side* of V_j if $i < j$, and on the *right side* of V_j if $i > j$. Let G' be a connected subgraph of G, suppose G' occurs in $(V_l, \ldots, V_{l'})$. We say that G' occurs on the left side of V_j if $l' < j$, and on the right side of V_j if $l > j$. In the same way, we speak about the left and right sides of a sequence $(V_j, \ldots, V_{j'})$, i.e. a node is on the left side of $(V_j, \ldots, V_{j'})$ if it is on the left side of V_j, and a node is on the right side of $(V_j, \ldots, V_{j'})$ if it is on the right side of $V_{j'}$.

Let G be a graph, $PD = (V_1, \ldots, V_t)$ a path decomposition of G, $V' \subseteq V$ and suppose $G[V']$ occurs in $(V_j, \ldots, V_{j'})$, $1 \leq j \leq j' \leq t$. The path decomposition of $G[V']$ induced by PD is denoted by $PD[V']$ and is obtained from the sequence $PD[V'] = (V_j \cap V', \ldots, V_{j'} \cap V')$ by deleting all empty nodes and all nodes $V_i \cap V'$, $j \leq i < j'$, for which $V_i \cap V' = V_{i+1} \cap V'$.

The *reversed path decomposition* of PD is denoted as $\mathrm{rev}(PD)$ and is defined as follows.

$$\mathrm{rev}(PD) = (V_t, V_{t-1}, \ldots, V_1)$$

Let $PD' = (W_1, \ldots, W_{t'})$ be another path decomposition. The *concatenation* of PD and PD' is denoted by $PD +\!\!\!+ PD'$ and is defined as follows.

$$PD +\!\!\!+ PD' = (V_1, \ldots, V_t, W_1, \ldots, W_{t'})$$

Lemma 2.1. *Let $G = (V, E)$ be a graph, $PD = (V_1, \ldots, V_t)$ a path decomposition of G. Let $G' = (V, E')$ be a supergraph of G with*

$$E' = \{ \{v, v'\} \mid \exists_{1 \leq i \leq t}\; v, v' \in V_i \}.$$

The graph G' is an interval graph.

Proof. Let $\Phi : V \to \{1, \ldots, n\}$ be defined as follows. For each $v \in V$, if v occurs in nodes $(V_j, \ldots, V_l)$, then $\Phi(v) = [j, l]$. Then $\{v, v'\} \in E'$ if and only if $\Phi(v)$ and $\Phi(v')$ overlap. $\square$

The graph G' is called the *interval completion* of G for PD.

A path decomposition $PD = (V_1, \ldots, V_t)$ of a graph G which is k-colored is called a *proper path decomposition* if for each node V_i and each pair $v, w \in V_i$, if $v \neq w$ then $c(v) \neq c(w)$.

Lemma 2.2. *Let $G = (V, E)$ be a graph, $c : V \to \{1,...,k\}$ a k-coloring of G. G has an intervalization if and only if there is a proper path decomposition of G, which has width $k-1$ at most.*

Proof. (See also [FHW93].) For the 'if' part, suppose $PD = (V_1,...,V_t)$ is a proper path decomposition of G. Note that PD has width $k-1$. Then the interval completion of G for PD is a properly k-colored interval graph.

For the 'only if' part, suppose $G' = (V, E')$ is an intervalization of G. Let $\Phi : V \to \mathcal{I}$ be a function for G' such that for each $v, w \in V$, $v \neq w$, $\{v, w\} \in E \Leftrightarrow \Phi(v) \cap \Phi(w) \neq \emptyset$. Let $(u_1,...,u_n)$, $n = |V|$, be an ordering of V in such a way that for all i, j with $1 \leq i < j \leq n$, $\Phi(u_i)$ starts on the left side of or at the same point as $\Phi(u_j)$. For each i let $V_i = \{ v \in V \mid \Phi(v) \cap \Phi(u_i) \neq \emptyset \}$. Then $PD = (V_1,...,V_n)$ is a proper path decomposition of G' and hence of G. Furthermore, each node contains at most k vertices, since there are at most k vertices with different colors. Hence PD has pathwidth $k-1$ at most. $\square$

Thus, the following problem is equivalent to ICG.

> **Instance:** A graph $G = (V, E)$, a k-coloring $c : V \to \{1,...,k\}$
> **Question:** Is there a proper path decomposition of G?

In this paper, we use both problems. Note that the proof of Lemma 2.2 also gives an easy way to transform a solution for one problem into a solution for the other problem.

For the case that $k = 2$, the question whether there is a proper path decomposition of G is equal to the question whether G is a properly colored partial one-path (see also [FHW93]). This is because if G is properly colored, then we can transform each path decomposition of width one of G into a proper path decomposition of width one by simply deleting all nodes which contain no edge, and then adding a node at the right side of the path decomposition for each isolated vertex containing this vertex only. Checking whether a graph has pathwidth one can be done in linear time, and checking whether it is properly colored also.

Theorem 2.1. *For $k = 2$, ICG can be solved in linear time.*

We now give some lemmas, which are frequently used in the remainder of this report.

The following two lemmas are well-known.

Lemma 2.3. *Let $(V_1,...,V_r)$ be a path-decomposition of $G = (V, E)$. Suppose $i < j < k$, and suppose P is a path from $v \in V$ to $w \in V$, $v \in V_i$, $w \in V_k$. Then V_j contains at least one vertex from P.*

Proof. Follows from the definition of path decompositions by induction on the length of the path. $\square$

The following Lemma is proved in e.g. [BM93].

Lemma 2.4. *(Clique Containment) Let $G = (V, E)$ be a graph, $PD = (V_1,\dots, V_t)$, a path decomposition of G, suppose $V' \subseteq V$ forms a clique in G. There is an i, $1 \le i \le t$, such that $V' \subseteq V_i$.*

Proof. We prove this by induction on $|V'|$. If $|V'| = 2$, then there is a V_i containing V' by definition. Suppose $|V'| > 2$. Let $v \in V'$. There is a node V_i, such that $V' - \{v\} \subseteq V_i$. Suppose v occurs in $(V_j,\dots, V_{j'})$. Suppose w.l.o.g. that $i \le j'$. If $i \ge j$, then clearly $V' \subseteq V_i$. If $i < j$, then for each $w \in V'$, there is an l, $j \le l \le j'$, such that $w \in V_l$. Hence $V' \subseteq V_j$, which gives a contradiction. □

Lemma 2.5. *Let G be a connected partial k-path, $k \ge 1$, and $V' \subseteq V$ such that $G[V']$ is connected. At most two of the connected components of $G[V - V']$ have pathwidth k.*

Proof. Suppose there are three components G_1, G_2 and G_3 of $G[V - V']$ which have pathwidth k. Let $PD = (V_1,\dots, V_t)$ be a path decomposition of G of width k. Suppose G_i, $i = 1, 2, 3$, occurs in $(V_{j_i},\dots, V_{l_i})$, and $j_1 \le j_2 \le j_3$. Then $l_1 \le l_2$, since otherwise, each V_i, $j_2 \le i \le l_2$, contains a vertex of G_1, which is not possible because G_2 has pathwidth k. Analogously, $l_2 \le l_3$. However, $G' = G[V(G_1) \cup V(G_3) \cup V']$ is a connected subgraph of G which has no vertices in common with G_2. Hence each V_i, $j_1 \le i \le l_3$, contains at least one vertex G'. But $j_1 \le j_2 \le l_2 \le l_3$, and G_2 has pathwidth k, which gives a contradiction. □

Lemma 2.6. *Let $G = (V, E)$ be a connected partial two-path, $V' \subseteq V$. Let $PD = (V_1,\dots, V_t)$ be a path decomposition of width two of G such that the vertices of V' occur in $(V_j,\dots, V_{j'})$. On each side of $(V_j,\dots, V_{j'})$, edges of at most two components of $G[V - V']$ occur.*

Proof. Suppose there are edges of at least three components of $G[V - V']$ on the left side of V_j. Let G_1, G_2, G_3 be three of these components. Let V_l, $1 \le l < j$, be the rightmost node on the left side of V_j containing an edge of one of the components G_1, G_2 and G_3, say G_1. V_l contains a vertex of G_2 and of G_3. Hence $|V_l| = 4$. □

3 The Structure of Partial Two-Paths

In this section, we first give a characterization of biconnected partial two-paths. After that, we give a characterization of trees of pathwidth two, and finally of partial two-paths in general.

3.1 Biconnected Partial Two-Paths

Given a graph $G = (V, E)$, the graph $\bar{G}$ which is obtained from G by adding all edges $\{v, w\} \notin E$ such that there are three disjoint paths from v to w in G is called the *cell completion* of G. (Two paths from v to w are disjoint if they only have vertices v and w in common.) The following lemma has been proved in [BK93] in the setting of partial two-trees.

Lemma 3.1. *Let G be a partial two-path. The cell-completion $\bar{G}$ of G is a subgraph of any intervalization of G of pathwidth at most two.*

In terms of path decomposition, the lemma says that each path decomposition of width two of a partial two-path G is a path decomposition of the cell-completion $\bar{G}$. The cell completion of a partial two-path can be found in linear time [BK93]. In the cell completion of a graph, each two distinct chordless cycles have at most one edge in common. In [BK93], it has been shown, that the cell completion of a biconnected partial two-tree is a tree of chordless cycles. We will show that the cell completion of a biconnected partial two-path is a path of chordless cycles. Before we prove this, we first give a definition and prove a number of lemmas.

Definition 3.1. (Path of Chordless Cycles). *A* path of chordless cycles *is a pair $(\mathcal{C}, \mathcal{S})$, where $\mathcal{C}$ is a sequence $(C_1, \ldots, C_p)$ of chordless cycles, $p \geq 1$, and $\mathcal{S}$ is a sequence $(e_1, \ldots, e_{p-1})$ of edges, such that for each i, $1 \leq i < p$, $V(C_i) \cap V(C_{i+1}) = e_i$, $E(C_i) \cap E(C_{i+1}) = \{e_i\}$ and for each i, $1 \leq i < p-1$, if $e_i = e_{i+1}$, then $|V(C_{i+1})| = 3$.*

In Figure 1, an example of a path of chordless cycles is given with six chordless cycles.

Lemma 3.2. *Let G be a biconnected partial two-path, C a cycle of $\bar{G}$, and $PD = (V_1, \ldots, V_t)$ a path decomposition of G of width two. Suppose C occurs in $(V_j, \ldots, V_{j'})$, and $\{x, y\}$ is an edge of C occurring in V_j, $\{x', y'\}$ an edge occurring in $V_{j'}$. The following holds.*

1. *If $|V(C)| > 3$, then $\{x, y\} \neq \{x', y'\}$.*
2. *For each i, $j \leq i \leq j'$, $|V_i \cap V(C)| \geq 2$ and for each edge $e \in E(C)$ there is an i, $j \leq i \leq j'$, such that $e \subseteq V_i$ and $|V_i \cap V(C)| = 3$.*

Proof. 1. Suppose $x = x'$, $y = y'$. Because $|V(C)| > 3$, there is an edge $\{v, w\}$ in C with $\{v, w\} \cap \{x, y\} = \emptyset$. There must be a V_i, $j \leq i \leq j'$, with $v, w, x, y \in V_i$, hence $|V_i| \geq 4$.

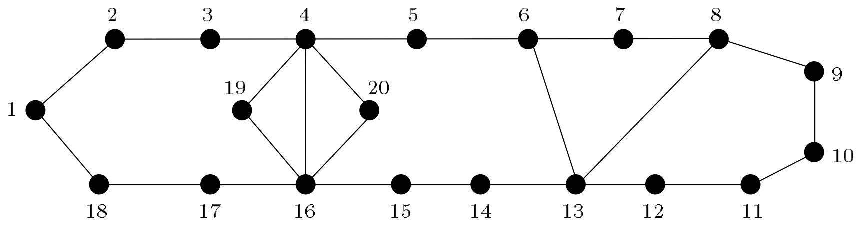

Figure 1: A path of chordless cycles $(\mathcal{C}, \mathcal{S})$ with $\mathcal{C} = (C_1, ..., C_6)$, $\mathcal{S} = (e_1, ..., e_5)$. $V(C_1) = \{1, 2, 3, 4, 16, 17, 18\}$, $V(C_2) = \{4, 16, 19\}$, $V(C_3) = \{4, 16, 20\}$, $V(C_4) = \{4, 5, 6, 13, 14, 15, 16\}$, $V(C_5) = \{6, 7, 8, 13\}$ and $V(C_6) = \{8, 9, 10, 11, 12, 13\}$. Furthermore, $e_1 = e_2 = e_3 = \{4, 16\}$, $e_4 = \{6, 13\}$ and $e_5 = \{8, 13\}$.

2. Suppose w.l.o.g. that x and x' are connected by a path in C which does not contain y or y'. Denote this path by P_1. Denote the path between y and y' not containing x or x' by P_2. See also Figure 2. The part of the path decomposition containing vertices of P_1 must be connected, according to Lemma 2.3, hence each V_i, $j \leq i \leq j'$, contains a vertex of P_1. Analogously, each V_i contains a vertex of P_2. Since P_1 and P_2 are vertex disjoint, $|V_i \cap V(C)| \geq 2$ for each i, $j \leq i \leq j'$. Suppose P_1 contains at least one edge. Let e be an edge of P_1. Let V_l, $j \leq l \leq j'$ such that $e \subseteq V_l$. This V_l also contains a vertex of P_2, hence there is an i such that $e \subseteq V_i$ and $|V_i \cap V(C)| \geq 3$ for each edge e on P_1 and P_2. Now consider edge $\{x, y\} \subseteq V_j$. If there is another vertex of C in V_j, then the lemma holds for $\{x, y\}$. If $V_j \cap V(C) = \{x, y\}$, then there must be an i, $j \leq i \leq j'$, such that $\{x, y\} \subseteq V_i$ and V_i contains a neighbor of x or y. Hence $|V_i \cap V(C)| = 3$. Similar for edge $\{x', y'\}$. $\square$

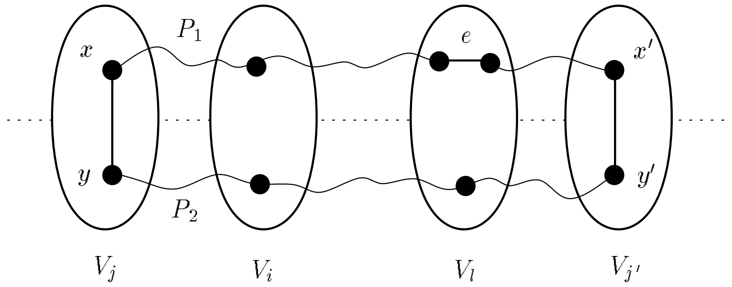

Figure 2: The occurrence of chordless cycle C as in part 2 of the proof of Lemma 3.2.

Let G be a biconnected partial two-path. The lemma implies that the occurrences of two chordless cycles of $\bar{G}$ which do not have a vertex in common can not overlap in any path decomposition of width two of G. If two chordless cycles have one edge in common, then the occurrences of these two cycles can only overlap in their common edge, as we show in the next lemma.

Lemma 3.3. *Let G be a biconnected partial two-path with cycles C and C' which have one edge $\{x, y\}$ and no other vertices in common. Let $PD = (V_1, ..., V_t)$ be a path decomposition of G of pathwidth two. Suppose C occurs in $(V_j, ..., V_{j'})$, C' occurs in $(V_l, ..., V_{l'})$. Then the following holds.*

1. *$j \leq l$ and $j' \leq l'$ or $j \geq l$ and $j' \geq l'$. If $j = l$ and $j' = l'$, then $|V(C)| = |V(C')| = 3$.*

2. *If $j \leq l$, $j' \leq l'$, then $j' \geq l$, $\{x, y\}$ is an end edge of C and of C' and it occurs in $V_{j'}$ and in V_l, and there is an i, $l \leq i < j'$, such that $V(C) \cap (V_{i+1} \cup ... \cup V_t) = \{x, y\}$ and $V(C') \cap (V_1 \cup ... \cup V_i) = \{x, y\}$ (or possibly vice versa, if $j = l$ and $j' = l'$), so $\{x, y\}$ is a middle edge of $C \cup C'$, and an end edge of C and of C'.*

Proof. 1. Suppose $j < l$ and $j' > l'$, then $|V(C')| = 3$, say $V(C') = \{x, y, z\}$, since each of $V_j, ..., V_{j'}$ contains two vertices of C. Let $j < i < j'$, such that $V_i = \{x, y, z\}$. Suppose $\{a, b\}, \{c, d\} \in E(C)$ and $\{a, b\} \subseteq V_j$, $\{c, d\} \subseteq V_{j'}$, such that there is a path from a to c not containing b or d. Let P_1 denote this path, and P_2 denote the path from b to d not containing a and c. $\{a, b\} \neq \{x, y\}$ and $\{c, d\} \neq \{x, y\}$, so suppose $\{x, y\} \in E(P_1)$. V_i contains a vertex of P_2, which is not x, y or z. Hence $|V_i| \geq 4$, which is a contradiction. So either $j \leq l$ and $j' \leq l'$ or $j \geq l$ and $j' \geq l'$. If $j = l$ and $j' = l'$, then $|V(C)| = |V(C')| = 3$, since each V_i, $j \leq i \leq j'$, contains two vertices of C and two vertices of C'.

2. It is clear that $j' \geq l$, since $\{x, y\}$ is an edge of both C and C'. There are nodes V_m and $V_{m'}$ such that $V_m = \{x, y, z\}$ for some $z \in V(C)$ with $z \neq x, y$, and $V_{m'} = \{x, y, z'\}$ for some $z' \in V(C')$ with $z' \neq x, y$. Note that $l \leq m, m' \leq j'$. Suppose first that $l \leq m < m' \leq j'$. We show that all vertices of $V(C) - \{x, y\}$ occur only on the left side of $V_{m'}$. Suppose there is a vertex $v \in V(C) - \{x, y\}$ which occurs on the right side of $V_{m'}$. There is a path from v to z in C which does not contain x and y. Node $V_{m'}$ contains a vertex of this path. Hence $|V_{m'}| \geq 4$. This is a contradiction. Since each V_i, $m \leq i \leq m'$, contains x and y, this means that there is an i, $m \leq i < m'$, such that all vertices of $V(C) - \{x, y\}$ occur only in $(V_1, ..., V_i)$, and the vertices of $V(C') - \{x, y\}$ occur only in $(V_{i+1}, ..., V_t)$. Furthermore, since $i < j'$ and $V_{j'}$ contains an edge of C, $V_{j'}$ contains x and y. Similarly, V_l contains x and y.

Now suppose $l \leq m' < m \leq j'$. In the same way as before, we can show that the vertices of $V(C) - \{x, y\}$ occur only on the right side of $V_{m'}$, and the vertices of $V(C') - \{x, y\}$ occur only on the left side of V_m. Hence there is an i, $m' \leq i < m$, such that all vertices of $V(C) - \{x, y\}$ occur only in $(V_{i+1}, ..., V_t)$ and all vertices of $V(C') - \{x, y\}$ occur only in $(V_1, ..., V_i)$. Furthermore, V_l is the leftmost node which contains an edge of C', which means that $j = l$. In the same way, we can prove that $j' = l'$, and V_l and $V_{j'}$ both contain x and y. □

Note that in part 2 of the lemma, the part $(V_j, ..., V_i)$ of PD restricted to $V(C)$ is a path decomposition of C, and $(V_{i+1}, ..., V_l)$ restricted to $V(C')$ is a path decomposition of C'. We say that C occurs on the left side of C'. In other words, Lemma 3.3 says that, if there are two cycles which have one edge in common, then in each path decomposition, one occurs on the left side of the other one.

Lemma 3.4. *Let G be a biconnected partial two-path, C a chordless cycle of $\bar{G}$ which has edges e_1 and e_2, $e_1 \neq e_2$, in common with chordless cycles C_1 and C_2, respectively. Then C_1 and C_2 do not have a common edge.*

Proof. If C_1 and C_2 have an edge in common, then K_4, the complete graph on four vertices, is a minor of $\bar{G}$, and hence $\bar{G}$ does not have pathwidth two. □

Lemma 3.5. *Let G be a biconnected partial two-path, C a chordless cycle in $\bar{G}$. If C has two edges in common with two other chordless cycles C_1 and C_2 of $\bar{G}$, then C_1 and C_2 can not both occur on the same side of the occurrence of C.*

Proof. Let $PD = (V_1,..., V_t)$ be a path decomposition of width two of G, suppose C occurs in $(V_j,..., V_{j'})$. Suppose $e_1 = \{x_1, y_1\}$ and $e_2 = \{x_2, y_2\}$ are the edges that C has in common with C_1 and C_2, respectively, and C_1 and C_2 occur on the left side of C. Then e_1 and e_2 occur in V_j. e_1 and e_2 must have a common vertex, otherwise $|V_j| \geq 4$, say $y_1 = x_2$. All vertices of C_1 and C_2 other than x_1, x_2 and y_2 occur only on the left side of V_j, since V_j contains x_1, x_2 and y_2 (see proof of Lemma 3.3). Suppose the leftmost edge of C_1 occurs in V_l, the leftmost edge of C_2 occurs in $V_{l'}$, and $l \leq l'$. Then each V_i, $l' \leq i \leq j$, contains at least two vertices of C_1 and there is a V_i which contains three vertices of C_2. Because of Lemma 3.4, C_1 and C_2 have only one vertex in common, which means that $|V_i| \geq 4$. □

The following corollary follows directly from Lemma 3.5.

Corollary 3.1. *Let G be a biconnected partial two-path, C a chordless cycle in $\bar{G}$. C has at most two edges in common with two other chordless cycles.*

We have now shown that the chordless cycles of the cell completion of a biconnected partial two-path form a sequence, such that each chordless cycle has exactly one edge in common with the following chordless cycle in the sequence.

Lemma 3.6. *Let G be a partial two-path, let $e \in E(G)$ such that e is an edge of three or more chordless cycles of $\bar{G}$, then at most two of these cycles have four or more vertices.*

Proof. Suppose e is an edge of $s \geq 3$ chordless cycles C_i, $3 \leq i \leq s$. Let PD be a path decomposition of width two of G, and suppose w.l.o.g. that C_i occurs on the left side of C_j for all i and j with $i < j$. Since C_1 and C_s have x and y in common, x and y occur in the first and the last V_j containing an edge of all C_i with $1 < i < s$. Hence $|V(C_i)| = 3$ for all i, $1 < i < s$. □

We can now prove the main result of this section.

Theorem 3.1. *Let G be a biconnected graph. G is a partial two-path if and only if $\bar{G}$ can be written as a path of chordless cycles.*

Proof. If $\bar{G}$ can be written as a path of chordless cycles, then we can make a path decomposition of width two of G as follows. Let $(\mathcal{C}, \mathcal{S})$ be a path of chordless cycles for $\bar{G}$, with $\mathcal{C} = (C_1, \ldots, C_p)$ and $\mathcal{S} = (e_1, \ldots, e_{p-1})$. Let e_0 be an arbitrary edge in C_1 with $e_0 \neq e_1$, and let e_p be an arbitrary edge in C_p with $e_p \neq e_{p-1}$. For each i, $1 \leq i \leq s$, we make a path decomposition $(V_1, \ldots, V_t)$ of C_i as follows. If $|V(C_i)| = 3$, make one node containing all vertices of C_i. Otherwise, do the following. Let e_{i-1} occur in V_1, let e_i occur in V_t. Let $e_i = \{x, y\}$ and $e_{i+1} = \{x', y'\}$ such that there is a path from x to x' which does not contain y or y', Let $P_1 = (u_1, \ldots, u_q)$ denote the path in C_i from x to x' which does not contain y or y', and let $P_2 = (v_1, \ldots, v_r)$ denote the path in C_i from y to y' not containing x or x'. Then $t = q + r - 2$, for each i, $1 \leq i < q$, $V_i = \{u_i, u_{i+1}, v_1\}$, and for each i, $1 \leq i < r$, $V_{i+q-1} = \{u_q, v_i, v_{i+1}\}$. The path decompositions for the chordless cycles that are obtained in this way are concatenated in the order in which the chordless cycles occur in $(\mathcal{C}, \mathcal{S})$.

In Figure 3, an example of a path decomposition of width two is given for the graph of Figure 1. The path decomposition is constructed in the way that is given here, with $e_0 = \{1, 18\}$ and $e_p = \{9, 10\}$.

If G is a partial two-path, then it follows directly from Lemmas 3.1, 3.4, 3.5 and 3.6 that $\bar{G}$ can be written as a path of chordless cycles. $\square$

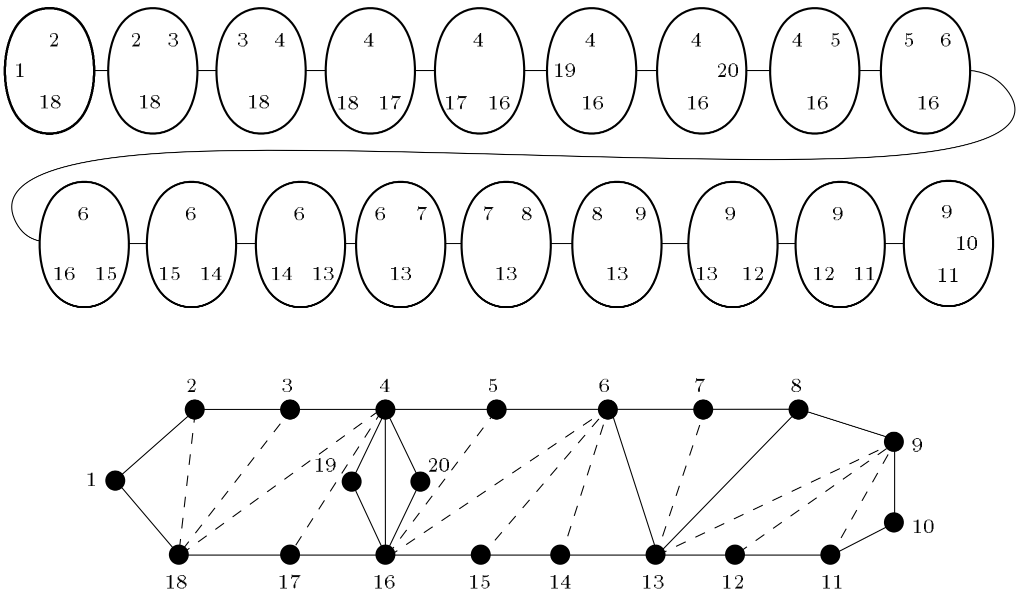

Figure 3: A path decomposition of width two for the graph of Figure 1 as constructed in the proof of Theorem 3.1, and the corresponding interval completion. The dashed edges are the edges that are added.

In the same way as in [BK93], we can check whether $\bar{G}$ is a tree of chordless cycles, and make a list of all chordless cycles in linear time. After that, we can check in linear time whether the tree of chordless cycles is a path of chordless cycles.

3.2 Trees of Pathwidth Two

The following result, describing the structure of trees of pathwidth k, is similar to a result in [EST94].

Lemma 3.7. *Let H be a tree. H is a partial k-path, $k \geq 1$, if and only if there is a path $P = (v_1,..., v_s)$ in H such that the connected components of $H[V - V(P)]$ have pathwidth $k-1$ at most, i.e. H consists of a path with partial $(k-1)$-paths connected to it.*

Proof. If G consists of a path $P = (v_1,..., v_s)$ with partial $(k-1)$-paths connected to it, then we can make a path decomposition of G, by making a path decomposition of width $k-1$ for each connected component of $G[V - V(P)]$, then adding the vertex v_i to each node of the path decomposition of the components connected to v_i, then 'gluing' these path decompositions together in the following way. For all components that are connected to the same v_i, the path decompositions are concatenated in arbitrary order. Two new path decompositions are glued to each other by a node containing v_i and v_{i+1} if they are connected to v_i and v_{i+1}, respectively. This gives a path decomposition of width k of G.

Suppose $(V_1,..., V_t)$ is a path decomposition of G of width k. Select $v, w \in V$ such that $v \in V_1$ and $w \in V_t$. Let P be the path from v to w in G. Then each V_i, $1 \leq i \leq t$, contains a vertex of P. Hence each component of $G[V - V(P)]$ has pathwidth $k-1$. □

Because graphs of pathwidth one do not contain cycles, a graph of pathwidth one is a tree which consists of a path with 'sticks', which are vertices of degree one adjacent only to a vertex on the path ('caterpillars with hair length one'). For an example of a partial one-path, see Figure 4.

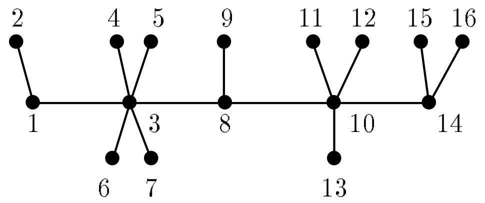

Figure 4: Example of a partial one-path.

Lemma 3.8. *Let H be a tree of pathwidth k, $k \geq 1$, suppose there is no vertex $v \in V(H)$ such that $H[V - \{v\}]$ has pathwidth $k-1$ or less. Then there is a unique path P in H such that the components of $H[V - V(P)]$ have pathwidth $k-1$ or less, and P is shorter than and contained in all other paths having this property.*

Proof. If P is a path in G such that the components of $G[V - V(P)]$ are partial $(k-1)$-paths, then all the paths in G containing P have that same property. Suppose there are two distinct paths P and P', such that the components of $G[V - V(P)]$ and the components of $G[V - V(P')]$ have pathwidth $k-1$ at most. We first show that $V(P) \cap V(P') \neq \emptyset$. Suppose $V(P) \cap V(P') = \emptyset$. Let H' be the component of $H[V - V(P)]$ which contains P', let H'' be the component of $H[V - V(P')]$ which contains P, and let $v \in V(P)$ be the vertex to which H' is connected, i.e. there is a $w \in V(H')$ such that $\{v, w\} \in E(H)$. See Figure 5. Consider the components of $H[V - \{v\}]$. H' is one of these components, and has pathwidth one. All other components contain no vertex of P', and hence are subgraphs of H'', which also has pathwidth one. Hence $H[V - \{v\}]$ has pathwidth one, which gives a contradiction.

Let P'' be the intersection of P and P', which is again a (non-empty) path. The components of $G[V - V(P'')]$ have at most pathwidth $k-1$, since each such component contains no vertices of P or no vertices of P', hence is a component or a subgraph of a component of either $G[V - V(P)]$ or $G[V - V(P')]$.

This means that the intersection P' of all paths P for which $H[V - V(P)]$ has pathwidth one also has the property that $H[V - V(P')]$ has pathwidth one, and it is unique and shorter than all other paths having this property. □

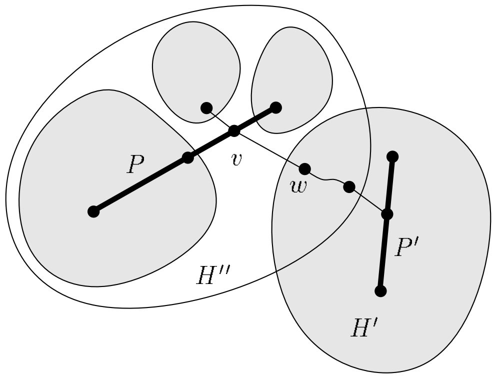

Figure 5: Example of a tree of pathwidth two for proof of Lemma 3.8. The graphs $H[V - V(P)]$ and $H[V - V(P')]$ have pathwidth one, which means that $H[V - \{v\}]$ also has pathwidth one.

Let H be a tree of pathwidth k. In the next two lemmas, we show that for $k = 1$ and $k = 2$, if there is a vertex $v \in V(H)$ such that $H[V - \{v\}]$ has pathwidth $k - 1$, then there are at most a constant number of vertices for which this holds.

Lemma 3.9. *Let H be a tree of pathwidth one, let $W \subseteq V(H)$ be the set of vertices which separate H in components of pathwidth zero, suppose $|W| \geq 1$. Then $|W| \leq 2$, and if $|V(H)| > 2$, then $|W| = 1$.*

Proof. Let $v \in W$. Then $H[V - \{v\}]$ consists of single vertices. If $|V| = 2$, then G consists of one edge, so $|W| = 2$. If $|V| > 2$, then all (at least two) edges of G are incident with v. Hence for each $w \in V - \{v\}$, $H[V - \{w\}]$ contains at least one edge incident with v, and does not have pathwidth zero. So if $|V| > 2$, then $|W| = 1$. □

Lemma 3.10. *Let H be a tree of pathwidth two, let $W \subseteq V(H)$ be the set of vertices which separate H in components of pathwidth at most one. Suppose $|W| \geq 1$. The following holds.*

1. *$H[W]$ is a connected graph.*
2. *If there is a $v \in W$ such that $H[V - \{v\}]$ has four or more components of pathwidth one, then $|W| = 1$.*
3. *There is a vertex $v \in W$ such that $H[V - \{v\}]$ has two or more components of pathwidth one.*
4. *$|W| \leq 7$.*

Proof. 1. Suppose $|W| \geq 2$. Let $v, v' \in W$ be distinct vertices. Let w be a vertex on the path from v to v' in H. Then each component of $H[V - \{w\}]$ does not contain v or does not contain v'. Hence each component is a subgraph of a component of $H[V - \{v\}]$ or of $H[V - \{v'\}]$, so $w \in W$.

2. Let $v \in W$, let H_i, $1 \leq i \leq s$, be the components of $H[V - \{v\}]$ which have pathwidth one. Suppose $s \geq 4$. Let $w \in V(H)$ for some $w \neq v$, and let H' be the component of $H[V - \{w\}]$ containing v. If $w \in V(H_j)$ for some j, then H' contains all H_i with $i \neq j$. Otherwise, H' contains all H_i. In both cases, H' has pathwidth two, according to Lemma 2.5, since v separates H' in three or more components of pathwidth one. Hence $|W| = 1$.

3. Suppose W does not contain a vertex $v \in W$ such that $H[V - \{v\}]$ has two or more components of pathwidth one. Let $v \in W$. There is one component of $H[V - \{v\}]$ which has pathwidth one, otherwise, H has pathwidth one at most. Let H' be this component, and let $w \in V(H')$ such that $\{v, w\} \in E(H)$. There are two possibilities for w. Either w is an inner vertex of the path $P_1(H')$ of H', or w is a stick of an inner vertex w' of $P_1(H')$. In all other cases, H has pathwidth one. If w is inner vertex of $P_1(H')$, then $H[V - \{w\}]$ has at least two components of pathwidth one, namely the two components which contain vertices of $P_1(H')$. Furthermore, all components of $H[V - \{w\}]$ have pathwidth one, since all neighbors of v except w have degree one. Hence the component containing v has pathwidth one. If w is a stick of inner vertex w' of $P_1(H')$, then $H[V - \{w'\}]$ has at least two components of pathwidth one for the same reason, and all components of $H[V - \{w'\}]$ have pathwidth one.

4. If W contains a vertex v for which $H[V - \{v\}]$ has four or more components of pathwidth one, then $|W| = 1$.

Consider the case that for all $v \in W$, $H[V - \{v\}]$ has at most three components of pathwidth one. First suppose W contains a vertex v such that $H[V - \{v\}]$ has three components of pathwidth one. Let H_1, H_2 and H_3 denote these components. For all $w \in V$ such that $w \neq v$ and $w \notin V(H_1) \cup V(H_2) \cup V(H_3)$, $H[V - \{w\}]$ has a component of pathwidth two, namely the component containing v. Let $w \in H_1$. All components of $H[V - \{w\}]$ which do not contain v have pathwidth one. Let H' be the component of $H[V - \{w\}]$ containing v. If H' has pathwidth one then v is an inner vertex of $P_1(H')$, since H_2 and H_3 both have pathwidth one. In that case, either $V(H') = V(H_2) \cup V(H_3) \cup \{v\}$ or $V(H') = V(H_2) \cup V(H_3) \cup \{v, w'\}$ for some $w' \in V(H_1)$ with $\{v, w\} \in E(H)$. This means that there are at most two possibilities for $w \in V(H_1)$ such that $H[V - \{w\}]$ has pathwidth one. The same holds for H_2 and H_3, hence $|W| \leq 7$.

Now suppose W contains no vertex $v \in W$ such that $H[V - \{v\}]$ has three components of pathwidth one. Let $v \in W$ such that $H[V - \{v\}]$ has two components of pathwidth one. Let H_1 and H_2 be the components of $H[V - \{v\}]$ which have pathwidth one, and let $w_1 \in V(H_1)$ and $w_2 \in V(H_2)$ such that $\{v, w_1\}, \{v, w_2\} \in E(H)$. Then for $i = 1, 2$, w_i is either an inner vertex or a stick adjacent to an inner vertex of the path $P_1(H_i)$, since otherwise either H does not have pathwidth two, or W contains a vertex w such that $H[V - \{w\}]$ has three components of pathwidth one. For each $w \in W$ with $w \neq v$ and $w \notin V(H_1) \cup V(H_2)$, $H[V - \{w\}]$ has pathwidth two. If w_1 is inner vertex of $P_1(H_1)$ and v has degree two, then w_2 is the only vertex in H_2 for which $H[V - \{w_2\}]$ has pathwidth one, otherwise, there is no such vertex in H_2. Similar for w_1. Hence $|W| \leq 3$. This completes the proof. $\square$

Note that the bound $|W| \leq 7$ is sharp: in Figure 6, the tree H has pathwidth two and for each vertex $v \in V(H)$ it holds that $H[V - \{v\}]$ has pathwidth one.

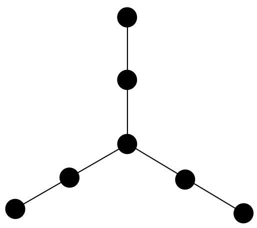

Figure 6: A tree $H = (V, E)$ with pathwidth two, such that for each vertex $v \in V$, $H[V - \{v\}]$ has pathwidth one.

Definition 3.2. *Let H be a tree of pathwidth k, $k \geq 1$. $\mathcal{P}_k(H)$ denotes the set of all paths P in H for which $H[V - V(P)]$ is a partial one-path, and there is no strict sub-path P' of P for which $H[V - V(P')]$ is a partial one-path. If $|\mathcal{P}_k(H)| = 1$, then $P_k(H)$ denotes the unique element of $\mathcal{P}_k(H)$.*

Note that if $\mathcal{P}_k(H)$ contains more than one element, then the elements are all paths consisting of one vertex.

For a tree of pathwidth one, all path decompositions of width one of H are essentially the same.

Lemma 3.11. *Let $H = (V, E)$ be a tree of pathwidth one and let $PD = (V_1, \ldots, V_t)$ be a path decomposition of width one of H. Suppose $|V(H)| > 2$, and let $P_1(H) = (v_1, \ldots, v_s)$. For each $e \in E(H)$, let $f(e)$ be such that $V_{f(e)}$ is the leftmost node containing e. If*

$s \geq 3$, *then either for each* i, $1 \leq i < s-1$, $f(\{v_i, v_{i+1}\}) < f(\{v_{i+1}, v_{i+2}\})$, *or for each* i, $f(\{v_i, v_{i+1}\}) > f(\{v_{i+1}, v_{i+2}\})$. *Suppose the first case holds. Then for each* i, $1 \leq i \leq s$, *and each* $w \in V(H)$ *such that* $\{v_i, w\} \in E(H)$, *the following holds. If* $i < s$, *then* $f(\{v_i, w\}) < f(\{v_i, v_{i+1}\})$, *and if* $i > 1$, *then* $f(\{v_i, w\}) > f(\{v_{i-1}, v_i\})$.

Proof. Follows straightforwardly from the definition of path decomposition. □

In Figure 7, a path decomposition of the partial one-path of Figure 4 is given.

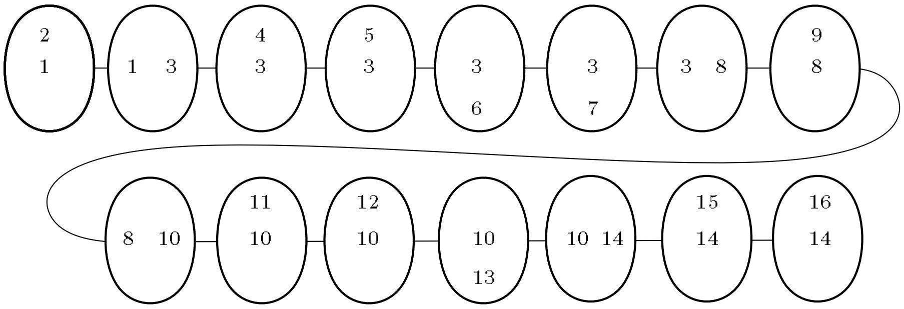

Figure 7: A path decomposition of width one of the partial one-path of Figure 4.

The following lemma is used in the next sections.

Lemma 3.12. *Let* H *be a tree of pathwidth two such that there is a* $v \in V(H)$ *for which* $H[V - \{v\}]$ *has pathwidth one. For each path* P *in* H *for which* $H[V - V(P)]$ *has pathwidth one, there is a* $v \in V(P)$ *such that* $H[V - \{v\}]$ *has pathwidth one.*

Proof. Let P be a path in H for which $H[V - V(P)]$ has pathwidth one. Let $v \in V(H)$ be such that $H[V - \{v\}]$ has pathwidth one. Suppose $v \notin V(P)$. Let H' denote the component of $H[V - V(P)]$ containing v. Let $v' \in V(P)$ be such that there is a $w \in V(H')$ such that $\{v', w\} \in E(H)$. We show that $H[V - \{v'\}]$ has pathwidth one. The components that do not contain a vertex of P have pathwidth one because they are components of $H[V - V(P)]$. All other components are subgraphs of the component of $H[V - \{v\}]$ which contains P. Hence these components also have pathwidth one. □

The lemma implies that if $|\mathcal{P}_2(H)| = 1$, then the element of $\mathcal{P}_2(H)$ is the intersection of all paths P for which $H[V - V(P)]$ has pathwidth one. Furthermore, it implies the following result, which will be frequently used in the next section.

Corollary 3.2. *Let* H *be a tree of pathwidth two,* $PD = (V_1, \ldots, V_t)$ *a path decomposition of width two of* H. *Let* $v \in V_1$ *and* $v' \in V_t$. *Then the path* P *from* v *to* v' *contains one of the paths in* $\mathcal{P}_2(H)$ *as a sub-path.*

The linear time algorithms in [EST94] or [Möh90] to compute the pathwidth of a tree can also be used to find the path $P_2(H)$ if it is unique, or to find all paths in $\mathcal{P}_2(H)$ otherwise.

3.3 Partial Two-Paths

A partial two-path consists of a number of biconnected components, and a number of trees of pathwidth two, which are connected to each other in a certain way.

Definition 3.3. *Let G be a partial two-path. The subgraph G_{T} is the graph obtained from G by deleting all edges of biconnected components of G.*

Let G be a partial two-path. Clearly, the cell completion of each biconnected component of G, can be written as a path of chordless cycles, and each component of G_{T} consists of a path with partial one-paths and sticks connected to it. Note that the cell completion of G is equal to the graph obtained by making a cell completion of each biconnected component of G. The number of possible ways in which biconnected components and components of G_{T} can be connected to each other is large. In this section, we give a complete description of this structure. First we show that for each component H of G_{T}, the vertices of H which are contained in a biconnected component of G all lie on one path, which also contains a path of $\mathcal{P}_2(H)$. After that, we give for each biconnected component of $\bar{G}$ all possible interconnections with other biconnected components of $\bar{G}$ and components of G_{T}.

Lemma 3.13. *Let G be a partial two-path, H a component of G_{T}. Let $V' \subseteq V(H)$ be the set of vertices which are vertices of biconnected components of $\bar{G}$. There is a path in H which contains all vertices of V' and a path of $\mathcal{P}_2(H)$.*

Proof. Let $PD = (V_1,..., V_t)$ be a path decomposition of G, suppose the vertices of H occur in $(V_j,..., V_{j'})$. Select $v \in V_j$ and $v' \in V_{j'}$ such that $v, v' \in V(H)$. Let P denote the path from v to v'. All vertices of V' are on P, since for each $w \in V'$, there is a cycle C which contains w, hence there is a node V_i, $j \leq i \leq j'$, such that V_i contains w and two other vertices of C, so $V_i \cap V(H) = \{w\}$. Furthermore, if H has pathwidth two, then there is a path in $\mathcal{P}_2(H)$ which is a sub-path of P. □

Definition 3.4. *Let G be a partial two-path and H a component of G_{T}. Let $V' \subseteq V(H)$ be the set of all vertices of H which are contained in a biconnected component of G. $\mathcal{P}_H$ denotes the set of all paths P in H for which there is a path in $\mathcal{P}_2(H)$ which is a sub-path of P', $V' \subseteq V(P)$ and there is no strict sub-path P' of P for which there is a path in $\mathcal{P}_2(H)$ which is a sub-path of P' and $V' \subseteq V(P')$. If $|\mathcal{P}_H| = 1$, then P_H denotes the unique element of $\mathcal{P}_H$, and P_H is called* the path *of H.*

Let G be a partial two-path and H a component of G_{T}. If $|\mathcal{P}_2(H)| = 1$, then clearly $|\mathcal{P}_H| = 1$. If $|\mathcal{P}_2(H)| > 1$, then all elements of $\mathcal{P}_2(H)$ are paths consisting of one vertex, and all these vertices form a connected subgraph H' of H. This means that if there is one vertex $v \in V(H)$ for which v is contained in a biconnected component, then there is a unique shortest path containing v and a path from $\mathcal{P}_2(H)$, since one of the vertices of H' is closer to v than the others. If there are two of more vertices of H which are contained in a biconnected component, then a similar argument holds. Hence $|\mathcal{P}_H| = 1$

if there is at least one vertex of H which is contained in a biconnected component of G.

In Figure 8, a partial two-path G is given in which G_{T} has one component H, with $P_2(H) = (v_3, v_4, v_5)$ and $P_H = (v_1, \ldots, v_5)$.

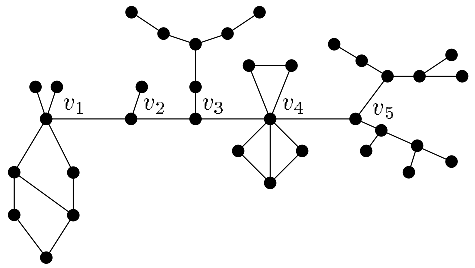

Figure 8: A partial two-path G with one component H in G_{T}. $P_2(H) = (v_3, v_4, v_5)$ and $P_H = (v_1, v_2, v_3, v_4, v_5)$.

From the proof of Lemma 3.13 it can be seen that an analog of Corollary 3.2 also holds for $\mathcal{P}_H$.

Corollary 3.3. *Let G be a partial two-path, G not a tree, H a component of G_{T}. Let $PD = (V_1, \ldots, V_t)$ be a path decomposition of width two of G, suppose H occurs in $(V_j, \ldots, V_{j'})$. There is a $v \in V_j \cap V(H)$ and a $v' \in V_{j'} \cap V(H)$ such that the path from v to v' contains P_H.*

The following lemma shows some conditions for the structure of biconnected components of a partial two-path G which contain a vertex of a component H of G_{T}.

Lemma 3.14. *Let G be a connected partial two-path which is not a tree, H a component of G_{T}, $P_H = (v_1 \ldots, v_s)$ the path of H. Let $G' = G[V - V(P_H)]$. At most two components of G' may have pathwidth two. For each component G'' of G' of pathwidth two, there must be a $v \in V(G')$ such that either $\{v, v_1\} \in E(G)$ or $\{v, v_s\} \in E(G)$, i.e. G'' is connected to v_1 or v_s. If $s > 1$, then at most one component of pathwidth two may be connected to v_1, and at most one to v_s.*

Proof. Because of Lemma 2.5, at most two components of G' may have pathwidth two. If there is a component of width two adjacent to v_i, $1 < i < s$, then v_i is a vertex which separates G into three or more components of width two, and hence G has pathwidth three. If $s \neq 1$ and there are two or more components of width two adjacent to v_1, or if $s = 1$ and there are three or more components of width two adjacent to v_1, then v_1 separates G into three components of width two, and hence G has pathwidth three. $\square$

For the vertices of each biconnected component of a partial two-path, we define states, which reflect the structure of the subgraphs which are connected to them. In Figure 9, an example is given for all possible states.

Definition 3.5. (Vertex States). *Let G be a partial two-path, B a biconnected component of G. Let $v \in V(B)$, and let H denote the component of G_{T} containing v. The* (vertex) state *of v is denoted by $st(v)$, and is defined as follows.*

$st(v) = \mathrm{N}$ *if v has no neighbors outside of B.*

$st(v) = \mathrm{S}$ *if v has only neighbors of degree one outside of B: only sticks are connected to v.*

$st(v) = \mathrm{E1}$ *if H has pathwidth one, $P_H = (v)$, v is adjacent to exactly one vertex $w \notin B$ which does not have degree one and $w \in V(H)$, and either v or w is end point of $P_1(H)$.*

In other words, B is the only biconnected component containing v, H has pathwidth one and contains at least one edge which is not incident with v (hence $|\mathcal{P}_1(H)| = 1$), $P_H = (v)$, and v is not an inner vertex of $P_1(H)$, but there is a path in H containing v and $P_1(H)$.

$st(v) = \mathrm{I1}$ *if B is the only biconnected component containing v, H has pathwidth one and contains at least one edge which is not incident with v, $P_H = (v)$, and v is an inner vertex of $P_1(H)$.*

$st(v) = \mathrm{E2}$ *if there is another biconnected component containing v, or H has pathwidth one, $P_H = (v)$ and there is no path in H containing v and a path of $\mathcal{P}_1(H)$, or H has pathwidth at most two and $P_H \neq (v)$ but v is an end point of P_H.*

$st(v) = \mathrm{I2}$ *if H has pathwidth at most two and v is an inner vertex of P_H.*

The states are ordered in the following way. $\mathrm{I2} \succ \mathrm{E2} \succ \mathrm{I1} \succ \mathrm{E1} \succ \mathrm{S} \succ \mathrm{N}$.

Note that all possibilities are covered for v, and that all states are disjoint. In the remainder of this section, we derive what combinations of states are possible for all vertices of a biconnected component.

Lemma 3.15. *Let G be a partial two-path, C a cycle in G. Let $v \in V(C)$, G' be a component of $G[V - V(C)]$ for which there is a vertex $v' \in V(G')$ such that $\{v, v'\} \in E(G)$. If G' contains at least one edge, then v is an end vertex of C.*

Proof. Let $PD = (V_1,\ldots,V_t)$ be a path decomposition of width two of G, suppose C occurs in $(V_j,\ldots,V_{j'})$, and let $\{x, y\} \in E(C)$ such that $x, y \in V_j$. Suppose $E(G') \neq \emptyset$, let $\{u, w\} \in E(G')$ such that $\{u, v\} \in E$. Edge $\{u, w\}$ can not occur in $(V_j,\ldots,V_{j'})$, so suppose $\{u, w\}$ occurs in V_l, $l < j$. Then either $v \in V_j$ or $u \in V_j$. Suppose $u \in V_j$, and let V_p, $j \leq p \leq j'$, be the leftmost node containing v. Then each node in $V_j,\ldots,V_p$ contains u. Furthermore, there is a node containing x, y, and another vertex of C (Lemma 3.2), which means that $x, y \in V_p$. This is only possible if $v = x$ or $v = y$, which means that v is an end vertex. □

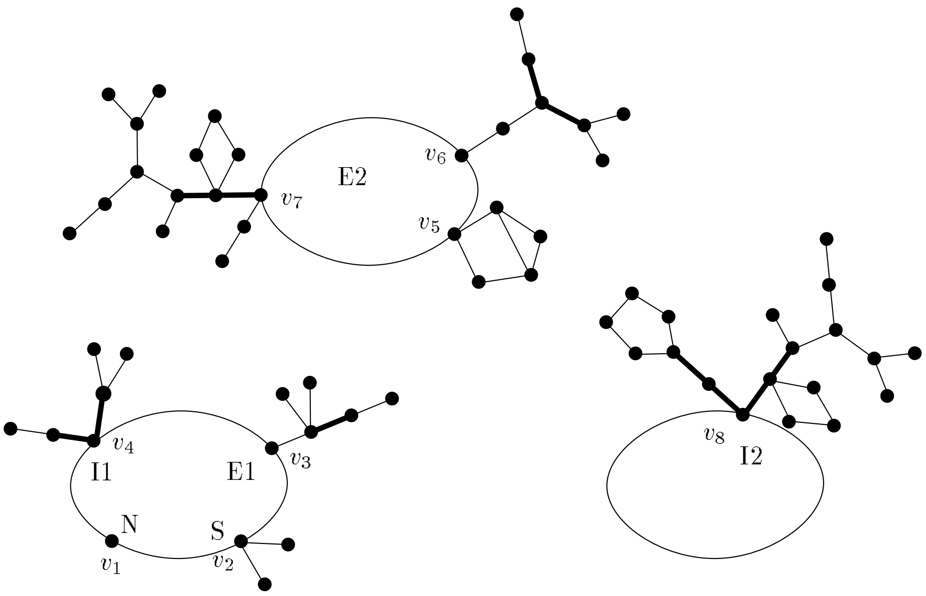

Figure 9: Examples of all vertex states. $st(v_1) = \text{N}$, $st(v_2) = \text{S}$, $st(v_3) = \text{E1}$, $st(v_4) = \text{I1}$, $st(v_5) = st(v_6) = st(v_7) = \text{E2}$ and $st(v_8) = \text{I2}$. For each i, let H_i denote the component of G_{T} which contains v_i. H_1 and H_5 consist of one single vertex. Vertices v_5, v_6 and v_7 give an example for each possibility with state E2. Combinations of these possibilities are also possible. For $i \in \{3, 4, 6\}$, the fat edges in H_i form the path $P_1(H_i)$. For $i \in \{7, 8\}$, the fat edges in H_i form the path P_{H_i}. For $i \in \{1, ..., 6\}$, $P_{H_i} = (v_i)$.

We can now show that, for a biconnected component B of the cell completion of a partial two-path G, there is a path of chordless cycles $(\mathcal{C}, \mathcal{S})$ with $\mathcal{C} = (C_1, ..., C_p)$ in which all vertices of B which have state E1, I1, E2 or I2, are in C_1 or C_p, and all vertices v which are in some C_i with $1 < i < p$ and with $e_i = e_{i+1}$ and $v \notin e_i$ have state N.

Definition 3.6. (Correct Path of Chordless Cycles). *Let G be a partial two-path, B a biconnected component of $\bar{G}$, and let $(\mathcal{C}, \mathcal{S})$ be a path of chordless cycles of B with $\mathcal{C} = (C_1, ..., C_p)$ and $\mathcal{S} = (e_1, ..., e_{p-1})$. If $(\mathcal{C}, \mathcal{S})$ satisfies the following condition, then $(\mathcal{C}, \mathcal{S})$ is called a* correct path of chordless cycles.

$$\forall_{v \in V(B)}\ v \notin V(C_1) \cup V(C_p) \Rightarrow st(v) \in \{\text{N}, \text{S}\} \ \wedge$$
$$\forall_{1 \le i < p-1, v \in V(C_{i+1})}\ e_i = e_{i+1} \wedge v \notin e_i \Rightarrow st(v) = \text{N}$$

Lemma 3.16. *Let G be a partial two-path. Each biconnected component B of $\bar{G}$ can be represented by a correct path of chordless cycles.*

Proof. Let $PD = (V_1, ..., V_t)$ be a path decomposition of width two of G, suppose B occurs in $(V_j, ..., V_{j'})$. According to Lemma 3.3, the chordless cycles occur in some order. Let $\mathcal{C} = (C_1, ..., C_p)$ denote this order. Let $\mathcal{S} = (e_1, ..., e_{p-1})$ be the sequence of edges of B for which $e_i = V(C_i) \cap V(C_{i+1})$ for each i, $1 \leq i < p$. Clearly, $(\mathcal{C}, \mathcal{S})$ is a path of chordless cycles of B.

Let C_i be such that $e_{i-1} = e_i$, let $v \in V(C_i) - e_i$. Then $st(v) = \mathrm{N}$, since e_i is double end edge of C_i, and hence any edge adjacent to v could not occur within the occurrence of C_i, and not within the occurrence of any other C_j.

Finally, we prove that all vertices of the component which are not in $V(C_1)$ or $V(C_p)$ may not be adjacent to something else than sticks. Suppose there is a $v \in V(B) - (V(C_1) \cup V(C_p))$ which does not have state N or S. Let C be the cycle in B with $V(C)$ the set of vertices of $V(B)$ except all $v \in V(B)$ for which $v \in V(C_i) - e_i$ for some i, $1 < i < p$, for which $e_{i-1} = e_i$, and $E(C)$ the set of edges in $B[V(C)]$ except the edges e_i, $1 \leq i < p$. Then v is an end vertex of C. C occurs within $(V_j, ..., V_{j'})$, and V_j and $V_{j'}$ can not contain any vertices of B which are not in C_1 or C_p, which is a contradiction. □

From Lemma 2.6, we can derive that there may be at most four vertices of B which have state E1, I1, E2 or I2. Furthermore, if $(\mathcal{C}, \mathcal{S})$ is a correct path of chordless cycles, and then $V(C_1) - V(C_p)$ and $V(C_p) - V(C_1)$ may each have at most two vertices with state in {E1,I1,E2,I2}.

Let G be a partial two-path, B a biconnected component of $\bar{G}$, $x \in V(B)$ and $st(x) \in \{\mathrm{I2}, \mathrm{E2}, \mathrm{I1}, \mathrm{E1}\}$. Let X be a component of $G[V - V(B)]$ which is connected to x in G such that $|V(X)| > 1$, and let X' denote $G[V(X) \cup \{x\}]$. Then in each path decomposition of width two of G, all edges of X' occur on the same side of the occurrence of B, since suppose there are two edges $e, e' \in E(X')$ which occur on different sides of the occurrence of B. There is a path between e and e' which does not contain x, hence each node in the occurrence of B contains a vertex of this path, which is not possible since B has pathwidth two.

Lemma 3.17. *Let G be a partial two-path, C a cycle of $\bar{G}$. Let $PD = (V_1, ..., V_t)$ be a path decomposition of width two of G, suppose C occurs in $(V_j, ..., V_{j'})$. Let $v \in V_j$ such that $v \in V(C)$. V_j also contains a neighbor of v.*

Proof. Let $\{x, y\} \in E(C)$ be such that $x, y \in V_j$. Let V_m, $j \leq m \leq j'$, be the leftmost node which contains another edge of C. Then V_m contains x, y and a neighbor z of x or y in C. Then either $m = j$ and $v = z$ or $v \in \{x, y\}$. □

In the next lemmas, we show that the vertices which have state E1, I1, E2 or I2 must have a 'small distance' to each other.

Lemma 3.18. *Let G be a partial two-path, B a biconnected component of $\bar{G}$. Let PD be a path decomposition of width two of G, such that B occurs in $(V_j, ..., V_{j'})$, let $(\mathcal{C}, \mathcal{S})$ be a path of chordless cycles of B, such that the order in which the chordless cycles of B occur in PD corresponds with $\mathcal{C}$. Let $x, y \in V(B)$, suppose $st(x), st(y) \in \{\mathrm{I2}, \mathrm{E2}, \mathrm{I1}, \mathrm{E1}\}$. Let X' be the graph consisting of all components of $G[V - V(B)]$ which are connected to x in G, and which occur on the left side of $(V_j, ..., V_{j'})$, and let X denote $G[V(X') \cup \{x\}]$.*

Similarly, let Y and Y' be defined for y (see for example Figure 10). Then $x, y \in V(C_1)$, and

1. *either $\{x,y\} \in E(C_1) - \{e_1\}$ or there is a vertex $z \in V(B)$ such that $\{x,z\} \in E(C_1) - \{e_1\}$ and $\{z,y\} \in E(C_1) - \{e_1\}$ and $st(z) = \mathrm{N}$ and*
2. *either X is a partial one-path such that x is not an inner vertex of $P_1(X)$ but there is a path containing $P_1(X)$ and x, or Y is a partial one-path such that y is not an inner vertex of $P_1(Y)$ but there is a path containing $P_1(Y)$ and y.*

Proof. 1. Both x and y occur in V_j, so $x, y \in V(C_1)$. There is a neighbor of x in V_j and a neighbor of y in V_j. This means that, according to Lemma 3.17, either $\{x,y\} \in E(C_1)$ or there is a $z \in V(C_1)$ such that $\{x,z\} \in E(C_1)$ and $\{y,z\} \in E(C_1)$. If $\{x,y\} = e_1$, then $\{x,y\}$ is a double end edge of C_1, hence $|V(C_1)| = 3$, so there is a $z \in V(C_1)$ such that $\{x,z\}, \{y,z\} \in E(C_1) - \{e_1\}$. If there is a $z \in V(C_1)$ such that $\{x,z\} \in E(C_1)$ and $\{y,z\} = e_1$, then e_1 also is a double end vertex, hence $|V(C_1)| = 3$, and $\{x,y\} \in E(C_1) - \{e_1\}$.

Suppose $\{x,y\} \notin E(C_1) - \{e_1\}$, and let z be the common neighbor of x and y such that $V_j = \{x,y,z\}$. Let V_i, $i < j$, be the rightmost node containing an edge of X' or Y'. Then $V_i = \{x,y,z'\}$ for some $z' \in V(X') \cup V(Y')$. This means that there can be no edge incident with z which occurs on the left side of V_j. In the same way, we can prove that there can be no edge incident with z which occurs on the right side of V_j.

2. Suppose X occurs in $(V_l, \ldots, V_{l'})$, $1 \leq l \leq l' \leq j$, and Y occurs in $(V_m, \ldots, V_{m'})$, $1 \leq m \leq m' \leq j$, and suppose that $m < l$. See also part II of Figure 10. It is clear that $x \in V_{l'}$ and $y \in V_{m'}$, and that X has pathwidth one. Furthermore, the rightmost node containing an edge of X contains an end point v of the path $P_1(X)$ and a stick v' adjacent to it. This means that $x \in \{v, v'\}$, hence x is either an end point of $P_1(X)$ or a stick adjacent to an end point of $P_1(X)$. □

From this lemma, we can derive the following corollary.

Corollary 3.4. *Let G be a partial two-path, B a biconnected component of $\bar{G}$, $(\mathcal{C}, \mathcal{S})$ a correct path of chordless cycles of B. Let $x_1, \ldots, x_s \in V(B)$ such that $st(x_i) \in \{\mathrm{I2}, \mathrm{E2}, \mathrm{I1}\}$. Then $s \leq 3$. Furthermore, if $s = 3$, there is a j, $1 \leq j \leq 3$, such that $st(x_j) = \mathrm{I1}$ and x_j is a double end vertex of B, which implies that $x_j \in e_i$ for each i.*

To be able to give the possible states for the biconnected components in a partial two-path, we first give a definition.

Definition 3.7. (Distance). *Let G be a partial two-path, B a biconnected component of $\bar{G}$ and $(\mathcal{C}, \mathcal{S})$ a correct path of chordless cycles for B, $\mathcal{C} = (C_1, \ldots, C_p)$, $\mathcal{S} = (e_1, \ldots, e_{p-1})$. For each $u, v \in V(B)$, $\mathrm{dst}_1(u,v) \in \{\mathrm{true}, \mathrm{false}\}$ and $\mathrm{dst}_p(u,v) \in \{\mathrm{true}, \mathrm{false}\}$ are defined as follows. If $p = 1$, then*

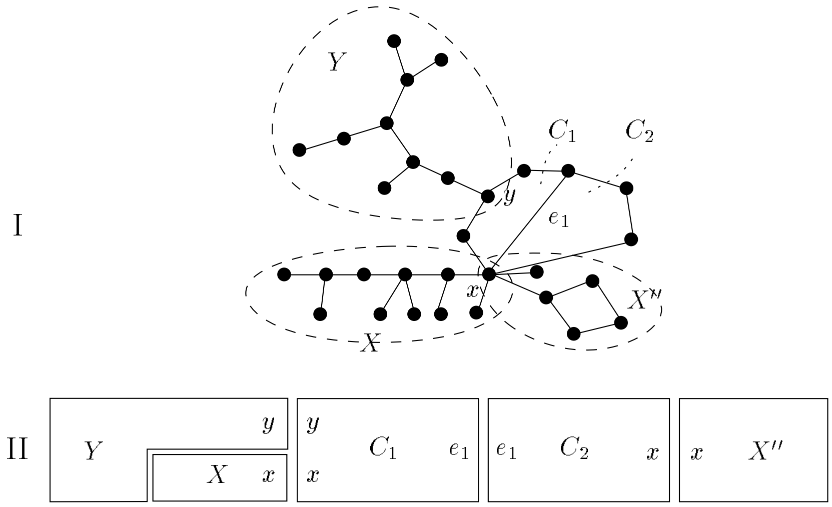

Figure 10: Part I is a partial two-path G which contains a path with chordless cycles $(\mathcal{C}, \mathcal{S})$ with $\mathcal{C} = (C_1, C_2)$, $\mathcal{S} = (e_1)$. Vertices $x, y \in V(C_1)$ both have state E2. Part II shows the order of the occurrences of C_1, C_2, X, Y and X'' in a possible path decomposition of width two of G, as it is used for the proof of Lemma 3.18.

$$
\begin{aligned}
\mathrm{dst}_1(u,v) \quad \Leftrightarrow \quad & u, v \in V(C_1) \wedge \\
& (\{u,v\} \in E(C_1) \vee \\
& \exists_{w \in V(C_1)} \{u,w\}, \{v,w\} \in E(C_1) \wedge st(w) = \mathrm{N})
\end{aligned}
$$

If $p > 1$ *then*

$$
\begin{aligned}
\mathrm{dst}_1(u,v) \quad \Leftrightarrow \quad & u, v \in V(C_1) \wedge \\
& (\{u,v\} \in E(C_1) - \{e_1\} \vee \\
& \exists_{w \in V(C_1)} \{u,w\}, \{v,w\} \in E(C_1) - \{e_1\} \wedge st(w) = \mathrm{N}) \\
\mathrm{dst}_p(u,v) \quad \Leftrightarrow \quad & u, v \in V(C_p) \wedge \\
& (\{u,v\} \in E(C_p) - \{e_{p-1}\} \vee \\
& \exists_{w \in V(C_p)} \{u,w\}, \{v,w\} \in E(C_p) - \{e_{p-1}\} \wedge st(w) = \mathrm{N})
\end{aligned}
$$

In Figure 11, an example is given of dst for a path of chordless cycles with two chordless cycles.

In the following definition, the state of a biconnected component is defined. Furthermore, for each state a definition is given of a condition which must hold for the biconnected component of that state, such that the graph can be a partial two-path. In Figures 13, 14, 15, and 16 there is a picture of the condition for each state. The pictures are symbolically. In Figure 12 the legend is given.

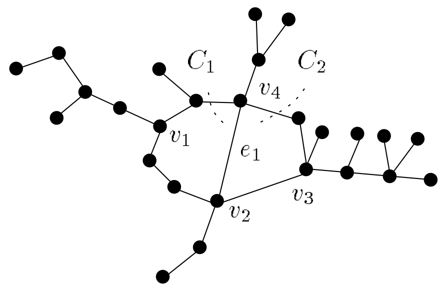

Figure 11: Example for the definition of $\mathrm{dst}_1(v_i, v_j)$. The picture shows a path of chordless cycles $(\mathcal{C}, \mathcal{S})$ with $\mathcal{C} = (C_1, C_2)$, $\mathcal{S} = (e_1)$. $\mathrm{dst}_2(v_2, v_3)$ and $\mathrm{dst}_2(v_3, v_4)$ hold. $\mathrm{dst}_1(v_2, v_4)$ and $\mathrm{dst}_2(v_2, v_4)$ do not hold, since the edge between v_2 and v_4 is edge e_1. $\mathrm{dst}_1(v_1, v_4)$ does not hold, since the common neighbor of v_1 and v_3 has state S.

Definition 3.8. (Biconnected Component States). *Let G be a partial two-path, B a biconnected component of $\bar{G}$, and $(\mathcal{C}, \mathcal{S})$ a correct path of chordless cycles for B, $\mathcal{C} = (C_1,\ldots, C_p)$, $\mathcal{S} = (e_1,\ldots, e_{p-1})$. Let $v_1,\ldots, v_s$ denote the vertices of B which do not have state* N *or* S, *such that $st(v_i) \succeq st(v_{i+1})$ for each i, $1 \leq i < s$. The* state of B *is denoted by $st(B)$, and is defined as $st(B) = (st(v_1), st(v_2),\ldots, st(v_s))$. Because G is a partial two-path, the vertices $v_1,\ldots, v_s$ satisfy a number of conditions. For each value of $st(B)$, we denote these conditions by* $\mathrm{cond}(st(B))$. *The conditions will be defined in the following tables.*

$s = 0$: $\mathrm{cond}(()) = \mathrm{true}$.

$s = 1$

$st(B)$	$\mathrm{cond}(st(B))$
(I2)	$\forall_{1 \leq i < p}\ v_1 \in e_i$
(E2)	$v_1 \in V(C_1) \cup V(C_p)$
(I1)	$v_1 \in V(C_1) \cup V(C_p)$
(E1)	$v_1 \in V(C_1) \cup V(C_p)$

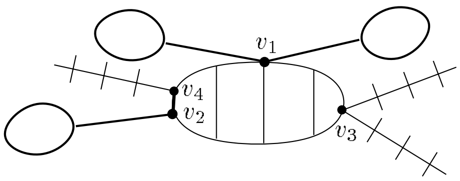

Figure 12: Legend for Figures 13, 14, 15, 16 and 17. A path of chordless cycles $(\mathcal{C}, \mathcal{S})$ is represented by an ellipsis in which the vertical lines denote the common edges of the chordless cycles. The leftmost chordless cycle represents C_1, the rightmost one represents C_p. The vertices that have one of the states in $\{\mathrm{I2}, \mathrm{E2}, \mathrm{I1}, \mathrm{E1}\}$ are represented by a dot. All other vertices are not drawn. A vertex that has state I2 is represented as vertex v_1, a vertex with state E2 is represented as vertex v_2, a vertex with state I1 is represented as vertex v_3, a vertex that has state E1 is represented as vertex v_4. If $\mathrm{dst}_1(u, v)$ holds for two vertices, then the vertices are both drawn in the leftmost chordless cycle, and they are connected by a fat edge. If $\mathrm{dst}_p(u, v)$ holds, then u and v are both in the rightmost cycle, and they are connected by a fat edge. In the figure, $\mathrm{dst}_1(v_2, v_4)$ holds.

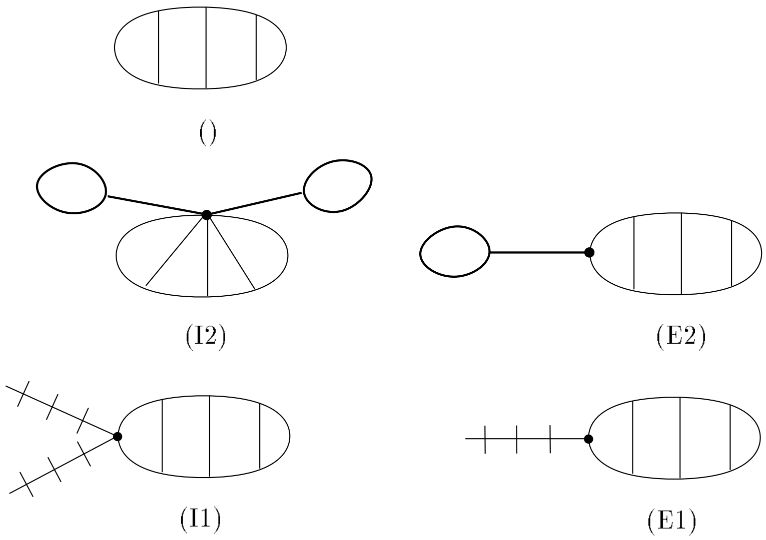

Figure 13: Symbollic representation of $\mathrm{cond}(S)$ for each possible biconnected component state S for $s = 0$ and $s = 1$. Cases that are symmetrical in C_1 and C_p are given only once.

$s = 2$

$st(B)$	cond$(st(B))$
(I2, I1)	$((\forall_{1\leq i<p}\ v_1 \in e_i \wedge v_2 \in e_i) \wedge \lvert V(C_1)\rvert = \lvert V(C_p)\rvert = 3 \wedge \mathrm{dst}_1(v_1, v_2) \wedge \mathrm{dst}_p(v_1, v_2)) \vee (p = 1 \wedge V(C_1) = \{v_1, v_2, u, w\} \wedge E(C_1) = \{\{v_1, u\}, \{v_2, u\}\}, \{v_1, w\}, \{v_2, w\} \wedge p = 1 \wedge st(u) = st(w) = \mathrm{N})$
(I2, E1)	$(\forall_{1\leq i<p}\ v_1 \in e_i) \wedge (\mathrm{dst}_1(v_1, v_2) \vee \mathrm{dst}_p(v_1, v_2))$
(E2, E2)	$(v_1 \in C_1 \wedge v_2 \in C_p) \vee (v_1 \in C_p \wedge v_2 \in C_1)$
(E2, I1)	cond((E2, E2))
(E2, E1)	$((v_1 \in C_1 \wedge v_2 \in C_p) \vee (v_1 \in C_p \wedge v_2 \in C_1) \vee \mathrm{dst}_1(v_1, v_2) \vee \mathrm{dst}_p(v_1, v_2))$
(I1, I1)	cond((E2, E2))
(I1, E1)	cond((E2, E1))
(E1, E1)	cond((E2, E1))

$s = 3$

$st(B)$	cond$(st(B))$
(I2, E1, E1)	$(\mathrm{dst}_1(v_1, v_2) \wedge \mathrm{dst}_p(v_1, v_3)) \vee (\mathrm{dst}_p(v_1, v_2) \wedge \mathrm{dst}_1(v_1, v_3))$
(E2, E2, I1)	$(\mathrm{dst}_1(v_1, v_3) \wedge \mathrm{dst}_p(v_2, v_3)) \vee (\mathrm{dst}_p(v_1, v_3) \wedge \mathrm{dst}_1(v_2, v_3))$
(E2, E2, E1)	$(v_1 \in V(C_1) \wedge \mathrm{dst}_p(v_2, v_3)) \vee (v_1 \in V(C_p) \wedge \mathrm{dst}_1(v_2, v_3)) \vee (v_2 \in V(C_1) \wedge \mathrm{dst}_p(v_1, v_3)) \vee (v_2 \in V(C_p) \wedge \mathrm{dst}_1(v_1, v_3))$
(E2, I1, I1)	$(\mathrm{dst}_1(v_1, v_3) \wedge \mathrm{dst}_p(v_2, v_3)) \vee (\mathrm{dst}_p(v_1, v_3) \wedge \mathrm{dst}_1(v_2, v_3)) \vee (\mathrm{dst}_1(v_1, v_2)) \wedge \mathrm{dst}_p(v_3, v_2)) \vee (\mathrm{dst}_p(v_1, v_2) \wedge \mathrm{dst}_1(v_3, v_2))$
(E2, I1, E1)	cond((E2, E2, E1))
(E2, E1, E1)	$((v_1 \in V(C_1) \wedge \mathrm{dst}_p(v_2, v_3)) \vee (v_1 \in V(C_p) \wedge \mathrm{dst}_1(v_2, v_3)) \vee (v_2 \in V(C_1) \wedge \mathrm{dst}_p(v_1, v_3)) \vee (v_2 \in V(C_p) \wedge \mathrm{dst}_1(v_1, v_3)) \vee (v_3 \in V(C_1) \wedge \mathrm{dst}_p(v_1, v_2)) \vee (v_3 \in V(C_p) \wedge \mathrm{dst}_1(v_1, v_2)))$
(I1, I1, I1)	$(\mathrm{dst}_1(v_1, v_3) \wedge \mathrm{dst}_p(v_2, v_3)) \vee (\mathrm{dst}_p(v_1, v_3) \wedge \mathrm{dst}_1(v_2, v_3)) \vee (\mathrm{dst}_1(v_1, v_2) \wedge \mathrm{dst}_p(v_3, v_2)) \vee (\mathrm{dst}_p(v_1, v_2) \wedge \mathrm{dst}_1(v_3, v_2)) \vee (\mathrm{dst}_1(v_2, v_1) \wedge \mathrm{dst}_p(v_3, v_1)) \vee (\mathrm{dst}_p(v_2, v_1) \wedge \mathrm{dst}_1(v_3, v_1))$
(I1, I1, E1)	cond((E2, E2, E1))
(I1, E1, E1)	cond((E2, E1, E1))
(E1, E1, E1)	cond((E2, E1, E1))

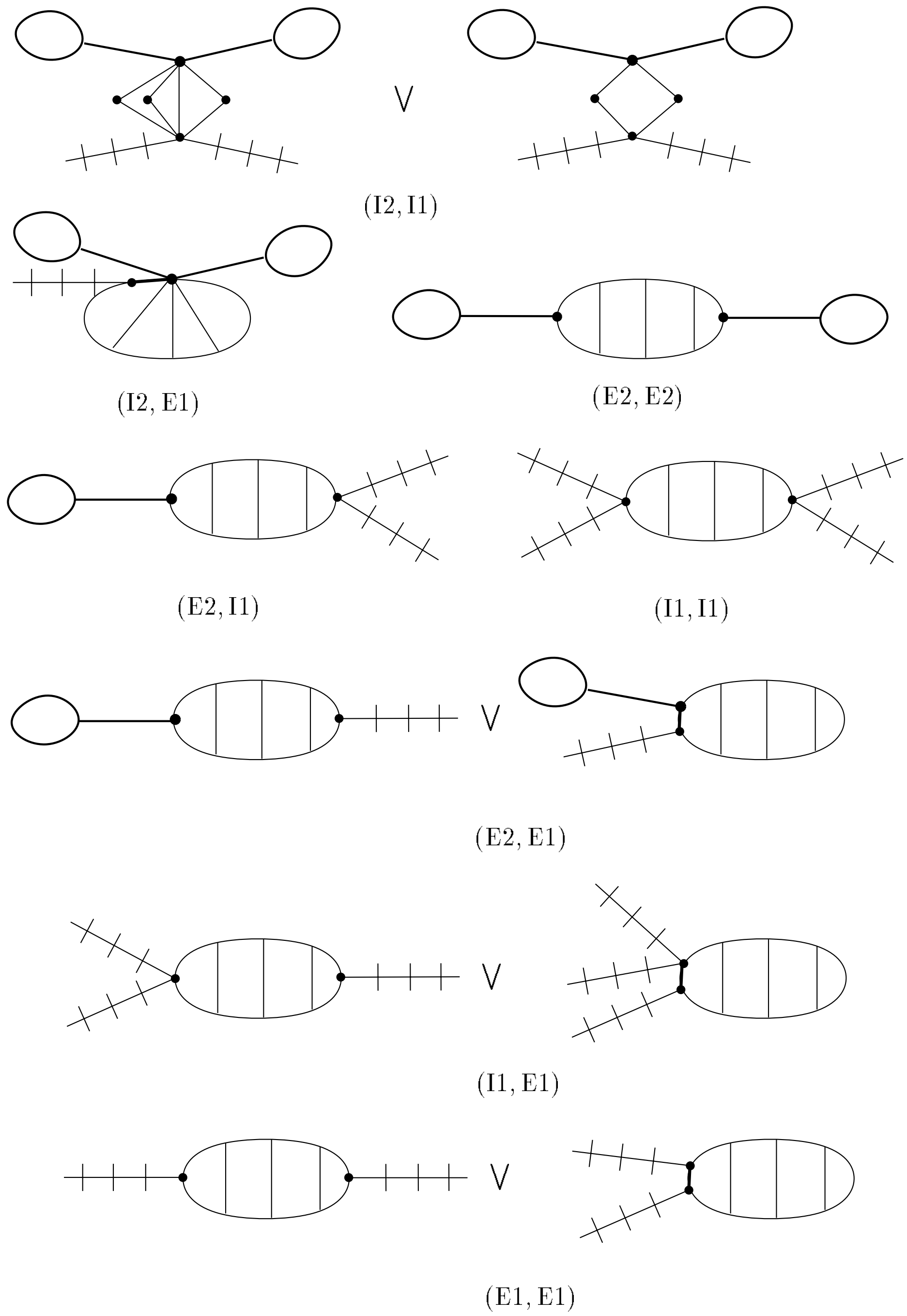

Figure 14: Symbollic representation of cond(S) for each possible biconnected component state S for $s = 2$. For state (I2, I1), the biconnected component is represented in its normal way. Cases that are symmetrical in C_1 and C_p, or in distinct vertices v_i with the same state are given only once.

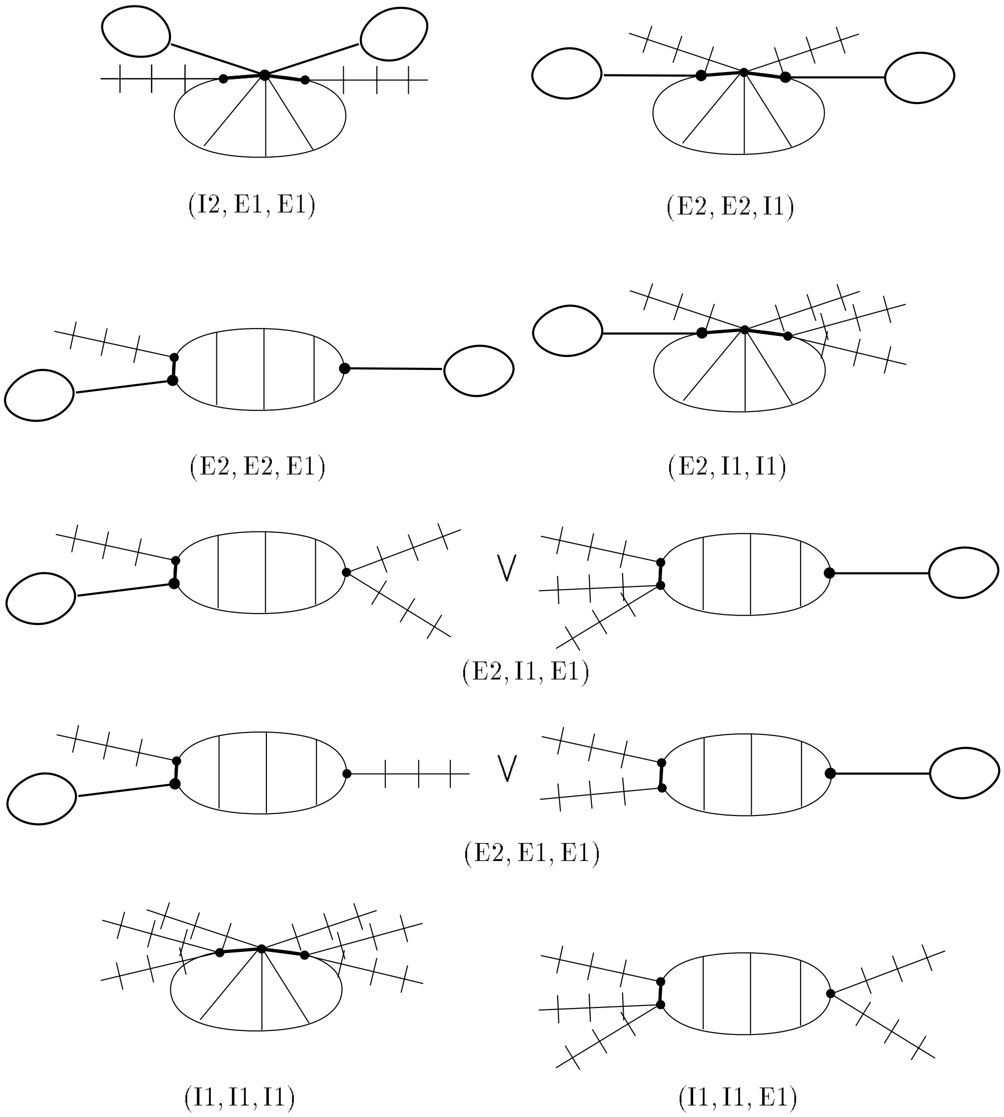

Figure 15: Symbollic representation of cond(S) for some possible biconnected component state S for $s = 3$. Cases that are symmetrical in C_1 and C_p, or in distinct vertices v_i with the same state are given only once.

$s = 4$

$st(B)$	$\text{cond}(st(B))$
$(\text{E2}, \text{E2}, \text{E1}, \text{E1})$	$(\text{dst}_1(v_1, v_3) \wedge \text{dst}_p(v_2, v_4)) \vee (\text{dst}_p(v_1, v_3) \wedge \text{dst}_1(v_2, v_4)) \vee$ $(\text{dst}_1(v_1, v_4) \wedge \text{dst}_p(v_2, v_3)) \vee (\text{dst}_p(v_1, v_4) \wedge \text{dst}_1(v_2, v_3))$
$(\text{E2}, \text{I1}, \text{E1}, \text{E1})$	$\text{cond}((\text{E2}, \text{E2}, \text{E1}, \text{E1}))$
$(\text{E2}, \text{E1}, \text{E1}, \text{E1})$	$(\text{dst}_1(v_1, v_2) \wedge \text{dst}_p(v_3, v_4)) \vee (\text{dst}_p(v_1, v_2) \wedge \text{dst}_1(v_3, v_4)) \vee$ $(\text{dst}_1(v_1, v_3) \wedge \text{dst}_p(v_2, v_4)) \vee (\text{dst}_p(v_1, v_3) \wedge \text{dst}_1(v_2, v_4)) \vee$ $(\text{dst}_1(v_1, v_4) \wedge \text{dst}_p(v_2, v_3)) \vee (\text{dst}_p(v_1, v_4) \wedge \text{dst}_1(v_2, v_3))$
$(\text{I1}, \text{I1}, \text{E1}, \text{E1})$	$\text{cond}((\text{E2}, \text{E2}, \text{E1}, \text{E1}))$
$(\text{I1}, \text{E1}, \text{E1}, \text{E1})$	$\text{cond}((\text{E2}, \text{E1}, \text{E1}, \text{E1}))$
$(\text{E1}, \text{E1}, \text{E1}, \text{E1})$	$\text{cond}((\text{E2}, \text{E1}, \text{E1}, \text{E1}))$

Let $a \in \{\text{I2}, \text{E2}, \text{I1}, \text{E1}\}$. We denote by S_a the set of biconnected component states for which $s \geq 1$ and $st(v_1) = a$.

Note that all the biconnected component states are disjoint, i.e. each biconnected component can have at most one state.

Lemma 3.19. *Let G be a partial two-path. Each biconnected component B of $\bar{G}$ has one of the states in $S_{\text{I2}} \cup S_{\text{E2}} \cup S_{\text{I1}} \cup S_{\text{E1}}$, and satisfies* $\text{cond}(st(B))$.

Proof. Let B be a biconnected component of $\bar{G}$, let $(\mathcal{C}, \mathcal{S})$ be a correct path of chordless cycles of B with $\mathcal{C} = (u_1, \ldots, u_p)$, $\mathcal{S} = (e_1, \ldots, e_{p-1})$. Furthermore, let $v_1, \ldots, v_s$ denote the vertices of B which have one of the states in $\{\text{I2}, \text{E2}, \text{I1}, \text{E1}\}$, such that $st(v_1) \succeq st(v_2) \succeq \cdots \succeq st(v_s)$. Then clearly $s \leq 4$. We have to show that $(st(v_1), \ldots, , st(v_s)) \in S_{st(v_1)}$ and that $\text{cond}((st(v_1), \ldots, st(v_s)))$ holds. If $s = 0$, then this is clear.

Suppose $s > 0$, let H be the component of G_{T} which contains v_1. If $st(v_1) = \text{I2}$, then v_1 is an inner vertex of the path P_H, and it follows from Lemma 3.14 that the component of $G[V - \{v_1\}]$ which contains vertices of B must have pathwidth one. It can easily be checked that if this is the case, then $st(B) \in S_{\text{I2}}$ and $\text{cond}(st(B))$ holds.

Suppose $st(v_1) \in \{\text{E2}, \text{I1}, \text{E1}\}$. Vertex v_1 is end point of $P_H = (u_1, \ldots, u_p)$. Lemma 3.18 shows that $st(B) \in S_{st(v_1)}$ and that $\text{cond}(st(B))$ holds. □

Definition 3.9. *Let G be a partial two-path, B a biconnected component of $\bar{G}$, and $(\mathcal{C}, \mathcal{S})$ a correct path of chordless cycles for B, $\mathcal{C} = (C_1, \ldots, C_p)$, $\mathcal{S} = (e_1, \ldots, e_{p-1})$. Let $v_1, \ldots, v_s$ denote the vertices of B which do not have state* N *or* S, *such that $st(v_i) \succeq st(v_{i+1})$ for each i, $1 \leq i < s$, suppose $s \geq 1$ and $st(v_1) = \text{E2}$. Let G' be the component of $G[V - \{v_1\}]$ which contains $V(B) - \{v_1\}$.* $\text{cond}_1(st(B))$ *is defined as follows.*

$$\text{cond}_1(st(B)) \Leftrightarrow \text{cond}((\text{I2}, st(v_2), \ldots, st(v_s)))$$

Note that if $st(v_1) = \text{E2}$ and $\text{cond}_1(st(B))$ holds, then also $\text{cond}(st(B))$ holds. In Figure 17, pictures are given of $\text{cond}_1(st(B))$ for all values of $st(B)$.

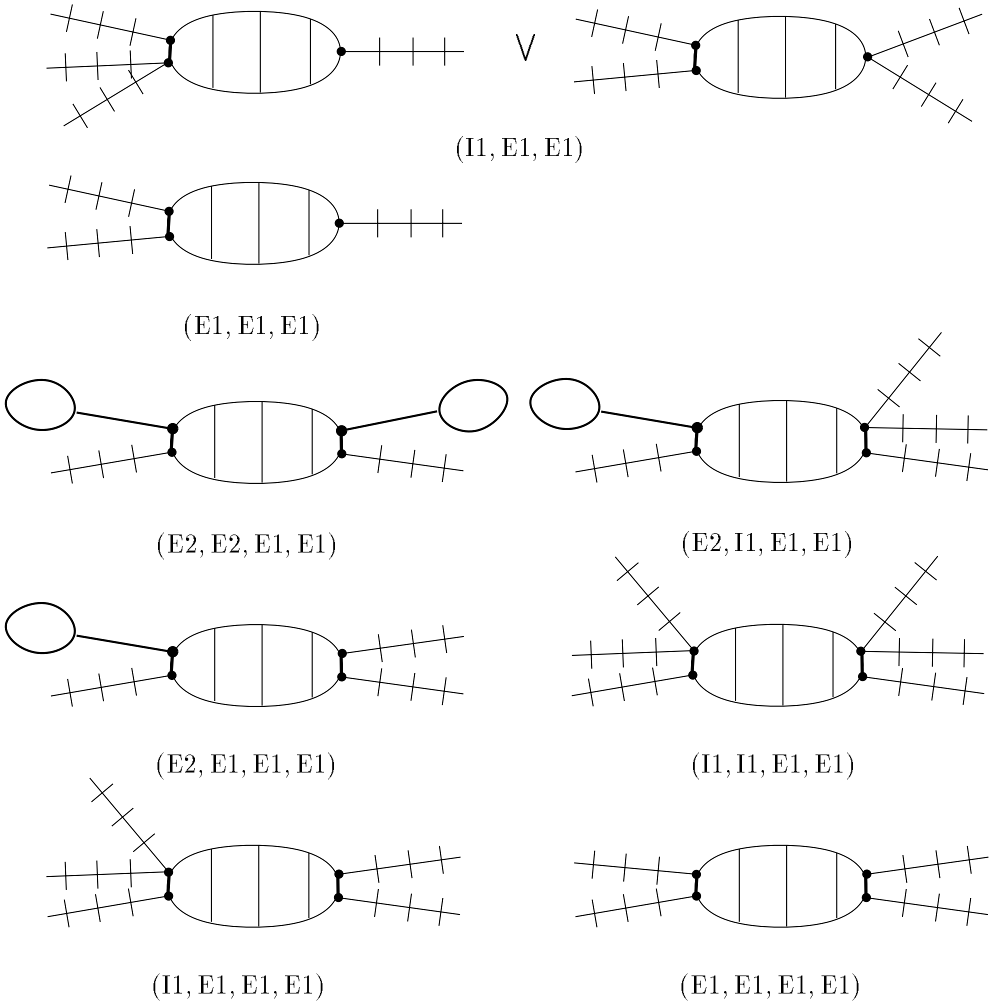

Figure 16: Symbollic representation of $\mathrm{cond}(S)$ for some possible biconnected component state S for $s = 3$ and all possible states for $s = 4$. Cases that are symmetrical in C_1 and C_p, or in distinct vertices v_i with the same state are given only once.

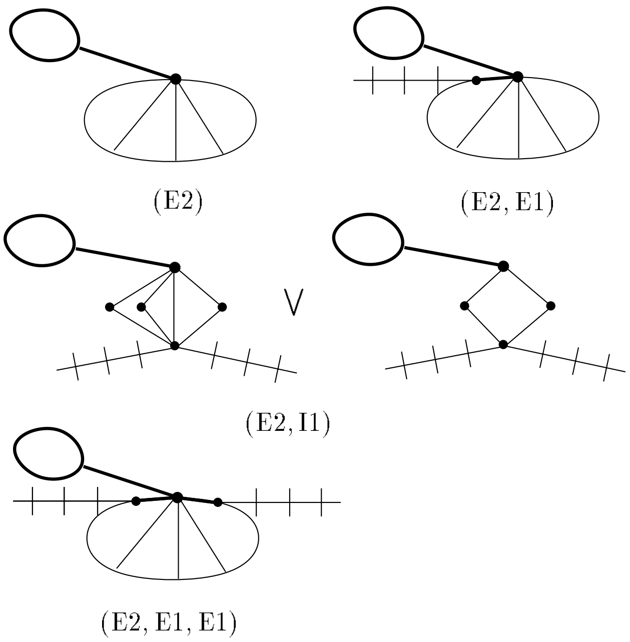

Figure 17: Symbollic representation of $\mathrm{cond}_1(S)$ for possible biconnected component state $S = (st_1, ..., st_s)$ with $st_1 = \mathrm{E2}$. Cases that are symmetrical in C_1 and C_p, or in distinct vertices v_i with the same state are given only once.

Theorem 3.2. *Let G be a graph. G is a partial two-path if and only if the following holds.*

1. *Each component H of G_{T} has pathwidth at most two, and there is a path in H which contains all vertices that are in a biconnected component of G and a path of $\mathcal{P}_2(H)$.*

2. *Each biconnected component B of $\bar{G}$ contains only vertices which have one of the states* I2, E2, I1, E1, S *and* N, *and at most four vertices of B do not have state* S *or* N.

3. *Each biconnected component of $\bar{G}$ can be written as a correct path of chordless cycles.*

4. *Each biconnected component B of $\bar{G}$ has one of the states in $S_{\mathrm{I2}} \cup S_{\mathrm{E2}} \cup S_{\mathrm{I1}} \cup S_{\mathrm{E1}} \cup \{()\}$ and satisfies $\mathrm{cond}(st(B))$.*

5. *Let H be a component of G_{T}, suppose $G \neq H$, let $P_H = (u_1,\ldots,u_p)$. If $p > 1$ and u_1 is a vertex of a biconnected component and $st(u_1) = \mathrm{E2}$, then at most one of the biconnected components which contain u_1 does not satisfy $\mathrm{cond}_1(st(B))$. Similar for u_p.*

   If $p = 1$, u_1 is a vertex of a biconnected component and $st(u_1) = \mathrm{E2}$, then at most two biconnected components containing u_1 do not satisfy $\mathrm{cond}_1(st(B))$.

Proof. We first prove the 'if' part. Suppose G is a partial two-path, then it follows directly from Lemmas 3.13, 3.16, and 3.19 that 1, 2, 3 and 4 hold.

We now prove that 5 holds. Let H be a component of G_{T}, suppose $G \neq H$, let $P_H = (u_1,\ldots,u_p)$. Suppose $u_1 \in V(B)$ for some biconnected component of G, and $st(u_1) = \mathrm{E2}$. If $p > 1$ and u_1 is a vertex of a biconnected component and $st(u_1) = \mathrm{E2}$, then at most one of the biconnected components which contain u_1 does not satisfy $\mathrm{cond}_1(st(B))$. Similar for u_p. If $p > 1$, then, according to Lemma 3.14, there may be at most one component in $G' = G[V(G) - V(P_H)]$ which has pathwidth two and which is adjacent to u_1 in G. This means that at most one biconnected component B containing u_1 is allowed not to satisfy $\mathrm{cond}_1(st(B))$, since $\mathrm{cond}_1(st(B))$ holds if the component of $G[V - \{u_1\}]$ which contains $V(B) - \{u_1\}$ has pathwidth one, as is shown in the proof of Lemma 3.19. If $p = 1$, then in the same way, we can show that at most two biconnected components B containing u_1 are allowed not to satisfy $\mathrm{cond}_1(st(B))$.

Now we prove the 'only if' part. Suppose G is a connected graph, which satisfies conditions 1, 2, 3, 4 and 5. If G is a tree or G is biconnected, then G has pathwidth two, as is shown in Theorem 3.1 and Lemma 3.7. Suppose G_{T} is not empty and G contains at least one biconnected component. We construct a path decomposition of width two of G.

First consider G_{T}. Let H be a component of G_{T}. Let $P_H = (u_1,\ldots,u_p)$. If $p = 1$ and $st(u_1) = \mathrm{E1}$, then make a path decomposition PD_H of width one of H in which u_1 is in the rightmost node. If $p = 1$ and $st(u_1) = \mathrm{I1}$, then make a path decomposition PD_H of width one of H. Let H_1 and H_2 be the components of $H[V - \{v_1\}]$ which contain edges of H, such that the leftmost node of PD_H contains vertices of H_1 and the rightmost node contains vertices of H_2. Let $PD_H^1 = PD[V(H_1) \cup \{v_1\} \cup \{\text{sticks of } v_1\}]$, and $PD_H^2 = PD[V(H_2) \cup \{v_1\}]$. Note that v_1 is in the rightmost node of PD_H^1 and in the leftmost node of PD_H^2. Furthermore, make a path decomposition PD'_H of width two of H, which is similar to PD_H, but with vertex v_1 added to each node.

In the final path decomposition of G, PD'_H is used if component H may occur completely on the same side of the biconnected component which contains v_1, and PD_H^1 and PD_H^2 are used if two parts of H must occur on different sides. In this case, PD_H^1 occurs on the left side and PD_H^2 on the right side.

If $p > 1$, or $p = 1$ and $st(u_1) \succeq$ E2, then do the following. Let G_H denote the induced subgraph of G which contains H and all components of $G[V - V(P_H)]$ which have pathwidth zero or one. For each u_i, each component of $G_H[V(G_H) - V(P_H)]$ which is connected to u_i, make a path decomposition of width zero or one, and add u_i to each node of this path decomposition. For each u_i, concatenate the obtained path decompositions of all components which are connected to u_i, and let PD_i denote this path decomposition. Now make the following path decomposition: $PD_H = PD_1 \mathbin{+\!\!+} (\{u_1, u_2\}) \mathbin{+\!\!+} PD_2 \mathbin{+\!\!+} \cdots \mathbin{+\!\!+} (\{u_{p-1}, u_p\}) \mathbin{+\!\!+} PD_p$. See for example Figure 18.

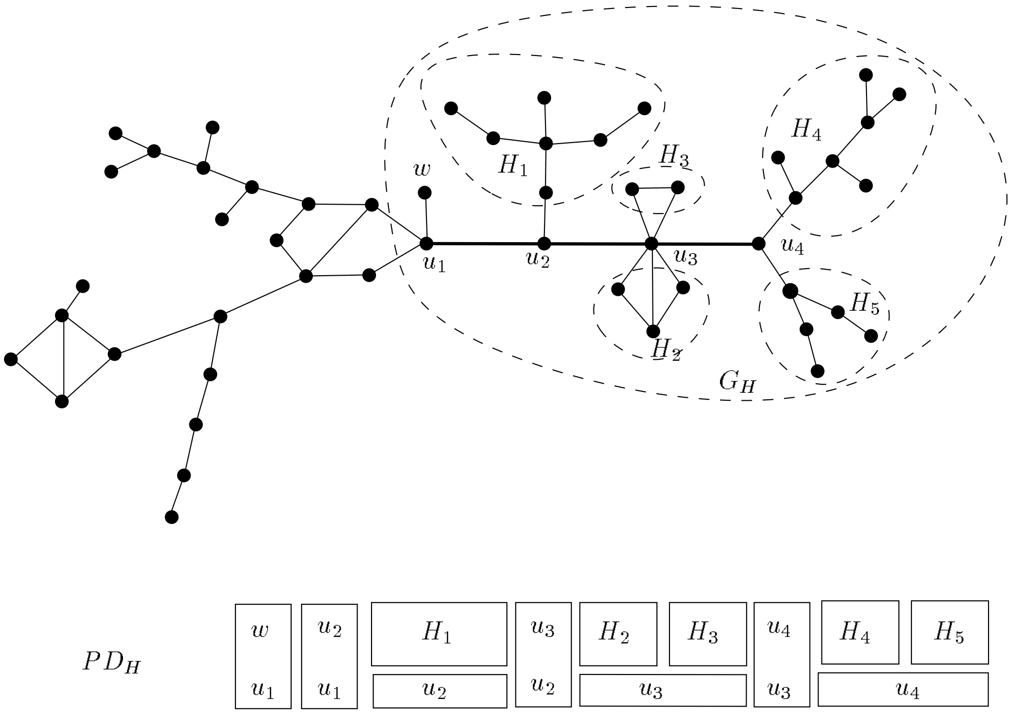

Figure 18: Example of the construction of PD_H if the path P_H has more than one vertex. In the picture, H is the component of G_{T} which contains u_1, and H_1,..., H_5 are the components of G_H which have pathwidth one.

PD_H is a path decomposition of width two of the graph G_H. Furthermore, the leftmost node of PD_H contains u_1, the rightmost node contains u_p. There are at most two components of $G[V - V(P_H)]$ which have pathwidth two, and if $p > 1$, then at most one of these components is connected to u_1, and at most one to u_p.

Now consider the biconnected components which are not contained in some G_H for H a component of G_{T}. For each biconnected component B of $\bar{G}$ for which this holds, let $(\mathcal{C}, \mathcal{S})$ be a correct path of chordless cycles with $\mathcal{C} = (C_1, ..., C_p)$ and $\mathcal{S} = (e_1, ..., e_{p-1})$. Let $v_1, ..., v_s$ denote the vertices of B which have one of the states in $\{\mathrm{E2}, \mathrm{I1}, \mathrm{E1}\}$. Note that B has no vertices with state I2, since then B would be in some graph G_H, where H is a component of G_{T}. Let G_B denote the subgraph of G which contains B and all sticks of B which are adjacent to vertices with state S.

If $s = 0$, then make a path decomposition of width two of G_B as follows. First make a path decomposition PD_B of width two of B, as is shown in the proof of Theorem 3.1, but add one node on the left side which contains one of the edges in the former leftmost node, and add one node on the right side which contains one of the edges in the former rightmost node. For each $v \in V(B)$ which has state S, do the following. If B consists of more than three vertices, then it can be seen that there are two nodes V_i and V_{i+1}, such that $V_i \cap V_{i+1} = \{v, u\}$ for some $u \neq v$. See e.g. Figure 3. For each stick w adjacent to v, we add a node $\{w, v, u\}$ between V_i and V_{i+1}. If B has three vertices, let $V(B) = \{w_1, w_2, w_3\}$. Then $PD_B = (\{w_1, w_2, w_3\})$. Then we can make a path decomposition of width two of G_B by adding on the left side for each stick w of w_1 or w_2 a node $\{w_1, w_2, w\}$, and on the right side for each stick w of w_3 a node $\{w_3, w\}$.

If $s > 1$, then make a path decomposition of G_B in the same way as for $s = 0$, but with the appropriate vertices of $\{v_1, ..., v_s\}$ occurring in the leftmost and rightmost node. It can be derived from the pictures of all conditions (see Figures 13, 14, 15, and 16 which vertex must occur on which side; e.g. if $v_1 \in V(C_1)$ and the component H of G_{T} which contains v_1 is drawn on the left side of the biconnected componentin the picture representing this case, then v_1 must occur in the leftmost node, but if $st(v_1) = \mathrm{I1}$, $v_1 \in V(C_1) \cap V(C_p)$ and part of H is drawn on left side of the biconnected component, and the other part is drawn on the right side, then v_1 must occur in both end nodes of the path decomposition. Note that this is well possible, since in the conditions, the distance between two vertices v_i and v_j of which the components must occur on the same side must be small enough.

If all these path decompositions are made, then they can be combined rather straightforwardly into a path decomposition of width two of G. In Figure 19, an example is given of how the combination is done. □

For a given graph G, conditions 1, 2, 3, 4 and 5 can be checked in linear time: conditions 1 and 3 can be checked in linear time in the way that is shown in Section 3.1 and Section 3.2. All other conditions can straightforwardly be checked in linear time.

Let G be a connected three-colored partial two-path, which is not a tree. We now extend the definition of the path P_H for all components H of G_{T} to the path P_G. Consider the set $\mathcal{H}$ of all components of G_{T} which contain a vertex w of a biconnected component which has state I2 or E2. Each biconnected component has at most one

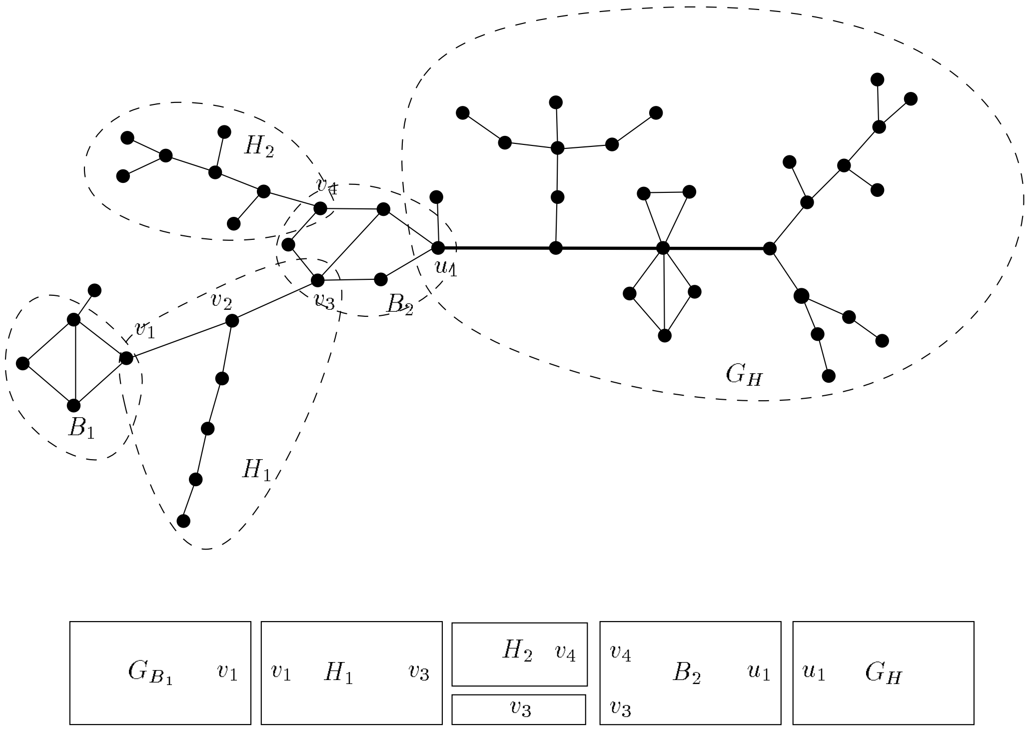

Figure 19: Example of the construction of a path decomposition of width two of a partial two-path G, after the path decompositions of all components of G_{T} and all biconnected components, including their sticks, are constructed as in the proof of Theorem 3.2.

vertex with state I2, at most two vertices with state E2, and if it has a vertex with state I2, then it has no vertices with state E2. This means that we can give the following definition.

Definition 3.10. *Let G be a connected partial two-path which is not a tree. Let $\mathcal{H}$ be the set of all components of G_{T} which contain a vertex w of a biconnected component which has state* I2 *or* E2*, let $\mathcal{B}$ be the set of biconnected components of G. The* path *P_G of G is a graph which is defined as follows.*

$$
\begin{aligned}
V(P_G) &= \bigcup_{H \in \mathcal{H}} V(P_H) \\
E(P_G) &= \{\, e \in E(G) \mid \exists_{H \in \mathcal{H}}\, e \in E(P_H) \,\} \cup \\
&\quad \{\, \{v, v'\} \mid \exists_{H,H' \in \mathcal{H}, B \in \mathcal{B}}\, H \neq H' \wedge v \in V(P_H) \wedge v' \in V(P_H) \wedge v, v' \in V(B) \,\}
\end{aligned}
$$

Note that P_G is unique if G is not a tree, since if G is not a tree, then each component H of G_{T} has at least one vertex in a biconnected component, and hence $|\mathcal{P}_H| = 1$. $V(P_G)$ may be empty in the case that G contains only one biconnected component. Note furthermore that P_G is in fact a concatenation of all paths P_H of trees $H \in \mathcal{H}$, in such a way that two paths which have an end point in a common biconnected component are directly concatenated in P_G. P_G is not a real path of G, but it is the largest common subsequence of all paths in G between the two end points of P_G. The biconnected components of G which contain two vertices of P_G are called *connecting* biconnected components. All other biconnected components are called non-connecting biconnected components.

In each path decomposition $PD = (V_1, \ldots, V_t)$ of width two of G, the occurrences of the paths P_H, $H \in \mathcal{H}$, do not overlap, since they have no vertices in common. Furthermore, they occur in the same order as in P_G or in reversed order, because they are connected to each other by biconnected components, which have pathwidth two.

We show the analog of Corollary 3.2 for general partial two-paths.

Lemma 3.20. *Let G be a connected partial two-path, not a tree. Let $P_G = (v_1, \ldots, v_s)$. Let $PD = (V_1, \ldots, V_t)$ be a path decomposition of width two of G. For each $v \in V_1$, $v' \in V_t$, the path from v to v' contains P_G as a subsequence.*

Proof. If $|V(P_G)| = 0$, the result clearly holds. Suppose $|V(P_G)| \geq 1$. Let G_1 be the subgraph of G induced by vertex v_1 and the components of $G[V - \{v_1\}]$ which do not contain vertices of P_G. Similarly, let G_2 be the subgraph of G induced by v_s and the components of $G[V - \{v_s\}]$ which do not contain vertices of P_G. We prove the lemma by proving that $V_1 \subseteq V(G_1)$ and $V_s \subseteq V(G_2)$ or vice versa, and if $s = 1$, then V_1 and V_t do not contain vertices of the same component of $G[V - \{v_1\}]$.

Suppose V_1 contains a vertex $v \notin V(G_1) \cup V(G_2)$. We distinguish two cases.

1. $v \in V(P_G) - \{v_1, v_s\}$ or there is an inner vertex v' of P_G such that v is a vertex of a component of $G[V - \{v'\}]$ which does not contain vertices of P_G.

2. $v \notin V(P_G)$ and there is a connecting biconnected component B of G such that v is in the component of $G[V - V(P_G)]$ which contains vertices of B. (recall that a connecting biconnected component is a biconnected component which contains two vertices of P_G.)

First suppose case 1 holds. Let $v' \in V(P_G)$ such that either $v = v'$ or v is in a component of $G[V - \{v'\}]$ which does not contain vertices of P_G. Let G' and G'' denote the components of $G[V - \{v'\}]$ which contain vertices of P_G. G' and G'' have pathwidth two, hence there are nodes V_j and $V_{j'}$ in PD such that V_j contains three vertices of G' and $V_{j'}$ contains three vertices of H_2. Suppose w.l.o.g. that $j < j'$. Then V_j contains a vertex of $G[V - V(G')]$, since V_1 contains v, and $V_{j'}$ contains vertices of G''. Contradiction.

Next suppose case 2 holds. Let B be the biconnected component of G for which v is in the component of $G[V - V(P_G)]$ which contains a vertex of B. Let i, $1 \leq i \leq s$, be such that $v_i, v_{i+1} \in V(B)$. Let G' be the subgraph of G induced by v_i and the component of $G[V - \{v_i\}]$ containing G_1. Similarly, let G'' be the subgraph of G induced by v_{i+1} and the subgraph of $G[V - \{v_{i+1}\}]$ containing G_2. In the same way as for case 1, we can derive a contradiction.

We next show that V_1 and V_t can not both contain a vertex of G_1, unless $s = 1$. Suppose $s > 1$ and $v \in V_1$, $v' \in V_t$ such that $v, v' \in V(G_1)$. G_2 has pathwidth two, which means that there is a node V_j, $1 \leq j \leq t$, such that V_j contains three vertices of G_2. But V_j also contains a vertex of G_1, which is a contradiction. In the same way, we can prove that if $s = 1$, then V_1 and V_t can not both contain a vertex of the same component of $G[V - \{v_1\}]$. □

4 Algorithm for Intervalizing Three-Colored Graphs

In this section, we give an algorithm for determining whether there is an intervalization of a given three-colored graph. The main algorithm has the following form: first the structure of G is determined, as described in Section 3, and then the algorithms of this section are used.

We first give an algorithm for biconnected graphs. After that, we give an algorithm for partial two-paths which consist of a biconnected component with sticks, i.e. each vertex v of the biconnected component has state N or state S. The algorithm for graphs of this kind is used for the tree algorithm, which is given thereafter. In the last subsection, we construct an algorithm for general graphs, by combining the other algorithms.

4.1 Biconnected Graphs

To make a proper path decomposition of a properly three-colored biconnected partial two-path G, we can make proper path decompositions of the chordless cycles of $\bar{G}$, thereby taking into account which edges of each chordless cycle are shared with other chordless cycles: these are the end edges of the chordless cycle. The proper path decompositions of the chordless cycles can then be concatenated in the order in which they occur in the path of chordless cycles of G, and this gives a proper path decomposition of G.

Hence we concentrate now on checking whether there exists a proper path decomposition of a chordless cycle C. Let C be a three-colored chordless cycle. We first give some notations. We denote the vertices and edges of C by $V(C) = \{v_0, v_1, \ldots, v_{n-1}\}$, and $E(C) = \{\, \{v_i, v_{(i+1) \bmod n}\} \mid 0 \leq i < n\,\}$. For each j, by v_j we denote vertex $v_{j \bmod n}$. For each j, l, $(j - l) \bmod n \neq 0$, let $I(j,l)$ denote the set of vertices of $V(C)$ between v_l and v_j, to be precise, those seen when going from v_j to v_l in negative direction, i.e.,

$$\begin{aligned} I(j,l) \;=\; & \{\, v_i \mid (j \bmod n > l \bmod n \wedge l \leq i \leq j) \\ & \vee\ (j \bmod n < l \bmod n \wedge (l \leq i < n \vee 0 \leq i \leq j))\,\}, \end{aligned}$$

Furthermore, let $C(j,l)$ denote the cycle with

$$\begin{aligned} & V(C(j,l)) = I(j,l) \\ & E(C(j,l)) = \{\{v_j, v_l\}\} \cup \{\, \{v_i, v_{i+1}\} \mid v_i \in I(j,l) - \{v_j\}\,\} \end{aligned}$$

Figure 20 shows an example of a chordless cycle C and some examples of $C(j,l)$. Note that if $(j-l) \bmod n = 1$ then by definition $C(j,l)$ is a cycle consisting of two edges between two vertices. The following lemma is used to obtain a dynamic programming algorithm for our problem.

Lemma 4.1. *Let C be a properly three-colored cycle, suppose $|V(C)| \geq 3$. Let j and l be integers. There is a proper path decomposition $PD = (V_1, \ldots, V_t)$ of C such that $\{v_l, v_{l+1}\} \subseteq V_1$ and $\{v_j, v_{j+1}\} \subseteq V_t$ if and only if there is a proper path decomposition*

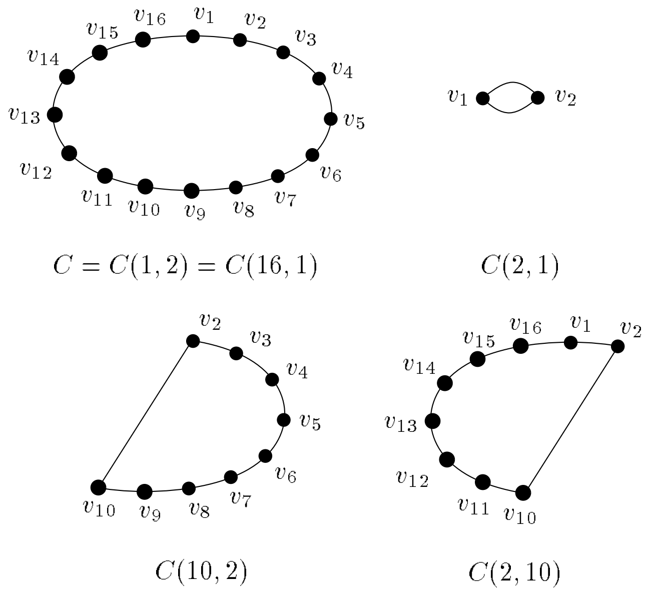

Figure 20: A chordless cycle C, and the cycles $C(2,1)$, $C(10,2)$ and $C(2,10)$. C is equal to $C(j, j+1)$ for all j.

$PD' = (V'_1,\ldots,V'_r)$ of $C(j, j+2)$ such that $\{v_l, v_{l+1}\} \subseteq V'_1$ and $\{v_j, v_{j+2}\} \subseteq V'_r$ or there is a proper path decomposition $PD' = (V'_1,\ldots,V'_r)$ of $C(j-1, j+1)$ such that $\{v_l, v_{l+1}\} \subseteq V'_1$ and $\{v_{j-1}, v_{j+1}\} \subseteq V'_r$.

Proof. For the 'if' part, suppose there is a proper path decomposition $PD' = (V'_1,\ldots,V'_r)$ of $C(j, j+2)$ with $\{v_l, v_{l+1}\} \subseteq V'_1$ and $\{v_j, v_{j+2}\} \subseteq V'_r$. Then $PD = PD' + (\{v_j, v_{j+1}, v_{j+2}\})$ is a proper path decomposition of C which satisfies the appropriate conditions. The other case is similar.

For the 'only if' part, suppose there is a proper path decomposition $PD = (V_1,\ldots,V_t)$ of C such that $\{v_l, v_{l+1}\} \subseteq V_1$ and $\{v_j, v_{j+1}\} \subseteq V_t$. Let V_m and $V_{m'}$, $1 \leq m, m' \leq t$, be the rightmost nodes containing edge $\{v_{j-1}, v_j\}$ and $\{v_j, v_{j+1}\}$, respectively. First suppose $m' \leq m$. Then $V_m = \{v_{j-1}, v_j, v_{j+1}\}$. Furthermore, for each i, $m < i \leq t$, $V_i = \{v_j, v_{j+1}\}$, since if there is a V_i, $m < i \leq t$, such that $v \in V_i$ for some $v \in V(C) - \{v_j, v_{j+1}\}$, then $v \in V_m$, which gives a contradiction. Let PD' be the path decomposition obtained from $(V_1,\ldots,V_m)$ by deleting v_j from all nodes containing it. Then PD' is a proper path decomposition of $C(j-1, j+1)$ with edge $\{v_{j-1}, v_{j+1}\}$ in the rightmost node and edge $\{v_l, v_{l+1}\}$ in the leftmost node. For the case that $m \leq m'$, we get a path decomposition for $C(j, j+2)$ in the same way. □

Let G be a biconnected partial two-path, $(\mathcal{C}, \mathcal{S})$ a path of chordless cycles of $\bar{G}$ with $\mathcal{C} = (C_1,\ldots,C_p)$ and $\mathcal{S} = (e_1,\ldots,e_{p-1})$. Let $1 \leq i \leq p$. We try to make a proper path decomposition PD of G such that the chordless cycles of $\mathcal{C}$ occur in the same order in PD as in $\mathcal{C}$. If $i > 1$, then we want edge e_i to occur in the leftmost node of the proper path decomposition of C_i. If $i = 1$, then any edge of C_i may occur in the leftmost end node. The set of edges of which one must occur in the leftmost end node of the proper path decomposition of C_i is called the set of *starting edges*, and is denoted by E_{S}. So if $i > 1$, then $E_{\mathrm{S}} = \{e_i\}$, and if $i = 1$, then $E_{\mathrm{S}} = \{E(C_i)$. In the same way we define the set of *ending edges* E_{E}, which is the set of edges of which one must occur in the rightmost end node of the proper path decomposition of C_i. So if $i < p$, then $E_{\mathrm{E}} = \{e_{i-1}\}$, and if $i = p$, then $E_{\mathrm{E}} = E(C_p)$. Note that if $p = 1$, then the set of starting edges and the set of ending edges for C_1 both consist of $E(C_1)$.

If $|V(C_i)| = 3$, then there is a proper path decomposition if and only if C_i is properly colored, and this path decomposition can consists of one node, namely $(V(C_i))$.

We define $PPW2$ as follows. Let $E_{\mathrm{S}} \subseteq E(C)$ be a set of starting edges, let $(j - l) \bmod n \neq 0$.

$$PPW2(C, E_{\mathrm{S}}, j, l) = \begin{cases} \text{true} & \text{if } \exists_{PD=(V_1,..,V_t)}\ PD \text{ is a proper path decomposition} \\ & \text{of } C(j,l)\ \wedge v_j, v_l \in V_t \wedge \exists_{e \in E_{\mathrm{S}}}\ e \subseteq V_1 \\ \text{false} & \text{otherwise} \end{cases}$$

Let C be a three-colored chordless cycle which is properly colored, $E_{\mathrm{S}} \subseteq E(C)$. From the definition we can see that $PPW2(C, E_{\mathrm{S}}, j, j-1)$ is true if and only if edge $\{v_j, v_{j-1}\} \in E_{\mathrm{S}}$. We use Lemma 4.1 to describe $PPW2$ recursively. Let $E_{\mathrm{S}} \subseteq E(C)$, $(j - l) \bmod n \neq 0$.

$$PPW2(C, E_{\mathrm{S}}, j, l) = \begin{cases} \{v_j, v_l\} \in E_{\mathrm{S}} & \text{if } j - l \bmod n = 1 \\ c(v_j) \neq c(v_l)\ \wedge & \\ (\ PPW2(C, E_{\mathrm{S}}, j-1, l) \vee PPW2(C, E_{\mathrm{S}}, j, l+1)\ ) & \text{if } j - l \bmod n > 1 \end{cases}$$

For a given properly three-colored cycle C, $|V(C)| = n$, and set of starting edges $E_{\mathrm{S}} \subseteq E(C)$, and ending edges $E_{\mathrm{E}} \subseteq E(C)$, we can compute whether there is a proper path decomposition of C with these starting and ending edges in $O(n^2)$ time using dynamic programming with the following function as follows.

function COMP_PPW2(C,n,c,E_{S},E_{E})
  var i : int;
    $P : \{0,...,n-1\} \times \{1,...,n-1\} \rightarrow \{\text{true}, \text{false}\}$;
  {P denotes $PPW2$ as follows: $P(j,k) \equiv PPW2(C, E_{\mathrm{S}}, j, j-k)$ at the end }

  $n := |V(C)|$;
  for $j := 0$ to $n-1$
  $\rightarrow$ $P(j,1) :=$ false;
  rof;
  for all $\{v_j, v_{j-1}\} \in E_{\mathrm{S}}$
  $\rightarrow$ $P(j,1) :=$ true;
  rof;

```
  for k := 2 to n − 1
  →  for j := 0 to n − 1
       →  P(j,k) := c(v_j) ≠ c(v_{j−k}) ∧
                    ( P((j − 1) mod n, k − 1) ∨ P(j, k − 1) )
     rof
  rof;
  for all {v_j, v_{j+1}} ∈ E_E
     if   P(j, n − 1) → return true
     □   ¬P(j, n − 1) → skip
     fi
  rof;
  return false
end
```

Let G be a biconnected partial two-path, $(\mathcal{C}, \mathcal{S})$ a path of chordless cycles of G with $\mathcal{C} = (C_1,..., C_p)$. There is a proper path decomposition of G if and only if for each i, $1 \leq i \leq p$, there is a proper path decomposition of C_i with set of starting edges $\{e_i\}$ if $i > 1$, $E(C_i)$ otherwise, and set of ending edges $\{e_{i+1}\}$ if $i < p$, $E(C_i)$ otherwise.

For a given three-colored biconnected graph G, the algorithm is now as follows.

1. Find the cell completion $\bar{G}$ of G and check if $\bar{G}$ is properly three-colored. If not, stop, the answer is no.
2. Check if $\bar{G}$ can be written as a path of chordless cycles. If so, construct such a path $(\mathcal{C}, \mathcal{S})$ with $\mathcal{C} = (C_1,..., C_p)$ and $\mathcal{S} = (e_1,..., e_{p-1})$. If not, stop, the answer is no.
3. For each chordless cycle C_i in the path, let $m = |V(C_i)|$, let $E_{\mathrm{S}} = \{e_{i-1}\}$ if $i > 1$, otherwise $E_{\mathrm{S}} = E(C_i)$, and let $E_{\mathrm{E}} = \{e_{i+1}\}$ if $i < p$, $E_{\mathrm{E}} = E(C_i)$ otherwise. Compute $COMP\_PPW2(C_i, m, c, E_{\mathrm{S}}, E)$. If the computed value is true for each C_i, the answer is yes, otherwise it is no.

Step 1 and 2 run in $O(n)$ time, step 3 runs in $O(n^2)$ time, where $n = |V(G)|$.

The algorithm can be made constructive, in the sense that if there exists an intervalization, then the algorithm outputs one, as follows. In function COMP_PPW2, construct an array PP of pointers, such that $PP(j,k)$ contains the nil pointer if $k = 1$ or if $P(j,k)$ is false, and if $P(j,k)$ is true and $k > 1$, then $PP(j,k)$ contains a pointer to $PP(j,k-1)$ or to $PP((j-1) \bmod n, k-1)$. It contains a pointer to $P(j,k-1)$ if $P((j-1) \bmod n, k-1)$ is false, and a pointer to $P((j-1) \bmod n, k-1)$ if $P(j,k-1)$ is false, and arbitrarily to $P(j,k-1)$ or $P((j-1) \bmod n, k-1)$ if $P((j-1) \bmod n, k-1)$ and $P(j,k-1)$ are both true. The computation of PP can be done during the computation of P. Afterwards, if there is an intervalization, then one can be constructed as follows. Start with a j, $0 \leq j < n$ for which $\{v_j, v_{j+1}\} \in E_{\mathrm{E}}$ and $P(j, n-1)$ is true. Then follow the pointers from $PP(j, n-1)$ until the nil pointer is reached, and add edge $\{v_i, v_{i-k}\}$ for each i and k for which $PP(i,k)$ is passed. Note that the nil pointer is reached if the previous pointer pointed to $PP(i,1)$ for some i such that $\{v_i, v_{(i-1) \bmod n}\} \in E_{\mathrm{S}}$.

4.2 Biconnected Partial Two-Paths with Sticks

Before giving an algorithm for trees, we first give an algorithm for partial two-paths which consist of a biconnected component with sticks connected to it. A biconnected component with sticks is a connected graph $G = (V, E)$ which contains one biconnected component B, and all vertices in the set $W = V - V(B)$ are adjacent to exactly one vertex, which is in $V(B)$. The vertices in W are the sticks. The algorithm for biconnected components with sticks will be used for the tree algorithm, and for the algorithm for general partial two-paths.

The algorithm for biconnected components with sticks is derived from the algorithm for biconnected components. Therefore, we first consider chordless cycles with sticks.

Cycles with Sticks

Let G be a properly colored graph, which consists of a cycle C and sticks connected to the vertices of C. Let $v \in V(C)$ and i such that $1 \leq i \leq 3$. We show that it is not important how many sticks of color i are connected to vertex v, but we only need to know whether v has sticks of color i. Suppose v has a stick w of color i, and there is a proper path decomposition PD of C in which there is a node V_l with $\{v, w\} \subseteq V_l$. Let G' be the graph obtained from G by adding a stick w' of color i which is connected to v. We can make a proper path decomposition PD' of G' as follows. Remove w from all nodes V_j with $j \neq l$, and add a node W between V_l and V_{l+1} with $W = V_l \cup \{w'\} - \{w\}$. So there is a proper path decomposition of G if and only if there is a proper path decomposition of G'.

Definition 4.1. *Let G be a properly colored graph, which consists of a cycle C and sticks connected to C. For each $v \in V(C)$ and each i, $1 \leq i \leq 3$,*

$$s_i(v) = \textit{true} \Leftrightarrow v \textit{ has a stick of color } i$$

The following lemmas shows the conditions which must hold for three-colored partial two-paths which consist of a chordless cycle with sticks, to have a proper path decomposition.

Lemma 4.2. *Let G be a colored partial two-path consisting of a chordless cycle C with sticks, let $e = \{x, y\} \in E(C)$ and $e' = \{x', y'\} \in E(C)$. Suppose there is path from x to x' which does not contain y or y', and let P_1 denote this path. Let P_2 denote the path from y to y' which does not contain x or x'. There is a proper path decomposition $PD = (V_1, ..., V_t)$ of G such that $e \subseteq V_1$ and $e' \subseteq V_t$ if and only if there is a proper path decomposition $PD' = (V_1', ..., V_r')$ of C such that $e \subseteq V_1'$ and $e' \subseteq V_r'$ and PD' contains no two subsequent nodes which are the same, and for each vertex $v \in V(P_i)$, $i \in \{1, 2\}$,*

$$\forall_{j \in \{1,2,3\}} \left(s_j(v) \Rightarrow \exists_{1 \leq l \leq r} \exists_{v' \in V(P_{2-i})} c(v') \neq j \wedge v \in V_l' \wedge v' \in V_l' \right). \qquad (1)$$

Proof. For the 'if' part, suppose $PD = (V_1, ..., V_t)$ is a proper path decomposition of C, such that $e \subseteq V_1$, $e' \subseteq V_t$, and for all $v \in V(C)$, the conditions stated in the lemma are satisfied. We transform PD into a proper path decomposition of G with

e in the leftmost node, and e' in the rightmost node by doing the following for each vertex $v \in V(C)$, each $j \in \{1,2,3\}$, and each stick w of v with color j. In Figure 21, an example of this transformation is given. Suppose w.l.o.g. that $v \in V(P_1)$. There is a node V_i, $1 < i < t$, and a vertex $v' \in V(P_2)$ with $c(v') \neq j$ and $\{v, v'\} \notin E(C)$ such that V_i contains v and v'. Let C' be the graph obtained from C by adding edge $\{v, v'\}$. PD is also a proper path decomposition of C'. Furthermore, C' consists of two chordless cycles which have edge $\{v, v'\}$ in common. Lemma 3.3 shows that edge $\{v, v'\}$ is a middle edge of C'. Hence either there is a node $\{v, v'\}$ in PD or we can add such a node to PD. Furthermore, we can add stick w to this node. This completes the proof of the 'if' part.

For the 'only if' part, suppose $PD = (V_1, ..., V_t)$ is a proper path decomposition of G with $e \subseteq V_1$, $e' \subseteq V_t$. We show that $PD' = PD[V(C)] = (V'_1, ..., V'_r)$ is a proper path decomposition of C which satisfies the conditions stated in the lemma. Each node V_i contains at least one vertex of P_1 and at least one vertex of P_2. Let $v \in V(P_1)$, $j \in \{1,2,3\}$, and suppose $s_j(v)$ is true. Let w be a stick of v of color j. Then there is a $v' \in V(P_2)$ and a node V_i, $1 \leq i \leq t$, such that $V_i = \{v, v', w\}$. Hence there is a node $V'_{i'}$ in PD' such that $1 \leq i' \leq r$ and $V'_{i'}$ contains v and v'. This completes the proof of the 'only if' part. □

Lemma 4.3. *Let G be a properly colored partial two-path consisting of a chordless cycle C with sticks. There is a proper path decomposition of G if and only if there are vertices $v, v' \in V(C)$ such that there is a proper path decomposition $PD = (V_1, ..., V_t)$ of the graph $G' = G[V - W]$, where W is the set of sticks of v and v' in G, and V_1 and V_t contain an edge of C, $v \in V_1$ and $v' \in V_t$.*

Proof. For the 'if' part, suppose there are v and v' such that there is a proper path decomposition of $G' = G[V - W]$, where W is the set of sticks adjacent to v and v', such that v is in the leftmost node and v' is in the rightmost node, and the leftmost and rightmost node contain an edge of C. Then we can make a proper path decomposition of G as follows. For each stick w adjacent to v, add a node $\{v, w\}$ in front of the leftmost node. If $v' \neq v$, do the same for v' on the right side of the rightmost node.

For the 'only if' part, suppose there is a proper path decomposition $PD = (V_1, ..., V_t)$ of G. Suppose w.l.o.g. that V_1 and V_t contain an edge. Suppose C occurs in $(V_j, ..., V_{j'})$, $1 \leq j \leq j' \leq t$. We transform PD in such a way that there is at most one $v \in C$ which has a stick w such that $\{v, w\}$ occurs on the left side of V_j, and similar for the right side of $V_{j'}$. If $j = 1$, then there is no stick occurring on the left side of V_j. Suppose $j > 1$. Then there is a $u \in V_1$ and $w \in V_1$ such that $u \in V(C)$, $w \notin V(C)$ and $\{u, w\} \in E(G)$. There is at most one other $u' \in V(C)$ for which there is a stick w' of u' such that $\{w', u'\}$ occurs within $(V_1, ..., V_{j-1})$, since otherwise there would be a node V_i, $1 \leq i < j$ for which $|V_i| > 3$. Suppose there is such a vertex u'. Then V_j contains u and u', but since V_{j-1} also contains u and u', $\{u, u'\} \notin E(G)$, hence there is a $u'' \in V(C)$ such that $V_j = \{u, u', u''\}$ and $\{u, u''\} \in E(C)$ and $\{u', u''\} \in E(C)$. There can be no sticks adjacent to u'', hence we can transform PD as follows: let l and l' be such that $1 \leq l, l' < j$, V_l is the leftmost node of PD containing u and a stick of u, and $V_{l'}$ is the

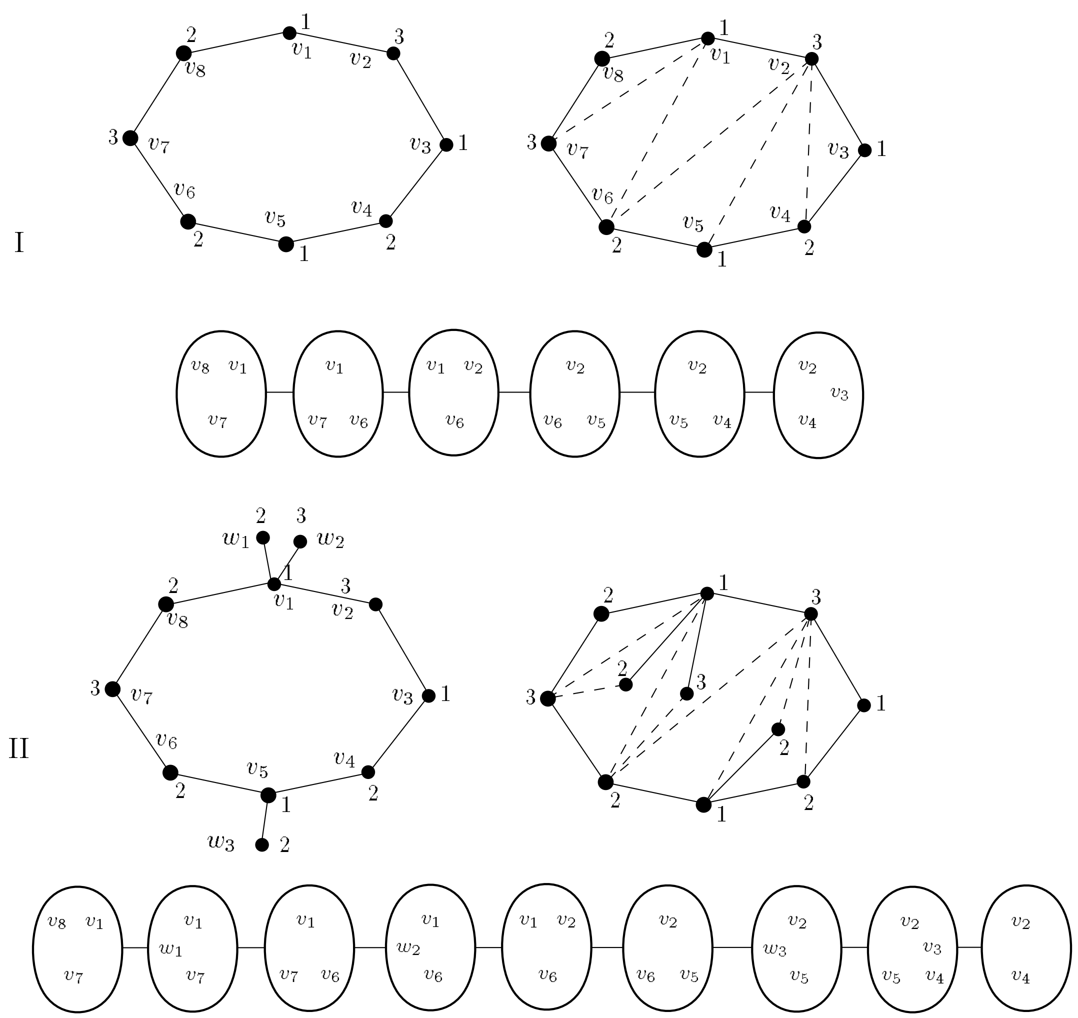

Figure 21: Part I shows a three-colored chordless cycle C, $V(C) = \{v_1,\ldots, v_8\}$, with a proper path decomposition of C and the corresponding interval completion. Each vertex $v_i \in V(C)$ has a color in $\{1, 2, 3\}$, which is given with the vertex. In the path decomposition, edge $\{v_7, v_8\}$ occurs in the leftmost node and edge $\{v_2, v_3\}$ occurs in the rightmost node. with a proper path decomposition of C. In Part II, the chordless cycle C is extended with three sticks w_1, w_2 and w_3, and the proper path decomposition of C is extended into a proper path decomposition for C and its sticks, with edge $\{v_7, v_8\}$ in the leftmost node and edge $\{v_2, v_3\}$ in the rightmost node. The extension is done as is shown in the proof of Lemma 4.2. Note that if, e.g. v_5 would have a stick of color 3, then the proper path decomposition of part I could not have been extended.

leftmost node of PD containing u' and a stick of u'. Delete u and all sticks of u from all nodes in $(V_1,\ldots,V_{l-1})$, and delete u' and all sticks of u' from all nodes in $(V_1,\ldots,V_{l'-1})$. Note that the obtained path decomposition is still a proper path decomposition of G. Suppose w.l.o.g. that $l < l'$. $V_{l'}$ contains u, u' and a stick w of u', but $V_{l'-1}$ does not contain w. Hence we can transform PD as follows. Delete u'' from all nodes, and add a node $\{u, u', u''\}$ between $V_{l'-1}$ and $V_{l'}$. The obtained path decomposition is indeed a proper path decomposition of G, and there is at most one vertex u for which a stick of u occurs on the left side of the occurrence of C. In the same way, we can transform PD such that there is at most one node u for which a stick of u occurs on the right side of the occurrence of C. Now select v and v' as follows. Let $(V_j,\ldots,V_{j'})$ again denote the occurrence of C in (possibly transformed) PD. If $j = 1$, let v be an arbitrary vertex of $V_1 \cap V(C)$. If $j > 1$, let $v \in V_j \cap V(C)$ such that there is a stick w of v such that $\{v, w\}$ occurs on the left side of V_j. Similarly for v' and V_t. Let W be the set of sticks adjacent to v and v'. Then $PD[V - W]$ is a proper path decomposition of $G[V - W]$ in which v occurs in the leftmost node and v' in the rightmost node. □

For each j, l, $j \neq l$, let $G(j, l)$ denote the graph consisting of $C(j, l)$, and the sticks adjacent to all vertices in $I(j, l) - \{v_j, v_l\}$, i.e.

$$
\begin{aligned}
V(G(j,l)) &= V(C(j,l)) \cup \{\, w \in V(G) - V(C) \mid \exists_{v \in V(C(j,l)) - \{v_j, v_l\}} \{v, w\} \in E(G)\,\},\\
E(G(j,l)) &= \{\{v_j, v_l\}\} \cup \{\, \{v, w\} \in E(G) \mid v, w \in V(G(j,l))\,\}.
\end{aligned}
$$

For an example, see Figure 22.

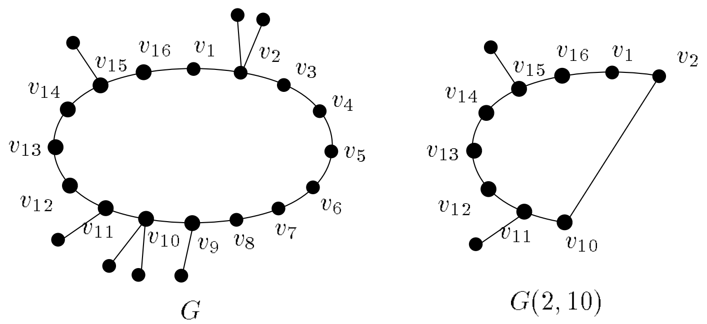

Figure 22: A graph G which consists of a chordless cycle with sticks, and the graph $C(2, 10)$. Note that G is not equal to $G(j, j+1)$ if $v_j$or v_{j+1} has sticks.

We now extend $PPW2$ as follows. Let E_S be a set of starting edges of C. Then for each j, l, $j \neq l$, $PPW2(G, E_S, j, l)$ is a record with fields ft and lt, that are defined as follows.

$$PPW2(G, E_S, j, l).ft =$$

$$\begin{cases} \text{true} & \text{if there is a proper path decomposition } PD = (V_1,\ldots,V_t) \\ & \text{of } G(j,l) \cup \{\{v_j, w\} \in E(G) \mid w \in V(G) - V(C)\} \\ & \wedge\, v_j, v_l \in V_t \wedge \exists_{e \in E_S}\, e \subseteq V_1 \\ \text{false} & \text{otherwise} \end{cases}$$

$PPW2(G, E_S, j, l).lt =$

$$\begin{cases} \text{true} & \text{if there is a proper path decomposition } PD = (V_1,\ldots,V_t) \\ & \text{of } G(j,l) \cup \{\{v_l, w\} \in E(G) \mid w \in V(G) - V(C)\} \\ & \wedge\, v_j, v_l \in V_t \wedge \exists_{e \in E_S}\, e \subseteq V_1 \\ \text{false} & \text{otherwise} \end{cases}$$

We say that, for given j and l, the sticks of v_j are processed if $PPW2(G, E_S, j, l).ft$ is true, and the sticks of v_l are processed if $PPW2(G, E_S, j, l).lt$ is true. Note that, because of Lemma 4.3, there is a proper path decomposition $PD = (V_1,\ldots,V_t)$ of G such that $e \subseteq V_1$ for some $e \in E_S$ if and only if there is a j for which $PPW2(G, E_S, j, j+1).ft \vee PPW2(G, E_S, j, j+1).lt$. We are also interested in some other cases, which will be given later. First we show how $PPW2$ can be described recursively.

Lemma 4.4. *Let G be a properly three-colored partial two-path consisting of a chordless cycle with sticks. Let $(j-l) \bmod n \neq 0$, and let $E_S \subseteq E(C)$ be a set of starting edges. Then $PPW2(G, E_S, j, l)$ can be defined recursively as follows.*

$PPW2(G, E_S, j, l).ft =$

$$\begin{cases} \neg s_{c(v_l)}(v_j) \wedge \{v_j, v_l\} \in E_S & \textit{if } (j-l) \bmod n = 1 \\ c(v_j) \neq c(v_l) \wedge & \\ (\, PPW2(G, E_S, j-1, l).ft \wedge \neg s_{c(v_l)}(v_j) \,\vee & \\ \quad PPW2(G, E_S, j, l+1).lt\,) & \textit{if } (j-l) \bmod n > 1 \end{cases}$$

$PPW2(G, E_S, j, l).lt =$

$$\begin{cases} \neg s_{c(v_j)}(v_l) \wedge \{v_j, v_l\} \in E_S & \textit{if } (j-l) \bmod n = 1 \\ c(v_j) \neq c(v_l) \wedge & \\ (\, PPW2(G, E_S, j-1, l).ft \,\vee & \\ \quad PPW2(G, E_S, j, l+1).lt \wedge \neg s_{c(v_j)}(v_l)\,) & \textit{if } (j-l) \bmod n > 1 \end{cases}$$

Proof. We only prove the lemma for $PPW2(G, E_S, j, l).ft$. The proof for $PPW2(G, E_S, j, l).lt$ is similar. If $(j-l) \bmod n = 1$, then there is a proper path decomposition of $G(j,l)$ with the sticks of v_j with v_j and v_l in the leftmost and the rightmost node if and only if $\{v_j, v_l\} \in E_S$, and no stick of v_j has color $c(v_l)$. Now suppose $(j-l) \bmod n \neq 1$.

For the 'if' part, first suppose $PPW2(G, E_S, j-1, l).ft \wedge \neg s_{c(v_l)}(v_j)$ holds. Let $PD = (V_1,\ldots,V_t)$ be a proper path decomposition of $G(j-1,l)$ and the sticks of v_{j-1} such that $e \subseteq V_1$, $e \in E_S$, and $v_{j-1}, v_l \in V_t$. Let G' be the supergraph of $G(j,l)$ which consists of $G(j,l)$ and all sticks of v_j, i.e. $G' = G(j,l) \cup \{\text{sticks of } v_j\}$. Then we can transform PD into a proper path decomposition of G' such that e is in the leftmost node and v_j and v_l are in the rightmost node as follows. First add a node $V_{t+1} = \{v_{j-1}, v_j, v_l\}$ on the right side of V_t, next for each stick w of v_j, add a node $\{v_j, v_l, w\}$ on the right side of V_{t+1}. This is possible since $c(w) \neq c(v_l)$ for each stick w of v_j. Then the constructed path decomposition satisfies the conditions, hence $PPW2(G, E_{\mathrm{S}}, j, l).ft$ holds.

Next suppose $PPW2(G, E_{\mathrm{S}}, j, l+1).lt$ holds. Let $PD = (V_1,..., V_t)$ be a proper path decomposition of $G(j, l+1)$ and the sticks of v_{l+1} such that $e \subseteq V_1$, $e \in E_{\mathrm{S}}$, and $v_j, v_{l+1} \in V_t$. Then we can make a proper path decomposition of G' as follows. Note that each stick of v_j either has color $c(v_l)$ or $c(v_{l+1})$. First, for each stick w of v_j of color $c(v_l)$, add a node $\{v_j, v_{l+1}, w\}$ on the right side of V_t. Next, add a node $\{v_j, v_{l+1}, v_l\}$ on the right side of these nodes. After that, add a node $\{v_j, v_l, w\}$ for each stick w of v_j which has color $c(v_{l+1})$. This gives the desired path decomposition, and hence $PPW2(G, E_{\mathrm{S}}, j, l).ft$ holds.

For the 'only if' part, suppose $PPW2(G, E_{\mathrm{S}}, j, l).ft$ is true. Again, let G' be the supergraph of $G(j, l)$ which consists of $G(j, l)$ and all sticks of v_j. Let $PD = (V_1,..., V_t)$ be a proper path decomposition of G' such that $e \subseteq V_1$ for some $e \in E_{\mathrm{S}}$ and $v_j, v_l \in V_t$. Then clearly $c(v_j) \neq c(v_l)$. Let V_m and $V_{m'}$, $1 \leq m, m' \leq t$, be the rightmost nodes containing edge $\{v_{j-1}, v_j\}$ and $\{v_l, v_{l+1}\}$, respectively. First suppose $m' \leq m$. Then $PPW2(G, E_{\mathrm{S}}, j-1, l).ft$ holds, since $(V_1,..., V_m)$ is a proper path decomposition of $G(j-1, l)$ and the sticks of v_{j-1}, and $v_{j-1}, v_l \in V_m$. Furthermore, the sticks in G' adjacent to v_j must occur on the right side of V_m, since v_j and its sticks are not in $C(j-1, l)$, and hence can not occur within its occurrence. But all nodes V_i, $m+1 \leq i \leq t$, contain only v_j and v_l of $C(j, l)$, hence all sticks of v_j can not have color $c(v_l)$. Hence $\neg s_{c(v_l)}(v_j)$. In the same way we can show that for the case that $m \leq m'$, $PPW2(G, E_{\mathrm{S}}, j, l+1).lt$ must hold. □

Let G be a partial two-path consisting of a chordless cycle C with sticks. If we want to know whether there is a proper path decomposition of G in which the leftmost node contains one of the edges in E_{S} for some $E_{\mathrm{S}} \subseteq E(C)$, and the rightmost node contains one of the edges in E_{E}, for some $E_{\mathrm{E}} \subseteq E(C)$, then we can use the definition of $PPW2$ in the form as it is given: this proper path decomposition exists if and only if for some j such that $\{v_j, v_{j+1}\} \in E_{\mathrm{E}}$, $PPW2(G, \{e\}, j, j+1).ft$ is true and v_{j+1} has only sticks of color $i \in \{1, 2, 3\}$ with $i \neq c(v_j)$, or $PPW2(G, \{e\}, j, j+1).lt$ is true and v_j has only sticks of color $i \in \{1, 2, 3\}$ with $i \neq c(v_{j+1})$. However, it is also possible that we want to know whether there exists any proper path decomposition of G. In that case, we can not use the definition of $PPW2$ in the form in which it is given above. According to Lemma 4.3, there is a proper path decomposition of G if and only if there is a proper path decomposition of G in which sticks of at most one vertex of C occur on the left side of the occurrence of C, and sticks of at most one vertex of C occur on the right side of the occurrence of C. In that case, we can consider the problem for a given set $V_{\mathrm{S}} \subseteq V(C)$ of starting vertices and a set $V_{\mathrm{E}} \subseteq V(C)$ of ending vertices, whether there exists a proper path decomposition of G in which a vertex of V_{S} occurs in the leftmost node, and a vertex of V_{E} occurs in the rightmost node. We can use a modified version of $PPW2$, which we call $PPW2'$. It is defined as follows.

Let $V_{\mathrm{S}} \subseteq V(C)$ be a set of starting vertices, let $(j-l) \bmod n \neq 0$.

$$PPW2'(G, V_{\mathrm{S}}, j, l).ft = \begin{cases} \text{true} & \text{if there is a proper path decomposition } PD = (V_1, \ldots, V_t) \\ & \text{of } G(j,l) \cup \{\{v_j, v\} \in E(G) \mid v \in V(G) - V(C)\} \\ & \ \wedge\, v_j, v_l \in V_t \wedge \exists_{v \in V_{\mathrm{S}}}\, v \in V_1 \\ \text{false} & \text{otherwise} \end{cases}$$

$$PPW2'(G, V_{\mathrm{S}}, j, l).lt = \begin{cases} \text{true} & \text{if there is a proper path decomposition } PD = (V_1, \ldots, V_t) \\ & \text{of } G(j,l) \cup \{\{v_l, v\} \in E(G) \mid v \in V(G) - V(C)\} \\ & \ \wedge\, v_j, v_l \in V_t \wedge \exists_{v \in V_{\mathrm{S}}}\, v \in V_1 \\ \text{false} & \text{otherwise} \end{cases}$$

Because of Lemma 4.3, the recursive description of $PPW2'$ is the same as the recursive description of $PPW2$, but with a different initialization.

$$PPW2'(G, V_{\mathrm{S}}, j, l).ft = \begin{cases} (v_j \in V_{\mathrm{S}}) \vee (v_l \in V_{\mathrm{S}} \wedge \neg s_{c(v_l)}(v_j)) & \text{if } (j-l) \bmod n = 1 \\ c(v_j) \neq c(v_l) \wedge & \\ (\, PPW2'(G, V_{\mathrm{S}}, j-1, l).ft \wedge \neg s_{c(v_l)}(v_j) \vee & \\ \quad PPW2'(G, V_{\mathrm{S}}, j, l+1).lt\,) & \text{if } (j-l) \bmod n > 1 \end{cases}$$

$$PPW2'(G, V_{\mathrm{S}}, j, l).lt = \begin{cases} (v_l \in V_{\mathrm{S}}) \vee (v_j \in V_{\mathrm{S}} \wedge \neg s_{c(v_j)}(v_l)) & \text{if } (j-l) \bmod n = 1 \\ c(v_j) \neq c(v_l) \wedge & \\ (\, PPW2'(G, V_{\mathrm{S}}, j-1, l).ft \vee & \\ \quad PPW2'(G, V_{\mathrm{S}}, j, l+1).lt \wedge \neg s_{c(v_j)}(v_l)\,) & \text{if } (j-l) \bmod n > 1 \end{cases}$$

Using this description, there is a proper path decomposition of G in which one of the vertices of V_{S} occurs in the leftmost node, and one of the vertices of V_{E} occurs in the rightmost node, if and only if there is some j with $v_j \in V_{\mathrm{E}}$, such that $PPW2'(G, V_{\mathrm{S}}, j, j+1).ft$ is true and $\neg s_{c(v_j)}(v_{j+1})$, or $PPW2'(G, V_{\mathrm{S}}, j, j+1).lt$ is true, or $PPW2'(G, V_{\mathrm{S}}, j-1, j).ft$ is true, or $PPW2'(G, V_{\mathrm{S}}, j-1, j).lt$ is true and $\neg s_{c(v_j)}(v_{j-1})$. Note that with these definitions, cases with starting vertices and ending edges can be handled using $PPW2'$, and cases with starting edges and ending vertices can be handled with $PPW2$.

For a given partial two-path G which consists of a chordless cycle C with sticks connected to it, we can compute $PPW2$ and $PPW2'$ in $O(n^2)$, where $n = |V(G)|$, with a similar function as COMP_PPW2 in Section 4.1.

Biconnected Components with Sticks

We now consider partial two-paths which consist of a biconnected component with sticks connected to it.

Lemma 4.5. *Let G be a biconnected three-colored partial two-path which consists of a biconnected component with sticks. Suppose $\bar{G}$ is properly colored and let B denote the biconnected component of $\bar{G}$, and let $(\mathcal{C}, \mathcal{S})$ be a correct path of chordless cycles for B, with $\mathcal{C} = (C_1,..., C_p)$ and $\mathcal{S} = (e_1,..., e_{p-1})$. Let $G' = \bar{G}[V - W]$, where W is the set of vertices $w \in V(G)$ for which $w \in V(C_i) - e_i$ for some i, $1 < i < p$, and $e_{i-1} = e_i$. There is a proper path decomposition of $\bar{G}$ if and only if there is a proper path decomposition of $\bar{G}[V - W]$.*

Proof. If PD is a proper path decomposition of $\bar{G}$, then $PD[V - W]$ is a proper path decomposition of G'.

Suppose PD is a proper path decomposition of G'. For each $e_i = \{v, v'\}$ in $\bar{G}$, $1 < i < p$, with $e_{i-1} = e_i$, e_i is a middle edge of G', and hence we can add a node containing $\{v, v', w\}$ for the vertex $w \in W \cap V(C_i)$, since $c(w) \neq c(v)$ and $c(w) \neq c(v')$. □

So we may further assume that there is no e_i, $1 < i < p$, such that $e_{i-1} = e_i$. Let j be such that $1 \leq j < p$. The sticks that are adjacent to vertices $v \in V(C_j)$ with $v \notin e_{j-1} \cup e_j$ clearly must occur within the occurrence of C_j in any proper path decomposition of G. Let $e_j = \{v, v'\}$. We now consider the sticks adjacent to v and v'.

Lemma 4.6. *Let G be a biconnected three-colored partial two-path which consists of a biconnected component with sticks. Suppose $\bar{G}$ is properly colored and let B denote the biconnected component of $\bar{G}$, and let $(\mathcal{C}, \mathcal{S})$ be a correct path of chordless cycles for B, with $\mathcal{C} = (C_1,..., C_p)$ and $\mathcal{S} = (e_1,..., e_{p-1})$. Suppose there is no i, $1 \leq i < p - 1$, such that $e_i = e_{i+1}$. Let $W_1, W_2 \subseteq V(G) - V(B)$ be defined as follows.*

$$\begin{aligned} W_1 &= \{ w \in V(G) - V(B) \mid \exists_{i, e_i = \{v,v'\}} \{v, w\} \in E(G) \wedge c(w) \neq c(v') \} \\ W_2 &= \{ w \in V(G) - V(B) \mid \exists_{i, e_i = \{v,v'\}} v \in e_{i+1} \wedge \{v, w\} \in E(G) \} \end{aligned}$$

There is a proper path decomposition of $\bar{G}$ if and only if there is a proper path decomposition of $G' = \bar{G}[V - W_1 - W_2]$.

Proof. If PD is a proper path decomposition of $\bar{G}$, then $PD[V - W_1 - W_2]$ is a proper path decomposition of G'.

Let PD be a proper path decomposition of G'. Let $1 \leq j \leq p - 1$ and $e_j = \{v, v'\}$ such that $\{v, w\} \in E(G)$ and $c(w) \neq c(v')$. Since e_j is a middle edge, in each proper path decomposition of G, we can add a node $\{v, v', w\}$ for each stick w of v or v' if $c(w) \neq c(v)$ and $c(w) \neq c(v')$. Let PD' be the path decomposition obtained from PD by doing this for all vertices of W_1. PD' is a proper path decomposition of width two of $\bar{G}[V - W_2]$.

Let j, $1 < j \leq p - 1$, such that $e_{j-1} \cap e_j = \{v\}$. Suppose C_j occurs in $(V_l,..., V_{l'})$ in PD'. e_{j-1} and e_j are end edges of C_j, which means that for all i, $l \leq i \leq l'$, $v \in V_i$. There are at least two vertices $u, u' \in V(C_j) - \{v\}$ for which $\{u, u'\} \in E(C_j)$, and hence $c(u) \neq c(u')$. This means that there is a node in PD' which contains v and u, and there is a node which contains v and u'. Hence, according to Lemma 4.2, for all sticks w of v of color $c(u)$, we can add a node $\{v, u', w\}$ in PD', and for all sticks w of v of color $c(u')$, we can add a node $\{v, u, w\}$ in PD'. The path decomposition which is obtained by doing this for all sticks in W_2 is a proper path decomposition of $\bar{G}$. □

So we may further assume that for all $e_i = \{v, v'\}$, v and v' have no sticks with color i, where $i \in \{1, 2, 3\}$ such that $i \neq c(v)$ and $i \neq c(v')$. And furthermore we may assume that for all j, $1 < j \leq p - 1$, if $e_{j-1} \cap e_j = \{v\}$, then v has no sticks.

Let $e_j = \{v, v'\}$, suppose $v \notin e_{j-1}$ and $v \notin e_{j+1}$. Suppose there is a stick w of v which has color $c(v')$, then in each proper path decomposition of G, edge $\{v, w\}$ must occur in a node which contains v, either in the occurrence of C_j or in the occurrence of C_{j+1}.

Lemma 4.7. *Let G be a biconnected three-colored partial two-path which consists of a biconnected component with sticks. Suppose $\bar{G}$ is properly colored and let B denote the biconnected component of $\bar{G}$, and let $(\mathcal{C}, \mathcal{S})$ be a correct path of chordless cycles for B, with $\mathcal{C} = (C_1, \ldots, C_p)$ and $\mathcal{S} = (e_1, \ldots, e_{p-1})$. Let $e_j = \{v, v'\}$, let $v \in V(G)$, and suppose $e_{j-1} \cap e_j = e_j \cap e_{j+1} = \emptyset$. Suppose there is a proper path decomposition PD of G. If v has a stick of color $c(v')$ which occurs within the occurrence of C_{j-1}, then all sticks of v' of color $c(v)$ occur within the occurrence of C_j.*

Proof. Let $PD = (V_1, \ldots, V_t)$, suppose C_j occurs in $(V_l, \ldots, V_{l'})$, such that $v, v' \in V_{l'}$. Suppose v has a stick w of color $c(v')$ which occurs within the occurrence of C_j. Let $v'' \in V(C_j)$ and i, $l \leq i \leq l'$, be such that V_i is the rightmost node containing v and w and $V_i = \{v, w, v''\}$. Then $v'' \neq v'$ and all nodes of $(V_{i+1}, \ldots, V_{l'})$ contain v, hence if v' has a stick w' of color $c(v)$, then edge $\{v', w'\}$ can not occur within the occurrence of C_j, and hence $\{v', w'\}$ must occur within the occurrence of C_{j+1}. □

Using these lemmas, we can derive an algorithm for computing whether there is a proper path decomposition of a graph G that is a partial two-path consisting of a biconnected component with sticks. Let $(\mathcal{C}, \mathcal{S})$ be a path of chordless cycles for the biconnected component of $\bar{G}$, with $\mathcal{C} = (C_1, \ldots, C_p)$ and $\mathcal{S} = (e_1, \ldots, e_{p-1})$. For each i, $1 \leq i \leq p$, let $n_i = |V(C_i)|$, and let $V(C_i) = (v_0^i, \ldots, v_{n_i-1}^i)$ such that $E(C_i) = \{v_j^i, v_{j+1}^i\} \mid 0 \leq j < n_i\}$ and for each i, $1 \leq i < p$, $e_i = \{v_1^i, v_2^i\}$. For an example, see Figure 23. Furthermore, for each i, $1 < i \leq p$, let G_i^1 denote the induced subgraph of $\bar{G}$ consisting of C_i and all sticks adjacent to vertices of $V(C_i) - \{v_1^{i-1}\}$. Similarly, let G_i^2 denote the induced subgraph of G consisting of C_i and all sticks adjacent to vertices of $V(C_i) - \{v_2^{i-1}\}$. For an example, see Figure 24. Furthermore, let G_1 denote the induced subgraph of $\bar{G}$ consisting of C_1 and all sticks connected to vertices of $V(C_1)$. The algorithm is as follows.

Find the cell-completion $\bar{G}$ of G and check if $\bar{G}$ is properly colored.
Let B be the biconnected component of $\bar{G}$.
Check if B can be written as a correct path of chordless cycles.
If so, let $(\mathcal{C}, \mathcal{S})$ denote this path.
Delete all chordless cycles C_i, $1 < i < p$, for which $e_{i-1} = e_i$ from $(\mathcal{C}, \mathcal{S})$.

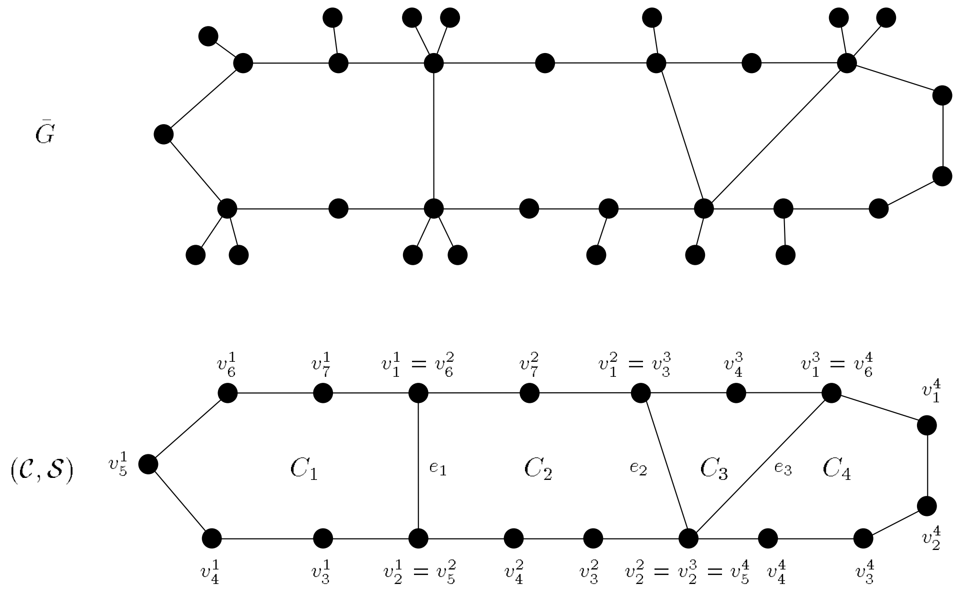

Figure 23: Example of the cell completion $\bar{G}$ of a graph G which consists of a chordless cycle with sticks, and a path of chordless cycles $(\mathcal{C}, \mathcal{S})$ for the biconnected component of $\bar{G}$.

Delete all sticks w of all vertices v that belong to more than one e_i.
$ft := PPW2'(G_1, V(C_1), 1, 2).ft$;
$lt := PPW2'(G_1, V(C_1), 1, 2).lt$;
if $\neg(ft \vee lt) \rightarrow$ **return** false **fi**;
for $i := 2$ to $p - 1$
$\rightarrow$ $ftold := ft$; $ltold := lt$;
$ft := (ftold \wedge PPW2(G_i^1, \{e_{i-1}\}, 1, 2).ft) \ \vee \ (ltold \wedge PPW2(G_i^2, \{e_{i-1}\}, 1, 2).ft)$;
$lt := (ftold \wedge PPW2(G_i^1, \{e_{i-1}\}, 1, 2).lt) \ \vee \ (ltold \wedge PPW2(G_i^2, \{e_{i-1}\}, 1, 2).lt)$;
if $\neq (ft \vee lt) \rightarrow$ **return** false **fi**;
rof;

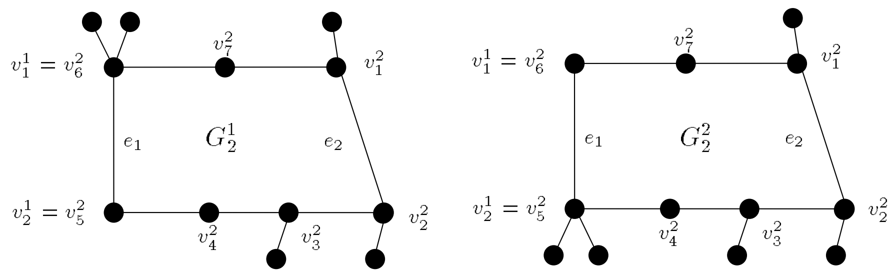

Figure 24: Example of the graphs G_2^1 and G_2^2 for the graph G of Figure 23.

$ftold, ltold := ft, lt;$
$ft := (ftold \wedge \bigcup_{j=0}^{n_p-1} PPW2(G_p^1, \{e_{i-1}\}, j, j+1).ft)$
$\quad \vee \ (ltold \wedge \bigcup_{j=0}^{n_p-1} PPW2(G_p^2, \{e_{i-1}\}, j, j+1).ft);$
$lt := (ftold \wedge \bigcup_{j=0}^{n_p-1} PPW2(G_p^1, \{e_{i-1}\}, j, j+1).lt)$
$\quad \vee \ (ltold \wedge \bigcup_{j=0}^{n_p-1} PPW2(G_p^2, \{e_{i-1}\}, j, j+1).lt);$
return $ft \vee lt$

The algorithm is correct, as follows from the discussion above. Furthermore, it runs in $O(n^2)$ time, where $n = |V(G)|$, because $PPW2$ has to be computed at most twice for each chordless cycle C_i, and $PPW2$ can be computed in $O(n_i^2)$ time for each i. Furthermore, $PPW2'$ has to be computed twice for C_1, and it can be computed in $O(n_1^2)$ time. All other steps take $O(n)$ time.

The algorithm can be modified such that it returns an intervalization of G if there exists one. This can be done in the same way as for biconnected components.

4.3 Trees

In this section, we first show that there is a proper path decomposition of a tree H which is properly colored with three colors if and only if there is a proper path decomposition which has some 'nice' structure. After that, we show how to compute for a given properly colored tree H of pathwidth two whether there is such a nice proper path decomposition of H. First we distinguish different types of partial one-paths connected to a path, corresponding to the way they are connected to the path.

Definition 4.2. (Types of Partial One-Paths). *Let H be a tree of pathwidth two, P a path in H such that $H[V - V(P)]$ has pathwidth one. Let $v \in V(P)$, and H' a component of $H[V - V(P)]$ such that H' has pathwidth one and has a vertex which is adjacent to v, i.e. H' is* connected *to v. Let $w \in V(H')$ be the vertex for which $\{v, w\} \in E(H)$. Let $P' \in \mathcal{P}_1(H')$. We say that H' is of type* I *if w is an end point of P', or if w is adjacent to an end point of P' and $w \notin V(P')$. H' is of type* II *if w is an inner vertex of P'. H' is of type* III *if $w \notin V(P')$ and w is adjacent to an inner vertex of P'.*

Figure 25 gives an example for each type of partial one-path. Note that the type of a partial one-path H' connected to a vertex v of the path P does not depend on the choice of the path $P' \in \mathcal{P}_1(H')$, since if $|\mathcal{P}_1(H')| > 1$, then for each $P' \in \mathcal{P}_1(H')$, $|V(P')| = 1$, so P' does not have any inner vertices, and hence H' has type I.

From now on, by partial one-paths connected to a path P, we only mean the partial one-paths of type I, II and III connected to P, and not the sticks connected to P.

We now give a definition of the kind of path decomposition that we want to use for the algorithm.

Definition 4.3. (Nice Path Decomposition). *Let H be a properly three-colored tree of pathwidth two, $PD = (V_1, ..., V_t)$ a proper path decomposition of H. PD is a* nice path

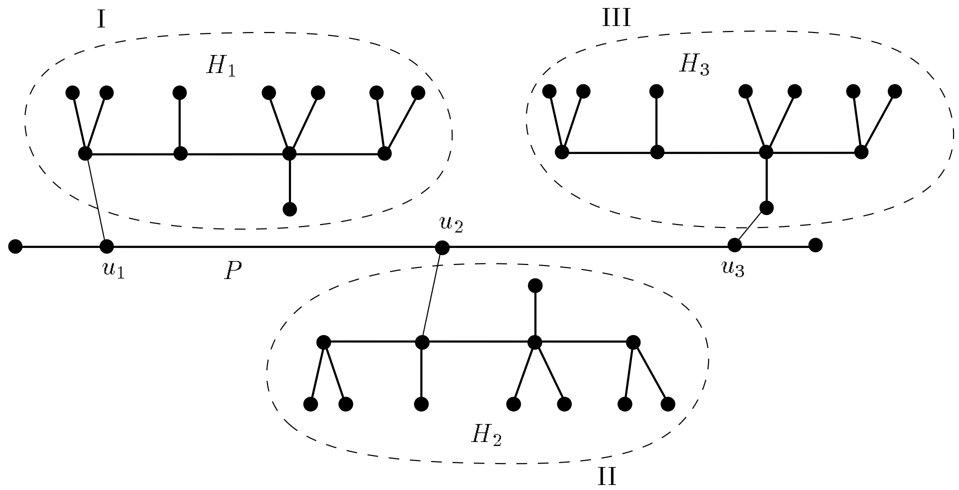

Figure 25: Example of a tree of pathwidth two which contains a path P with $u_1, u_2, u_3 \in V(P)$, and a partial one-path H_1 of type I connected to u_1, a partial one-path H_2 of type II connected to u_2, and a partial one-path H_3 of type III connected to u_3.

decomposition *of* H *of width two if there are no two consecutive nodes which are equal,* V_1 *contains an edge* $\{w, w'\} \in E$ *and* V_t *contains an edge* $\{x, x'\} \in E$, *such that there is a* $P = (v_1, \ldots, v_s) \in \mathcal{P}_2(H)$ *for which there is a partial one-path* H' *that is connected to* v_1 *and a partial one-path* H'' *that is connected to* v_s, $H' \neq H''$, $w, w' \in V(H')$, w *is an end point of some path* $P' \in \mathcal{P}_1(H')$, $x, x' \in V(H'')$, *and* x *is an end point of some path* $P'' \in \mathcal{P}_1(H'')$. *The path from* w *to* x *is called the* nice path *of* PD.

Figure 26 shows an example of a tree H of pathwidth two and a nice path decomposition of width two of H. We will show that for a given properly three-colored tree H of pathwidth two, there is a proper path decomposition of H if and only if there is a nice proper path decomposition of H. First we prove another lemma, which is needed for the case that $|\mathcal{P}_2(H)| > 1$.

Lemma 4.8. *Let* H *be a properly three-colored tree of pathwidth two, such that there is a vertex* $v \in V(H)$ *for which* $H[V - \{v\}]$ *has pathwidth one. Let* $PD = (V_1, \ldots, V_t)$ *be a proper path decomposition of* H, *then there is a proper path decomposition* $PD' = (V'_1, \ldots, V'_q)$ *of* H *such that* V_1 *contains an edge* $e \in E(H)$, V_t *contains an edge* $e' \in E(H)$, $e \neq e'$, *and the shortest path* P *in* H *which contains* e *and* e', *contains a vertex* $v' \in V(H)$ *for which* $H[V - \{v'\}]$ *has pathwidth one and there are two or three components in* $H[V - \{v'\}]$ *which have pathwidth one.*

Proof. We transform PD into a proper path decomposition PD' for which the condition holds. First delete the leftmost node of PD until it contains an edge, and do the same for the rightmost node of PD. Now let $e = \{v, v'\} \in E(H)$ such that $e \subseteq V_1$ and $e' = \{w, w'\} \in E(H)$ such that $e' \subseteq V_t$. Let P be the shortest path containing e and

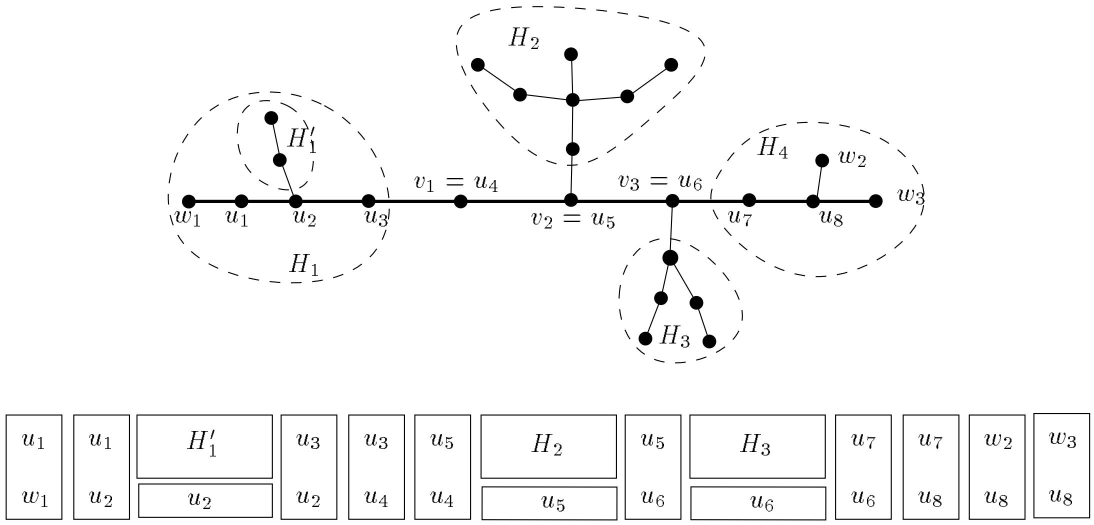

Figure 26: Example of a tree H of pathwidth two with $P_2H = (v_1, v_2, v_3)$, and a nice path decomposition of H of width two with nice path $(u_1, u_2, u_3, u_4, u_5, u_6, u_7, u_8)$. The leftmost node of the path decomposition contains u_1 and stick w_1 of u_1, the rightmost node contains u_8 and stick w_3 of u_8. H_1 is a partial one-path connected to v_1, and u_1 is an end vertex of the path $P_1(H_1)$. H_4 is a partial one-path connected to v_3 and u_8 is an end vertex of the path $P_1(H_4)$.

e', suppose w.l.o.g. that $P = (v, v', \ldots, w', w)$. Note that $e \neq e'$, since if $e = e'$, then each vertex of H is either adjacent to v or to v', and H has pathwidth one. If there is a $u \in V(P)$ such that $H[V - \{u\}]$ has pathwidth one and has two or three components of pathwidth one, then the lemma is proved.

Suppose there is no $u \in V(P)$ such that $H[V - \{u\}]$ has two or more components of pathwidth one. We show that $H[V - V(P)]$ has exactly one component of pathwidth one. If $H[V - V(P)]$ has no components of pathwidth one, then H has pathwidth at most one. If $H[V - V(P)]$ has more than one component of pathwidth one, then there is a vertex $u \in V(P)$ such that $H[V - \{u\}]$ has more than one component of pathwidth one, which gives a contradiction.

Let H' be the component of $H[V - V(P)]$ which has pathwidth one, let $u \in V(P)$ and $u' \in V(H')$ such that $\{u, u'\} \in E(H)$. $H[V - \{u\}]$ has exactly one component of pathwidth one, namely H'. This means that $u = v' = w'$ and that v and w both have degree one. Now transform PD as follows. Delete all neighbors of u which have degree one from all nodes of PD, and for each such neighbor x, add a node $\{u, x\}$ on the left side of the leftmost node of PD. Furthermore, delete the rightmost node from PD until it contains an edge. The resulting path decomposition is proper, and it satisfies the appropriate conditions, since the leftmost node contains an edge $\{u, x\}$, where x has degree one, while the rightmost node can not contain such an edge, and hence contains another edge. Hence the shortest path containing these two edges must contain a vertex u' such that $H[V - \{u'\}]$ has two or more components of pathwidth one. $\square$

Lemma 4.9. *Let H be a properly colored tree of pathwidth two. There is a proper path decomposition of H if and only if there is a nice proper path decomposition of H.*

Proof. The 'if' part is trivially true.

For the 'only if' part, suppose there is a proper path decomposition of H. If $|\mathcal{P}_2(H)| > 1$, let $PD = (V_1,..., V_t)$ be a proper path decomposition of H such that V_1 and V_t contain an edge, and the shortest path containing these edges contains a vertex v_1 for which $H[V - \{v_1\}]$ has pathwidth one, and has two or three components of pathwidth one. Furthermore, let $P = (v_1)$ $(s = 1)$. If $|\mathcal{P}_2(H)| = 1$, let $PD = (V_1,..., V_t)$ be an arbitrary proper path decomposition of H, and let $P = P_2(H) = (v_1,..., v_s)$.

We show how PD can be 'unfolded' until it satisfies the described condition. Suppose PD is not of the required form.

First suppose $s > 1$. Let H_1 be the component of $H[V(H) - \{v_2\}]$ containing v_1, and let H_s be the component of $H[V(H) - \{v_{s-1}\}]$ containing v_s. For each $v \in V_1$ and $v' \in V_t$, the path from v to v' contains P, by Corollary 3.2. This means that $v \in V(H_1)$ and $v' \in V(H_2)$ or vice versa. If the second case holds, transform PD into rev(PD).

Suppose $s = 1$. If $|\mathcal{P}_2(H)| = 1$, then for each $v \in V_1$ and each $v' \in V_t$, the path from v to v' contains P, and hence V_1 and V_t can not contain vertices of the same partial one-path connected to v_1. If $|\mathcal{P}_2(H)| > 1$, then P is chosen in such a way that V_1 and V_t do not contain vertices of the same partial one-path connected to v_1. Let H_1 denote the induced subgraph of H consisting of vertex v_1 and all components of $H[V - \{v_1\}]$ of which V_1 contains a vertex, and let H_2 denote the induced subgraph of H consisting of v_1 and all components of $H[V - \{v_1\}]$ of which V_t contains a vertex. Note that V_1 contains only vertices of H_1, V_t contains only vertices of H_2, and $V(H_1) \cap V(H_2) = \{v_1\}$.

The following cases may occur for V_1.

1. $V_1 = \{v, v'\}$ for some edge $\{v, v'\} \in E(H_1)$ such that v and v' both have at most one neighbor which does not have degree one.

2. V_1 contains no edge.

3. $|V_1| = 3$ and V_1 contains an edge.

4. $V_1 = \{v, v'\}$ for some edge $\{v, v'\} \in E(H_1)$, but v or v' has more than one neighbor which does not have degree one.

For V_t, the possible cases are similar.

If case 1 holds for V_1, then either v or v' has degree one. Suppose v' has degree one. Note that v and v' can not both have degree one, since then H has pathwidth one. $v \neq v_1$, since then v has at least two neighbors which do not have degree one, namely one neighbor in a partial one-path connected to v_1, and v_2 if $s > 1$, or a neighbor in another partial one-path connected to v_1 if $s = 1$. Furthermore, v can not be an inner vertex of $P_1(H')$ for some partial one-path H' which is connected to v_1, since then the two neighbors of v in $P_1(H')$ do not have degree one. Hence v is an end point of some path $P \in \mathcal{P}_1(H')$ for some partial one-path H' that is connected to v_1, which is exactly what we need.

Now, we apply the following transformations on PD such that one of the previous cases holds again after each transformation, until case 1 holds for both V_1 and V_t. First transform PD using the following rules until case 1 applies for V_1, next transform PD using the following rules, adapted for V_t, until case 1 applies for V_t.

If case 2 applies, delete V_1.

If case 3 applies, let $e \in E(H_1)$ such that $e \subseteq V_1$, and add a node containing e only on the left side of V_1.

If case 4 applies, do the following. Suppose w.l.o.g. that the path from v to v_1 contains v'. Consider the components of $H[V - \{v\}]$ which consist of more than one vertex. Note that one of these components is a subgraph of H_1 which does not contain v_1 or v', and hence V_t does not contain any vertex of this component. Let H' be such a component. Now transform PD into $\text{rev}(PD[V(H') \cup \{v\}]) + PD[V - V(H')]$, and let $H_1 = H[V(H') \cup \{v\}]$. The new path decomposition is indeed a proper path decomposition of H, since v is the only vertex that $H[V(H') \cup \{v\}]$ and $H[V - V(H')]$ have in common, and v occurs in the rightmost node of $\text{rev}(PD[V(H') \cup \{v\}])$ and in the leftmost node of $PD[V - V(H')]$. Furthermore, the new H_1 is contains at least one vertex less than the old H_1, the leftmost node of the new PD contains only vertices of the new H_1 and the rightmost node of the new PD contains only vertices of H_2.

Note that the number of transformations is finite, since if the transformation of case 4 is done, then H_1 or H_2 gets smaller, and after each time the transformation of case 4 is done, the transformations of case 2 and 3 can only be done a finite number of times before case 4 holds again. □

The total number of nice paths in a tree H of pathwidth two may be $\Omega(n^2)$, where $n = |V(H)|$. The algorithm we construct has the following structure, in which function Check_Nice_Path(P) returns true if there is a nice proper path decomposition of H with nice path P, and false otherwise.

$b :=$ false;
for certain possible nice paths P of H
$\rightarrow$ $b := b \vee$ Check_Nice_Path(P)
rof
$\{b \Leftrightarrow$ there is a proper path decomposition of H. $\}$

The algorithm will run in $O(n^2)$ time, because the number of nice paths that is tried is bounded by a constant, and function Check_Nice_Path runs in $O(n^2)$ time. In the remainder of this section, we first show which nice paths have to be tried, and which nice paths do not have to be tried. After that, we show how function Check_Nice_Path works. First, we prove some lemmas.

Lemma 4.10. *Let H be a three-colored tree of pathwidth two, $PD = (V_1, ..., V_t)$ a nice proper path decomposition of H, $P = (v_1, ..., v_q)$ the nice path of PD. Let $v_m \in V(P)$ and H' a partial one-path connected to v, let $w \in V(H')$ such that $\{v_m, w\} \in E(H)$.*

1. *If H' is of type* II*, then there is an i, $1 \leq i \leq t$, such that $PD' = (V_1,..., V_i, \{v, w\}, V_{i+1},..., V_t)$ is a nice proper path decomposition of H.*

2. *If H' is of type* III*, then let w' be the inner vertex of $P_1(H')$ that is adjacent to w. Then there is an i, $1 \leq i \leq t$, such that $V_i = \{v_m, w, w'\}$.*

Proof. 1. Suppose H' occurs in $(V_j,..., V_{j'})$. Each node V_i, $j \leq i \leq j'$, contains at most two vertices of H'. There is a node containing v_m and w, since $\{v, w\} \in E(H)$. First we prove the case that H' has type II. If there is a node $V_i = \{v_m, w\}$, then we are done. Suppose there is no such node. Suppose $\{v_m, w\}$ occurs in $(V_l,..., V_{l'})$. Note that edges of one component of $H'[V(H') - \{w\}]$ occur on the left side of V_l and edges of another component of $H'[V(H') - \{w\}]$ occur on the right side of $V_{l'}$. Furthermore, note that $1 < m < q$, since v_1 is an end point of a path $P' \in P_1(H'')$ for some partial one-path H'' which is connected to an end point of a path of $P_2(H)$, and the same holds for v_q. Hence edges of one component of $H[V - \{v\}]$ occur on the left side of V_l and edges of another component of $H[V - V(H') - \{v_m\}]$ occur on the right side of $V_{l'}$. No edges of $H[V - \{v_m, w\}]$ occur within $(V_l,..., V_{l'})$, since each node already contains v_m and w. If $v_m \notin V_{l-1}$, then there is a neighbor u of v_m in one of the four components with edges of $H[V - \{v_m, w\}]$ with $u \in V_l$. If $w \notin V_{l-1}$, then there is a neighbor u or w in one of the components of the four components with edges of $H[V - \{v_m, w\}]$. Let u be the neighbor of v_m or w which occurs in V_l. Similarly, let u' be the neighbor of v_m or w which occurs in $V_{l'}$. Note that $u' \neq u$, since u and u' are in different components of $H[V - \{v_m, w, \}]$. Hence $V_l = \{v_m, w, u\}$ and $V_{l'} = \{v_m, w, u'\}$. This implies that there must be a node V_i, $l \leq i < l'$, such that $V_i \cap V_{i+1} = \{v_m, w\}$. Then $(V_1,..., V_i, \{v_m, w\}, V_{i+1},..., V_t)$ is also a proper path decomposition of H.

2. Now suppose that H' has type III. Because of the structure of path decompositions of width two, there is no node containing w but not w', since w' is an inner vertex of $P_1(H)$, and w is a stick connected to w'. Hence there must be a node containing w, w' and v_m, since $\{w, v_m\} \in E$. □

From this lemma, the following can be concluded immediately. For a three-colored tree H of pathwidth two, and a given nice path P, the partial one-paths H' connected to a vertex $v \in V(P)$ of type III can be handled as a partial one-path of type II by deleting the vertex $w \in V(H')$ which is adjacent to v, and adding edge $\{v, w'\}$, where $w' \in V(H')$ is adjacent to w. If $c(v) = c(w')$, then the resulting graph is colored improperly, and hence there exists no nice proper path decomposition of H with nice path P.

Lemma 4.11. *Let H be a three-colored tree of pathwidth two, $PD = (V_1,..., V_t)$ a nice proper path decomposition of H and $P = (v_1,..., v_q)$ the nice path of PD. Let v be an inner vertex of P and let $H_1,..., H_l$ be the partial one-paths connected to v. There are at most two partial one-paths in $H_1,..., H_l$ which have a vertex of color $c(v)$.*

Proof. Suppose $v_1 \in V_1$ and $v_q \in V_t$, and suppose v occurs in $(V_j,..., V_{j'})$. Note that $1 < j \leq j' < t$. Let H' and H'' be the components of $H[V - \{v\}]$ which contain vertices of P, such that $v_1 \in V(H')$ and $v_q \in V(H'')$. There is an edge of H' which occurs on the left side of $(V_j,..., V_{j'})$, and there is an edge of H'' which occurs on the right side of $(V_j,..., V_{j'})$. Hence it follows directly from Lemma 2.6 that there are at most two partial one-paths connected to v which may have a vertex of color $c(v)$. □

Lemma 4.12. *Let H be a three-colored partial two-path, suppose there is a proper path decomposition of H. There is a proper path decomposition PD of H in which for each $v \in V(H)$ such that $H[V - \{v\}]$ has at least four components which contain an edge, PD contains a node $\{v\}$.*

Proof. Let $PD = (V_1,..., V_t)$ be a proper path decomposition of H. For each $v \in V$ for which $H[V - \{v\}]$ contains four or more components which contain an edge, transform PD as follows. Suppose v occurs in $(V_j,..., V_{j'})$. Let H_1 be the induced connected subgraph of H containing v and all components of $H[V - \{v\}]$ of which there is an edge occurring on the left side of V_j, and let H_2 be the induced subgraph containing v and all components of $H[V - \{v\}]$ of which there is an edge occurring on the right side of $V_{j'}$. Note that $V(H_1) \cap V(H_2) = \{v\}$, since no component of $H[V - \{v\}]$ can have edges occurring on the left side of V_j and edges occurring on the right side of $V_{j'}$. Furthermore, let H_3 be the induced subgraph of H containing v and all components of $H[V - \{v\}]$ which are not in H_1 or H_2. Then $H = H_1 \cup H_2 \cup H_3$. If there are vertices of H_1 which occur on the right side of $V_{j'}$, then they can be deleted, since there are no edges containing these vertices occurring on the right side of $V_{j'}$. Similarly for H_2 on the left side of V_j, and for H_3 on the right side of $V_{j'}$ and on the left side of V_j. Let PD' be the path decomposition PD after deleting these vertices. Then $PD'' = PD[V(H_1)] + (\{v\}) + PD[V(H_3)] + (\{v\}) + PD[V(H_2)]$ is a proper path decomposition of H, since the rightmost node of $PD[V(H_1)]$ contains v, the leftmost node of $PD[V(H_2)]$ contains v, and all nodes of $PD[V(H_3)]$ contain v. □

The following lemmas are important to bound the number of nice paths that has to be tried during the algorithm.

Lemma 4.13. *Let H be a three-colored tree of pathwidth two. Suppose there is no vertex $v \in V(H)$ for which $H[V - \{v\}]$ has pathwidth one. Let $P_2(H) = (v_1,..., v_s)$, and suppose there is a proper path decomposition of H. Let PD be a nice proper path decomposition of H with nice path $P = (u_1,..., u_q)$. The following holds.*

1. *If $H[V - \{v_1\}]$ has three or less components, then there is a partial one-path H' which is connected to v_1, and u_1 is an end point of some $P'' \in \mathcal{P}_1(H')$.*
2. *If $H[V - \{v_1\}]$ has four or more components, and there is a partial one-path connected to v_1 which has a vertex of color $c(v)$, then there is a partial one-path H' which is connected to v_1 and which contains a vertex of color $c(v_1)$, such that there is a nice proper path decomposition PD' of H with nice path $P' = (w_1,..., w_r)$, such that $w_r = u_q$ and w_1 is end point of some $P'' \in \mathcal{P}_1(H')$.*

3. *If $H[V - \{v_1\}]$ has four or more components, no partial one-path H' connected to v_1 has a vertex of color $c(v)$, then for all partial one-paths H' connected to v_1, there is a nice proper path decomposition of H with nice path $(w_1,..., w_r)$, such that $w_r = u_q$ and w_1 is end point of some path in $\mathcal{P}_1(H')$.*

The analog for v_s also holds.

Proof. Let $PD = (V_1,..., V_t)$.
1. If $H[V - \{v_1\}]$ has three of less components, then clearly case 1 holds.

2. If $H[V - \{v_1\}]$ has four or more components, and at least one of these components has a vertex of color $c(v_1)$, then PD is transformed as follows. Let H' be the partial one-path connected to v_1 for which $u_1 \in V(H')$. If H' contains a vertex of color $c(v_1)$, then no transformation is performed. Otherwise, first the transformation of the proof in Lemma 4.12 with $v = v_1$ is done. Note that the resulting $PD = (V_1,..., V_t)$ is still a nice path decomposition with nice path P. Suppose v_1 occurs in $(V_j,..., V_{j'})$, let V_l, $j \leq l \leq j'$, be a node of PD for which $V_l = \{v_1\}$. For each partial one-path H'' connected to v_1 that has an edge occurring on the left side of V_j and that has no vertex of color $c(v)$, do the following. Make a proper path decomposition of width one of H'' and add v_1 to each node. The result is a proper path decomposition PD' of $H[V(H'') \cup \{v_1\}]$. Delete all vertices of H'' from all nodes of PD, and add PD' between V_l and V_{l+1} in PD. Let PD denote the obtained path decomposition of H, and suppose again that v_1 occurs in $(V_j,..., V_{j'})$. If there is no partial one-path connected to v_1 of which an edge occurs on the left side of V_j, let H'' denote a partial one-path connected to v_1 which does contain a vertex of color $c(v)$. H'' occurs within $(V_j,..., V_t)$. Note that $v_1 \in V_1$. Let $PD' = \text{rev}(PD[V(H'') \cup \{v_1\}]) +\!\!\!+ PD[V - V(H'')]$. Now use unfolding as in the proof of Lemma 4.9 to make sure that PD is a nice proper path decomposition and that the end point of the nice path is an end point of a path $P'' \in \mathcal{P}_1(H'')$. Case 2 now holds.

3. If $H[V - \{v_1\}]$ has four or more components, but no partial one-path connected to v_1 has a vertex of color $c(v)$, then PD can be transformed as follows. First apply the transformations as in the proof of Lemma 4.12 with $v = v_1$. Let V_l denote a node of PD for which $V_l = \{v_1\}$. Next, for each partial one-path H' that is connected to v_1, delete all vertices of H' from PD, make a proper path decomposition of width one of H', add v_1 to each node of this path decomposition, and put the obtained proper path decomposition of $H[V(H') \cup \{v_1\}]$ between V_l and V_{l+1}. Delete all empty nodes from PD. Note that V_1 contains v_1 now. For each partial one-path H' connected to v_1 and for each end point w of a path $P' \in \mathcal{P}_1(H')$, we can now make a nice proper path decomposition of H with nice path $P = (u_1,..., u_q)$, such that $u_1 = w$ as follows. Make a proper path decomposition $PD' = (W_1,..., W_r)$ of width one of H', such that $w \in W_1$. Let $w' \in V(H')$ such that $\{v_1, w'\} \in E(H)$. Let m, $1 \leq m \leq r$, be such that W_m is the rightmost node which contains w'. If $m = 1$, then let PD' be revPD', and let $m = r$. Add v_1 to each W_i, $i \geq m$. Let PD' denote this path decomposition. Then $PD' +\!\!\!+ PD[V - V(H')]$ is a nice proper path decomposition that satisfies the condition. □

Lemma 4.14. *Let H be a three-colored tree of pathwidth two such that there is a $v \in V(H)$ for which $H[V - \{v\}]$ has pathwidth one, and has at least two components which have pathwidth one. Let $P = (v_1) \in \mathcal{P}_2(H)$. Suppose there is a nice proper path decomposition PD of H with nice path $P = (u_1,..., u_q)$ such that P contains v_1. Then the following holds.*

1. *If $H[V - \{v_1\}]$ has three or less components, then there are two partial one-paths H' and H'', $H' \neq H''$, connected to v_1, such that u_1 is an end point of some path in $\mathcal{P}_1(H')$, and u_q is an end point of some path in $\mathcal{P}_1(H'')$.*

2. *If $H[V - \{v_1\}]$ has four or more components and there are two partial one-paths connected to v_1 which have a vertex of color $c(v)$, then there are two partial one-paths H' and H'', $H' \neq H''$, connected to v_1, such that H' and H'' both contain a vertex of color $c(v_1)$, and there is a nice proper path decomposition of H with nice path $(w_1,..., w_r)$ such that w_1 is an end point of some path in $\mathcal{P}_1(H')$, and w_r is an end point of some path in $\mathcal{P}_1(H'')$.*

3. *If $H[V - \{v_1\}]$ has four or more components and exactly one partial one-path H' connected to v_1 has a vertex of color $c(v)$, then for each partial one-path H'' connected to v_1, $H' \neq H''$, there is a nice proper path decomposition of H with nice path $(w_1,..., w_r)$ such that w_1 is an end point of some path in $\mathcal{P}_1(H')$, and w_r is an end point of some path in $\mathcal{P}_1(H'')$.*

4. *If $H[V - \{v_1\}]$ has four or more components and no partial one-path connected to v_1 has a vertex of color $c(v)$, then for each two partial one-paths H' and H'' connected to v_1, $H' \neq H''$, there is a nice proper path decomposition PD' of H with nice path $(w_1,..., w_r)$ such that w_1 is an end point of some path in $\mathcal{P}_1(H')$, and w_r is an end point of some path in $\mathcal{P}_1(H'')$.*

Proof. Similar to the proof of Lemma 4.13. □

Let H be a three-colored tree of pathwidth two. It now follows that the number of nice paths that have to be tried to find out whether there is a nice proper path decomposition of H is bounded by a constant. If there is no vertex $v \in V(H)$ such that $H[V - \{v\}]$ has pathwidth one, in case 1 of Lemma 4.13, we have at most 6 possible left end points for a nice path. In case 2, there are at most two partial one-paths connected to v_1 which have a vertex of color $c(v_1)$, because of Lemma 4.12, which also gives at most 6 possible end points for a nice path. In case 3 there is only one possibility. Hence there are at most $6 \cdot 6 = 36$ possible nice paths that have to be checked in the algorithm. If there is a $v \in V(H)$ such that $H[V - \{v\}]$ has pathwidth one, then $|\mathcal{P}_2(H)| \leq 7$, and for each $P' \in \mathcal{P}_2(H)$, there are at most 8 possible left end points for the nice path, and at most 6 for the right end point. This gives a total number of at most $7 \cdot 8 \cdot 6/2 = 168$ possible nice paths that have to be checked in the algorithm. A more precise analysis will give a smaller constant.

Now that we have shown that the number of possible nice paths to try is constant, we construct function Check_Nice_Path(P), which checks for a given nice path whether there is a nice proper path decomposition of H with this nice path.

For each vertex v of the nice path, for each partial one-path H' connected to v, Check_Nice_Path computes certain local information, which denotes whether there is a locally correct nice proper path decomposition of H'. This local information is combined with previously computed global information, which, at the end of the algorithm, denotes whether there is nice proper path decomposition of H with nice path P. Hence, the function Check_Nice_Path(P) has the following structure.

function Check_Nice_Path(P: Path): boolean;
{pre: $P = (v_1, ..., v_q)$ is a nice path of H }
{output: true if there is a proper path decomposition of H
  with nice path P, false otherwise
}

  for $m := 1$ to q
  $\rightarrow$ **for** each partial one-path H' connected to v_m
      $\rightarrow$ compute certain values for H' (the local information)
      rof;
      Combine the computed values for v_m and its partial one-paths (local info)
          with previously processed part (global info).
  rof;

  if combination succeeded
  $\rightarrow$ **return** true
  $\square$ else
  $\rightarrow$ **return** false
  fi
end

In the remainder of this section, we first show what local information must be computed and how this is done. After that we show how the local information of each vertex on the nice path can be combined with the global information into the new global information.

We first show that the number of partial one-paths that is connected to one vertex of the nice path for which the algorithm has to compute a local proper path decomposition is bounded.

Corollary 4.1. *Let H be a three-colored tree of pathwidth two, suppose $PD = (V_1, ..., V_t)$ is a nice proper path decomposition of H with nice path P. Then there is a nice proper path decomposition PD' of H with nice path P in which for each $v \in V(P)$ for which $H[V - \{v\}]$ has at least four components which contain at least two vertices, the following holds. For each partial one-path H' that is connected to v by a vertex $w \in V(H')$, if H' does not contain vertices of color $c(v)$, then H' occurs within the occurrence of v in PD'.*

Proof. For each $v \in V(P)$ for which $H[V - \{v\}]$ has four or more components, we can transform PD into a path decomposition PD' satisfying the stated conditions in the same way as in the proof of Lemma 4.13. □

Corollary 4.1 and Lemma 4.11 show that if a vertex v of the nice path has two or more partial one-paths connected to it, then the algorithm has to do significant computations for at most two partial one-paths connected to v, since there are at most two of these partial one-paths which have a vertex of color $c(v)$.

We now concentrate on the kind of local information that has to be computed.

Lemma 4.15. *Let H be a three-colored tree of pathwidth two, suppose $PD = (V_1,\ldots, V_t)$ is a nice proper path decomposition of H with nice path P. There is a nice proper path decomposition PD' with the same nice path P in which no two partial one-paths of $H[V - V(P)]$ overlap, i.e. for each pair of distinct partial one-paths H' and H'' connected to P, there is no node V_i containing a vertex of H' and a vertex of H''.*

Proof. Suppose there are two partial one-paths H' and H'' connected to $v \in V(P)$ and $v' \in V(P)$, respectively, for which there is a node V_m containing vertices of H' and of H''. Suppose the vertices of H' occur in $(V_j,\ldots, V_{j'})$ and the vertices of H'' occur in $(V_l,\ldots, V_{l'})$. It is not possible that $j \le l \le l' \le j'$, since each V_i, $j \le i \le j'$, contains a vertex of P and a vertex of H', but H'' has pathwidth one. Similarly, it is not possible that $l \le j \le j' \le l'$. Suppose w.l.o.g. that $j \le l \le j' \le l'$. Let i be such that $l \le i \le j'$. V_i does not contain an edge of H' or an edge of H'', since H' and H'' have no vertices in common. This means that $V_{j'},\ldots, V_l$ all contain the same vertex of H', say w, the same vertex of H'', say w', and the same vertex of P, say v. Hence $j' = l$. But w and w' are not adjacent, hence V_l can be split into V_l' and V_l'', with $V_l' = \{v, w\}$, and $V_l'' = \{v, w'\}$. Then $PD' = (V_1,\ldots, V_{l-1}, V_l', V_l'', V_{l+1},\ldots, V_t)$ is also a nice path decomposition of width two of H with nice path P. In this way, all overlaps can be removed from PD, which results in a nice path decomposition with nice path P, without overlapping partial one-paths. □

From now on, if we have a nice proper path decomposition of H with nice path P, we assume that the partial one-paths connected to P do not overlap.

Lemma 4.16. *Let H be a three-colored tree of pathwidth two, suppose PD is a nice proper path decomposition of H with nice path $P = (v_1,\ldots, v_q)$, let $v_m \in V(P)$, let H' be a partial one-path connected to v_m, and suppose H' occurs in $(V_j,\ldots, V_{j'})$. Let $v_l \in V(P)$ be the leftmost vertex on P which occurs in $(V_j,\ldots, V_{j'})$, and $v_{l'} \in V(P)$ the rightmost. Then $v_l \in V_j$, $v_{l'} \in V_{j'}$, and for all i, $l < i < l'$, v_i and sticks adjacent to v_i occur only within $(V_j,\ldots, V_{j'})$, and there is no partial one-path connected to v_i, except H' possibly.*

Proof. Node V_j contains a vertex on the path from v_1 to v_l. But V_j does not contain any vertex v_i with $1 \le i < l$. Hence $v_l \in V_j$, and $v_{l'} \in V_{j'}$. Furthermore, V_j and $V_{j'}$ both contain an edge of H'. This means that V_j and $V_{j'}$ can not contain another vertex of $V(H) - V(H')$. Hence for each i, $l < i < l'$, it is not possible that v_i or any vertex of a stick or a partial one-path connected to v_i is an element of V_p for some p, $1 \leq p < j \vee j' < p \leq t$. So all vertices and edges on the path from v_l to $v_{l'}$ occur within $(V_j,\ldots,V_{j'})$. Suppose there is a partial one-path $H'' \neq H'$ which is connected to v_i for some i, $l < i < l'$. Then H'' must occur within $(V_j,\ldots,V_{j'})$. But each node in $(V_j,\ldots,V_{j'})$ contains a vertex of P and a vertex of H'. This gives a contradiction. □

Definition 4.4. *Let H be a three-colored tree of pathwidth two, PD a nice proper path decomposition of H with nice path $P = (v_1,\ldots,v_q)$, $v_m \in V(P)$, H' partial one-path connected to v_m, H' occurs in $(V_j,\ldots,V_{j'})$. Let v_l be the leftmost vertex on P which occurs in $(V_j,\ldots,V_{j'})$, and $v_{l'}$ the rightmost. We say that H'* uses *the interval $[l,l']$.*

Figure 27 shows an example of a partial one-path H' that uses $[l,l]$.

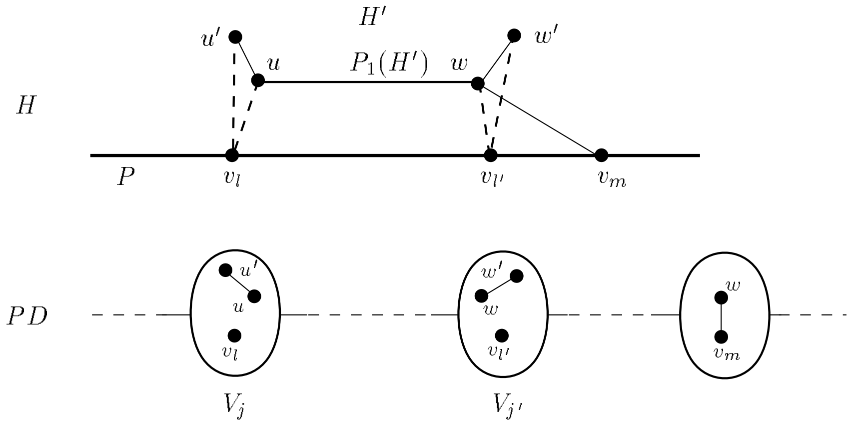

Figure 27: Example of a partial one-path H' that is connected to a vertex v_m of the path P in a tree H of pathwidth two. $P_1(H')$ is the path from u to w. In the occurrence $(V_j,\ldots,V_{j'})$ of H' in the path decomposition PD of width two, v_l, u and a stick u' of u occur in V_j, and $v_{l'}$, w and a stick w' of w occur in V_j. Hence H' uses $[l,l']$, which is shown by the dashed lines in the graph (note that the dashed lines are edges of the interval completion of PD). All vertices v_i, $l < i < l'$, and sticks adjacent to v_i occur only within $(V_j,\ldots,V_{j'})$.

In the following corollaries, we summarize some earlier lemmas in terms of intervals.

Corollary 4.2. *Let H be a three-colored tree of pathwidth two, $PD = (V_1,\ldots,V_t)$ a nice proper path decomposition of H with nice path $P = (v_1,\ldots,v_q)$. Let $v_m \in V(P)$, H' a partial one-path which is connected to v_m. Let H'' be another partial one-path which is connected to P. Suppose H' uses $[j,j']$ and H'' uses $[l,l']$. See e.g. Figure 28. The following holds.*

1. *Either $j \geq l'$ or $l \geq j'$.*
2. *Either $l' \leq m$ or $l \geq m$.*

Proof. 1. Follows from the fact that there is no node in PD which contains a vertex of H' and a vertex of H'' and Lemma 4.16.

2. Follows from Lemma 4.16. □

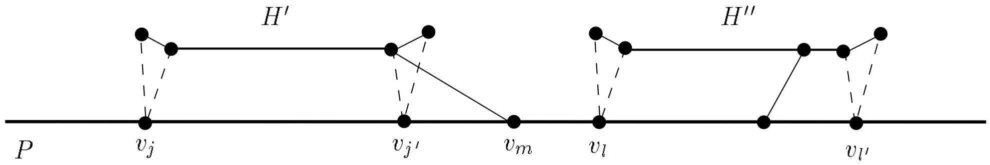

Figure 28: Example of a partial one-path H' that is connected to a vertex v_m of the path P, and another partial one-path H'' that is connected to P. H' uses $[j, j']$, H'' uses $[l, l']$.

Corollary 4.3. *Let H be a three-colored tree of pathwidth two, $PD = (V_1,..., V_t)$ a nice proper path decomposition of H with nice path $P = (v_1,..., v_q)$. Let $v_m \in V(P)$, $H_1,..., H_{nr}$ the partial one-paths connected to v_m. For each i, $1 \le i \le nr$, suppose H_i uses $[j_i, j_i']$.* (See e.g. Figure 29).

1. *There is at most one i, $1 \le i \le nr$, for which $j_i' > m$ and there is at most one i', $1 \le i' \le nr$, for which $j_i < m$, and all others have $j_i = j_i' = m$.*
2. *If there is an i such that $j_i < m$ and $j_i' > m$, then $nr = 1$.*
3. *If $nr \ge 2$, then PD can be transformed into nice proper path decomposition with the same nice path, such that for each H_i, $1 \le i \le nr$, which contains no vertices of color $c(v_m)$, $j_i = j_i' = m$.*

Proof. 1. Follows from Lemma 2.6.

2. Follows from Lemma 4.12.

3. Follows from Corollary 4.1. □

For each partial one-path H' connected to P, the local information denotes certain possible intervals $[j, j']$ for which H' can use $[j, j']$.

In the next lemmas, we further bound the number of possible values for the intervals $[j, j']$ that a partial one-path can use.

Lemma 4.17. *Let H be a three-colored tree of pathwidth two, $PD = (V_1,..., V_t)$ a nice proper path decomposition of H with nice path $P = (v_1,..., v_q)$. Let $v_m, v_{m'} \in V(P)$, $m' > m$, and let H' be a partial one-path connected to v_m, H'' a partial one-path connected to $v_{m'}$. Suppose H' uses $[j, j']$, $m' \le j \le j' \le q$ and H'' uses $[l, l']$, $1 \le l \le l' \le q$. Then the following holds.*

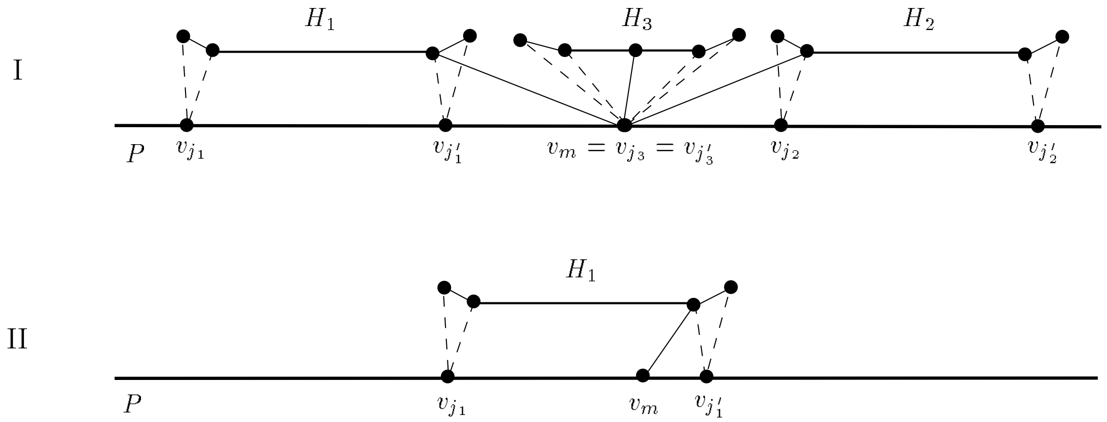

Figure 29: Example of partial one-paths H_1,..., H_{nr} which are connected to a vertex v_m of the path P. For each i, H_i uses $[j_i, j'_i]$. In Part I, $nr = 3$. In Part II, H_1 uses $[j_1, j'_1]$ with $j_1 < m < j'_1$. Hence $nr = 1$.

- *either $l' \leq m$ or $l \geq j'$, and*
- *if $l \geq j'$ then H'' occurs on the right side of H' and $j' = j = m'$.*

Proof. There are three possibilities for $[l, l']$, namely

1. $1 \leq l \leq l' \leq m$,
2. $j' \leq l \leq l' \leq q$, or
3. $m \leq l \leq l' \leq j$ and neither case 1 nor case 2 holds.

We first show that case 3 is not possible. Suppose $m \leq l \leq l' \leq j$ and case 1 and case 2 do not hold. Suppose H' occurs in $(V_r,..., V_{r'})$, H'' occurs in $(V_s,..., V_{s'})$. See also Figure 30. Vertex v_l is the only vertex of $H[V - V(H'')]$ occurring in V_s and $m < l'$, which means that v_m does not occur in $V_{s'}$ or on the right side of $V_{s'}$. Furthermore, $v_{l'}$ is the only vertex of $H[V - V(H'')]$ occurring in $V_{s'}$ and $l < j'$, which means that vertices of H' occur on the right side of $V_{s'}$. But $V_{s'}$ does contain a vertex of H'' or vertex v_m, as can be seen from Figure 30, which gives a contradiction. Hence only cases 1 and 2 are possible.

We now have to prove that if $l \geq j'$, then H'' occurs on the right side of H' and $j' = j = m'$. Suppose H'' occurs on the left side of H'. Then $s \leq s' < r \leq r'$. $m < m' \leq l$, so v_m occurs only on the left side of V_s. But no node of $(V_s,..., V_{s'})$ contains a vertex of H' or v_m, which gives a contradiction. Hence H'' occurs on the right side of H'. Suppose $j' > m'$, see also Figure 31. Then $v_{m'}$ only occurs on the left side of $V_{r'}$. But $V_{r'}$ does not contain a vertex of H'', which gives a contradiction. Hence $j = j' = m'$. □

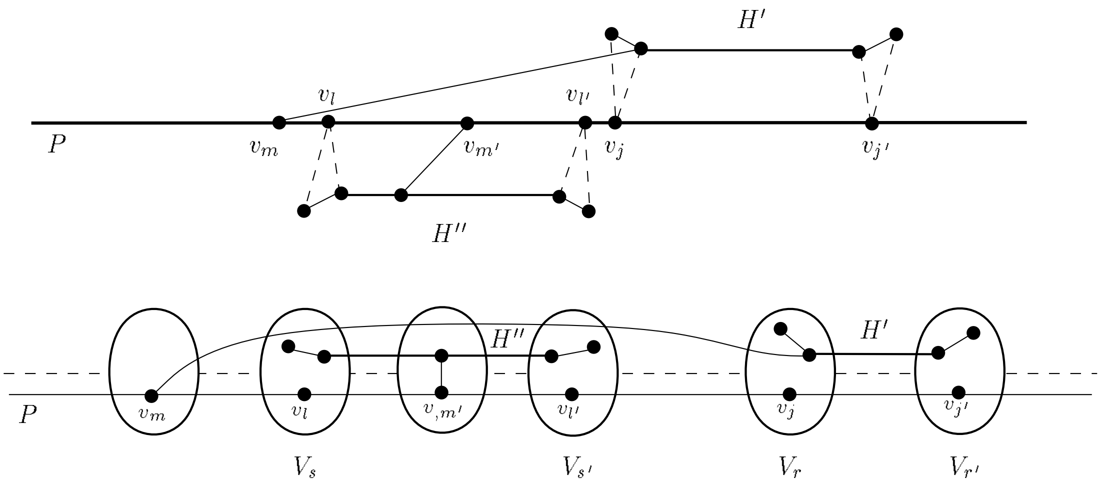

Figure 30: Example of partial one-paths H' and H'' as used in the proof of part 1 of Lemma 4.17.

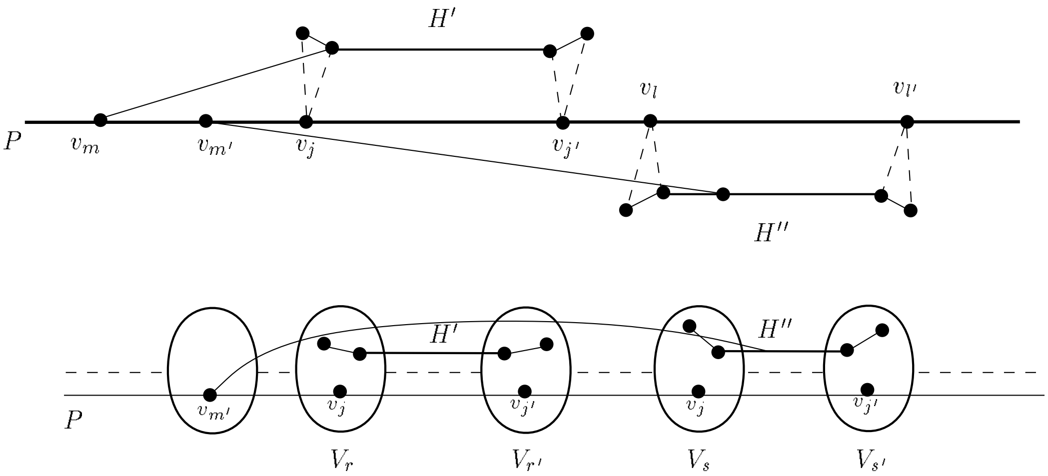

Figure 31: Example of partial one-paths H' and H'' as used in the proof of part 2 of Lemma 4.17.

Lemma 4.18. *Let H be a three-colored tree of pathwidth two, $PD = (V_1, \ldots, V_t)$ a nice proper path decomposition of H with nice path $P = (v_1, \ldots, v_q)$. Let $v_m, v_{m'} \in V(P)$, $m' > m$, and let H' be a partial one-path connected to v_m, H'' a partial one-path connected to $v_{m'}$. Suppose H' uses $[j, j']$, $m' \leq j \leq j' \leq q$ and H'' uses $[l, l']$, $1 \leq l \leq l' \leq m$. Then $m' = m+1$ or $m' = m+2$ and v_{m+1} has degree two; there is a node in PD containing v_m, v_{m+1} and $v_{m'}$, and H' and H'' have type* I.

Proof. Suppose H' occurs in $(V_r, \ldots, V_{r'})$ and H'' occurs in $(V_s, \ldots, V_{s'})$. Then $s' < r$, since $l' < j$. Let $V_{r'} = \{v_{j'}, u, u'\}$, $u, u' \in V(H')$ and $V_s = \{v_l, w, w'\}$, $w, w' \in V(H'')$. Suppose u is an end point of a path $P' \in \mathcal{P}_1(H')$ and w is an end point of a path $P'' \in \mathcal{P}_1(H'')$. See also Figure 32. Vertex v_m does not occur in $(V_r, \ldots, V_{r'})$, hence u and u' are not adjacent to v_m. Similarly, w and w' are not adjacent to $v_{m'}$. Let $G = H \cup \{\{u', v_{j'}\}, \{w', v_l\}\}$. PD is also a path decomposition of G. We first prove that $m' = m+1$ or $m' = m+2$ and v_{m+1} has degree two and that there is a node containing v_m, v_{m+1} and $v_{m'}$.

Suppose $m' > m+1$. Then G contains three disjoint paths between v_m and $v_{m'}$, as can be seen in Figure 32. According to Lemma 3.1, PD is a proper path decomposition of the graph G' which is obtained from G by adding edge $\{v_m, v_{m'}\}$. Graph G' contains three chordless cycles which have edge $\{v_m, v_{m'}\}$ in common. At least one of these chordless cycles, say C, must have three vertices, and the vertex $v \in V(C)$ with $v \neq v_m, v_{m'}$ has degree two, i.e. it is only adjacent to v_m and $v_{m'}$. Cycle C can not be the cycle containing vertices of H' or H'', since the path from v_m to u' in H' contains at least two edges, and the path from $v_{m'}$ to w' in H'' also contains at least two edges. Hence it must be the cycle consisting of $v_m, \ldots, v_{m'}$. So either $m' = m+1$ or $m' = m+2$ and v_{m+1} has degree two. Furthermore, the two or three vertices v_m, v_{m+1} and $v_{m'}$ occur in one node, which also means that they must have different colors.

We now have to prove that H' and H'' both have type I. Let C' be the chordless cycle of G' which contains v_l and let C'' be the chordless cycle of G which contains $v_{j'}$. C' and C'' have edge $\{v_m, v_{m'}\}$ in common. All edges between vertices $v_l, \ldots, v_{j'}$, edges between vertices $v_{l+1}, \ldots, v_{j'-1}$ and their adjacent vertices, and all edges of H' and H'' occur within $(V_s, \ldots, V_{r'})$. Suppose H' has type II or III, then let $v \in V(P_1(H'))$ be such that v is adjacent to v_m if H' has type II, or v has distance two to v_m if H' has type III. Then $v \in V(C')$, and there is a vertex connected to v that does not have degree one. This means that v should occur in the leftmost node containing an edge of C'. This is node $V_{r'}$, but $V_{r'} = \{v_{j'}, u, u'\}$, and $u', u \neq v$. Contradiction. $\square$

Let H be a properly colored tree of pathwidth two, $PD = (V_1, \ldots, V_t)$ a nice proper path decomposition of H with nice path $P = (v_1, \ldots, v_q)$. Let $v_m \in V(P)$, $1 \leq m \leq q$, let $H_1, \ldots, H_{nr}$ be the partial one-paths connected to v_m, $nr \geq 1$, for each i, $1 \leq i \leq nr$, suppose H_i uses $[j_i, j_i']$ such that for all i, $1 \leq i < nr$, $j_i' \leq j_{i+1}$. Using Corollaries 4.2 and 4.3, and Lemma 4.18, we can derive what situations are possible for the intervals $[j_i, j_i']$. Let pp, p, n and nn, $1 \leq pp \leq p \leq m \leq n \leq nn \leq q$, be such that v_p is the rightmost vertex on the left side of v_m which has partial one-paths connected to it, or $p = 1$ if there is no such vertex, v_n is the leftmost vertex on the right side of v_m which has partial one-paths connected to it, or $n = q$ if there is no such vertex, v_{pp} is the

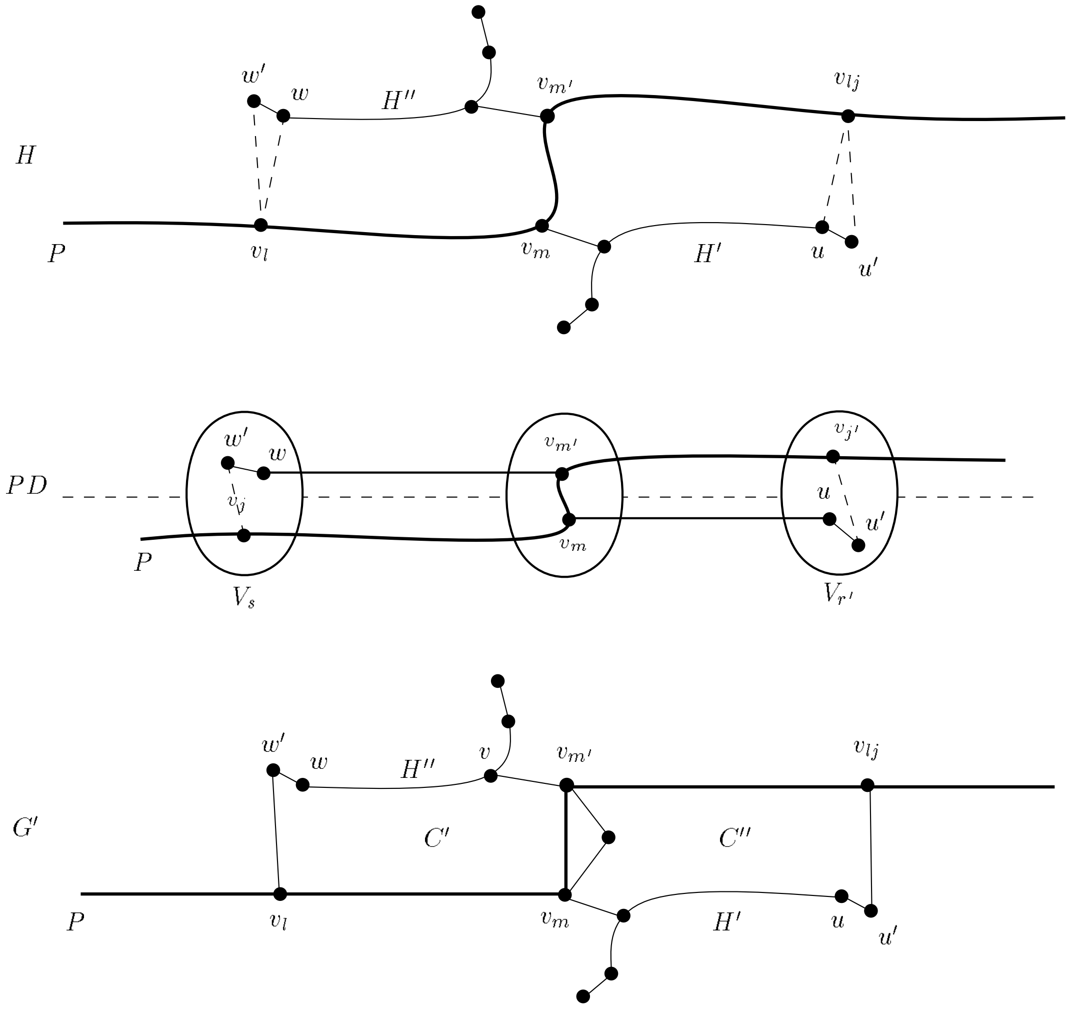

Figure 32: Example of the use of partial one-paths H' and H'' in a tree H of path-width two, a path decomposition PD of H, and the graph G' as given in the proof of Lemma 4.18.

rightmost vertex on the left side of v_p which has partial one-paths connected to it, or $pp = 1$ if there is no such vertex, and v_{nn} is the left most vertex on the right side of v_n having partial one-paths connected to it, or $nn = q$ if there is no such vertex.

Suppose $nr \geq 1$. If $nr = 1$, then the following three cases are possible (see Figure 33).

1. $pp \leq j_1 \leq j_1' \leq p$.

2. $p \leq j_1 \leq j_1' \leq n$.

3. $n \leq j_1 \leq j_1' \leq nn$.

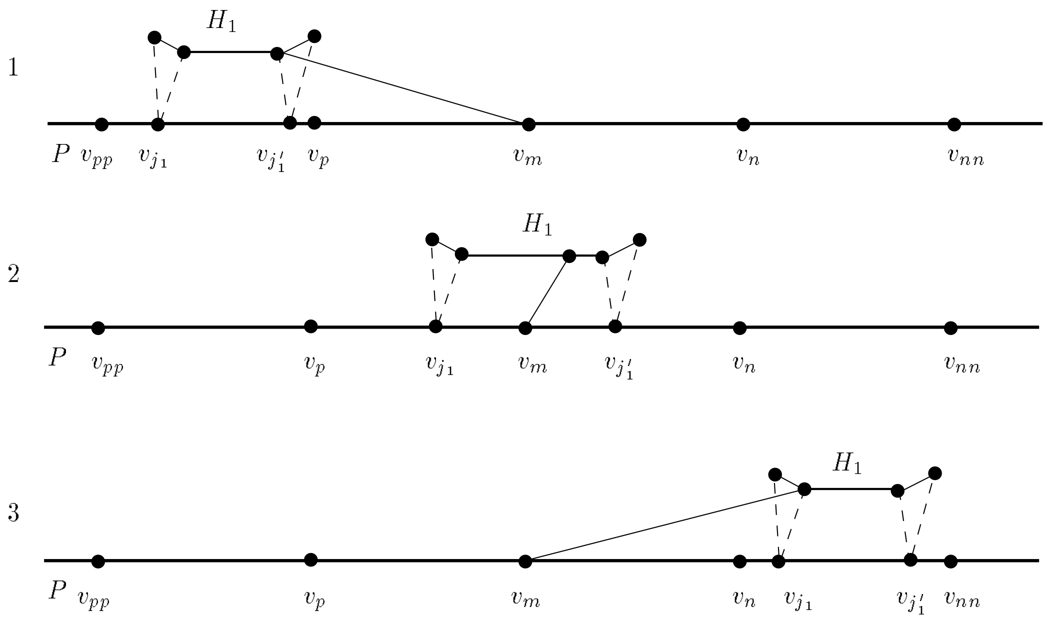

Figure 33: The three possible cases of the use $[j_1, j_1']$ of a partial one-path H_1.

If $nr > 1$, then the following four cases are possible (see Figure 34).

1. $pp \leq j_1 \leq j_1' \leq p$, $m \leq j_{nr} \leq j_{nr}' \leq n$, and for all i, $1 < i < nr$, $j_i = j_i' = m$.

2. $pp \leq j_1 \leq j_1' \leq p$, $n \leq j_{nr} \leq j_{nr}' \leq nn$, and for all i, $1 < i < nr$, $j_i = j_i' = m$.

3. $p \leq j_1 \leq j_1' \leq m$, $m \leq j_{nr} \leq j_{nr}' \leq n$, and for all i, $1 < i < nr$, $j_i = j_i' = m$.

4. $p \leq j_1 \leq j_1' \leq m$, $n \leq j_{nr} \leq j_{nr}' \leq nn$, and for all i, $1 < i < nr$, $j_i = j_i' = m$.

The local information that is computed by function Check_Nice_Path, consists of certain values for each partial one-path connected to the nice path. If $nr = 1$, then for partial one-path H_1, the local information consists certain intervals $[j, j']$ which can be used by H_1 for each of the three cases above.

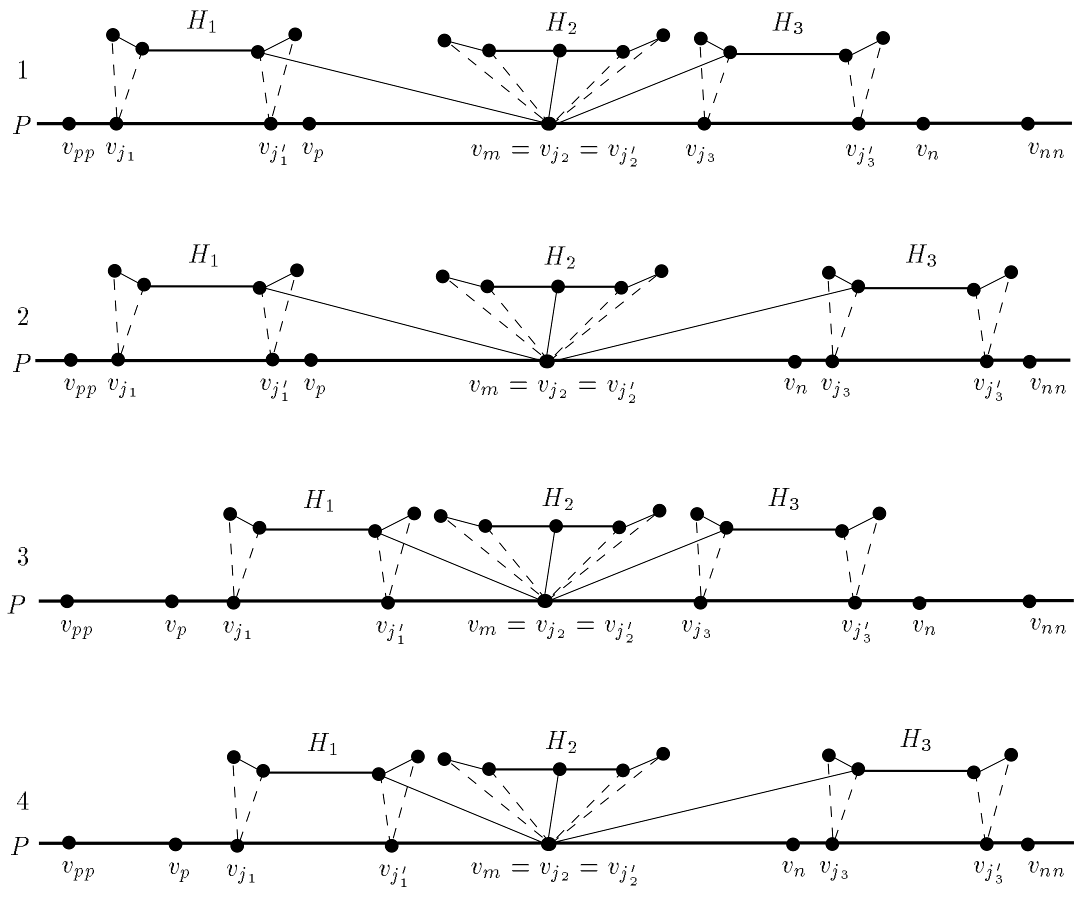

Figure 34: The four possible cases of the use $[j_i, j_i']$ of partial one-paths H_i, $1 \leq i \leq 3$, with $j_1' \leq j_2$ and $j_2' \leq j_3$.

If $nr > 1$, then for all partial one-paths H_i connected to v_m which have no vertex of color $c(v_m)$, the local information consists of the interval $[m, m]$ only. For all other partial one-paths H_i, the local information consists of certain intervals $[j, j']$ which can be used by H_i, for the case that $pp \leq j \leq j' \leq p$, the case that $p \leq j \leq j' \leq m$, the case that $m \leq j \leq j' \leq n$ and the case that $n \leq j \leq j' \leq nn$. These values for different partial one-paths connected to v_m can then be combined such that they satisfy one of the four cases that are given above.

Let H' be a partial one-path that is connected to v_m. We distinguish five possibilities for the interval that H' can use in a nice proper path decomposition of H.

1. $nr \geq 2$ and there are j, j', $p \leq j \leq j' \leq m$, such that H' uses $[j, j']$.
2. $nr = 1$ and there are j, j', $p \leq j \leq j' \leq n$, such that H' uses $[j, j']$.
3. There are j, j', $pp \leq j \leq j' \leq p$, such that H' uses $[j, j']$.
4. $nr \geq 2$ and there are j, j', $m \leq j \leq j' \leq n$, such that H' uses $[j, j']$.
5. There are j, j', $n \leq j \leq j' \leq nn$, such that H' uses $[j, j']$.

We now describe what information is computed for cases 1, 2 and 3, and how it is computed. Cases 4 and 5 are similar to cases 1 and 3. Suppose all partial one-paths of type III are transformed into partial one-paths of type II. In each of the cases 1, 2 and 3, we first analyze how a proper path decomposition looks if this case holds, after which we show what the local information is that has to be computed, and how this information can be computed.

Case 1 $nr \geq 2$ and $p \leq j \leq j' \leq m$

We first analyze the structure of a proper path decomposition in which H' uses $[j, j']$ for some $p \leq j \leq j' \leq m$. We assume that there is no partial one-path H'' which is connected to v_p and which uses $[l, l']$ for some $l \geq m$, since in that case, $j = j' = p$, and hence this case is considered in case 3. Furthermore, we assume that H' contains a vertex of color $c(v_m)$.

Claim 4.1. If H' uses $[j, j']$ for some j, j' with $p \leq j \leq j' \leq m$, then a partial one-path H'' connected to $v_{m'}$, $H' \neq H''$, can use $[l, l']$, with $l' \leq j$ if $m' < m$ and $l \geq m$ if $m' \geq m$.

Proof. Suppose there is a nice proper path decomposition $PD = (V_1, \ldots, V_t)$ of H with nice path P in which H' uses $[j, j']$ for some j and j' with $p \leq j \leq j' \leq m$. Let H'' be a partial one-path connected to P, $H'' \neq H'$, which uses $[l, l']$. Clearly, $l' \leq j$ if $m' < m$, and $l \geq m$ if $m' \geq m$.

We have to show that it is possible that $l' = j$ or $l = m$. Therefore, we show that we can modify PD slightly, such that there is a node $\{v_j\}$ in PD which occurs on the left side of the occurrence of H', and there is a node $\{v_m\}$ in PD which occurs on the right side of the occurrence of H'.

First consider v_m. Vertex v_m separates H in four or more components which contain an edge. Hence PD can be modified such that there is a node V_a with $V_a = \{v_m\}$, PD is still a nice proper path decomposition with nice path P, and H' uses $[j, j']$ (see Lemma 4.12). Suppose H' occurs in $(V_s,..., V_{s'})$ and suppose v_m occurs in $(V_r,..., V_{r'})$. Note that $s' < a$, since H' contains a vertex of color $c(v_m)$, which means that $j < m$ and hence $s < a$.

Next consider v_j. Suppose $V_s = \{v_j, u, u'\}$, for some $u, u' \in V(H')$. For all i, $i < s$, it is not necessary that there is a $v \in V(H')$ such that $v \in V_i$, since all edges containing a vertex of H' occur within $(V_s,..., V_t)$. Furthermore, no V_i, $i < s$, contains a vertex of the path $(v_{j+1},..., v_q)$ or a vertex of a stick or partial one-path that is connected to this path. This means that we can delete all vertices of H' from nodes V_i, $i < s$, and add a node $\{v_j\}$ between V_{s-1} and V_s. □

It follows from the claim that we only need the largest value of j, such that H' can use $[j, j']$ for some j', $p \leq j \leq j' \leq m$.

Definition 4.5. *The local information for H' for the case that H' uses $[j, j']$, $p \leq j \leq j' \leq m$ is j_1, $p \leq j_1 \leq m$, which is the largest value of j for which there is a j', $j \leq j' \leq m$, such that H' can use $[j, j']$.*

We now show how to compute j_1.

Let $P' \in \mathcal{P}_1(H')$, let u and w be the two end points of P'. Let $v \in V(H')$ such that $\{v, v_m\} \in E(H)$. For each j, $p \leq j \leq m$, let G_j^u denote the graph obtained from H as follows (see e.g. Figure 35). Add edge $\{u, v_j\}$. If $v \neq w$ and v is not a stick of w, then also add edge $\{w, v_m\}$. If v is a stick of u and $u \neq w$, then also add edge $\{u, v_m\}$. Furthermore, delete vertices $\{v_1,..., v_{j-1}, v_{m+1},..., v_q\}$ and all sticks and partial one-paths adjacent to these vertices, all sticks adjacent to v_j and v_m, all partial one-paths adjacent to v_j and all partial one-paths except H' that are adjacent to v_m. Define G_j^w in the same way for each j, $p \leq j \leq m$. Note that G_j^u and G_j^w are biconnected components with sticks.

Let l_1 be the largest j, $p \leq j \leq m$, for which either there is a proper path decomposition of G_j^u with vertex v_m in the rightmost node and edge $\{v_j, u\}$ in the leftmost node, undefined if there is no such proper path decomposition.

Let l_2 be the largest j, $p \leq j \leq m$, for which there is a proper path decomposition of G_j^w with vertex v_m in the rightmost node and edge $\{v_j, w\}$ in the leftmost node, undefined if there is no such proper path decomposition.

Claim 4.2. Suppose j_1 is defined, i.e. there is a nice proper path decomposition with nice path P in which H' uses $[j, j']$, $p \leq j \leq j' \leq m$. Then $j_1 = \max\{l_1, l_2\}$.

Proof. We first show that $j_1 \leq \max\{l_1, l_2\}$.

Let $PD = (V_1,..., V_t)$ be a nice proper path decomposition of H with nice path P in which H' uses $[j_1, j']$ for some j' with $j_1 \leq j' \leq m$. Suppose there is a node V_a with $V_a = \{v_m\}$. Note that $j_1 < m$, since H' has a vertex of color $c(v_m)$. Note also that for each $P' \in \mathcal{P}_1(H')$, V_s and $V_{s'}$ each contain an end point of P' and a stick adjacent to

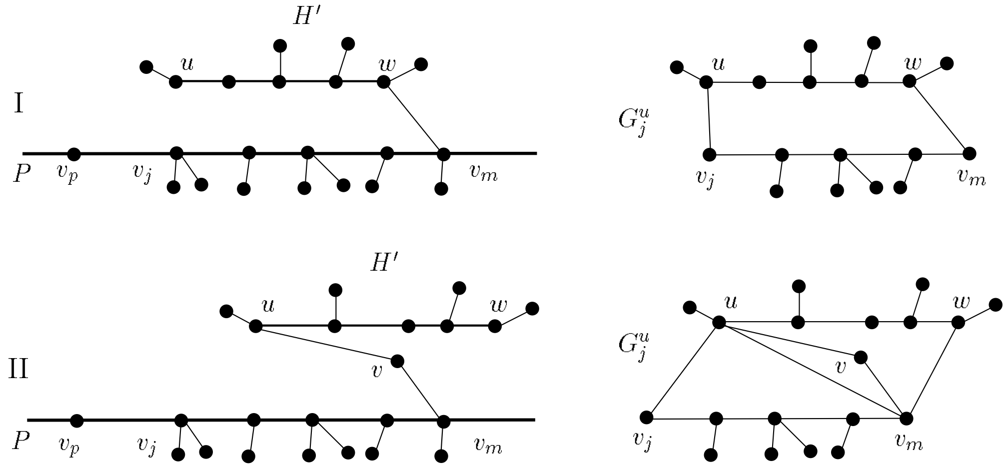

Figure 35: Examples of G^u_j for the case that v_m is adjacent to w (I), and for the case that v_m is not adjacent to w or a stick of w, but v_m is adjacent to a stick of u (II).

this end point. Let $P' \in \mathcal{P}_1(H')$ as defined before, with end points u and w. Suppose w.l.o.g. that $u \in V_s$ and $w \in V_{s'}$. Let $u', w' \in V(H')$ such that $u' \in V_s$, $w' \in V_{s'}$, and u' is a stick adjacent to u, w' is a stick adjacent to w. For an example, see Figure 36. Let $v \in V(H')$ be such that $\{v, v_m\} \in E(H)$. If v is an inner vertex of P' (i.e. H' has type II), or if $v \neq w$ and $v = u$ or v is a stick adjacent to u, then $j' = m$, since v occurs only on the left side of $V_{s'}$, and there is a V_i, $s \leq i < s'$, with $v \in V_i$ and $v_m \in V_i$ (Lemma 3.11). Note also that, if v is a stick of u, then there is a node containing v_m, v and u (also Lemma 3.11).

If $v = w$ or v is a stick adjacent to w, let G be the graph obtained from H by adding edge $\{u, v_{j_1}\}$ only. Otherwise, let G be the graph obtained from H by adding edges $\{u, v_{j_1}\}$ and $\{w, v_{j'}\} = \{w, v_m\}$, and if v is a stick of u, add edge $\{u, v_m\}$, and delete v and its incident edges. Note that PD is a proper path decomposition of G, and that $G^u_{j_1}$ is a subgraph of G (see Figure 36). Suppose G' occurs in $(V_b, \ldots, V_{b'})$. Clearly, $b \leq s$ and $s' \leq b' \leq a$. In fact, $s = b$, since all vertices $v_{j_1+1}, \ldots, v_q$ and sticks adjacent to these vertices occur on the right side of V_s only. Furthermore, $v_m \in V_{b'}$ and $\{v_j, u\} \subseteq V_a$. Hence there is a proper path decomposition of $G^u_{j_1}$ with edge $\{u, v_j\}$ in the leftmost node and vertex v_m in the rightmost node, so $j_1 \leq l_1$.

We now show that $j_1 \geq \max\{l_1, l_2\}$.

Suppose there is a nice proper path decomposition $PD = (V_1, \ldots, V_t)$ of H in which P uses $[j_1, j']$, $p \leq j \leq j' \leq m$. Let $j = \max\{l_1, l_2\}$. We modify PD such that it is a nice proper path decomposition with nice path P, and H' uses $[j, l]$ for some $j \leq l \leq m$.

Let H_1 be the induced subgraph of H consisting of vertices $v_1, \ldots, v_{j_1}$, and all sticks and partial one-paths connected to these vertices. Let H_2 be the induced subgraph of H consisting of vertices $v_m, \ldots, v_q$, and all sticks and partial one-paths connected to these vertices, except H'. Note that the rightmost node of $PD[H_1]$ contains v_{j_1} only, and the leftmost node of $PD[H_2]$ contains v_m. We have shown that $j_1 \leq j$. Let H_3

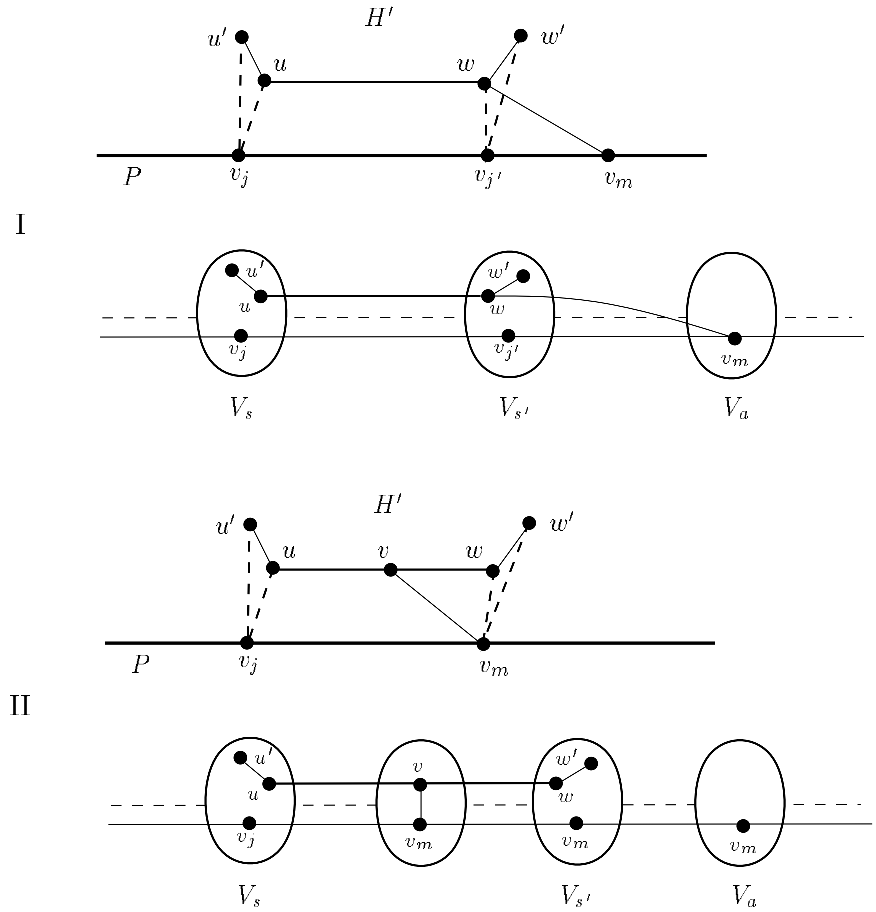

Figure 36: Examples of H' if it uses $[j_1, j']$, $j_1 \leq j' \leq m$, and of the occurrence of H'. In part I, v_m is adjacent to w. In part II, v_m is adjacent to an inner vertex v of P', which means that $j' = m$.

be the induced subgraph of H consisting of vertices $v_{j_1},\ldots, v_j$ and all sticks adjacent to vertices $v_{j_1+1},\ldots, v_j$. Note that H_3 has pathwidth one at most. Let PD' be a proper path decomposition of H_3 with v_{j_1} in the leftmost node and v_j in the rightmost node. Let PD'' be a proper path decomposition of G_j^u or G_j^w with v_m in the rightmost node and $\{u, v_j\}$ in the leftmost node, or $\{w, v_j\}$ in the leftmost node, respectively. Then $PD[H_1] + PD' + PD'' + PD[H_2]$ is a nice proper path decomposition of H with nice path P, such that H' uses $[j, l]$ for some $j \leq l \leq m$, hence $j_1 \geq j$. □

Claim 4.3. j_1 can be computed in $O(n^2)$ time, where n is the number of vertices of G_p^u.

Proof. For all j, $p \leq j \leq m$, G_j^u is a biconnected component with sticks, hence we can compute in $O(n^2)$ time whether there is a proper path decomposition of G_j^u with v_m in the rightmost node and $\{u, v_j\}$ in the leftmost node. This can be done by computing $PPW2'$ with v_m as starting vertex and edge $\{u, v_j\}$ as end edge.

However, if this is done for all j, $p \leq j \leq m$, and both for u and w, then this may result in an $\Omega(n^3)$ algorithm. Fortunately, we can use the structure of the algorithm to compute PPW2' to compute j_1 in such a way, that the algorithm has to be called only twice: once for u and once for w.

Let p', $p \leq p' \leq m$, be such that p' is as small as possible and $c(u) \neq c(v_{p'})$. If $v = w$ or v is a stick of w, then $G_{p'}^u$ contains one chordless cycle C. Number the vertices of C in order as $\{u_0,\ldots, u_{n-1}\}$ in such a way that $u_j = v_j$ for each j, $p' \leq j \leq m$, and hence $u = u_{p'-1}$ (note that for all i, u_i denotes $u_{i \bmod n}$). See for example Figure 37. For each j, $p' \leq j \leq m$, determine $PPW2'(G_{p'}^u, \{v_m\}, p'-1, p').ft$ and $PPW2'(G_{p'}^u, \{v_m\}, p'-1, p').lt$. During this computation, $PPW2'(G_{p'}^u, \{v_m\}, p'-1, j).ft$ and $PPW2'(G_{p'}^u, \{v_m\}, p'-1, j).lt$ are computed for each j, $p' \leq j \leq m$, and hence we can determine the largest j, $p' \leq j \leq m$, for which $PPW2'(G_{p'}^u, \{v_m\}, p'-1, j).ft$ holds, which is exactly the value we want. If $v \neq w$ and v is not a stick of w, then $G_{p'}^u$ contains two chordless cycles, and we can do a similar thing.

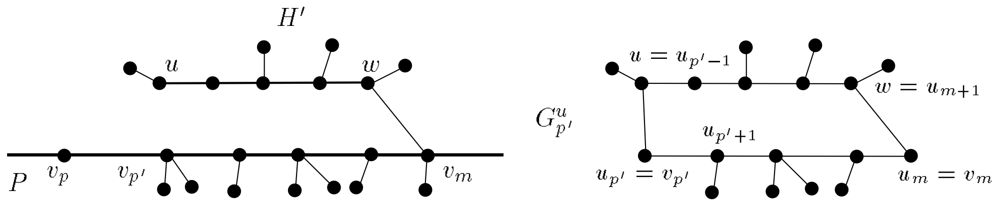

Figure 37: Example of $G_{p'}^u$, and the numbering of the vertices in its chordless cycle.

In the same way this can be done for w. This gives (at most) two values for j. j_1 is the largest of these two values, so it can be computed in $O(n^2)$ time. □

Case 2 $nr = 1$ and $p \leq j \leq j' \leq n$

We consider three sub cases, namely

2.1 $p \leq j \leq m \leq j' \leq n$,

2.2 $p \leq j \leq j' < m$, and

2.3 $m < j \leq j' \leq n$.

For each case, we show which local information must be computed, and how it is computed.

Case 2.1 $p \leq j \leq m \leq j' \leq n$
We first analyze the structure of a nice proper path decomposition with nice path P in which H' uses $[j, j']$, $p \leq j \leq m \leq j' \leq n$.

Claim 4.4. If H' uses $[j, j']$ for some j, j', $p \leq j \leq m \leq j' \leq n$, then a partial one-path H'' connected to $v_{m'}$, $H' \neq H''$, can use $[l, l']$ with $l' \leq j$ if $m' < m$, and $l \geq j'$ if $m' > m$.

Proof. Suppose there is a nice proper path decomposition $PD = (V_1, ..., V_t)$ with nice path P in which H' uses $[j, j']$ for some j and j' with $p \leq j \leq m \leq j' \leq n$. Let H'' be a partial one-path connected to $v_{m'}$, $H'' \neq H'$, and suppose H'' uses $[l, l']$. Clearly, if $m' < m$, then $l' \leq j$, and if $m' \geq m$ then $l \geq j$.

In the same way as for Claim 4.1, we can show that it is possible that $l' = j$ of $l = j'$. □

The claim implies that we only need all values of (j, j'), $p \leq j \leq m \leq j \leq n$, for which H' can use $[j, j']$, and there are no l, l', $j \leq l \leq m \leq l' \leq j'$, such that H' can use $[l, l']$ and $j < l$ or $l' < j'$.

Definition 4.6. *The local information for H' for the case that H' uses $[j, j']$, $p \leq j \leq m \leq j' \leq n$, is the set*

$$\begin{aligned} Q_1 \;=\; & \{ (j, j') \mid p \leq j \leq m \leq j' \leq n \wedge H' \textit{ can use } [j, j'] \\ & \wedge \neg \exists_{l,l'} (j < l \leq m \leq l' \leq j' \vee j \leq l \leq m \leq l' < j') \wedge H' \textit{ can use } [l, l'] \} \end{aligned}$$

We now show how to compute the set Q_1, and that $|Q_1| \leq 4$.

Let $P' \in \mathcal{P}_1(H')$, let u and w be the two end points of P'. Let $v \in V(H')$ such that $\{v, v_m\} \in E(H)$. For each j, $p \leq j \leq m$, let G_j^u denote the graph obtained from H as follows (see e.g. Figure 38). Add edge $\{u, v_j\}$. If v is a stick of u, check if $c(u) \neq c(v_m)$, and if so, add edge $\{u, v_m\}$, and delete v and its incident edges. Similarly, if v is a stick of w, check if $c(w) \neq c(v_m)$, and if so, add edge $\{w, v_m\}$, and delete v and its incident edges. If $c(w) = c(v_m)$, then G_j^u is undefined. Let v again denote the vertex of H' for which $\{v, v_m\}$ is an edge. Furthermore, delete vertices $\{v_1, ..., v_{j-1}, v_{m+1}, ..., v_q\}$, and all sticks and partial one-paths connected to $\{v_1, ..., v_j, v_m, ..., v_q\}$, except H'. Delete all components of $H'[V(H') - \{v\}]$ which do not contain u. Note that the remaining graph G_j^u is a chordless cycle with sticks. In a similar way, define G_j^u for all j, $m \leq j' \leq n$, and G_j^w for all j with $p \leq j \leq m$, or $m \leq j \leq n$.

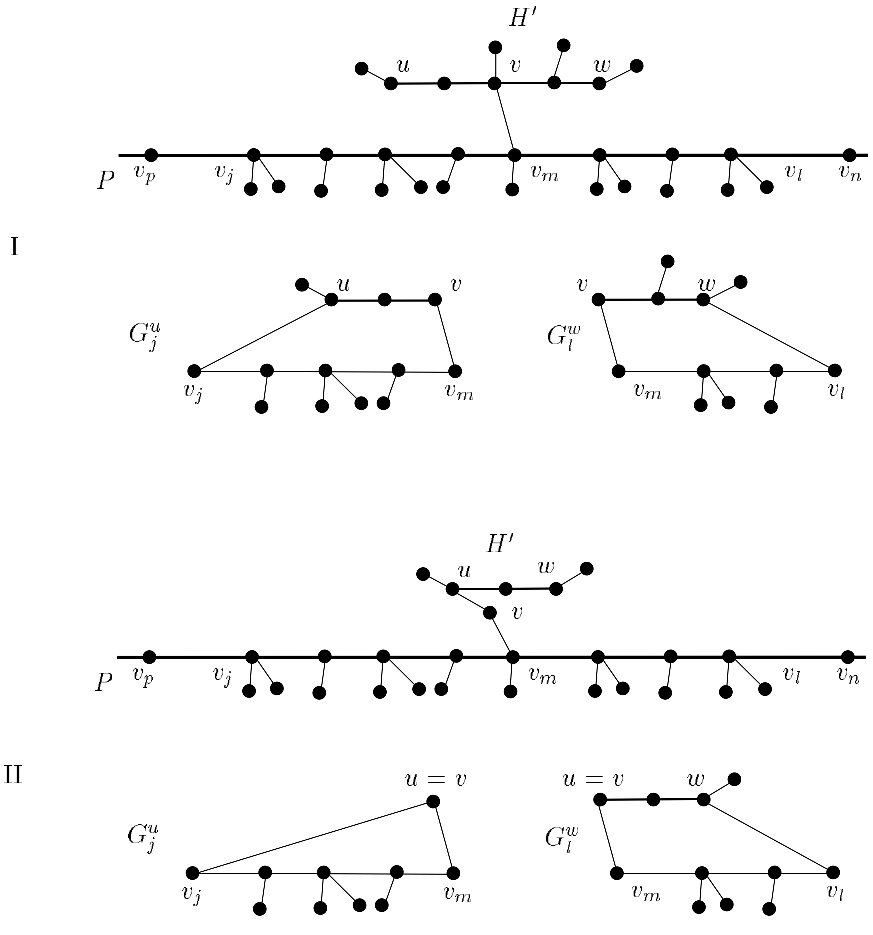

Figure 38: Example of G_j^u and G_l^w, with $p \leq j \leq m$ and $m \leq l \leq n$. In Part I, v is a vertex of $P_1(H')$. In Part II, v is a stick of u, which means that $c(v_m) \neq c(u)$ must hold, and v is deleted (u is the new vertex that is adjacent to v_m).

Let l_1^u be the largest value of j, $p \le j \le m$, for which G_j^u is defined, and there is a proper path decomposition of $G_j^u \cup \{$ sticks of v_m $\}$ with edge $\{v_m, v\}$ in the rightmost node, edge $\{v_j, u\}$ in the leftmost node, or $j = m$ and there is a proper path decomposition of G_m^u with edge $\{u, v_m\}$ in the leftmost node and edge $\{v, v_m\}$ in the rightmost node. If there is no such j, then l_1^u is undefined.

Let r_1^w be the smallest value of j', $m \le j' \le n$, for which $G_{j'}^w$ is defined, and there is a proper path decomposition of $G_j^w \cup \{$ sticks of v $\}$ with edge $\{v_m, v\}$ in the leftmost node and edge $\{v_{j'}, u\}$ in the rightmost node. If there is no such j', then r_1^w is undefined.

Let l_2^u be the largest value of j, $p \le j \le m$, for which G_j^u is defined, and there is a proper path decomposition of $G_j^u \cup \{$ sticks of v $\}$ with edge $\{v_m, v\}$ in the rightmost node and $\{v_j, u\}$ in the leftmost node. If there is no such j, l_2^u is undefined.

Let r_2^w be the smallest value of j', $m \le j' \le n$, for which G_j^w is defined, and there is a proper path decomposition of $G_j^w \cup \{$ sticks of v_m $\}$ with edge $\{v_m, v\}$ in the leftmost node and edge $\{v_{j'}, u\}$ in the rightmost node, of $j' = m$ and there is a proper path decomposition of $G_{j'}^w$ with edge $\{v, v_m\}$ in the leftmost node and edge $\{w, v_m\}$ in the rightmost node. If there is such a j', then r_2^w is undefined.

Similarly, define l_1^w, r_1^u, l_2^w and r_2^u.

Let Q_1' be defined as follows.

$$Q_1' = \{(l_1^u, r_1^w), (l_2^u, r_2^w), (l_1^w, r_1^u), (l_1^w, r_1^u)\}$$

Claim 4.5.

$Q_1 = \{\, (j, j') \in Q_1' \mid j$ and j' are defined $\wedge \neg\exists_{(l,l') \in Q_1'} (j < l \le l' \le j \vee j \le l \le l' < j') \,\}$

Proof. We first show that for each pair $(j, j') \in Q_1$, there is a pair $(l, l') \in Q_1'$, such that $j \le l \le l' \le j'$.

Let $PD = (V_1,..., V_t)$ be a nice proper path decomposition of H with nice path P, such that H' uses $[j, j']$ for some pair $(j, j') \in Q_1$. Suppose v_m occurs in $(V_r,..., V_{r'})$ and H' occurs in $(V_s,..., V_{s'})$. Let $P' \in \mathcal{P}_1(H')$ as defined before, with end points u and w, suppose w.l.o.g. that $u \in V_s$ and $w \in V_{s'}$. Let $u', w' \in V(H')$ such that $u' \in V_s$, $w' \in V_{s'}$, and u' is a stick adjacent to u, w' is a stick adjacent to w. Let $v \in V(H')$ such that $\{v, v_m\} \in E(H)$. If v is a stick of u or w, then there is a node containing v_m, v and u, or v_m, v and w, respectively, because of Lemma 3.11, and because $j \le m \le j'$.

Let G be the graph obtained from H by adding edges $\{u, v_j\}$ and $\{w, v_{j'}\}$, and if v is a stick of u, add edge $\{u, v_m\}$, and delete v and its incident edges, similarly if v is a stick of w. Let v denote the new vertex of H' for which $\{v, v_m\} \in E(H)$. Note that PD is a proper path decomposition of G. Let G' be the induced subgraph of G obtained by deleting the vertices of $\{v_1,..., v_{j-1}, v_{j'+1},..., v_q\}$ and the sticks and partial one-paths connected to vertices $\{v_1,..., v_j, v_{j'},..., v_q\}$. See e.g. Figure 39. Then G' is a biconnected component with sticks, and G' is the union of the graphs G_j^u and $G_{j'}^w$, and the sticks of v_m and v. Suppose G' occurs in $(V_a,..., V_{a'})$. Clearly, $a \le s$ and $a' \ge s'$. In fact, $s = a$ and $s' = a'$, since all vertices $v_{j+1},..., v_{j'-1}$ and sticks adjacent to these vertices occur only within $(V_s,..., V_{s'})$. Furthermore, $\{u, v_j\} \subseteq V_s$ and $\{w, v_{j'}\} \subseteq V_{s'}$.

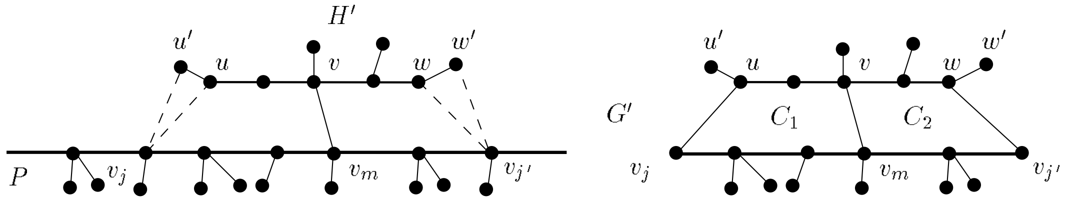

Figure 39: Example of a partial one-path H' which uses $[j, j']$, $p \leq j \leq m \leq j' \leq n$, and the corresponding graph G' with chordless cycles C_1 and C_2.

G' contains two chordless cycles which have edge $\{v_m, v\}$ in common. Let C_1 and C_2 denote these chordless cycles, such that C_1 contains edge $\{u, v_j\}$ and C_2 contains edge $\{w, v_{j'}\}$, let b, $a \leq b \leq a'$, be such that no vertex of $V(C_1) - \{v_m, v\}$ occurs on the right side of V_b, and no vertex of $V(C_2) - \{v_m, v\}$ occurs on the left side of V_b (this is possible, see Lemma 3.3). Note that $\{v_m, v\} \subseteq V_b$. The sticks of v_m which have color $c(v)$ occur either on the left side of the occurrence of V_b or on the right side of the occurrence of V_b (it is not necessary that one of them occurs on the left side, and another one on the right side, since they all have the same color), and the sticks of v which have color $c(v_m)$ occur either on the left side of V_b or on the right side. If v_m has sticks of color $c(v)$ and v has sticks of color $c(v_m)$, then the sticks of v and v_m do not both occur on the same side of V_b (see Lemma 4.7). Delete all sticks of v_m which do not have color $c(v)$, and all sticks of v which do not have color $c(v_m)$ from PD, and for each of these sticks w, add a node $\{v_m, v, w\}$ between V_b and V_{b+1}. Let $(V_a, \ldots, V_{a''})$ be the new occurrence of G'', and let $(V_b, \ldots, V_{b'})$ be the occurrence of all these sticks. Suppose w.l.o.g. that the sticks of v_m of color $c(v)$ occur on the left side of V_b. Then $(V_a, \ldots, V_{b'})$ is a proper path decomposition of $G^u_j \cup \{\text{sticks of } v_m\}$ if $j < m$, and of G^u_j if $j = m$, such that $\{u, v_j\}$ is in the leftmost node and $\{v, v_m\}$ is in the rightmost node, and $(V_b, \ldots, V_{a''})$ is a proper path decomposition of $G^w_{j'} \cup \{\text{sticks of } v\}$ with $\{v, v_m\}$ in the leftmost node and $\{w, v_{j'}\}$ in the rightmost node. Hence $j \leq l^u_1$ and $j' \geq r^w_1$.

Showing that for all pairs $(j, j') \in Q'_1$, there is a pair $(l, l') \in Q_1$ is similar to part 2 of the proof of Claim 4.2. □

Claim 4.6. Q_1 can be computed in time $O(n^2)$, where n is the number of vertices of $G^u_p \cup G^w_n \cup \{\text{sticks of } v_m \text{ and } v\}$.

Proof. The values of l^u_1, r^w_1, etc. can be computed by using PPW1, which has to be computed once for each of the four values (see proof of Claim 4.3). □

Case 2.2 $p \leq j \leq j' < m$

We first analyze the structure of a nice proper path decomposition with nice path P in which H' uses $[j, j']$, $p \leq j \leq j' < m$.

Claim 4.7. Suppose H' uses $[j, j']$ for some j, j', $p \leq j \leq j' < m$, and let w be the end point of some path $P' \in \mathcal{P}_1(H')$ such that there is a node which contains w and

$v_{j'}$. H' is a partial one-path of type I, and either w is adjacent to v_m, or a stick of w is adjacent to v_m.

Proof. Suppose H' occurs in $(V_s,\ldots,V_{s'})$. $m > j'$, which means that v_m occurs only on the right side of $V_{s'}$. But w and some stick of w are the only vertices of H' which occur in $V_{s'}$, hence w or this stick is adjacent to v_m. $\square$

Claim 4.8. Suppose there is a nice proper path decomposition $PD = (V_1,\ldots,V_t)$ of H with nice path P in which H' uses $[j,j']$ for some j,j', $p \le j \le j' < m$. Let a be the maximum of m and the largest value of i, $i \le n$, for which there is a node V_b in PD, an integer $i' < m$, and a stick w of $v_{i'}$, such that V_b contains v_i and edge $\{v_{i'}, w\}$. Then a partial one-path H'' with $H' \ne H''$, H'' connected to $v_{m'}$, can use $[l,l']$ with $l' \le j$ if $m' < m$, and $l \ge a$ if $m' > m$.

Proof. Let H'' be a partial one-path connected to $v_{m'}$, $H'' \ne H'$, and suppose H'' uses $[l,l']$. Clearly, if $m' < m$, then $l' \le j$, and $l' = j$ is possible (see also proof of Claim 4.1).

Consider the case that $m' > m$. Clearly, $l \ge m$. Suppose v_m occurs in $(V_r,\ldots,V_{r'})$ and H' occurs in $(V_s,\ldots,V_{s'})$. Note that $s' < r$. Let $P' \in \mathcal{P}_1(H')$, let $u \in V(H')$ such that u is end point of P', and $u \in V_s$. Let G be the graph obtained from H by adding edge $\{u, v_j\}$ (see e.g. Figure 40). Note that PD is a proper path decomposition of G. Let G' be the subgraph of G induced by the vertices of H' and vertices $v_j,\ldots,v_m$ and all sticks adjacent to $v_{j+1},\ldots,v_{m-1}$. Note that G' is a chordless cycle with sticks. Let C denote the chordless cycle in G'. Suppose G' occurs in $(V_a,\ldots,V_{a'})$. Then $a = s$. Vertex v_m occurs in the rightmost node of the occurrence of C, so if $i \le m$, then $v_m \in V_{a'}$. In this case, all vertices of $V(G') - \{v_m\}$ may be deleted from all nodes on the right side of $V_{a'}$, and we can add a node $\{v_m\}$ between $V_{a'}$ and $V_{a'+1}$, which means that $l = m$ is possible.

Suppose $i > m$. Let G'' be the graph obtained from G by adding edge $\{w, v_i\}$. Note that PD is a proper path decomposition of G''. See for example Figure 40. Let G''' be the subgraph of G'' induced by the vertices of H', vertices $v_j,\ldots,v_i$, and all sticks adjacent to vertices $v_{j+1},\ldots,v_{i-1}$. Note that G''' is a biconnected component with sticks, which contains two chordless cycles, and $\{u, v_j\}$ occurs in the leftmost node of its occurrence, and vertex $\{v_i\}$ occurs in the rightmost node. Each node in the occurrence of G''' contains at least two vertices of G''' which means that there is no vertex v_c, $m < c < i$, which has a partial one-path connected to it, and it is not possible that $l < i$. Furthermore, $l = i$ is possible, since we can add a node $\{v_i\}$ on the right side of the occurrence of G'''. $\square$

The claim implies that we only need all values of (j,j'), $p \le j \le m \le j' \le n$, for which H' can use $[j,j'']$, for some $j \le j'' < m$, and other partial one-paths can use $[l,l']$, where $l' \le j$ or $l \ge j'$, and there are no a,a', $j < a \le m \le a' < j'$, such that H' can use $[a,a'']$ for some $a \le a'' < m$, and other partial one-paths can use $[b,b']$, $b' \le a$ or $b \ge a'$.

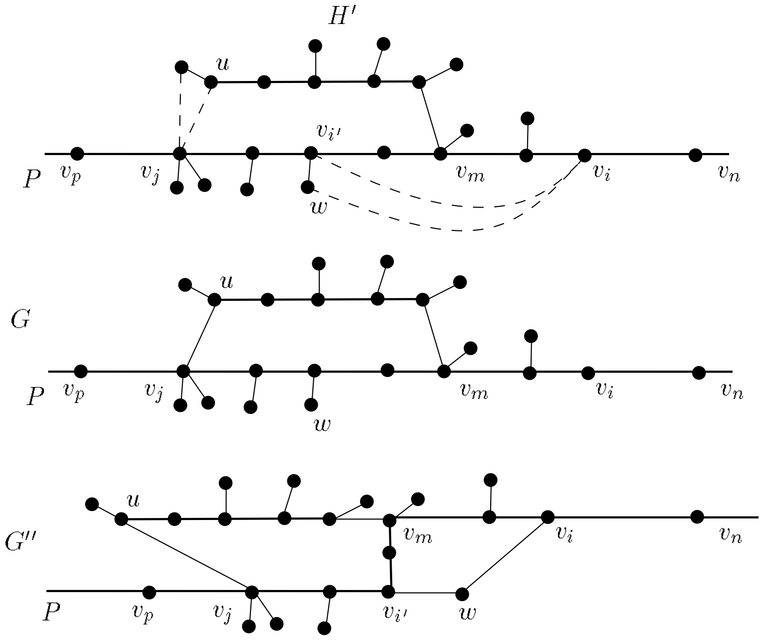

Figure 40: Example of a partial one-path H' which uses $[j, j']$, $p \leq j \leq j' < m$, such that the edge $\{v_{i'}, w\}$ 'uses' i, $i \geq m$, and the corresponding graphs G and G'',

Definition 4.7. *The local information for H' for the case that H' uses $[j, j']$, $p \leq j \leq j' < m$, is the set*

$$\begin{aligned} Q_2 \;=\; \{\, (j, j') \mid{} & p \leq j \leq m \leq j' \leq n \\ & \wedge\ H' \text{ can use } [j, j''],\ j \leq j'' < m, \\ & \wedge \text{ other partial one-paths can use } [l, l'],\ l' \leq j \text{ or } l \geq j' \\ & \wedge\ (\neg\exists_{a,a'}\ (j < a \leq m \leq a' \leq j' \vee j \leq a \leq m \leq a' < j') \\ & \quad \wedge\ H' \text{ can use } [a, a''],\ a \leq a'' < m, \\ & \quad \wedge \text{ other partial one-paths can use } [b, b'],\ b' \leq j \text{ or } b \geq a')\,\} \end{aligned}$$

We now show how to compute the set Q_2, and that $|Q_2| \leq 2$.

Let $P' \in \mathcal{P}_1(H')$, let u and w be the two end points of P', such that w or a stick of w is adjacent to v_m. For each j, $p \leq j \leq m$, let G_j denote the graph obtained from H as follows (see e.g. Figure 41). Add edge $\{u, v_j\}$. Furthermore, delete vertices $\{v_1, \ldots, v_{j-1}, v_{m+1}, \ldots, v_q\}$, and all sticks and partial one-paths connected to $\{v_1, \ldots, v_j, v_m, \ldots, v_q\}$, except H'. Note that the graph G_j is a chordless cycle with sticks.

Furthermore, for each j, $p \leq j \leq m$, let G'_j denote the graph obtained from H as follows. Add edge $\{u, v_j\}$. Furthermore, delete vertices $\{v_1, \ldots, v_{j-1}, v_{m+1}, \ldots, v_q\}$, and all sticks and partial one-paths connected to $\{v_1, \ldots, v_j, v_{m+1}, \ldots, v_q\}$. If v_{m-1} has sticks, delete them. If v_{m-1} does not have sticks, check if $c(v_{m-2}) = c(v_m)$, if so, G'_j is

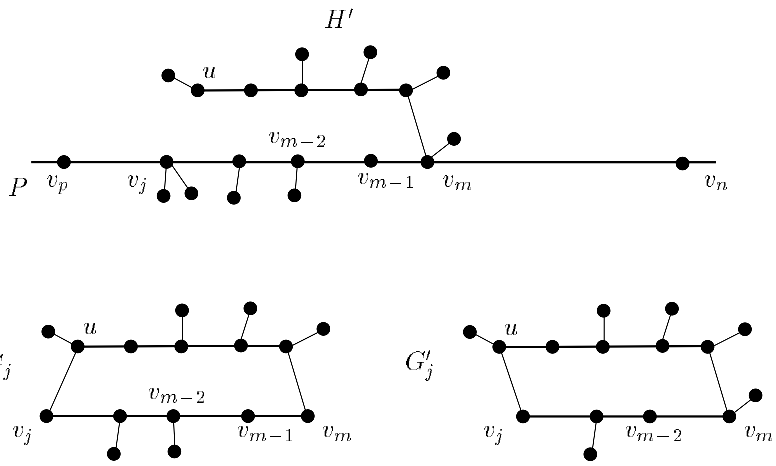

Figure 41: Example of graphs G_j and G'_j for a partial one-path H' connected to v_m, where v_{m-1} has no sticks.

undefined, otherwise, delete v_{m-1} and its incident edges, and delete the sticks of v_{m-2}. Note that the graph G'_j is also a chordless cycle with sticks.

Let l_1 be the largest j, $p \leq j < m$, for which there is a proper path decomposition of G_j with vertex v_m in the rightmost node and edge $\{u, v_j\}$ in the leftmost node. If there is no such proper path decomposition, then l_1 is undefined.

Let l_2 be the largest j, $p \leq j < m$, for which G'_j is defined and there is a proper path decomposition of G'_j with edge $\{u, v_j\}$ in the leftmost node and edge $\{v_{m-2}, v_m\}$ in the rightmost node if v_{m-1} is deleted, end edge $\{v_{m-1}, v_m\}$ in the rightmost node otherwise. If there is no such proper path decomposition, then l_2 is undefined.

Let Q'_2 be defined as follows.

$$Q'_2 = \{(l_1, m), (l_2, m + 1)\}$$

Claim 4.9.

$$Q_2 = \{\, (j, j') \in Q'_2 \mid j \text{ is defined } \wedge \neg\exists_{(l,l') \in Q'_2} (j < l \leq l' \leq j' \vee j \leq l \leq l' < j')\,\}$$

Proof. We first show that for each pair $(j, j') \in Q_2$, there is a pair $(l, l') \in Q_2$ with $j \leq l \leq l' \leq j'$.

Suppose there is a nice proper path decomposition $PD = (V_1, ..., V_t)$ of H with nice path P, such that H' uses $[j, j'']$, $p \leq j \leq j'' < m$, and other partial one-paths may use $[l, l']$, $l' \leq j$ or $l' \geq j'$ for some $j' \geq m$. Suppose w.l.o.g. that $(j, j') \in Q_2$.

Let $P' \in \mathcal{P}_1(H')$ be as defined above, with end points u and w. Suppose w.l.o.g. that v_m is adjacent to w or a stick adjacent to w. Let G be the graph obtained from H by adding edge $\{v_j, u\}$. Note that PD is a proper path decomposition of G, and that G_j is a subgraph of G. Let $(V_s, ..., V_{s'})$ denote the occurrence of G_j in PD. Then $\{u, v_j\} \subseteq V_s$ and if $j' = m$, then $v_m \in V_{s'}$, and hence there is an l, $j \leq l \leq m$, for which $(l, m) \in Q'_2$.

Suppose $j' > m$. Let $i < m$ such that there is a node containing $v_{j'}, v_i$ and a stick of v_i. Let G' be the graph obtained from G by adding edge $\{v_{j'}, v_i\}$, and deleting all vertices $v_1,..., v_{j-1}$, $v_{j'+1},..., v_q$, and all sticks adjacent to vertices $v_1,..., v_j$, $v_{j'},..., v_q$. See for example Figure 42. Note that $PD[V(G')]$ is a proper path decomposition of

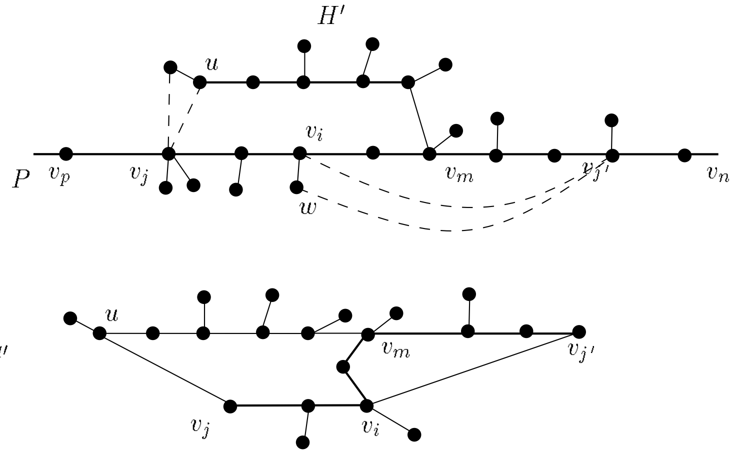

Figure 42: Example of graph G'.

G', and note that G' is a biconnected component with sticks. There are three disjoint paths in G' from v_i to v_m, which means that there is a node containing v_i and v_m (Lemma 3.1). Let G'' be the graph obtained from G by adding edge $\{v_i, v_m\}$. See for example Figure 43. If $i = m - 1$, then G'' consists of two chordless cycles which have edge $\{v_i, v_m\}$ in common. If $i < m - 1$, then G'' contains three chordless cycles which

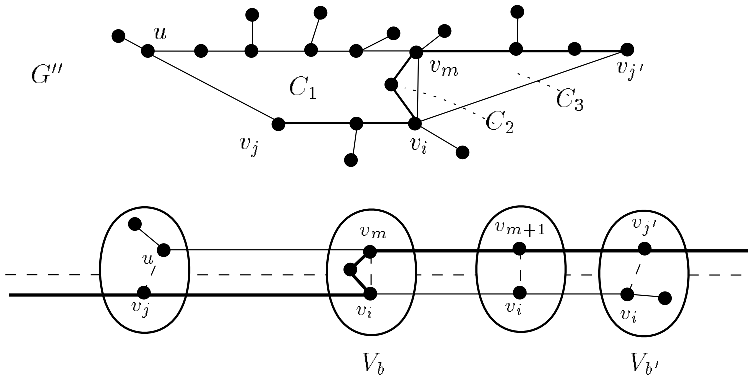

Figure 43: Example of graph G'', and the occurrence of G''. Chordless cycle C_3 occurs in $(V_b,..., V_{b'})$.

have edge $\{v_i, v_m\}$ in common (Theorem 3.1). Let C_1 denote the chordless cycle which contains v_j, let C_3 denote the chordless cycle which contains $v_{j'}$, and if $i < m - 1$, let C_2 denote the chordless cycle which contains vertices $v_i,..., v_m$, If $i < m - 1$, then one of the three chordless cycles must have three vertices, such that the third vertex of this cycle has no sticks. This must be C_2, since C_2 is the chordless cycle which occurs in between C_1 and C_3 in PD (Theorem 3.1). Hence if $i < m-1$, then $i = m-2$, and v_{m-1} has no sticks, so G'_j is a subgraph of G''.

Consider the occurrence (V_b,..., $B_{b'}$) of C_3 (see Figure 43). Edge $\{v_i, v_m\} \subseteq V_b$, and edge $\{v_i, v_{j'}\} \subseteq V_{b'}$. Since $j' \geq m+1$, this means that there is a node V_c, $b \leq c \leq c'$, such that $v_i, v_{m+1} \in V_c$. This means that $c(v_m) \neq c(v_i)$ and $c(v_{m+1}) \neq c(v_i)$. Furthermore, $c(v_m) \neq c(v_{m+1})$, which means that all sticks of v_i either have color $c(v_m)$ or color $c(v_{m+1})$. So we can modify PD in such a way that for each stick w of v_i with color $c(v_m)$, there is a node $\{v_i, v_{m+1}, w\}$ in PD, and for each stick w of color $c(v_{m+1})$ of v_i, there is a node $\{v_i, v_m, w\}$ in PD, and v_i can be deleted from all nodes which contain v_d, $d > m+1$ (see Lemma 4.2). In this modified version of PD, the occurrence of G'_j contains edge $\{v_j, u\}$ in the leftmost node and edge $\{v_i, v_{m+1}\}$ in the rightmost node. Hence l_2 is defined, and $l_2 \geq j$, so $(l_2, m+1) \in Q'_2$.

Showing that for all pairs $(j, j') \in Q'_2$, there is a pair $(l, l') \in Q_2$ such that $j \leq l \leq l' \leq j'$ is similar to part 2 of the proof of Claim 4.2. □

Claim 4.10. Q_2 can be computed in time $O(n^2)$, where n is the number of vertices of $G_j \cup \{\text{sticks of } v_m\}$.

Proof. The value of l_1 can be computed by using $PPW2'$, and the value of l_2 can be computed by using PPW2. Both have to be computed once (see proof of Claim 4.3). □

Case 2.3 $m < j \leq j' \leq n$

This case is similar to case 2.2. The local information consists of the set Q_3, which contains at most two pairs (j, j'), and if there are two, then one of them has $j = m$, the other one has $j = m-1$.

This results in the following local information for Case 2.

Definition 4.8. *The local information for H' for the case that H' uses $[j, j']$, $p \leq j \leq j' \leq n$, is the set*

$$Q = \{\, (j, j') \in Q_1 \cup Q_2 \cup Q_3 \mid \neg\exists_{(l,l') \in Q_1 \cup Q_2 \cup Q_3}\, (j < l \leq l' \leq j' \vee j \leq l \leq l' < j')$$

Case 3 $pp \leq j \leq j' \leq p$

We first analyze the structure of a proper path decomposition in which H' uses $[j, j']$ for some $pp \leq j \leq j' \leq p$. We assume that there is a partial one-path H'' which is connected to v_p and which uses $[l, l']$ for some $l \geq m$, since otherwise, $j = j' = p$ and this case is considered in Case 1.

Claim 4.11. If H' uses $[j, j']$ for some j, j' with $pp \leq j \leq j' \leq p$, and there is a partial one-path H'' connected to v_p which uses $[l, l']$, $m \leq l \leq l' \leq n$, then a partial one-path H''' connected to $v_{m'}$, $H' \neq H'''$ and $H'' \neq H'''$, can use $[a, a']$, with $a' \leq j$ if $m' < m$ and $a \geq l'$ if $m' \geq m$.

Proof. It is clear that $a' \leq j$ if $m' < m$, and that $a \geq l'$ if $m' \geq m$ (Lemma 4.17). Showing that $a' = j$ and $a = l'$ are possible can be done in the same way as in the proof of Claim 4.1. □

It follows from the claim that we only need all pairs (j, l'), $pp \leq j \leq m \leq l' \leq n$, for which there are j' and l, $j \leq j' \leq m$ and $m \leq l \leq l'$, such that H' can use $[j, j']$ and there is a partial one-path H'' connected to v_p which can use $[l, l']$, and there is no pair (a, b') for which this holds with $j < a \leq b' \leq l'$ or $j \leq a \leq b' < l'$.

However, the local information for H' consists only of information about H'. Therefore, we first further analyze the occurrences of H' and H'', to show how H' and H'' can be handled independently.

Claim 4.12. Suppose there is a proper path decomposition $PD = (V_1,..., V_t)$ of H with nice path P, such that H' uses $[j, j']$, $pp \leq j \leq j' \leq p$, and there is a partial one-path H'' connected to v_p which uses $[l, l']$, $m \leq l \leq l' \leq n$. The following holds.

1. $m = p + 1$ or $m = p + 2$, and if $m = p + 2$, then v_{p+1} has degree two.
2. There is a node V_b which contains v_p, v_{p+1} and v_m.
3. If the sticks of v_p which have color $c(v_m)$ occur on the left side of the occurrence of $\{v_p, v_m\}$, then the sticks of v_m which have color $c(v_p)$ occur on the right side of the occurrence of $\{v_p, v_m\}$.
4. If the sticks of v_p which have color $c(v_m)$ occur on the right side of the occurrence of $\{v_p, v_m\}$, then the sticks of v_m which have color $c(v_p)$ occur on the left side of the occurrence of $\{v_p, v_m\}$.

Proof. 1 and 2 are proven in Lemma 4.18, so we only prove 3 and 4. Suppose H' occurs in $(V_r,..., V_{r'})$ and H'' occurs in $(V_s,..., V_{s'})$. Note that $r' < s$ (see Lemma 4.18). Let $P' \in \mathcal{P}_1(H')$ and $P'' \in \mathcal{P}_1(H'')$, and let $u \in V(H')$ be the end point of P' for which $u \in V_r$, and let $w \in V(H'')$ be the end point of P'' for which $w \in V_{s'}$. Note that the path from u to v_m contains P', and the same holds for H''. Let G be the graph obtained from H by adding edges $\{u, v_j\}$ and $\{w, v_{l'}\}$, and if $m = p + 1$, adding edge $\{v_p, v_m\}$ and deleting vertex v_{p+1} and its incident edges. See e.g. Figure 44. PD is a proper path decomposition of G. Let G' be the subgraph of G induced by the vertices of H', H'', $\{v_j,..., v_{l'}\}$ and the sticks of vertices $\{v_{j+1},..., v_{l'-1}\}$. G' is a biconnected component with sticks, which has two chordless cycles which have edge $\{v_p, v_m\}$ in common. Let C_1 and C_2 be the chordless cycles of G', such that C_1 contains vertices of H' and C_2 contains vertices of H''. Graph G' occurs in $(V_r,..., V_{s'})$, edge $\{u, v_j\}$ occurs in V_r and $\{w, v_{l'}\}$ occurs in $V_{s'}$. Let $(V_a,..., V_{a'})$ be the occurrence of $\{v_p, v_m\}$.

We first prove 3. If v_p has sticks of color $c(v_m)$, and these sticks occur on the left side of V_a, then either $j = p$ and the sticks occur on the left side of V_r, or the sticks occur within the occurrence of C_1. In the first case, each node of the occurrence of C_1 contains vertex v_p, and v_m does not occur on the left side of V_r, hence the sticks of v_m which have color $c(v_p)$ must occur on the right side of $V_{a'}$. In the second case, the

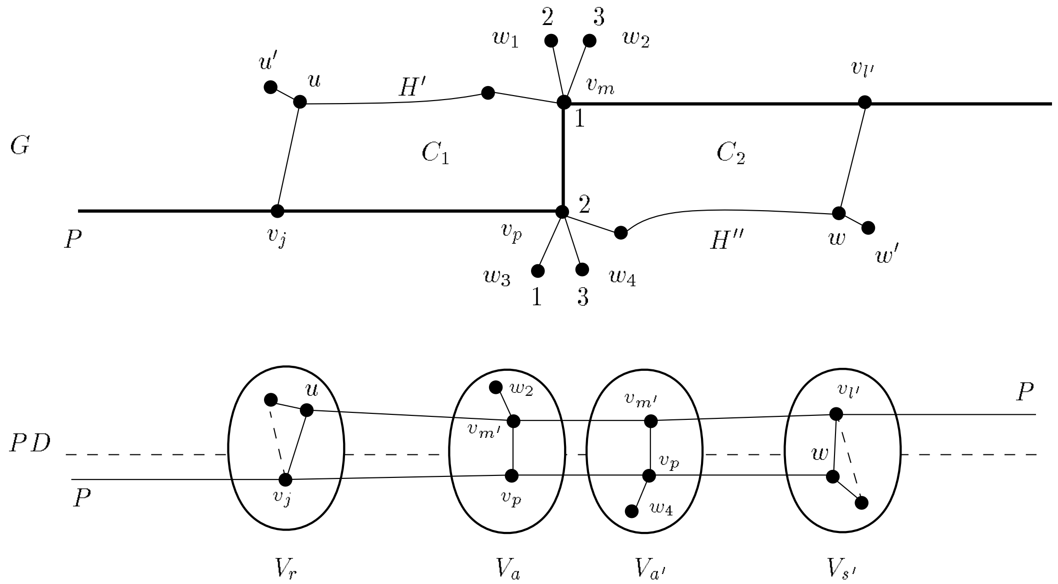

Figure 44: Example of the graph G as defined in the proof of Claim 4.12, and a part of a proper path decomposition of G. $c(v_m) = 1$, $c(v_p) = 2$, w_1 and w_2 are sticks of v_m, w_3 and w_4 are sticks of v_p, with $c(w_1) = 2$, $c(w_2) = c(w_4) = 3$, and $c(w_3) = 1$. Sticks w_1 and w_4 occur within the occurrence of $\{v_p, v_m\}$.

sticks of v_m of color $c(v_p)$ can not occur within the occurrence of C_1 (see Lemma 4.7). Furthermore, they can not occur on the left side of V_r, since v_m does not occur there. Hence the sticks of v_m of color $c(v_p)$ occur on the right side of $V_{a'}$.

We now prove 4. If v_p has sticks of color $c(v_m)$, and these sticks occur on the right side of $V_{a'}$, then they must occur within the occurrence of C_2, since v_p does not occur on the right side of $V_{s'}$. Then the sticks of v_m which have color $c(v_p)$ can not occur within the occurrence of C_2 (Lemma 4.7). Furthermore, $l' > m$, because if $l' = m$, then each node of the occurrence of C_2 contains v_m, which means that the sticks of v_p which have color $c(v_m)$ can not occur within this occurrence. Because $l' > m$, it is not possible that the sticks of v_m which have color $c(v_p)$ occur on the right side of $V_{s'}$. Hence they must occur on the left side of V_a. □

The claim implies that if H' uses $[j, j']$, $pp \leq j \leq j' \leq p$, then either the sticks of v_p which have color $c(v_m)$ occur on the left side of the occurrence of $\{v_p, v_m\}$, or the sticks of v_m which have color $c(v_p)$ occur on the left side of the occurrence of $\{v_p, v_m\}$, but not both. Therefore, the local information is defined as follows.

Definition 4.9. *The local information for H' for the case that H' uses $[j, j']$, $pp \leq j \leq j' \leq p$, and there may be a partial one-path H'' connected to v_m which uses $[l, l']$, $pp \leq l \leq l' \leq p$, is the pair (j_1, j_2), $pp \leq j_1, j_2 \leq p$, where*

- *j_1 is the largest value of j, $pp \leq j \leq p$, for which H' can use $[j, j']$ for some $j \leq j' \leq p$, and the sticks of v_p which have color $c(v_m)$ occur on the left side of the occurrence of $\{v_p, v_m\}$ (j_1 is undefined if there is no such j) and*

- *j_2 is the largest value of j, $pp \leq j \leq p$, for which H' can use $[j, j']$ for some $j \leq j' \leq p$, and the sticks of v_m which have color $c(v_p)$ occur on the left side of the occurrence of $\{v_p, v_m\}$ (j_2 is undefined if there is no such j).*

We now show how to compute j_1 and j_2.

Let $P' \in \mathcal{P}_1(H')$, let u be the end point of P' for which the path from u to v_m contains P'. For each j, $pp \leq j \leq p$, let G_j denote the graph obtained from H as follows (see e.g. Figure 45). Take the graph induced by H', v_m, $\{v_j,..., v_p\}$ and the sticks of $v_{j+1},..., v_p$ and v_m. Add edge $\{u, v_j\}$ and if $m = p+2$, check if $c(v_p) \neq c(v_m)$, add edge $\{v_p, v_m\}$, and delete v_{p+1} and its incident edges. If $m > p+2$ or $c(v_p) = c(v_m)$, then G_j is undefined. Note that G_j is a biconnected component with sticks.

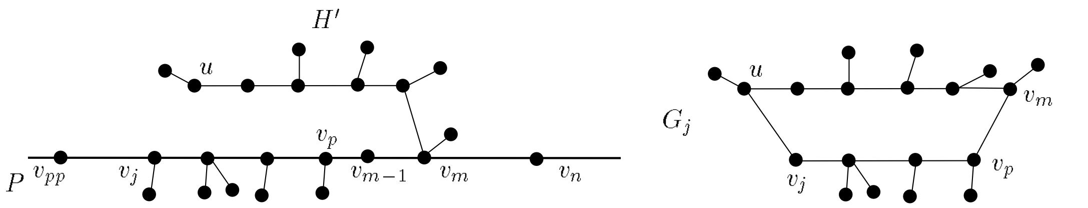

Figure 45: Example of G_j for the case that $m = p + 2$.

Let l_1 be the largest value of j, $pp \leq j \leq p$, for which there is a proper path decomposition of $G_j - \{$ sticks of v_m $\}$ in which edge $\{v_j, u\}$ occurs in the leftmost node and edge $\{v_p, v_m\}$ occurs in the rightmost node. If there is no such proper path decomposition, then l_1 is undefined.

Let l_2 be the largest value of j, $pp \leq j \leq p$, for which there is a proper path decomposition of $G_j - \{$ sticks of v_p $\}$ in which edge $\{v_j, u\}$ occurs in the leftmost node and edge $\{v_p, v_m\}$ occurs in the rightmost node. If there is no such proper path decomposition, then l_2 is undefined.

Claim 4.13. $j_1 = l_1$ and $j_2 = l_2$.

Proof. We first show that $j_1 \leq l_1$ and $j_2 \leq l_2$.

Suppose j_1 is defined, and suppose there is a nice proper path decomposition $PD = (V_1,..., V_t)$ of H with nice path P such that H' uses $[j_1, j']$ for some j' with $j_1 \leq j' \leq p$, there is a partial one-path H'' connected to v_p which uses $[l, l']$ for some $m \leq l \leq l' \leq n$, and sticks of v_p which have color $c(v_m)$ occur on the left side of the occurrence of $\{v_p, v_m\}$.

Suppose H' occurs in $(V_r,..., V_{r'})$ and H'' occurs in $(V_s,..., V_{s'})$. Let $P' \in \mathcal{P}_1(H')$ be as defined above, such that $u \in V_s$, let $P'' \in \mathcal{P}_1(H'')$ and let $w \in V(H'')$ be the end point of P'' for which $w \in V_{s'}$. Let G, C_1 and C_2 be as defined in the proof of

Claim 4.11 of this case (see also Figure 44). Note that G_{j_1} is a subgraph of G. There is a node which contains $\{v_p, v_m\}$, hence we can modify PD in such a way that for each stick w of v_p which has color $c(v_m)$, or w stick of v_m which has color $c(v_p)$, there is a node $\{v_p, v_m, w\}$. Let $(V_a,..., V_{a'})$ be the occurrence of $\{v_p, v_m\}$ in the modified path decomposition. The sticks of v_p which have color $c(v_m)$ occur on the left side of V_a, which means that $G_{j_1} - \{$ sticks of $v_m\}$ occurs in $(V_s,..., V_{a'})$, with edge $\{v_{j_1}, u\}$ in the leftmost node and edge $\{v_m, v_p\}$ in the rightmost node. Hence l_1 is defined, and $l_1 \geq j_1$.

In the same way we can prove that $l_2 \geq j_2$.

Showing that $j_1 \geq l_1$ and $j_2 \geq l_2$ can be done in the same way as in the proof of Claim 4.2. □

Claim 4.14. j_1 and j_2 can be computed in $O(n^2)$ time, where n is the number of vertices of G_{pp}.

Proof. PPW2 can be used to compute j_1 and j_2. The procedure to compute PPW2 must be called once for j_1, and once for j_2 (see also proof of Claim 4.3). □

This completes the description of Cases 1, 2, and 3.

During the algorithm, we use the following record to store all local information for each vertex of the path to which one or more partial one-paths are connected.

Definition 4.10. *Let H be a three-colored partial two-path, $P = (v_1,..., v_q)$ a possible nice path for H. For each m, $1 \leq m \leq q$, v_m is a record with fields nr and p.*

The field $v_m.nr$ denotes the number of partial one-paths connected to v_m, $v_m.p$ is an array of $v_m.nr$ records with fields H, t, l, r, lr, lll, llr, rrl and rrr, which are defined as follows. Let pp, p, n and nn be as defined before. For each i, $1 \leq i \leq v_m.nr$,

- *$v_m.p[i].H$ denotes the ith partial one-path connected to v_m.*
- *$v_m.p[i].t$ denotes the type of $v_m.p[i].H$, i.e. $v_m.p[i].t \in \{\mathrm{I}, \mathrm{II}, \mathrm{III}\}$.*
- *$v_m.p[i].l$ stores the local information for Case 1:*

  $v_m.p[i].l$ has two fields: ok and v which denote the following. $v_m.p[i].l.ok$ is a boolean which is true if and only if $v_m.nr > 1$ and j_1 as defined in Definition 4.5 exists, false otherwise. If not $v_m.p[i].l.ok$, then $v_m.p[i].l.v = p$, otherwise $v_m.p[i].l.v = j_1$.
- *$v_m.p[i].r$ stores the local information for Case 4:*

  $v_m.p[i].r$ has two fields: ok and v which are defined in the as for $v_m.p[i].l$, but for the case that $v_m.p[i].H$ uses $[j, j']$, $m \leq j \leq j' \leq n$.
- *$v_m.p[i].lr$ stores the local information for Case 2:*

  $v_m.p[i].lr$ has 17 fields: ok, and for all a, $1 \leq a \leq 8$, fields l_a and r_a, which denote the following. $v_m.p[i].lr.ok$ is a boolean which is true if and only if $v_m.nr = 1$

and the set Q as defined in Definition 4.8 is not empty. If not $v_m.p[i].lr.ok$, then $v_m.p[i].lr.l_a = p$ and $v_m.p[i].lr.r_a = n$ for each a, $1 \leq a \leq 8$. If $v_m.p[i].lr.ok$, then $v_m.p[i].lr.l_a$ and $v_m.p[i].lr.r_a$, $1 \leq a \leq 8$, are such that

$$Q = \{ \ (v_m.p[i].lr.l_a, v_m.p[i].lr.r_a) \mid 1 \leq a \leq 8 \ \}$$

- *$v_m.p[i].lll$ and $v_m.p[i].llr$ store the local information for Case 3.*
  - *$v_m.p[i].lll$ has two fields: ok and v, which denote the following. $v_m.p[i].lll.ok$ is true if and only if j_1 as defined in Definition 4.9 is defined, and $v_m.nr \leq 1$ or $v_m.p[i].H$ has a vertex of color $c(v_m)$. If not $v_m.p[i].lll.ok$, then $v_m.p[i].lll.v = pp$. If $v_m.p[i].lll.ok$, then $v_m.p[i].lll.v = j_1$.*
  - *$v_m.p[i].llr$ has two fields: ok and v, which denote the following. $v_m.p[i].llr.ok$ is true if and only if j_2 as defined in Definition 4.9 is defined, and $v_m.nr \leq 1$ or $v_m.p[i].H$ has a vertex of color $c(v_m)$. If not $v_m.p[i].llr.ok$, then $v_m.p[i].llr.v = pp$. If $v_m.p[i].llr.ok$, then $v_m.p[i].llr.v = j_2$.*
- *$v_m.p[i].rrl$ and $v_m.p[i].rrr$ stores the local information for Case 5.*
  - *$v_m.p[i].rrl$ has two fields: ok and v, which are defined in the same way as for $v_m.p[i].lll$, except that $v_m.p[i].H$ must use $[j, j']$ for some $n \leq j \leq j' \leq nn$, and if $v_m.p[i].rrl.ok$ then $v_m.p[i].rrl.v$ is the largest j' for which this holds.*
  - *$v_m.p[i].rrr$ has two fields: ok and v, which are defined in the same way as for $v_m.p[i].llr$, except that $v_m.p[i].H$ must use $[j, j']$ for some $n \leq j \leq j' \leq nn$, and if $v_m.p[i].rrl.ok$ then $v_m.p[i].rrl.v$ is the largest j' for which this holds.*

The local information that is computed for each vertex $v \in V(P)$ in function Check_Nice_Path(P) consists of $v.p[i].l$, $v.p[i].r$, $v.p[i].lr$, $v.p[i].lll$, $v.p[i].llr$, $v.p[i].rrl$, and $v.p[i].rrr$, for all i, $1 \leq i \leq v.nr$.

Next we discuss which *global information* is computed in Check_Nice_Path, and how it is computed. Let $i_1,..., i_t$ denote the vertices of P for which $v_{i_j}.nr \geq 1$ for all j, $1 \leq j \leq t$, and $i_1 < i_2 < \cdots < i_t$. The main loop of Check_Nice_Path(P) has the following structure.

initialize global information variables
for $j := 1$ to t
$\rightarrow$ $m := i_j$;
  for $i := 1$ to $v_m.nr$
  $\rightarrow$ compute local information for v_m
  rof
  adapt global information variables
rof

Suppose we have processed $v_{i_1},..., v_{i_{j-1}}$, for some j, $1 \leq j \leq t$. Let $m = i_j$, $p = i_{j-1}$, $pp = i_{j-2}$, $n = i_{j+1}$ and $nn = i_{j+2}$ (suppose $j_0 = j_{-1} = 1$, $j_{t+1} = j_{t+2} = q$). We now discuss which information is needed from the processed part to be able to process v_m and its partial one-trees.

First consider the case that we want to combine $v_m.p[i].l$ or $v_m.p[i].lr$ for some i, $1 \leq i \leq v_m.nr$ with the previously processed part. If, for example, $v_m.p[i].l.ok$ holds, and we want to combine $v_m.p[i].l.v$ with the processed part, then we need to know whether there is a 'partial' path decomposition of the processed part of H in which the processed partial one-paths connected to P do not use any v_l, $l \geq v_m.p[1].l.v$. See e.g. Figure 46. Similarly for $v_m.p[i].lr.l_a$ for all i, $1 \leq i \leq v_m.nr$ and all a, $1 \leq a \leq 8$.

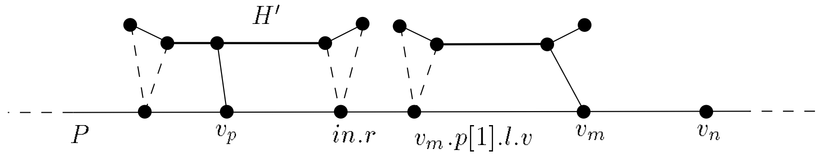

Figure 46: If $v_m.p[1].l.v$ is combined, then there must be a partial nice proper path decomposition in which partial one-paths connected to v_i, $i < m$, use $v_m.p[1].l.v$ at most. $in.ok$ is true if there is a j, $j \leq m$, and a partial nice proper path decomposition in which the partial one-paths connected to v_i, $i < m$, use j at most, and if $in.ok$ is true, then the smallest value of j for which this is possible is $in.r$.

Therefore, it suffices to know the smallest i, $p \leq i \leq m$, for which there is a partial path-decomposition of the processed part, such that v_i is the rightmost vertex of P that is used by some processed partial one-path connected to v_1,..., v_p. We keep track of this information by a variable in, which has a field ok which is true if there is such an i, false otherwise, and a field r denoting this smallest i, if $in.ok$ is true. The name in refers to the fact that the partial one-paths connected to v_j, $j < m$, only use vertices within $[1, m]$.

Next consider the case that we want to combine $v_m.p[i].lll$ or $v_m.p[i].llr$ for some i, $1 \leq i \leq v_m.nr$ with the previously processed part. Suppose for example that $v_m.p[1].llr.ok$ holds. We only have to show how to combine the values of $v_m.p[1].llr.l$ etc. with a partial path decomposition in which there is a partial one-path connected to v_p that uses vertices of v_m,..., v_n. See e.g. Figure 47. Thus, we need to know if there are partial path decompositions in which there is a partial one-path H' connected to v_p which uses vertices of v_m,..., v_n such that the sticks of v_p which have color $c(v_m)$ occur on the right side of the occurrence of $\{v_p, v_m\}$, which vertices of v_{pp},..., v_p are used by a partial one-path H'' which is connected to any v_i, $i \leq p$, and which vertices of v_m,..., v_n are used by H', since these vertices can not be used by partial one-paths connected to v_m, v_n or v_{nn}. It suffices to know all pairs (j, l), $pp \leq j \leq p$, $m \leq l \leq n$, for which the vertices of v_j,..., v_p and the vertices of v_l,..., v_n can be used for the partial one-paths of v_m, v_n and v_{nn}, and there is no such pair (j', l') for which this also holds, and either $(j' < j \wedge l' \leq l)$ or $(j' \leq j \wedge l' < l)$.

To keep track of this information, we use a variable out, which has two fields l and r, where $out.l$ is the one that can be combined with $v_m.p[1].llr$, and $out.r$ can be

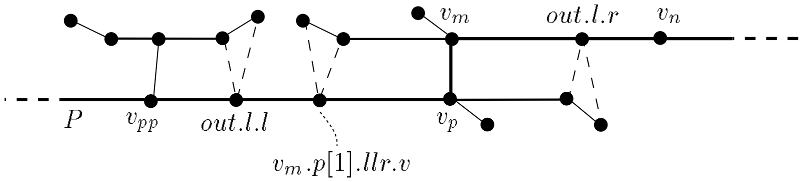

Figure 47: If $v_m.p[1].llr.v$ is combined, then there must be a partial nice proper path decomposition in which a partial one-path connected to v_p uses $[l, l']$, $m \leq l \leq l' \leq n$, such that the sticks of v_p which have color $c(v_m)$ occur on the right side of the occurrence of $\{v_p, v_m\}$. Furthermore, all other partial one-paths connected to v_i, $i < m$, use $[a, a']$ with $a' \leq v_m.p[1].llr.v$ at most. *out.l.ok* is true if there is a partial nice proper path decomposition in which a partial one-path connected to v_p uses $[l, l']$, $m \leq l \leq l' \leq n$, such that the sticks of v_p which have color v_m occur on the right side of the occurrence of $\{v_p, v_m\}$. If *out.l.ok* is true, then the pair $(out.l.l, out.l.r)$ is the lexicographically smallest pair (j, l') for which l' is as given above, and all other partial one-paths connected to some v_i, $i < m$, use $[a, a']$ with $a' \leq j$ at most.

combined with $v_m.p[1].lll$. Both *out.l* and *out.r* have three fields *ok*, *l* and *r*, which denote the following. *out.l.ok* is true if and only if there is a 'partial' nice proper path decomposition in which

- a partial one-path H' connected to v_p uses $[l', l]$, for some l and l', $m \leq l' \leq l \leq n$,
- it is possible that a partial one-path H'' which is connected to v_m uses $[j, j']$ for some j and j', $pp \leq j \leq j' \leq p$, and
- the sticks of v_p which have color $c(v_m)$ occur on the right side of the occurrence of $\{v_p, v_m\}$.

If *out.l.ok* is true, then *out.l.l* and *out.l.r* are such that $(out.l.l, out.l.r)$ is the lexicographically smallest pair (j, l), $m \leq l \leq n$ and $pp \leq j \leq p$, for which there is a 'partial' nice proper path decomposition in which a partial one-path H' connected to v_p uses $[l', l]$, $m \leq l' \leq l$, the sticks of v_p which have color $c(v_m)$ occur on the right side of the occurrence of $\{v_p, v_m\}$, and all partial one-paths connected to v_i, $i \leq p$, except H', use j at most. We show that one pair is sufficient after giving the algorithm.

The fields of *out.r* are defined in the same way, except that the sticks of v_m which have color $c(v_m)$ occur on the right side of the occurrence of $\{v_m, v_p\}$.

The name *out* refers to the fact that the rightmost partial one-path connected to v_j, $j < m$, use vertices outside of $[1, m]$.

Definition 4.11. *The global information consists of two records in and out, which are defined as follows.*

- *in is a record with two fields ok and r, which are defined as above.*

- *out is a record with two fields l and r, which each have three fields: ok, l and r, which are defined as above.*

We now show how variables *in* and *out* are initialized and adapted by giving a complete description of function Check_Nice_Path. In Figures 48, 49, 50, and 51, a symbolic representation of all cases in the algorithm is given.

function Check_Nice_Path(P: Path): boolean;
{ pre: $P = (v_1,..., v_q)$ is a nice path of H.
  $\forall_{1 \le m \le t}$ ($v_m.nr$ = # partial one-paths connected to v_m, and
    $\forall_{1 \le i \le v_m.nr}$ ($v_m.p[i].H$ is partial one-path i and $v_m.p[i].t$ is type of $v_m.p[i].H$))
}
{ output: true if there is a proper path decomposition of H
  with nice path P, false otherwise
}

  $in.ok$:= true; $in.r := 1$;
  $out.l.ok$:= false;
  $out.r.ok$:= false;
  $i_1,..., i_t$ denote vertices of P for which $v_{i_j}.nr > 0$,
    for all j, $1 \le j \le t$, such that $i_1 < i_2 < \cdots < i_t$
  $i_0, i_{-1}, i_{t+1}, i_{t+2} := 1, 1, q, q$;
  for $j := 1$ to t
  $\rightarrow$ $in' := in$; $out' := out$;
    {initialize in and out }
    $in.ok$:= false; $in.r := i_{j+1}$;
    $out.l.ok$:= false; $out.l.l, out.l.r := q, q$;
    $out.r.ok$:= false; $out.r.l, out.r.r := q, q$;
    $m := i_j$;
    $p := i_{j-1}$;
    $pp := i_{j-2}$;
    $n := i_{j+1}$;
    $nn := i_{j+2}$;

    Permute partial one-paths such that no $v_m.p[i].H$, $2 < i \le v_m.nr$,
    has a vertex of color $c(v_m)$. If this is not possible, **return** false
    for $i := 1$ to $v_m.nr$
    $\rightarrow$ **if** $v_m.p[i].H$ has vertex of color $c(v_m)$ or $v_m.nr = 1$
      $\rightarrow$ Compute $v_m.p[i].l$, $v_m.p[i].r$, $v_m.p[i].lr$, $v_m.p[i].lll$, $v_m.p[i].llr$, $v_m.p[i].rrl$,
          and $v_m.p[i].rrr$ using $PPW2$ and $PPW2'$
      $\square$ else
      $\rightarrow$ $v_m.p[i].l.ok$:= true;
        $v_m.p[i].l.v := m$;
        $v_m.p[i].r.ok$:= true;
        $v_m.p[i].r.v := m$;
        $v_m.p[i].lr.ok$:= false;
        $v_m.p[i].lll.ok$:= false;
        $v_m.p[i].llr.ok$:= false;
        $v_m.p[i].rrl.ok$:= false;
        $v_m.p[i].rrr.ok$:= false;

```
    fi
rof
{compute in }
if  v_m.nr = 1
→  {try in' }
    if  in'.ok ∧ v_m.p[1].lr.ok
    →  for a := 1 to 8
        →  if  v_m.p[1].lr.l_a ≥ in'.r
            →  {see Figure 48, part I }
                in.ok := true;
                in.r := min{in.r, v_m.p[1].lr.r_a}
            fi
        rof
    fi;

    {try out'.l }
    if  out'.l.ok
    →  if  v_m.p[1].llr.ok ∧ v_m.p[1].llr.v ≥ out'.l.l
        →  {see Figure 48, part II }
            in.ok := true;
            in.r := min{in.r, out'.l.r}
        fi
    fi;

    {try out'.r }
    if  out'.r.ok
    →  if  v_m.p[1].lll.ok ∧ v_m.p[1].lll.v ≥ out'.r.l
        →  {see Figure 48, part III }
            in.ok := true;
            in.r := min{in.r, out'.r.r}
        fi;
    fi

□  v_m.nr > 1
→  {try in' }
    if  in'.ok
    →  for i := 1 to 2
        →  if  v_m.p[i].l.ok ∧ v_m.p[3 − i].r.ok ∧ v_m.p[i].l.v ≥ in'.r
            →  {see Figure 49, part I }
                in.ok := true;
                in.r := min{in.r, v_m.p[3 − i].r.v}
            fi
        rof
    fi;

    {try out'.l }
    if  out'.l.ok ∧ out'.l.r = m
    →  for i := 1 to 2
        →  if  v_m.p[i].llr.ok ∧ v_m.p[3 − i].r.ok ∧ v_m.p[i].llr.v ≥ out'.l.l
            →  {see Figure 49, part II }
```

```
                in.ok := true;
                in.r := min{in.r, v_m.p[3 − i].r.v}
            fi
        rof
    fi;

    {try out'.r }
    if  out'.r.ok ∧ out'.r.r = m
    →  for i := 1 to 2
        →  if  v_m.p[i].lll.ok ∧ v_m.p[3 − i].r.ok ∧ v_m.p[i].lll.v ≥ out'.r.l
            →  {see Figure 49, part III }
                in.ok := true;
                in.r := min{in.r, v_m.p[3 − i].r.v}
            fi;
        rof
    fi
fi;

{compute out }
if  v_m.nr = 1
→ {try in' }
    if  in'.ok
    →  {compute out.l }
        if  v_m.p[1].rrl.ok
        →  {see Figure 50, part I }
            out.l.ok := true;
            out.l.l := in'.r;
            out.l.r := v_m.p[1].rrl.v;
        fi;
        {compute out.r }
        if  v_m.p[1].rrr.ok
        →  {see Figure 50, part II }
            out.r.ok := true;
            out.r.l := in'.r;
            out.r.r := v_m.p[1].rrr.v;
        fi;
    □  else
    →  {out' does not have to be tried since v_m.nr = 1}
        skip
    fi

□  v_m.nr > 1
→  {try in' }
    if  in'.ok
    →  for i := 1 to 2
        →  {compute out.l }
            if  v_m.p[3 − i].rrl.ok ∧ v_m.p[i].l.ok ∧ v_m.p[i].l.v ≥ in'.r
            →  {see Figure 51, part I }
                out.l.ok := true;
                out.l.l := m;
```

```
            out.l.r := min{out.l.r, v_m.p[3 − i].rrl.v}
        fi;
        {compute out.r }
        if  v_m.p[3 − i].rrr.ok ∧ v_m.p[i].l.ok ∧ v_m.p[i].l.v ≥ in'.r
        →  {see Figure 51, part II }
            out.r.ok := true;
            out.r.l := m;
            out.r.r := min{out.r.r, v_m.p[3 − i].rrr.v}
        fi;
    rof
fi;

{try out'.l }
if  out'.l.ok ∧ out'.l.r = m
→  for i := 1 to 2
    →  {compute out.l }
        if  v_m.p[3 − i].rrl.ok ∧ v_m.p[i].llr.ok ∧ v_m.p[i].llr.v ≥ out'.l.l
        →  {see Figure 51, part III }
            out.l.ok := true;
            out.l.l := m;
            out.l.r := min{out.l.r, v_m.p[3 − i].rrl.v}
        fi;
        {compute out.r }
        if  v_m.p[3 − i].rrr.ok ∧ v_m.p[i].llr.ok ∧ v_m.p[i].llr.v ≥ out'.l.l
        →  {see Figure 51, part IV }
            out.r.ok := true;
            out.r.l := m;
            out.r.r := min{out.r.r, v_m.p[3 − i].rrr.v}
        fi;
    rof
fi;

{try out'.r }
if  out'.r.ok ∧ out'.r.r = m
→  for i := 1 to 2
    →  {compute out.l }
        if  v_m.p[3 − i].rrl.ok ∧ v_m.p[i].lll.ok ∧ v_m.p[i].lll.v ≥ out'.r.l
        →  {see Figure 51, part V }
            out.l.ok := true;
            out.l.l := m;
            out.l.r := min{out.l.r, v_m.p[3 − i].rrl.v}
        fi;
        {compute out.r }
        if  v_m.p[3 − i].rrr.ok ∧ v_m.p[i].lll.ok ∧ v_m.p[i].lll.v ≥ out'.r.l
        →  {see Figure 51, part VI }
            out.r.ok := true;
            out.r.l := m;
            out.r.r := min{out.r.r, v_m.p[3 − i].rrr.v}
        fi;
    rof
```

fi
fi
rof

return *in.ok*
end

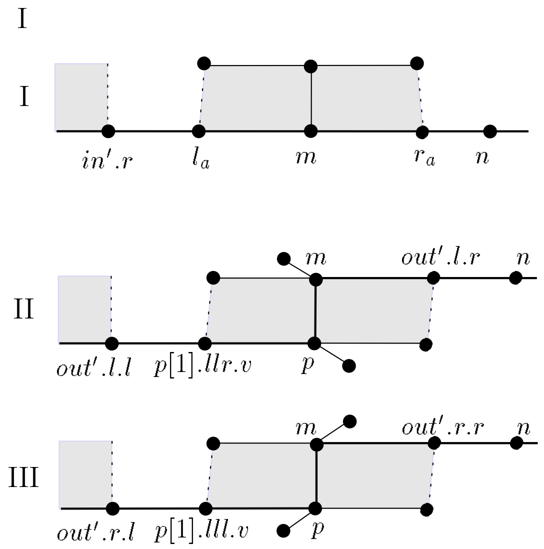

Figure 48: Cases in the algorithm in which in is computed, and $v_m.nr = 1$.

Lemma 4.19. *If suffices to keep track of only one pair* $(out.l.l, out.l.r)$*, and one pair* $(out.r.l, out.r.r)$*.*

Proof. Consider the computation of the new value of $out.l$ at vertex v_m of the path. If $out.l.ok$ holds, then want to keep track of all pairs (l_i, r_i), $p \leq l_i \leq m$ and $n \leq r_i \leq nn$ for which there is a partial proper path decomposition of the processed part in which one partial one-path H' connected to v_m uses $[l, r_i]$ for some l, $n \leq l \leq r_i$ and the sticks of v_m of color $c(v_n)$ occur on the right side of the occurrence of $\{v_m, v_n\}$, and all other partial one-paths connected to some v_i, $i \leq m$, use l_i at most, and furthermore, there is no pair (l, r), for which this also holds, $p \leq l \leq l_i$, $n \leq r \leq r_i$ and either $l < l_i$ or $r < r_i$. It seems that may be more than one pair (l_i, r_i) for which this holds. However, if $nr = 1$, there is at most one such pair possible, namely the pair $(in'.r, v_m.p[1].rrl.v)$. If $nr > 1$, then $l_i = m$ for all possible pairs, which means that there is only one such pair. Hence it suffices to keep track of only one pair (l, r) for $out.l$, and similar for $out.r$. □

The main result of this section is as follows.

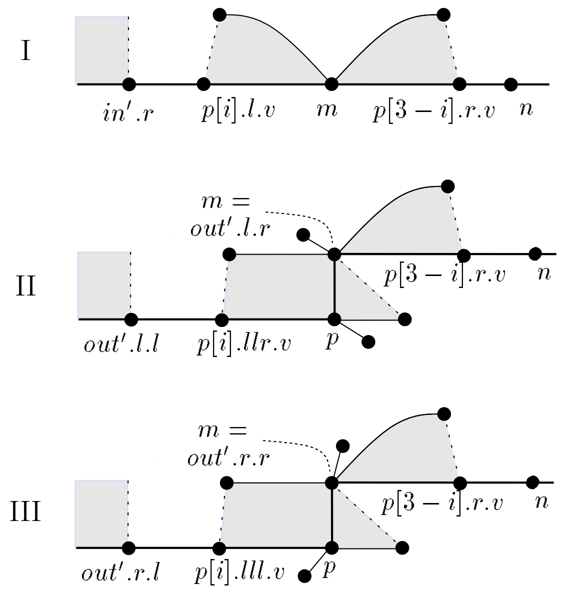

Figure 49: Cases in the algorithm in which in is computed, and $v_m.nr > 1$.

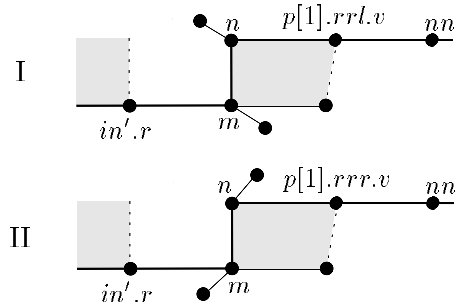

Figure 50: Cases in the algorithm in which out is computed, and $v_m.nr = 1$.

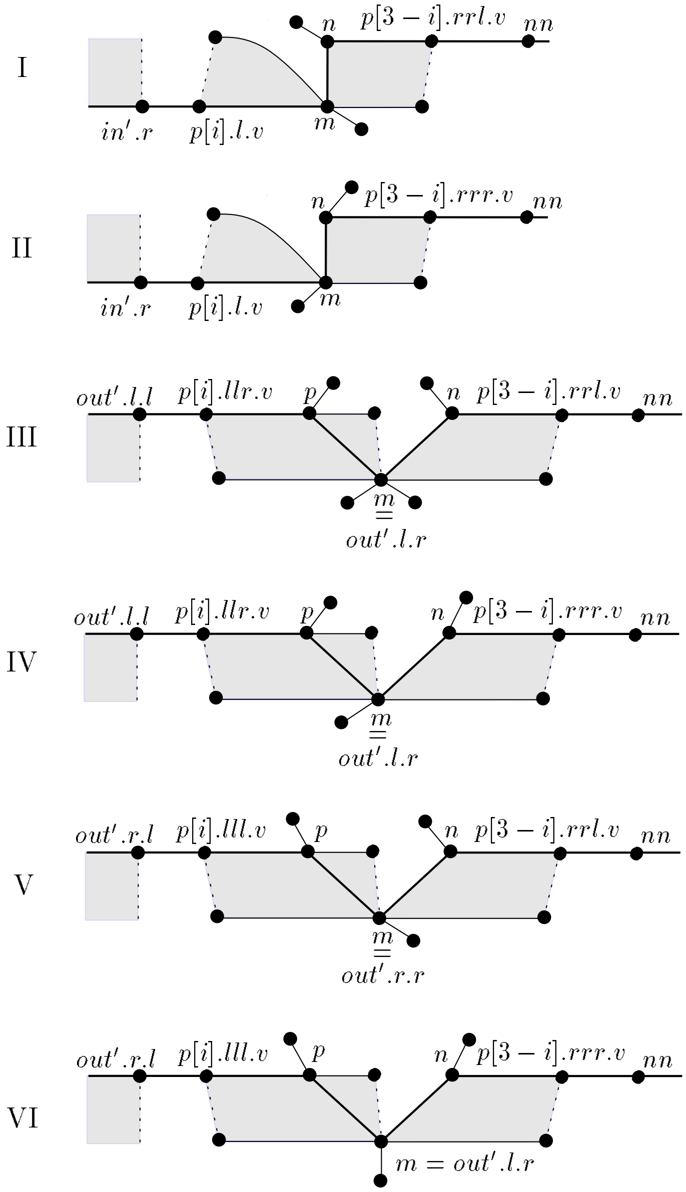

Figure 51: Cases in the algorithm in which *out* is computed, and $v_m.nr > 1$.

Theorem 4.1. *The algorithm given in this section computes in $O(n^2)$ time whether there is a proper path decomposition of a three-colored tree H ($n = |V(H)|$).*

Proof. The correctness of the algorithm follows from previous lemmas. We show that the total time taken by the algorithm is $O(n^2)$. We only have to show that for a given candidate nice path P, function Check_Nice_Path runs in $O(n^2)$ time, since the number of candidate nice paths is constant.

To show that function Check_Nice_Path runs in $O(n^2)$, we only have to show that the total time to compute all local information for all i, $1 \leq i \leq q$, is $O(n^2)$. Let $v_m \in V(P)$. For each partial one-path $v_m.p[i].H$ connected to v_m, the number of calls of $PPW2$ and $PPW2'$ is constant, since for $v_m.p[i].l$, $v_m.p[i].r$, etc., $PPW2$ and $PPW2'$ are called a constant number of times, as is shown above.

If $v_m.nr > 1$, then for all i, $1 \leq i \leq v_m.nr$, for which $v_m.p[i].H$ does not have any vertex of color $c(v_m)$, $PPW2$ and $PPW2'$ are not called at all, since $v_m.p[i].l.ok$ and $v_m.p[i].r.ok$ are both true, $v_m.p[i].l.v = v_m.p[i].r.v = m$, and all other ok-fields are false. This means that each $v \in V(P)$ is involved in the computation of local information of a constant number of partial one-paths connected to P, and hence v is involved in a constant number of calls of $PPW2$ and $PPW2'$. Hence each vertex $v \in V(H)$ is involved in a constant number of calls of $PPW2$ and $PPW2'$.

Since $PPW2$ and $PPW2'$ run in quadratic time, it follows that the computation of local information takes $O(n^2)$ time. □

This completes the description of the algorithm to check for a given properly three-colored tree H of pathwidth two, whether there is a proper path decomposition of H. The algorithm can be made constructive in the sense that it returns an intervalization if there exists one as follows. For each vertex v_m of a nice path P, for each i, $1 \leq i \leq v_m.p[i].nr$, if $v_m.p[i].l.ok$ is true, keep a pointer to a list of edges that is present in an intervalization corresponding to a partial path decomposition for this value of $v_m.p[i].l.v$. Such a list can be made during the computation of $PPW2$ or $PPW2'$, as is shown in Section 4.2, Do the same for $v_m.p[i].r$, etc. Furthermore, in the main loop of Check_Nice_Path, keep a pointer to a list of edges that is present in a partial intervalization of the processed part of H for variables in, $out.l$ and $out.r$, which correspond to the values found for these variables. The adaptation of these lists of edges is done by adding the lists of edges pointed to by the variables that are combined.

4.4 General Graphs

In this section we give an algorithm to determine for a given three-colored partial two-path G whether there is a proper path decomposition of G. This algorithm is an extension of the algorithm for trees of pathwidth two. Therefore, we first extend the notion of nice paths. After that, we show what extra local and global information has to be computed, and how this extra information can be computed.

Definition 4.12. (Nice Path Decomposition). *Let G be a connected three-colored graph with pathwidth two, G not a tree, let $P_G = (v_1,..., v_s)$, let $PD = (V_1,..., V_t)$ be a* *proper path decomposition of G. Then PD is a* nice path decomposition *of G if there are no two consecutive nodes which are equal, V_1 contains an edge $e = \{v, v'\} \in E$ and V_t contains an edge $e' = \{x, x'\} \in E$, in such a way that $x \neq v$ and the path from v to x contains P_G. Furthermore, one of the following condition holds for V_1 and e, and analogously for V_t and e'.*

1. *$s = 0$, B is the only biconnected component of G, $e \in E(H')$ for some component H' of G_{T} containing a vertex $w \in V(B)$ of state* E1 *or* I1, *such that v is an end point of the path P' containing $P_1(H')$ and w, and $v \neq w$.*
2. *$s = 0$, B is the only biconnected component of G, $e \in E(G)$, $v \in V(B)$ and either v' is a stick adjacent to v, or $v' \in V(B)$.*
3. *$s \geq 1$, $e \in E(H')$ for some partial one-path H' connected to v_1 such that v is an end point of some path $P' \in \mathcal{P}_1(H')$,*
4. *$s \geq 1$, $e \in E(H')$ for some component H' of G_{T} containing a vertex w of state* E1 *or* I1 *of a biconnected component containing v_1, such that v is an end point of the path P' containing $P_1(H')$ and w, and $v \neq w$.*
5. *$s \geq 1$, there is a biconnected component B containing v_1 such that $v \in V(B) - \{v_1\}$, and either $\{v, v'\} \in E(B)$ or v' is a stick adjacent to v.*

The nice path *P' corresponding to nice path decomposition PD is defined as follows. If $s = 0$, then P' is the empty path if condition 2 holds for both V_1 and V_t. If condition 1 holds for V_1, and 2 for V_t, then P' is the path from v to the vertex $w \in V(B)$ for which v and w are in the same component of G_{T}. Analogously, if condition 1 holds for V_t and 2 holds for V_1, then P' is the path from the vertex $w \in V(B)$ to x, such that w and x are in the same component of G_{T}. If condition 1 holds for both V_1 and V_t, then P' is the largest common subsequence of all paths from v to x. If $s \geq 1$, then P' is the largest common subsequence of all paths from w to w' in G, where $w = v_1$ if condition 5 holds for V_1, $w = v$ otherwise, and $w' = v_s$ if condition 5 holds for V_t, $w = x$ otherwise.*

Figure 52 shows an example of all conditions in Definition 4.12.

Note that each nice path contains P_G. If there is a nice proper path decomposition of G for which condition 5 of Definition 4.12 holds for v_1 or v_s, then B is called the *ending biconnected component.*

We now show that there is a nice proper path decomposition of G if and only if there is a proper path decomposition of G.

Lemma 4.20. *Let G be a connected three-colored graph with pathwidth two. There is a proper path decomposition of G if and only if there is a nice proper path decomposition of G.*

Proof. The 'if' part is clearly true.

The proof of the 'only if' part is similar to the proof of Lemma 4.9. If G is a tree, then it clearly holds, because of Lemma 4.9. Suppose G is not a tree, $s \geq 1$. Suppose

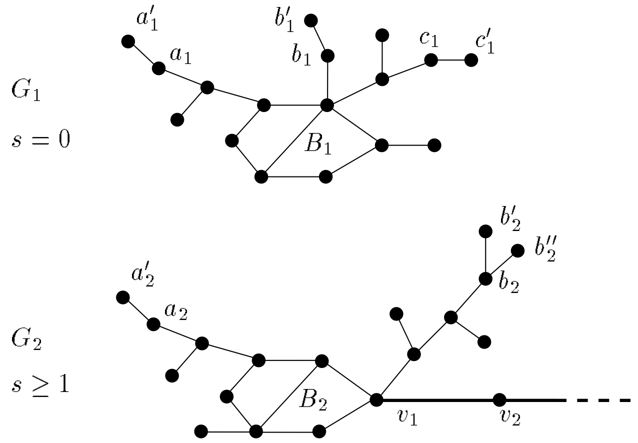

Figure 52: Examples of possible values of v and v' as defined in Definition 4.12. In G_1, $s = 0$, and in G_2, $s \geq 1$. If v and v' are equal to a_1 and a_1', b_1 and b_1' or c_1 and c_1', then case 1 holds. If $v \in V(B_1)$, and v' is either a stick adjacent to v, or $\{v, v'\} \in E(B)$, then case 2 holds. If $v = b_2$, and v' is equal to b_2' or b_2'', then case 3 holds. If $v = a_2$ and $v' = a_2'$, then case 4 holds, and if $v \in V(B_2)$, and v' is either a stick adjacent to v, or $\{v, v'\} \in E(B_2)$, then case 5 holds.

$PD = (V_1,..., V_t)$ is a proper path decomposition of G. Let G_1 be the subgraph of G induced by v_1 and the components of $G[V - \{v_1\}]$ of which V_1 contains at least one vertex (note that G_1 contains no vertices of the component of $G[V - \{v_1\}]$ which contains v_s, because of Lemma 3.20). Similarly, let G_2 be the subgraph of G induced by v_s and the components of $G[V - \{v_s\}]$ of which V_t contains at least one vertex. Note that, if $s = 1$ then $V(G_1) \cap V(G_2) = \{v_1\}$).

We now show how PD can be transformed into a nice proper path decomposition of G by 'unfolding' PD until it satisfies the described condition. The following cases may occur for V_1.

1. $V_1 = \{v, v'\}$ for some edge $\{v, v'\} \in E(G_1)$ such that v' has degree one and $G[V - \{v\}]$ has exactly one component which contains two or more vertices.
2. $V_1 = \{v, v'\}$ for some edge $\{v, v'\} \in E(G_1)$, such that there is a biconnected component B in G_1 for which $v, v' \in V(B)$ and $st(v) \in \{\mathrm{N}, \mathrm{S}\}$, and either $v' = v_1$ or $st(v') \in \{\mathrm{N}, \mathrm{S}\}$.
3. V_1 contains no edge.
4. $|V_1| = 3$, but contains an edge.
5. $V_1 = \{v, v'\}$ for some edge $\{v, v'\} \in E(G_1)$, and $G[V - \{v\}]$ has two or more components which contain at least one edge, but 2 does not hold.

For V_t, the possible cases are similar.

If case 1 holds for V_1, then there are three possibilities:

1. $\{v, v'\} \in E(H')$ for some partial one-path H' connected to v_1 such that v is an end point of some path $P' \in \mathcal{P}_1(H')$,

2. $\{v, v'\} \in E(H')$ for some component H' of G_{T} containing a vertex w of state E1 or I1 of a biconnected component containing v_1, such that v is an end point of the path P' containing a path of $\mathcal{P}_1(H')$ and w such that $v \neq w$ if $|V(P')| > 1$, or

3. $v \in V(B)$ for some biconnected component B which contains v_1, $st(v) = \mathrm{S}$, and v' is a stick adjacent to v.

(See also the proof of Lemma 4.9.)

Hence if case 1 or case 2 holds, then we are ready. Now, we apply the following transformations on PD such that one of the previous cases holds again after each transformation, until case 1 or case 2 holds for V_1, and case 1 or case 2 holds for V_t. First transform PD using the following rules until case 1 or case 2 applies for V_1, next transform PD using the following rules, adapted for V_t, until case 1 or case 2 applies for V_t. During the transformations, G_1 and G_2 are changed, in order to show that the number of transformations is finite.

If case 3 holds, delete V_1. Note that still $V_1 \subseteq V(G_1)$.

If case 4 holds, let $e \in E(G_1)$ such that $e \subseteq V_1$, and add a node containing e only on the left side of V_1.

If case 5 holds, do the following. Consider the components of $G[V - \{v\}]$ which consist of more than one vertex. Note that at least one of these components is a subgraph of G_1 which does not contain v_1, and hence V_t does not contain any vertex of this component. If $G[V - \{v'\}]$ does not have two or more components which contain two or more vertices, then v' has degree one, otherwise case 2 would hold. This means that in this case, there is a component of $G[V - \{v\}]$ which has two or more vertices, does not contain vertices of P_G, and does not contain v'. In this case, let G' be such a component. Note that G' is a subgraph of G_1. If $G[V - \{v\}]$ and $G[V - \{v'\}]$ both have two or more components which have two or more vertices, then either $G[V - \{v\}]$ has a component which contains v' and vertices of P_G, or $G[V - \{v'\}]$ has a component which contains v and vertices of P_G. Suppose w.l.o.g. that the first one holds. In this case, let G' be a component of $G[V - \{v\}]$ which has at least two vertices, and which does not contain v'. Note again that G' is a subgraph of G_1. Let G_1' be the subgraph of G induced by $V(G')$ and v, and note that G_1' is a proper subgraph of G_1, and it contains at least one edge. Now transform PD into $\mathrm{rev}(PD[V(G_1')] + PD[V - V(G')]$, and let G_1 be equal to G_1'. The new path decomposition is indeed a proper path decomposition of G, since v is the only vertex that $H[V(G_1)]$ and $H[V - V(G')]$ have in common, and v occurs in the rightmost node of $\mathrm{rev}(PD[V(G_1)])$ and in the leftmost node of $PD[V - V(G')]$. Furthermore, the leftmost node of the new PD contains only vertices of G_1 and the rightmost node of the new PD contains only vertices of G_2.

The total number of transformations that is done this way is finite, because of the following. The transformation of case 5 can only be done for a finite number of times, since each time this transformation is done, the size of G_1 or G_2 decreases. The transformation of case 4 can only be done once per transformation of case 5. The transformation of case 3 can only be done a finite number of times for each transformation of case 5, since the length of the path decomposition remains finite.

This completes the proof for the case that $s \geq 1$. If $s = 0$, then the proof is similar, so it is omitted here. $\square$

Next we have to show that the number of nice paths that has to be tried is constant. This is done in the same way as for trees of pathwidth two. Let G be a three-colored partial two-path, $P = (v_1,..., v_q)$ a nice path of G. The analog of Lemma 4.12 holds for three-colored partial two-paths, i.e. if there is a proper path decomposition of G, then there is a proper path decomposition PD of G in which for each $v \in V$, if $G[V - \{v\}]$ has four or more components, then there is a node $\{v\}$ in PD. We can now state which nice paths have to be tried and which do not have to be tried.

Lemma 4.21. *Let G be a connected three-colored graph with pathwidth two which is not a tree. Let $P_G = (v_1,..., v_s)$, suppose $q > 1$ and suppose there is a proper path decomposition of G. Let PD be a nice proper path decomposition of G with nice path $P = (u_1,..., u_q)$. One of the following conditions holds.*

1. *There is a non-connecting biconnected component B which contains v_1 and for which $\mathrm{cond}_1(st(B))$ does not hold. Then one of the following conditions holds.*
   (a) *There is a component H' of G_{T} which contains a vertex w of B, $st(w) \in \{\mathrm{I1}, \mathrm{E1}\}$, and u_1 is an end point of some path in $\mathcal{P}_1(H')$.*
   (b) *$u_1 = v_1$ and B is ending biconnected component.*
2. *For each non-connecting biconnected component B which contains v_1, $\mathrm{cond}_1(st(B))$ holds, and furthermore $G[V - \{v_1\}]$ has three or less components. Then one of the following conditions holds.*
   (a) *There is a partial one-path H' connected to v_1, and u_1 is an end point of some path in $\mathcal{P}_1(H')$.*
   (b) *There is a non-connecting biconnected component B which contains v_1, and either B is ending biconnected component and $u_1 = v_1$, or there is a component H' of G_{T} which contains a vertex w of B with $st(w) \in \{\mathrm{I1}, \mathrm{E1}\}$, and u_1 is an end point of some path in $\mathcal{P}_1(H')$.*
3. *For each non-connecting biconnected component B which contains v_1, $\mathrm{cond}_1(st(B))$ holds, and furthermore $G[V - \{v_1\}]$ has four or more components, and there is a component H' of $G[V - \{v_1\}]$, $v_s \notin V(H')$, which has a vertex of color $c(v_1)$. Then one of the following conditions holds.*
   (a) *There is a partial one-path H' connected to v_1 which has a vertex of color $c(v_1)$, and there is a nice proper path decomposition PD' with nice path $P' = (w_1,..., w_r)$, such that $w_r = u_q$ and w_1 is an end point of some path in $\mathcal{P}_1(H')$.*

(b) *There is a non-connecting biconnected component B which contains v_1, such that the component of $G[V-\{v_1\}]$ which contains $V(B)-\{v_1\}$ has a vertex of color $c(v_1)$, there is nice proper path decomposition PD' with nice path $P'=(w_1,\ldots,w_r)$ such that $w_r=u_q$ and either $w_1=v_1$ and B is ending biconnected component, or there is a component H' of G_{T} which contains a vertex w of B with $st(w)\in\{\mathrm{I1},\mathrm{E1}\}$, and w_1 is an end point of some path in $\mathcal{P}_1(H')$.*

4. *For each non-connecting biconnected component B which contains v_1, $\mathrm{cond}_1(st(V))$ holds, and furthermore $G[V-\{v_1\}]$ has four or more components, and there is no component H' of $G[V-\{v_1\}]$, $v_s\notin V(H')$, which has a vertex of color $c(v_1)$. Then both of the following conditions holds.*

   (a) *For all partial one-paths H' connected to v_1, there is a nice proper path decomposition PD' with nice path $P'=(w_1,\ldots,w_r)$, such that $w_r=u_q$ and w_1 is an end point of some path in $\mathcal{P}_1(H')$.*

   (b) *For all non-connecting biconnected components B which contain v_1, there is nice proper path decomposition PD' with nice path $P'=(w_1,\ldots,w_r)$ such that $w_r=u_q$ and either B is ending biconnected component and $w_1=v_1$, or there is a component H' of G_{T} which contains a vertex w of B with $st(w)\in\{\mathrm{I1},\mathrm{E1}\}$, and w_1 is an end point of some path in $\mathcal{P}_1(H')$.*

Proof. If there is a non-connecting biconnected component B which contains v_1 such that $\mathrm{cond}_1(st(B))$ does not hold, then the component G' of $G[V-\{v_1\}]$ which contains $V(B)-\{v_1\}$ has pathwidth two, which means that in each path decomposition of G, V_1 contains only vertices of the G'. Hence case 1 holds.

If for each non-connecting biconnected component B which contains v_1, $\mathrm{cond}_1(st(B))$ holds, then each component of $G[V-\{v_1\}]$ which does not contain v_s has pathwidth at most one. Hence cases 2, 3 and 4 can be proved in the same way as Lemma 4.13. □

If $|V(P_G)|=1$, then a similar lemma holds, which is omitted here, since the number of cases is large (but constant).

If $|V(P_G)|>1$, then there are at most three components of $G[V-\{v_1\}]$ which do not contain a vertex of P_G, which have a vertex of color $c(v_1)$. The partial one-paths connected to v_1 which have a vertex of color $c(v_1)$ each give two end points to try. The biconnected components containing v_1 each give at most three end points to try, since they have at most three vertices of state E1, or at most one vertex of state I1 and at most one vertex of state E1. Hence there are at most nine end points to try, together with end point v_1, this gives at most ten end points to try on one side, and at most ten on the other side, which gives at most 100 nice paths in total. If $|V(P_G)|=1$, a similar calculation can be made.

This shows that the number of nice paths that has to be tried is constant, since if $|V(P_G)|=0$, then the number of vertices with state I1 or E1 is at most four, which means that the number of choices for end points of possible nice paths is bounded.

It follows directly from Lemma 4.21 how the possible nice paths can be selected.

Next we concentrate on the computation of local information for each vertex of a nice path of G. Let $P_G = (u_1,..., u_s)$, and let $P = (v_1,..., v_q)$ be a possible nice path of G.

Let $v_m \in V(P)$, $1 < m < q$, suppose there there is a non-connecting biconnected component B which contains v_m. The component H' of $G[V - \{v_m\}]$ which contains $V(B) - \{v_m\}$ has pathwidth one (Lemma 3.14), hence it can be handled in the same way as other partial one-paths connected to P. Therefore, we extend the types of partial one-paths as follows.

Definition 4.13. (Types of Partial One-Paths). *Let G be a tree of pathwidth two, P a path in G. Let $v \in V(P)$, and H' a component of $H[V - V(P)]$ such that H' has pathwidth one and H' has only vertices which are adjacent to v, i.e. H' is* connected *to v. Let $W \subseteq V(H')$ be the set of vertices for which $\{v, w\} \in E(H)$. Let $P' \in \mathcal{P}_1(H')$. If $|W| = 1$, then the type of H' is as defined in Definition 4.2. If $|W| > 1$, then H' has type* IV.

From now on, by partial one-paths connected to a path P, we do not only mean the partial one-paths of type I, II and III connected to P, but also the partial one-paths of type IV connected to P, unless stated otherwise.

In the same way as for trees of pathwidth two (Corollary 4.1 and Lemma 4.15), we can show that if there is a proper path decomposition of G with nice path P, then there is a proper path decomposition of G in which for each m, $1 \leq m \leq q$, for which $G[V - \{v_m\}]$ has four or more components which contain at least one edge, all components of $G[V - \{v_m\}]$ which do not have a vertex of color $c(v_m)$ and which have pathwidth one, occur within the occurrence of v_m, and furthermore for each two components H' and H'' of $G[V - V(P)]$ which have pathwidth one, such that $H' \neq H''$, PD contains no node which contains a vertex of H' and a vertex of H''. Hence the notion of *use* can also be used for partial one-paths of type IV.

Definition 4.14. *Let G be a three-colored partial two-path, $P = (v_1,..., v_q)$ a possible nice path for G. Let B be a biconnected component of G. If $V(B) \cap V(P) = \{v_m\}$, then G_B is the subgraph of G induced by v_m and the vertices of the component of $G[V - \{v_m\}]$ which contains $V(B) - \{v_m\}$. If $V(B) \cap V(P) = \{v_m, v_{m+1}\}$, then G_B is the subgraph of G induced by v_m, v_{m+1}, and the vertices of the component of $G[V - \{v_m, v_{m+1}\}]$ which contains $V(B) - \{v_m, v_{m+1}\}$.*

Now consider the local information for connecting biconnected components.

Lemma 4.22. *Let G be a three-colored partial two-path, suppose there is a nice proper path decomposition of G with nice path $P = (v_1,..., v_q)$. Let B be a connecting biconnected component which connects v_m and v_{m+1}, $1 \leq m < q$. Suppose G_B occurs in $(V_j,..., V_{j'})$. Let v_l be the leftmost vertex on P which occurs in $(V_j,..., V_{j'})$, and $v_{l'}$ the rightmost vertex on P which occurs in $(V_j,..., V_{j'})$. Then $v_l \in V_j$, $v_{l'} \in V_{j'}$, $l \leq m < l'$, and for all i, $l < i < m$ or $m + 1 < i < l'$, v_i and all sticks adjacent to v_i occur* *within $(V_j,\ldots,V_{j'})$, and there is no partial one-path connected to v_i, or a connecting biconnected component $B' \neq B$ containing v_i, and there is no partial one-path which uses $[a,a']$, with $l \leq a \leq l'$ or $l \leq a' \leq l'$.*

Proof. Node V_j contains a vertex of the path from v_1 to v_m. But V_j does not contain any vertex v_i with $1 \leq i < l$, hence $v_l \in V_j$, and $l \leq m$. Similarly, $v_{l'} \in V_{j'}$ and $l' \geq m+1$. Furthermore, V_j and $V_{j'}$ contain an edge of G_B, which means that V_j and $V_{j'}$ contain no vertices of $\{v_{l+1},\ldots,v_{m-1},v_{m+2},\ldots,v_{l'-1}\}$, or any other vertices which are not in $V(G_B)\cup\{v_l,v_{l'}\}$. Hence all sticks, partial one-paths, and connecting biconnected components which are connected to some v_i, $l < i < m$ or $m+1 < i < l'$, occur with $(V_j,\ldots,V_{j'})$. Suppose B occurs in $(V_s,\ldots,V_{s'})$, $j \leq s \leq s' \leq j'$. For each a, $s \leq a \leq s'$, each node V_a contains two vertices of B. For each b, $j \leq b < s$, or $s' < b \leq j'$, V_i contains a vertex of P and a vertex of $V(G_B)-\{v_m,v_{m+1}\}$. Hence it is not possible that there is a partial one-path or a connecting biconnected component which is connected to any v_i, $l < i < m$ or $m+1 < i < l'$, or a partial one-path which uses $[a,a']$, for some $l \leq a \leq l'$ or $l \leq a' \leq l'$. □

Lemma 4.22 implies the following corollary.

Corollary 4.4. *Let G be a three-colored partial two-path, suppose there is a nice proper path decomposition of G with nice path $P=(v_1,\ldots,v_q)$. Let B be a connecting biconnected component which connects v_m and v_{m+1}, $1 \leq m < q$. Let v_l be the leftmost vertex on P which occurs within the occurrence of G_B and let $v_{l'}$ be the rightmost vertex on P which occurs within the occurrence of G_B. Let H' be a partial one-path which is connected to $v_{m'}$, $1 \leq m' \leq q$. If $m' < m$, then H' can at most use l, if $m' > m+1$, then H' can at least use l', and if $m'=m$ or $m'=m+1$ then H' can use at most l, or at least l'.*

Let G be a three-colored graph with pathwidth two, suppose there is a nice proper path decomposition of G with nice path $P=(v_1,\ldots,v_q)$, and there is a connecting biconnected component B between v_m and v_{m+1}, $1 \leq m < q$. Partial one-paths which are connected to v_m or v_{m+1} can both occur on the left side and on the right side of the occurrence of G_B. The following Lemma gives conditions for the case that a partial one-path connected to v_m occurs on the right side of the occurrence of G_B.

Lemma 4.23. *Let G be a three-colored partial two-path, suppose there is a nice proper path decomposition of G with nice path $P=(v_1,\ldots,v_q)$. Let B be a connecting biconnected component which connects v_m and v_{m+1}, $1 \leq m < q$. Suppose G_B occurs in $(V_j,\ldots,V_{j'})$. Let H' be a partial one-path which is connected to v_m, suppose H' uses $[l,l']$, $l \geq m+1$. Then $v_m,v_{m+1} \in V_{j'}$, H' has type* I, *there is no other partial one-path connected to v_m which uses $[n,n']$, $m+1 \leq n$, and there is no partial one-path which uses $[n,n']$, $m+1 \leq n \leq n' \leq l$.*

Proof. Suppose B occurs in $(V_a,\ldots,V_{a'})$. Clearly, $j \leq a \leq a' \leq j'$ and $v_m,v_{m+1} \in V_{a'}$. Suppose $v_m \notin V_{j'}$. Then $V_{j'}$ contains a vertex of the path $(v_{m+1},\ldots,v_q)$, a vertex of H', and an edge of G_B, which does not contain v_{m+1} because G_B contains only edges incident with v_{m+1} which are in B. But then $|V_{j'}| \geq 4$, hence $v_m \in V_{j'}$. Suppose $v_{m+1} \notin V_{j'}$. Then $V_{j'}$ contains v_m, a vertex of $v_{m+2},..., v_q$, and an edge of G_B, which cannot contain v_m. But then $|V_{j'}| \geq 4$, hence $v_{m+1} \in V_{j'}$.

We now show that H' has type I. Suppose H' occurs in $(V_b,..., V_{b'})$. Then $v_m \in V_b$ and each node V_i, $b \leq i \leq b'$, contains a vertex of $v_{m+1},..., v_q$, hence only an end point of a path in $\mathcal{P}_1(H')$, or a stick adjacent to such an end point can be adjacent to v_m.

Each node V_i, $j' \leq i \leq b$, contains v_m and a vertex of the path $v_{m+1},..., v_q$, which means that there can be no partial one-path which uses $[n, n']$, $m + 1 \leq n \leq n' \leq l$. Furthermore, there can be no partial one-path connected to v_m which uses $[n, n']$, $n \geq l'$, since it is not possible that $v_m \in V_{b'}$. □

The following Lemma gives conditions for the case that a partial one-path connected to v_m occurs on the left side of the occurrence of G_B.

Lemma 4.24. *Let G be a three-colored partial two-path, suppose there is a nice proper path decomposition of G with nice path $P = (v_1,..., v_q)$. Let B be a connecting biconnected component which connects v_m and v_{m+1}, $1 \leq m < q$. Suppose G_B occurs in $(V_j,..., V_{j'})$. Let H' be a partial one-path which is connected to v_m, suppose H' uses $[l, l']$, $l \leq m$. Then $v_m \in V_j$.*

Proof. Suppose $v_m \notin V_j$. Then V_j contains a vertex of the path $v_1,..., v_{m-1}$, a vertex of H', and an edge of G_B. This means that $|V_j| \geq 4$, hence $v_m \in V_j$. □

Consider the local information for ending biconnected components.

We now prove the analog of Lemma 4.22 for ending biconnected components.

Lemma 4.25. *Let G be a three-colored partial two-path, suppose there is a nice proper path decomposition of G with nice path $P = (v_1,..., v_q)$, such that $v_1 \in V(B)$, B is ending biconnected component. Suppose G_B occurs in $(V_1,..., V_j)$. Let v_l be the rightmost vertex on P which occurs in $(V_1,..., V_j)$. Then $v_l \in V_j$, and for all i, $1 < i < l$, v_i and all sticks adjacent to v_i occur within $(V_1,..., V_j)$, and there is no partial one-path connected to v_i, or a connecting biconnected component $B' \neq B$ containing v_i. Furthermore, there is no partial one-path connected to v_1, or $v_1 \in V_j$, or a partial one-path connected to v_1 occurs on the left side of the occurrence of B.*

Proof. Node V_j contains a vertex of the path from v_1 to v_q, but it does not contain any vertex v_i with $i > l$, hence $v_l \in V_j$. Furthermore, V_j contains an edge of G_B, which means that V_j contains no vertices of $\{v_2,..., v_{l-1}\}$, or any other vertices which are not in G_B or $\{v_l\}$. Hence all sticks, partial one-paths, and connecting biconnected components which are connected to some v_i, $1 < i < l$, occur within $(V_1,..., V_j)$. Suppose B occurs in $(V_s,..., V_{s'})$, $1 \leq s \leq s' \leq j$. For each a, $s \leq a \leq s'$, each node V_a contains two vertices of B. For each a, $s' < a \leq j'$, V_a contains a vertex of P and a vertex of $V(G_B) - \{v_1\}$. Furthermore, partial one-paths connected to v_i, $1 < i < l$, can not occur on the left side of the occurrence of V_s. Hence it is not possible that there is a partial one-path or a connecting biconnected component which is connected to any v_i, $1 < i < l$. Furthermore, a partial one-path connected to v_1 either occurs on the left side of V_s or on the right side of V_j. If it occurs on the right side of V_j, then $v_1 \in V_j$, since V_j can not contain any vertex which is not in G_B or $\{v_l\}$. $\square$

Lemma 4.25 implies the following corollary.

Corollary 4.5. *Let G be a three-colored partial two-path, suppose there is a nice proper path decomposition of G with nice path $P = (v_1, \ldots, v_q)$, such that $v_1 \in V(B)$, B is ending biconnected component. Let v_l be the rightmost vertex on P which occurs within the occurrence of G_B. Let H' be a partial one-path which is connected to v_n, $1 \leq n \leq q$. If $n > 1$, then H' can at least use v_l. If $n = 1$, then H' occurs on the left side of the occurrence of B, or H' can at least use v_l.*

In the following lemma, the number of possibilities for ending biconnected components are bounded.

Lemma 4.26. *Let G be a three-colored partial two-path, suppose there is a nice proper path decomposition $PD = (V_1, \ldots, V_t)$ of G with nice path $P = (v_1, \ldots, v_q)$ with ending biconnected component B. If there is an edge e with $e \cap V(B) = \emptyset$, and e occurs on the left side of the occurrence of B, then there is a nice proper path decomposition of G with nice path $P' = (w_1, \ldots, w_r, v_1, \ldots, v_q)$, where $w_r \in V(B)$, and either $\{w_r, v_1\} \in E(G)$ and there is a partial one-path H' connected to v_1 such that $w_i \in V(H')$ for all i, $1 \leq i \leq r$, and w_1 is end point of a path in $\mathcal{P}_1(H')$, or $w_r \in V(B)$, $st(w_r) \in \{\mathrm{E1}, \mathrm{I1}\}$, and w_1 is end point of the path $P_1(H')$, where H' is the component of G_{T} which contains w_r.*

Proof. Suppose B occurs in $(V_j, \ldots, V_{j'})$. Note that V_1 does not contain an edge of B, since then each V_i, $i < j$, contains two vertices of B and can not contain e. Let $x \in V(B)$, and x' a stick adjacent to x such that $x, x' \in V_1$. Note that $x \in V_j$.

For all i, $1 \leq i < j$, V_i can not contain vertices from the component of $G[V - \{v_1\}]$ which contains v_2, if $q > 1$. Furthermore, V_i contains a vertex of B, or a stick of a vertex of B, which means that there is no biconnected component $B' \neq B$ which occurs on the left side of V_j. Hence there is a partial one-path H' with $e \subseteq E(H')$, and either H' is connected to v_1, or there is $w \in V(B)$ with $st(w) \in \{\mathrm{I1}, \mathrm{E1}\}$, and H' is the component of G_{T} which contains w. If H' is connected to v_1, let $w = v_1$. Let H_e be the component of $G[V - \{w\}]$ which contains e. Let H'_e be $G[V(H_e) \cup W]$, where W contains w and all sticks of w which occur on the left side of V_j. Note that H'_e occurs completely on the left side of V_j, and $w \in V_j$. Furthermore, note that w is an end point of $P_1(H'_e)$ or a stick adjacent to this end point, since each V_i, $1 \leq i < j$, contains x.

Suppose H'_e occurs in $(V_l, \ldots, V_{l'})$, $1 \leq l \leq l' < j$. There are no edges e' with $e' \notin E(H'_e)$, and e' occurs on the left side of V_j, except edges $\{x, x''\}$, where x'' is a stick of x, since each V_i, $l \leq i < j$, contains x and at least one vertex of H'_e, and there is at least one node V_i, $l \leq i < j$, which contains x and two vertices of H'_e. No vertex of H'_e has color $c(x)$. Let $(w_1, \ldots, w_r)$ be the shortest path in H'_e which contains $P_1(H'_e)$ and w, such that $w = w_r$.

We now transform PD into a nice proper path decomposition of G with nice path $(w_1, \ldots, w_{r-1}, v_1, \ldots, v_q)$ if $w = v_1$, and nice path $(w_1, \ldots, w_r, v_1, \ldots, v_q)$ otherwise. Let $PD_1 = (W_1, \ldots, W_b)$ be a proper path decomposition of width two of H'_e with w_1 and a stick of w_1 in the leftmost node, and w_r in the rightmost node, such that there is a node W_a which contains w_{r-1} only. Note that $a < b$. For each i, $a \leq i \leq b$ add vertex x to W_i. Note that $c(w_{r-1}) \neq c(w_r)$ so all sticks of x have color $c(w_{r-1})$ or color $c(w_r)$. For each stick x'' of x with $c(x'') = c(w_r)$, add a node $\{x, x'', w_{r-1}\}$ between V_a and V_{a+1}. For each stick x'' of x with $c(x'') = c(w_{r-1})$, add a node $\{x, x'', w_r\}$ on the right side of W_b. Let PD_1 again denote this proper path decomposition. Let PD_2 denote that path decomposition obtained from PD by deleting $(V_1,..., V_{j-1})$. Then $PD' = PD_1 \mathbin{+\!\!+} PD_2$ is the desired nice proper path decomposition of G. □

The following lemma gives the analog of Lemma 4.26 for the case that the nice path is empty. The proof of this lemma is the same as for Lemma 4.26.

Lemma 4.27. *Let G be a three-colored partial two-path, suppose there is a nice proper path decomposition $PD = (V_1,..., V_t)$ of G with nice path $P = ()$. Let B be the biconnected component of G. If there is an edge e with $e \cap V(B) = \emptyset$, and e occurs on the left side of the occurrence of B, then there is a nice proper path decomposition of G with nice path $P' = (w_1,..., w_r)$, where $w_r \in V(B)$, $st(w_r) \in \{\mathrm{E1}, \mathrm{I1}\}$, and w_1 is end point of a path $\mathcal{P}_1(H')$, where H' is the component of G_{T} which contains w_r.*

The analogs of Lemma 4.26 and Lemma 4.27 also hold for the right side of the path decomposition. Hence Lemma 4.27 implies that an empty nice path has to be tried only if G is a biconnected component with sticks.

We show what local information is computed, and how it is computed for all vertices of the nice path $P = (v_1,..., v_q)$ to which a partial one-path is connected, or which contains a biconnected component. We distinguish four different kinds of vertices of P. Suppose $q \geq 1$, let $1 \leq m \leq q$, such that there is a partial one-path connected to v_m or there is a biconnected component which contains v_m. The following cases are distinguished for v_m.

1. $1 < m < q$, and there is no connecting biconnected component between v_{m-1} and v_m or between v_m and v_{m+1},
2. $m \in \{1, q\}$, and there is no connecting biconnected component between v_m and v_{m+1}, or between v_{m-1} and v_m,
3. $1 < m < q$, and there is a connecting biconnected component between v_{m-1} and v_m, or between v_m and v_{m+1}, and
4. $m \in \{1, q\}$, and there is connecting biconnected component between v_m and v_{m+1}, or between v_{m-1} and v_m.

Figure 53 gives an example for each case.

For case 1, the local information that is computed for each partial one-path connected to v_m is the same as for trees of pathwidth two. For case 2, we have to compute information if there is a biconnected component which contains v_m. For case 3, we have to compute extra information for the connecting biconnected components. Case

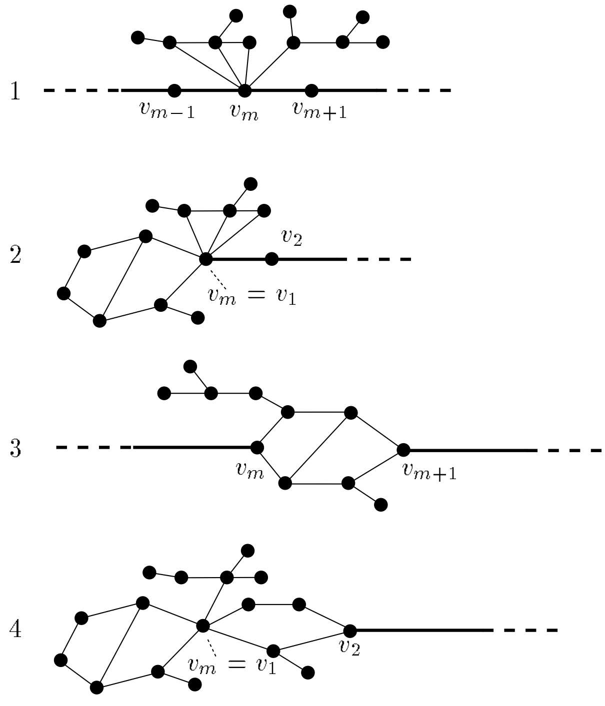

Figure 53: Examples of possible kinds of vertices in nice path $P = (v_1,..., v_q)$. In case 1, $1 < m < q$, and there is no connecting biconnected component between v_{m-1} and v_m or between v_m and v_{m+1}. In case 2, $m = 1$, and there is no connecting biconnected component between v_1 and v_2. In case 3, $1 < m < q$ and there is a connecting biconnected component between v_m and v_{m+1}. In case 4, $m = 1$, and there is a connecting biconnected component between v_1 and v_2.

4 is a combination of cases 2 and 3. All other information remains the same, although it must be computed slightly different.

Case 1 $v_m \in V(P)$, $1 < m < q$, and no connecting biconnected component contains v_m

Let $v_m \in V(P)$ such that $1 < m < q$, there is no connecting biconnected component containing v_m, and there is at least one partial one-path connected to v_m. Let pp, p, n and nn, $pp \leq p \leq m \leq n \leq nn$, be defined as follows. Vertex v_p is the rightmost vertex on the left side of v_m which is contained in a biconnected component or to which a partial one-path is connected, or $p = 1$ if there is no such vertex, and v_n is the leftmost vertex on the right side of v_m which is contained in a biconnected component or to which a partial one-path is connected, or $n = q$ if there is no such vertex. If there is a connecting biconnected component between v_{p-1} and v_p, then $pp = p$, otherwise, v_{pp} is the rightmost vertex on the left side of v_p which is contained in a biconnected component or to which a partial one-path is connected, $pp = 1$ if there is no such vertex. Analogously, if there is a connecting biconnected component between v_n and

v_{n+1}, then $nn = n$, otherwise, v_{nn} is the leftmost vertex on the right side of v_n which is contained in a biconnected component or to which a partial one-path is connected, $nn = q$ if there is no such vertex.

Note that this definition is correct, e.g. if there is a connecting biconnected component between v_{p-1} and v_p, then partial one-paths connected to v_m can not use any j, $j < p$, because of Corollary 4.4.

First consider the local information for partial one-paths of type I, II or III which are connected to v_m. For these partial one-paths, we compute the same information as for trees of pathwidth two, i.e. if there is more then one partial one-path connected to v_m then the fields l, r, lll, llr, rrl and rrr are computed for all partial one-paths which have a vertex of color $c(v_m)$, and only fields l and r are computed for all partial one-paths which have no vertex of color $c(v_m)$. If there is only one partial one-path connected to v_m, then fields lr, lll, llr, rrl and rrr are computed for this partial one-path. This information can be computed in the same way as for trees of pathwidth two.

Now consider the partial one-paths of type IV which are connected to v_m. We use the same local information for these partial one-paths, i.e. we compute fields l, r, lr, lll, llr, rrl and rrr, as is shown here. Let H' be a partial one-path of type IV which is connected to v_m. Let PD be a nice proper path decomposition of G with nice path P, suppose H' uses $[j, j']$. We consider three different cases.

1.1 $pp \leq j \leq j' \leq p$ or $n \leq j \leq j' \leq nn$.

1.2 There are two or more partial one-paths connected to v_m, and $p \leq j \leq j' \leq m$ or $m \leq j \leq j' \leq n$.

1.3 H' is the only partial one-path connected to v_m and $p \leq j \leq j' \leq n$.

Case 1.1 $pp \leq j \leq j' \leq p$ or $n \leq j \leq j' \leq nn$

It is not possible that $j \geq n$, and there is a partial one-path H'' connected to v_n which uses $[l, l']$, $p \leq l \leq l' \leq m$, because of Lemma 4.18. Hence $p \leq j \leq j' \leq n$. This means that the fields $lll.ok$, $llr.ok$, $rrl.ok$ and $rrr.ok$ are false.

Case 1.2 There are two or more partial one-paths connected to v_m, and $p \leq j \leq j' \leq m$ or $m \leq j \leq j' \leq n$

This case corresponds to Cases 1 and 4 for trees of pathwidth two (see page 112). Claim 4.1 in Case 1 for trees holds for H', so we can use Definition 4.5 for the local information for this case, which means that the local information is an integer j_1, $p \leq j_1 \leq m$.

We now show how to compute j_1.

Let $P' \in \mathcal{P}_1(H')$, let u and w be the two end points of P'. For each j, $p \leq j \leq m$, let G^u_j denote the graph obtained from G as follows (see e.g. Figure 54). Add edge $\{u, v_j\}$, and edge $\{w, v_m\}$. For each stick v' of some $v \in V(P')$ for which $\{v', v_m\} \in G$, add edge $\{v_m, v\}$. Furthermore, delete vertices $\{v_1, \ldots, v_{j-1}, v_{m+1}, \ldots, v_q\}$ and all sticks and partial one-paths adjacent to these vertices, all sticks adjacent to v_j and v_m, all partial one-paths adjacent to v_j and all partial one-paths except H' that are adjacent to v_m. Define G^w_j in the same way for each j, $p \leq j \leq m$. Note that G^u_j and G^w_j are biconnected components with sticks.

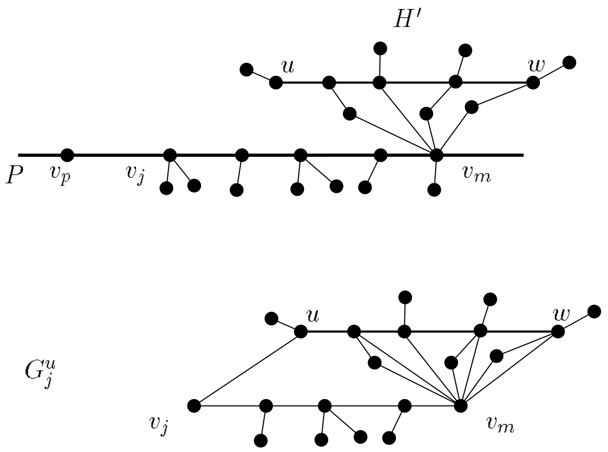

Figure 54: Example of G^u_j for a given partial one-path H' of type IV which is connected to v_m.

Let l_1 be the largest j, $p \leq j \leq m$, for which there is a proper path decomposition of G^u_j with vertex v_m in the rightmost node and edge $\{v_j, u\}$ in the leftmost node, undefined if there is no such proper path decomposition. Let l_2 be the largest j, $p \leq j \leq m$, for which there is a proper path decomposition of G^w_j with end vertex v_m and end edge $\{v_j, w\}$, undefined if there is no such proper path decomposition.

Claim 4.2 of Case 1 for trees now holds for l_1 and l_2, which can be shown in the same way as for Claim 4.2. The values of l_1 and l_2 can be computed in $O(n^2)$ time with use of $PPW2$ and $PPW2'$, in the same way as in Case 1 for trees.

Hence field l can be computed as follows. If H' has no vertex of color $c(v_m)$, then $l.ok$ is true and $l.v = m$. If H' has vertices of color $c(v_m)$, then j_1 is computed. If j_1 is defined, then $l.ok$ is true, and $l.v = j_1$, otherwise, $l.ok$ is false and $l.v = p$.

This completes the description of the local information for the case that $p \leq j \leq j' \leq m$. The case that $m \leq j \leq j' \leq n$ can be done in the same way.

Case 1.3 H' is the only partial one-path connected to v_m and $p \leq j \leq j' \leq n$
This case corresponds to Case 2 for trees of pathwidth two (see page 75). In fact, it corresponds to case 2.1, since the graph $G[V(H') \cup \{v_m\}]$ contains a biconnected component of which v_m is double end point. This means that there is an edge $e \in E(H')$ for which there is a node in the path decomposition which contains v_m and e, so $j \leq m \leq j'$.

Claim 4.4 of Case 2.1 for trees also holds for H', hence Definition 4.6 defines the local information for H', which consists of the set Q_1.

We now show how to compute the set Q_1 for H', and that $|Q_1| \leq 4$.

Let $P' \in \mathcal{P}_1(H')$, let u and w be the two end points of P'. Let G' be the graph obtained from G by adding all edges $\{v, v_m\}$, for which $v \in V(P')$ and there is a stick

v' of v in H' for which $\{v', v_m\} \in E(G)$. Let $v_u \in V(P')$ be the vertex with smallest distance to u, such that $\{v_u, v_m\} \in E(G')$, and let $v_w \in V(P')$ be the vertex with smallest distance to w, such that $\{v_w, v_m\} \in E(G')$. (Note that $v_u = u$ and $v_w = w$ are possible.)

For each j, $p \leq j \leq m$, let G_j^u denote the graph obtained from G' as follows (see e.g. Figure 57). Add edge $\{u, v_j\}$. Delete vertices $\{v_1,..., v_{j-1}, v_{m+1},..., v_q\}$, and all sticks and partial one-paths connected to $\{v_1,..., v_j, v_m,..., v_q\}$, except H'. Delete all components of $G[V(H') - \{v_u\}]$ which do not contain u. Note that the remaining graph G_j^u is a chordless cycle with sticks.

For each j', $m \leq j' \leq n$, let $G_{j'}^w$ be the graph obtained from G' as follows. Add edge $\{w, v_{j'}\}$. Delete vertices $\{v_1,..., v_{m-1}, v_{j'+1},..., v_q\}$, and all sticks and partial one-paths connected to $\{v_1,..., v_m, v_{j'},..., v_q\}$, except H'. Delete the component of $G[V(H') - \{v_u\}]$ which contains u.

Similarly, for each j, $p \leq j \leq m$, let G_j^w be the graph obtained from G' as follows. Add edge $\{w, v_j\}$. Delete vertices $\{v_1,..., v_{j-1}, v_{m+1},..., v_q\}$, and all sticks and partial one-paths connected to $\{v_1,..., v_j, v_m,..., v_q\}$, except H'. Delete all components of $G[V(H') - \{v_w\}]$ which do not contain w.

Furthermore, for each j', $m \leq j' \leq n$, let $G_{j'}^u$ denote the graph obtained from G as follows. Delete vertices $\{v_1,..., v_{m-1}, v_{j'+1},..., v_q\}$, and all sticks and partial one-paths connected to $\{v_1,..., v_m, v_{j'},..., v_q\}$, except H'. Delete the components of $G[V(H') - \{v_w\}]$ which contains w.

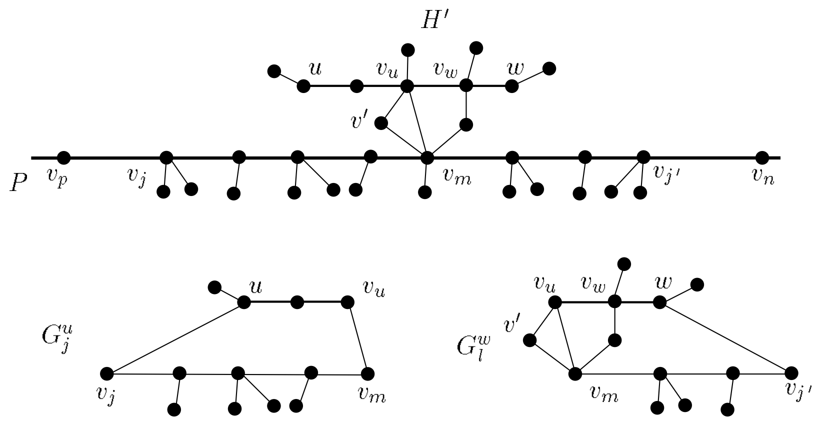

Figure 55: Example of G_j^u and $G_{j'}^w$, for a given partial one-path H' of type IV, which is connected to v_m, and with $p \leq j \leq m$ and $m \leq j' \leq n$.

Let l_1^u be the largest value of j, $p \leq j \leq m$, for which there is a proper path decomposition of $G_j^u \cup \{$ sticks of v_m $\}$ with edge $\{v_m, v\}$ in the rightmost node, edge $\{v_j, u\}$ in the leftmost node, or $j = m$ and there is a proper path decomposition of G_m^u with edge $\{u, v_m\}$ in the leftmost node and edge $\{v, v_m\}$ in the rightmost node. If there is no such j, then l_1^u is undefined.

Let r_1^w be the smallest value of j', $m \leq j' \leq n$, for which there is a proper path decomposition of $G_{j'}^w \cup \{$ sticks of v_u $\}$ with edge $\{v_m, v\}$ in the leftmost node and edge $\{v_{j'}, u\}$ in the rightmost node. If there is no such j', then r_1^w is undefined.

Let l_2^u be the largest value of j, $p \leq j \leq m$, for which there is a proper path decomposition of $G_j^u \cup \{$ sticks of v_u $\}$ with edge $\{v_m, v\}$ in the rightmost node and $\{v_j, u\}$ in the leftmost node. If there is no such j, l_2^u is undefined.

Let r_2^w be the smallest value of j', $m \leq j' \leq n$, for which there is a proper path decomposition of $G_j^w \cup \{$ sticks of v_m $\}$ with edge $\{v_m, v\}$ in the leftmost node and edge $\{v_{j'}, u\}$ in the rightmost node, or $j' = m$ and there is a proper path decomposition of $G_{j'}^w$ with edge $\{v, v_m\}$ in the leftmost node and edge $\{w, v_m\}$ in the rightmost node. If there is no such a j', then r_2^w is undefined.

Define l_1^w, r_1^u, l_2^w and r_2^u similarly.

Let Q_1' be defined as follows.

$$Q_1' = \{ (j, j') \in \{(l_1^u, r_1^w), (l_2^u, r_2^w), (l_1^w, r_1^u), (l_1^w, r_1^w)\} \mid j \text{ and } j' \text{ are not undefined} \}$$

Claim 4.5 also holds for Q_1', which can be shown in the same way as for Claim 4.5. The values of l_1^u, r_2^w, etc. can be computed in $O(n^2)$ time in the same way as for Case 2.1 for trees.

Field lr is now computed as follows. If there are two or more partial one-paths connected to v_m, then $lr.ok$ is false, If H' is the only partial one-path connected to v_m, then $lr.ok$ is true if and only if Q_1 is not empty. In this case, the values of $lr.l_a$ and $lr.r_a$, $1 \leq a \leq 8$, are such that

$$Q_1 = \{ (v_m.p[i].lr.l_a, v_m.p[i].lr.r_a) \mid 1 \leq a \leq 8 \}.$$

If $lr.ok$ is false, then $lr.l_a = p$ and $lr.r_a = n$, for all a, $1 \leq a \leq 8$. This completes the description of Case 1.

Case 2 $v_m \in V(P)$, $m \in \{1, q\}$, and there is no connecting biconnected component containing v_m.

First consider the case that $m = 1$ and $q > 1$. Suppose there is no connecting biconnected component between v_1 and v_2. If there is no biconnected component at all which contains v_1, then there is no partial one-path connected to v_1, because of the choice of nice paths, and there is no local information to compute. Suppose there is a non-connecting biconnected component containing v_1. Let $pp = p = 1$, let v_n be the leftmost vertex of P on the right side of v_1 which is contained in a biconnected component or to which a partial one-path is connected, if there is no such vertex then $n = q$. If there is a connecting biconnected component between v_n and v_{n+1}, then let $nn = n$, otherwise let v_{nn} be the leftmost vertex on the right side of v_n which is contained in a biconnected component or to which a partial one-path is connected. If there is no such vertex then let $nn = q$. Let $H_1, \ldots, H_{nr}$ be the partial one-paths of type I, II and III which are connected to v_1. Let $B_1, \ldots, B_{nr'}$ be the (non-connecting) biconnected components which contain v_1. For each i, let $G_i = G[V(G_{B_i}) - \{v_1\}]$.

First consider the local information for biconnected components which contain v_1.

The local information for the case that a biconnected component is handled as ending biconnected component is called *ending biconnected component info*, and the local information for the case that a biconnected component is handled as a partial one-path of type IV is called *partial one-path info*. Lemma 4.21 and Lemma 4.26 show that for a given biconnected component B_i, the following local information must be computed (assumed that there are at most three i, $1 \leq i \leq nr'$, for which G_i has a vertex of color $c(v_1)$).

1. There is an i' for which $\mathrm{cond}_1(st(B_{i'}))$ does not hold.
   (a) If $i' \neq i$, then the partial one-path info is computed.
   (b) If $i' = i$, then the ending biconnected component info is computed.
2. For all i', $\mathrm{cond}_1(st(B_{i'}))$ holds.
   (a) If $nr' + nr = 1$, then the ending biconnected component info is computed.
   (b) If $nr' + nr = 2$, then the ending biconnected component info and the partial one-path info are computed.
   (c) $nr' + nr \geq 3$ and G_i has a vertex of color $c(v_1)$, then the ending biconnected component info is computed.
   (d) $nr' + nr \geq 3$ and G_i has no vertex of color $c(v_1)$, but there is an $i' \neq i$ for which $G_{i'}$ has a vertex of color $c(v_1)$, or there is a j, $1 \leq j \leq nr$, for which H_j has a vertex of color $c(v_1)$, then the partial one-path info is computed.
   (e) $nr' + nr \geq 3$ and there is no i', $1 \leq i' \leq nr'$ for which $G_{i'}$ has a vertex of color $c(v_1)$, and there is no j, $1 \leq j \leq nr$, for which H_j has a vertex of color $c(v_1)$, and B_i is selected to be ending biconnected component(in the sense of Lemma 4.21), then the ending biconnected component info is computed.
   (f) $nr' + nr \geq 3$ and there is no i', $1 \leq i' \leq nr'$, for which $G_{i'}$ has a vertex of color $c(v_1)$, there is no j, $1 \leq j \leq nr$, for which H_j has a vertex of color $c(v_1)$, and B_i is not selected to be ending biconnected component, then the partial one-path info is computed.

Note that if for all i, $1 \leq i \leq nr'$, $\mathrm{cond}_1(st(B_i))$ holds, $nr' + nr \geq 3$ and there is no i, $1 \leq i \leq nr'$ for which G_i has a vertex of color $c(v_1)$, and there is no j, $1 \leq j \leq nr$, for which H_j has a vertex of color $c(v_1)$, then at most one B_i is selected to be ending biconnected component, because of Lemma 4.21.

Note furthermore that if for all i, $1 \leq i \leq nr'$, $\mathrm{cond}_1(st(B_i))$ does not holds, and B_i has two or more vertices of state E1, or one vertex of state E2, then we do not have to compute the ending biconnected component info for B_i, because of Lemma 4.26.

For partial one-paths of type I, II or III, also the partial one-path information is computed.

We now show what the ending biconnected component info and the partial one-path info consist of, and how they are computed.

Ending biconnected component info. Let r, $1 \leq r \leq nr'$, be such that ending biconnected component info must be computed for B_r. If B_r has two or more vertices of state E1, or one vertex of state I1, then the ending biconnected component info for B_r is false. Suppose B_r has at most one vertex of state E1, and no vertices of state I1. We first analyze the structure of a nice proper path decomposition with nice path $P = (v_1,..., v_q)$ in which G_{B_r} occurs in $(V_1,..., V_{j'})$, and the rightmost vertex of P which occurs in $V_{j'}$ is v_l.

Claim 4.15. If G_{B_r} occurs in $(V_1,..., V_{j'})$, and the rightmost vertex of P which occurs in $V_{j'}$ is v_l, then a partial one-path H' of type I, II, III or IV connected to $v_{m'}$, $m' \geq 1$, can use $[a, a']$, with $a \geq m'$.

Proof. It follows from Corollary 4.5 that $a \geq m'$. Showing that it is possible that $a = m'$ can be done in the same way as for in the proof of Claim 4.1, Case 1 for trees. □

It follows from the claim that we only need the smallest value of l, $1 \leq l \leq n$, for which it is possible that v_l is the rightmost vertex of P which occurs within the occurrence of G_{B_r}.

Definition 4.15. *The ending biconnected component info for B_r is j_1, $1 \leq j_1 \leq n$, which is the smallest value of l for which G_{B_r} can occur in $(V_1,..., V_{j'})$, and v_l is the rightmost vertex of P which occurs in $V_{j'}$.*

We now show how to compute j_1. Therefore, we consider two cases, namely the case that B_r has a vertex of state E1, and the case that B_r has no vertices of state E1.

Suppose B_r has a vertex w of state E1. Let H_w be the component of G_{T} which contains w. Note that $P_1(H_w)$ is unique. Let $P' = (w_1,..., w_r)$ be the shortest path containing w and $P_1(H_w)$, such that $w_1 = w$. Let H'_w be the graph obtained by deleting all sticks of w of H_w.

It must be the case that v_1 and w are in the same chordless cycle of $\bar{B}_r$, and that either $\{v_1, w\} \in E(B_i)$, or v_1 and w have a common neighbor which has no sticks.

Let G' be the graph obtained from G by adding edge $\{v_1, w\}$ if it is not present, and deleting $v_2,..., v_q$, and all sticks, partial one-paths and biconnected components connected to these vertices, $H_w - \{w_1\}$, and all partial one-paths and sticks connected to v_1. Note that G' is a biconnected component with sticks. See for example Figure 56.

For each j, $1 \leq j \leq n$, let G_j be the graph obtained from G by adding edge $\{v_1, w\}$ if it is not present, adding edge $\{w_r, v_j\}$, and deleting $G_i - \{w\}$, all sticks and partial one-paths connected to v_1, all vertices $v_{j+1},..., v_q$ and all sticks, partial one-paths and biconnected components connected to vertices $v_j,..., v_q$. Note that G_j is a cycle with sticks. See for example Figure 56.

Let l_1 be the smallest j, $1 \leq j \leq n$, for which there is a proper path decomposition of $G' \cup \{\text{sticks of } v_1\}$ with edge $\{v_1, w\}$ in the rightmost node, and there is a proper path decomposition of $G_j \cup \{\text{sticks of } w\}$ with edge $\{v_1, w\}$ in the leftmost node and edge $\{w_r, v_j\}$ in the rightmost node. If there is no such j, then l_1 is undefined.

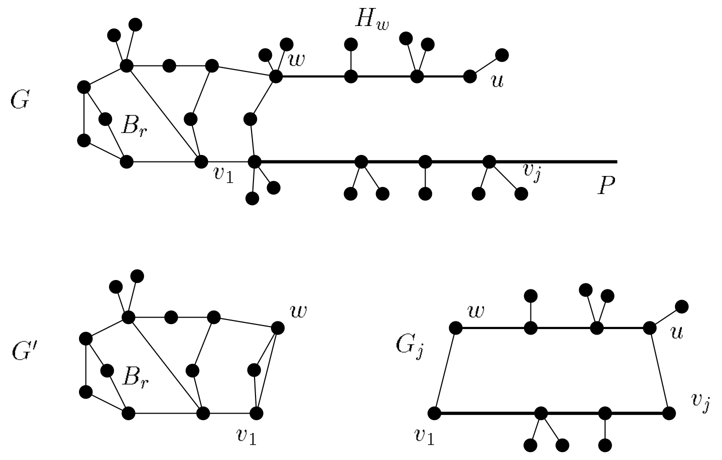

Figure 56: Examples of G' and G_j for the case that B_r has a vertex w of state E1.

Let l_2 be the smallest j, $1 \leq j \leq n$, for which there is a proper path decomposition of $G' \cup \{\text{sticks of } w\}$ with edge $\{v_1, w\}$ in the rightmost node, and either there is a proper path decomposition of $G_j \cup \{\text{sticks of } v_1\}$ with edge $\{v_1, w\}$ in the leftmost node and edge $\{w_r, v_j\}$ in the rightmost node, or $j = 1$, and there is a proper path decomposition of G_1 with edge $\{w_r, v_1\}$ in the rightmost node. If there is no such j, then l_2 is undefined.

Claim 4.16. Suppose B_r has a vertex w of state E1, and suppose j_1 is defined. Then $j_1 = \min\{l_1, l_2\}$.

Proof. This can be shown in the same way as Claim 4.2, Case 1 for trees. □

Claim 4.17. If B_r has a vertex of state E2, then j_1 can be computed in $O(n^2)$ time, where n is the number of vertices of $G_{B_r} \cup \{v_1,..., v_n\} \cup \{\text{sticks adjacent to } v_1..., v_{n-1}\}$.

Proof. The computations can be done using $PPW2$: $PPW2$ has to be used twice for G' and twice for G_n. □

Suppose B_r has no vertices of state E2. We now show how to compute j_1 for this case. Let $(\mathcal{C}, \mathcal{S})$ be a correct path of chordless cycles for $\bar{B}_r$.

Claim 4.18. Suppose B_r has no vertices of state E1. $j_1 = 1$ if there is a proper path decomposition of G_{B_r} with v_1 in the rightmost node. $j_1 = 2$ if there is no proper path decomposition of G_{B_r} with v_1 in the rightmost node, but there is a proper path decomposition of $G_{B_r} - \{\text{sticks of } w\} \cup \{\text{sticks of } v_1\}$ with w and v_1 in the rightmost node, where $w \in V(B_r)$ and $\text{dst}_1(v_1, w)$ or $\text{dst}_p(v_1, w)$ holds. Otherwise, j_1 is undefined.

Proof. Clearly, there is a proper path decomposition of G_{B_r} with v_1 in the rightmost node if and only if $j_1 = 1$.

Suppose there is no such proper path decomposition, but j_1 is defined. Suppose $j_1 > 1$, let $PD = (V_1,..., V_t)$ be a proper path decomposition of $G[V(G_{B_r}) \cup \{v_2,..., v_{j_1}\} \cup \{\text{sticks adjacent to } v_1,..., v_{j_1-1}\}]$ with $v_{j_1} \in V_t$. Suppose B_r occurs in $(V_j,..., V_{j'})$. Then there must be a vertex $w \in V(B)$ of which a stick occurs on the right side of the occurrence of B_r in PD. There can be at most one w for which this holds, and furthermore, $w \in V_{j'}$, and $\mathrm{dst}_1(v_1, w)$ or $\mathrm{dst}_p(v_1, w)$ must hold. Hence $V_{j'}$ contains v_1 and w, and $w \in V_{j_1}$. This means that $c(w) \neq c(v_2)$, $c(v_1) \neq c(v_2)$ and $c(v_1) \neq c(w)$, so the sticks of w have color $c(v_1)$ or color $c(v_2)$. Furthermore, all sticks of v_1 of color $c(w)$ must occur on the left side of $V_{j'}$, since each V_i, $i \geq j'$, contains w.

Let PD' be the proper path decomposition of $G_{B_r} \cup \{\text{sticks of } v_1\}$ which is obtained from PD as follows. Delete $V_{j'+1},..., V_t$, add a node $\{v_1, w, w'\}$ on the right side of $V_{j'}$ for each stick w' of w which has color $c(v_2)$, then add a node $\{v_1, v_2, w\}$ on the right side, then add a node $\{v_2, w, w'\}$ on the right side for each stick w' of w with color $c(v_1)$. v_2 is in the rightmost node of this proper path decomposition, hence $j_1 = 2$. Contradiction. □

Claim 4.19. If B_r has no vertices of state E2, then j_1 can be computed in $O(n^2)$ time, where n is the number of vertices of $G_{B_r} \cup \{\text{sticks of } v_1\}$.

Proof. The computations can be done using $PPW2$ and $PPW2'$: there are at most two candidates for vertex w, and $PPW2$ has to be used twice, $PPW2'$ once. □

This completes the description of the ending biconnected component info.

Partial one-path info. Let H' be a partial one-path connected to v_1, i.e. either $H' = H_i$ for some i, $1 \leq i \leq nr$, or $H' = G_i$ for some i, $1 \leq i \leq nr'$ for which the partial one-path info must be computed.

Claim 4.20. If the ending biconnected component info $j_1 > 1$ for some biconnected component B_i, then there is no proper path decomposition of $G_i \cup \{v_1,..., v_{j_1}\} \cup \{\text{ sticks of } v_1,..., v_{j_1-1} \}$ with v_1 in rightmost node.

Proof. Suppose there is a proper path decomposition $PD = (V_1,..., V_t)$ of $G_i \cup \{v_1,..., v_{j_1}\} \cup \{\text{ sticks of } v_1,..., v_{j_1-1} \}$ with v_1 in rightmost node. Then $PD[V(G_i')]$ is a proper path decomposition of G_i' with v_1 in the rightmost node. Hence $j_1 = 1$. □

The claim implies that the partial one-path info can be computed in the same way as in Case 1, for partial one-paths connected to v_m, $1 < m < q$ in which no non-connecting biconnected component contains v_m (note that $pp = p = 1$). This completes the case that $m = 1$ and $q > 1$.

The case that $m = q$ and $q > 1$ is similar, except that $n = nn = q$, p is the largest j, $j < q$, for which there is a partial one-path connected to v_j, or there is a biconnected component containing v_j. If there is no such j, then $p = 1$. Furthermore, $pp = p$ if there is a connecting biconnected component between v_{p-1} and v_p, otherwise pp is the largest j, $j < p$, for which there is a partial one-path connected to v_j, or there is a biconnected component containing v_j. If there is no such j, then $pp = 1$. Furthermore, the ending biconnected component info for a biconnected component B_i containing v_q consists of value j_2, which is the largest j, $p \leq j \leq q$, for which there is a proper path decomposition of $G_i \cup \{v_j, ..., v_q\} \cup \{\text{sticks of } v_{j+1}, ..., v_q\}$ with vertex v_j in the leftmost node.

The case that $m = q = 1$ is also similar to the case that $m = 1$ and $q > 1$, except that $pp = p = m = n = nn = 1$. The ending biconnected component info and the partial one-path info are the same as for the case that $q > 1$, but there may be more biconnected components for which the ending biconnected component info is computed. It is not shown here for which ending biconnected components the ending biconnected component info must be computed, but it can be shown in the same way as for the case that $q > 1$, and the number of biconnected components for which it must be done is still at most four.

Case 3 $v_m \in V(P)$, $1 < m < q$, and there is a connecting biconnected component containing v_m.

First suppose there is a connecting biconnected component B which connects v_{m-1} and v_m.

We first consider the local information for partial one-paths connected to v_m. After that, we consider the local information for biconnected component B.

Local information for partial one-paths connected to v_m. Let pp, p, n and nn be defined as follows. $p = m$, $pp = m - 1$. If there is a connecting biconnected component between v_m and v_{m+1}, then $n = m$ and $nn = m + 1$, otherwise, n is the smallest $j > m$ such that there is a connecting biconnected component containing v_j, or a partial one-path connected to v_j. If there is no such j, $n = q$. Furthermore, if there is a connecting biconnected component between v_n and v_{n-1}, then $nn = n$, otherwise nn is the smallest $j > n$ such that there is a connecting biconnected component containing v_j, or a partial one-path connected to v_j. If there is no such j, $nn = q$. Note that pp, p, n and nn are well-defined, since partial one-paths of type II, III and IV can use $[j, j']$, with $p \leq j \leq j' \leq n$ only, and partial one-paths of type I can use $[j, j']$ with $p \leq j \leq j' \leq n$, $n \leq j \leq j' \leq nn$, or $pp \leq j' \leq j' \leq p$ (see Lemma 4.23).

For partial one-paths of type II, III and IV, the local information for this case is the same as for the case that v_m does not contain a connecting biconnected component, because of Corollary 4.4 and Lemmas 4.23 and 4.24. Now consider a partial one-path H' of type I which is connected to v_m. For the case that H' uses $[j, j']$ for some $p \leq j \leq j' \leq m$, $m \leq j \leq j' \leq n$ or $p \leq j \leq j' \leq n$, the local information is the same as for the case that there is no connecting biconnected component containing v_m. For the case that H' uses $[j, j']$, $n \leq j \leq j' \leq nn$, and there is no connecting biconnected component between v_m and v_{m+1}, the local information is also the same.

Consider the case that H' uses $[j, j']$, $pp \leq j \leq j' \leq p$. This case is similar to Case 3 for trees (see Page 84). The analogs of Claim 4.11 and Claim 4.12 in Case 3 for trees also hold for H', because of Lemma 4.23. This means that we can use Definition 4.9 for the local information for H' if it uses $[j, j']$, $pp \leq j \leq j' \leq p$, and Claim 4.13 and

Claim 4.14 in Case 3 for trees also hold.

If there is no connecting biconnected component between v_m and v_{m+1}, then the local information for the case that H' uses $[j, j']$, $n \leq j \leq j' \leq nn$ is the same as in Case 1. If there is a connecting biconnected component then the local information for the case that H' uses $[j, j']$, $n \leq j \leq j' \leq nn$ is similar to the local information for the case that H' uses $[l, l']$, $pp \leq l \leq l' \leq p$.

Local information for biconnected component B connecting v_{m-1} and v_m. We first analyze the structure of a nice proper path decomposition $PD = (V_1, ..., V_t)$ of G with nice path P in which G_B uses $(V_j, ..., V_{j'})$. Consider the different possibilities in PD for partial one-paths connected to v_{m-1} and v_m. There are four different cases.

3.1 One partial one-path connected to v_{m-1} occurs on the right side of $V_{j'}$ and one partial one-path connected to v_m occurs on the left side of V_j.

3.2 One partial one-path connected to v_{m-1} occurs on the right side of $V_{j'}$ and all partial one-paths connected to v_m occur on the right side of $V_{j'}$.

3.3 All partial one-paths connected to v_{m-1} occur on the left side of V_j and one partial one-path connected to v_m occurs on the left side of V_j.

3.4 All partial one-paths connected to v_{m-1} occur on the left side of V_j and all partial one-paths connected to v_m occur on the right side of $V_{j'}$.

For each of these cases we have to compute local information which shows whether this case is possible w.r.t. G_B.

Case 3.1 A partial one-path connected to v_{m-1} occurs on the right side of $V_{j'}$, a partial one-path connected to v_m occurs on the left side of V_j

According to Lemma 4.23, v_{m-1} and v_m are both in V_j and in $V_{j'}$. See e.g. Figure 57. This means that G_B is a biconnected component with sticks, and there is a proper path decomposition of G_B with v_{m-1} and v_m in the leftmost and rightmost end node. The sticks of v_{m-1} which have color $c(v_m)$, and the sticks of v_m which have color $c(v_{m-1})$ must occur either on the right side or on the left side of the occurrence of G_B. The sticks of v_m which do not have color $c(v_{m-1})$ and the sticks of v_{m-1} which do not have color $c(v_m)$ can always be made to occur within the occurrence of G_B, because edge $\{v_{m-1}, v_m\}$ is a middle edge. Hence this gives the following definition of the local information for this case.

Definition 4.16. *The local information for B for the case that there is a partial one-path connected to v_{m-1} which may occur on the right side of the occurrence of G_B, and there is a partial one-path connected to v_m which may occur on the left side of the occurrence of G_B is a boolean b which is true if and only if there is a proper path decomposition of G_B with v_{m-1} and v_m in the leftmost and in the rightmost end node.*

Note that b can be computed in $O(n^2)$ using $PPW2$, where $n = |V(G_B)|$, since G_B is a biconnected component with sticks.

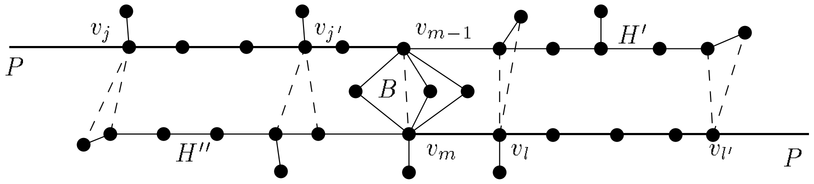

Figure 57: Example of the case that a partial one-path connected to v_{m-1} uses $[l, l']$, $l \geq m$, and a partial one-path connected to v_m uses $[j, j']$, $j \leq m-1$. In this example, there is one partial one-path connected to v_{m-1}, and one connected to v_m.

Case 3.2 A partial one-path connected to v_{m-1} occurs on the right side of $V_{j'}$, all partial one-paths connected to v_m occur on the right side of $V_{j'}$
According to Lemma 4.23, v_{m-1} and v_m both occur in the rightmost node of the occurrence of G_B. For example, see Figure 58.

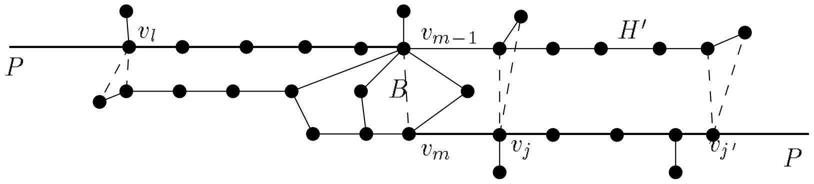

Figure 58: Example of the case that a partial one-path H' connected to v_{m-1} uses $[j, j']$, $j \geq m$.

Claim 4.21. If G_B occurs in $(V_j,..., V_{j'})$, $v_{m-1}, v_m \in V_{j'}$, and l is the smallest integer, $l \leq m-1$, for which $v_l \in V_j$, then a partial one-path H' connected to $v_{m'}$, $1 \leq m' \leq q$, can use $[a, a']$, where $a' \leq l$ if $m' < m-1$, $a \geq m$ if $m' \geq m$, and $a' \leq l$ or $a \geq m$ if $m' = m-1$.

Proof. It is clear that $a' \leq l$ if $m' < m-1$, that $a' \leq l$ or $a \geq l'$ if $m' = m-1$, and $a \geq m$ if $m' \geq m$ (Corollary 4.4, Lemmas 4.23 and 4.24). Showing that $a' = l$ and $a = m$ are possible can be done in the same way as in the proof of Claim 4.1, Case 1 for trees. □

If there are two or more partial one-paths connected to v_{m-1}, let $p = m-1$, otherwise let p be the largest integer $p < m-1$ for which there is a partial one-path or biconnected component connected to v_p. It follows from the claim that for a partial one-path H' which occurs on the left side of the occurrence of G_B, we only need the largest l, $p \leq l \leq m-1$, for which H' can use l at most. For a partial one-path H' connected to v_{m-1} which uses $[j, j']$, $j \geq m$, we need more information about the sticks of v_{m-1} and v_m, since there is a node containing v_{m-1} and v_m, and the sticks of v_{m-1} which have color $c(v_m)$ occur either on the left side or on the right side of the occurrence of $\{v_{m-1}, v_m\}$. If they occur on the left side, then the sticks of v_m with color $c(v_{m-1})$ occur on the right side, and vice versa (see also Case 3 for trees, Page 84). This means that we can define the local information for this case as follows.

Definition 4.17. *The local information for B for the case that there is a partial one-path H' connected to v_{m-1} which uses $[j, j']$, $j \geq m$, is the pair (j_1, j_2), $p \leq j_1, j_2 \leq m-1$, where*

- *j_1 is the largest value of j, $p \leq j \leq m-1$, for which there is a proper path decomposition of $G_B \cup \{v_j, ..., v_{m-1}\} \cup \{$sticks of $v_{j+1}, ..., v_{m-1}\}$ with v_{m-1} and v_m in the rightmost node and v_j in the leftmost node (j_1 is undefined if there is no such j), and*

- *j_2 is the largest value of j, $p \leq j \leq m-1$, for which there is a proper path decomposition of $G_B \cup \{v_j, ..., v_{m-1}\} \cup \{$sticks of $v_{j+1}, ..., v_{m-2}, v_m\}$ with v_{m-1} and v_m in the rightmost node and v_j in the leftmost node (j_2 is undefined if there is no such j).*

Note that B has at most one vertex of state E1, and no vertices of state I1, since edges of $G_B - B$ can only occur on the left side of the occurrence of B.

The computation of j_1 and j_2 can be done with $PPW2$ in $O(n^2)$ time, where $n = |V(G_B) \cup \{v_j, ..., v_{m-1}\} \cup \{\text{sticks of } v_{j+1}, ..., v_m\}|$. This can be shown in the same way as for Case 2 for trees.

Case 3.3 All partial one-paths connected to v_{m-1} occur on the left side of V_j, a partial one-path connected to v_m occurs on the left side of V_j

According to Lemma 4.23, v_{m-1} and v_m both occur in the leftmost node of the occurrence of G_B. This case is similar to Case 3.2. If there are two or more partial one-paths connected to v_m, let $n = m$, otherwise let n be the smallest integer $n > m$ for which there is a partial one-path or biconnected component connected to v_n. The local information is defined as follows.

Definition 4.18. *The local information for B for the case that there is a partial one-path H' connected to v_m which uses $[j, j']$, $j' \leq m-1$, is the pair (j_1, j_2), $m \leq j_1, j_2 \leq n$, where*

- *j_1 is the smallest value of j, $m \leq j \leq n$, for which there is a proper path decomposition of $G_B \cup \{v_m, ..., v_j\} \cup \{$sticks of $v_{m-1}, v_{m+1} ..., v_{j-1}\}$ with v_{m-1} and v_m in the leftmost node and v_j in the rightmost node (j_1 is undefined if there is no such j), and*

- *j_2 is the smallest value of j, $m \leq j \leq n$, for which there is a proper path decomposition of $G_B \cup \{v_m, ..., v_j\} \cup \{$sticks of $v_m, ..., v_{j-1}\}$ with v_{m-1} and v_m in the leftmost node and v_j in the rightmost node (j_2 is undefined if there is no such j).*

Case 3.4 All partial one-paths connected to v_{m-1} occur on the left side of V_j, all partial one-paths connected to v_m occur on the right side of $V_{j'}$

In this case, v_{m-1} and v_m do not have to occur in an end node of the occurrence of G_B.

Claim 4.22. If G_B occurs in $(V_j,\dots,V_{j'})$, l is the smallest integer, $1 \le l \le m-1$, for which $v_l \in V_j$ and l' is the largest integer, $m \le l' \le q$, for which $v_{l'} \in V_{j'}$, then a partial one-path H'' connected to $v_{m'}$, can use $[a, a']$ with $a' \le l$ if $m' \le m-1$, and $a \ge l'$ if $m' \ge m$.

Proof. Corollary 4.4 shows that $a' \le l$ if $m' \le m-1$, and $a \ge l'$ if $m' \ge m$.

In the same way as for Claim 4.1, Case 1 for trees, we can show that it is possible that $a' = l$ of $a = l'$. □

Let p be the largest integer, $p \le m-1$, for which there is a partial one-path or biconnected component (except B) connected to v_p. Let n be the smallest integer, $n \ge m$, for which there is a partial one-path or biconnected component (except B) connected to v_n. Claim 4.22 implies that we only need all values of (j, j'), $p \le j < m \le j' \le n$, for which partial one-paths connected to $v_{m'}$ can use j at most if $m' \le m-1$, and can use j' at least if $m' \ge m$, and there is no pair (l, l') for which this also holds and $j \le l \le l' \le j'$ and $j < l$ or $l' < j'$.

Definition 4.19. *The local information for B for the case that all partial one-paths connected to v_{m-1} occur on the left side of the occurrence of G_B and all partial one-paths connected to v_m occur on the right side of the occurrence of G_B is the set*

$$\begin{aligned} Q \;=\; \{\,(j,j') \mid\; & p \le j < m \le j' \le n \wedge \textit{there is a proper path decomposition of} \\ & G_B \cup \{v_j,\dots,v_{j'}\} \cup \{\textit{sticks of } v_{j+1},\dots,v_{j'-1}\} \textit{ with } v_j \textit{ in the leftmost node} \\ & \textit{and } v_{j'} \textit{ in the rightmost node} \\ & \wedge \neg\exists_{l,l'}\, (j < l < m \le l' \le j' \vee j \le l \le m \le l' < j') \wedge \textit{there is a} \\ & \textit{proper path decomposition of } G_B \cup \{v_l,\dots,v_{l'}\} \cup \{\textit{sticks of } v_{l+1},\dots,v_{l'-1}\} \\ & \textit{with } v_l \textit{ in the leftmost node and } v_{l'} \textit{ in the rightmost node}\,\} \end{aligned}$$

We now briefly show how Q can be computed and that $|Q| \le 4$. Let W_B be the set of vertices of B which have state I1 or E1. We consider four cases, namely

1. $W_B = \{w, w'\}$, $w \ne w'$,
2. $W_B = \{w\}$ and $st(w) = \text{I1}$,
3. $W_B = \{w\}$ and $st(w) = \text{E1}$, and
4. $W_B = \emptyset$.

Figure 59 gives an example for each case.

We only show how to compute Q for the first case. All other cases are similar.

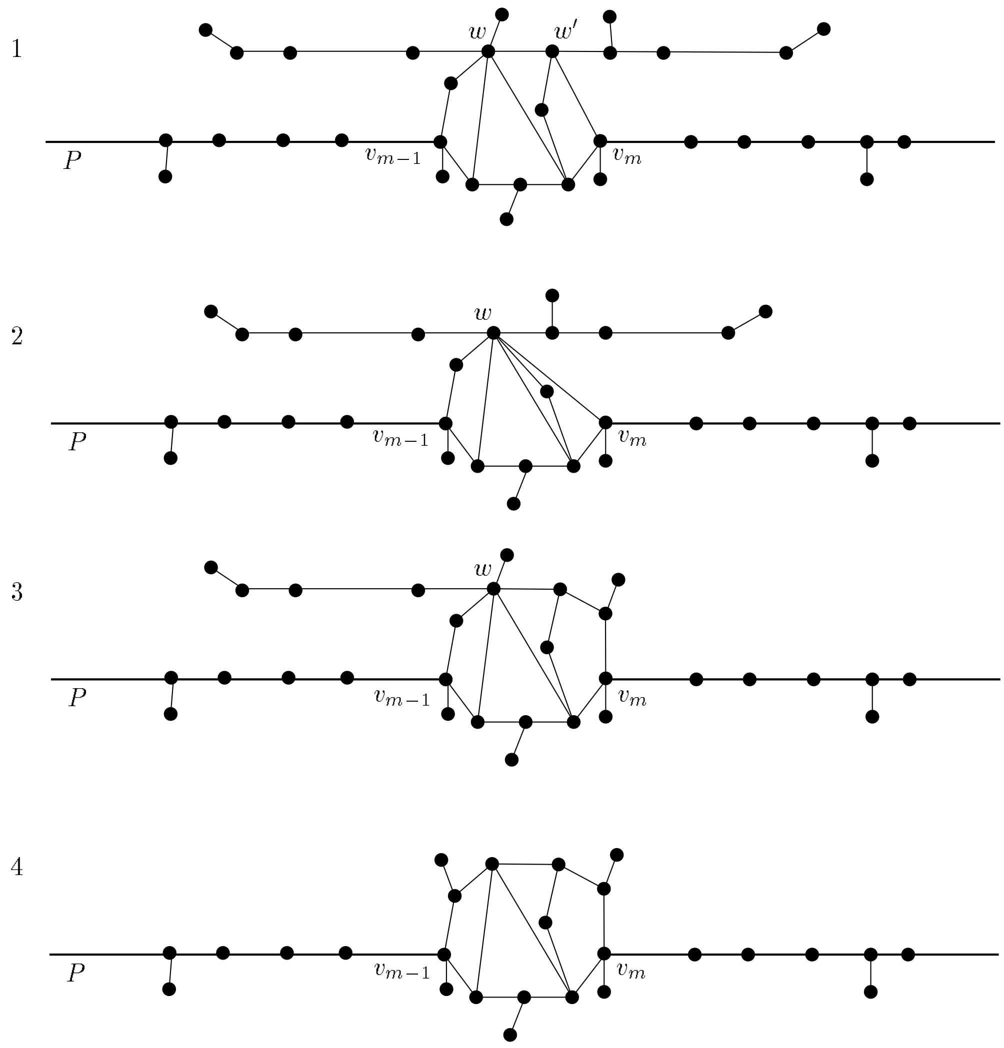

Figure 59: Examples for the different cases in values of W_B, and the different occurrences of G_B.

$\boldsymbol{W_B = \{w, w'\}, \ w \neq w'}$**.** If $W_B = \{w, w'\}$, then $st(w) = st(w') = \text{E1}$. Let H_w and $H_{w'}$ be the components of G_{T} which contain w and w', respectively, let $H'_w = G[V(H_w) - \{\text{sticks of } w\}]$, and let $H'_{w'} = G[V(H_{w'}) - \{\text{sticks of } w'\}]$. Let $(\mathcal{C}, \mathcal{S})$ be a path of chordless cycles for $\bar{B}$, with $\mathcal{C} = (C_1,\ldots, C_p)$, such that $v_{m-1} \in V(C_1)$ and $v_m \in V(C_p)$. Let u be the end point of $P_1(H'_w)$ such that the path from u to w contains $P_1(H'_w)$. Similarly, let u' be the end point of $P_1(H'_{w'})$ such that the path from u' to w' contains $P_1(H'_{w'})$. See Figure 60 for an example.

If $\text{dst}_1(v_{m-1}, w)$ and $\text{dst}_p(v_m, w')$ hold, then for each j, $p \leq j \leq m-1$, let G^u_j be the subgraph of G obtained by deleting $G_B - \{v_{m-1}\} - H'_w$, vertices $\{v_1,\ldots, v_{j-1}, v_m,\ldots, v_q\}$ and all sticks, partial one-paths and biconnected components except B which are connected to vertices $\{v_1,\ldots, v_j, v_m,\ldots, v_q\}$, and by adding edges $\{v_{m-1}, w\}$ and $\{v_j, u\}$. See e.g. Figure 60. Furthermore, for each j', $m \leq j' \leq n$, let $G^{u'}_{j'}$ be the subgraph of G obtained by deleting $H_w - \{w\}$, vertices $\{v_1,\ldots, v_{m-2}, v_{j'+1},\ldots, v_q\}$, and all sticks, partial one-paths and biconnected components except B which are connected to vertices $\{v_1,\ldots, v_{m-1}, v_{j'},\ldots, v_q\}$, and by adding edges $\{v_{m-1}, w\}$ and $\{v_{j'}, u'\}$. If $\text{dst}_1(v_{m-1}, w)$ or $\text{dst}_p(v_m, w')$ does not hold, then G^u_j and $G^{u'}_{j'}$ are undefined for all j, j', $p \leq j \leq m-1$ and $m \leq j' \leq n$. See e.g. Figure 60.

For all j, $p \leq j \leq m-1$, and all j', $m \leq j' \leq n$, let $G^{u'}_j$ and $G^u_{j'}$ be defined analogously (if $\text{dst}_1(v_{m-1}, w')$ or $\text{dst}_p(v_m, w)$ does not hold, $G^{u'}_j$ and $G^u_{j'}$ are undefined).

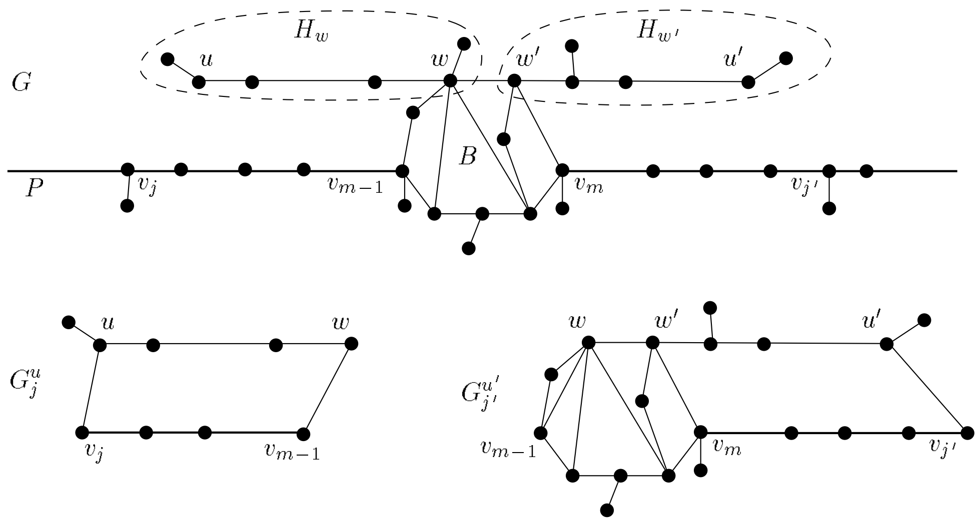

Figure 60: Example of a connecting biconnected component B with $W_B = \{w, w'\}$, and graphs G^u_j and $G^{u'}_{j'}$. Note that in $\bar{B}$, $\text{dst}_1(v_{m-1}, w)$ and $\text{dst}_p(v_m, w')$ hold (if $(\mathcal{C}, \mathcal{S})$ is a path of chordless cycles for B with $v_{m-1} \in V(C_1)$ and $v_m \in V(C_p)$), but $\text{dst}_p(v_m, w)$ and $\text{dst}_1(v_{m-1}, w')$ do not hold, which means that $G^{u'}_j$ and $G^u_{j'}$ are undefined.

Let l^u_1 be the largest value of j, $p \leq j \leq m-1$, for which there is a proper path decomposition of $G^u_j \cup \{\text{sticks of } v_{m-1}\}$ with $\{v_j, u\}$ in the leftmost node and $\{v_{m-1}, w\}$

in the rightmost node, or there is a proper path decomposition of G^u_{m-1} with $\{v_{m-1}, u\}$ in the leftmost node and $\{v_{m-1}, w\}$ in the rightmost node, and $j = 1$. If G^u_j is not defined, or there is no such j, let l^u_1 be undefined.

Let l^u_2 be the largest value of j, $p \leq j \leq m-1$, for which there is a proper path decomposition of $G^u_j \cup \{\text{sticks of } w\}$ with $\{v_j, u\}$ in the leftmost node and $\{v_{m-1}, w\}$ in the rightmost node. If G^u_j is not defined, or there is no such j, let l^u_2 be undefined.

Let $r^{u'}_1$ be the smallest value of j', $m \leq j' \leq n$, for which there is a proper path decomposition of $G^{u'}_{j'} \cup \{\text{sticks of } w\}$ with $\{v_{m-1}, w\}$ in the leftmost node and $\{v_{j'}, u'\}$ in the rightmost node. If $G^{u'}_{j'}$ is not defined, or there is no such j, let r^u_1 be undefined.

Let $r^{u'}_2$ be the smallest value of j', $m \leq j' \leq n$, for which there is a proper path decomposition of $G^{u'}_{j'} \cup \{\text{sticks of } v_{m-1}\}$ with $\{v_{m-1}, w\}$ in the leftmost node and $\{v_{j'}, u'\}$ in the rightmost node. If $G^{u'}_{j'}$ is not defined, or there is no such j, let r^u_2 be undefined.

Let $l^{u'}_1$, $l^{u'}_2$, r^u_1 and r^u_2 be defined analogously.

Let Q' be defined as follows.

$$Q' = \{(l^u_1, r^{u'}_1), (l^u_2, r^{u'}_2), (l^{u'}_1, r^u_1), (l^{u'}_1, r^u_1)\}$$

Claim 4.23. If $W_B = \{w, w'\}$, then

$$Q = \{\, (j, j') \in Q' \mid j \text{ and } j' \text{ are defined } \wedge \neg\exists_{(l,l') \in Q'} (j < l \leq l' \leq j \vee j \leq l \leq l' < j')\,\}$$

Proof. Can be shown in the same way as Claim 4.5 in Case 2 for trees. □

The computation of l^u_1 and l^u_2 can be done in $O(n^2)$ time, where $n = |V(G^u_p) \cup \{\text{ sticks of } v_{m-1} \text{ and } w\ \}|$, using $PPW2$. Analogously for $r^{u'}_1$, $r^{u'}_2$, $l^{u'}_1$, $l^{u'}_2$, r^u_1, and r^u_2.

This completes the description of the case that $W_B = \{w, w'\}$. All other cases are similar.

Case 4. $v_m \in V(P)$, $m \in \{1, q\}$, and there is a connecting biconnected component containing v_m.

This case a straightforward combination of cases 2 and 3.

This completes the description of the four cases. During the algorithm, we use the following record to store all local information for each vertex of the path.

Definition 4.20. *Let G be a three-colored partial two-path, $P = (v_1, \ldots, v_q)$ a possible nice path for G. For each m, $1 < m < q$, v_m is a record with fields nr, p and bc. The fields v_m.nr and v_m.p are as defined in Definition 4.10. The field v_m.bc has eight fields, namely ok, G, rl, lll, llr, rrl, rrr, and lr, which are defined as follows. v_m.bc.ok is a boolean which is true if and only if there is a connecting biconnected component between v_m and v_{m+1}. If v_m.bc.ok is true, then the other fields are defined as follows (let B denote the connecting biconnected component between v_m and v_{m+1}).*

- *v_m.bc.G denotes the graph G_B.*

- $v_m.bc.rl$ stores the local information for Case 3.1:

  $v_m.bc.rl$ has a field ok which is true if and only if there is a nice proper path decomposition of $v_m.bc.G$ such that v_m and v_{m+1} are in all nodes ($v_m.bc.rl.ok = b$, where b is as defined in Definition 4.16).

- $v_m.bc.lll$ and $v_m.bc.llr$ store the local information for Case 3.2:

  - $v_m.bc.lll$ has two fields: ok and v which denote the following. $v_m.bc.lll.ok$ is true if and only if j_1 as defined in Definition 4.17 is defined. If not $v_m.bc.lll.ok$, then $v_m.bc.lll.v = p$, otherwise, $v_m.bc.lll.v = j_1$.
  - $v_m.bc.llr$ has two fields: ok and v which denote the following. $v_m.bc.llr.ok$ is true if and only if j_2 as defined in Definition 4.17 is defined. If not $v_m.bc.llr.ok$, then $v_m.bc.llr.v = p$, otherwise, $v_m.bc.llr.v = j_2$.

- $v_m.bc.rrl$ and $v_m.bc.rrr$ store the local information for Case 3.3:

  - $v_m.bc.rrl$ has two fields: ok and v which denote the following. $v_m.bc.rrl.ok$ is true if and only if j_1 as defined in Definition 4.18 is defined. If not $v_m.bc.rrl.ok$, then $v_m.bc.rrl.v = n$, otherwise, $v_m.bc.rrl.v = j_1$.
  - $v_m.bc.rrr$ has two fields: ok and v which denote the following. $v_m.bc.rrr.ok$ is true if and only if j_2 as defined in Definition 4.18 is defined. If not $v_m.bc.rrr.ok$, then $v_m.bc.rrr.v = n$, otherwise, $v_m.bc.rrr.v = j_2$.

- $v_m.bc.lr$ stores the local information for Case 3.4:

  $v_m.bc.lr$ has nine fields: ok and for a, $1 \leq a \leq 4$, l_a and r_a, which denote the following. $v_m.bc.lr.ok$ is a boolean which is true if and only the set Q as defined in Definition 4.19 is non-empty. If $v_p.bc.lr.ok$ is true, then $v_m.bc.lr.l_a$ and $v_m.bc.lr.r_a$, $1 \leq a \leq 4$, are such that

  $$Q = \{\, (v_m.bc.lr.l_a, v_m.bc.lr.r_a) \mid 1 \leq a \leq 4 \,\}.$$

Furthermore, for $m = 1$ and $m = q$, v_m is a record with fields nr, p, bc, nrb and b, which are defined as follows. Fields $v_m.nr$, $v_m.p$ and $v_m.bc$ are as defined before, but $v_m.nr$ and $v_m.p$ are only defined for partial one-paths of type I, II *and* III. *$v_m.nrb$ denotes the number of non-connecting biconnected components which contain v_m. $v_m.b$ is an array of $v_m.nrb$ records with fields G, t, e, l, r and lr, which are defined as follows. For each i, $1 \leq i \leq v_m.nrb$:*

- $v_m.b[i].G$ denotes the graph G_B, where B is the ith non-connecting biconnected component which contains v_m.

- $v_m.b[i].t$ is the type of $v_m.b[i].G$, i.e. $v_m.b[i].t \in \{\mathrm{PW1}, \mathrm{PW2}\}$, where type *PW1* denotes that $v_m.b[i].G - \{v_m\}$ has pathwidth one, and type *PW2* denotes that $v_m.b[i].G - \{v_m\}$ has pathwidth two.

- *$v_m.b[i].e$ stores the ending biconnected component info of Case 2:*

  $v_m.b[i].e$ has two fields: ok and v, which denote the following. $v_m.b[i].e.ok$ is true if and only if j_1 as defined in Definition 4.15 is defined and one of the following conditions holds

  - *$v_m.b[i].t = \mathrm{PW2}$,*
  - *$v_m.nrb = 1$ and $q > 1$,*
  - *$v_m.nrb = 2$ and $q = 1$,*
  - *$q > 1$, there is no i' with $v_m.b[i'].t = \mathrm{PW2}$ and $v_m.nrb + v_m.nr \leq 2$,*
  - *$q = 1$, there is no i' with $v_m.b[i'].t = \mathrm{PW2}$ and $v_m.nrb + v_m.nr \leq 3$,*
  - *there is no i' with $v_m.b[i'].t = \mathrm{PW2}$, and $v_m.b[i].G - \{v_m\}$ has a vertex of color $c(v_m)$, or*
  - *$q > 1$, there is no i' with $v_m.b[i'].t = \mathrm{PW2}$ or for which $v_m.b[i'].G - \{v_m\}$ has a vertex of color $c(v_m)$, $v_m.nrb + v_m.nr \geq 3$, and $v_m.b[i].G$ is selected in the sense of Case 2.*
  - *$q = 1$, there is no i' with $v_m.b[i'].t = \mathrm{PW2}$ or for which $v_m.b[i'].G - \{v_m\}$ has a vertex of color $c(v_m)$, $v_m.nrb + v_m.nr \geq 4$, and $v_m.b[i].G$ is selected in the sense of Case 2.*

  If $v_1.b[i].e.ok$ is true, then $v_1.b[i].e.v = j_1$ (j_1 as defined in Definition 4.15).

- *$v_m.b[i].l$, $v_m.b[i].r$ and $v_m.b[i].lr$ store the partial one-path info of Case 2. They are defined in the same way as $v_m.p[i].l$, $v_m.p[i].r$ and $v_m.p[i].lr$, except that $v_m.b[i].l.ok$, $v_m.b[i].r.ok$ and $v_m.b[i].lr.ok$ can only be true if $v_m.b[i].t = \mathrm{PW1}$, and $v_m.b[i].nrb > 1$.*

We now show how the global information is computed. Therefore, we construct a modified version of the function Check_Nice_Path for trees. We consider three cases, namely the case that the nice path is empty, the case that the nice path consists of one vertex, and the case that the nice path consists of two or more vertices. If the nice path is empty, then G is a biconnected component with sticks, and we can check whether there is a nice proper path decomposition of G with nice path P by simply using $PPW2'$.

If the nice path consists of one vertex, then there are two ending biconnected components: one for each side. All partial one-paths of type I, II, III and IV may not have a vertex of color $c(v_1)$.

Next consider the case that the nice path consists of more than one vertex. In this case, the global information can be computed using a modified version of the for-loop of the function Check_Nice_Path for trees. We now show how the function Check_Nice_Path for trees is adapted for general partial two-paths. We use the same structure, and the same variable *in*. Only variable *out* has to be modified, since there is one case in which it does not suffice to have one pair $(out.l.l, out.l.r)$ and one pair

$(out.l.l, out.l.r)$, namely the case that $v_m.nr = 1$ and $v_m.bc.ok$ is true. However, two pairs suffice, as we will show after the algorithm.

Let $i_1,..., i_t$ denote the vertices of P, except v_1 and v_q, for which $v_{i_j}.nr > 0 \vee v_{i_j}.bc.ok$, for all j, $1 \leq j \leq t$, such that $i_1 < i_2 < \cdots < i_t$. Furthermore, let $i_0 = i_{-1} = 1$ and $i_{t+1} = i_{t+2} = q$.

Suppose the nice path is processed up to and including i_j, $0 \leq j \leq t$. Let $m = i_j$, $p = i_{j-1}$, $n = i_{j+1}$ and $nn = i_{j+2}$. The global information that is kept is defined as follows.

Definition 4.21. *The global information consists of two records in and out, which are defined as follows.*

- *in is a record with two fields ok and r, which are defined in Definition 4.11.*
- *out is a record with two fields l and r, which each have five fields: ok, l_1, l_2, r_1 and r_2. The ok field is as defined in Definition 4.11. If out.l.ok is true, then out.l.l_i and out.l.r_i, $1 \leq i \leq 2$, are such that*

$$\{ (out.l.l_i, out.l.r_i) \mid 1 \leq i \leq 2 \} = S,$$

where S is defined as follows.

S = $\{ (j, j') \mid p \leq j \leq m \wedge n \leq j' \leq nn \wedge$ there is a 'partial' nice proper path decomposition up to and including v_m and the partial one-paths connected to v_m and biconnected components containing v_m, such that it is possible that a partial one-path H' connected to v_n uses $[l, l']$, $j \leq l \leq l' \leq m$, the sticks of v_m which have color $c(v_n)$ occur on the right side of the occurrence of $\{v_m, v_n\}$, and all partial one-paths connected to v_i, $i \geq n$, except H', can use j' at least. Furthermore, there is no pair (s, s'), $j \leq s \leq m$ and $m \leq s' \leq j'$, for which this also holds, but $j < s$ or $j' < s'$. $\}$

If out.r.ok holds, then out.r.l_i and out.r.r_i, $1 \leq i \leq 2$, are such that the same condition holds, but with the sticks of v_n which have color $c(v_n)$ occurring on the right side of the occurrence of $\{v_m, v_n\}$.

We now show how variables *in* and *out* are initialized and adapted by giving a complete description of function Check_Nice_Path. In Figures 61 up to 69, a symbolic representation of all cases in the algorithm is given.

function Check_Nice_Path(P: Path): boolean;
{ pre:
  $P = (v_1,..., v_q)$ is a possible nice path for G.
  $\forall_{1 \leq m \leq q}$ ($v_m.nr$ = # partial one-paths of type I, II, III, and if $1 < m < q$, also
    of type IV connected to v_m, and
    $\forall_{1 \leq i \leq v_m.nr}$ ($v_m.p[i].H$ is partial one-path i and $v_m.p[i].t$ is type of $v_m.p[i].H$), and
      $v_m.bc.ok$ is true iff there is a connecting biconnected component between

```
        v_m and v_{m+1}, and
        if v_m.bc.ok then v_m.bc.G is graph G_B, where B is biconnected component which
        connects v_m and v_{m+1} )
  ∀_{m∈{1,q}}( v_m.nrb = # non-connecting biconnected components containing v_m, and
     ∀_{1≤i≤v_m.nrb} (v_m.b[i].G is graph G_B for ith non-connecting biconnected component B,
        and v_m.b[i].t is type of v_m.b[i].G))
}
{ output: true if there is a proper path decomposition of G
  with nice path P, false otherwise
}

  {q = 0 }
  if  q = 0
  → let B be biconnected component of G;
     if  B has vertices of state I1 or E1
        return false
     □  else
     →  use PPW2' to compute whether there is a proper path
        decomposition of G;
        return result of this computation
     fi
  fi;

  {q = 1 }
  if  q = 1
  → {v_1.nrb ≥ 2 }
     for i := 1 to v_1.nr
     → if  v_1.p[i].H has vertex of color c(v_1)
        → return false
        fi
     rof;
     for i := 1 to v_1.nrb
     → compute v_1.b[i].e
     rof;
     for i := 1 to v_m.nrb
     → if  v_1.b[i].e.ok ∧ v_1.b[i].e.v = 1
        → {this is at most four times }
           for j := 1 to v_1.nrb
           → if  j ≠ i ∧ v_1.b[j].e.ok ∧ v_1.b[j].v = 1
              → b := true
                 for l := 1 to v_1.nrb
                 → if  l ≠ j ∧ l ≠ i ∧
                          v_1.p[l].H has vertex of color c(v_1)
                    → b := false
                    fi
                 rof;
                 if  b → return true fi
              fi
           rof
        fi
```

```
    rof;
    return false
fi;

{q > 1 }
Let i_1,..., i_t denote thosa vertices of P, except v_1 and v_q, for which
v_{i_j}.nr > 0 ∨ v_{i_j}.bc.ok, for all j, 1 ≤ j ≤ t, such that i_1 < i_2 < ··· < i_t
i_0, i_{-1}, i_{t+1}, i_{t+2} := 1, 1, q, q;

{initialize in and out on false }
in.ok := false; in.r := q;
out.l.ok := false; out.l.l_1, out.l.r_1, out.l.l_2, out.l.r_2 := q, q, q, q;
out.r.ok := false; out.r.l_1, out.r.r_1, out.r.l_2, out.r.r_2 := q, q, q, q;

{handle v_1 }
if  v_1.nr + v_1.nrb = 0
→  {no ending biconnected component }
    in.ok := true; in.r := 1;
□  v_1.nr + v_1.nrb = 1 ∧ ¬v_1.bc.ok
→  {v_1.nr = 0 ∧ v_1.nrb = 1 }
    compute v_1.b[1].e
    if  v_1.b[1].e.ok
    →  {see Figure 61, part I }
        in.ok := true;
        in.r := v_1.b[1].e.v
    □  ¬v_1.b[1].e.ok
    →  return false
    fi
□  v_1.nr + v_1.nrb > 1 ∨ v_1.bc.ok
→  nr' := nr + nrb − 1; {nr' is # partial one-paths }
    for i := 1 to v_m.nrb
    →  Compute v_1.b[i].e
        if  v_1.b[i].e.ok ∧ v_1.b[i].e.v = 1
            {this can happen at most three times }
        →  {see Figure 61, part II }
            Handle biconnected components except v_1.b[i].G as partial one-paths of type IV;
            Compute local information for all partial one-paths, and for the connecting
                biconnected component if v_1.bc.ok holds;
            in'.ok := true;
            in'.r := 1;
            out'.l := out'.r := false;
            Compute in and out in same way as for i_j, 1 ≤ j ≤ t.
        fi
    rof
fi;
if  ¬in.ok ∧ ¬out.l.ok ∧ ¬out.r.ok
→  return false
fi;

{handle v_{i_j}, for all j, 1 ≤ j ≤ t }
```

```
for j := 1 to t
→ in' := in; out' := out;
  {initialize in and out }
  in.ok := false; in.r := q;
  out.l.ok := false; out.l.l_1, out.l.r_1, out.l.l_2, out.l.r_2 := q, q, q, q;
  out.r.ok := false; out.r.l_1, out.r.r_1, out.r.l_2, out.r.r_2 := q, q, q, q;
  m := i_j;
  p := i_{j-1};
  pp := i_{j-2};
  n := i_{j+1};
  nn := i_{j+2};

  Permute array v_m.p of partial one-paths such that no v_m.p[i].H, 2 < i ≤ v_m.nr,
  has a vertex of color c(v_m). If this is not possible, return false
  for i := 1 to v_m.nr
  → if  v_m.p[i].H has vertex of color c(v_m) or v_m.nr = 1
    → Compute v_m.p[i].l, v_m.p[i].r, v_m.p[i].lr, v_m.p[i].lll, v_m.p[i].llr, v_m.p[i].rrl,
         and v_m.p[i].rrr using PPW2 and PPW2'
    □ else
    → v_m.p[i].l.ok := true;
      v_m.p[i].l.v := m;
      v_m.p[i].r.ok := true;
      v_m.p[i].r.v := m;
      v_m.p[i].lr.ok := false;
      v_m.p[i].lll.ok := false;
      v_m.p[i].llr.ok := false;
      v_m.p[i].rrl.ok := false;
      v_m.p[i].rrr.ok := false;
    fi
  rof;
  if  ¬v_m.bc.ok
  → {No connecting biconnected component between v_m and v_{m+1} }
    see Check_Nice_Path for trees
  □ v_m.bc.ok
  → compute v_m.bc.lr, v_m.bc.lll, v_m.bc.llr, v_m.bc.rrl, v_m.bc.rrr and v_m.bc.rl

    {compute in }
    if  v_m.nr = 0
    → {try in' }
      if  in'.ok ∧ v_m.bc.lr.ok
      → for a := 1 to 4
        → if  v_m.bc.l_a ≥ in'.r
          → {see Figure 62, part I }
            in.ok := true;
            in.r := min{in.r, v_m.bc.lr.r_a}
          fi
        rof
      fi;
      {no need to try out' }
```

```
□   v_m.nr = 1
→   {try in' }
    if  in'.ok
    → if v_m.bc.lr.ok ∧ v_m.p[1].l.ok ∧ v_m.p[1].l.v ≥ in'.r
        → for a := 1 to 4
            → {see Figure 62, part II }
              in.ok := true;
              in.r := min{in.r, v_m.bc.lr.r_a}
            rof
        fi;
        if  v_m.bc.lll.ok ∧ v_m.bc.lll.v ≥ in'.r ∧ v_m.p[1].rrr.ok
        → {see Figure 62, part III }
          in.ok := true;
          in.r := min{in.r, v_m.p[1].rrr.v}
        fi;
        if  v_m.bc.llr.ok ∧ v_m.bc.llr.v ≥ in'.r ∧ v_m.p[1].rrl.ok
        → {see Figure 62, part IV }
          in.ok := true;
          in.r := min{in.r, v_m.p[1].rrl.v}
        fi;
    fi;

    {try out'.l }
    if  out'.l.ok
    → for b := 1 to 2
        → if  v_m.p[1].llr.ok ∧ v_m.p[1].llr.v ≥ out'.l.l_b ∧ v_m.bc.lr.ok
            → for a := 1 to 4
                → {see Figure 63, part I }
                  in.ok := true;
                  in.r := min{in.r, v_m.bc.lr.r_a}
                rof
            fi;
        rof;
    fi;
    {try out'.r }
    similar to out'.l
    {see Figure 63, part II }

□   v_m.nr > 1
→   {try in' }
    if  in'.ok
    → if v_m.bc.lr.ok
        → for i := 1 to 2
            → if  v_m.p[i].l.ok ∧ v_m.p[i].l.v ≥ in'.r ∧
                  v_m.p[3 − i].r.ok ∧ v_m.p[3 − i].r.v = m
                → for a := 1 to 4
                    → {see Figure 64, part I }
                      in.ok := true;
                      in.r := min{in.r, v_m.bc.lr.r_a}
                    rof
```

```
          fi
        rof
      fi;
      if   v_m.bc.lll.ok
      →  for i := 1 to 2
         →  if   v_m.p[i].l.ok ∧ v_m.p[i].l.v ≥ in'.r ∧
                 v_m.p[3 − i].rrr.ok ∧ v_m.p[3 − i].rrr.v ≤ n
            →  {see Figure 64, part II }
                 in.ok := true;
                 in.r := min{in.r, v_m.p[3 − i].rrr.v}
            fi
         rof
      fi;
      if   v_m.bc.llr.ok
      →  for i := 1 to 2
         →  if   v_m.p[i].l.ok ∧ v_m.p[i].l.v ≥ in'.r ∧
            v_m.p[3 − i].rrl.ok ∧ v_m.p[3 − i].rrl.v ≤ n
            →  {see Figure 64, part III }
                 in.ok := true;
                 in.r := min{in.r, v_m.p[3 − i].rrl.v}
            fi;
         rof
         fi
      fi;
fi;

{try out'.l }
if   out'.l.ok
→  for b := 1 to 2
   →  if   v_m.bc.lr.ok
      →  for i := 1 to 2
         →  if   v_m.p[i].llr.ok ∧ v_m.p[i].llr.v ≥ out'.l.l_b ∧ v_m.p[3 − i].r.ok
            →  for a := 1 to 4
               →  {see Figure 65, part I }
                    in.ok := true;
                    in.r := min{in.r, v_m.bc.lr.r_a}
               rof
            fi
         rof;
      fi;
      if   v_m.bc.lll.ok
      →  for i := 1 to 2
         →  if   v_m.p[i].llr.ok ∧ v_m.p[i].llr.v ≥ out'.l.l_b ∧
                 v_m.p[3 − i].rrr.ok ∧ v_m.p[3 − i].rrr.v ≤ n
            →  {see Figure 65, part II }
                 in.ok := true;
                 in.r := min{in.r, v_m.p[3 − i].rrr.v}
            fi
         rof;
      fi;
```

```
            if   v_m.bc.llr.ok
            →  for i := 1 to 2
                →  if   v_m.p[i].llr.ok ∧ v_m.p[i].llr.v ≥ out'.l.l_b ∧
                        v_m.p[3 − i].rrl.ok ∧ v_m.p[3 − i].rrl.v ≤ n
                   →  {see Figure 65, part III }
                      in.ok := true;
                      in.r := min{in.r, v_m.p[3 − i].rrl.v}
                   fi
               rof;
            fi
        rof
    fi;
    {try out'.r }
    similar to out'.l

{compute out }
if   v_{m+1}.nr = 0
→  {no partial one-path connected to v_m can use [j, j'], n ≤ j ≤ j' ≤ nn }
    skip
□  v_{m+1}.nr ≥ 1
→  if   v_m.nr = 0
    → {try in' }
        if   in'.ok
        →  {compute out.l }
            if   v_m.bc.rrl.ok
            →  {see Figure 66, part I }
                out.l.ok := true;
                out.l.l_1 := out.l.l_2 := in'.r;
                out.l.r_1 := out.l.r_2 := v_m.bc.rrl.v;
            fi;
            {compute out.r }
            similar {see Figure 66, part II }
        fi
        {out' does not have to be tried since v_m.nr = 0}

    □  v_m.nr = 1
    → {try in' }
        if   in'.ok
        →  {compute out.l }
            if   v_m.bc.rrl.ok ∧ v_m.p[1].l.ok ∧ v_m.p[1].l.v ≥ in'.r
            →  {see Figure 67, part I }
                out.l.ok := true;
                out.l.l_1 := m;
                out.l.r_1 := v_m.bc.rrl.v;
            fi;
            if   v_m.bc.rl.ok ∧ v_m.p[1].rrl.ok
            →  {see Figure 67, part II }
                out.l.ok := true;
                out.l.l_2 := in'.r;
                out.l.r_2 := v_m.p[1].rrl.v
```

```
        fi;
        {compute out.r }
        similar
    fi;

    {try out'.l }
    if  out'.l.ok
    →   for b := 1 to 2
        →   {compute out.l }
            if  v_m.p[1].llr.ok ∧ v_m.p[1].llr.v ≥ out'.l.l_b ∧
                v_m.bc.rrl.ok
            →   {see Figure 67, part III }
                out.l.ok := true;
                out.l.l_1 := m;
                out.l.r_1 := min{out.l.r_1, v_m.bc.rrl.v}
            fi;
            {compute out.r }
            similar
        rof
    fi;
    {try out'.r }
    similar

    {make sure out.l and out.r are as defined in Definition 4.21 }
    if  out.l.ok
    →   if  out.l.l_1 ≤ out.l.l_2 ∧ out.l.r_1 ≤ out.l.r_2
        →   out.l.l_2 := out.l.l_1;
            out.l.r_2 := out.l.r_1;
        □   out.l.l_2 ≤ out.l.l_1 ∧ out.l.r_2 ≤ out.l.r_1
        →   out.l.l_1 := out.l.l_2;
            out.l.r_1 := out.l.r_2;
        fi;
    fi;
    if  out.r.ok
    →   similar
    fi;

□   v_m.nr > 1
→   {try in' }
    if  in'.ok
        {compute out.l }
        if  v_m.bc.rrl.ok
        →   for i := 1 to 2
            →   if  v_m.p[i].l.ok ∧ v_m.p[i].l.v ≥ in'.r ∧ v_m.p[3 − i].r.ok
                →   {see Figure 68, part I }
                    out.l.ok := true;
                    out.l.l_1 := out.l.l_2 := m;
                    out.l.r_1 := out.l.r_2 := min{out.l.r_1, v_m.bc.rrl.v}
                fi;
            rof
```

```
                        fi;
                        {compute out.r }
                        similar
                        if   v_m.bc.rl.ok
                        →  for i := 1 to 2
                            →  {compute out.l }
                                if   v_m.p[i].l.ok ∧ v_m.p[i].l.v ≥ in'.r ∧ v_m.p[3 − i].rrl.ok
                                →  {see Figure 68, part II }
                                    out.l.ok := true;
                                    out.l.l_1 := out.l.l_2 := m;
                                    out.l.r_1 := out.l.r_2 := min{out.l.r_1, v_m.p[3 − i].rrl.v}
                                fi;
                                {compute out.r }
                                similar
                            rof
                        fi;
                    fi;

                    {try out'.l }
                    if   out'.l.ok
                    →  for b := 1 to 2
                        →  for i := 1 to 2
                            →  {compute out.l }
                                if   v_m.bc.rrl.ok ∧ v_m.p[i].llr.ok ∧
                                      v_m.p[i].llr.v ≥ out'.l.l_b ∧ v_m.p[3 − i].r.ok
                                →  {see Figure 68, part III }
                                    out.l.ok := true;
                                    out.l.l_1 := out.l.l_2 := m;
                                    out.l.r_1 := out.l.r_2 := min{out.l.r_1, v_m.bc.rrl.v};
                                fi;
                                if   v_m.bc.rl.ok ∧ v_m.p[i].llr.ok ∧
                                      v_m.p[i].llr.v ≥ out'.l.l_b ∧
                                      v_m.p[3 − i].rrl.ok
                                →  {see Figure 68, part IV }
                                    out.l.ok := true;
                                    out.l.l_1 := out.l.l_2 := m;
                                    out.l.r_1 := out.l.r_2 :=
                                        min{out.l.r_1, v_m.p[3 − i].rrl.v};
                                fi
                                {compute out.r }
                                similar
                            rof
                        rof
                    fi
                    {try out'.r }
                    similar
                fi
            fi
        fi;
        if   ¬in.ok ∧ ¬out.l.ok ∧ ¬out.r.ok
```

```
    → return false
    fi

rof

{handle v_q }
if  v_q.nr + v_q.nrb = 0
→  {no ending biconnected component }
    return in.ok
□  v_q.nr + v_q.nrb = 1
→  {v_q.nr = 0 ∧ v_q.nrb = 1 }
    compute v_q.b[1].e;
    {see Figure 69, part I }
    return (in.ok ∧ v_q.b[1].e.ok ∧ v_q.b[1].e.v ≥ in.r)
□  v_q.nr + v_q.nrb > 1
→  for i := 1 to v_q.nrb
    → compute v_q.b[i].e
    rof;
    nr' := nr + nrb − 1; {nr' is # partial one-paths }
    for i := 1 to v_m.nrb
    → if  v_q.b[i].e.ok ∧ v_q.b[i].e.v = q
        → {this can happen at most three times }
            add other biconnected components to array v_q.p;
            permute new array v_q.p such that no v_q.p[i].H, 1 < i ≤ v_q.nr',
            has a vertex of color c(v_q);
            if this is not possible, then return false

            compute v_q.p[1].l, v_q.p[1].lll and v_q.p[1].llr;
            {compute final result }
            {try in }
            if  in.ok ∧ v_q.p[1].l.ok ∧ v_q.p[1].l.r ≥ in.r
            → {see Figure 68, part II }
                return true
            fi;
            {try out.l }
            if  out.l.ok ∧ v_q.p[1].llr.ok
            → for b := 1 to 2
                → if  out.l.l_b ≤ v_q.p[1].llr.v
                    → {see Figure 68, part III }
                        return true
                    fi
                rof
            fi;
            {try out.r }
            if  out.r.ok ∧ v_q.p[1].lll.ok
            → similar
            fi
        fi
    rof
fi;
```

return false
end

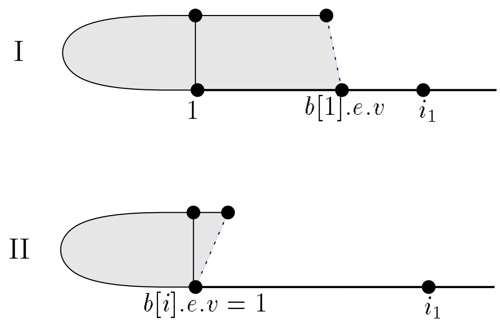

Figure 61: Cases for v_1 in the algorithm. In part I, $nrb = 1$ and $nr = 0$. In part II, $nr + nrb > 1$, and only the ending biconnected component is shown.

Lemma 4.28. *If suffices to keep track of two pairs $(out.l.l_i, out.l.r_i)$, and two pairs $(out.r.l_i, out.r.r_i)$.*

Proof. Consider the computation of the new value of $out.l$ at vertex v_m of the path. If $out.l.ok$ holds, then we want to keep track of all pairs $(l_i, r_i) \in S$, where S is as defined in Definition 4.21.

If $v_m.bc.ok$ is false, then $|S| \leq 1$, as is shown in Lemma 4.19. If $v_m.bc.ok$ is true and there $v_m.nr = 0$, then $|S| \leq 1$, because the only possible pair is $(in'.r, v_m.bc.rrl.v)$. If $v_m.nr > 1$, then $|S| \leq 1$, since for each possible pair (l_i, r_i), $l_i = m$. If $v_{nr} = 1$, $|S| \leq 2$, since there is one possible pair (l_i, r_i) with $l_i = m$, and one possible pair with $l_i = in'.r$. □

The main result of this section is as follows.

Theorem 4.2. *The algorithm given in this section computes in $O(n^2)$ time whether there is a proper path decomposition of a three-colored partial two-path G ($n = |V(G)|$).*

Proof. The correctness of the algorithm follows from previous lemmas. The total time taken by the algorithm is $O(n^2)$, since the number of candidate nice paths is constant, and for each nice path, the function Check_Nice_Path runs in $O(n^2)$ time, which can be shown in the same way as in the proof of Theorem 4.1. □

This completes the description of the algorithm to check for a given three-colored graph G whether there is a proper path decomposition of G. The algorithm can be made constructive in the sense that it returns an intervalization if there exists one in the same way as the algorithm for trees.

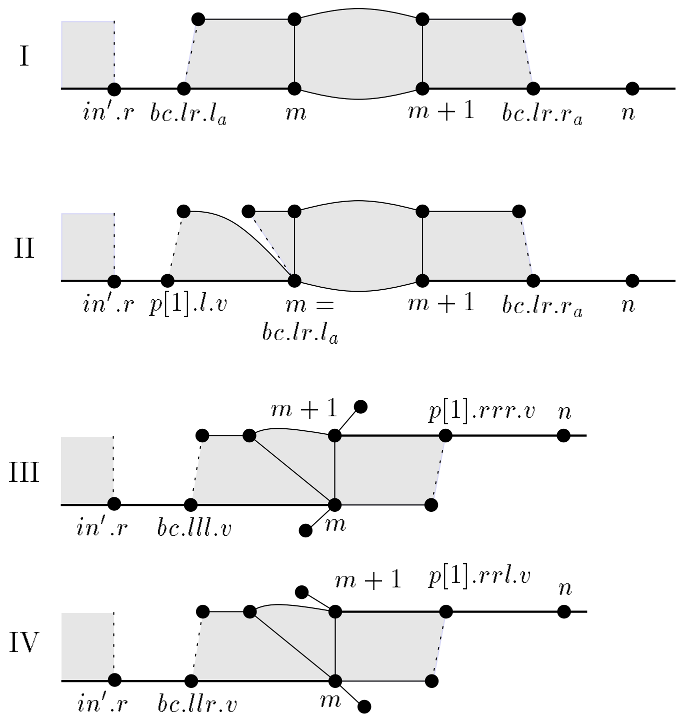

Figure 62: Cases in the algorithm in which *in* is computed, $v_m.bc.ok$ holds, $v_m.nr = 0$ (Part I) or $v_m.nr = 1$ and in' is used (Part II, III and IV).

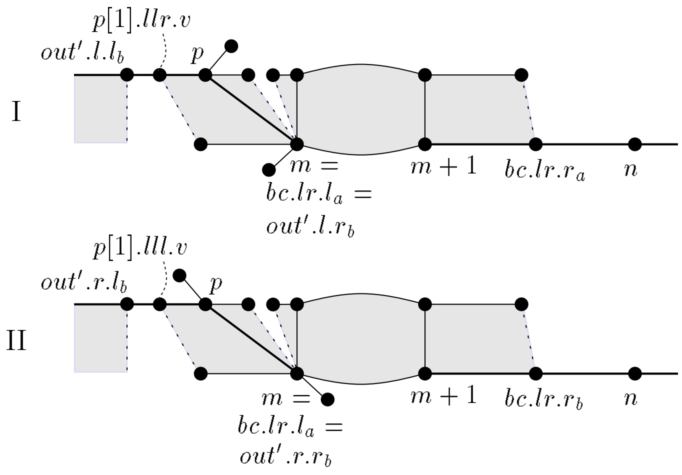

Figure 63: Cases in the algorithm in which *in* is computed, $v_m.bc.ok$ holds, $v_m.nr = 1$ and out' is used.

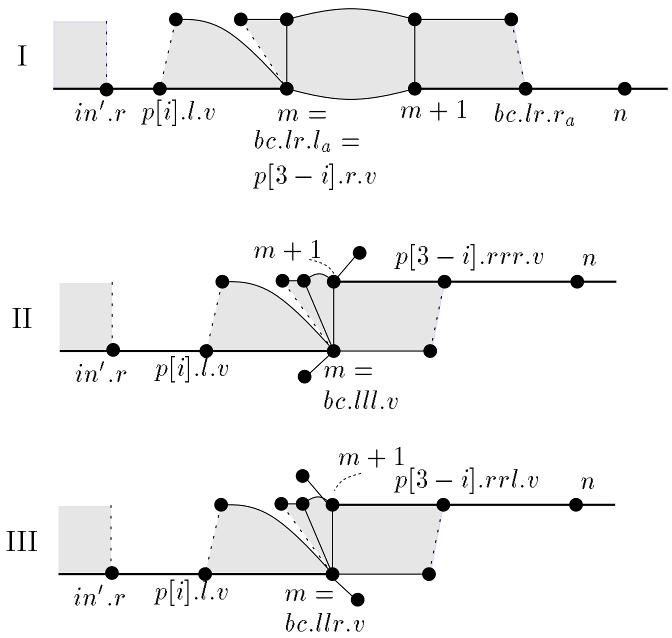

Figure 64: Cases in the algorithm in which in is computed, $v_m.bc.ok$ holds, $v_m.nr > 1$ and in' is used. In part I, $v_m.p[3-i].H$ is not drawn.

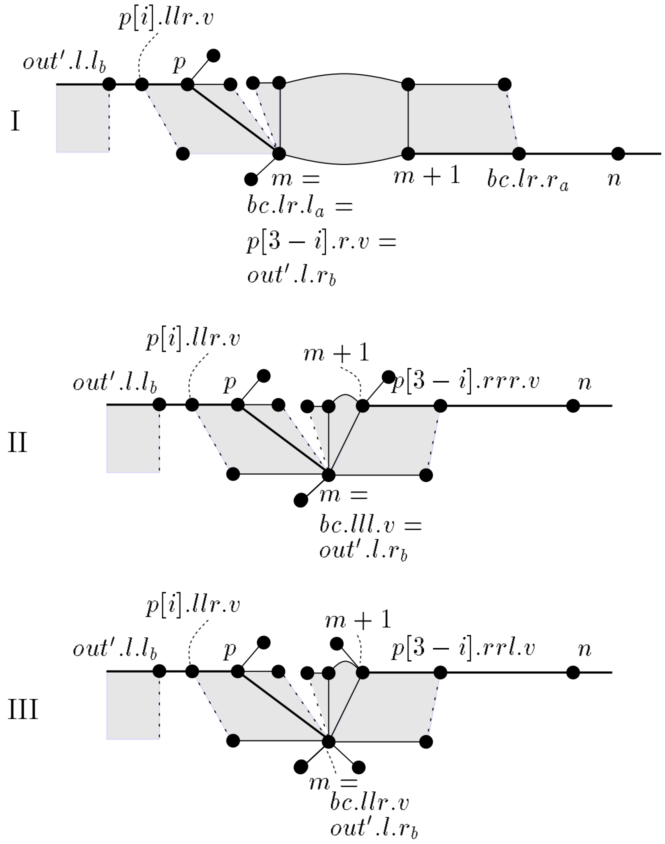

Figure 65: Cases in the algorithm in which in is computed, $v_m.bc.ok$ holds, $v_m.nr > 1$ and $out'.l$ is used. In part I, $v_m.p[3-i].H$ is not drawn.

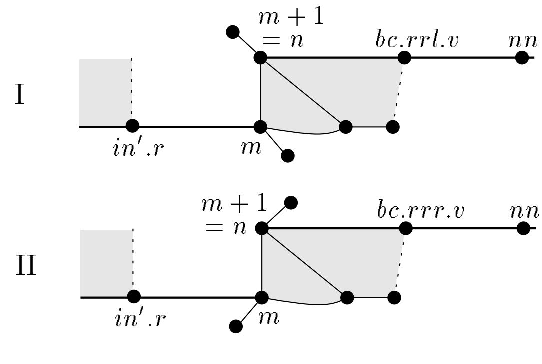

Figure 66: Cases in the algorithm in which out is computed, $v_m.bc.ok$ holds, $v_m.nr = 0$ and in' is used.

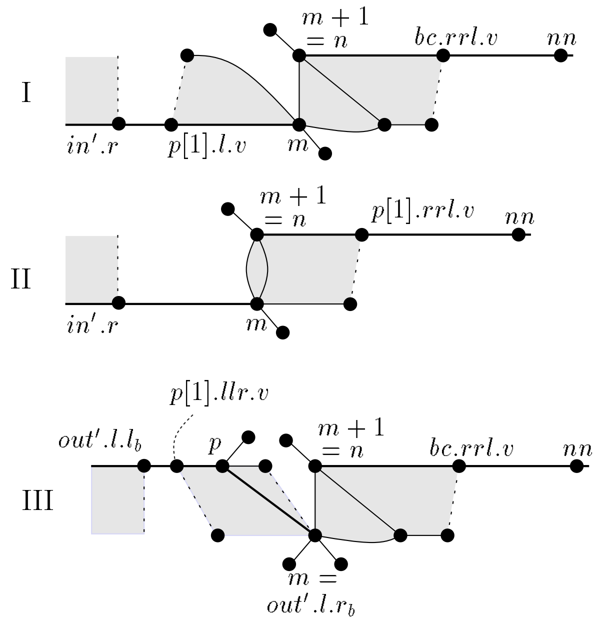

Figure 67: Cases in the algorithm in which *out* is computed, $v_m.bc.ok$ holds, and $v_m.nr = 1$.

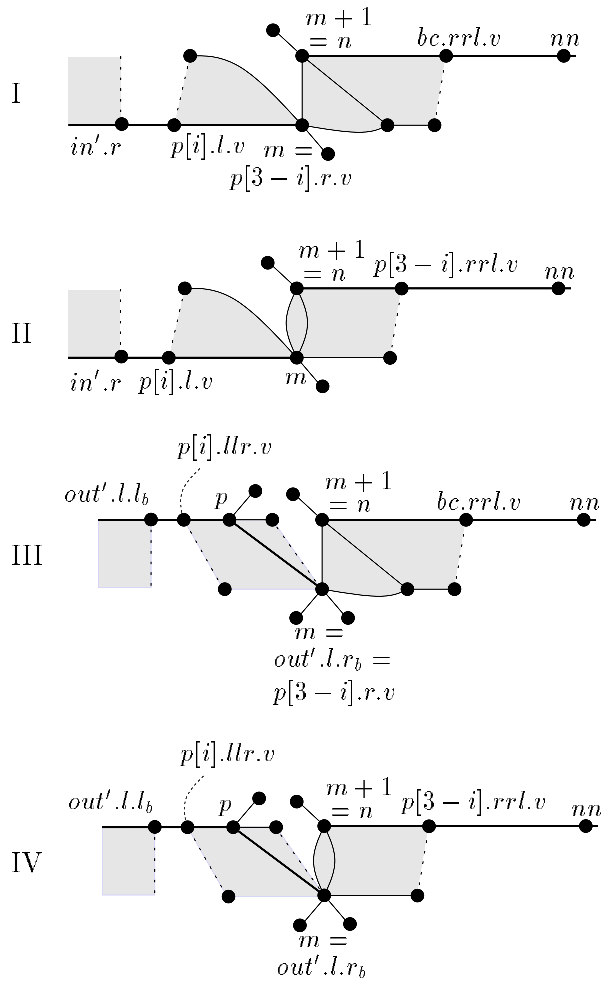

Figure 68: Cases in the algorithm in which *out* is computed, $v_m.bc.ok$ holds, and $v_m.nr > 1$. In parts I and III, $v_m.p[3-i].H$ is not drawn.

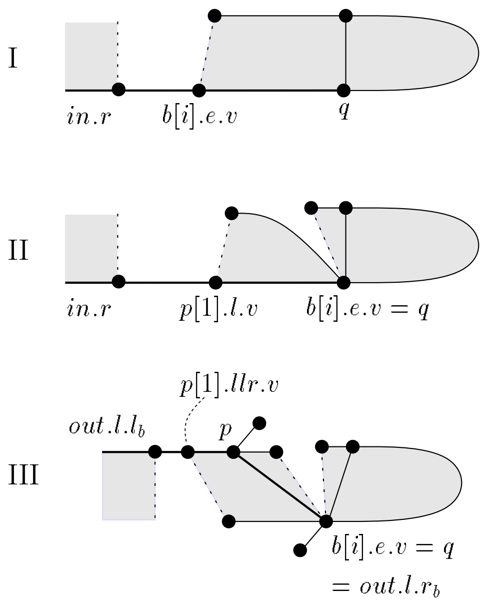

Figure 69: Cases for v_q in the algorithm.

5 Intervalizing Four-Colored Graphs

For some time, it has been an open problem whether there existed polynomial time algorithms for ICG for a constant number of colors, $k \geq 4$. Older results showed fixed parameter intractability [FHW93, BFH94], but did not resolve the question. Our NP-completeness result resolves this open problem in a negative way (assuming $P \neq NP$).

Theorem 5.1. *ICG is NP-complete for four-colored graphs.*

Proof. Clearly, ICG $\in$ NP.

To prove NP-hardness, we transform from 3-partition, which is strongly NP-complete [GJ79].

> 3-PARTITION
> **Instance:** Integers $m \in \mathbf{N}$ and $Q \in \mathbf{N}$, a sequence $s_1,..., s_{3m} \in \mathbf{N}$ such that
>
> - $\sum_{i=1}^{3m} s_i = mQ$, and
> - $\forall_{1 \leq i \leq 3m}\ \frac{1}{4}Q < a_i < \frac{1}{2}Q$.
>
> **Question:** Can the set $\{1,..., 3m\}$ be partitioned into m disjoint sets $S_1,..., S_m$ such that
> $$\forall_{1 \leq j \leq m} \sum_{i \in S_j} s_i = Q$$

Suppose input $m, Q, s_1, s_2,..., s_{3m} \in \mathbf{N}$ is given. Now, we define a graph $G = (V, E)$, which consists of the following parts (see Figure 70):

Start clique Take vertices $A = \{a_1, a_2, a_3, a_4\}$. Color vertex a_i with color i ($i = 1, 2, 3, 4$). Add edges between every two vertices in A.

End clique Take vertices $B = \{b_1, b_2, b_3, b_4\}$. Color vertex b_i with color i ($i = 1, 2, 3, 4$). Add edges between every two vertices in B.

Middle cliques Take vertices $C = \{c_{i,j} \mid 1 \leq i \leq m - 1,\ 1 \leq j \leq 3\}$. Color each vertex $c_{i,j} \in C$ with color j. Make each set $C_i = \{c_{i,1}, c_{i,2}, c_{i,3}\}$ into a clique.

Tracks Take vertices $D = \{d_{i,j} \mid 1 \leq i \leq m,\ 1 \leq j \leq 24Q\}$. Color each vertex $d_{i,j} \in D$ with color 1 if $j \bmod 3 = 1$, with 2 if $j \bmod 3 = 2$ and with 3 if $j \bmod 3 = 0$. Identify vertex a_1 with $d_{1,1}$, vertex b_3 with $d_{m,24Q}$, and, for all i, $1 \leq i \leq m-1$, identify $d_{i,24Q}$ with $c_{i,3}$, and $d_{i+1,1}$ with $c_{i,1}$. These track vertices form m paths: take edges $\{d_{i,j}, d_{i,j+1}\}$ for all i, j, $1 \leq i \leq m$, $1 \leq j \leq 24Q - 1$.

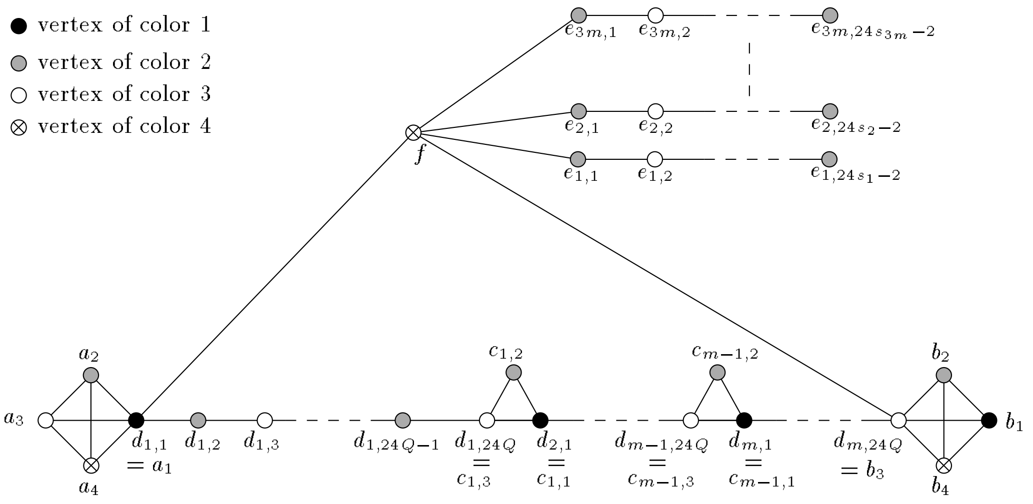

Figure 70: The constructed graph $G = (V, E)$.

Number representing paths Take vertices $E = \{e_{l,j} \mid 1 \le l \le 3m,\ 1 \le j \le 24s_l - 2\}$. Color each vertex $e_{l,j} \in E$ with color 2 if $j \bmod 3 = 1$, with color 3 if $j \bmod 3 = 2$, and with color 1 if $j \bmod 3 = 0$. For each l, the vertices $E_l = \{e_{l,j} \mid 1 \le j \le 24s_l - 2\}$ form a path: add edges $\{e_{l,j}, e_{l,j+1}\}$ for all l, j, $1 \le l \le 3m$, $1 \le j \le 24s_l - 3$.

Attachment vertex Take one vertex f. Color f with color 4. Take edges $\{f, a_1\}$ $\{f, b_3\}$, and for all l, $1 \le l \le 3m$, edge $\{f, e_{l,1}\}$.

The four-colored graph, resulting from this construction, is the graph $G = (V, E)$. Note that the transformation can be done in polynomial time in Q and m.

Claim 5.1. There exists a partition of the set $\{1,..., 3m\}$ into sets $S_1,..., S_m$ such that $\sum_{i \in S_j} s_i = Q$ for each j if and only if there is an intervalization of G.

Proof. Suppose that G is a subgraph of a properly colored interval graph. So, we have a proper path decomposition $(V_1,..., V_r)$ of G. We may assume that there are no V_i, V_{i+1} with $V_i \subseteq V_{i+1}$ or $V_{i+1} \subseteq V_i$. (Otherwise, we may omit the smaller of these two sets from the path decomposition and still have a path decomposition of G.)

Note that, by the clique containment lemma (Lemma 2.4), there exist i_0 with $V_{i_0} = A$, and i_1 with $V_{i_1} = B$. Without loss of generality suppose $i_0 < i_1$. If $i_0 \ne 1$, then there exists a $v \in V_{i_0-1}$ with $v \notin A$. Note that such a vertex v has a path to a vertex in B that avoids A. It follows that V_{i_0} must contain a vertex from this path, but this will yield a color conflict with a vertex in A, contradiction. So, $i_0 = 1$. A similar argument shows that $i_1 = r$.

Also, from Lemma 2.4 it follows that for each i, $1 \le i \le m - 1$, there is a j_i, $2 \le j_i \le r - 1$ with $C_i \subseteq V_{j_i}$. We must have $j_1 < j_2 < j_3 < \cdots < j_{m-1}$, otherwise a color conflict will arise between a track vertex and a vertex in a set C_i. Write $j_0 = 1$,

$j_m = r$. As there is a path from $d_{1,1}$ to $d_{m,24Q}$ in G that does not contain vertices with color 4 or vertices in E, it follows that each set V_i contains at least one vertex in $C \cup D$ with color 1, 2 or 3.

For each i, $1 \leq i \leq m$, call the interval $[j_{i-1}+1, j_i - 1]$ the *ith valley.* Each vertex $d_{i,j}$ must be in one or more successive nodes V_α with α in the ith valley. It can not be in another valley, since that gives a color conflict. Note that there are exactly $8Q$ vertices $d_{i,j}$ (for fixed i) with color 2. For a 2-colored vertex $d_{i,j}$, we call the interval $\{\alpha \mid d_{i,j} \in V_\alpha\}$ a 2-range. Note that all 2-ranges are disjoint, otherwise we have a color conflict. So, in each valley, we have exactly $8Q$ 2-ranges.

For each l, $1 \leq l \leq 3m$, look at the vertices E_l. Note that all vertices in E_l must be contained in nodes V_α with all α's in the same valley. Otherwise, the path induced by E_l will cross a middle clique, and we have a color conflict between a vertex in E_l and a vertex in C. Write $S_i = \{l \mid$ vertices in E_l are in sets V_α with α in the ith valley$\}$. We show that $S_1,\ldots, S_m$ is a partition of $\{1,\ldots, 3m\}$ such that for each j, $\sum_{i \in S_j} s_i = Q$.

For each edge $\{e_{l,j}, e_{l,j+1}\}$ with $e_{l,j}$ of color 3 (and hence, $e_{l,j+1}$ has color 1), there must be a node α with $\{e_{l,j}, e_{l,j+1}\} \subseteq V_\alpha$. α must be in a 2-range, as otherwise V_α contains a 1-colored or 3-colored vertex from $C \cup D$, and we have a color conflict. If there exists an α with $\{e_{l,j}, e_{l,j+1}, d_{i,j'}\} \subseteq V_\alpha$, with $d_{i,j'}$ of color 2, then we say that the 2-range of $d_{i,j'}$ contains the 1-3-E-edge $\{e_{l,j}, e_{l,j+1}\}$.

Claim 5.2. No 2-range contains two or more 1-3-E-edges.

Proof. Suppose $\{e_{l_1,j_1}, e_{l_1,j_1+1}\}$ and $\{e_{l_2,j_2}, e_{l_2,j_2+1}\}$ are distinct 1-3-E-edges, and there is a $d_{i,j'}$ such that $\{e_{l_1,j_1}, e_{l_1,j_1+1}, d_{i,j'}\} \subseteq V_\alpha$, $\{e_{l_2,j_2}, e_{l_2,j_2+1}, d_{i,j'}\} \subseteq V_\beta$. Suppose w.l.o.g. that $\alpha < \beta$. Note that both $v = e_{l_1,j_1}$ and $w = e_{l_1,j_1+1}$ are adjacent to a 2-colored vertex. Let $[\gamma, \delta]$ be the 2-range of $d_{i,j'}$. Note that $\gamma \leq \alpha < \beta \leq \delta$. If $V_{\gamma-1}$ contains a 1-colored vertex from $C \cup D$, then consider the 1-colored vertex w. It cannot belong to $V_{\gamma-1}$ and it cannot belong to V_β. So, if $w \in V_\epsilon$, then $\gamma \leq \epsilon \leq \delta$. Hence, there cannot be a set V_ϵ that contains w and its 2-colored neighbor e_{l_1,j_1+2}, contradiction. If $V_{\gamma-1}$ does not contain a 1-colored vertex from $C \cup D$, then it contains a 3-colored vertex from $C \cup D$, and by considering v and using a similar argument, also a contradiction arises. □

Let $1 \leq i \leq m$. Suppose $S_i = \{l_1, l_2, \ldots, l_t\}$. Note that $E_{l_1} \cup \cdots \cup E_{l_t}$ induces $8s_{l_1} - 1 + 8s_{l_2} - 1 + \cdots + 8s_{l_t} - 1$ 1-3-E-edges. As there are $8Q$ 2-ranges in a valley, we must have

$$8(s_{l_1} + s_{l_2} + \cdots s_{l_t}) - t \leq 8Q$$

By noting that each $s_l \geq Q/4 + 1/4$, it follows that $8(Q/4 + 1/4)t - t \leq 8Q$, so $t \leq 3$, and that hence also, by integrality,

$$8(s_{l_1} + s_{l_2} + \cdots s_{l_t}) \leq 8Q$$

So, we have a partition of $\{1,\ldots, 3m\}$ into sets $S_1,\ldots, S_m$, such that for all $j, 1 \leq j \leq m$, $\sum_{i \in S_j} s_i \leq Q$. As $\sum_{j=1}^{m} \sum_{i \in S_j} s_i = mQ$, it follows that for all j, $1 \leq j \leq m$, $\sum_{i \in S_j} s_i = Q$.

Now, suppose $S_1, S_2, \ldots, S_m$ is a partition of $\{1, \ldots, 3m\}$, such that for all j, $1 \leq j \leq m$, $\sum_{i \in S_j} s_i = Q$. We will give a path decomposition $(V_1, \ldots, V_r)$ of $G = (V, E)$, such that no V_i contains two vertices of the same color. We leave most of the easy verification that the given path decomposition fulfills the requirements to the reader.

Take $t = 48Q$, $r = mt + 1$.

Take $V_1 = A$, $V_r = B$.

For each vertex $c_{i,j} \in C$, put $c_{i,j}$ in set V_{ti+1}.

For each vertex $d_{i,j} \in D$, put $d_{i,j}$ in sets $V_{t(i-1)+2j-1}$, $V_{t(i-1)+2j}$, and $V_{t(i-1)+2j+1}$. (Identified vertices are just put in every set, indicated by their 'different names'; one easily observes that these are consecutive sets.)

For each i, $1 \leq i \leq m$, suppose $S_i = \{l_1, l_2, l_3\}$. Put vertex $e_{l_1,1}$ in set $V_{t(i-1)+2}$. For all j, $2 \leq j \leq 24s_{l_1} - 2$, put vertex $e_{l_1,j}$ in sets $V_{t(i-1)+2j-2}$, $V_{t(i-1)+2j-1}$, $V_{t(i-1)+2j}$. For all j, $1 \leq j \leq 24s_{l_2} - 2$, put vertex $e_{l_2,j}$ in sets $V_{t(i-1)+48s_{l_1}+2j-2}$, $V_{t(i-1)+48s_{l_1}+2j-1}$, $V_{t(i-1)+48s_{l_1}+2j}$. For all j, $1 \leq j \leq 24s_{l_3} - 2$, put vertex $e_{l_3,j}$ in sets $V_{t(i-1)+48s_{l_1}+48s_{l_2}+2j-2}$, $V_{t(i-1)+48s_{l_1}+48s_{l_2}+2j-1}$, $V_{t(i-1)+48s_{l_1}+48s_{l_2}+2j}$.

Finally, put f in all sets $V_2, \ldots, V_{r-1}$.

A straightforward, but somewhat tedious verification shows that the resulting path decomposition is indeed a path decomposition of G, and that no set V_i contains two different vertices with the same color.

□

As 3-partition is strongly NP-complete and our transformation is polynomial in Q and m, the claimed theorem now follows. □

Note that we even proved a slightly stronger result:

Corollary 5.1. *ICG is NP-complete for four-colored graphs G, with the property that there is one color that is only given to three vertices of G.*

6 Conclusions

In this paper, we have given an $O(n^2)$ time algorithm to determine whether we can add edges to a given three-colored graph such that it becomes a properly colored interval graph. The algorithm can be modified such that it outputs an intervalization, if existing, and still uses quadratic time. To get a faster algorithm for the problem considered in this paper might well be a hard problem. It seems that even the simplest cases, e.g., when G is a simple cycle, need $O(n^2)$ time to resolve, and might well already capture the main difficulties for speed-up.

We have shown that this problem is NP-complete for four or more colors. We feel however that the graphs, arising in the reduction of this proof, will not be typical for the type of colored graphs, arising in the sequence reconstruction application. It may well be that special cases of ICG, which capture characteristics of the application data, have efficient algorithms. Further research could perhaps give new meaningful results here.

References

[BFH94] Hans L. Bodlaender, Michael R. Fellows, and Michael Hallett. Beyond *NP*-completeness for problems of bounded width: Hardness for the W hierarchy. In *Proceedings of the 26th Annual Symposium on Theory of Computing*, pages 449–458, New York, 1994. ACM Press.

[BFW92] Hans L. Bodlaender, Michael R. Fellows, and Tandy J. Warnow. Two strikes against perfect phylogeny. In *Proceedings 19th International Colloquium on Automata, Languages and Programming*, pages 273–283, Berlin, 1992. Springer Verlag, Lecture Notes in Computer Science, vol. 623.

[BK93] Hans L. Bodlaender and Ton Kloks. A simple linear time algorithm for triangulating three-colored graphs. *J. Algorithms*, 15:160–172, 1993.

[BM93] Hans Leo Bodlaender and Rolf H. Möhring. The pathwidth and treewidth of cographs. *SIAM J. Disc. Meth.*, 6:181–188, 1993.

[EST94] John A. Ellis, Ivan Hal Sudborough, and J. Turner. The vertex separation and search number of a graph. *Information and Computation*, 113:50–79, 1994.

[FHW93] Michael R. Fellows, Michael T. Halett, and H. Todd Wareham. DNA physical mapping: Three ways difficult (extended abstract). In Thomas Lengauer, editor, *Proceedings 1st Annual European Symposium on Algorithms ESA'93*, pages 157–168. Springer Verlag, Lecture Notes in Computer Science, vol. 726, 1993.

[GJ79] Michael R. Garey and David S. Johnson. *Computers and Intractability, A Guide to the Theory of NP-Completeness.* W.H. Freeman and Company, New York, 1979.

[IS93] R. Idury and A. Schaffer. Triangulating three-colored graphs in linear time and linear space. *SIAM J. Disc. Meth.*, 2:289–293, 1993.

[KS93] Haim Kaplan and Ron Shamir. Pathwidth, bandwidth and completion problems to proper interval graphs with small cliques. Technical Report 285/93, Inst. for Computer Science, Tel Aviv University, Tel Aviv, Israel, 1993. To appear in SIAM J. Comput.

[KST94] Haim Kaplan, Ron Shamir, and Robert Endre Tarjan. Tractability of parameterized completion problems on chordal and interval graphs: Minimum fill-in and physical mapping. In *Proceedings of the 35th annual symposium on Foundations of Computer Science (FOCS)*, pages 780–791. IEEE Computer Science Press, 1994.

[KW92] Sampath Kannan and Tandy Warnow. Triangulating 3-colored graphs. *SIAM J. Disc. Meth.*, 5:249–258, 1992.

[Möh90] Rolf H. Möhring. Graph problems related to gate matrix layout and PLA folding. In E. Mayr, H. Noltemeier, and M. Sysło, editors, *Computational Graph Theory, Comuting Suppl. 7*, pages 17–51. Springer Verlag, 1990.

[MWW94] F. R. McMorris, Tandy Warnow, and T. Wimer. Triangulating vertex-colored graphs. *SIAM J. Disc. Meth.*, 7(2):296–306, 1994.

[NON94] Shin-Ichi Nakano, Takashi Oguma, and Tako Nishizeki. A linear time algorithm for c-triangulating three-colored graphs. *Trans. Institute of Electronics, Information and Communication, Eng., A.*, 377-A(3):543–546, 1994. In Japanese.