

**REPORTS  
IN  
INFORMATICS**

**ISSN 0333-3590**

**How to use planarity efficiently: new  
tree-decomposition based algorithms**

**Frederic Dorn**

**REPORT NO 349**

**March 2007**



*Department of Informatics*  
**UNIVERSITY OF BERGEN**  
*Bergen, Norway*

This report has URL <http://www.ii.uib.no/publikasjoner/textrap/pdf/2007-349.pdf>

Reports in Informatics from Department of Informatics, University of Bergen, Norway, is available at  
<http://www.ii.uib.no/publikasjoner/textrap/>.

Requests for paper copies of this report can be sent to:  
Department of Informatics, University of Bergen, Høyteknologisenteret,  
P.O. Box 7800, N-5020 Bergen, Norway

# How to use planarity efficiently: new tree-decomposition based algorithms

Frederic Dorn\*

Department of Informatics, University of Bergen, PO Box 7800, 5020 Bergen, Norway

## Abstract

We prove new structural properties for tree-decompositions of planar graphs that we use to improve upon the runtime of tree-decomposition based dynamic programming approaches for several NP-hard planar graph problems. We give for example the fastest algorithm for PLANAR DOMINATING SET of runtime  $3^{tw} \cdot n^{O(1)}$ , when we take the treewidth  $tw$  as the measure for the exponential worst case behavior. We also introduce a tree-decomposition based approach to solve non-local problems efficiently, such as PLANAR HAMILTONIAN CYCLE in runtime  $6^{tw} \cdot n^{O(1)}$ . From any input tree-decomposition, we compute in time  $O(nm)$  a tree-decomposition with geometric properties, which decomposes the plane into disks, and where the graph separators form Jordan curves in the plane.

## 1 Introduction

Many separator results for topological graphs, especially for planar embedded graphs base on the fact that separators have a structure that cuts the surface into two or more pieces onto which the separated subgraphs are embedded on. The celebrated and widely applied (a.o., in many divide-and-conquer approaches) result of Lipton and Tarjan [21] finds in planar graphs a small sized separator. However, their result says nothing about the structure of the separator, it can be any set of discrete points. Applying the idea of Miller for finding small simple cyclic separators [22] in planar triangulations, one can find small separators whose vertices can be connected by a closed curve in the plane intersecting the graph only in vertices, so-called *Jordan curves* (a.o. see [4]). Tree-decompositions have been historically the choice when solving NP-hard optimization and FPT problems with a dynamic programming approach (see for example [7] for an overview). Although much is known about the combinatorial structure of tree-decompositions (a.o. [6, 29]), no result is known to the author relating to the topology of tree-decompositions of planar graphs. A branch-decomposition is another tool, that was introduced by Robertson and Seymour in their proof of the Graph Minors Theorem and the parameters of these similar structures, the *treewidth*  $tw(G)$  and *branchwidth*  $bw(G)$  of the graph  $G$  have the relation  $bw(G) \leq tw(G) + 1 \leq 1.5 bw(G)$  [25]. Recently, branch-decompositions started to become a more popular tool than tree-decompositions, in particular for problems whose input is a topologically embedded graph [10, 18, 11, 15, 14], mainly for two reason: the branchwidth of planar graphs can be computed in polynomial time (yet there is no algorithm known for treewidth) with

---

\*Email: [frederic.dorn@ii.uib.no](mailto:frederic.dorn@ii.uib.no). Supported by the Research Council of Norway.

Table 1: Worst-case runtime expressed by treewidth  $tw$  and branchwidth  $bw$  of the input graph. The PLANAR HAMILTONIAN CYCLE stands representatively for all planar graph problems posted in [15] such as METRIC TSP, whose algorithms we can improve analogously. In [13], only those graph problems are improved upon, which are unweighted or of small integer weights. Therefore, we state the improvements independently for weighted and unweighted graph problems. In some calculations, the fast matrix multiplication constant  $\omega < 2.376$  is hidden.

	Previous results	New results
weighted PLANAR DOMINATING SET	$O(n2^{\min\{2\ tw, 2.38\ bw\}})$	$O(n2^{1.58\ tw})$
unweighted PLANAR DOMINATING SET	$O(n2^{1.89\ bw})$	$O(n2^{\min\{1.58\ tw, 1.89\ bw\}})$
weighted PLANAR INDEPENDENT DOMINATING SET	$O(n2^{\min\{2\ tw, 2.28\ bw\}})$	$O(n2^{1.58\ tw})$
unweighted PLANAR INDEPENDENT DOMINATING SET	$O(n2^{1.89\ bw})$	$O(n2^{\min\{1.58\ tw, 1.89\ bw\}})$
weighted PLANAR TOTAL DOMINATING SET	$O(n2^{\min\{2.58\ tw, 3\ bw\}})$	$O(n2^{2\ tw})$
unweighted PLANAR TOTAL DOMINATING SET	$O(n2^{2.38\ bw})$	$O(n2^{\min\{2\ tw, 2.38\ bw\}})$
weighted PLANAR PERFECT TOTAL DOMINATING SET	$O(n2^{\min\{2.58\ tw, 3.16\ bw\}})$	$O(n2^{\min\{2.32\ tw, 3.16\ bw\}})$
unweighted PLANAR PERFECT TOTAL DOMINATING SET	$O(n2^{2.53\ bw})$	$O(n2^{\min\{2.32\ tw, 2.53\ bw\}})$
weighted PLANAR HAMILTONIAN CYCLE	$O(n2^{3.31\ bw})$	$O(n2^{\min\{2.58\ tw, 3.31\ bw\}})$
unweighted PLANAR HAMILTONIAN CYCLE	$O(n2^{2.66\ bw})$	$O(n2^{\min\{2.58\ tw, 2.66\ bw\}})$

better constants for the upper bound than treewidth. Secondly, planar branch decompositions have geometrical properties, i.e. they are assigned with separators that form Jordan curves. Thus, one can exploit planarity in the dynamic programming approach in order to get an exponential speedup, as done by [15, 13]. We give the first result which employs planarity obtained by the structure of tree-decompositions for getting faster algorithms. This enables us to give the first tree-decomposition based algorithms for planar Hamiltonian-like problems with slight runtime improvements compared to [15]. We emphasize our result in terms of the width parameters  $tw$  and  $bw$  with the example of DOMINATING SET. The graph problem DOMINATING SET asks for a minimum vertex set  $S$  in a graph  $G = (V, E)$  such that every vertex in  $V$  is either in  $S$  or has a neighbor in  $S$ . Telle and Proskurowski [28] gave a dynamic programming approach based on tree-decompositions with runtime  $9^{tw} \cdot n^{O(1)}$ , and that was improved to  $4^{tw} \cdot n^{O(1)}$  by Alber et al [1]. Note that in the extended abstract [2], the same authors first stated the runtime wrongly to be  $3^{tw} \cdot n^{O(1)}$ . Fomin and Thilikos [18] gave a branch-decomposition based approach of runtime  $3^{1.5\ bw} \cdot n^{O(1)}$ . In an award winning work [13], Dorn combined dynamic programming with fast matrix multiplication to get  $4^{bw} \cdot n^{O(1)}$  and for PLANAR DOMINATING SET even  $3^{\frac{\omega}{2}\ bw} \cdot n^{O(1)}$ , where  $\omega$  is the constant in the exponent of fast matrix multiplication (currently,  $\omega \leq 2.376$ ). Exploiting planarity, we improve further upon the existing bounds and give a  $3^{tw} \cdot n^{O(1)}$  algorithm for PLANAR DOMINATING SET, representative for a number of improvements on results of [3, 15, 16] as shown in Table 1.

Given any tree-decomposition as an input, we show how to compute a geometric tree-decomposition that has the same properties as planar branch decompositions. Employing structural results on minimal graph separators for planar graphs, we create in polynomial time a *parallel* tree-decomposition that is assigned by a set of pairwise parallel separators that form pairwise non-crossing Jordan curves in the plane. In a second step, we show how to obtain a *geometric* tree-decomposition, that has a ternary tree and is assigned Jordan curves that exhaustively decompose the plane into disks (one disk being the infinite disk). In fact, geometric tree-decompositions have all the properties in common with planar branch decompositions, that are algorithmically exploited in [18] and [15].

**Organization of the paper:** after giving some preliminary results in Section 2, we introduce in Section 3

our algorithm to compute a parallel tree-decomposition. In Section 4, we describe how Jordan curves and separators in plane graphs influence each other and we get some tools for relating Jordan curves and tree-decompositions in Section 5. Finally, we show how to compute geometric tree-decompositions and state in Section 6 their influence on dynamic programming approaches. In Section 7, we argue how our results may lead to faster algorithms when using fast matrix multiplication as in [13].

## 2 Preliminaries

A *line* is a subset of a surface  $\Sigma$  that is homeomorphic to  $[0, 1]$ . A closed curve on  $\Sigma$  that is homeomorphic to a cycle is called *Jordan curve*. A planar graph embedded crossing-free onto the sphere  $\mathbb{S}_0$  is defined as a *plane graph*, where every vertex is a point of  $\mathbb{S}_0$  and each edge a line. In this paper, we consider Jordan curves that intersect with a plane graph only in vertices. For a Jordan curve  $J$ , we denote by  $V(J)$  the vertices  $J$  intersects with. For two Jordan curve  $J, J'$ , we define  $J + J'$  to be the symmetric difference of  $J$  and  $J'$ .

Given a connected graph  $G = (V, E)$ , a set of vertices  $S \subset V$  is called a *separator* if the subgraph induced by  $V \setminus S$  is non-empty and has several components.  $S$  is called an  *$u, v$ -separator* for two vertices  $u$  and  $v$  that are in different components of  $G[V \setminus S]$ .  $S$  is a *minimal  $u, v$ -separator* if no proper subset of  $S$  is a  $u, v$ -separator. Finally,  $S$  is a *minimal separator* of  $G$  if there are two vertices  $u, v$  such that  $S$  is a minimal  $u, v$ -separator. For a vertex subset  $A \subseteq V$ , we *saturate*  $A$  by adding edges between every two non-adjacent vertices, and thus, turning  $A$  into a clique.

A *chord* in a cycle  $C$  of a graph  $G$  is an edge joining two non-consecutive vertices of  $C$ . A graph  $H$  is called *chordal* if every cycle of length  $> 3$  has a chord. A *triangulation* of a graph  $G = (V, E)$  is a chordal graph  $H = (V, E')$  with  $E \subseteq E'$ . The edges of  $E' \setminus E$  are called *fill edges*. We say,  $H$  is a *minimal triangulation* of  $G$  if every graph  $G' = (V, E'')$  with  $E \subseteq E'' \subset E'$  is not chordal. Note that a triangulation of a planar graph may not be planar—not to confuse with the notion of “planar triangulation” that asks for filling the facial cycles with chords. Consider the following algorithm on a graph  $G$  that triangulates  $G$ , known as the *elimination game* [24]. Repeatedly choose a vertex, saturate its neighborhood, and delete it. Terminate when  $V = \emptyset$ . The order in which the vertices are deleted is called the *elimination ordering*  $\alpha$ , and  $G_\alpha^+$  is the chordal graph obtained by adding all saturating (fill) edges to  $G$ . Another way of triangulating a graph  $G$  can be obtained by using a tree-decomposition of  $G$ .

### 2.1 Tree-decompositions

Let  $G$  be a graph,  $T$  a tree, and let  $\mathcal{Z} = (Z_t)_{t \in T}$  be a family of vertex sets  $Z_t \subseteq V(G)$ , called *bags*, indexed by the nodes of  $T$ . The pair  $\mathcal{T} = (T, \mathcal{Z})$  is called a *tree-decomposition* of  $G$  if it satisfies the following three conditions:

- $V(G) = \cup_{t \in T} Z_t$ ,
- for every edge  $e \in E(G)$  there exists a  $t \in T$  such that both ends of  $e$  are in  $Z_t$ ,
- $Z_{t_1} \cap Z_{t_3} \subseteq Z_{t_2}$  whenever  $t_2$  is a vertex of the path connecting  $t_1$  and  $t_3$  in  $T$ .

The width  $\text{tw}(\mathcal{T})$  of the tree-decomposition  $\mathcal{T} = (T, \mathcal{Z})$  is the maximum size over all bags minus one. The *treewidth* of  $G$  is the minimum width over all tree-decompositions.

**Lemma 1.** [8] Let  $\mathcal{T} = (T, \mathcal{Z})$ ,  $\mathcal{Z} = (Z_t)_{t \in T}$  be a tree-decomposition of  $G = (V, E)$ , and let  $K \subseteq V$  be a clique in  $G$ . Then there exists a node  $t \in T$  with  $K \subseteq Z_t$ .

As a consequence, we can turn a graph  $G$  into another graph  $H'$  by saturating the bags of a tree-decomposition, i.e., add an edge in  $G$  between any two non-adjacent vertices that appear in a common bag. Automatically, we get that for every clique  $K$  in  $H'$ , there exists a bag  $Z_t$  such that  $K = Z_t$ . Note that the width of the tree-decomposition is not changed by this operation. It is known (e.g. in [29]) that  $H'$  is a triangulation of  $G$ , actually a so-called *k-tree*. Although there exist triangulations that cannot be computed from  $G$  with the elimination game, van Leeuwen [29] describes how to change a tree-decomposition in order to obtain the elimination ordering  $\alpha$  and thus  $G_\alpha^+ = H'$ . For finding a minimal triangulation  $H$  that is a super-graph of  $G$  and a subgraph of  $G_\alpha^+$ , known as the *sandwich* problem, there are efficient  $O(nm)$  runtime algorithms (For a nice survey, we refer to [19]).

## 2.2 Minimal separators and triangulations

We want to use triangulations for computing tree-decompositions with “nice” separating properties. By Rose et al [26], we have also the following lemma:

**Lemma 2.** Let  $H$  be a minimal triangulation of  $G$ . Any minimal separator of  $H$  is a minimal separator of  $G$ .

Before we give our new tree-decomposition algorithm, we are interested in an additional property of minimal separators. Let  $\mathcal{S}_G$  be the set of all minimal separators in  $G$ . Let  $S_1, S_2 \in \mathcal{S}_G$ . We say that  $S_1$  *crosses*  $S_2$ , denoted by  $S_1 \# S_2$ , if there are two connected components  $C, D \in G \setminus S_2$ , such that  $S_1$  intersects both,  $C$  and  $D$ . If  $S_1$  does not cross  $S_2$ , we say that  $S_1$  is *parallel* to  $S_2$ , denoted by  $S_1 || S_2$ . Note that “||” is an equivalence relation on a set of pairwise parallel separators.

**Theorem 3.** [23] Let  $H$  be a minimal triangulation of  $G$ . Then,  $\mathcal{S}_H$  is a maximal set of pairwise parallel minimal separators in  $G$ .

## 3 Algorithm for a new tree-decomposition

Before we give the whole algorithm, we need some more definitions. For a graph  $G = (V, E)$ , let  $\mathcal{K}$  be the set of *maximal cliques*, that is, the cliques that have no superset in  $V$  that forms a clique in  $G$ . Let  $\mathcal{K}_v$  be the set of all maximal cliques of  $G$  that contain the vertex  $v \in V$ . For a chordal graph  $H$  we define a *clique tree* as a tree  $T = (\mathcal{K}, \mathcal{E})$  whose vertex set is the set of maximal cliques in  $H$ , and  $T[\mathcal{K}_v]$  forms a connected subtree for each vertex  $v \in V$ . Vice versa, if a graph  $H$  has a clique tree, then  $H$  is chordal. Even though finding all maximal cliques of a graph is NP-hard in general, there exists a linear time modified algorithm of [27], that exploits the property of chordal graphs having at most  $|V|$  maximal cliques. By definition, a clique tree of  $H$  is also a tree-decomposition of  $H$  (where the opposite is not necessarily true). With [5], we obtain a linear time algorithm computing the clique tree of a graph  $H$ . It follows immediately from Lemma 1 that the treewidth of any chordal graph  $H$  equals the size of the largest clique. Let us define an edge  $(C_i, C_j)$  in a clique tree  $T$  to be equivalent to the set of vertices  $C_i \cap C_j$  of the two cliques  $C_i, C_j$  in  $H$  which correspond to the endpoints of the edge in  $T$ . For us, the most interesting property of clique trees is given by [20]:

**Theorem 4.** *Given a chordal graph  $H$  and any clique tree  $T$  of  $H$ , a set of vertices  $S$  is a minimal separator of  $H$  if and only if  $S = C_i \cap C_j$  for an edge  $(C_i, C_j)$  in  $T$ .*

We get our lemma following from Theorem 3 and Theorem 4:

**Lemma 5.** *Given a clique tree  $T = (\mathcal{K}, \mathcal{E})$  of a minimal triangulation  $H$  of a graph  $G$ . Transform  $T$  into a tree-decomposition  $\mathcal{T}$  of  $G$ , where  $\text{tw}(\mathcal{T}) = \text{tw}(H)$ , by deleting the fill edges from all vertex sets in  $\mathcal{K}$ . Then the set of all edges  $(C_i, C_j)$  in  $T$  form a maximal set of pairwise parallel minimal separators in  $G$ .*

We call such a tree-decomposition of  $G$  *parallel*. We give the algorithm in Figure 1.

**Algorithm TransfTD**

Input: Graph  $G = (V, E)$  with tree-decomposition  $\mathcal{T} = (T, \mathcal{Z})$ ,  $\mathcal{Z} = (Z_t)_{t \in T}$ .

Output: Parallel tree-decomposition  $\mathcal{T}'$  of  $G$  with  $\text{tw}(\mathcal{T}') \leq \text{tw}(\mathcal{T})$ .

Triangulation step: Saturate every bag  $Z_t$ ,  $t \in T$ ;

Output the chordal graph  $H' = (V, E \cup F)$  with fill edges  $F$ .

Minimal triangulation step: Compute a minimal triangulation  $H = (V, E \cup F')$  of  $G$  with  $F' \subseteq F$ .

Clique tree step: Compute the clique tree of  $H$  and turn it into a tree-decomposition  $\mathcal{T}'$  of  $G$ .

Figure 1: Algorithm **TransfTD**.

The worst case analysis for the runtime of **TransfTD** comes from the Minimal triangulation step, that needs time  $O(nm)$  for an input graph  $G = (V, E)$ , ( $|V| = n$ ,  $|E| = m$ ).

## 4 Plane graphs and minimal separators

In the remainder of the paper, we consider 2-connected plane graphs. The Jordan curve theorem (a.o. [12]) states that a Jordan curve  $J$  on a sphere  $\mathbb{S}_0$  divides the rest of  $\mathbb{S}_0$  into two connected parts, namely into two open discs  $\Delta_J$  and  $\Delta_{\bar{J}}$ , i.e.,  $\Delta_J \cup \Delta_{\bar{J}} \cup J = \mathbb{S}_0$ . Hence, every Jordan curve  $J$  is a separator of a plane graph  $G$ . Bouchitté et al [9] use results of [17] to show the following:

**Lemma 6.** [9] *Every minimal separator  $S$  of a 2-connected plane graph  $G$  forms the vertices of a Jordan curve.*

That is, in any crossing-free embedding of  $G$  in  $\mathbb{S}_0$ , one can find a Jordan curve only intersecting with  $G$  in the vertices of  $S$ . Note that a minimal separator  $S$  is not necessarily forming a unique Jordan curve. If an induced subgraph  $G'$  of  $G$  (possibly a single edge) has only two vertices  $u, v$  in common with  $S$ , and  $u, v$  are successive vertices of the Jordan curve  $J$ , then  $G'$  can be drawn on either side of  $J$ . This is the only freedom we have to form a Jordan curve in  $G$ , since on one side of  $J$ , there is a connected subgraph of  $G$  that is adjacent to all vertices of  $J$ . We call two Jordan curves  $J, J'$  *equivalent* if they share the same vertex set and intersect the vertices in the same order. Two Jordan curves  $J, J'$  *cross* if  $J$  and  $J'$  are not equivalent and one has vertices of the other on both sides.

**Lemma 7.** *Let  $S_1, S_2$  be two minimal separators of a 2-connected plane graph  $G = (V, E)$  and each  $S_i$  forms a Jordan curve  $J_i$ ,  $i = 1, 2$ . If  $S_1 \parallel S_2$ , then  $J_1, J_2$  are non-crossing. Vice versa, if two Jordan curves  $J_1, J_2$  in  $G$  are non-crossing and  $\Delta_{J_1} \cap V$  and  $\Delta_{\bar{J}_1} \cap V$ , ( $i = 1, 2$ ) all are non-empty, then the vertex sets  $S_i = V(J_i)$ , ( $i = 1, 2$ ) are parallel separators.*

We say two non-crossing Jordan curves  $J_1, J_2$  *touch* one another if they intersect in a non-empty vertex set and if the symmetric difference  $J_1 + J_2$  is the (possibly empty) union of Jordan curves, each having a single edge on one side.

**Lemma 8.** *Let two non-crossing Jordan curves  $J_1, J_2$  be formed by two parallel separators  $S_1, S_2$  of a 2-connected plane graph  $G$ . If  $J_1$  and  $J_2$  touch, and there exists a Jordan curve  $J_3 \subseteq J_1 + J_2$  such that there are vertices of  $G$  on both sides of  $J_3$ , then the vertices of  $J_3$  form another separator  $S_3$  that is parallel to  $S_1$  and  $S_2$ .*

If  $J_1 + J_2$  forms exactly one Jordan curve  $J_3$  then we say  $J_1$  touches  $J_2$  *nicely*. Note that if  $J_1$  and  $J_2$  only touch in one vertex, the vertices of  $J_1 + J_2$  may not form any Jordan curve. The following lemma gives a property of “nicely touching” that we need later on.

**Lemma 9.** *If in a 2-connected plane graph  $G$ , two non-crossing Jordan curves  $J_1$  and  $J_2$  touch nicely, then  $|V(J_1) \cap V(J_2) \cap V(J_1 + J_2)| \leq 2$ .*

## 5 Jordan curves and geometric tree-decompositions

We now want to turn a parallel tree-decomposition  $\mathcal{T}$  into a *geometric* tree-decomposition  $\mathcal{T}' = (T, \mathcal{Z})$ ,  $\mathcal{Z} = (Z_t)_{t \in T}$  where  $T$  is a ternary tree and for every two adjacent edges  $(Z_r, Z_s)$  and  $(Z_r, Z_t)$  in  $T$ , the minimal separators  $S_1 = Z_r \cap Z_s$  and  $S_2 = Z_r \cap Z_t$  form two Jordan curves  $J_1, J_2$  that touch each other nicely. Unfortunately, we cannot arbitrarily connect two Jordan curves  $J, J'$  that we obtain from the parallel tree-decomposition  $\mathcal{T}$ —even if they touch nicely, since the symmetric difference of  $J, J'$  may have more vertices than  $\text{tw}(T)$ . We now give some structural results that will enable us to find above tree-decomposition.

Define for a vertex set  $Z \subseteq V$  the subset  $\partial Z \subseteq Z$  to be the vertices adjacent in  $G$  to some vertices in  $V \setminus Z$ . Let  $G$  be planar embedded,  $Z$  connected, and  $\partial Z$  form a Jordan curve. We define  $\bar{\Delta}_Z$  to be the closed disk, onto which  $Z$  is embedded and  $\Delta_Z$  the open disk with the embedding of  $Z$  without the vertices of  $\partial Z$ . For a non-leaf tree node  $X$  with degree  $d$  in a parallel tree-decomposition  $\mathcal{T}$ , let  $Y_1, \dots, Y_d$  be its neighbors. Let  $T_{Y_i}$  be the subtree including  $Y_i$  when removing the edge  $(Y_i, X)$  from  $T$ . We define  $G_{Y_i} \subseteq G$  to be the subgraph induced by the vertices of all bags in  $T_{Y_i}$ . For  $Y_i$ , choose the Jordan curve  $J_i$  formed by the vertex set  $\partial Y_i = Y_i \cap X$  to be the Jordan curve that has all vertices of  $G_{Y_i}$  on one side and  $V(G) \setminus V(G_{Y_i})$  on the other. For each edge  $e$  with both endpoints being consecutive vertices of  $J_i$  we choose if  $e \in E(G_{Y_i})$  or if  $e \in E(G) \setminus E(G_{Y_i})$ .

We say, a set  $\mathcal{J}$  of non-crossing Jordan curves is *connected* if for every partition of  $\mathcal{J}$  into two subsets  $\mathcal{J}_1, \mathcal{J}_2$ , there is a Jordan curve of  $\mathcal{J}_1$  that touches a Jordan curve of  $\mathcal{J}_2$ . A set  $\mathcal{J}$  of Jordan curves is *k-connected* if for every partition of  $\mathcal{J}$  into two connected sets  $\mathcal{J}_1, \mathcal{J}_2$ , the Jordan curves of  $\mathcal{J}_1$  touch the Jordan curves of  $\mathcal{J}_2$  in at least  $k$  vertices. Note that if two Jordan curves touch nicely then they intersect in at least two vertices. The proofs of the followings lemmas can be found in the appendix.

**Lemma 10.** *For every inner node  $X$  of a parallel tree-decomposition  $\mathcal{T}$  of a 2-connected plane graph, the collection  $\mathcal{J}_X$  of pairwise non-crossing Jordan curves formed by  $\partial X$  is 2-connected.*

**Lemma 11.** *Every bag  $X$  in a parallel tree-decomposition  $\mathcal{T}$  can be decomposed into  $X_1, \dots, X_\ell$  such that each vertex set  $\partial X_i$  forms a Jordan curve in  $G$  and  $\bigcup_{i=1}^\ell \partial X_i = \partial X$ .*

**Lemma 12.** *In a decomposition of the sphere  $\mathbb{S}_0$  by a 2-connected collection  $\mathcal{J}$  of non-crossing Jordan curves, one can repeatedly find two Jordan curves  $J_1, J_2 \in \mathcal{J}$  that touch nicely, and substitute  $J_1$  and  $J_2$  by  $J_1 + J_2$  in  $\mathcal{J}$ .*



We get that  $X_1, \dots, X_\ell$  and  $G_{Y_1}, \dots, G_{Y_d}$  are embedded inside of closed disks each bounded by a Jordan curve. Thus, the union  $\mathcal{D}$  over all these disks together with the Jordan curves  $\mathcal{J}_X$  fill the entire sphere  $\mathbb{S}_0$  onto which  $G$  is embedded. Each subgraph embedded onto  $\Delta \cup J$  for a disk  $\Delta \in \mathcal{D}$  and a Jordan curve  $J$  bounding  $\Delta$ , forms either a bag  $X_i$  or a subgraph  $G_{Y_j}$ . Define the collection of bags  $\mathcal{Z}^X = \{X_1, \dots, X_\ell, Y_1, \dots, Y_d\}$ . In Figure 2, we give the algorithm **TransfTD II** for creating a geometric tree-decomposition using the idea of Lemma 8.

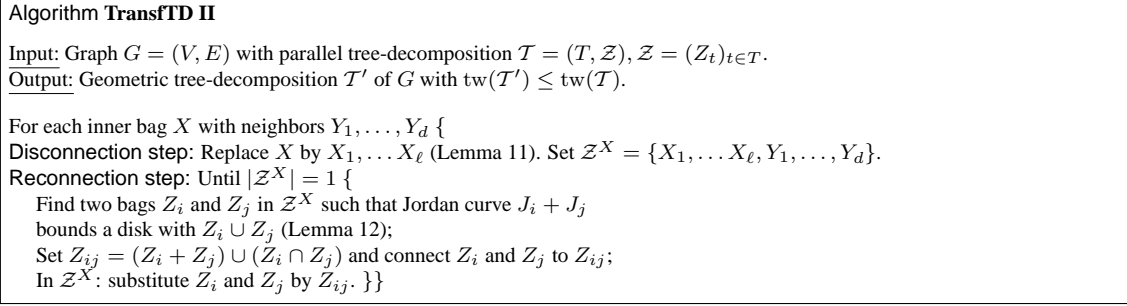


Figure 2: Algorithm **TransfTD II**.

Since by Lemma 9,  $|V(\partial Z_i \cap \partial Z_j \cap \partial Z_{ij})| \leq 2$ , we have that at most two vertices in all three bags are contained in any other bag of  $\mathcal{Z}^X$ . Note that geometric tree-decompositions have a lot in common with *sphere-cut decompositions* (introduced in [15]), namely that both decompositions are assigned with vertex sets that form “sphere-cutting” Jordan curves. For our new dynamic programming algorithm, we use much of the structured results obtained in [15].

## 6 Jordan curves and dynamic programming

We show how to improve the existing algorithm of Alber et al [1] for weighted PLANAR DOMINATING SET. The algorithm is based on dynamic programming on *nice tree-decompositions* and has the running time  $4^{\text{tw}(\mathcal{T})} \cdot n^{O(1)}$ . Our algorithm is a dynamic programming approach on geometric tree-decompositions of time  $3^{\text{tw}(\mathcal{T})} \cdot n^{O(1)}$ . Since it is of similar structure to those in [15] and [18], we restate the technique in the appendix and give here only the new idea. Namely, to exploit the planar structure of the nicely touching separators to improve upon the runtime. In [1], the worst case in the runtime for PLANAR DOMINATING SET is determined by the number of vertices that are in the intersection of three adjacent bags  $r, s, t$ . Using the notion of [15] for a geometric tree-decomposition, we partition the vertex sets of three bags  $Z_r, Z_s, Z_t$  into sets  $L, R, F, I$ , where  $Z_r$  is adjacent to  $Z_s, Z_t$ . The sets  $L, R, F$  represent the vertices that are in exactly two of the bags. Let us consider the *Intersection* set  $I := \partial Z_r \cap \partial Z_s \cap \partial Z_t$ . By Lemma 9,  $|I| \leq 2$ . Thus,  $I$  is not any more part of the runtime calculation for which we refer the reader to the appendix.

## 7 Conclusion

A natural question to pose, is it possible to solve PLANAR DOMINATING SET in time  $2.99^{\text{tw}(\mathcal{T})} \cdot n^{O(1)}$  and equivalently, PLANAR INDEPENDENT SET in  $1.99^{\text{tw}(\mathcal{T})} \cdot n^{O(1)}$ ? Though, we cannot give a positive

answer yet, we have a formula that needs the property “well-balanced” separators in a geometric tree-decomposition  $\mathcal{T}$ : we assume that the three sets  $L, R, F$  are of equal cardinality for every three adjacent bags. Since  $|L| + |R| + |F| \leq \text{tw}$ , we thus have that  $|L|, |R|, |F| \leq \frac{\text{tw}}{3}$ . Applying the fast matrix multiplication method from [13] for example to PLANAR INDEPENDENT SET, this leads to a  $2^{\frac{\omega}{3} \text{tw}(\mathcal{T})} \cdot n^{O(1)}$  algorithm, where  $\omega < 2.376$ . Does every planar graph have a geometric tree-decomposition with well-balanced separators?

**Acknowledgments.** The author thanks Frédéric Mazoit for some enlightening discussion on Theorem 3.

## References

- [1] J. ALBER, H. L. BODLAENDER, H. FERNAU, T. KLOKS, AND R. NIEDERMEIER, *Fixed parameter algorithms for dominating set and related problems on planar graphs*, *Algorithmica*, 33 (2002), pp. 461–493. 1, 6, 8.5
- [2] J. ALBER, H. L. BODLAENDER, H. FERNAU, AND R. NIEDERMEIER, *Fixed parameter algorithms for planar dominating set and related problems.*, in *Algorithm Theory - SWAT 2000, 7th Scandinavian Workshop on Algorithm Theory*, vol. 1851 of *Lecture Notes in Computer Science*, Springer, 2000, pp. 97–110. 1
- [3] J. ALBER AND R. NIEDERMEIER, *Improved tree decomposition based algorithms for domination-like problems*, in *LATIN’02: Theoretical informatics (Cancun)*, vol. 2286 of *Lecture Notes in Computer Science*, Berlin, 2002, Springer, pp. 613–627. 1
- [4] S. ARORA, M. GRIGNI, D. R. KARGER, P. N. KLEIN, AND A. WOLOSZYN, *A polynomial-time approximation scheme for weighted planar graph TSP.*, in *Proceedings of the Ninth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 1998, pp. 33–41. 1
- [5] P. A. BERNSTEIN AND N. GOODMAN, *Power of natural semijoins.*, *SIAM Journal on Computing*, 10 (1981), pp. 751–771. 3
- [6] H. L. BODLAENDER, *A tourist guide through treewidth*, *Acta Cybernet.*, 11 (1993), pp. 1–21. 1
- [7] H. L. BODLAENDER, *Treewidth: Algorithmic techniques and results.*, in *MFCS’97: Mathematical Foundations of Computer Science 1997, 22nd International Symposium (MFCS)*, vol. 1295 of *Lecture Notes in Computer Science*, Springer, 1997, pp. 19–36. 1
- [8] H. L. BODLAENDER AND R. H. MÖHRING, *The pathwidth and treewidth of cographs.*, *SIAM Journal on Discrete Mathematics*, 6 (1993), pp. 181–188. 1
- [9] V. BOUCHITTÉ, F. MAZOIT, AND I. TODINCA, *Chordal embeddings of planar graphs.*, *Discrete Mathematics*, 273 (2003), pp. 85–102. 4, 6
- [10] W. COOK AND P. SEYMOUR, *Tour merging via branch-decomposition*, *INFORMS Journal on Computing*, 15 (2003), pp. 233–248. 1
- [11] E. D. DEMAINE, F. V. FOMIN, M. HAJIAGHAYI, AND D. M. THILIKOS, *Subexponential parameterized algorithms on graphs of bounded genus and  $H$ -minor-free graphs*, *Journal of the ACM*, 52 (2005), pp. 866–893. 1
- [12] R. DIESTEL, *Graph theory, Third edition*, Springer-Verlag, Heidelberg, 2005. 4
- [13] F. DORN, *Dynamic programming and fast matrix multiplication.*, in *Algorithms - ESA 2006, 14th Annual European Symposium (ESA)*, vol. 4168 of *Lecture Notes in Computer Science*, Springer, 2006, pp. 280–291. 1, 1, 7
- [14] F. DORN, F. V. FOMIN, AND D. M. THILIKOS, *Fast subexponential algorithm for non-local problems on graphs of bounded genus*, in *Proceedings of the 10th Scandinavian Workshop on Algorithm Theory (SWAT 2006)*, vol. 4059 of *LNCS*, Springer, Berlin, 2006, pp. 172–183. 1

- [15] F. DORN, E. PENNINKX, H. L. BODLAENDER, AND F. V. FOMIN, *Efficient exact algorithms on planar graphs: Exploiting sphere cut branch decompositions*, in Proceedings of the 13th Annual European Symposium on Algorithms (ESA 2005), vol. 3669 of LNCS, Springer, Berlin, 2005, pp. 95–106. 1, 1, 5, 6
- [16] F. DORN AND J. A. TELLE, *Two birds with one stone: The best of branchwidth and treewidth with one algorithm.*, in LATIN 2006: Theoretical Informatics, 7th Latin American Symposium, vol. 3887 of Lecture Notes in Computer Science, Springer, 2006, pp. 386–397. 1
- [17] D. EPPSTEIN, *Subgraph isomorphism in planar graphs and related problems*, J. Graph Algorithms Appl., 3 (1999), pp. 1–27. 4
- [18] F. V. FOMIN AND D. M. THILIKOS, *Dominating sets in planar graphs: branch-width and exponential speed-up*, in SODA'03: Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms (Baltimore, MD, 2003), New York, 2003, ACM, pp. 168–177. 1, 1, 6
- [19] P. HEGGERNES, *Minimal triangulations of graphs: A survey*, Discrete Mathematics, 306 (2006), pp. 297–317. 2.1
- [20] C. W. HO AND R. C. T. LEE, *Counting clique trees and computing perfect elimination schemes in parallel*, Inf. Process. Lett., 31 (1989), pp. 61–68. 3
- [21] R. J. LIPTON AND R. E. TARJAN, *A separator theorem for planar graphs*, SIAM J. Appl. Math., 36 (1979), pp. 177–189. 1
- [22] G. L. MILLER, *Finding small simple cycle separators for 2-connected planar graphs.*, Journal of Computer and System Science, 32 (1986), pp. 265–279. 1
- [23] A. PARRA AND P. SCHEFFLER, *Characterizations and algorithmic applications of chordal graph embeddings.*, Discrete Applied Mathematics, 79 (1997), pp. 171–188. 3
- [24] S. PARTER, *The use of linear graphs in Gauss elimination*, SIAM Review, 3 (1961), pp. 119–130. 2
- [25] N. ROBERTSON AND P. D. SEYMOUR, *Graph minors. X. Obstructions to tree-decomposition*, J. Combin. Theory Ser. B, 52 (1991), pp. 153–190. 1
- [26] D. ROSE, R. E. TARJAN, AND G. LUEKER, *Algorithmic aspects of vertex elimination on graphs*, SIAM Journal on Computing, 5 (1976), pp. 146–160. 2.2
- [27] R. E. TARJAN AND M. YANNAKAKIS, *Simple linear-time algorithms to test chordality of graphs, test acyclicity of hypergraphs, and selectively reduce acyclic hypergraphs.*, SIAM Journal on Computing, 13 (1984), pp. 566–579. 3
- [28] J. A. TELLE AND A. PROSKUROWSKI, *Algorithms for vertex partitioning problems on partial  $k$ -trees*, SIAM J. Discrete Math, 10 (1997), pp. 529–550. 1
- [29] J. VAN LEEUWEN, *Graph algorithms*, MIT Press, Cambridge, MA, USA, 1990. 1, 2.1

## 8 Appendix

### 8.1 Proof of Lemma 8

Let  $G_i, \bar{G}_i$  be the subgraphs of  $G$  separated by  $J_i (i = 1, 2)$ . Since the vertex set  $V(J_3)$  is a subset of  $V(J_1) \cup V(J_2)$  we have that  $V(J_3) \cap (V(G_i) \cup V(\bar{G}_i)) = \emptyset (i = 1, 2)$ . Hence  $S_3 = V(J_3)$  is parallel to both,  $S_i = V(J_i) (i = 1, 2)$ .

### 8.2 Proof of Lemma 10

We first show that  $\mathcal{J}_X$  is connected. Assume that  $\mathcal{J}_X$  is not connected, that is, there is a partition of  $\mathcal{J}_X$  into  $\mathcal{J}_1, \mathcal{J}_2$  such that  $\mathcal{J}_1$  is connected but no Jordan curve of  $\mathcal{J}_1$  touches any Jordan curve of  $\mathcal{J}_2$ . We have two cases: first assume

that no vertex of the Jordan curves of  $\mathcal{J}_1$  is adjacent to any vertex in a Jordan curve of  $\mathcal{J}_2$ . Each vertex of the Jordan curves of  $\mathcal{J}_1$  is adjacent to some vertices in  $X_0 := X \setminus \bigcup_{k=1}^d Y_k$ , for the neighbors  $Y_1, \dots, Y_d$  of  $X$ . Hence, there is a Jordan curve  $J_0$  formed exclusively by vertices in  $X_0$  such that  $\mathcal{J}_1$  is on one side of  $J_0$  and  $\mathcal{J}_2$  on the other. Suppose, there is a pair of vertices  $u, v$  where  $u$  is a vertex of some  $Y_i$  bounded by the Jordan curve  $J_i \in \mathcal{J}_1$  and  $v$  is a vertex of some  $Y_j$  bounded by the Jordan curve  $J_j \in \mathcal{J}_2$ . By Lemma 7,  $J_0$  is non-crossing  $J_i$  and  $J_j$ . Choose  $J_0$  minimal, i.e., no subset of  $V(J_0)$  forms a Jordan curve. Thus,  $V(J_0) \subseteq X_0$  is a minimal  $u, v$ -separator that is parallel to the maximal  $\mathcal{S}_G$  set of pairwise parallel minimal separators in  $G$ . That is contradicting the maximality of  $\mathcal{S}_G$ . For the second case assume there are some edges  $E_J \subseteq E(X)$  between Jordan curves in  $\mathcal{J}_1$  and Jordan curves in  $\mathcal{J}_2$ . Then there is a closed curve  $C_J$  separating  $\mathcal{J}_1$  from  $\mathcal{J}_2$  touching some (or none) vertices of  $X_0$  and crossing the edges of  $E_J$ . Turn  $C_J$  into a Jordan curve  $J_{1,2}$ : for each crossed edge  $e$ , move the curve to one endpoint of  $e$ , alternately to a vertex of  $\mathcal{J}_1$  and a vertex of  $\mathcal{J}_2$ . Then,  $J_{1,2}$  is neither an element of  $\mathcal{J}_1$  nor of  $\mathcal{J}_2$ , and with Lemma 7 and the same arguments as above,  $V(J_{1,2})$  is a minimal separator parallel to  $\mathcal{S}_G$  what again is a contradiction to the maximality of  $\mathcal{S}_G$ .

Now we prove that  $\mathcal{J}_X$  is 2-connected. First note that  $G$  itself is 2-connected. Thus, if  $\mathcal{J}$  is only 1-connected, there must be a path (or edge) in  $X_0$  from some partition  $\mathcal{J}_1$  to  $\mathcal{J}_2$ , if  $\mathcal{J}_1$  and  $\mathcal{J}_2$  intersect only in one vertex. The proof is very similar to the first case, so we only sketch it. The only difference is that we now assume that there is one vertex  $w$  in the intersection of the Jordan curves of  $\mathcal{J}_1$  with those of  $\mathcal{J}_2$ . As in both previous cases, we find a minimal separator  $S$ . In the first case,  $S \subseteq X_0 \cup \{w\}$  and in the second  $S \subseteq X_0 \cup \{w\} \cup V(E_J)$  for the edges  $E_J$  with one endpoint in  $\mathcal{J}_1$  and the other in  $\mathcal{J}_2$ . Again, we obtain a contradiction since  $S$  is parallel to  $\mathcal{S}_G$ .

### 8.3 Proof of Lemma 11

Let  $X$  have neighbors  $Y_1, \dots, Y_d$ . Due to Lemma 10,  $\partial X$  forms a 2-connected set of Jordan curves, each bounding a disk inside which one of the subgraphs  $G_{Y_j}$  is embedded onto. If we remove the disks  $\Delta_{Y_j}$  for all  $1 \leq j \leq d$  and the set of Jordan curves  $\mathcal{J}_X$  from the sphere, we obtain a collection  $\mathcal{D}_X$  of  $\ell$  disjoint open disks each bounded by a Jordan curve of  $\mathcal{J}_X$ . Note that  $\ell \leq \max\{d, |X|\}$ . Let  $Z_i$  be the subgraph in  $X \cap \Delta_i$  for such an open disk  $\Delta_i \in \mathcal{D}_X$  for  $1 \leq i \leq \ell$ . Then each  $Z_i$  is either empty or consisting only of edges or subgraphs of  $G$  and the closed disk  $\bar{\Delta}_i$  is bounded by a Jordan curve  $J_i$  formed by a subset of  $\partial X$ . We set  $X_i = Z_i \cup V(J_i)$  with  $\partial X_i$  the vertices of  $J_i$ .

### 8.4 Proof of Lemma 12

Removing  $\mathcal{J}$  from  $\mathbb{S}_0$  decomposes  $\mathbb{S}_0$  into a collection  $\mathcal{D}$  of open discs each bounded by a Jordan curve in  $\mathcal{J}$ . For each  $\Delta_1 \in \mathcal{D}$  bounded by  $J_1 \in \mathcal{J}$  there is a “neighboring” disk  $\Delta_2 \in \mathcal{D}$  bounded by  $J_2 \in \mathcal{J}$  such that the intersection  $J_1 \cap J_2$  forms a line of  $\mathbb{S}_0$ . Then,  $J_1 + J_2$  bounds  $\Delta_1 \cup \Delta_2$ . Replace,  $J_1, J_2$  by  $J_3$  in  $\mathcal{J}$  and continue until  $|\mathcal{J}| = 1$ , that is, we are left with one Jordan curve separating  $\mathbb{S}_0$  into two open disks.

### 8.5 Algorithm of Section 6

Given a geometric tree-decomposition  $\mathcal{T} = (T, \mathcal{Z})$ ,  $\mathcal{Z} = (Z_t)_{t \in T}$ , we root  $T$  by arbitrarily choosing a node  $r$  as a *root*. Each internal node  $t$  of  $T$  now has one adjacent node on the path from  $t$  to  $r$ , called the *parent node*, and two adjacent nodes toward the leaves, called the *children nodes*. To simplify matters, we call them the *left child* and the *right child*.

Let  $T_t$  be a subtree of  $T$  rooted at node  $t$ .  $G_t$  is the subgraph of  $G$  induced by all bags of  $T_t$ . For a subset  $U$  of  $V(G)$  let  $w(U)$  denote the total weight of vertices in  $U$ . That is,  $w(U) = \sum_{u \in U} w_u$ . Define a set of subproblems for each subtree  $T_t$ .

Alber et al. [1] introduced the so-called “monotonicity”-property of domination-like problems for their dynamic programming approach that we will use, too. For every node  $t \in T$ , we use three colors for the vertices of bag  $Z_t$ :

**black:** represented by 1, meaning the vertex is in the dominating set.

**white:** represented by 0, meaning the vertex has a neighbor in  $G_t$  that is in the dominating set.

**gray:** represented by 2, meaning the vertex has a neighbor in  $G$  that is in the dominating set.

For a bag  $Z_t$  of cardinality  $\ell$ , we define a *coloring*  $c(Z_t)$  to be a mapping of the vertices  $Z_t$  to an  $\ell$ -vector over the color-set  $\{0, 1, 2\}$  such that each vertex  $u \in Z_t$  is assigned a color, i.e.,  $c(u) \in \{0, 1, 2\}$ . We further define the weight  $w(c(Z_t))$  to be the minimum weight of the vertices of  $G_t$  in the minimum weight dominating set with respect to the coloring  $c(Z_t)$ . If no such dominating set exists, we set  $w(c(Z_t)) = +\infty$ . We store all colorings of  $Z_t$ , and for two child nodes, we update each two colorings to one of the parent node.

Before we describe the update-process, let us make the following comments:

We defined the color “gray” according to the monotonicity property: for a vertex  $u$  colored gray, we do not have (or store) the information if  $u$  is already dominated by a vertex in  $G_t$  or if  $u$  still has to be dominated in  $G \setminus G_t$ . Thus, a solution with a vertex  $v$  colored white has at least the same the weight as the same solution with  $v$  colored gray.

By the definition of bags, for three adjacent nodes  $r, s, t$ , the vertices of  $\partial Z_r$  have to be in at least on of  $\partial Z_s$  and  $\partial Z_t$ . The reader may simply recall that the parent bag is formed by the union of the vertices of two nicely touching Jordan curves.

For the sake of a refined analysis, we partition the bags of parent node  $r$  and left child  $s$  and right child  $t$  into four sets  $L, R, F, I$  as follows:

- *Intersection*  $I := \partial Z_r \cap \partial Z_s \cap \partial Z_t$ ,
- *Forget*  $F := (Z_s \cup Z_t) \setminus \partial Z_r$ ,
- *Symmetric difference*  $L := \partial Z_r \cap \partial Z_s \setminus I$  and  $R := \partial Z_r \cap \partial Z_t \setminus I$ .

We define  $F'$  to be actually those vertices of  $F$  that are only in  $(\partial Z_s \cup \partial Z_t) \setminus \partial Z_r$ . The vertices of  $F \setminus F'$  do not exist in  $Z_r$  and hence are irrelevant for the continuous update process. We say that a coloring  $c(Z_r)$  is *formed* by the colorings  $c_1(Z_s)$  and  $c_2(Z_t)$  subject to the following rules:

- (R1) For every vertex  $u \in L \cup R$ :  $c(u) = c_1(u)$  and  $c(u) = c_2(u)$ , respectively.
- (R2) For every vertex  $u \in F'$  either  $c(u) = c_1(u) = c_2(u) = 1$  or  $c(u) = 0 \wedge c_1(u), c_2(u) \in \{0, 2\} \wedge c_1(u) \neq c_2(u)$ .
- (R3) For every vertex  $u \in I$   $c(u) \in \{1, 2\} \Rightarrow c(u) = c_1(u) = c_2(u)$  and  $c(u) = 0 \Rightarrow c_1(u), c_2(u) \in \{0, 2\} \wedge c_1(u) \neq c_2(u)$ .

We define  $U_c$  to be the vertices  $u \in Z_s \cap Z_t$  for which  $c(u) = 1$  and update the weights by:

$$w(c(Z_r)) = \min\{w(c_1(Z_s)) + w(c_2(Z_t)) - w(U_c) \mid c_1, c_2 \text{ forms } c\}$$

The number of steps by which  $w(c(Z_r))$  is computed for every possible coloring of  $Z_r$  is given by the number of ways a color  $c$  can be formed by the three rules (R1), (R2), (R3), i.e.,

$$3^{|L|+|R|} \cdot 3^{|F'|} \cdot 4^{|I|}$$

steps.

By Lemma 9,  $|I| \leq 2$  and since  $|L| + |R| + |F| \leq \text{tw}(\mathcal{T})$  we need at most  $3^{\text{tw}(\mathcal{T})} \cdot n$  steps to computes all weights  $w(c(Z_r))$  that are usually stored in a table assigned to bag  $Z_r$ .