Universiteit Utrecht

Master Thesis
ICA-3117375

Kernelization Rules for Special Treewidth and Spaghetti Treewidth

Author:
V.J.C. Kreuzen
vjckreuzen@gmail.com

Supervisor:
Dr. Hans L. Bodlaender
H.L.Bodlaender@uu.nl

Secondary Supervisor:
Dr. Stefan Kratsch
S.Kratsch@uu.nl

Department of Information and Computing Sciences,
Faculty of Science
Utrecht University, Netherlands

June 26, 2012

Abstract

Using the framework of kernelization we study whether efficient preprocessing schemes for the SPECIAL TREEWIDTH problem can give provable bounds on the size of the processed instances. In this paper it is shown that SPECIAL TREEWIDTH has a kernel with $\mathcal{O}(\ell^3)$ vertices, where ℓ denotes the size of a vertex cover. This implies that given an instance (G, k) of SPECIAL TREEWIDTH we can efficiently reduce its size to $\mathcal{O}((\ell^*)^3)$ vertices, where ℓ is the size of a vertex cover in G. Next we provide a characterization of the special-partial 2-trees, the class of graphs bounded by Special Treewidth 2, using the notion of *mambas* and *Paths of Cycles*.

We also introduce a new cousin of TREEWIDTH and SPECIAL TREEWIDTH, the SPAGHETTI TREEWIDTH problem. It is shown that SPAGHETTI TREEWIDTH also has a kernel with $\mathcal{O}(\ell^3)$ vertices, where l denotes the size of a vertex cover.

Contents

Chapter 1

Introduction

Treewidth is a well-studied graph parameter, with many theoretical and practical applications. In treewidth we use tree-decompositions, which can be defined informally as a tree-like representation of an original graph, which is used to solve certain problems on the original graph more easily/faster. The tree-decomposition is a tree of so called *bags*, with each bag holding a number of vertices from the original graph. The maximum size of the bags determines the treewidth of the graph. Many NP-hard combinatorial problems on graphs are solvable in polynomial time when restricted to graphs of bounded (preferably low) treewidth.

A new notion of treewidth, *Special Treewidth*, was introduced by Courcelle [8, 1]. SPECIAL TREEWIDTH is a graph complexity measure between pathwidth and treewidth and is defined based on the operations that define clique-width. Courcelle devised the notion of special treewidth to aid in the model-checking problem for monadic second-order logic. For graphs of bounded treewidth, the automata used for the verification of monadic second-order properties expressed with edge set quantifications have exponential size in the bound of treewidth. Courcelle noted that the exponential blow-up did not occur if path-decompositions were used instead of tree-decompositions. This was the prime motivation to introduce special treewidth, a graph complexity measure for which the basic automata are no more difficult to construct or specify than for graphs of bounded clique-width.

An even newer notion of treewidth, *Spaghetti Treewidth*, will be introduced in this work, which is a complexity measure between special treewidth and treewidth. Spaghetti treewidth is akin to treewidth, with the exception that vertices are required to be in bags which form a (undirected) *subpath* in the decomposition rather than a *subtree*. The introduction of this new graph complexity measure came from the fact that it is the "logical missing measure" between treewidth, which requires vertices to be in bags which form a subtree in the decomposition, and special treewidth, which requires vertices to be in bags which form a directed subpath in the decomposition. In this work the decision problems related to these width parameters are studied, which given a graph G and integer k ask whether the special/spaghetti treewidth of G is at most k. For precise definitions, see Chapter 2.

In the field of *parameterized complexity* [11], a theoretical analysis of the potential of preprocessing for TREEWIDTH has been performed in [3], studying whether there are efficient preprocessing procedures whose effectiveness can be proven, and what the resulting size bounds look like. These studies have been made possible by using the concept of *kernelization* [12], a relatively young subfield of algorithm design and analysis. Further research on parameterized complexity in the field of monadic second-order logic has been performed by Courcelle, Downey and Fellows [9]. The concept of kernelization is used in this paper as well, to study whether there are efficient preprocessing procedures for SPECIAL TREEWIDTH and SPAGHETTI TREEWIDTH whose effectiveness can be proven, and what the resulting size bounds look like.

Definition 1 (Bodlaender et al. [3])**.** *Let $Q \in \Sigma^* \times \mathbb{N}$ be a parameterized problem. A* kernelization algorithm *Q (or* kernel*) is a polynomial-time algorithm which given an instance $(x, k) \in \Sigma^* \times \mathbb{N}$ of Q, computes an equivalent instance (x', k') whose size is bounded by a function $f(k)$ depending only on the chosen parameter, i.e., $|x'|, k' \leq f(k)$.*

It is unlikely that there is a polynomial-time algorithm that reduces the size of an instance (G, k) of Treewidth to a polynomial in the desired treewidth k [2]. The same arguments also apply to Special Treewidth. We therefore turn to other parameters (i.e., the vertex cover number of the input graph), and determine whether an input of Special Treewidth can be efficiently shrunk to a size which is polynomial in such a parameter. The parameterized problem that is considered fits the following template, where $\mathcal{F}$ is a class of graphs:

Special Treewidth parameterized by a modulator to $\mathcal{F}$
Instance: A graph $G = (V, E)$, a positive integer k, and a set $S \subseteq V$ such that $G \backslash S \in F$.
Parameter: $\ell := |S|$.
Question: Special Treewidth of $(G) \leq k$?

The set S is a *modulator* to the class $\mathcal{F}$.

Since it is unlikely that there is a polynomial-time algorithm that reduces the size of an instance (G, k) of Special Treewidth to a polynomial in the desired treewidth k [2], it is also very unlikely that such a polynomial-time algorithm exists for Spaghetti Treewidth. We therefore again define the following template where $\mathcal{F}$ is a class of graphs:

Spaghetti Treewidth parameterized by a modulator to $\mathcal{G}$
Instance: A graph $G = (V, E)$, a positive integer k, and a set $S \subseteq V$ such that $G \backslash S \in F$.
Parameter: $\ell := |S|$.
Question: Spaghetti Treewidth of $(G) \leq k$?

and let the parameterized problem for Spaghetti Treewidth that is considered fit this template. The set S is a *modulator* to the class $\mathcal{G}$.

This work. In this paper we introduce a kernel for Special Treewidth parameterized by the size of a vertex cover of G, resulting in the problem Special Treewidth parameterized by a vertex cover (which fits into the given template when using $\mathcal{F}$ as the class of edgeless graphs). It is proven that this problem admits a polynomial kernel with $\mathcal{O}(\ell^3)$ vertices. Since we can first compute a 2-approximation for the minimum vertex cover and then feed this to the kernelization algorithm, this implies that an instance (G, k) of Special Treewidth on a graph with a minimum vertex cover of size ℓ^* can be shrunk in polynomial-time into an instance with $\mathcal{O}((\ell^*)^3)$ vertices. We can do this even if we are not given a minimum vertex cover in the input. Next we a characterization for the Special Treewidth problem on sp-partial 2-trees, and we present a set of forbidden minors. We also introduce the notion of Spaghetti Treewidth, and show that we can acquire a polynomial kernel (similar to the kernel for Special Treewidth) for Spaghetti Treewidth with $\mathcal{O}(\ell^3)$ vertices.

Organization of the paper. After this introduction, preliminary results are given in Section 2. In Chapter 3, we show that Special Treewidth parameterized by a vertex cover has a kernel with $\mathcal{O}(\ell^3)$ vertices. To do so, we introduce a number of 'safe' reduction rules, that are variants of rules from the existing treewidth kernel, and some new rules that guarantee an upper bound on the number of simplicial vertices parameterized by the size of the vertex cover.

In Section 4 we present a characterization of the set of sp-partial 2-trees, and introduce a set of *forbidden minors* [13, 14] for this class of graphs.

In Sections 5 and 5.4, we show that Spaghetti Treewidth parameterized by a vertex cover also has a kernel with $\mathcal{O}(\ell^3)$ vertices. To do so, we again introduce a number of 'safe' reduction rules, five of which are the same as the rules we used for Special Treewidth parameterized by a vertex cover and one new rule, which together guarantee an upper bound on the number of simplicial vertices parameterized by the size of the vertex cover.

Some final remarks are made in Section 6.

Chapter 2

Preliminaries

In this work, all graphs are finite, simple and undirected (we will show below that the proofs hold for directed graphs as well). We denote graphs as $G = (V, E)$ with V the vertices and E the edges in G, and we assume that $V \neq \emptyset$ and $E \neq \emptyset$. Since our graphs are undirected, edges are denoted by unordered sets of size two.
The open neighbourhood of a vertex $v \in V$ is the set of all neighbours of v and is denoted by $N_G(v)$, and its closed neighbourhood is the set of all neighbours of v including v itself, and is denoted by $N_G[v]$. Let $\delta_G(v)$ denote the degree δ of a vertex v in G. A vertex v is simplicial in a graph G if $N_G(v)$ is a clique.

Definition 2. *A* tree-decomposition *of a graph G is a pair* $(\{X_i | i \in I\}, T = (I, F))$ *with* $\{X_i | i \in I\}$ *a family of subsets of* V*, and* T *a tree such that:*

1. $\bigcup_{i \in I} X_i = V$
2. *For all* $\{v, w\} \in E$*, there is an* $i \in I$ *with* $v, w \in X_i$.
3. *For each* $v \in V$*, the set* $I_v = \{i \in I | v \in X_i\}$ *induces a subtree in* T.

The kernelization algorithms presented in this work consist of a number of reduction rules. In each case, the input to the rule is a graph $G = (V, E)$, an integer k, and a deletion set $S \subseteq V$ such that $G \backslash S$ is a member of the relevant graph class $\mathcal{F}$, and the output is an instance $(G' = (V', E'), k', S')$. A rule is said to be safe for Special Treewidth if for all inputs (G, k, S) which satisfy $G \backslash S \in \mathcal{F}$ we have $\mathbf{sptw}(G) \leq k \Leftrightarrow \mathbf{sptw}(G') \leq k'$ and $G' \backslash S' \in \mathcal{F}$. A rule is said to be safe for Spaghetti Treewidth if for all inputs (G, k, S) which satisfy $G \backslash S \in \mathcal{G}$ we have $\mathbf{spghtw}(G) \leq k \Leftrightarrow \mathbf{spghtw}(G') \leq k'$ and $G' \backslash S' \in \mathcal{G}$ We will sometimes say that the algorithm answers **Yes** or **No**; this should be interpreted as outputting a constant-size **Yes** or **No** instance of the problem at hand, i.e., a clique on three vertices with $k = 2$, respectively the same clique with $k = 1$.

In this work, we also provide a rule for a preprocessing setting for each kernelization rule. These preprocessing rules can be used to preprocess graphs before running algorithms to discover the treewidth. The key difference between the kernelization rules and these preprocessing rules is the fact that in the preprocessing setting, there is no target treewidth k which can be used in the rules. The preprocessing rules circumvent this by using lower and upper bounds. We use low_G and $upper_G$ to denote the lower and upper bounds on the special/spaghetti treewidth of G respectively. Upper bounds can easily be acquired through use of the GreedyDegree heuristic [5, 6].

Definition 3. *A* chordal graph *is a graph where each of its cycles of 4 or more vertices has a* chord*. A chord is an edge which connects two vertices on a cycle which are not adjacent on the cycle. Chordal graphs are also know as* Triangulated graphs*, and we call the process of turning a graph G in a chordal graph G′* triangulation*.*

2.1 Special Treewidth

In [8], Courcelle introduced a new notion of treewidth, special treewidth. Special treewidth differs from treewidth in the sense that bags in a tree-decomposition T containing a vertex v need to form a subtree in T, whereas bags in a special tree-decomposition T' containing a vertex v need to form a directed subpath in T'.

Definition 4 (Courcelle [8])**.** *A* special tree-decomposition *of a graph G is a pair $(\{X_i|i \in I\}, T = (I, F))$ with $\{X_i|i \in I\}$ a family of subsets of V, and T a rooted and directed tree such that:*

1. $\bigcup_{i \in I} X_i = V$
2. *For all $\{v, w\} \in E$, there is an $i \in I$ with $v, w \in X_i$.*
3. *For each $v \in V$, the set $I_v = \{i \in I | v \in X_i\}$ induces a directed (rooted) path in T.*

Condition 3) characterizes special tree-decompositions. The sets X_i are called the bags of the special tree-decomposition. The width of a special tree-decomposition $(\{X_i|i \in I\}, T = (I, F))$ is $max_{i \in I}|X_i| - 1$, and the special treewidth of G is denoted by the minimum over all special tree-decompositions of G. Let $\mathbf{sptw}(G)$ be the special treewidth of G.

We define the set of *sp-partial k-trees* to be the set of graphs of special treewidth at most k.

Proposition 1 (Courcelle [1])**.** *For each k, the class of sp-partial k-trees is closed under the following transformations:*

1. *Removal of vertices and edges,*
2. *Reversals of edge directions,*
3. *Addition and removal of loops incident with existing vertices,*
4. *Addition of edges parallel to existing edges,*
5. *Smoothing vertices of degree 2.*

Smoothing a vertex of degree 2 means contracting any one of its two incident edges.

Since the class of sp-partial k-trees is closed under 4) and 2), we know that we can get from any undirected graph $G = (V, E)$ to a directed graph $G' = (V, E')$ where $E' \subseteq E$ while retaining the same special treewidth.

Proposition 2 (Courcelle [8])**.** *The special treewidth of a graph is the maximal special treewidth of its connected components. It is at most one plus the maximal special treewidth of its biconnected components. This upper bound is tight.*

We define a *bridge* (or: *cut-edge*) in a graph to be an edge whose deletion increases the number of connected components, i.e. an edge in a graph G is a bridge if and only if it is not contained in any cycle in G.

Let P and P' be directed paths in a special tree-decomposition $(\{X_i|i \in I\}, T = (I, F))$. We introduce three new structures for sets of bags $P \cup P'$ in T namely the Join, Split and Prior.

1. $P \cup P'$ is a Join **iff** $P \cap P' \neq \emptyset$ i.e. paths P and P' have at least 1 bag in common.
2. $P \cup P'$ is a Split **iff** $P \cap P' = \emptyset$ and there does not exist a directed path P'' in T such that $P \subset P''$ and $P' \subset P''$ i.e. P and P' do not have any bags in common and there is no directed path P'' with P and P' on that path.
3. $P \cup P'$ is a Prior **iff** $P \cap P' = \emptyset$ and there exists a directed path P'' in T such that $P \subset P''$ and $P' \subset P''$ i.e. P and P' are on the same path P'' but do not have any bags in common

An illustration of the different occurrences of the three structures can be found in Figure 2.1.

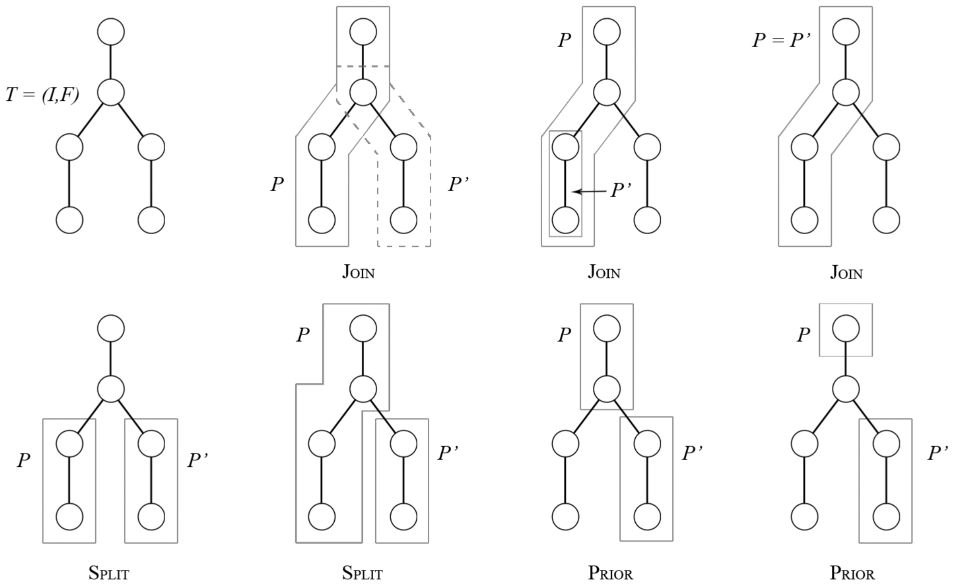

Figure 2.1: Examples of the Join, Split and Prior.

2.2 Spaghetti Treewidth

We define a new cousin of the Treewidth-family called *Spaghetti Treewidth*. Spaghetti Treewidth behaves almost the same as Special Treewidth, only it allows for bags containing the same vertex to form an *un*directed path in the spaghetti tree-decomposition, whereas a special tree-decomposition only permits a directed path of bags. We define spaghetti treewidth with the notion of a *spaghetti tree-decomposition*.

Definition 5. *A* spaghetti tree-decomposition *of a graph G is a pair* $(\{X_i | i \in I\}, T = (I, F))$ *with* $\{X_i | i \in I\}$ *a family of subsets of* V*, and* T *a rooted and directed tree such that:*

1. $\bigcup_{i \in I} X_i = V$
2. *For all* $\{v, w\} \in E$*, there is an* $i \in I$ *with* $v, w \in X_i$*.*
3. *For each* $v \in V$ *, the set* $I_v = \{i \in I | v \in X_i\}$ *induces a simple path in* T*.*

Condition 3) characterizes the spaghetti tree-decomposition. The width of a decomposition T is the the maximal cardinality minus 1 of a bag, i.e. of a set I_x.

Proposition 3. *The spaghetti treewidth of a graph is the minimal width of a spaghetti tree-decomposition of this graph. There are linear-time algorithms for converting a graph G into a spaghetti tree-decomposition of width k and vice-versa.*

The name *spaghetti treewidth* is chosen since it gives a nice intuition of how a spaghetti tree-decomposition looks like. Each vertex v in a graph G forms an undirected path of bags, like a spaghetti string. When we pull the spaghetti out of the pan, some strings stick together completely and some strings stick together for some part.

We define the set of *spgh-partial k-trees* to be the set of graphs of spaghetti treewidth at most k. It should be easy to see that from Proposition 1 we get the following proposition for spaghetti treewidth;

Proposition 4. *For each k, the class of* spgh-partial k-trees *is closed under the following transformations:*

1. *Removal of vertices and edges,*

2. *Reversals of edge directions,*

3. *Addition and removal of loops incident with existing vertices,*

4. *Addition of edges parallel to existing edges,*

5. *Smoothing vertices of degree 2.*

From the fact that each vertex in a clique must be in the same bag as its clique-neighbours we get the following proposition.

Proposition 5. *For the Spaghetti Treewidth of a clique C we have that* $\mathbf{spghtw}(C) = |C| - 1$

In Section 12, we look further into a kernelization for SPAGHETTI TREEWIDTH.

In the next chapter, the kernelization for SPECIAL TREEWIDTH PARAMETERIZED BY A VERTEX COVER (i.e., parameterized by a modulator to an independent set) is presented. The kernelization focuses mostly on simplicial vertices, and reducing the size and occurrences of these simplicial vertices. We first give a set of 6 reduction rules, and then prove that with these rules we have a kernel of size $\mathcal{O}(l^3)$.

Chapter 3

Kernelization Rules for Special Treewidth

In this chapter, a kernelization for Special Treewidth parameterized by a Vertex Cover (i.e., parameterized by a modulator to an independent set) is presented. The kernelization focuses mostly on simplicial vertices and reducing the size and occurrences of these simplicial vertices. A set of 6 reduction rules is presented, and it is proven that with these rules a kernel of size $\mathcal{O}(\ell^3)$ can be made.

3.1 Trivial rules

The Islet Rule

We specify a rule which deals with vertices of degree 0.

Rule 1 (Islet Rule). *If v is a vertex of degree 0 then remove v.*

Proof. It is clear that $S' = S$ is a vertex cover of $G' = G \backslash \{v\}$.
Removing a vertex does not increase the special treewidth of the graph [8].

Now let $G' = (V', E')$ be the reduced graph acquired from removing vertex v, with $V' = V \backslash \{v\}$ and $E' = E$. From Proposition 2 we know that the special treewidth of a graph is equal to the maximum treewidth of its disjoint connected components. Thus we have that $\mathbf{sptw}(G) = max(\mathbf{sptw}(G'), \mathbf{sptw}(\{v\}))$. Since $\delta_G(v) = 0$ we have that $\mathbf{sptw}(\{v\}) = 0$ (The special tree-decomposition of v consists of one bag with one vertex and thus has $\mathbf{sptw} = 0$) and thus $\mathbf{sptw}(G) = max(\mathbf{sptw}(G'), 0)$, and since $E' \neq \emptyset$ (E' is a non-empty set of edges) this implies that there are at least two connected vertices which have to be in the same bag in any special tree-decomposition of G' which implies that $\mathbf{sptw}(G') \geq 1$, and thus $\mathbf{sptw}(G) = \mathbf{sptw}(G')$. □

An illustration of the application of the rule can be found in Figure 3.1.

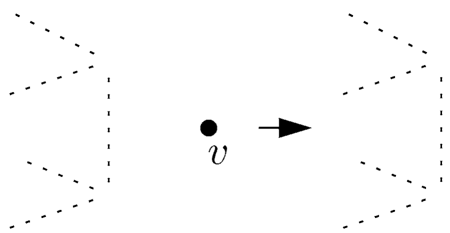

Figure 3.1: Application of the Islet Rule.

Next we also define a corresponding preprocessing rule for vertices of degree 0.

Reduction Rule I (Islet Rule). *Let v be a vertex of degree 0. Remove v from V.*

Proof. Disjoint vertices of degree 0 do not contribute to the special treewidth of a graph. Thus it is safe to remove v from G. □

Trivial Decision

We specify a rule which deals with parameters k which are greater than the size of our vertex cover.

Rule 2 (Trivial Decision). *If $k \geq |S|$, then answer* **Yes**.

Proof. This rule is safe since the special treewidth of G is at most $|S|$. We show this as follows: For each $v \in V \backslash S$, make a bag with vertex set $S \cup \{v\}$, and connect these bags such that they form a directed path. This gives a special tree decomposition of G of width $|S|$. □

The Twig Rule

We specify a rule which deals with vertices of degree 1.

Rule 3 (Twig Rule). *If v is a vertex of degree 1 then remove v. If $v \in S$, then let $S' := S\backslash\{v\}$, else let $S' := S$.*

Proof. It is clear that S' is a vertex cover of G'. Removing a vertex does not increase the special treewidth of the graph [8].

Let w be the neighbour of v. Now let $G' = (V', E')$ be the reduced graph acquired from removing vertex v, with $V' = V\backslash\{v\}$ and $E' = E\backslash\{v, w\}$.

Now let $(\{X_i | i \in I'\}, T' = (I', F'))$ be a special tree-decomposition of G'. Suppose $\mathbf{sptw}(G') = h$ with minimal special tree-decomposition T'. We can transform T' into T with special width h as follows:

Select the bag X_w from T' such that $w \in X_w$ and $|\{\{w, y\} \in F | w \in X_y\}| \leq 1$, i.e. $w \notin children(X_w)$. Make a new bag $X_v = \{v, w\}$ and add X_v as a child of X_w to T' to get T. T is now a special tree-decomposition of G with $\mathbf{sptw}(G) \geq 1$.

Since X_w was the last bag on the directed path of bags in T' containing vertex w, X_v must now be the last bag on the directed path of bags in T containing vertex w. Since we look only at graphs with special treewidth ≥ 1 and the size of our bag $|X_v| = 2$, we know that adding X_v to T' will not increase its special width, and thus this rule is safe. □

An illustration of the application of the rule can be found in Figure 3.2.

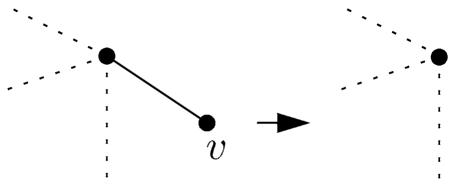

Figure 3.2: Application of the Twig Rule.

Next we also define a corresponding preprocessing rule for vertices of degree 1.

Reduction Rule III (Twig Rule). *Let v be a vertex of degree 1. Remove v from V.*
Set $low_G = max(low_G, 1)$. This is redundant: $\mathbf{sptw} \geq 1$ *if $E \neq \emptyset$*

Proof. A graph consisting of vertices of degree ≤ 1 only has bags of size ≤ 2 i.e. trees/forests. The special treewidth of such trees is 1. Thus vertices of degree 1 do not contribute to graphs of special treewidth ≥ 2 and thus we can safely remove v and set the lower bound of the graph to $max(low_G, 1)$. □

3.2 Simple rules for simplicial vertices

The Duplicate Simplicial Vertex Rule

We specify a rule which deals with duplicate simplicial vertices. Removing simplicial vertices is a well known and often used preprocessing and kernelization rule for TREEWIDTH; see [3, 7].

Rule 4 (Duplicate Simplicial Vertex Rule). *Let v and w be two simplicial vertices with $N_G(v) = N_G(w)$. If $v \neq w$ then remove w. If $w \in S$ then let $S' := S \backslash \{w\}$. Else let $S' := S$.*

Proof. It is clear that S' is a vertex cover of G'.
Removing a vertex does not increase the special treewidth of the graph [8].

Let $G' = (V', E')$ be the reduced graph acquired from removing vertex w, with $V' = V \backslash \{w\}$ and $E' = E \backslash \bigcup_{a \in N_G(w)} \{w, a\}$.

Now let $(\{X_i | i \in I'\}, T' = (I', F'))$ be a special tree-decomposition of G'. Suppose $\mathbf{sptw}(G') = h$ with minimal special tree-decomposition T'. We can transform T' into T with special width h as follows:

Select the bag X_v from T' such that $v \in X_v$ and $N_G(v) \in X_v$. This bag must exist since $N_G[v]$ is a clique and thus must exist in at least 1 bag. Since v is simplicial and T' is minimal, we know that v need only be in this bag X_v. Make a new bag $X_w = (X_v \backslash \{v\}) \cup \{w\}$. Clearly we have that $|X_v| = |X_w|$. Now add X_w as a child of $parent(X_v)$ to T' and let X_v be the child of X_w in T to get T. Since v was only contained in bag X_v we do not break any directed paths by adding bag X_w in this way to T'. T is now a special tree-decomposition of G with $\mathbf{sptw}(G) > 1$. □

An illustration of the application of the rule can be found in Figure 3.3.

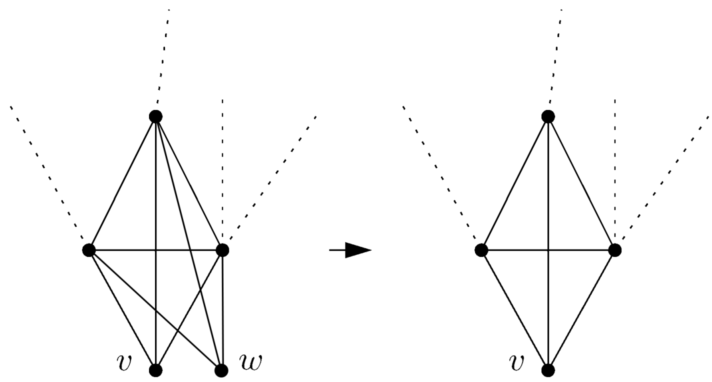

Figure 3.3: Application of the Duplicate Simplicial Vertex Rule.

Next we also define a corresponding preprocessing rule for duplicate simplicial vertices.

Reduction Rule IV (Duplicate Simplicial Vertex Rule). *Let v and w be two simplicial vertices with $N_G(v) = N_G(w)$. If $v \neq w$ then remove w.*

Proof. The proof is the same as above. If v and w are simplicial on the same clique, then the special treewidth does not change by deleting one of the two simplicial vertices. □

Singular Simplicial Vertices of degree$\leq k$

Next we look at singular simplicial vertices of degree $\leq k$. A singular simplicial vertex is a simplicial vertex $v \in V$ for which there is no vertex $w \in V$ for which we have $w \neq v$ and $N_G(v) = N_G(w)$.

Using a counterexample, we show that we cannot safely remove singular simplicial vertices from a graph G when looking at bounded special treewidth. Let $G = (V, E)$ be a 3-sun(S_3) as in Figure 3.4.

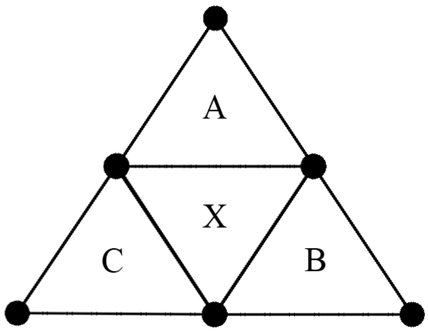

Figure 3.4: Counterexample for removing simplicial vertices in special treewidth

Let A, B, C be the three sets containing the vertices of the three outer cliques and let X be the set containing the vertices of the inner clique. Since $|A| = |B| = |C| = |X| = 3$ we have that the special treewidth of G is at least 2.

Let $v \in A$ be the simplicial vertex of G which we will try to remove. Now let $(\{X_i | i \in I'\}, T' = (I', F'))$ be the special tree-decomposition of graph $G' = (V', E')$ where $G' = G \backslash \{v\}$. We can construct T' as follows:

Let $I' = \{B, C, X\}$ and let $F' = \{\{B, X\}, \{X, C\}\}$. Now T' is a special tree-decomposition of special width 2 rooted at B.

When constructing a special tree-decomposition $(\{X_i | i \in I\}, T = (I, F))$, the root of tree T has to be either X, the bag containing the center clique, or A, B or C, the bags containing the outer cliques. In case of the first, we cannot add bags A, B and C to T such that for each vertex $x \in V$ the bags containing x form a directed path in the graph. In case of the latter, we also cannot add X and the other two remaining bags to T such that for each vertex $x \in V$ the bags containing x form a directed path in the graph. Hence we can conclude that T has to contain at least 1 bag of size ≥ 4 and thus the special width of T is 3.

Thus, since $\mathbf{sptw}(G') = 2$ and $\mathbf{sptw}(G) = 3$, we can conclude that we cannot safely remove singular simplicial vertices from a graph G when looking at bounded special treewidth.

We now give two definitions which we will use to lift the counterexample to a higher level.

Definition 6 (Special Treewidth Defining Bag)**.** *A special treewidth defining bag (sptw-defining bag) of a special tree-decomposition* $(\{X_i | i \in I\}, T = (I, F))$ *of* $G = (V, E)$*, is a bag* X_v *such that* $|X_v| = \mathbf{sptw}(G) + 1$.

Definition 7 (Uniquely Special Treewidth Defining Bag)**.** *A uniquely special treewidth defining bag (uniquely sptw-defining bag) of a special tree-decomposition* $(\{X_i | i \in I\}, T = (I, F))$ *of* $G = (V, E)$*, is a bag* X_u *such that* $\forall_{i \in I \backslash \{u\}} |X_i| < |X_u|$.

The counter example for removing singular simplicial vertices can be easily lifted to show that removing a singular simplicial vertex v from a graph G when looking at bounded special treewidth is unsafe, when v is in the uniquely sptw-defining bag X_v in the minimal special tree-decomposition T with special treewidth h. When removing this vertex v, for the special treewidth h' of T' we will have that $h \geq h'$. Since altering the graph is only safe when $max\{\mathbf{sptw}(G), low_G\} = max\{\mathbf{sptw}(G'), low'_G\}$ holds, we have that in this case $low'_G = |X_v|$ must hold. However, since X_v is the uniquely sptw-defining bag, we know that $|X_v| = \mathbf{sptw}(G) + 1$. Thus we have that $low'_G = \mathbf{sptw}(G) + 1$ must hold. Thus we first have to know the special treewidth of G before we can safely remove singular simplicial vertices from G to get G'. This implies that we cannot safely remove singular simplicial vertices.

The High Degree Simplicial Vertex Rule

Next we define another trivial rule which handles simplicial vertices of degree $> k$. This rule is not necessary to achieve our kernelization of Special Treewidth parameterized by a Vertex Cover, but it is an elegant rule which could save some extra work when dealing with simplicial vertices.

Rule 4b (High Degree Simplicial Vertex Rule). *If v is a simplicial vertex of degree $> k$ then answer* **No**.

Proof. If v is simplicial with degree $> k$, then there is a clique $N_G[v]$ in G of size at least $k+1$. This implies that the special treewidth is at least $k+1$ and thus we can safely answer **No**. □

Next we also define a corresponding preprocessing rule for simplicial vertices of degree $> low_G$.

Reduction Rule IVb (High Degree Simplicial Vertex Rule). *Let v be a simplicial vertex of degree $> low_G$ Then $low_G = max\{low_G, \delta_G(v)\}$.*

Proof. Clearly, a simplicial vertex v of degree $\delta_G(v)$ implies a clique of size $\geq \delta_G(v)$ and thus implies a special treewidth $\geq \delta_G(v) - 1$. Thus we can safely remove v and set the lower bound of G to $\delta_G(v) - 1$. □

3.3 Complex rules for removing simplicial vertices

The Common Neighbours Improvement Rule

We specify a rule which deals with pairs of vertices with at least $k+1$ common neighbours. Adding edges between vertices with many common neighbours is a well known and often used preprocessing and kernelization rule for Treewidth; see [3].

Although adding edges when trying to reduce instance sizes may seem counterproductive at first, it can be easily seen that by adding edges between vertices with many common neighbours, we might introduce new cliques into our graph, which in turn could make certain vertices simplicial, allowing for the use of Rules 4 and 6.

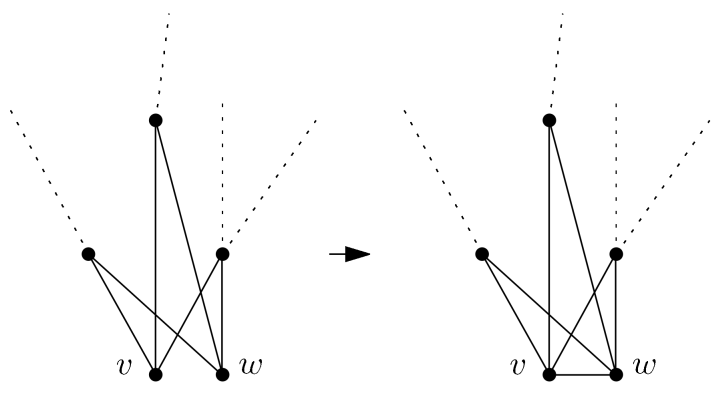

Figure 3.5: Application of the Common Neighbours Improvement Rule.

Rule 5 (Common Neighbours Improvement Rule). *Suppose that $\{v, w\} \notin E$ and that $v \in S$ or $w \in S$. If $|N_G(v) \cap N_G(w)| \geq k+1$, then add edge $\{v, w\}$ to E. Let $S' := S$.*

Proof. It is clear that S' is a vertex cover of G'.
Clearly, adding the edge $\{v, w\}$ does not decrease the special treewidth of the graph, since in a minimal special tree-decomposition T of G, v and w could already have been in the same bag.

Let $G' = (V', E')$ be the reduced graph acquired from adding the edge $\{v, w\}$, with $V' = V$ and

$E' = E \cup \{v, w\}$, and let Q denote the set containing the common neighbours of v and w, i.e. $Q = N_G(v) \cap N_G(w)$.

Now let $(\{X_i | i \in I\}, T = (I, F))$ be a special tree-decomposition of G. We now make a special tree-decomposition T' for G' of the same width as T. In T, there must be a directed path $\{P_v = \{X_{v1}, \ldots, X_{vn}\} | \forall_i v \in X_{vi}\}$ which contains Q, and there must be a directed path $\left\{P_w = \{X_{w1}, \ldots, X_{wn}\} | \forall_j w \in X_{wj}\right\}$ which contains Q. According to the definition of special treewidth, $P_v \cup P_w$ has to be either a Join or a Prior (Since a Split would imply that Q would either be in two disjoint paths or would be in a non-directed path. Both situations would break the special treewidth rules).

If $P_v \cup P_w$ is a Join, then there is at least 1 bag $X_j | \{v, w\} \subseteq X_j$. Thus adding the edge $\{v, w\}$ to E to get G' would result in a special tree-decomposition $T' = T$. If $P_v \cup P_w$ is a Prior, then there must be a directed path P_Q of size ≥ 0 in between P_v and P_w, and in particular there must be a bag X_{vn} at the tail of P_v and a bag X_{w1} at the head of P_w, with $Q \cup \{v\} \subseteq X_{vn}$ and $Q \cup \{w\} \subseteq X_{w1}$. Clearly, $|X_{vn}| \geq k + 2$ and $|X_{w1}| \geq k + 2$, and thus we would have special treewidth $\geq k + 1$ and thus answer **No** for both $\mathbf{sptw}(G) \leq k$ and $\mathbf{sptw}(G') \leq k$ (since adding edges does not decrease the special treewidth).

Thus we can safely add edge $\{v, w\}$ to G to get G', where we have that $\mathbf{sptw}(G) = \mathbf{sptw}(G') \geq 2$ □

An illustration of the application of the rule can be found in Figure 3.5.

Next we also define a corresponding preprocessing rule for pairs of vertices with at least $upper_G + 1$ common neighbours.

Reduction Rule V (Common Neighbours Improvement Rule). *Suppose that $\{v, w\} \notin E$. If $|N_G(v) \cap N_G(w)| \geq upper_G + 1$, then add edge $\{v, w\}$ to E.*

Proof. The proof goes as the proof for the kernelization rule. $P_v \cup P_w$ is either a Join, in which case v and w are already in a bag in the minimal decomposition and thus adding the edge does not change the minimum decomposition, or $P_v \cup P_w$ is a Prior, in which case there is at least a bag in P_v and a bag in P_w with size $\geq upper_G + 2$. This would imply that the special treewidth of G' is $\geq upper_G + 1$. However, since $upper_G$ denotes the upper bound for our graph, we know that there exists a special tree-decomposition of width $\leq upper_G$. Thus we know that we have that $P_v \cup P_w$ has to be a Join for the special tree-decomposition to have width $\leq upper_G$ and thus we can safely add the edge. □

The Simplicial Vertex Partition Rule

We specify a rule which deals with singular simplicial vertices of degree $\leq k$. A singular simplicial vertex is a simplicial vertex $v \in V$ for which there is no vertex $w \in V$ for which we have $w \neq v$ and $N_G(v) = N_G(w)$.

This rule, combined with Rules 2 and 4, removes all simplicial vertices v with $\delta_G(v) > 2$.

Rule 6 (Simplicial Vertex Partition Rule). *If v is a simplicial vertex with degree $3 \leq \delta_G(v) \leq k$, then for each pair of vertices $\{w, x\} \subset N_G(v)$, make a new vertex incident to both w and x. Then remove v. If $v \in S$ then let $S' := S \backslash \{v\} \cup \{z\}$ where $z \in N_G(v), z \notin S$, else let $S' := S$.*

Proof. Since v is simplicial, either $v \in S$ or $N_G(v) \in S$. If $N_G(v) \in S$, then splitting v into vertices incident to vertices in $N_G(v)$ will not affect the vertex cover S; each newly created edge is incident to a vertex in S. If $v \in S$, then there is at most 1 vertex in $N_G(v)$ which is not in S; $N_G[v]$ is a clique, and thus to cover all edges in the clique at least all vertices but one have to be in the vertex cover. Let z be the vertex which is not in the vertex cover. Now if we add z to the vertex cover, we have that $N_G(v) \in S$ and the reasoning above tells us that we have a vertex cover when splitting v. Thus S' is a vertex cover of G'.

Removing a vertex does not increase the special treewidth of the graph [8].

Let $G' = (V', E')$ be the reduced graph acquired from doing the following: let $\delta_G(v) = q$ be the degree of v. Since v is simplicial, we have a clique of size q. Now replace v with the vertex set $N = \{v_1, \ldots, v_{\frac{1}{2}n \cdot (n-1)}\}$, and assign to each pair $\{x, y\} \subset N_G(v)$ a vertex $v_i \in N$ such that $\{v_i, x\} \in E'$ and $\{v_i, y\} \in E'$.

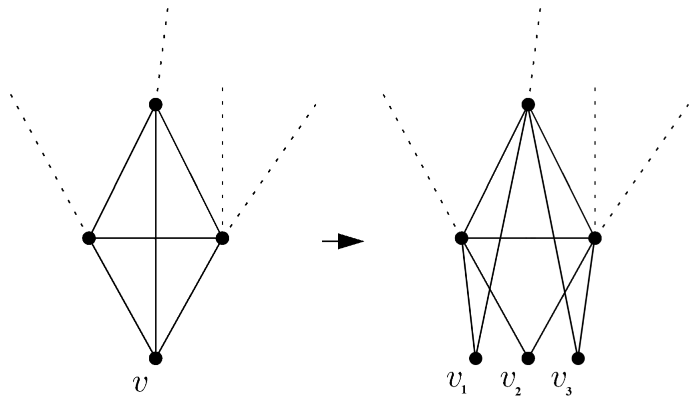

Figure 3.6: Application of the Simplicial Vertex Partition Rule.

Now let $(\{X_i | i \in I\}, T = (I, F))$ be a special tree-decomposition of G. We show that $\mathbf{sptw}(G') \leq \mathbf{sptw}(G)$ as follows: Let T be the special tree-decomposition of G with special treewidth h, and v in only one bag X_i. This is a reasonable assumption; since v is simplicial it has to be in a bag together with $N_G(v)$, and it does not need to be in any other bag. Thus take $\{X_i | N_G[v] \subseteq X_i\}$. We can now split bag X_i such that we get a directed path of bags $P\{X_1, \ldots, X_{\frac{1}{2}n \cdot (n-1)}\}$ such that for each $w \in N$ there is exactly 1 bag $\{X_w \in P | X_w = \{w\} \cup (X_i \backslash \{v\})\}$. Now replace X_i in T with P to get T'. Clearly, we now have a special tree-decomposition T' of G' of width h and thus we can conclude that $\mathbf{sptw}(G') \leq \mathbf{sptw}(G)$.

We now show that $\mathbf{sptw}(G') \geq \mathbf{sptw}(G)$ as follows: Let T be the minimal special tree-decomposition of G with special treewidth h, and v in only one bag X_i. This is a reasonable assumption; since v is simplicial it has to be in a bag together with $N_G(v)$, and it does not need to be in any other bag. Thus take $\{X_i | N_G[v] \subseteq X_i\}$. Since T is minimal, we know that $X_i = N_G[v]$. Now go from T to T' as follows: Replace $X_i \in T$ by $\{X_q | X_q = X_i \backslash \{v\}\}$. Clearly, we now have that $|X_i| = |X_q| + 1$. We will now try to get T' from T by adding all vertices $x \in N$ to the bags in T, such that each x is in a bag with its neighbours and $\mathbf{sptw}(G') < \mathbf{sptw}(G)$.

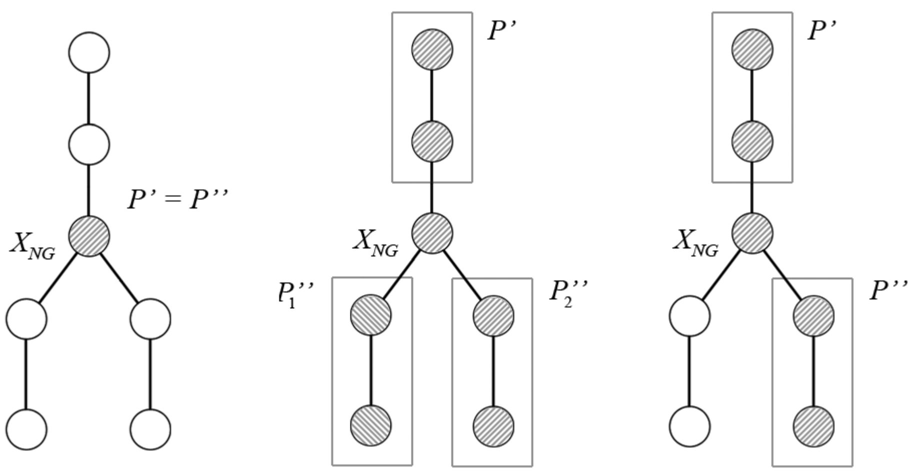

Figure 3.7: Illustration of the proof for Rule 06.

We first look at the case where X_q has multiple children-paths P'' which contain vertices $\in N_G(v)$. Let $Y = N_G(v) = Y_1 \cup Y_2$ with $Y_1 \cap Y_2 = \emptyset$ and $Y_1 \neq \emptyset, Y_2 \neq \emptyset$. Let $N_G(v) = \{y_1, \ldots, y_r\}$

and let $\{s_{ij} \in V' | \{s_{ij}, y_i\} \in E, \{s_{ij}, y_j\} \in E\}$. First look at the case that there are multiple paths P'' below X_q. Let P_1'' be the path below X_q which contains vertices from set Y_1 but not from Y_2, and let P_2'' be the path below X_q which contains vertices from set Y_2 but not from Y_1. Now for each vertex $\{s_{ij} \in V' | y_i \in Y_1, y_j \in Y_2\}$ we know that $\{s_{ij}, y_i, y_j\}$ must be on path P', since it cannot be on paths P_1'' or P_2'' (Since neither of these paths contain both y_i and y_j), nor in bag X_q (since that would mean that we would have $\mathbf{sptw}(G') \geq \mathbf{sptw}(G)$). Since for every vertex $y_i \in Y_1$ there is at least 1 vertex $y_j \in Y_2$, and for every vertex $y_j \in Y_2$ there is at least 1 vertex $y_i \in Y_1$, we have that every vertex $y \in Y$ must be on path P'. Since P' must now contain $Y = N_G(v)$, we know that the lowest bag on P' has to contain $N_G(v)$. Thus, the lowest bag $X_{P'}$ on P' which contains any s_{ij} must also contain $N_G(v)$. Since $s_{ij} \not\subseteq N_G(v)$, we know that $|X_{P'}| \geq |N_G(v)| + 1$ and thus $\mathbf{sptw}(G') \geq \mathbf{sptw}(G)$ if $N_G(v)$ is partitioned to be in multiple children-paths of X_q.

We now look at the case where X_q has only 1 single child-path P'' which contains vertices $\in N_G(v)$. Since we have that $N_G(v) = X_q$, we know that every vertex $x \in N$ will have to be in a bag on the directed path through X_q (since it will have to be in bags together with vertices from $N_G(v)$). Let P be this path through X_q where $P = P' \cup X_q \cup P''$ and P' and P'' the directed paths in T neighbouring X_q, with P' the parent path and P'' the child-path.

Now let $X_{P''}$ be the highest bag on path P'' with $Y \backslash Y_1 \subseteq X_{P''}$ and let $X_{P'}$ be the lowest bag on path P' with $Y \backslash Y_2 \subseteq X_{P'}$. Clearly, since all vertices S_{ij} must be on P we have again that $Y = N_G(v) = Y_1 \cup Y_2$ with $Y_1 \cap Y_2 = \emptyset$. Thus every vertex s_{ij} such that $y_i \in Y \backslash Y_1$ and $y_j \in Y_1$ must be on P' and every vertex s_{ij} such that $y_i \in Y \backslash Y_2$ and $y_j \in Y_2$ must be on P''. Since $Y_1 \cap Y_2 = \emptyset$ and $Y \subseteq P' \cup P''$, we have that $Y \backslash Y_1 = Y_2$ and $Y \backslash Y_2 = Y_1$ and thus we have that every vertex $y \in Y$ must be on path P'. Since P' must contain $N_G(v)$, we know that the lowest bag on P' has to contain $N_G(v)$. Thus, the lowest bag $X_{P'}$ on P' which contains any s_{ij} must also contain $N_G(v)$. Since $\{s_{ij} \not\subseteq N_G(v)$, we know that $|X_{P'}| \geq |N_G(v)| + 1$ and thus $\mathbf{sptw}(G') \geq \mathbf{sptw}(G)$ if $N_G(v)$ is partitioned to be in multiple children-paths of X_q.

Thus we can conclude that $\mathbf{sptw}(G') = \mathbf{sptw}(G)$ and thus this rule is safe. □

An illustration of the application of the rule can be found in Figure 3.6.

Next we also define a corresponding preprocessing rule for the rule.

Reduction Rule VI (Simplicial Vertex Partition Rule). *If v is a simplicial vertex with degree $3 \leq \delta_G(v) \leq low_G$, then for each pair of vertices $\{w, x\} \subset N_G(v)$, make a new vertex incident to w and x. Then remove v.*

Proof. The proof is the same as above. If v is simplicial with $\delta_G(v) \leq low_G$, then the special treewidth remains bounded by low_G if v is replaced with vertices of degree 2 on each pair of vertices in $N_G(v)$, and the special treewidth does not increase as shown above. □

3.4 A Kernel for Special Treewidth

We can now reason that the exhaustive application of Rule 1 through 6 (i.e., until we answer **Yes** or **No** or no application of one of these rules is possible) gives a polynomial kernel for Special Treewidth parameterized by a vertex cover. It is clear that this reduction can be performed in polynomial time (it is easy to do it in time $O(|V| \cdot |E|)$). Let S denote a vertex cover of G.

Theorem 1. Special Treewidth parameterized by a vertex cover *has a kernel with $\mathcal{O}(\ell^3)$ vertices, where ℓ denotes the size of a vertex cover.*

Proof. Let $|S| = \ell$. Let (G, k, S) be an instance of Special Treewidth parameterized by a vertex cover. Let (G', k', S') be the instance obtained from exhaustive application of Rules 01 through 06. By safety of the reduction rules, we have that (G', k', S') answers as **Yes iff** (G, k, S) answers as **Yes**.

The reduction rules guarantee that $S' \subseteq S$ is a vertex cover in G', with $|S'| \leq \ell$. Each vertex $v \in V' \backslash S'$ either has at least one pair of distinct neighbors in S' that are not adjacent, or v has exactly 1 pair of distinct neighbours in S' that are adjacent (a clique of size 2 which does not have a simplicial vertex besides v, otherwise it would have been handled by Rule 4), otherwise v is simplicial and has degree ≥ 3 and would have been handled by Rules 4 and 6, or v is simplicial and has degree ≤ 1 and would have been handled by Rules 1 and 3.

In some cases it might be beneficial to not apply Rule 06, since this actually increases the amount of vertices outside the vertex cover. However, this does not affect our worst-case analysis.

Assign v to this pair. If we assign v to the pair $\{w, x\}$, then v is a common neighbor of w and x. Hence a pair of vertices in S cannot have more than k vertices in $V \backslash S$ assigned to it, otherwise Rule 5 applies, which would make all vertices in $V \backslash S$ assigned to $\{w, x\}$ simplicial, and thus Rule 4 would apply until only 1 vertex in $V \backslash S$ assigned to $\{w, x\}$ remains. As there are at most $\ell \cdot (\ell - 1)/2$ pairs of neighbors in S', we have $|V' \backslash S'| \leq k \cdot \ell \cdot (\ell - 1)/2$. Since Rule 2 handles instances where $k \geq \ell$, we have that $k \leq \ell$. Thus we have that $|V' \backslash S'| \leq \ell^2 \cdot (\ell - 1)/2 \in \mathcal{O}(\ell^3)$. □

By combining Theorem 1 with a polynomial-time 2-approximation algorithm for vertex cover, we obtain the following corollary.

Corollary 1. *There is a polynomial-time algorithm that given an instance* $(G = (V, E), k)$ *of* Special Treewidth *computes an equivalent instance* $(G' = (V', E'), k)$ *such that* $V' \subseteq V$ *and* $|V'| \in \mathcal{O}((\ell^*)^3)$*, where* ℓ^* *is the size of a minimum vertex cover of* G*.*

In the next chapter, we present a characterization for the Special Treewidth problem on sp-partial 2-trees using the notion of *mamba-trees* and *Paths of Cycles*.

Chapter 4

Special Treewith on sp-partial 2-trees

In this chapter, a characterization for the Special Treewidth problem on sp-partial 2-trees is presented.

4.1 Mambas

In this section we define the notion of mambas.

Lemma 1. *If a biconnected graph G has special treewidth ≤ 2 then G has pathwidth ≤ 2.*

Proof. If G has special treewidth ≤ 2, then we can make a minimal special tree-decomposition $(\{X_i | i \in I\}, T = (I, F))$ of width 2. We now prove by contradiction. Suppose T is not a path-decomposition. Then there is a bag X_j such that X_j has two child-bags. Let these bags be X_a and X_b.

Suppose $X_j \cap X_a = \emptyset$. Then we know that G is not connected. *Contradiction.*

Now suppose $|X_j \cap X_a| = 1$. Let $v \in X_j \cap X_a$ and let $W = \bigcup\{X_k | k = a$ or X_k is a descendant of $X_a\}$.

- If $|W| = 1$, then $W = \{v\}$, and thus T is not minimal. *Contradiction.*
- Else take $w \in W$ such that $w \neq v$. All paths from w to a vertex in X_b must use v, thus v is a cut vertex and thus we know that G is not biconnected. *Contradiction.*

Thus we know that $|X_j \cap X_a| \geq 2$, and by the same reasoning as above we know that $|X_j \cap X_b| \geq 2$. Since T is a special tree-decomposition, we know that $X_a \cap X_b = \emptyset$ and thus $|X_j| \geq 4$. Thus T has width ≥ 3. *Contradiction.*

Thus we know that T is a path-decomposition of G of width ≤ 2. □

Lemma 2. *If a graph G has pathwidth ≤ 2 then G has special treewidth ≤ 2.*

Proof. Trivial. If G has pathwidth ≤ 2, then we can make a path-decomposition P of width 2. Now let P be a special tree-decomposition of G. P has width 2 and thus G has special treewidth ≤ 2. □

Lemma 3. *A biconnected graph G has special treewidth ≤ 2 if and only if G has pathwidth ≤ 2.*

Proof. This lemma follows directly from Lemma 1 and Lemma 2. □

We now define the notion of *mambas*.

Definition 8. *We call a subgraph H of a graph G a* mamba *if H is a maximal biconnected component in G with special treewidth ≤ 2.*

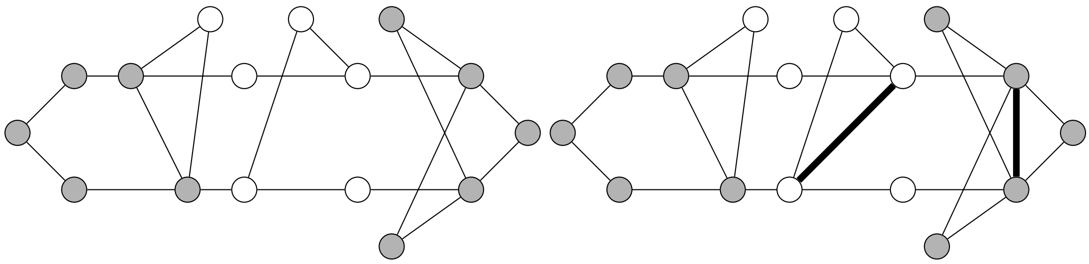

Figure 4.1: **Left Graph:** A mamba M, **Right Graph:** The cell completion of M, $\bar{M}$. Grey vertices are *head-vertices*

4.2 Defining sp-partial 2-trees through means of mamba-trees

In this section we show how to characterize sp-partial 2-trees using the notion of mambas.

Definition 9. *A* head-vertex *is a vertex v in a graph G such that there is a directed(rooted) special tree-decomposition $(\{X_i | i \in I\}, I = \{1, \ldots, m\}, T = (I, F))$ of width 2 such that $v \in X_1$.*

We now give a recursive definition of a *mamba-tree.*

Definition 10. *The class of* mamba-trees *is the class of graphs recursively defined as follows.*

- *Each mamba is a mamba-tree.*
- *For each mamba-tree G and each mamba M, the graph obtained by identifying a vertex in G with a head-vertex in M, is a mamba-tree.*

Next, we show that the special treewidth of a mamba-tree is ≤ 2 by constructing a special tree-decomposition.

Lemma 4. *The special treewidth of a mamba-tree is ≤ 2.*

Proof. We will prove this lemma by means of induction. Let $G = (V, E)$ be a mamba-tree. We construct a special tree-decomposition of width ≤ 2 as follows:

- If G is a mamba then we know from Definition 8 that we can make a special tree-decomposition T of width ≤ 2
- Otherwise, select a maximal biconnected component $B \subset G$ such that only one vertex $v \in B$ is a cut-vertex. Clearly, B is a mamba. We know from Definition 10 that v must be a head-vertex. Thus we can make a special tree-decomposition $(\{X_i | i \in I\}, I = \{1, \ldots, m\}, T_B = (I, F))$ of B of width 2 such that $v \in X_1$.

  Now make a special tree-decomposition $T_{G \backslash B}$ of $G \backslash B$ by induction. Next, let X_1 be the child of the lowest bag in $T_{G \backslash B}$ containing v. We now have a special tree-decomposition of G of width ≤ 2.

Thus we know that the special treewidth of a mamba-tree is ≤ 2. □

Next we will characterize the set of graphs with special treewidth ≤ 2 using the notion of mamba-trees.

Lemma 5. *If a graph $G = (V, E)$ is a disjoint union of mamba-trees, then G has special treewidth ≤ 2.*

Proof. Trivial. We know from Lemma 4 that a mamba-tree has special treewidth ≤ 2, and thus G has special treewidth ≤ 2. □

Lemma 6. *If a graph G has special treewidth ≤ 2, then G is a disjoint union of mamba-trees.*

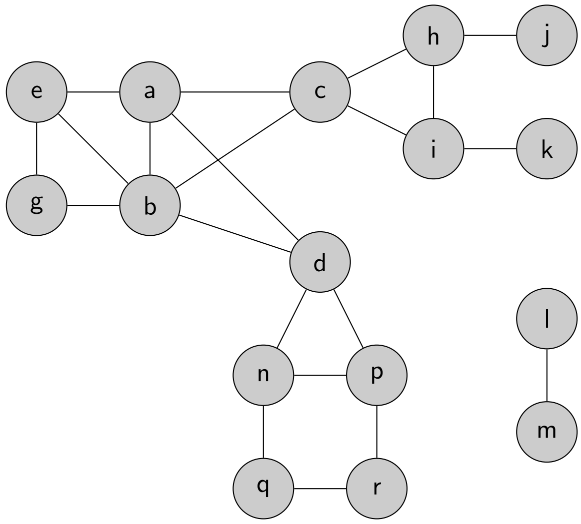

Figure 4.2: A disjoint union of two mamba-trees

Proof. We will prove this lemma with a minimal counterexample. Let G be a minimal graph of special treewidth ≤ 2 that is not a disjoint union of mamba-trees. Since G is a minimal counterexample, we know that every subgraph of G has special treewidth ≤ 2 and must be a disjoint union of mamba-trees.

Now let X_r be a bag in T and let X_c be a child of X_r, let $V = \bigcup\{X_k | k = c \text{ or } X_k \text{ is a descendant of } X_c\}$ and let $W = (\bigcup\{X_k | k = r \text{ or } X_k \text{ is a descendant of } X_r\}) \setminus V$. Clearly, $V \bigcap W = G$.

- Suppose $X_c \cap X_r = \emptyset$. Then we know that $V \cap W = \emptyset$. Thus we know that V and W are two disjoint graphs in G. Since V and W are both a disjoint union of mamba-trees, we know that the union of V and W is also a disjoint union of mamba-trees. *Contradiction.*
- Now suppose $|X_c \cap X_r| = 1$. Let $v \in X_c \cap X_r$. Then we know that $V \cap W = \{v\}$, and thus v is a cut vertex between V and W in G. Since X_c is the top bag of the special tree-decomposition of V and X_c contains v, we know that v is a head-vertex on V. Since V and W are both a disjoint union of mamba-trees, we know from the recursive definition of mamba-trees that G can be constructed by identifying $v \in V$ with vertex $v \in W$. Hence we know that G is a disjoint union of mamba-trees. *Contradiction.*
- Next suppose $|X_c \cap X_r| = 2$. We know that V and W are both a disjoint union of mamba-trees, X_r is a bag of a mamba $M_w \subseteq W$ and $X_c \subseteq V$ is the top bag of the root mamba of V. Since there can be only one child X_c such that $|X_c \cap X_r| = 2$ for each parent bag X_r, we know that X_r and X_c are part of the same path-decomposition for a biconnected component in G. Thus we know that X_r and X_c belong to the same mamba, and thus G is a disjoint union of mamba-trees. *Contradiction.*
- Finally, suppose $|X_c \cap X_r| \geq 3$: since T is minimal, we know that no two bags contain all the same vertices, and a bag in a special tree-decomposition of width 2 can contain at most 3 vertices. Thus we know that G was not a minimal graph of special treewidth ≤ 2. *Contradiction.*

Thus we know that G is a disjoint union of mamba-trees. □

We can now derive the following corollary from Lemma 5 and Lemma 6.

Corollary 2. *A graph $G = (V, E)$ has special treewidth $\leq 2 \Leftrightarrow G$ is a disjoint union of mamba-trees.*

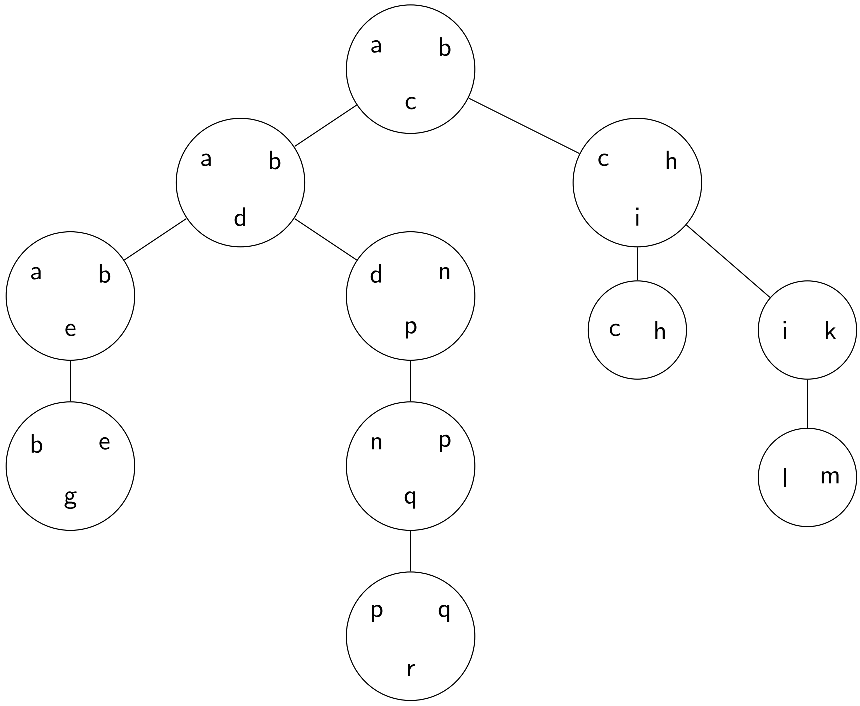

Figure 4.3: A special tree-decomposition of the graph shown in Figure 4.2

4.3 Defining sp-partial 2-trees through means of paths of cycles

In this section, we will use results by de Fluiter [10] to further characterize mamba-trees.

Definition 11 (de Fluiter [10])**.** *Given a biconnected graph $G = (V, E)$, the* cell completion $\bar{G}$ *of G is the graph which is obtained from G by adding an edge $\{u, v\}$ for all pairs u, v of non-adjacent vertices in V, $u \neq v$, such that $G[V(G)\backslash\{u,v\}]$ has at least three connected components.*

Definition 12 (de Fluiter [10], Bodlaender and Kloks [4])**.** *The class of* trees of cycles *is the class of graphs recursively defined as follows.*

- *Each cycle is a tree of cycles.*
- *For each tree of cycles G and each cycle C, the graph obtained from G and C by taking the disjoint union and then identifying an edge and its end vertices in G with an edge and its end vertices in C, is a tree of cycles.*

Note that two different chordless cycles in a tree of cycles have at most one edge in common

Definition 13 (de Fluiter [10])**.** *A* path of cycles *is a tree of cycles G for which the following holds.*

1. *Each chordless cycle of G has at most two edges which are contained in other chordless cycles of G.*
2. *If an edge $e \in E(G)$ is contained in $m \geq 3$ chordless cycles of G, then at least $m - 2$ of these cycles have no other edges in common with other chordless cycles, and consist of three vertices.*

Lemma 7 (de Fluiter [10])**.** *Let G be a biconnected graph. G is a partial two-path if and only if $\bar{G}$ is a path of cycles.*

Lemma 8. *A biconnected graph $G = (V, E)$ is a mamba if and only if $\bar{G}$ is a path of cycles.*

Proof. Suppose G is a mamba. We know from the definition of mambas that G has special treewidth ≤ 2. Since G is biconnected, we know from Lemma 3 that G must have pathwidth ≤ 2, and thus G is a partial two-path. Thus we know from Lemma 7 that $\bar{G}$ is a path of cycles.

Now suppose $\bar{G}$ is a path of cycles. Since G is biconnected, we know from Lemma 7 that G is a partial two-path. Thus we know that G has pathwidth ≤ 2. Thus we know from Lemma 3 that G must have special treewidth ≤ 2, and thus we now know that G is a mamba. □

Definition 14 (de Fluiter [10])**.** *Let G be a path of cycles, let $C = (C_1, \ldots, C_p)$ be a sequence of chordless cycles as defined above, and let $E = (e_1, \ldots, e_{p-1})$ be the corresponding set of common edges. The pair (C, E) is called a* cycle path *for G.*

Lemma 9. *Let $G = (V, E)$ be a biconnected graph with $v \in V$. We can make a special tree-decomposition $(\{X_i | i \in I\}, I = \{1, \ldots, m\}, T = (I, F))$ of width ≤ 2 such that $v \in X_1$, if and only if there exists a cycle path $(C, E), C = (C_1, \ldots, C_p)$ of $\bar{(G)}$ such that $v \in C_1$.*

Proof. Suppose we have a special tree-decomposition $(\{X_i | i \in I\}, I = \{1, \ldots, m\}, T = (I, F))$ of width ≤ 2 such that $v \in X_1$. Suppose there does not exist a cycle path $(C, E), C = (C_1, \ldots, C_p)$ of $\bar{(G)}$ such that $v \in C_1$. Then we know that v has to be in a cycle C_v such that C_v shares two edges with other chordless cycles. However, this means that we cannot make a special tree-decomposition of width ≤ 2 such that $v \in X_1$. Thus we know that there must exist a cycle path $(C, E), C = (C_1, \ldots, C_p)$ of $\bar{(G)}$ such that $v \in C_1$.

Now suppose we have a cycle path $(C, E), C = (C_1, \ldots, C_p)$ of $\bar{(G)}$ such that $v \in C_1$. For each cycle $C_k \in C$, make a special tree-decomposition T_k of width ≤ 2, such that the vertices neighbouring edge e_{k-1} are in the first bag, and the vertices neighbouring e_{k+1} are in the last bag. Make the special tree-decomposition T_1 such that v is in each bag in T_1. Now make the path-decomposition of G by pasting the special tree-decompositions of each cycle in order. We now have a special tree-decomposition $(\{X_i | i \in I\}, I = \{1, \ldots, m\}, T = (I, F))$ of width ≤ 2 such that $v \in X_1$. □

Lemma 10. *A vertex v in a mamba M is a head-vertex if and only if v is on a chordless cycle in $\bar{M}$ which has at most one edge which is contained in other chordless cycles in $\bar{M}$.*

Proof. Suppose v is a head-vertex in mamba M. Then we know that we can make a special tree-decomposition $(\{X_i | i \in I\}, I = \{1, \ldots, m\}, T = (I, F))$ of width ≤ 2 such that $v \in X_1$. Thus, from Lemma 9, we know that there must exist a cycle path $(C, E), C = (C_1, \ldots, C_p)$ of $\bar{(M)}$ such that $v \in C_1$. Since a cycle C_i in a cycle path shares exactly one edge with the cycles before it on the cycle path, and one edge with the cycles that come after it on the cycle path, we know that C_1 shares at most one edge with other cycles.

Now suppose v is on a chordless cycle C_v in $\bar{M}$ which has at most one edge which is contained in other chordless cycles in $\bar{M}$. Since M is a mamba, we know from Lemma 8 that $\bar{M}$ is a path of cycles. Since C_v shares at most one edge with other cycles, we know that there exists a cycle path $(C, E), C = (C_1, \ldots, C_p)$ such that $C_1 = C_v$. We now know from Lemma 9 that we can now make a special tree-decomposition $(\{X_i | i \in I\}, I = \{1, \ldots, m\}, T = (I, F))$ of width ≤ 2 such that $v \in X_1$. □

Lemma 11. *G is a disjoint union of mamba-trees if and only if*

- *G is a graph where the cell completion of each maximal biconnected component is a path of cycles,*
- *and for each connected component C there exists a tree-like ordering $\mathcal{O}_c$ of the maximal biconnected components in C such that each maximal biconnected component M in C is connected to its parent in $\mathcal{O}_c$ through a cut vertex $v \in V$ that is on a chordless cycle in $\bar{M}$ which has at most one edge which is contained in other chordless cycles in $\bar{M}$.*

Proof. Suppose G is a disjoint union of mamba-trees. Then we know from the definition of mamba-trees that every biconnected component in G is a mamba. We know from Lemma 8 that if a graph-component M is a mamba, then the cell completion $\bar{M}$ of M is a path of cycles. Thus G

is a graph where the cell completion of each maximal biconnected component is a path of cycles. We know from the recursive definition of mamba-trees that there is a tree-like ordering $\mathcal{O}_c$ of the mambas in a mamba-tree C, such that each mamba $M \in \mathcal{O}_c$ is connected to its parent through a head vertex v on M. We know from Lemma 10 that v is a head-vertex in M if and only if v is on a chordless cycle in $\bar{M}$ which has at most one edge which is contained in other chordless cycles in $\bar{M}$. Thus we know that G is a graph where the cell completion of each maximal biconnected component is a path of cycles, and for each connected component C there exists a tree-like ordering $\mathcal{O}_c$ of the maximal biconnected components in C such that each $M \in \mathcal{O}_c$ is connected to its parent through a cut vertex $v \in V$, such that v is on a chordless cycle in $\bar{M}$ which has at most one edge which is contained in other chordless cycles in $\bar{M}$.

Now suppose G is a graph where the cell completion of each maximal biconnected component is a path of cycles, and for each connected component C there exists a tree-like ordering $\mathcal{O}_c$ of the maximal biconnected components in C such that each $M \in \mathcal{O}_c$ is connected to its parent through a cut vertex $v \in V$, such that v is on a chordless cycle in $\bar{M}$ which has at most one edge which is contained in other chordless cycles in $\bar{M}$. We know from Lemma 8 that if the cell completion $\bar{M}$ of a graph-component M is a path of cycles, then M is a mamba. Thus we know that each biconnected component in G is a mamba. We know from Lemma 10 that v is a head-vertex in M if and only if v is on a chordless cycle in $\bar{M}$ which has at most one edge which is contained in other chordless cycles in $\bar{M}$. Thus we know that there exists an ordering $\mathcal{O}_c$ of the mambas in G such that each mamba $M \in \mathcal{O}_c$ is connected to its parent through a head-vertex on M. Thus we know that we can recursively construct a mamba-tree from each connected component in G and thus G is a disjoint union of mamba-trees. □

We can now derive the following corollary from Corollary 2 and Lemma 11:

Corollary 3. *A graph $G = (V, E)$ has special treewidth ≤ 2 if and only if*

- *G is a graph where the cell completion of each maximal biconnected component is a path of cycles,*
- *and for each connected component C there exists a tree-like ordering $\mathcal{O}_c$ of the maximal biconnected components in C such that each maximal biconnected component M in C is connected to its parent in $\mathcal{O}_c$ through a cut vertex $v \in V$ that is on a chordless cycle in $\bar{M}$ which has at most one edge which is contained in other chordless cycles in $\bar{M}$.*

In the next chapter, the kernelization for Spaghetti Treewidth parameterized by a Vertex Cover (i.e., parameterized by a modulator to an independent set) is presented. The kernelization again focuses mostly on simplicial vertices, and reducing the size and occurrences of these simplicial vertices. We recycle five of the reduction rules used for Special Treewidth parameterized by a Vertex Cover and introduce one new rule. With these rules we prove that we acquire a kernel of size $\mathcal{O}(l^3)$.

Chapter 5

Kernelization Rules for Spaghetti Treewidth

In this chapter, the kernelization for SPAGHETTI TREEWIDTH PARAMETERIZED BY A VERTEX COVER (i.e., parameterized by a modulator to an independent set) is presented. Like the kernelization for the SPECIAL TREEWIDTH problem, this kernelization again focuses mostly on simplicial vertices and reducing the size and occurrences of these simplicial vertices. Five of the reduction rules used for SPECIAL TREEWIDTH PARAMETERIZED BY A VERTEX COVER are recycled to be used for this kernel, and one new rule is introduced. It is shown that with these rules a kernel of size $\mathcal{O}(\ell^3)$ can be made.

5.1 Trivial rules

In this section a number of reduction rules are listed for instances of SPAGHETTI TREEWIDTH. The five rules are the same as rules 1-4 and 6 for instances of SPECIAL TREEWIDTH, which hold for both instances. Proofs for these rules have been omitted, since they are exactly the same as discussed in previous sections.

The Islet Rule

We specify a rule which deals with vertices of degree 0.

Rule 7 (Islet Rule). *If v is a vertex of degree 0 then remove v.*

Proof. Removing a vertex does not increase the spaghetti treewidth of the graph (Proposition 4). The rest of the proof goes the same as the proof for Rule 1 for SPECIAL TREEWIDTH. □

Trivial Decisision

We specify a rule which deals with parameters k which are greater than the size of our vertex cover.

Rule 8 (Trivial Decision). *If $k \geq |S|$, then answer* **YES**.

Proof. The proof goes the same as the proof for Rule 2 for SPECIAL TREEWIDTH. □

The Twig Rule

We specify a rule which deals with vertices of degree 1.

Rule 9 (Twig Rule). *If v is a vertex of degree 1 then remove v. If $v \in S$, then let $S' := S \backslash \{v\}$, else let $S' := S$.*

Proof. Removing a vertex does not increase the spaghetti treewidth of the graph (Proposition 4). The rest of the proof goes the same as the proof for Rule 3 for SPECIAL TREEWIDTH. □

5.2 Simple rules for simplicial vertices

The Duplicate Simplicial Vertex Rule

We specify a rule which deals with duplicate simplicial vertices.

Rule 10 (Duplicate Simplicial Vertex Rule). *Let v and w be two simplicial vertices with $N_G(v) = N_G(w)$. If $v \neq w$ then remove w. If $w \in S$ then let $S' := S \backslash \{w\}$. Else let $S' := S$.*

Proof. Removing a vertex does not increase the spaghetti treewidth of the graph (Proposition 4). The rest of the proof goes the same as the proof for Rule 4 for Special Treewidth. □

The High Degree Simplicial Vertex Rule

Next we define another trivial rule which handles simplicial vertices of degree $> k$. Again, this rule is not necessary to achieve our kernelization of Spaghetti Treewidth parameterized by a Vertex Cover, but it is an elegant rule which could save some extra work when dealing with simplicial vertices.

Rule 10b (High Degree Simplicial Vertex Rule). *If v is a simplicial vertex of degree $> k$ then answer* **No***.*

Proof. The proof goes the same as the proof for Rule 4b for Special Treewidth. □

5.3 Complex rules for removing simplicial vertices

The Common Neighbours Improvement Rule

We specify a rule which deals with pairs of vertices with at least $k + 1$ common neighbours.

Rule 11 (Common Neighbours Improvement Rule). *Suppose that $\{v, w\} \notin E$ and that $v \in S$ or $w \in S$. If $|N_G(v) \cap N_G(w)| \geq k + 1$, then add edge $\{v, w\}$ to E. Let $S' := S$.*

Proof. The proof goes the same as the proof for Rule 5 for Special Treewidth. □

The Simplicial Vertex Partition Rule

In this subsection it is shown that Rule 6 for Special Treewidth, which removes simplicial vertices of degree ≥ 3, can not be applied on instances of Spaghetti Treewidth. Instead, a new rule is introduced which removes simplicial vertices of degree ≥ 4. The new set of rules provide us with a kernel for Spaghetti Treewidth parameterized by a vertex cover.

We first show that Rule 6 for Special Treewidth does not apply to the case of Spaghetti Treewidth, unlike the other 5 rules for Special Treewidth, which did.

Suppose we have a clique C with $|C| = 4$. It is clear from Proposition 5 that $spghtw(C) = 3$. Now if we try to apply Rule 6 on vertex $v \in C$, we get a graph C' which is a 3-sun(S_3) as in Figure 5.1. Let A, B, C be the three sets containing the vertices of the three outer cliques and let X be the set containing the vertices of the inner clique. Since $|A| = |B| = |C| = |X| = 3$ we have that the spaghetti treewidth of G is at least 2.

We can now make a spaghetti tree-decomposition T as follows: Let bag X be the root of T, and let A,B and C be the children of X in T. It should be clear that T is now a spaghetti tree-decomposition of spaghetti treewidth 2. Thus we cannot safely split simplicial vertices into vertices of degree 2.

We now give two definitions which we will use to prove simplicial vertices cannot simply be removed.

Definition 15 (Spaghetti Treewidth Defining Bag). *A spaghetti treewidth defining bag (spghtw-defining bag) of a spaghetti tree-decomposition $(\{X_i | i \in I\}, T = (I, F))$ of $G = (V, E)$, is a bag X_v such that $|X_v| = \mathbf{spghtw}(G) + 1$.*

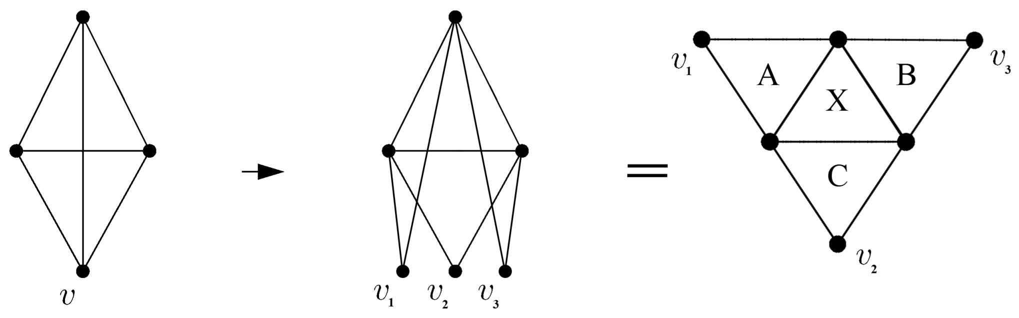

Figure 5.1: Counterexample for splitting simplicial vertices into vertices of degree 2 in spaghetti treewidth

Definition 16 (Uniquely Spaghetti Treewidth Defining Bag). *A uniquely spaghetti treewidth defining bag (uniquely spghtw-defining bag) of a spaghetti tree-decomposition* $(\{X_i|i \in I\}, T = (I, F))$ *of* $G = (V, E)$*, is a bag* X_u *such that* $\forall_{i \in I \setminus \{u\}} |X_i| < |X_u|$.

A counter proof for removing singular simplicial vertices can be easily constructed to show that removing a singular simplicial vertex v from a graph G when looking at bounded spaghetti treewidth is unsafe, when v is in the uniquely spghtw-defining bag X_v in the minimal spaghetti tree-decomposition T with spaghetti treewidth h. When removing this vertex v, for the spaghetti treewidth h' of T' we will have that $h \geq h'$. Since altering the graph is only safe when $max\{\mathbf{spghtw}(G), low_G\} = max\{\mathbf{spghtw}(G'), low'_G\}$ holds, we have that in this case $low'_G = |X_v|$. However, if we want to know the size of bag X_v, we first have to make a spaghetti tree-decomposition of spaghetti width k. However, in order to make this we would first have to know whether the spaghetti treewidth is bounded by k. But this was what we wanted to know in the first place! Thus we cannot safely remove singular simplicial vertices.

We now specify a rule which deals with singular simplicial vertices of degree $\leq k$. A singular simplicial vertex is a simplicial vertex $v \in V$ for which there is no vertex $w \in V$ for which we have $w \neq v$ and $N_G(v) = N_G(w)$.

This rule, combined with rules 8 and 10, removes all simplicial vertices v with $\delta_G(v) > 3$.

Rule 12 (Simplicial Vertex Partition Rule). *If v is a simplicial vertex with degree $4 \leq \delta_G(v) \leq k$, then for each triplet of vertices $\{w, x, y\} \subset N_G(v)$, make a new vertex incident to w,x and y. Then remove v. If $v \in S$ then let $S' := S \setminus \{v\} \cup \{z\}$ where $z \in N_G(v), z \notin S$, else let $S' := S$.*

Proof. It is clear that S' is a vertex cover of G'.
Removing a vertex does not increase the spaghetti treewidth of the graph (Proposition 4).
Let $G' = (V', E')$ be the reduced graph acquired from doing the following: let $\delta_G(v) = n$ be the degree of v. Since v is simplicial, we have a clique of size n. Let $q = n \cdot (n-1) \cdot (n-2)/5$. Now replace v with the vertex set $N = \{v_1, \ldots, v_q\}$, and assign to each pair $\{x, y\} \subset N_G(v)$ a vertex $v_i \in N$ such that $\{v_i, x\} \in E'$ and $\{v_i, y\} \in E'$.

Now let $(\{X_i|i \in I\}, T = (I, F))$ be a spaghetti tree-decomposition of G and let $(\{X_i|i \in I'\}, T' = (I', F'))$ be a spaghetti tree-decomposition of G'. We show that $\mathbf{spghtw}(G') \leq \mathbf{spghtw}(G)$ as follows:

Let T be the spaghetti tree-decomposition of G with spaghetti treewidth h, and v in only one bag X_i. This is a reasonable assumption; since v is simplicial it has to be in a bag together with $N_G(v)$, and it does not need to be in any other bag. Thus take $\{X_i|N_G[v] \subseteq X_i\}$. We can now split bag X_i such that we get a directed path of bags $P\{X_1, \ldots, X_q\}$ such that for each $w \in N$ there is exactly 1 bag $\{X_w \in P|X_w = \{w\} \cup (X_i \setminus \{v\})\}$. Now replace X_i in T with P to get T'. Clearly, we now have a spaghetti tree-decomposition T' of width h and thus we can conclude that

spghtw$(G') \leq$ **spghtw**(G).

We assume **spghtw**$(G') <$ **spghtw**(G).

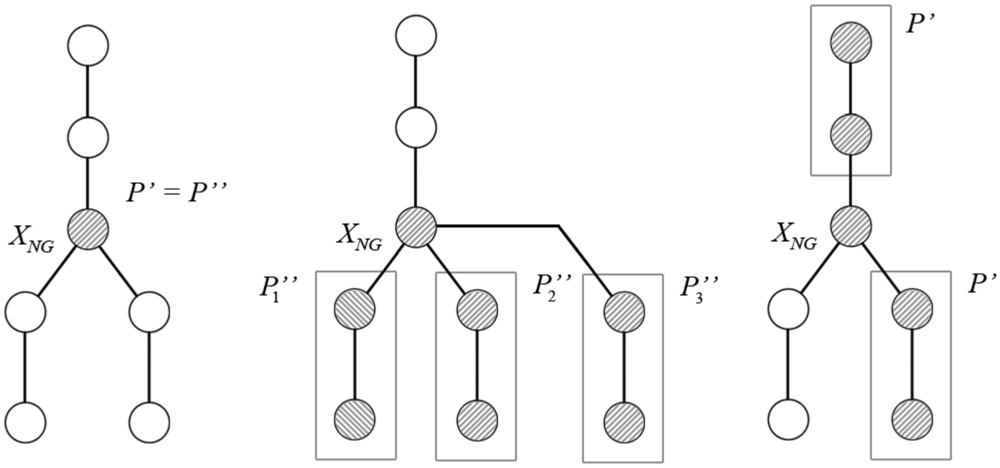

Figure 5.2: Illustration of the proof for Rule 12.

We now show that **spghtw**$(G') \geq$ **spghtw**(G) as follows: Let T be the spaghetti tree-decomposition of G with spaghetti treewidth h, and v in only one bag X_i. This is a reasonable assumption; since v is simplicial it has to be in a bag together with $N_G(v)$, and it does not need to be in any other bag. Thus take $\{X_i | N_G[v] \subseteq X_i\}$. We simplify the proof by stating that $X_i = N_G[v]$. This does not affect our proof: simply add $X_i \backslash N_G[v]$ to each bag X_t containing a vertex $w \in N_G[v]$, where X_t is a bag created to get from T to T'.

Now go from T to T' as follows: Replace $X_i \in T$ by $\{X_{NG} | X_{NG} = X_i \backslash \{v\}\}$. Clearly, we now have that $|X_i| = |X_{NG}| + 1$. We will now try to get T' from T by adding all vertices $x \in N$ to the bags in T, such that each x is in a bag with its neighbours and **spghtw**$(G') <$ **spghtw**(G).

We observe that according to the definition of Spaghetti Treewidth, we do not care if vertices are on a rooted path or on an unrooted path. Thus it suffices to only look at the case where X_{NG} has children (i.e. it is easy to see that we can root one of the children of X_{NG} to make it the parent path of X_{NG}). We look at the case where X_{NG} has multiple children-paths P'' which contain vertices $\in N_G(v)$. Let $Y = N_G(v) = Y_1 \cup Y_2 \cup Y_3$ with $Y_1 \cap Y_2 \cap Y_3 = \emptyset$ and $Y_1 \neq \emptyset, Y_2 \neq \emptyset, Y_3 \neq \emptyset$. Let $N_G(v) = \{y_1, \ldots, y_r\}$ and let $\{s_{ijk} \in V' | \{s_{ijk}, y_i\} \in E, \{s_{ijk}, y_j\} \in E, \{s_{ijk}, y_k\} \in E\}$. Let P''_1 be the path below X_{NG} which contains vertices from set Y_1 but not from $Y_2 \cup Y_3$, let P''_2 be the path below X_{NG} which contains vertices from set Y_2 but not from $Y_1 \cup Y_3$, and let P''_3 be the path below X_{NG} which contains vertices from set Y_3 but not from $Y_1 \cup Y_2$. Now for each vertex $\{s_{ijk} \in V' | y_i \in Y_1, y_j \in Y_2, y_k \in Y_3\}$ we know that $\{s_{ijk}, y_i, y_j, y_k\}$ must be on a single path P''_n; if s_{ijk} would be on more than one child-path, it would imply that s_{ijk} had to be in X_{NG} and since $s_{ijk} \not\subseteq N_G(v)$, we know that in this case $|X_{P''_n}| \geq |N_G(v)| + 1$ and thus **spghtw**$(G') \geq$ **spghtw**(G) if $N_G(v)$ is partitioned to be in multiple children-paths of X_{NG}.

We now know that every s_{ijk} must be on at most one of two paths P''_1 and P''_2. Let $P' = P''_1$ and $P'' = P''_2$, and let $Y = Y_1 \cup Y_2$.

Now let $X_{P''}$ be the highest bag on path P'' with $Y \backslash Y_1 \subseteq X_{P''}$ and let $X_{P'}$ be the lowest bag on path P' with $Y \backslash Y_2 \subseteq X_{P'}$. Clearly, since all vertices S_{ijk} must be on P we have that $Y = N_G(v) = Y_1 \cup Y_2$ with $Y_1 \cap Y_2 = \emptyset$. Thus every vertex s_{ijk} such that $y_i/y_j/y_k \in Y \backslash Y_1$ and $y_i/y_j/y_k \in Y_1$ must be on P' and every vertex s_{ijk} such that $y_i/y_j/y_k \in Y \backslash Y_2$ and $y_i/y_j/y_k \in Y_2$ must be on P''. Since $Y_1 \cap Y_2 = \emptyset$ and $Y \subseteq P' \cup P''$, we have that $Y \backslash Y_1 = Y_2$ and $Y \backslash Y_2 = Y_1$ and thus we have that every vertex $y \in Y$ must be on path P'. Since P' must contain $N_G(v)$, we know that the lowest bag on P' has to contain $N_G(v)$. Thus, the lowest bag $X_{P'}$ on P' which contains at least one s_{ijk} must also contain $N_G(v)$. Since $\{s_{ijk} \not\subseteq N_G(v)$, we know that $|X_{P'}| \geq |N_G(v)| + 1$ and thus **spghtw**$(G') \geq$ **spghtw**(G).

Thus we can conclude that $\mathbf{spghtw}(G') = \mathbf{spghtw}(G)$ and thus this rule is safe. □

5.4 A Kernel for Spaghetti Treewidth

We can now reason that the exhaustive application of Rule 7 through 12 (i.e., until we answer **Yes** or **No** or no application of one of these rules is possible) gives a polynomial kernel for Spaghetti Treewidth parameterized by a vertex cover. It is clear that this reduction can be performed in polynomial time (it is easy to do it in time $O(|V| \cdot |E|)$). Let S denote a vertex cover of G.

Theorem 2. Special Treewidth parameterized by a vertex cover *has a kernel with* $\mathcal{O}(\ell^3)$ *vertices, where* ℓ *denotes the size of a vertex cover.*

Proof. Let $|S| = \ell$. Let (G, k, S) be an instance of Spaghetti Treewidth parameterized by a vertex cover. Let (G', k', S') be the instance obtained from exhaustive application of Rules 7 through 12. By safety of the reduction rules, we have that (G', k', S') answers as **Yes iff** (G, k, S) answers as **Yes**.

The reduction rules guarantee that $S' \subseteq S$ is a vertex cover in G', with $|S'| \leq \ell$. Each vertex $v \in V' \backslash S'$ either has at least one pair of distinct neighbors in S' that are not adjacent, or v has exactly 1 pair of distinct neighbours in S' that are adjacent (a clique of size 2 which does not have a simplicial vertex besides v, otherwise it would have been handled by Rule 10), or v has exactly 1 triplet of distinct neighbours in S' that are adjacent (a clique of size 3 which does not have a simplicial vertex besides v, otherwise it would have been handled by Rule 10), otherwise v is simplicial and has degree ≥ 3 and would have been handled by Rules 10 and 12, or v is simplicial and has degree ≤ 1 and would have been handled by Rules 7 and 9.

Assign v to the triplet. If we assign v to the pair $\{w, x\}$ (if we assign v to the triplet $\{w, x, y\}$ then we implicitly assign v to the pairs $\{w, x\}$, $\{w, y\}$ and $\{x, y\}$), then v is a common neighbor of w and x. Hence any pair of vertices in S cannot have more than k vertices in $V \backslash S$ assigned to it, otherwise Rule 11 applies, which would make all vertices in $V \backslash S$ assigned to $\{w, x\}$ simplicial, and thus Rule 10 would apply until only 1 vertex in $V \backslash S$ assigned to $\{w, x\}$ remains. As there are at most $\ell \cdot (\ell - 1)/2$ pairs of neighbors in S' and at most $\ell \cdot (\ell - 1) \cdot (\ell - 2)/5$ triplets of neighbours in S', we have $|V' \backslash S'| \leq k \cdot \ell \cdot (\ell - 1)/2 + \ell \cdot (\ell - 1) \cdot (\ell - 2)/5$. Since Rule 8 handles instances where $k \geq \ell$, we have that $k \leq \ell$. Thus we have that $|V' \backslash S'| \leq \ell^2 \cdot (\ell - 1)/2 + \ell \cdot (\ell - 1) \cdot (\ell - 2)/5 \in \mathcal{O}(\ell^3)$. □

By combining Theorem 2 with a polynomial-time 2-approximation algorithm for vertex cover, we obtain the following corollary.

Corollary 4. *There is a polynomial-time algorithm that given an instance* $(G = (V, E), k)$ *of* Spaghetti Treewidth *computes an equivalent instance* $(G' = (V', E'), k)$ *such that* $V' \subseteq V$ *and* $|V'| \in \mathcal{O}((\ell^*)^3)$*, where* ℓ^* *is the size of a minimum vertex cover of* G.

Chapter 6

Conclusions

We introduced a new logical cousin to the TREEWIDTH and SPECIAL TREEWIDTH problems, the SPAGHETTI TREEWIDTH problem. We then considered the vertex cover parameterization for the SPECIAL TREEWIDTH and SPAGHETTI TREEWIDTH problems. Cubic kernels for both SPECIAL TREEWIDTH PARAMETERIZED BY A VERTEX COVER and SPAGHETTI TREEWIDTH PARAMETERIZED BY A VERTEX COVER were achieved through the use of a total of seven different reduction rules. It might be interesting to carry out experimental research to see how good rules 4, 6 and 12 behave when compared to the rules which remove simplicial vertices in TREEWIDTH PARAMETERIZED BY A VERTEX COVER.

Apart from improving upon the kernel size for SPECIAL TREEWIDTH PARAMETERIZED BY A VERTEX COVER and SPAGHETTI TREEWIDTH PARAMETERIZED BY A VERTEX COVER, it seems interesting to look for other polynomial kernels, e.g. by parameterization by a feedback vertex set.

In this work we also presented a characterization for the sp-partial 2-trees using the concept of *mamba-trees*, and a characterization using the concept of *Paths of Cycles*.

As a side-result, the characterization of mamba-trees led to the conjecture that the sp-partial 2-trees are closed under taking minors. Our conjecture proposes that that the obstruction-set for the sp-partial 2-trees consists of the minors K_4, S_3, D_3 and B_4. These minors are shown in Figure 6.1. Further research will be needed to prove this conjecture.

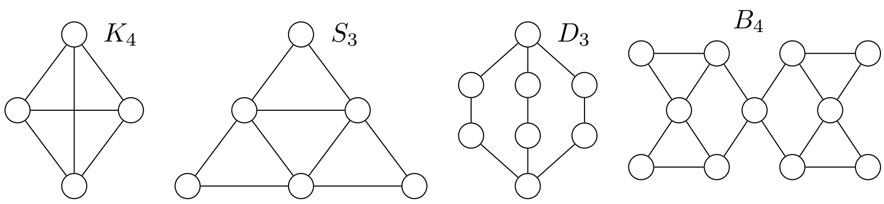

Figure 6.1: The forbidden sp-partial 2-tree minors K_4, S_3, D_3 and B_4.

Bibliography

[1] Achim Blumensath and Bruno Courcelle. On the monadic second-order transduction hierarchy. *Logical Methods in Computer Science*, 6(2), 2010.

[2] Hans L. Bodlaender, Rodney G. Downey, Michael R. Fellows, and Danny Hermelin. On problems without polynomial kernels. *J. Comput. Syst. Sci.*, 75(8):423–434, 2009.

[3] Hans L. Bodlaender, Bart M. P. Jansen, and Stefan Kratsch. Preprocessing for treewidth: A combinatorial analysis through kernelization. In Luca Aceto, Monika Henzinger, and Jiri Sgall, editors, *ICALP (1)*, volume 6755 of *Lecture Notes in Computer Science*, pages 437–448. Springer, 2011.

[4] Hans L. Bodlaender and Ton Kloks. A Simple Linear Time Algorithm for Triangulating Three-Colored Graphs. *J. Algorithms*, 15(1):160–172, 1993.

[5] Hans L. Bodlaender and Arie M. C. A. Koster. Treewidth computations I. Upper bounds. *Inf. Comput.*, 208(3):259–275, 2010.

[6] Hans L. Bodlaender and Arie M. C. A. Koster. Treewidth computations II. Lower bounds. *Inf. Comput.*, 209(7):1103–1119, 2011.

[7] Hans L. Bodlaender, Arie M. C. A. Koster, and Frank van den Eijkhof. Preprocessing rules for triangulation of probabilistic networks. *Computational Intelligence*, 21(3):286–305, 2005.

[8] Bruno Courcelle. Special tree-width and the verification of monadic second-order graph properties. In Kamal Lodaya and Meena Mahajan, editors, *FSTTCS*, volume 8 of *LIPIcs*, pages 13–29. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2010.

[9] Bruno Courcelle, Rodney G. Downey, and Michael R. Fellows. A Note on the Computability of Graph Minor Obstruction Sets for Monadic Second Order Ideals. *J. UCS*, 3(11):1194–1198, 1997.

[10] Babette de Fluiter. Algorithms for Graphs of Small Treewidth, Ph.D. thesis. 1997.

[11] Rodney G. Downey and M.R. Fellows. *Parameterized Complexity (Monographs in Computer Science)*. Springer, 1998.

[12] Jiong Guo and Rolf Niedermeier. Invitation to data reduction and problem kernelization. *SIGACT News*, 38(1):31–45, 2007.

[13] Neil Robertson and Paul D. Seymour. Graph minors. I. Excluding a forest. *J. Comb. Theory, Ser. B*, 35(1):39–61, 1983.

[14] Neil Robertson and Paul D. Seymour. Graph Minors. XX. Wagner's conjecture. *J. Comb. Theory, Ser. B*, 92(2):325–357, 2004.