Alcom-FT Technical Report Series
ALCOMFT-TR-02-62

Computing the treewidth and the minimum fill-in with the modular decomposition*

Hans L. Bodlaender† Udi Rotics‡

Abstract

Using the notion of modular decomposition we extend the class of graphs on which both the TREEWIDTH and the MINIMUM-FILL-IN problems can be solved in polynomial time. We show that if $\mathcal{C}$ is a class of graphs which is modularly decomposable into graphs that have a polynomial number of minimal separators, or graphs formed by adding a matching between two cliques, then both the TREEWIDTH and the MINIMUM-FILL-IN problems on $\mathcal{C}$ can be solved in polynomial time. For the graphs that are modular decomposable into cycles we give algorithms, that use respectively $O(n)$ and $O(n^3)$ time for TREEWIDTH and MINIMUM FILL-IN.

Keywords : treewidth, minimum fill-in, modular decomposition, minimal separators, polynomial algorithms, graph algorithms.

1 Introduction

A graph is chordal if it does not contains a chordless cycle of length at least four as an induced subgraph. A triangulation of a graph is a chordal supergraph with the same vertex set. The *treewidth* of a graph G, denoted as $treewidth(G)$ is the smallest clique number of all possible triangulations of G minus 1. The *minimum fill-in* of a graph G, denoted as *min-fill-in*(G), is the minimum of $|E(H) - E(G)|$ taken over all triangulations H of G. The TREEWIDTH problem is to find $treewidth(G)$ for a given graph G. The MINIMUM-FILL-IN problem is to find *min-fill-in*(G) for a given graph G. These problems have drawn much attention due to applications in areas such as Gaussian elimination of matrix, VLSI-layout, gate matrix layout and algorithmic graph theory (see e.g. [1, 5, 24]). Both problems are NP-hard in general [2, 27] but polynomial time algorithms exists for many special graph classes such as: permutation graphs [6], circular arc graphs [26], circle graphs [19], distance hereditary graphs [12], $(q, q-4)$-graphs [3] and HHD-free graphs [11]. Bouchitté and Todinca [8, 10] have shown that the treewidth and minimum fill-in of a graph can be computed in polynomial time if the graph has a polynomial number of minimal separators. This result generalizes several of the earlier results for special graph classes. In this paper we extend the class of graphs

*The first author was partially supported by EC contract IST-1999-14186: Project ALCOM-FT (Algorithms and Complexity - Future Technologies). The second author wishes to thank the Natural Science and Engineering Research Council of Canada and the Fields Institute for financial assistance.

†Institute of Information and Computing Sciences, Utrecht University, P.O. Box 80.089, 3508 TB Utrecht, the Netherlands, hansb@cs.uu.nl.

‡School of Mathematics and Computer Science, Netanya Academic College, P.O. Box 120, 42100 Netanya, Israel, rotics@mars.netanya.ac.il

on which these two problems can be solved in polynomial time using the notion of modular decomposition as described below.

A set M of vertices of a graph G is called a *module* of G if every vertex outside M is either adjacent to all vertices in M or to none of them. The graph G and the singletons of G are called the *trivial modules* of G. A graph G is called *prime* if all the modules in G are trivial. The *modular decomposition* $M(G)$ of G is a pair $(T(G), \pi(G))$, where $T(G)$ is a labeled tree and $\pi(G)$ is a set of prime graphs associated with some of the internal nodes of $T(G)$. We say also that G is *modularly decomposable* into $\pi(G)$. For more details, see Section 2. The modular decomposition of a graph is unique and can be found in linear time [14, 23]. We say that a class of graphs $\mathcal{C}$ is *modularly decomposable* into a class of graphs $\mathcal{D}$ if every graph $G \in \mathcal{C}$ is modularly decomposable into graphs in $\mathcal{D}$. We denote by n and m the number of vertices and edges of a graph, respectively. We call a graph a clique-matching graph if it can be obtained by taking two cliques with the same number r of vertices and then adding a matching with r edges between the cliques.

Dahlhaus [15] has shown that the TREEWIDTH and the MINIMUM-FILL-IN problems can be solved in polynomial time on the class of graphs which is modularly decomposable into chordal graphs. A result of a similar type is that TREEWIDTH [7] can be solved in linear time on cographs, i.e., on graphs modular decomposable into graphs with two vertices. In this paper, we extend these result to a much larger class of graphs. In particular, we show:

Theorem 1 *Let $\mathcal{C}_c$ be a class of graphs for which there exists a constant c such that for every graph $G \in \mathcal{C}$ all the graphs in $\pi(G)$ have at most n^c minimal separators (where n is the size of G) or are a clique-matching graph. Then the TREEWIDTH and the MINIMUM-FILL-IN problems on $\mathcal{C}$ can be solved in polynomial time.*

We think the most interesting part of this theorem is where we deal with graphs with a polynomial number of minimal separators; we added the result on clique-matching graphs as these have exponentially many minimal separators to show that computing TREEWIDTH and MINIMUM FILL-IN with help of the modular decomposition is not restricted to graphs with polynomially many separators. In fact, what we show is that we can compute the treewidth (minimum fill-in) in polynomial time given a prime graph which either has polynomially many minimal separators or is a clique-matching graph and for each module the treewidth (minimum fill-in) of the subgraph induced by the module and the number of vertices of the module. We expect that there are more types of graphs that have this property; if this property is established for a set of graphs $\mathcal{D}$, then Theorem 1 can be extended in the sense that we allow the graphs in $\pi(G)$ also belong to $\mathcal{D}$. In addition, we can allow a prime graph H in our modular decomposition with only singleton vertices below it, such that we can compute the treewidth (minimum fill-in) of H in any way (e.g., it is a regular grid, or it is a graph of treewidth bounded by some constant).

We define two new problems called the WI-TREEWIDTH and the WI-FILL-IN problems. We show (see Theorems 9 and 45) that an algorithm for solving the WI-TREEWIDTH (resp. the WI-FILL-IN) problem on a class of graphs $\mathcal{D}$ can be used to solve the TREEWIDTH (resp. the MINIMUM-FILL-IN) problem on a class of graphs which is modularly decomposable into $\mathcal{D}$ with the same time complexity. We then present polynomial time algorithms for the WI-TREEWIDTH and the WI-FILL-IN problems on the classes of graphs with polynomially many separators, cycles, and clique-matching graphs. Theorem 1 now follows by joining all these cases together. The result for cycles is not needed for Theorem 1, but this algorithm is faster than that obtained by applying the general result. (Cycles have a quadratic number of minimal separators.)

The paper is organized as follows. In Section 2 we define the WI-TREEWIDTH problem and indicate the relation between this problem and the TREEWIDTH problem. In Section 3 (resp. 4,5) we present a polynomial time algorithm for the TREEWIDTH problem on the class of graphs which is modularly decomposable into graphs with polynomially many minimal separators (resp. cycles, clique-matching graphs). In Section 6 we obtain these results also for the MINIMUM FILL-IN problem. Some final remarks are made in Section 7.

2 Definitions and preliminary results

The graphs we consider in this paper are undirected and loop-free. For a graph G we denote by $V(G)$ (resp. $E(G)$) the set of vertices (resp. edges) of G. For $X \subseteq V(G)$, we define by $G[X]$ the subgraph of G induced by X. The subgraph of G induced by $V(G) - X$ is denoted by $G - X$. $N(v)$ denotes the neighborhood of v in G, i.e., the set of vertices in G adjacent to v. For $W_1, \ldots, W_r \subseteq V(G)$, $G + clique(W_1, \ldots, W_r)$ denotes the graph G', obtained from G by making W_i a clique, for $1 \leq i \leq r$: $V(G') = V(G)$, $E(G') = E(G) \cup \{\{v, w\} \mid \exists i (v, w \in W_i \text{ and } v \neq w)\}$. A vertex v is *simplicial* in a graph G, if the neighbors of v form a clique.

Definition Let a and b be distinct nonadjacent vertices. A set $S \subset V$ is a *minimal a,b-separator* if a and b are in different connected components of $G - S$ and there is no subset of S with the same property. A *minimal separator* is a set S of vertices for which there exists vertices a and b such that S is a minimal a,b-separator.

For $X \subseteq V$ we say that X is a *potential maximal clique* in G if X is a maximal clique in some minimal triangulation H of G. We denote by $indp(G)$ the set of all independent sets in G, by $cc(G)$ the set of all connected components in G, by $msep(G)$ the set of all minimal separators of G and by $potm(G)$ the set of all potential maximal cliques in G.

The following is an equivalent definition of treewidth which was introduced by Robertson and Seymour in their work on graph minors [25].

Definition A *tree decomposition* of $G = (V, E)$ is a pair $(\{X_i : i \in I\}, T)$, where $\{X_i : i \in I\}$ is a collection of subsets of V and $T = (I, F)$ is a tree such that:

1. $\bigcup_{i \in I} X_i = V$.
2. $\forall \{u, w\} \in E, \exists i \in I : u, w \in X_i$.
3. $\forall i, j, k \in I$: if j is on a path in T from i to k then $X_i \cap X_k \subseteq X_j$.

The *width* of a tree decomposition $(\{X_i : i \in I\}, T)$ is $\max_{i \in I} |X_i| - 1$.
The *treewidth of* G (denoted as $treewidth(G)$) is the minimum width over all tree decompositions of G.

A tree decomposition $(\{X_i : i \in I\}, T)$ with T a path (i.e., every node in T has degree at most two) is called a *path decomposition*. A path decomposition is often denoted by listing the successive sets X_i: $(X_1, X_2, \ldots, X_r)$.

We say that a tree decomposition $(\{X_i \mid i \in I\}, T = (I, F))$ is *nice* if for every edge $\{i, j\} \in F$, $X_i \not\subseteq X_j$ and $X_j \not\subseteq X_i$. By contracting every edge $\{i, j\} \in F$ for which $X_i \subseteq X_j$, we can make a tree decomposition nice without increasing the treewidth. For a proof of Lemmas 2-5 see [7].

Lemma 2 *Let $(\{X_i \mid i \in I\}, T = (I, F))$ be a tree decomposition of G, and suppose $W \subseteq V(G)$ forms a clique in G. Then there exists an $i \in I$ with $W \subseteq X_i$.*

Lemma 3 *Let $(\{X_i \mid i \in I\}, T = (I, F))$ be a tree decomposition of G. Let $W_1, W_2 \subseteq V(G)$ and suppose for all $v \in W_1$, $w \in W_2$, $\{v, w\} \in E(G)$. Then there exists an $i \in I$ with $W_1 \subseteq X_i$ or $W_2 \subseteq X_i$.*

Lemma 4 *Let $(\{X_i \mid i \in I\}, T = (I, F))$ be a tree decomposition of G and let $j \in I$. Then $(\{X_i \mid i \in I\}, T = (I, F))$ is also a tree decomposition of $G + clique(X_j)$.*

For disjoint graphs H and F we denote by $H \times F$ the graph G obtained by taking the union of H and F and connecting all vertices of H to all vertices of F: $V(G) = V(H) \cup V(F)$ and $E(G) = E(H) \cup E(F) \cup \{\{v, w\} \mid v \in V(H) \text{ and } w \in V(F)\}$.

Lemma 5

$$treewidth(H \times F) = \min\{treewidth(H) + |V(F)|, treewidth(F) + |V(H)|\}.$$

For an independent set $X \in indp(G)$, we define $G : X$ to be the graph obtained from G by making each pair of neighbors of a vertex in X adjacent, and then removing all vertices in X. We assume that there are two weight functions w and t associating positive integer weights $w(v)$ and $t(v)$ for every vertex v of G. The motivation for these weights is that when G is considered as a prime graph in a modular decomposition then each vertex v of G corresponds to a module $M(v)$ and $w(v)$ and $t(v)$ will be the size and the treewidth of the module $M(v)$, respectively.

For a set of vertices S we define $w(S) = \sum_{v \in S} w(v)$.

The *weighted width* of a tree decomposition $(\{X_i \mid i \in I\}, T = (I, F))$ of G is $(\max_{i \in I} w(X_i)) - 1$. The *weighted treewidth* $wtw(G)$ of G is the minimum weighted width over all possible tree decompositions of G. Notice that $wtw(G)$ depends just on w and not on t.

For a set $X \in indp(G)$, the *weighted treewidth of G with independent set X* denoted as $wi(G, X)$ is defined by:

$$wi(G, X) = \max\{wtw(G : X), \max_{v \in X}\{t(v) + w(N(v))\}\}.$$

The *weighted independent treewidth* of G (shortly the *wi-treewidth* of G) denoted as $wi(G)$ is defined by:

$$wi(G) = \min_{X \in indp(G)} wi(G, X)$$

For a set $X \in indp(G)$ such that $wi(G) = wi(G, X)$ we say that X *establishes* the wi-treewidth of G. The WI-TREEWIDTH problem is to find $wi(G)$ for a given graph G with weight functions w and t.

In the following text whenever we refer to $wtw(G)$ we assume that the weights of the vertices of G are defined by a weight function w. Similarly whenever we refer to $wi(G)$

we assume that the w-weights and the t-weights of the vertices of G are defined by weight functions w and t, respectively.

Let $V(G) = \{v_1, \ldots, v_r\}$ and let $H_1, \ldots, H_r$ be disjoint graphs. We denote by $G(H_1, \ldots, H_r)$ the graph G' obtained from G by substituting the graph H_i for the vertex v_i, for $1 \leq i \leq r$: $V(G') = V(H_1) \cup \ldots V(H_r)$, $E(G') = E(H_1) \cup \ldots E(H_r) \cup \{\{u,v\} | u \in V(H_i)$ and $v \in V(H_j)$ and $\{v_i, v_j\} \in E(G)\}$.

The following lemma follows from the above definitions:

Lemma 6 *Let G be a graph, where $V(G) = \{v_1, \ldots, v_r\}$. Let $H_1, \ldots, H_r$ be disjoint cliques and let $G' = G(H_1, \ldots H_r)$. Let w be the weight function on the vertices of G defined by: $w(v_i) = |V(H_i)|$, for $1 \leq i \leq r$. Then $treewidth(G') = wtw(G)$.*

Lemma 7 *Let G be a graph, where $V(G) = \{v_1, \ldots, v_r\}$. Let $H_1, \ldots, H_r$ be disjoint graphs and let $G' = G(H_1, \ldots H_r)$. Let w and t be the weight functions on the vertices of G defined by: $w(v_i) = |V(H_i)|$ and $t(v_i) = treewidth(H_i)$, for $1 \leq i \leq r$. Then $treewidth(G') = wi(G)$.*

Proof. We first show that $wi(G) \leq treewidth(G')$. Let $k = treewidth(G')$ and let $\mathcal{T} = (\{X_i \mid i \in I\}, T = (I, F))$ be a tree decomposition of G' of width k. Let $S = \{v_j \mid \neg\exists i(i \in I$ and $V(H_j) \subseteq X_i)\}$. In other words S consists of all vertices v_j such that the set of vertices of the corresponding graph H_j is not included in any of the sets X_i of the tree decomposition $\mathcal{T}$. Let $P = V(G) - S$. For convenience we reorder the vertices of G such that $S = \{v_1, \ldots, v_l\}$ and $P = \{v_{l+1}, \ldots, v_r\}$. We also apply the same reordering to the sets $H_1, \ldots, H_r$ to keep the convention that the graph H_i corresponds to the vertex v_i, for $1 \leq i \leq r$. By Lemma 4 $\mathcal{T}$ is also a tree decomposition of $G_1 = G' + clique(V(H_{l+1}), \ldots, V(H_r))$.

For $v_j \in S$ let $Y_j = \bigcup\{V(H_i) \mid \{v_i, v_j\} \in E(G)\}$. Since $V(H_j)$ is not contained in any of the sets X_i of the tree decomposition $\mathcal{T}$ and for every pair of vertices $u \in V(H_j)$ and $v \in Y_j$ the edge $\{u, v\} \in E(G')$, we get from Lemma 3 that there exists $i \in I$ such that $Y_j \subseteq X_i$. Hence by Lemma 4 $\mathcal{T}$ is also a tree decomposition of $G_1 + clique(Y_j)$. Repeating the above argument for each vertex in S we obtain that $\mathcal{T}$ is also a tree decomposition of $G_2 = G_1 + clique(Y_1, \ldots, Y_l)$. Let $G_3 = G_2 - (V(H_1) \cup \ldots \cup V(H_l))$. Since G_3 is a subgraph of G_2 we have that $treewidth(G_3) \leq k$. We now observe that $G_3 = G : S(H_{l+1}, \ldots, H_r) + clique(V(H_{l+1}), \ldots, V(H_r))$. By Lemma 6 $treewidth(G_3) = wtw(G : S)$ and therefore we have shown that $wtw(G : S) \leq k$.

Let $Q_i = G_2[V(H_i) \cup Y_i]$, for $1 \leq i \leq l$. Clearly $G_2[Y_i]$ is a clique and $Q_i = H_i \times G_2[Y_i]$. By Lemma 5 $treewidth(Q_i) = treewidth(H_i) + |Y_i|$. Since $|Y_i| = w(N(v_i))$ and $treewidth(H_i) = t(v_i)$ we get that $treewidth(Q_i) = t(v_i) + w(N(v_i))$. Since Q_i is a subgraph of G_2 we obtain that $treewidth(Q_i) \leq k$. Hence we have shown that $t(v_i) + w(N(v_i)) \leq k$, for $1 \leq i \leq l$. Thus $wi(G) \leq wi(G, S) = \max\{wtw(G : S), \max_{v \in S}\{t(v) + w(N(v))\}\} \leq k$.

Conversely, we show that $treewidth(G') \leq wi(G)$. Suppose $wi(G) = k$ and let $S \in indp(G)$ be a set of vertices establishing $wi(G)$, i.e., $wi(G) = wi(G, S)$. Let $P = V(G) - S$. Again we reorder the vertices of G such that $S = \{v_1, \ldots, v_l\}$ and $P = \{v_{l+1}, \ldots, v_r\}$. We also apply the same reordering to the sets $H_1, \ldots, H_r$ to keep the convention that the graph H_i corresponds to the vertex v_i, for $1 \leq i \leq r$. Since $wtw(G : S) \leq k$ there exists a tree decomposition $\mathcal{T} = (\{X_i \mid i \in I\}, T = (I, F))$ of $G : S$ of weighted treewidth at most k. Let $G_1 = G : S(H_{l+1}, \ldots, H_r)$. Let $\mathcal{T}' = (\{X'_i \mid i \in I'\}, T' = (I', F'))$, where $X'_i = \bigcup\{V(H_j) \mid v_j \in X_i\}$, $I' = I$ and $F' = F$ be the tree decomposition obtained from $\mathcal{T}$ by replacing each vertex v_j occurring in X_i by the set of vertices $V(H_j)$. Since $w(v_j) = |V(H_j)|$

for $1 \leq j \leq r$ we get that $\mathcal{T}'$ is a tree decomposition of G_1 of width $\leq k$. Notice that G_1 is equal to $G'[V(H_{l+1}) \cup \ldots \cup V(H_r)] + clique(Y_1, \ldots, Y_l)$.

For $1 \leq j \leq l$ let $Y_j = \bigcup\{V(H_i) \mid \{v_i, v_j\} \in E(G)\}$ and let $clique(Y_j)$ denote the complete graph with vertex set Y_j. Since $wi(G) = k$ we have that $t(v_1) + w(N(v_1)) \leq k$ which implies that $treewidth(H_1) + |Y_1| \leq k$. By Lemma 5 we obtain that $treewidth(H_1 \times clique(Y_1)) \leq k$. Hence there exists a tree decomposition $\mathcal{T}^1 = (\{X^1{}_i \mid i \in I^1\}, T^1 = (I^1, F^1))$ for $H_1 \times clique(Y_1)$ of width at most k. By Lemma 2 there exist two nodes $i_1 \in I^1$ and $i' \in I'$ such that Y_j is included both in $X^1_{i_1}$ and in $X'_{i'}$. We define now a new tree decomposition $\mathcal{T}''$ which is obtained by hooking the two trees T' and T^1 to a new node z which is connected to the nodes i_1 and i'. In other words, $\mathcal{T}'' = (\{X''_i \mid i \in I''\}, T'' = (I'', F''))$, where $I'' = I^1 \cup I \cup \{z\}$, $X''_i = X'_i$ if $i \in I'$ or $X''_i = X^1_i$ if $i \in I^1$, $X''_z = Y_j$, and $F'' = F' \cup F^1 \cup \{\{z, i_1\}, \{z, i'\}\}$. It is now easy to see that $\mathcal{T}''$ is a tree decomposition of $G'[V(H_1) \cup V(H_{l+1}) \cup \ldots \cup V(H_r)] + clique(Y_1, \ldots, Y_l)$ of width $\leq k$. Repeating this argument for $j = 2, \ldots, l$ we obtain a tree decomposition of $G' + clique(Y_1, \ldots, Y_l)$ of width $\leq k$. Hence, $treewidth(G') \leq k$. $\square$

We now give more details on the definition of the modular decomposition $M(G) = (T(G), \pi(G))$ of a graph G. $T(G)$ is a labeled tree of which labels are either S (for Series), P (for Parallel) or N (for Neighborhood), and has the following properties:

- The leaves of $T(G)$ are the vertices of G.
- For an internal node h of $T(G)$, the set $M(h)$ of vertices of G occurring in the subtree of $T(G)$ rooted at h forms a module in G.
- Let h be an internal node of $T(G)$ and let $\{h_1, \ldots, h_r\}$ be the the set of sons of h in $T(G)$. If h is labeled S then every vertex of $M(h_i)$ is adjacent to every vertex of $M(h_j)$ for all $1 \leq i, j \leq r$, $i \neq j$. If h is labeled P then no vertex of $M(h_i)$ is adjacent to any vertex of $M(h_j)$ for all $1 \leq i, j \leq r$, $i \neq j$. If h is labeled N then h is associated with a unique prime graph G_h in $\pi(G)$ such that $V(G_h) = \{h_1, \ldots, h_r\}$ and two vertices $u \in M(h_i)$ and $v \in M(h_j)$, for $1 \leq i, j \leq r$, $i \neq j$ are adjacent if and only if $\{h_i, h_j\} \in E(G_h)$.

For more details on the modular decomposition of graphs, see for example [13, 14, 16, 23]. From the above properties it follows that:

Lemma 8 *Let $M(G) = (T(G), \pi(G))$ be the modular decomposition of a graph G. Let h be an internal node of $T(G)$ and let $h_1, \ldots, h_r$ be the sons of h in $T(G)$.*

1. *If h is labeled S then $G[M(h)] = C(G[M(h_1)], \ldots, G[M(h_r)])$, where C denote the complete graph with vertex set $V(C) = \{h_1, \ldots, h_r\}$.*
2. *If h is labeled P then $G[M(h)] = I(G[M(h_1)], \ldots, G[M(h_r)])$, where I denote the edgeless graph with vertex set $V(I) = \{h_1, \ldots, h_r\}$.*
3. *If h is labeled N then $G[M(h)] = G_h(G[M(h_1)], \ldots, G[M(h_r)])$, where G_h is the prime graph in $\pi(G)$ associated with h.*

Recall that we say that G is *modularly decomposable* into $\pi(G)$, where $\pi(G)$ is the set of prime graphs associated with the modular decomposition of G, i.e., $M(G) = (T(G), \pi(G))$. We say that a class of graphs $\mathcal{C}$ is modularly decomposable into a class of graphs $\mathcal{D}$ if every graph $G \in \mathcal{C}$ is modularly decomposable into graphs in $\mathcal{D}$.

Theorem 9 *Let $\mathcal{C}$ and $\mathcal{D}$ be classes of graphs such that $\mathcal{C}$ is modularly decomposable into $\mathcal{D}$. Suppose that the WI-TREEWIDTH problem can be solved in $O(f(n,m))$ time on $\mathcal{D}$, where f is some polynomial function in n and m. Then the TREEWIDTH problem can be solved in $O(f(n,m)+n+m)$ time on $\mathcal{C}$.*

Proof. Given a graph $G \in \mathcal{C}$ we can find $treewidth(G)$ by the following algorithm:

1. Construct the modular decomposition $M(G) = (T(G), \pi(G))$ of G using the algorithm of [14] or [23].

2. Scan $T(G)$ from bottom to top calculating $treewidth(G[M(h)])$ for each internal node h reached according to the following rules. Let $h_1, \ldots, h_r$ be the sons of h in $T(G)$.

   - If h is labeled with P then

   $$treewidth(G[M(h)]) = \max_{1 \le i \le r}\{treewidth(G[M(h_i)])\}.$$

   - If h is labeled with S then

   $$treewidth(G[M(h)]) = \\ min_{1 \le i \le r}\{treewidth(G[M(h_i)]) + |M(h) - M(h_i)|\}.$$

   - If h is labeled with N then obtain $treewidth(G[M(h)])$ from the wi-treewidth of the graph $G_h \in \pi(G)$ corresponding to h, where the weights of the vertices of G_h are defined by $w(h_i) = |M(h_i)|$ and $t(h_i) = treewidth(G[M(h_i)])$. In other words:

   $$treewidth(G[M(h)]) = wi(G_h).$$

   Note that since $G_h \in \mathcal{D}$ we assume that there is an $O(f(|V(G_h)|, |E(G_h)|))$ time algorithm for constructing $wi(G_h)$ and we use this algorithm to get $treewidth(G[M(h)])$ in this case.

3. Let r be the root of $T(G)$, then $treewidth(G) = treewidth(G[M(r)])$.

To prove the correctness of the above algorithm we claim that in step (2) above $treewidth(G[M(h)])$ is correctly obtained for each internal node h reached. If h is labeled with P or S the claim is trivial. If h is labeled N the claim follows from Lemmas 7 and 8.

As for the complexity, calculating the modular decomposition of a graph using [14] or [23] takes $O(n+m)$ time, i.e., step (1) takes $O(n+m)$ time. Noting that the number of nodes in $T(G)$ is $O(V(G))$ and that

$$\sum\{|V(G_h)| \mid h \text{ is an internal node of } T(G)\} = O(V(G)) = O(n) \text{ and}$$

$$\sum\{|E(G_h)| \mid h \text{ is an internal node of } T(G)\} = O(E(G)) = O(m)$$

we get that step (2) takes at most $O(f(n,m))$ time. □

3 Classes with a polynomial number of separators

In this section, we shall prove Theorem 10.

Theorem 10 *Let $\mathcal{C}$ be a class of graphs such that there is a polynomial $p(n)$, such that every graph in $\mathcal{C}$ with n vertices has at most $p(n)$ minimal separators. Then the WI-TREEWIDTH problem on $\mathcal{C}$ can be solved in polynomial time.*

In the remainder of this section, let $\mathcal{C}$ be a class with polynomially many minimal separators, as in the theorem above. First, we note that there is a polynomial time algorithm which given a graph $G \in \mathcal{C}$, lists all the minimal separator of G [20], and there is a polynomial time algorithm which given a graph $G \in \mathcal{C}$, lists all the potential maximal cliques of G [10].

We shall use the definition of treewidth given in the introduction, i.e., $treewidth(G)$ is the smallest clique number of all possible triangulations of G minus 1. Similarly, the weighted treewidth of G, $wtw(G)$ is the minimum weighted clique-number of a triangulation of G minus 1. We say that a triangulation H of G *establishes* the weighted treewidth of G if the weighted clique number of H minus 1 is equal to $wtw(G)$.

Let S be a sets of vertices in G. We denote by $wi_S(G)$ the minimum wi-treewidth of G assuming that all the vertices of S are not included in the independent set. Formally,

$$wi_S(G) = \min\{wi(G, X) \mid X \in indp(G) \text{ and } S \cap X = \emptyset\}.$$

For S, a set of vertices of a graph G, we denote by G_S the graph $G + clique(S)$, i.e., G_S is the graph obtained by adding edges to G such that all the vertices of S form a clique. For C, a connected component of $G - S$, we let $G_S(C)$ denote the subgraph of G_S induced by $S \cup C$, i.e., $G_S(C)$ is equal to $G_S[S \cup C]$.

We start with some lemmas which are true for any graph G.

Lemma 11 *Let K be a clique in G and let $C_1, \dots, C_t$ be the connected components of $G - K$. Then*

$$wi_K(G) = \max_{1 \le i \le t}\{wi_K(G_K(C_i))\}.$$

Proof. Clearly, $wi_K(G) \le \max_{1 \le i \le t}\{wi_K(G_K(C_i))\}$. Let $X \in indp(G)$ be a set of vertices establishing $wi_K(G)$ and let $G' = G : X$. By definition: $wi_K(G) = \max\{wtw(G'), max_{v \in X}\{t(v) + w(N(v))\}\}$. Let X_i be the set of all vertices of X which are either in C_i or have a neighbor in C_i. For $i \ne j$, $X_i \cap X_j = \emptyset$ or else there is a path from C_i to C_j in $G - K$, a contradiction. Hence $X_1, \dots, X_t$ form a partition of X. Since $G'[C_i - X]$ is connected, and for $i \ne j$ no vertex of X has a neighbor in C_i and in C_j it follows that $C_1 - X, \dots, C_t - X$ are the connected components of $G' - K$. Thus $wtw(G') = \max\{wtw(G'_K(C_i - X)) : 1 \le i \le t\}$. Thus selecting the set X_i in the graph $G_K(C_i)$ we obtain:

$$wi_K(G_K(C_i)) \le \max\{wtw(G'_K(C_i - X)), \max_{v \in X_i}\{t(v) + w(N(v))\}\}.$$

Hence,

$$\begin{aligned}&\max_{1 \le i \le t}\{wi_K(G_K(C_i))\} \le \\ &\qquad \max\{\max_{1 \le i \le t}\{wtw(G'_K(C_i - X))\}, \max_{v \in X}\{t(v) + w(N(v))\}\} = \\ &\qquad wi_K(G).\end{aligned}$$

□

The proof of Lemma 12 is similar to the proof of Lemma 11.

Lemma 12 *Let K be a clique in G, let C be a connected component of $G-K$ and let K^C be the set of all vertices in K having a neighbor in C. Then*

$$wi_K(G_K(C)) = \max\{w(K)-1, wi_{K^C}(G_{K^C}(C))\}.$$

Lemma 13 *Let K be a clique in G, and let $C_1,\ldots,C_t$ be the connected components of $G-K$. For $1 \le i \le t$ let K_i be the set of all vertices in K having a neighbor in C_i. Then*

$$wi_K(G) = \max\{w(K)-1, \max_{1\le i\le t}\{wi_{K_i}(G_{K_i}(C_i))\}\}.$$

Proof. Immediate from Lemmas 11 and 12. □

Lemma 14 *Let X be an independent set in G establishing $wi(G)$. Let S be a clique in a minimal triangulation Q of $G:X$ that establishes wtw(G:X). Let $C_1,\ldots,C_t$ be the connected components of $G-S$. Then*

$$wi(G) = \max_{1\le i\le t}\{wi_S(G_S(C_i))\}.$$

Proof. Since X establishes $wi(G)$ we have:

$$wi(G) = \max\{wtw(G:X), \max_{v\in X}\{t(v)+w(N(v))\}\}.$$

Since S is a clique in Q, $wtw(G:X) = wtw((G:X)+clique(S)) = wtw((G:X)_S)$. Likewise, for every $v \in X$, the neighborhood of v in G is the same as the neighborhood of v in G_S. Hence $wi(G) = wi_S(G) = wi_S(G_S)$. Now the formula follows from Lemma 11. □

Definition Let $S \subset V$ and let C be a connected component of $G-S$. We say that S is *close* to C if every vertex of S has a neighbor in C. In this case we say also that C is a *full component* of S.

The following lemma is due to [18]:

Lemma 15 *A set S of vertices is a minimal separator of a graph G if and only if there exists two connected components C_1 and C_2 of $G-S$ such that S is close to C_1 and to C_2.*

Lemma 16 *Let X be an independent set in G. Let $G' = G:X$ and let S be a minimal separator of G'. Let $C'_1,\ldots,C'_t$ be the connected components of $G'-S$. For $1\le i\le t$ let $X_i = \{v \mid v\in X$ and $N(v)\cap C'_i \ne \emptyset\}$. Let $U = \{v \mid v \in X$ and $N(v) \subseteq S\}$ and let $v_1,\ldots,v_l$ be the vertices in U. Then S is a minimal separator of G and the connected components of $G-S$ are $C'_1\cup X_1,\ldots,C'_t\cup X_t,\{v_1\},\ldots,\{v_l\}$.*

Proof. For $i\ne j$ $X_i\cap X_j=\emptyset$, or else there exists an edge in G' from some vertex in C'_i to some vertex in C'_j, a contradiction. Hence $X_1,\ldots,X_t,U$ form a partition of X. Clearly, $\{v_1\},\ldots,\{v_l\}$ are connected components of $G-S$.

Claim 17 *For $1\le i\le t$ $G[C'_i\cup X_i]$ is a connected component of $G-S$.*

Proof. We first show that $G[C'_i \cup X_i]$ is connected. Let u and v be any two vertices in C'_i. In $G'[C'_i]$ there is a path P connecting u and v. If all the edges of P occur in G then P connects u and v in $G[C'_i]$. If not all the edges of P occur in G then we can replace any such edge, say $e = \{w, t\}$, by the two edges $\{w, x\}, \{x, t\}$ occurring in $G[C'_i \cup X_i]$, where x is the vertex of X_i which is adjacent to both w and t. Thus, we can use P to construct a path connecting u and v in $G[C'_i \cup X_i]$. We have shown that for every two vertices $u, v \in C'_i$ u and v are connected in $G[C'_i \cup X_i]$. Since every vertex in X_i has a neighbor in C'_i, it follows that for every two vertices $u, v \in C'_i \cup X_i$ u and v are connected in $G[C'_i \cup X_i]$. Thus, $G[C'_i \cup X_i]$ is connected. Suppose that $G[C'_i \cup X_i]$ is not a connected component of $G - S$. Then there exist two vertices $u \in C'_i \cup X_i$ and $v \in C'_j \cup X_j$, $i \neq j$, such that u and v are connected by a path P in $G - S$. If all the edges of P occur also in $G' - S$ then C'_i is connected to C'_j in $G' - S$, a contradiction. The edges of P which do not occur in $G' - S$ must be of the form $\{x, w\}, \{x, t\}$ for some vertex $x \in X_i$. These two edges can be replaced by the edge $\{w, t\}$ occurring in $G' - S$. Hence we can use P to construct a path connecting C'_i to C'_j in $G' - S$, a contradiction. □

Since S is a minimal separator in G' by Lemma 15 there exist two connected components C'_i and C'_j such that S is close to C'_i and to C'_j in G'. Hence in G S is close to $C'_i \cup X_i$ and to $C'_j \cup X_j$. Thus, by Lemma 15 S is a minimal separator in G. □

The following lemma is due to [21].

Lemma 18 *Let Q be a minimal triangulation of a graph G, let S be a minimal separator of Q and let $C_1, \ldots, C_t$ be the connected components of $Q - S$. Then S is a minimal separator of G and $C_1, \ldots, C_t$ are also the connected components of $G - S$.*

Lemma 19 *Let X be an independent set in G. Let $G' = G : X$ and let Q be a minimal triangulation of G'. Let S be a minimal separator of Q and let $C'_1, \ldots, C'_t$ be the connected components of $Q - S$. For $1 \leq i \leq t$ let $X_i = \{v \mid v \in X \text{ and } N(v) \cap C'_i \neq \emptyset\}$. Let $U = \{v \mid v \in X \text{ and } N(v) \subseteq S\}$ and let $v_1, \ldots, v_l$ be the vertices in U. Then S is a minimal separator of G' and the connected components of $G' - S$ are $C'_1, \ldots, C'_t$. Moreover, S is a minimal separator of G and the connected components of $G - S$ are $C'_1 \cup X_1, \ldots, C'_t \cup X_t, \{v_1\}, \ldots, \{v_l\}$.*

Proof. By Lemma 18 S is a minimal separator of G' and the connected components of $G' - S$ are $C'_1, \ldots, C'_t$. By Lemma 16 S is a minimal separator of G and the connected components of $G - S$ are $C'_1 \cup X_1, \ldots, C'_t \cup X_t, \{v_1\}, \ldots, \{v_l\}$. □

We say that v is a *universal* vertex in G if $N(v) = V(G) - \{v\}$. We denote by $univ(G)$ the set of all universal vertices in G.

Lemma 20 *For every graph G with weight functions w and t,*

$$\begin{aligned} wi(G) = \min\{&\min\{t(v) + w(N(v)) \mid v \in univ(G)\}, \\ &\min_{S \in msep(G)} \{\max\{wi_S(G_S(C)) \mid C \in cc(G - S)\}\}\}. \end{aligned} \tag{1}$$

Proof. We show that the right hand side of the above formula is less than or equals to $wi(G)$ (the other direction is immediate). Let X be an independent set in G establishing

$wi(G)$. Let $G' = G : X$. If $univ(G) \cap X \neq \emptyset$ then X consists of a single vertex (say v) and $wi(G) = t(v) + w(N(v))$. Hence, we assume that $univ(G) \cap X = \emptyset$.

Suppose G' is a clique. Let v be any vertex in X. Let $S = N(v)$. Since $v \notin univ(G)$ and G' is a clique S is a minimal separator in G. Clearly G' is triangulated. By Lemma 14 $wi(G) = \max\{wi_S(G_S(C)) \mid C \in cc(G - S)\}$.

Suppose G' is not a clique. Let Q be a minimal triangulation of G' that establishes $wtw(G')$. Clearly Q is not a clique. Hence, there exist a minimal separator S in Q. By Lemma 19 S is a minimal separator in G. Since every minimal separator in a chordal graph is a clique [17], S is a clique in Q. Hence, by Lemma 14 $wi(G) = \max\{wi_S(G_S(C)) \mid C \in cc(G - S)\}$. □

For a set $S \in msep(G)$ and a set $C \in cc(G - S)$ we denote by S^C the set of vertices in S having a neighbor in C.

Lemma 21 *Let $S \in msep(G)$ and let $C \in cc(G - S)$. Then $S^C \in msep(G)$, S^C is close to C and*

$$wi_S(G_S(C)) = \max\{w(S) - 1, wi_{(S^C)}(G_{S^C}(C))\} \tag{2}$$

Proof. The claim that $S^C \in msep(G)$ and S^C is close to C was proved in [11]. Formula 2 follows from Lemma 12. □

From Lemmas 20 and 21 it follows that for every graph G, we can calculate $wi(G)$ by calculating $wi_S(G_S(C))$ for every pair (S, C) where S is a minimal separator of G and C is a connected component of $G - S$ such that S is close to C (i.e., C is a full component of S). For an arbitrary graph the number of such pairs may be exponential. However, if G is a weakly chordal graph the number of such pairs is polynomial and there exists a polynomial time algorithm for obtaining all these pairs. Our goal now is to show that for all pairs (S, C) such that S is a minimal separator of G and C is a full component of S, $wi_S(G_S(C))$ can be calculated in polynomial time on every class of graph $\mathcal{C}$ which satisfy the condition mentioned in Theorem 10. The key lemma for achieving this goal is Lemma 25, in which we shall use the following definitions and lemmas.

Definition Let $S \subset V$ and let x be a vertex of G which is not included in S. We say that x is *universal* to S in G, if x is adjacent to all the vertices of S in G. We let $univ(S)$ be the set of all vertices of G which are universal to S.

The following lemma is due to [9].

Lemma 22 *Let G be a chordal graph and let K be a maximal clique in G. Then K has no full component. In other words, for every connected component C of $G - K$, K is not close to C.*

The following lemma is due to [22].

Lemma 23 *Let S be a minimal separator of G, let $C_1, \ldots, C_t$ be the connected components of $G - S$, and let Q_i be a minimal triangulation of $G_S(C_i)$, for $1 \leq i \leq t$. Then the graph $\bigcup_{i=1}^{i=t} Q_i$ is a minimal triangulation of G.*

Lemma 24 *Let S be a minimal separator of G, let C be a connected component of $G - S$ and let Ω be a potential maximal clique of $G_S(C)$ such that $\Omega \cap C \neq \emptyset$. Then Ω is a potential maximal clique of G.*

Proof. Let $C_1, \ldots, C_t$ be the connected components of $G - S$. Assume without loss of generality that $C = C_1$. Since Ω is a potential maximal clique of $G_S(C_1)$ there exist a minimal triangulation Q_1 of $G_S(C_1)$ such that Ω is a maximal clique in Q_1. For $2 \leq i \leq t$ let Q_i be a minimal triangulation of $G_S(C_i)$. Be Lemma 23 the graph $Q = \bigcup_{i=1}^{i=t} Q_i$ is a minimal triangulation of G. Since for $i \neq j$ there is no edge in Q between a vertex in $V(Q_i) - S$ to a vertex in $V(Q_j) - S$, it follows that Ω is also a maximal clique in Q. $\square$

Recall that $potm(G)$ denote the set of all potential maximal cliques of a graph G.

Lemma 25 *Let S be a minimal separator of G and let C be a full component of S, such that $G_S(C)$ is not a clique, then*

$$\begin{aligned} wi_S(G_S(C)) = \min\{wi_\Omega(G_\Omega[S \cup C]) \mid \ & \textit{either } S \subset \Omega \subseteq S \cup C \\ & \textit{and } \Omega \in potm(G) \textit{ or there exist } x \in C \cap univ(S) \\ & \textit{such that } \Omega = N(x)\}. \end{aligned} \tag{3}$$

Proof. We first show that the left-hand side of Formula 3 is less then or equals to the right-hand side of this formula. Let Ω be a set of vertices of G reaching the minimum value for the right-hand side of Formula 3, i.e., the right-hand side of Formula 3 is equal to $wi_\Omega(G_\Omega[S\cup C])$. Let X be an independent set establishing $wi_\Omega(G_\Omega[S \cup C])$, and let $H = G_\Omega[S \cup C] : X$. By definition:

$$wi_\Omega(G_\Omega[S \cup C]) = \max\{wtw(H), \max_{v \in X} t(v) + w(N(v))\}.$$

$$\begin{aligned} wi_S(G_S(C)) \leq & wi(G_S(C), X) = \\ & \max\{wtw(G_S(C) : X), \max_{v \in X} t(v) + w(N(v))\}. \end{aligned}$$

The graph H can be obtained from the graph $G_S(C) : X$ by adding edges such that all the vertices in Ω form a clique. Thus, every triangulation of H is also a triangulation of $G_S(C) : X$ which implies that $wtw(G_S(C) : X) \leq wtw(H)$. It follows that

$$wi_S(G_S(C)) \leq \max\{wtw(H), \max_{v \in X} t(v) + w(N(v))\} = wi_\Omega(G_\Omega[S \cup C]).$$

We now show that the right-hand side of Formula 3 is less then or equals to the left-hand side of this formula. Let X be an independent set establishing $wi_S(G_S(C))$ and let $F = G_S(C) : X$. By definition:

$$wi_S(G_S(C)) = \max\{wtw(F), \max_{v \in X} t(v) + w(N(v))\}.$$

Suppose there exists $x \in X$, such that $x \in univ(S)$. Let $\Omega = N(x)$ and let $F' = G_\Omega[S\cup C] : X$. By definition:

$$\begin{aligned} wi_\Omega(G_\Omega[S \cup C]) \leq & wi(G_\Omega[S \cup C], X) = \\ & \max\{wtw(F'), \max_{v \in X} t(v) + w(N(v))\}. \end{aligned}$$

Since Ω is a clique in F it follows that the graph F is equal to the graph F'. Replacing F' with F in the above formula we obtain that $wi_\Omega(G_\Omega[S \cup C]) \leq wi_S(G_S(C))$. Thus, we have shown that if $X \cap univ(S) \neq \emptyset$ then the right-hand side of Formula 3 is less then or equals to the left-hand side of this formula.

Suppose that there is no $x \in X$ such that $x \in univ(S)$. Let Q be a minimal triangulation of F establishing $wtw(F)$. Let Ω be the maximal clique of Q containing S and let $F' = G_\Omega[S \cup C] : X$. Since $\Omega \cap X = \emptyset$ we get that:

$$\begin{aligned} wi_\Omega(G_\Omega[S \cup C]) \leq & wi(G_\Omega[S \cup C], X) = \\ & \max\{wtw(F'), \max_{v \in X} t(v) + w(N(v))\}. \end{aligned}$$

Since Q is a supergraph of F', $wtw(Q) \geq wtw(F')$. Now since $wtw(Q) = wtw(F)$, we get that $wtw(F) \geq wtw(F')$. Replacing F' with F in the above formula we obtain that $wi_\Omega(G_\Omega[S \cup C]) \leq wi_S(G_S(C))$. In Claims 27 and 28 we show that $S \subset \Omega$ and Ω is a potential maximal clique of G. Thus, we have shown that if $X \cap univ(S) = \emptyset$ the right-hand side of Formula 3 is less then or equals to the left-hand side of this formula.

Thus, the proof of Lemma 25 is completed by proving the following three claims.

Claim 26 *$C - X$ is a full component of S in F.*

Proof. We first show that the graph $F[C - X]$ is connected. Let u and v be two vertices in $C - X$. Since C is a connected component of $G - S$, there exists a path P in $G[C]$ from u to v. Let P' be the path obtained by omitting all the vertices of X from P. Since for every vertex x of P which is in X its predecessor and its successor on P are not in X and are connected by an edge in F, it follows that P' is a path from u to v in $F[C - X]$. Thus, $F[C - X]$ is connected.

Let v be a vertex in S. Since C is a full component of S in G, there is a vertex $u \in C$ such that v is adjacent to u in G. If $u \in X$ then let a be a vertex in $C - X$ such that u is adjacent to a in G. Note that since X is an independent set and C contains more than one vertex (or else $G_S(C)$ will be a clique), such a vertex a must exists. Now since a and v are both adjacent to u in G and $u \in X$ it follows that a and v are adjacent in F. We have shown that $F[C - X]$ is connected and that for every vertex $v \in S$ there exists a vertex in $C - X$ which is adjacent to v in F. Thus, $C - X$ is a full component of S in F. □

Claim 27 *$S \subset \Omega$.*

Proof. By Claim 26 $C - X$ is a full component of S in F. Since Q is obtained by adding some edges to F, none of which connects $C - X$ to other connected components of S in F, it follows that $C - X$ is a full component of S in Q. By Lemma 22 no maximal clique of a chordal graph has a full component. Since Q is chordal, it follows that S is not a maximal clique in Q. Thus, S is a proper subset of the maximal clique Ω in Q which contains S. □

Claim 28 *Ω is a potential maximal clique of G.*

Proof. Let Q' be the graph obtained from Q by adding all the vertices of X and all the edges connecting the vertices of X and their neighbors in G, i.e., $V(Q') = V(Q) \cup X$ and $E(Q') = E(Q) \cup \{\{x, v\} \mid x \in X \text{ and } v \in N(x)\}$. Since Q is chordal and for every vertex $x \in X$, $N(x)$ form a clique in Q, it follows that Q' is chordal. Thus, Q' is a triangulation of

$G_S(C)$. Let Q'' be the graph obtained from Q' by omitting edges which are not in $G_S(C)$ (chosen arbitrarily), such that Q'' is a minimal triangulation of $G_S(C)$. We now show that Ω is a clique in Q''.

Let u and v be two vertices in Ω. If u and v are adjacent in $G_S(C)$ then u and v are also adjacent in Q''. Suppose u and v are not adjacent in $G_S(C)$. Clearly, u and v are adjacent in Q. Suppose u and v are not adjacent in Q''. If the removal of the edge $\{u, v\}$ from Q from a chordless cycle, then this cycle is also chordless in Q'', a contradiction. Thus, the removal of the edge $\{u, v\}$ from Q does not form a chordless cycle. Since Q is a minimal triangulation of F, and the edge $\{u, v\}$ was not removed from Q, it follows that this edge is in F. Thus, u and v are adjacent in F but are not adjacent in $G_S(C)$, which implies that there exists a vertex $x \in X$ such that u and v are both adjacent to x in $G_S(C)$. Since we assume that x is not adjacent in $G_S(C)$ (and therefore x is not adjacent in Q'') to all the vertices of S, there exists a vertex y in S such that x is not adjacent to y in Q''. Now x, y, u, v form a chordless cycle in Q'', a contradiction.

We have shown that Ω is a clique in Q''. Now since Q'' is a minimal triangulation of $G_S(C)$, we obtain that Ω is a potential maximal clique in $G_S(C)$. By Lemma 24 Ω is also a potential maximal clique of G. □

As mentioned above, this completes the proof of Lemma 25. □

Using Formula 3 we can calculate $wi_S(G_S(C))$ by taking the minimum value of $wi_\Omega(G_\Omega[S\cup C])$ for all Ω such that either $\Omega \in potm(G)$ and $S \subset \Omega \subseteq S \cup C$ or there exist $x \in C \cap univ(S)$ such that $\Omega = N(x)$. But how can we calculate the values of $wi_\Omega(G_\Omega[S \cup C])$ for all these Ω's? If $\Omega \in potm(G)$ then the answer to this question is given in Formula 4 of Lemma 29. If $\Omega = N(x)$ for some $x \in C \cap univ(S)$ then the answer to this question is given in Formula 5 of Lemma 31.

Lemma 29 *Let $S \in msep(G)$ and let C be a connected component of $G - S$. Let Ω be a potential maximal clique of G such that $S \subset \Omega \subseteq S \cup C$. Let $C_1, \ldots, C_t$ be the connected components of $G_S(C) - \Omega$, and for $1 \le i \le t$ let Ω_i denote the set of all vertices in Ω having a neighbor (in the graph G) in C_i. Then for $1 \le i \le t$, $\Omega_i \subset \Omega$, $\Omega_i \in msep(G)$ and*

$$wi_\Omega(G_\Omega[S \cup C]) = \max\{w(\Omega) - 1, \max_{1\le i\le t}\{wi_{\Omega_i}(G_{\Omega_i}(C_i))\}\}. \tag{4}$$

Proof. Since Ω be a potential maximal clique of G such that $S \subset \Omega \subseteq S \cup C$, it is clear that Ω is also a potential maximal clique of $G_S(C)$. Let Q be a minimal triangulation of $G_S(C)$ such that Ω is a maximal clique in Q. Suppose that for some $1 \le i \le t$ $\Omega_i = \Omega$. It follows that C_i is a full component of Ω in Q. By Lemma 22 no maximal clique in Q has a full component, a contradiction. Thus, we have shown that for $1 \le i \le t$, $\Omega_i \subset \Omega$. The correctness of Formula 4 follows from Lemma 13. A proof of the claim that $\Omega_i \in msep(G)$ is given in Lemma 5 of [9]. We now present a different proof of this claim. For $1 \le i \le t$, let x_i be some vertex in $\Omega - \Omega_i$ and let y_i be some vertex in C_i. Clearly, Ω_i is a minimal x_i, y_i separator in Q.

Claim 30 *Ω_i is a minimal x_i, y_i separator in G_{Ω_i}.*

Proof. Since Ω_i is a minimal x_i, y_i separator in Q and Q is obtained by adding some more edges to G_{Ω_i}, we get that Ω_i separates x_i and y_i in G_{Ω_i}. Let a be a vertex in Ω_i. Since C_i is a full component of Ω_i there is a path P_1 from a to y_i such that besides a all the vertices of the path are in C_i. We now show that there exists a path P_2 connecting a and x_i such that besides a all the vertices of P_2 are not included in $\Omega_i \cup C_i$.

We shall use the property that the graph G_{Ω_i} can be obtained from Q by removing the edges of $E(Q) - E(G_{\Omega_i})$ one after the other (in an arbitrary order) such that each time an edge $\{u, v\}$ is removed a new chordless cycle containing u and v is formed.

Since Ω is a clique in Q, x_i and a are adjacent in Q. If the edge $\{x_i, a\}$ belongs to G_{Ω_i} then we can take $P_2 = \{x, a\}$. Suppose that the edge $\{x_i, a\}$ does not belong to G_{Ω_i}. Removing the edge $\{x_i, a\}$ from Q we obtain a graph Q' such that there is a chordless cycle in Q' containing $\{x_i, a\}$. This cycle can be considered as two disjoint paths from x_i to a. It follows that at least one of these two paths (say R') does not include any vertex of Ω_i besides a. Since Ω_i separates x_i from y_i, it follows that R' does not include any vertex of C_i. Thus R' is a path from x_i to a in Q' which does not include any vertex of $\Omega_i \cup C_i$ besides a. If all the edges of the path R' are in G_{Ω_i} then we can take $P_2 = R'$.

Suppose not all the edges of the path R' are in G_{Ω_i}. Let $\{u, v\}$ be any edge on R' which is not in G_{Ω_i} and let Q'' be the graph obtained from Q' by removing the edge $\{u, v\}$. By a similar argument we can obtain a path R_1 connecting the vertices u and v in Q'' which does not include any vertex of $\Omega_i \cup C_i$ besides a. Thus, there exist a path R'' from x_i to a in Q'', such that all the vertices of R'' are included in $R_1 \cup R'$. If all the edges of R'' are in G_{Ω_i} then we can take $P_2 = R''$. Otherwise, we repeat the above process until we finally obtain a path R from u to v in G_{Ω_i} which does not include any vertex $\Omega_i \cup C_i$ besides a, and we take $P_2 = R$. Note that the above process will terminate since after each iteration we remove from the considered graph one more edge of $E(Q) - E(G_{\Omega_i})$ and thus (in the worst case) we will finally reach the graph G_{Ω_i}.

Let P be the path obtained from the union of the paths P_1 and P_2. Thus, P is a path in G_{Ω_i} from x_i to y_i through a which does not include any vertex of Ω_i besides a. It follows that $\Omega_i - \{a\}$ does not separates x_i and y_i in G_{Ω_i}. Since a was chosen as an arbitrary vertex in Ω_i, we obtain that Ω_i is a minimal x_i, y_i separator in G_{Ω_i}. □

By Claim 30 there exists two vertices x_i and y_i such that Ω_i separates x_i and y_i in G. Let a be any vertex in Ω_i. By Claim 30 there exists a path P from x_i to y_i through a in G_{Ω_i} which does not include any vertex of Ω_i besides a. Since no edge of P connects two vertices of Ω_i, all the edges of P are included in G. Thus, that $\Omega_i - \{a\}$ does not separates x_i and y_i in G. Since a was chosen as an arbitrary vertex in Ω_i, we obtain that Ω_i is a minimal x_i, y_i separator in G. □

Lemma 31 *Let $S \in msep(G)$ and let C be a connected component of $G - S$. Let x be a vertex in $C \cap univ(S)$ and let $\Omega = N(x)$. Let $C_1, \ldots, C_t$ be the connected components of $G_S(C) - \Omega$, and for $1 \leq i \leq t$ let Ω_i denote the set of all vertices in Ω having a neighbor (in the graph G) in C_i. Then Formula 5 holds and for $1 \leq i \leq t$, either $G_{\Omega_i}(C_i)$ is a clique or $\Omega_i \in msep(G)$ and $\Omega_i \cup C_i \subset S \cup C$.*

$$wi_{\Omega}(G_{\Omega}[S \cup C]) = \max\{w(\Omega) - 1, \max_{1 \leq i \leq t}\{wi_{\Omega_i}(G_{\Omega_i}(C_i))\}\}. \tag{5}$$

Proof. The correctness of Formula 5 follows from Lemma 13. Suppose $G_{\Omega_i}(C_i)$ is not a clique. Clearly, one of the connected components of $G_S(C) - \Omega$ is $\{x\}$. Let C_j denote this connected component. Since $G_{\Omega_i}(C_i)$ is not a clique we obtain that $C_i \neq C_j$, (i.e., $i \neq j$). Thus, x is not in $\Omega_i \cup C_i$ which implies that $\Omega_i \cup C_i \subset S \cup C$. Let y be any vertex in C_i.

We claim that Ω_i is a minimal x, y separator in G. Since C_i is a connected component of $G_S(C) - \Omega$ it follows that any path from y to x in G must go through a vertex in Ω. Suppose there is a path from y to x which does not go through any vertex in Ω_i. It follows that there exists a vertex $z \in \Omega$ on this path adjacent to some vertex in C_i. But this implies that z must be in Ω_i, a contradiction. Thus, Ω_i is an x, y separator in G. Let a be any vertex in Ω_i. Since $a \in \Omega_i$ there is a neighbor $w \in C_i$ of a in G. Since $G[C_i]$ is connected there is a path P_1 from w to y such that all the vertices of P_1 are in C_i. Now the path $P = x, a, P_1$, goes in G from x to y through a and does not include any vertex in Ω_i besides a. It follows that $\Omega_i - \{a\}$ does not separates x and y in G. Since a was chosen as an arbitrary vertex in Ω_i, it follows that Ω_i is a minimal x, y separator in G. $\square$

We are now ready to present the main steps of the algorithm for calculating $wi(G)$ for a given graph $G \in \mathcal{C}$ with weight functions w and t, where $\mathcal{C}$ is a class of graphs satisfying the conditions mentioned in Theorem 10.

1. Compute and store in some data structure $\mathcal{D}$ the set of all pairs (S, C) such that S is a minimal separator in G and C is a connected component of $G - S$.

2. Compute and store in another data structure $\mathcal{D}'$ the set of all pairs (Ω, C) such that Ω is a potential maximal clique in G and C is a connected component of $G - \Omega$.

3. Order the elements in $\mathcal{D}$ with respect to increasing size of $|S| + |C|$. Now in this order calculate for the elements (S, C) in $\mathcal{D}$ the value of $wi_S(G_S(C))$ as follows:

   (a) If S is not close to C then let S^C be the set of vertices in S having a neighbor in C. By Lemma 21 S^C is a minimal separator in G. Thus, calculate $wi_S(G_S(C))$ using Formula 2.

   (b) If $G_S(C)$ is a clique then $wi_S(G_S(C))$ is equal to $\min_{v \in C} t(v) + w(S \cup C - \{v\})$.

   (c) If S is close to C and $G_S(C)$ is not a clique then calculate $wi_S(G_S(C))$ using Formula 3 by taking the minimum of $wi_\Omega(G_\Omega[S \cup C])$ for all the Ω's indicated in this formula. Let Y denote the set of all these Ω's. To construct Y first add to Y the set of vertices $\Omega = N(x)$, for each vertex $x \in C$ which is universal to S. Then use the data structure $\mathcal{D}'$ to find all the potential maximal cliques Ω of G such that $S \subset \Omega \subseteq S \cup C$, and add all these Ω's to Y. Now for each Ω in Y calculate the value of $wi_\Omega(G_\Omega[S \cup C])$ as follows:

      - If Ω is a potential maximal clique of G such that $S \subset \Omega \subseteq S \cup C$, calculate $wi_\Omega(G_\Omega[S \cup C])$ using Formula 4. Note that the values of $wi_{\Omega_i}(G_{\Omega_i}(C_i))$ mentioned in Formula 4 have already been calculated by previous steps of the algorithm since $\Omega_i \in msep(G)$, C_i is a connected component of $G - \Omega_i$ and $\Omega_i \cup C_i \subset S \cup C$.
      - If there exists a vertex x in $C \cap univ(S)$ such that $\Omega = N(x)$, calculate $wi_\Omega(G_\Omega[S \cup C])$ using Formula 5. Note that the values of $wi_{\Omega_i}(G_{\Omega_i}(C_i))$ mentioned in Formula 5 can be calculated as follows. If $G_{\Omega_i}(C_i)$ is not a clique then $wi_{\Omega_i}(G_{\Omega_i}(C_i))$ have already been calculated by previous steps of the algorithm since $\Omega_i \in msep(G)$, C_i is a connected component of $G - \Omega_i$ and $\Omega_i \cup C_i \subset S \cup C$. If $G_{\Omega_i}(C_i)$ is a clique then $wi_{\Omega_i}(G_{\Omega_i}(C_i))$ is equal to $\min_{v \in C_i} t(v) + w(\Omega_i \cup C_i - \{v\})$.

4. Now based on the value of $wi_S(G_S(C))$ for all pairs (S, C) in $\mathcal{D}$ calculate $wi(G)$ using Formula 1.

The correctness of the above algorithm follows from Lemmas 20,21,25, 29 and 31. From the assumptions on $\mathcal{C}$ it is clear that the above algorithm has a polynomial time complexity. Thus, we have shown that Theorem 10 holds.

We now analyze the complexity of the above algorithm as a function of the following parameters:

- let n and m denote the number vertices and edges in the input graph and assume that $m \geq n$.
- let s and p be the number of minimal separators and potential maximal cliques in the input graph, respectively. We assume that $p \geq n$.
- let the time complexity of the algorithms for listing all the minimal separators and all the potential maximal cliques of the input graph, be $O(s')$ and $O(p')$ respectively.

Constructing all the pairs (S, C) in the structure $\mathcal{D}$ can be done in $O(\max\{sm, s'\})$ time. The number of such pairs is $O(sn)$. Similarly, constructing all the pairs (Ω, C) in the structure $\mathcal{D}'$ can be done in $O(\max\{pm, p'\})$ time and the number of such pairs is $O(pn)$. Ordering the elements in $\mathcal{D}$ with respect to increasing size of $|S| + |C|$ can be done easily in $O(s(n^2))$ time using bucket sort (i.e., putting the elements of $\mathcal{D}$ in buckets $1, \ldots, n$ depending on their sizes).

Now for each pair (S, C) in $\mathcal{D}$ the algorithm scans all the potential maximal cliques and for each potential maximal clique Ω considers the pairs (Ω_i, C_i) in $\mathcal{D}'$. Clearly, all these pairs can be constructed in $O(pe_C)$, where e_C denotes the number of edges in $G[C]$. Summarizing on all the connected components of S, we obtain that the total time spent by the algorithm for S is $O(pm)$ and thus the total time spent for scanning all the pairs in $\mathcal{D}$ is $O(spm)$. We obtain that the total time complexity of the algorithm is $O(\max\{s', p', spm\})$. Note that $s' \leq n^3 s$ [4] and that $p' \leq n^2 m s^2$ [10].

For the class of weakly chordal graphs, it was shown in [9] that the $s = O(m)$ and $p = O(m)$ and that $spm > \max\{s', p'\}$. Thus, we have shown:

Corollary 32 *The WI-TREEWIDTH problem on the class of weakly chordal graphs can be solved in $O(m^3)$ time.*

Now by Theorem 9 we get:

Theorem 33 *Let $\mathcal{C}$ be a class of graphs which is modularly decomposable into the class of weakly chordal graphs. Then the TREEWIDTH problem on $\mathcal{C}$ can be solved in $O(m^3)$ time.*

Another easy consequence of the results in this section is:

Corollary 34 *Let $\mathcal{C}$ be a class of graphs which is modularly decomposable into graphs with at most $c \log n$ vertices, for some constant c. Then the TREEWIDTH problem on $\mathcal{C}$ can be solved in polynomial time.*

4 Cycles

In this section, we show that for the class of graphs $\mathcal{C}$ which is modularly decomposable into the class of cycles, the TREEWIDTH problem on $\mathcal{C}$ can be solved in linear time. Note that a polynomial (but not linear) time algorithm can be derived with the results of the previous section. We shall need the following definition.

Lemma 35 *Let G be a weighted cycle with weight function w. Then*

$$wtw(G) = \min_{v \in V(G)}\{w(v)\} + \max_{\{v,x\}\in E(G)}\{w(v) + w(x)\} - 1.$$

Proof. Suppose $v_0, v_1, \ldots, v_{n-1}$ are the successive vertices in the cycle G, and without loss of generality, suppose $w(v_0) = \min_{v\in V}\{w(v)\}$.

First, we give a construction that realizes the required weighted treewidth. Now, take a tree, that is actually a path, with successive nodes $i_1, i_2, \ldots, i_{n-2}$, and take $X_{i_j} = \{v_j, v_{j+1}, v_0\}$. One can verify that this is a tree decomposition of G with weighted treewidth exactly $w(v_0) + \max_{\{v,x\}\in E(G)}\{w(v) + w(x)\} - 1$.

We now show that the weighted treewidth of G is at least $w(v_0) + \max_{\{v,x\}\in E(G)}\{w(v) + w(x)\} - 1$. Let $(\{X_i \mid i \in I\}, T = (I, F))$ be a tree decomposition of G. Now, suppose $w(v_j) + w(v_{j+1}) = \max_{\{v,x\}\in E(G)}\{w(v) + w(x)\}$. Clearly v_i and v_{j+1} must be included in X_i for some $i \in I$. If there is no other vertex of G in X_i then it follows that the graph obtained from G after omitting the edge $\{v_j, v_{j+1}\}$ is disconnected, a contradiction (since G is a cycle). Thus X_i must contains at least one more vertex u which implies that the weighted treewidth of G is at least $w(u) + w(v_j) + w(v_{j+1}) - 1 \geq w(v_0) + w(v_j) + w(v_{j+1}) - 1$. $\square$

In the following lemmas, we assume that G is a cycle with vertices $v_0, v_1, \ldots, v_{n-1}$ and with edges $\{v_i, v_{i+1}\}$, $0 \leq i \leq n-1$, where we identify v_n with v_0, and v_{-1} with v_{n-1}. We also assume we have weights $w(v_i)$ and $t(v_i)$ for the vertices of G. For shorter notation, we write $w(i)$ for $w(v_i)$ and $t(i)$ for $t(v_i)$. We assume that $n \geq 5$. (This implies that for every independent set $I \subseteq V(G)$, $G : I$ is again a cycle: if $n \leq 4$, then this is not necessary). For $n \leq 4$, we can find $wi(G)$ in constant time.

We will now discuss how to compute $wi(G)$. Suppose, without loss of generality, that $w(0) = \min_{0\leq i\leq n-1} w(i)$. The following lemma states that we may assume that v_0 is not included in an independent set I establishing the wi-treewidth of G.

Lemma 36 *There is an independent set $I \subseteq V(G) - \{v_0\}$, such that $wi(G) = wi(G, I)$.*

Proof. By definition, there is an independent set $I \subseteq V(G)$, such that $wi(G) = wi(G, I)$. If $v_0 \notin I$, we are done. Suppose $v_0 \in I$. We will prove the lemma by showing that $wi(G, I) \geq wi(G, I - \{v_0\})$. It is sufficient to show that $wtw(G : I) \geq wtw(G : (I - \{v_0\}))$, as clearly $\max_{v_i\in I}\{t(i) + w(i-1) + w(i+1)\} \geq \max_{v_i\in I-\{v_0\}}\{t(i) + w(i-1) + w(i+1)\}$.

Write $r = \min_{0\leq i\leq n-1, v_i\notin I} w(i)$. By Lemma 35,

$$wtw(G : I) = r + \max_{\{x,y\}\in E(G:I)}\{w(x) + w(y)\} - 1, \textit{ and}$$

$$wtw(G : (I - \{v_0\})) = w(0) + \max_{\{x,y\}\in E(G:(I-\{v_0\}))}\{w(x) + w(y)\} - 1.$$

By assumption, $r \geq w(0)$. $G : I$ and $G : (I - \{v_0\})$ are cycles that only differ in a few edges: $G : I$ has an edge $\{v_{n-1}, v_1\}$ and $G : (I - \{v_0\})$ has edges $\{v_{n-1}, v_0\}$ and $\{v_0, v_1\}$. Now, as $w(n-1)+w(0) \leq w(n-1)+w(1)$, and $w(0)+w(1) \leq w(n-1)+w(1)$, we have that $wtw(G:I) \geq wtw(G:(I-\{v_0\}))$, and the claim then follows. $\square$

Now, assume $v_0 \notin I$. We now can write $wi(G,I)$ in more detail as:

$$wi(G,I) = \max \left\{ \begin{array}{ll} w(0)+w(j-1)+w(j+1)-1 & \mid 1 \leq j \leq n-1, v_j \in I \\ w(0)+w(j)+w(j+1)-1 & \mid 0 \leq j \leq n-1, v_j, v_{j+1} \notin I \\ t(j)+w(j-1)+w(j+1) & \mid 1 \leq j \leq n-1, v_j \in I \end{array} \right\}$$

To find the independent set $I \subseteq V(G)-\{v_0\}$ for which this term is minimal, we use dynamic programming.

For $i \geq 1$ and a set $I \subseteq \{v_1, \ldots, v_{i-1}\}$, define

$$h(I,i) = \max \left\{ \begin{array}{ll} w(0)+w(j-1)+w(j+1)-1 & \mid 1 \leq j \leq i-1, v_j \in I \\ w(0)+w(j)+w(j+1)-1 & \mid 0 \leq j \leq i-1, v_j, v_{j+1} \notin I \\ t(j)+w(j-1)+w(j+1) & \mid 1 \leq j \leq i-1, v_j \in I \end{array} \right\}$$

For $i=0$, define $h(\emptyset, 0) = w(0)-1$.

Write $h(i) = \min_{I \subseteq \{v_1,\ldots,v_{i-1}\}, I \in indp(G)} h(I,i)$. Clearly, $h(n) = wi(G)$.

Lemma 37 *(i)* $h(0) = w(0) - 1$.

(ii) $h(1) = 2w(0) + w(1) - 1$.

(iii) For $2 \leq i \leq n-1$,

$$\begin{aligned} h(i) = \min\{ & \max\{h(i-2), w(0) + w(i-2) + w(i) - 1, \\ & \quad t(i-1) + w(i-2) + w(i)\}, \\ & \max\{h(i-1), w(0) + w(i-1) + w(i) - 1)\}\}. \end{aligned}$$

(iv)

$$\begin{aligned} h(n) = \min\{ & \max\{h(n-1), 2w(0) + w(n-1) - 1\}, \\ & \max\{h(n-2), 2w(0) + w(n-2) - 1, \\ & \quad t(n-1) + w(n-2) + w(0)\}\}. \end{aligned}$$

Proof. (i), (ii). The only candidate for set I is the empty set.

(iii). Let $2 \leq i \leq n-1$. Let A denote the minimum value of $h(I, i)$ over all independent sets $I \subseteq \{v_1, \ldots, v_{i-1}\}$ with $v_{i-1} \in I$. We claim that:

$$A = \max\{h(i-2), w(0) + w(i-2) + w(i) - 1, t(i-1) + w(i-2) + w(i)\}.$$

We show first A is at least the above term. For any independent set $I \subseteq \{v_1, \ldots, v_{i-1}\}$ with $v_{i-1} \in I$ note that $v_{i-2} \notin I$, and we have that $h(I, i) \geq h(I - \{v_{i-1}\}, i-2) \geq h(i-2)$. Also, we have $h(I, i) \geq w(0) + w(i-2) + w(i) - 1$, and $h(I, i) \geq t(i-1) + w(i-2) + w(i)$, as these terms appear in the definition of $h(I, i)$. Hence, A is at least the above term. To show that A is exactly the above term we observe that for any independent set $I' \subseteq \{v_1, \ldots, v_{i-3}\}$ with $h(I', i-2) = h(i-2)$, the set $I' \cup \{v_{i-1}\}$ is an independent set and $h(I' \cup \{v_{i-1}\}, i) = \max\{h(i-2), w(0) + w(i-2) + w(i) - 1, t(i-1) + w(i-2) + w(i)\}$.

A similar analysis shows that the minimum value of $h(I, i)$ over all independent sets $I \subseteq \{v_1, \ldots, v_{i-1}\}$ with $v_{i-1} \notin I$ is exactly $\max\{h(i-1), w(0) + w(i-1) + w(i) - 1\}$. Part (iii) of the claim now follows.

(iv) With arguments, similar to part (iii) of this proof, one observes that the minimum value of $h(I, n)$ over all $I \subseteq \{v_1, \ldots, v_{n-1}\}$ with $v_{n-1} \in I$ is exactly $\max\{h(n-2), 2w(0) + w(n-2) - 1, t(n-1) + w(n-2) + w(0)\}$, and that the minimum value of $h(I, n)$ over all $I \subseteq \{v_1, \ldots, v_{n-1}\}$ with $v_{n-1} \notin I$ is exactly $\max\{h(n-1), 2w(0) + w(n-1) - 1\}$. $\square$

The above lemma gives a dynamic programming algorithm to compute all values $h(i)$, and hence to compute the wi-treewidth of G (which equals $h(n)$) in $O(n)$ time. Hence by Theorem 9 we have:

Theorem 38 *Let $\mathcal{C}$ be a class of graphs which is modularly decomposable into the class of cycles. Then the WI-TREEWIDTH problem on $\mathcal{C}$ can be solved in $O(n + m)$ time.*

5 Clique-matching graphs

In the previous sections, we have seen that WI-treewidth can be solved in polynomial time for graphs with a polynomial number of minimal separators. There are however also graph classes with an exponential number of minimal separators for which we also can solve the WI-treewidth problem, and hence that can be used as prime graphs in a modular decomposition for graphs we want to compute the treewidth of. In this section, we give a simple set of graphs of this type. We expect that it is possible to obtain similar results for much more general classes of graphs, but leave that here for further research.

Say a graph G is a *clique-matching graph*, if it is isomorphic to a graph $C_r = (V_r \cup W_r, E_r)$, with $V_r = \{v_1, \ldots, v_r\}$, $W_r = \{w_1, \ldots, w_r\}$, and $E_r = \{\{v_i, v_j\} \mid 1 \le i, j \le r, i \ne j\} \cup \{\{w_i, w_j\} \mid 1 \le i, j \le r, i \ne j\} \cup \{\{v_i, w_i\} \mid 1 \le i \le r\}$, i.e., C_r is formed by taking two disjoint cliques of r vertices each, and then connecting the ith vertex in the first clique with the ith vertex of the second clique for each i, $1 \le i \le r$. An example is given in Figure 1.

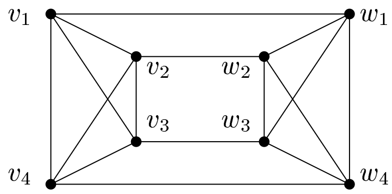

Figure 1: The clique-matching graph C_4.

First, we will show that we can compute the weighted treewidth of a clique-matching graph C_r given with a weight function w on its vertices. For a set of vertices S, we write $w(S) = \sum_{v \in S} w(v)$.

Lemma 39 *The weighted treewidth of C_r equals the minimum weighted width of a path decomposition $(X_1, \ldots, X_s)$ of C_r with $X_1 = V_r$ and $X_s = W_r$.*

Proof. As a path decomposition $(X_1, \ldots, X_s)$ of C_r with $X_1 = V_r$ and $X_s = W_r$ is also a tree decomposition of C_r, we have that the weighted treewidth is never larger than the weighted width of such a path decomposition.

Suppose we have a tree decomposition $(\{X_i \mid i \in I\}, T = (I, F))$ of C_r of width R. There must be a node $i_V \in I$ with $V_r \subseteq X_{i_V}$, and a node $i_W \in I$ with $W_r \subseteq X_{i_W}$. Let P be the path in T with endpoints i_V and i_W. For every edge $\{v_j, w_j\} \in E_r$, there must be a node i on P with $v_j, w_j \in X_i$. Thus, when we remove all nodes from I that do not belong to P, we still have a tree decomposition of C_r, and, as P is a path, this actually is a path decomposition. Add one additional node with set V_r and make it adjacent to i_V, and an additional node with set W_r and make it adjacent to i_W and we have the required path decomposition of C_r; clearly the weighted width of this path decomposition cannot be more than R. □

Lemma 40 *Suppose w_j is the vertex of minimum weight from all the vertices in $W \cup V$. Then*

$$wtw(C_r) = \max(wtw(C_r - \{v_j, w_j\}), w(V)) + w(w_j) \tag{6}$$

Proof. First, note that W is a full component of V. Thus, by Lemma 22, in every triangulation of C, V cannot be a maximal clique hence there is a clique that contains V

in such a triangulation; as the weight of this clique is at least $w(V) + w(w_j)$, we have that $w(V) + w(w_j) \leq wtw(C_r)$.

Consider a path decomposition $(X_1, \ldots, X_s)$ of C_r as in Lemma 39 of minimum weighted width, say R. There must be an α, $1 \leq \alpha \leq s$ with $v_j, w_j \in X_\alpha$. Now, if $1 \leq \beta \leq \alpha$, $v_j \in X_\beta$, and if $\alpha \leq \beta \leq s$, $w_j \in X_\beta$; hence $(X_1 - \{v_j, w_j\}, X_2 - \{v_j, w_j\}, \ldots, X_s - \{v_j, w_j\})$ is a path decomposition of $C_r - \{v_j, w_j\}$ of weighted width at most $R - min(w(v_j), w(w_j)) = R - w(w_j)$. So, we have $wtw(C_r - \{v_j, w_j\}) + w(w_j) \leq wtw(C_r)$.

If we have a path decomposition $(X_1, \ldots, X_s)$ of $C_r - \{v_j, w_j\}$ of minimum width R' with $X_1 = V - \{v_j\}$, $X_s = W - \{w_j\}$, then $(V, V \cup \{w_j\}, X_1 \cup \{w_j\}, X_2 \cup \{w_j\}, \ldots, X_s \cup \{w_j\})$ is a path decomposition of C_r, again of the form as stated in Lemma 39 of weighted width at most $\max(R' + w(w_j), w(V) + w(w_j))$. So, $wtw(C_r) \leq \max(wtw(C_r - \{v_j, w_j\}) + w(w_j), w(V) + w(w_j))$. The lemma now follows. □

Of course, when v_j is the minimum weight vertex from $W \cup V$, then

$$wtw(C_r) = \max(wtw(C_r - v_j, w_j), w(W)) + w(v_j) \tag{7}$$

Note also that $C_r - \{v_j, w_j\}$ is again a clique-matching graph. Thus, we can determine the weighted treewidth of C_r by repeatedly finding the vertex of minimum weight, removing this minimum weight vertex and its neighbor in the other clique, recursively computing the weighted treewidth of the resulting graph and then applying Equation 6 or 7. Thus, we have

Lemma 41 *The weighted treewidth problem can be solved in $O(n \log n)$ time on clique-matching graphs.*

Note that every independent set in a clique-matching graph either contains no vertices, a single vertex, or one vertex in each of the two cliques, so there are $O(n^2)$ independent sets in a clique-matching graph. Suppose we want to compute $wtw(C_r : \{v_i\})$. Note that w_i is simplicial in $C_r : \{v_i\}$. The following lemma is well known for the unweighted case; its extension to the weighted case is trivial.

Lemma 42 *Let v be a simplicial vertex in a graph G. Then $wtw(G) = \max(w(N(v) \cup \{v\}), wtw(G - \{v\}))$.*

As $C_r : \{v_i\} - \{w_i\}$ is again a clique-matching graph, this directly gives an $O(n \log n)$ time algorithm to compute the weighted treewidth of a graph of the form $C_r : \{v_i\}$ or $C_r : \{w_i\}$. Similarly, for $i' \neq i''$, $v_{i'}$ and $w_{i''}$ are simplicial in $C_r : \{v_{i'}, w_{i''}\}$, so we can also compute the weighted treewidth of these graphs in $O(n \log n)$ time. Actually, after we have an ordering of the vertices of C_r, we do not need to sort again the vertices of each of the graphs $C_r : X$, and thus for each of these, we can then compute its weighted treewidth in linear time. Hence, we have:

Theorem 43 *The WI-TREEWIDTH problem can be solved in $O(n^3)$ time on clique-matching graphs.*

With simple additional bookkeeping, the corresponding tree decompositions can be built. A more detailed analysis of the choice of vertices in the independent set can possibly bring the time for computed the WI-TREEWIDTH for clique-matching graphs down.

6 Minimum fill-in

In this section we show that the results presented at the previous sections hold also for the MINIMUM FILL-IN problem. We start by introducing more definitions and notations. The *minimum fill-in* of a graph G, denoted by *min-fill-in*(G), is the minimum of $|E(H) - E(G)|$ taken over all triangulations H of G. For a triangulation H of G such that $|E(H) - E(G)| =$ *min-fill-in*(G) we say that H *establishes* the minimum fill-in of G. The *complete fill-in* of G, denoted by *complete-fill-in*(G) is the number of edges that must be added to turn G into a clique:

$$\textit{complete-fill-in}(G) = \frac{1}{2}|V(G)| \cdot (|V(G)| - 1) - |E(G)|.$$

The MINIMUM FILL-IN problem is to find *min-fill-in*(G) for a given graph G.

Let G be a graph with weight functions w, f and c. The *weighted minimum fill-in* of G, denoted by $wf(G)$ is defined as the minimum of $\sum_{\{v,x\} \in E(H)-E(G)} w(v) \cdot w(x)$ taken over all triangulations H of G. For a triangulation H of G such that $wf(G) = \sum_{\{v,x\} \in E(H)-E(G)} w(v) \cdot w(x)$ we say that H *establishes* the weighted fill-in of G. The *weighted complete fill-in* of G, denoted by $wcf(G)$ is defined by $wcf(G) = \sum_{\{u,v\} \notin E(G)} w(u) \cdot w(v)$. Notice that $wf(G)$ and $wcf(G)$ depends just on w and not on f and c.

Recall that for a set of vertices S we define $f(S) = \sum_{v \in S} f(v)$ and $c(S) = \sum_{v \in S} c(v)$. For a set $X \in indp(G)$, the *weighted minimum fill-in of G with independent set X* denoted as $wif(G, X)$ is defined as follows:

$$wif(G, X) = f(X) + c(V(G) - X) + wf(G : X) + \sum_{\{u,v\} \in E(G:X)-E(G)} w(u) \cdot w(v).$$

The *weighted independent minimum fill-in* of G (shortly the *wi-fill-in* of G) denoted as $wif(G)$ is defined by:

$$wif(G) = \min_{X \in indp(G)} wif(G, X).$$

For a set $X \in indp(G)$ such that $wif(G) = wif(G, X)$ we say that X *establishes* the wi-fill-in of G. The WI-FILL-IN problem is to find $wif(G)$ for a given graph G with weight functions w, f and c.

Let S be a set of vertices in G. We denote by $wif_S(G)$ the minimum wi-fill-in of G assuming that none of the vertices of S are in the independent set. Formally,

$$wif_S(G) = \min\{wif(G, X) \mid X \in indp(G) \textit{ and } S \cap X = \emptyset\}$$

We omit the proofs of Lemma 44 and Theorem 45 since they are similar to the proofs of Lemma 7 and Theorem 9, respectively.

Lemma 44 *Let G be a graph, where $V(G) = \{v_1, \ldots, v_r\}$. Let $H_1, \ldots, H_r$ be disjoint graphs and let $G' = G(H_1, \ldots, H_r)$. Let w and f and c be the weight functions on the vertices of G defined by: $w(v_i) = |V(H_i)|$, $f(v_i) =$ min-fill-in(H_i) and $c(v_i) =$ complete-fill-in(H_i), for $1 \le i \le r$. Then min-fill-in$(G') = wif(G)$.*

Theorem 45 *Let $\mathcal{C}$ and $\mathcal{D}$ be classes of graphs such that $\mathcal{C}$ is modularly decomposable into $\mathcal{D}$. Suppose that the WI-FILL-IN problem can be solved in $O(f(n, m))$ time on $\mathcal{D}$, where f is some polynomial function in n and m. Then the MINIMUM FILL-IN problem on $\mathcal{C}$ can be solved in $O(f(n, m) + n + m)$ time.*

We now consider each of the graph classes mentioned in the previous sections.

6.1 Classes with a polynomial number of minimal separators

In Section 3 we presented an algorithm for solving the WI-TREEWIDTH problem on any class of graphs $\mathcal{C}$ with a polynomial number of minimal separators. The algorithm is based on Formulas 1-5. Using the same method we can solve the WI-FILL-IN problem on any class of graphs $\mathcal{C}$ with a polynomial number of minimal separators based on Formulas 8-12 below which corresponds to Formulas 1-5 respectively.

We omit the proofs of Lemmas 46,47,48,49,50,51,52,53,54 since they are similar to the proofs of Lemmas 11,12,13,14,20,21,25,29,31 respectively.

Lemma 46 *Let K be a clique in G and let $C_1, \ldots, C_t$ be the connected components of $G-K$. Then*

$$wif_K(G) = c(K) + \sum_{i=1}^{i=t}(wif_K(G_K(C_i)) - c(K)).$$

Lemma 47 *Let K be a clique in G, let C be a connected component of $G-K$ and let K^C be the set of all vertices in K having a neighbor in C. Then*

$$wif_K(G_K(C)) = c(K) + wif_{K^C}(G_{K^C}(C)) - c(K^C)$$

Lemma 48 *Let K be a clique in G, and let $C_1, \ldots, C_t$ be the connected components of $G-K$. For $1 \leq i \leq t$ let K_i be the set of all vertices in K having a neighbor in C_i. Then*

$$wif_K(G) = c(K) + \sum_{i=1}^{i=t}(wif_{K_i}(G_{K_i}(C_i)) - c(K_i)).$$

Lemma 49 *Let X be an independent set in G establishing $wif(G)$. Let S be a clique in a minimal triangulation Q of $G:X$ that establishes wf(G:X). Let $C_1, \ldots, C_t$ be the connected components of $G-S$. Then*

$$wif(G) = c(S) + wcf(G[S]) + \sum_{i=1}^{i=t}(wif_S(G_S(C_i) - c(S)))$$

Lemma 50 *For every graph G with weight functions w, f and c,*

$$\begin{aligned} wif(G) = \min\{&\min\{f(v) + c(N(v)) + wcf(G[N(v)]) \mid v \in univ(G)\}, \\ &\min_{S \in msep(G)}\{c(S) + wcf(G[S]) + \sum_{C \in cc(G-S)}(wif_S(G_S(C)) - c(S))\}\}. \end{aligned} \tag{8}$$

Lemma 51 *Let $S \in msep(G)$ and let $C \in cc(G-S)$. Then $S^C \in msep(G)$, S^C is close to C and*

$$wif_S(G_S(C)) = c(S) + wif_{(S^C)}(G_{S^C}(C)) - c(S^C) \tag{9}$$

Lemma 52 *Let S be a minimal separator of G and let C be a full component of S, such that $G_S(C)$ is not a clique, then*

$$
\begin{aligned}
wif_S(G_S(C)) = \min\{wif_\Omega(G_\Omega[S \cup C]) \mid\ & \textit{either } S \subset \Omega \subseteq S \cup C \\
& \textit{and } \Omega \in potm(G) \textit{ or there exist } x \in C \cap univ(S) \\
& \textit{such that } \Omega = N(x)\}.
\end{aligned} \tag{10}
$$

Lemma 53 *Let $S \in msep(G)$ and let C be a connected component of $G - S$. Let Ω be a potential maximal clique of G such that $S \subset \Omega \subseteq S \cup C$. Let $C_1, \dots, C_t$ be the connected components of $G_S(C) - \Omega$, and for $1 \le i \le t$ let Ω_i denote the set of all vertices in Ω having a neighbor (in the graph G) in C_i. Then for $1 \le i \le t$, $\Omega_i \subset \Omega$, $\Omega_i \in msep(G)$ and*

$$
wif_\Omega(G_\Omega[S \cup C]) = c(\Omega) + \sum_{i=1}^{i=t} (wif_{\Omega_i}(G_{\Omega_i}(C_i)) - c(\Omega_i)). \tag{11}
$$

Lemma 54 *Let $S \in msep(G)$ and let C be a connected component of $G - S$. Let x be a vertex in $C \cap univ(S)$ and let $\Omega = N(x)$. Let $C_1, \dots, C_t$ be the connected components of $G_S(C) - \Omega$, and for $1 \le i \le t$ let Ω_i denote the set of all vertices in Ω having a neighbor (in the graph G) in C_i. Then Formula 12 holds and for $1 \le i \le t$, either $G_{\Omega_i}(C_i)$ is a clique or $\Omega_i \in msep(G)$ and $\Omega_i \cup C_i \subset S \cup C$.*

$$
wif_\Omega(G_\Omega[S \cup C]) = c(\Omega) + \sum_{i=1}^{i=t} (wif_{\Omega_i}(G_{\Omega_i}(C_i)) - c(\Omega_i)) \tag{12}
$$

6.2 Cycles

In this section we show that for a class of graphs $\mathcal{C}$ which is modularly decomposable into the class of cycles the MIN-FILL-IN problem on $\mathcal{C}$ can be solved in $O(n^3)$ time. We obtain this result by showing that the WI-FILL-IN problem on cycles can be solved in $O(n^3)$ time. In other words we show how to compute $wif(G)$ for a given cycle G with weight functions w, f and c. We assume that G is of length at least 4 (otherwise we can calculate $wif(G)$ in constant time).

We denote by $\{v_1, \dots, v_n\}$ the successive vertices on G. For $1 \le i \le j-2 \le n$ we denote by $G(i,j)$ the cycle obtained by adding the edge $\{v_i, v_j\}$ to the subgraph of G induced by $\{v_i, \dots, v_j\}$, i.e., $G(i,j) = G[\{v_i, \dots, v_j\}] + \{\{v_i, v_j\}\}$. Similarly, we denote by $G(j,i)$ the cycle obtained by adding the edge $\{v_i, v_j\}$ to the subgraph of G induced by $\{v_j, \dots, v_n, v_1, \dots, v_i\}$. We define $G(i, i+1) = G[\{v_i, v_{i+1}\}]$ for $1 \le i \le n-1$, and $G(j, j-1) = G[\{v_{j-1}, v_j\}]$ for $2 \le j \le n$.

Lemma 55

$$
\begin{aligned}
wif(G) = \min_{1 \le i \le j-2 \le n} & wif(1(\{v_i, v_j\}), G(i,j)) + wif(1(\{v_i, v_j\}), G(j,i)) + \\
& w(v_i) \cdot w(v_j) - (c(v_i) + c(v_j))
\end{aligned} \tag{13}
$$

Proof.

($\leq$) Suppose that the minimum of the right hand side of the above formula is reached for the two integers i, j. Let X_1 and X_2 be the set of vertices establishing $wif(1(\{v_i, v_j\}), G(i,j))$ and $wif(1(\{v_i, v_j\}), G(j,i))$ and let Q_1 and Q_2 be triangulations establishing $wf(G(i,j) : X_1)$ and $wf(G(j,i) : X_2)$, respectively. Let $X = X_1 \cup X_2$ and let Q be the graph defined by: $V(Q) = V(G) - X$ and $E(Q) = E(Q_1) \cup E(Q_2)$. Now since H is a triangulation of $G : X$ we get that $wif(G, X)$ is at most the right hand side of Formula 13.

($\geq$) Let X be a set of vertices establishing $wif(G, X)$. Suppose $X = \emptyset$, i.e., $wif(G) = c(V(G)) + wf(G)$. Let Q be a triangulation of G establishing $wf(G)$. Let $\{v_i, v_j\}$ be any edge in $E(Q) - E(G)$. Let $Q_1 = Q[\{v_i, \ldots, v_j\}]$ and $Q_2 = Q[\{v_j, \ldots, v_n, v_1 \ldots, v_i\}]$. Now observing that Q_1 and Q_2 are triangulations of $G(i,j)$ and $G(j,i)$ respectively, we get that the right-hand side of Formula 13 for these i and j is at most $wif(G)$.

Suppose $X \neq \emptyset$, i.e., there exists $x \in X$. Let v_i and v_j be the two neighbors of x in G. Now using a similar argument we can obtain that the right-hand side of Formula 13 for these i and j is at most $wif(G)$. $\square$

Define

$$A(i,j) = \begin{cases} 1 & \text{if } |i-j| \geq 2, \\ 0 & \text{otherwise.} \end{cases}$$

We omit the proof of the following lemma which is similar to the proof of Lemma 55.

Lemma 56 *For* $1 \leq i \leq j-2 \leq n$,

$$\begin{aligned} &wif(1(\{v_i, v_j\}), G(i,j)) = \\ &\quad \min_{i<l<j} wif(1(\{v_i, v_l\}), G(i,l)) + wif(1(\{v_l, v_j\}), G(l,j)) + \\ &\quad\quad A(i,l) \cdot w(v_i) \cdot w(v_l) + A(l,j) \cdot w(v_l) \cdot w(v_j) - c(v_l). \end{aligned} \tag{14}$$

$$\begin{aligned} &wif(1(\{v_i, v_j\}), G(j,i)) = \\ &\quad \min_{l \in \{j+1, \ldots, n, 1, \ldots, i-1\}} wif(1(\{v_j, v_l\}), G(j,l)) + wif(1(\{v_l, v_i\}), G(l,i)) + \\ &\quad\quad A(j,l) \cdot w(v_j) \cdot w(v_l) + A(l,i) \cdot w(v_l) \cdot w(v_i) - c(v_l). \end{aligned} \tag{15}$$

We now order all pairs i, j such that $1 \leq i \leq j-2 \leq n$ by increasing order of $j-i$ and calculate in this order $wif(1(\{v_i, v_j\}), G(i,j))$ and $wif(1(\{v_i, v_j\}), G(j,i))$ using Formulas 14 and 15. Since each calculation takes $O(n)$ time and we have $O(n^2)$ pairs the total time for calculating these values is $O(n^3)$. Having all these values we can compute $wif(G)$ using Formula 13 in $O(n^2)$ time. We have shown that the WI-FILL-IN problem on the class of cycles can be solved in $O(n^3)$ time. Hence by Theorem 45 we get:

Theorem 57 *Let $\mathcal{C}$ be a class of graphs which is modularly decomposable into the class of cycles. Then the MINIMUM-FILL-IN problem on $\mathcal{C}$ can be solved in $O(n^3)$ time.*

6.3 Clique-matching graphs

In this section, we give a polynomial time algorithm for the WI-FILL-IN problem for clique-matching graphs. Suppose that $C_r = (V \cup W, E_r)$ is a weighted clique-matching graph with

$2r$ vertices. We first give an algorithm to compute the weighted fill-in of a clique-matching graph C_r.

We first show that there is an ordering of the vertices of W such that the triangulation is formed by making every vertex w_j adjacent to all vertices v_i with w_i later than w_j in the ordering.

Lemma 58 *Let Q be a minimal triangulation of C_r. There is an ordering $\preceq$ of the vertices of W such that $E(Q) = E_r \cup \{\{w_i, v_j\} \mid w_i \preceq w_j,\ 1 \leq i \leq r,\ 1 \leq j \leq r,\ i \neq j\}$.*

Proof. Use induction to r. For $r = 1$ the lemma follows immediately since $E(Q) = E(C)$. Suppose the lemma holds for $r - 1$. Let Q be a minimal triangulation of the graph C_r. As V has a full component, there must be a vertex w_i in W such that $V \cup \{w_i\}$ forms a clique in Q. If v_i is adjacent in Q to any vertex w_j, $j \neq i$, then triangulation Q is not minimal. It follows that the graph Q' , obtained by removing v_i and w_i and their adjacent edges from Q is a minimal triangulation of $C_r - \{v_i, w_i\}$. Let $\preceq'$ be the ordering on $W - \{w_i\}$ such that $E(Q') = E(C_r - \{v_i, w_i\}) \cup \{\{w_{i'}, v_j\} \mid w_{i'} \preceq' w_j,\ 1 \leq i' \leq r,\ 1 \leq j \leq r,\ i', j \neq i\}$. Now, let $\preceq$ be the ordering on W such that for all $w_j, w_{j'} \in W - \{w_i\}$, $w_j \preceq w_{j'}$ if and only if $w_j \preceq' w_{j'}$, and for all $w_j \in W$, $w_i \preceq w_j$, i.e., we take ordering $\preceq'$ and add w_i as smallest element. One now easily sees that $\preceq$ fulfils the condition of the lemma. □

We also have that given an ordering $\preceq$ of W, the edge set $E_r \cup \{\{w_i, v_j\} \mid w_i \preceq w_j,\ 1 \leq i \leq r,\ 1 \leq j \leq r,\ i \neq j\}$ gives a triangulation of C_r. For an ordering $\preceq$ of W, let the *fill-in* of the ordering be

$$FI(\preceq) = \sum_{w_i \preceq w_j,\ 1 \leq i \leq r,\ 1 \leq j \leq r,\ i \neq j} w(w_i) \cdot w(v_j)$$

$FI(\preceq)$ exactly denotes the total weight of all edges added in the triangulation corresponding to ordering $\preceq$, and thus the problem to compute the weighted fill-in of C_r becomes the problem to find an ordering $\preceq$ of W with minimum fill-in $FI(\preceq)$.

Lemma 59 *An ordering $\preceq$ has minimum fill-in among all orderings of W, if and only if for all w_i, $w_j \in W$*

$$w_i \preceq w_j \Rightarrow \frac{w(w_i)}{w(v_i)} \leq \frac{w(w_j)}{w(v_j)}$$

Proof. Consider an ordering $\preceq$ of W. Suppose w_i and w_j are successive elements in this ordering with $w_i \preceq w_j$, i.e., there is no $w_{i'} \notin \{w_i, w_j\}$ with $w_i \preceq w_{i'} \preceq w_j$. Let $\preceq'$ be the ordering obtained from $\preceq$ by switching the order of w_i and w_j (and keeping the relative order for all other pairs). Considering all the terms that appear in $FI(\preceq)$ and $FI(\preceq')$, we see that $FI(\preceq') - FI(\preceq) = w(w_j) \cdot w(v_i) - w(w_i) \cdot w(v_j)$.

If $\frac{w(w_i)}{w(v_i)} > \frac{w(w_j)}{w(v_j)}$ then $w(w_j) \cdot w(v_i) - w(w_i) \cdot w(v_j) < 0$, hence $FI(\preceq') < FI(\preceq)$. Thus, if $\preceq$ has minimum fill-in among all orderings of W, it must order the vertices of W with respect to non-decreasing values of $\frac{w(w_i)}{w(v_i)}$.

If $\frac{w(w_i)}{w(v_i)} = \frac{w(w_j)}{w(v_j)}$, then $w(w_j) \cdot w(v_i) - w(w_i) \cdot w(v_j) = 0$, so $FI(\preceq') = FI(\preceq)$. As all orderings of W that give the vertices in order of non-decreasing values $\frac{w(w_i)}{w(v_i)}$ can be obtained from each other by a number of switches of successive elements of equal such values, we have that all such orderings have the same fill-in, so all of these have minimum fill-in. □

Lemma 59 directly gives an $O(r^2)$ algorithm to solve the WEIGHTED FILL-IN problem on clique-matching graphs: sort the vertices in W with respect to the values $\frac{w(w_i)}{w(v_i)}$, and then build the corresponding triangulation as described above. As the triangulated graph has $\Theta(r^2)$ edges, this latter step dominates the running time. Only computing the value of the weighted fill-in can be done a little faster: one can compute in total linear time all terms $\sum_{w_i \preceq w_j} w(v_j)$ for all $w_i \in W$, and then directly compute $FI(\preceq) = \sum_{w_i \in W} w(w_i) \cdot \sum_{w_i \preceq w_j} w(v_j)$.

Lemma 60 *The WEIGHTED FILL-IN problem on clique-matching graphs can be solved in $O(n \log n)$ time. A triangulation of a clique-matching graph with minimum weighted fill-in can be found in $O(n^2)$ time.*

As in Section 5, we use that a clique-matching graph has independent sets of size at most two, and that for every $w_i \in X$, v_i is simplicial in $C_r : X$, and for every $v_i \in X$, w_i is simplicial in $C_r : X$. Moreover, we can use the following simple lemma.

Lemma 61 *Let v be a simplicial vertex in G. Then the weighted fill-in of G equals the weighted will-in of $G - v$.*

Thus, we can try all $O(n^2)$ independent sets X of G, and as the graphs obtained after removing simplicial vertices from $G : X$ are again clique-matching graphs, use in each case the algorithm to compute the weighted fill-in of each of these graphs. Note that we can reuse the orderings of the vertices; i.e., we need to sort the vertices only once for their values $\frac{w(w_i)}{w(v_i)}$. Thus, we have:

Lemma 62 *The WI-FILL-IN problem can be solved in $O(n^3)$ time for clique-matching graphs.*

7 Conclusions

Consider the following operation, that given a graph G and for every $v_i \in V(G)$ a graph H_{v_i}, gives the graph $G(H_{v_1}, \ldots, H_{v_n})$. Theorems 9 and 45 show that the treewidth and minimum fill-in of the composed graph are a function of the numbers of vertices and treewidths (respectively, minimum fill-ins) of the graphs H_{v_i}. These functions are expressed by respectively the WI-TREEWIDTH and WI-FILL-IN problems. In this paper, we have shown for a number of classes of graphs that for graphs in these classes these notions are computable, i.e., graphs in these classes can play the role of the graph G in the substitution operation when we want to compute the treewidth or minimum fill-in. In particular, we looked at graphs with a polynomial number of separators (a fairly large class of graphs) and at clique-matching graphs (a rather restricted class of graphs, introduced just to show that there are also solvable cases with an exponential number of separators.) A natural question is how to solve the WI-TREEWIDTH and WI-FILL-IN problems on other interesting classes of graphs in polynomial time, for instance the class of graphs of treewidth at most some fixed number k.

Acknowledgements

We thank the referees for their careful reading and helpful comments on an earlier version of this paper.

References

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

[2] S. Arnborg, D. G. Corneil, and A. Proskurowski. Complexity of finding embeddings in a k-tree. *SIAM J. Alg. Disc. Meth.*, 8:277–284, 1987.

[3] L. Babel. Triangulating graphs with few P_4's. *Disc. Appl. Math.*, 89:45–57, 1998.

[4] A. Berry, J.-P. Bordat, and O. Cogis. Generating all minimal separators of a graph. In *Proceedings 25nd International Workshop on Graph-Theoretic Concepts in Computer Science WG'99*, pages 167–172, 1999.

[5] H. L. Bodlaender, J. R. Gilbert, H. Hafsteinsson, and T. Kloks. Approximating treewidth, pathwidth, frontsize, and minimum elimination tree height. *J. Algorithms*, 18:238–255, 1995.

[6] H. L. Bodlaender, T. Kloks, and D. Kratsch. Treewidth and pathwidth of permutation graphs. *SIAM J. Disc. Math.*, 8(4):606–616, 1995.

[7] H. L. Bodlaender and R. H. Möhring. The pathwidth and treewidth of cographs. *SIAM J. Disc. Math.*, 6:181–188, 1993.

[8] V. Bouchitté and I. Todinca. Minimal triangulations for graphs with "few" minimal separators. In *Proceedings ESA'98*, pages 344–355. Springer Verlag, Lecture Notes in Computer Science, vol. 1461, 1998.

[9] V. Bouchitté and I. Todinca. Treewidth and minimum fill-in of weakly triangulated graphs. In *Proceedings STACS'99*, pages 197–208. Springer Verlag, Lecture Notes in Computer Science, vol. 1563, 1999.

[10] V. Bouchitté and I. Todinca. Listing all potential maximal cliques of a graph. In H. Reidel and S. Tison, editors, *Proceedings STACS'00*, pages 503–515. Springer Verlag, Lecture Notes in Computer Science, vol. 1770, 2000.

[11] H. Broersma, E. Dahlhaus, and T. Kloks. Algorithms for the treewidth and minimum fill-in of HHD-free graphs. In *Proceedings 23nd International Workshop on Graph-Theoretic Concepts in Computer Science WG'97*, pages 109–117. Springer Verlag, Lecture Notes in Computer Science, vol. 1335, 1997.

[12] H. Broersma, E. Dahlhaus, and T. Kloks. A linear time algorithm for minimum fill in and treewidth for distance hereditary graphs. *Disc. Appl. Math.*, 99:367–400, 2000.

[13] H. Buer and R. H. Möhring. A fast algorithm for the decomposition of graphs and posets. *Mathematics of Operations Research*, 8(2):170–184, 1983.

[14] A. Cournier and M. Habib. A new linear algorithm for modular decomposition. In T. Sophie, editor, *Trees in algebra and programming, CAAP'94*, pages 68–84. Springer Verlag, Lecture Notes in Computer Science, vol. 787, 1994.

[15] E. Dahlhaus. Minimum fill-in and treewidth for graphs modularly decomposable into chordal graphs. In *Proceedings 24nd International Workshop on Graph-Theoretic Concepts in Computer Science WG'98*, pages 351–358. Springer Verlag, Lecture Notes in Computer Science, vol. 1517, 1998.

[16] E. Dahlhaus, J. Gustedt, and R. M. McConnell. Efficient and practical modular decomposition. In *Proceedings of the 8th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 26–35, 1997.

[17] G. Dirac. On rigid circuit graphs. *Abh. Math. Sem. Univ. Hamburg*, 25:71–76, 1961.

[18] M. C. Golumbic. *Algorithmic Graph Theory and Perfect Graphs.* Academic Press, New York, 1980.

[19] T. Kloks. Treewidth of circle graphs. *Int. J. Found. Computer Science*, 7:111–120, 1996.

[20] T. Kloks and D. Kratsch. Listing all minimal separators of a graph. *SIAM J. Comput.*, 27(3):605–613, 1998.

[21] T. Kloks, D. Kratsch, and H. Müller. Approximating the bandwidth for asteroidal triple-free graphs. *J. Algorithms*, 32:41–57, 1999.

[22] T. Kloks, D. Kratsch, and J. Spinrad. On treewidth and minimum fill-in of asteroidal triple-free graphs. *Theor. Comp. Sc.*, 175:309–335, 1997.

[23] R. M. McConnell and J. Spinrad. Modular decomposition and transitive orientation. *Disc. Math.*, 201:189–241, 1999.

[24] R. 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.

[25] N. Robertson and P. D. Seymour. Graph minors. II. Algorithmic aspects of tree-width. *J. Algorithms*, 7:309–322, 1986.

[26] R. Sundaram, K. Sher Singh, and C. Pandu Rangan. Treewidth of circular-arc graphs. *SIAM J. Disc. Math.*, 7:647–655, 1994.

[27] M. Yannakakis. Computing the minimum fill-in is NP-complete. *SIAM J. Alg. Disc. Meth.*, 2:77–79, 1981.