Optimization and approximation on systems of geometric objects

van Leeuwen, E.J.

Chapter 5

Algorithms on Unit Disk Graph Decompositions

Decompositions form an important way to optimally solve graph optimization problems. Well known decompositions include branch and tree decompositions. We describe a new graph decomposition, called a *relaxed tree decomposition*, and use it to define the relaxed treewidth of a graph. The relaxed treewidth of a graph is no larger than the (ordinary) treewidth and might be up to a factor of two smaller. Hence it could also be smaller than the branchwidth. We investigate the relation of relaxed treewidth to ordinary treewidth and to the strong treewidth of a graph.

Relaxed tree decompositions and tree decompositions have a different structure. We need new algorithms that take a relaxed tree decomposition as input. We give such algorithms for Maximum Independent Set, Minimum Vertex Cover, and Minimum (Connected) Dominating Set. The performance of these algorithms matches that of algorithms using just a tree decomposition.

The motivation for studying relaxed tree decompositions is that they arise in a natural way in unit disk graphs. We propose a geometric parameter of unit disks, the so-called *thickness*, which translates into an upper bound on the relaxed treewidth. We use this to give algorithms for Maximum Independent Set, Minimum Vertex Cover, and Minimum (Connected) Dominating Set on unit disk graphs of bounded thickness. We show that for Minimum Connected Dominating Set one can improve significantly on the analysis in the general case if the input graph is a unit disk graph. By applying noncrossing partitions instead of general partitions we prove that unit disk graphs of bounded thickness have a relaxed tree decomposition with Catalan structure.

We start with a description and analysis of the graph decompositions we use. The definition of thickness follows in Section 5.2. Section 5.3 gives algorithms on general graphs and relaxed tree decompositions. Improved analysis for unit disk graphs of bounded thickness is offered in Section 5.4.

5.1 Graph Decompositions

The algorithms of this chapter use various decompositions of the graph. Although the definitions of tree and branch decompositions are widely known by

now, we give them here for completeness.

Definition 5.1.1 ([230]) *A* branch decomposition (T, l) *of a graph* G *is a ternary tree* T *and a bijection* l *between the edges of* $E(G)$ *and the leafs of* T*. Associated with every edge* $e \in E(T)$ *is the* middle set *of* e*, defined as the set of vertices in* $V(G)$ *for which there are two incident edges* e_1, e_2 *such that the leafs* $l(e_1)$ *and* $l(e_2)$ *are in different components of* $T - e$*. The* width *of a branch decomposition is the maximum cardinality of any middle set. The* branchwidth $\mathrm{bw}(G)$ *of a graph* G *is the minimum width of any branch decomposition of* G*.*

Definition 5.1.2 ([229]) *A* tree decomposition (T, X) *of a graph* G *is a tree* T *and a collection of* bags $X_t \subseteq V(G)$*, one for each vertex* $t \in V(T)$*, satisfying three conditions:*

(i). $\bigcup_{t \in V(T)} X_t = V(G)$*,*

(ii). for all $(u, v) \in E(G)$*, there is a vertex* $t \in V(T)$ *such that* $u, v \in X_t$*, and*

(iii). $X_t \cap X_{t''} \subseteq X_{t'}$ *for any* $t, t'' \in V(T)$ *and for any* $t' \in V(T)$ *on the* t*–*t'' *path in* T*.*

The width *of a tree decomposition is* $\max_{t \in V(T)}\{|X_t|\} - 1$*. The* treewidth $\mathrm{tw}(G)$ *of a graph* G *is the minimum width of any tree decomposition of* G*.*

If the tree of a tree decomposition is in fact a path, we speak of a *path decomposition*. The *pathwidth* $\mathrm{pw}(G)$ of a graph G is the minimum width of any path decomposition of G.

The notions of branch-, tree-, and pathwidth are closely related.

Theorem 5.1.3 ([230, 36]) *For any graph* G*,* $\max\{\mathrm{bw}(G), 2\} \leq \mathrm{tw}(G)+1 \leq \max\{\lfloor \frac{3}{2} \cdot \mathrm{bw}(G) \rfloor, 2\}$ *and* $\mathrm{tw}(G) \leq \mathrm{pw}(G) \leq \mathrm{tw}(G) \cdot \log |V(G)|$*.*

See Bodlaender [36, 37] for surveys on these graph decompositions.

The problems discussed in this chapter all have algorithms that leverage bounds on the branch-, tree-, or pathwidth. We summarize the running times of the current best results in the following table.

	Branchwidth	Treewidth	Pathwidth
Max. Indep. Set	$2^{\mathrm{bw}(G)\omega/2}$ [89]	$2^{\mathrm{tw}(G)}$ [249]	$2^{\mathrm{pw}(G)}$ [249]
Min. Vertex Cover	$2^{\mathrm{bw}(G)\omega/2}$ [89]	$2^{\mathrm{tw}(G)}$ [249]	$2^{\mathrm{pw}(G)}$ [249]
Min. Dom. Set	$2^{2\cdot\mathrm{bw}(G)}$ [89]	$2^{2\cdot\mathrm{tw}(G)}$ [6, 8]	$3^{\mathrm{pw}(G)}$ [6, 8]
Min. Con. D. Set	$\mathrm{bw}(G)^{\mathrm{bw}(G)}$ [89]	$\mathrm{tw}(G)^{\mathrm{tw}(G)}$ [84]	$\mathrm{pw}(G)^{\mathrm{pw}(G)}$ [84]

Here $\omega \leq 2.376$ is the best known matrix multiplication exponent.

For unit disk graphs, it is usually better to consider the following variation of treewidth, which might be interesting in its own right.

Definition 5.1.4 *A* relaxed tree decomposition (T, X) *is a tree* T *and a collection of* bags $X_t \subseteq V(G)$*, one for each vertex* $t \in V(T)$*, satisfying:*

(i). $\bigcup_{t \in V(T)} X_t = V(G)$*,*

(ii). for all $(u, v) \in E(G)$*, there is a vertex* $t \in V(T)$ *such that* $u, v \in X_t$ *or there is an edge* $(t, t') \in E(T)$ *such that* $u \in X_t$ *and* $v \in X_{t'}$*, and*

(iii). $X_t \cap X_{t''} \subseteq X_{t'}$ *for any* $t, t'' \in V(T)$ *and for any* $t' \in V(T)$ *on the* t–t'' *path in* T*.*

The width *of a relaxed tree decomposition is* $\max_{t \in V(T)}\{|X_t|\}$*. The* relaxed treewidth $\mathrm{rtw}(G)$ *of a graph* G *is the minimum width of any relaxed tree decomposition of* G*.*

If the tree of a relaxed tree decomposition is in fact a path, we speak of a *relaxed path decomposition*. The *relaxed pathwidth* $\mathrm{rpw}(G)$ of a graph G is the minimum width of any relaxed path decomposition of G.

The difference between treewidth and relaxed treewidth lies in constraint *(ii)* of the underlying decomposition and the absence or presence of the -1 term in the definition of width (note that $\mathrm{rtw}(T) = \mathrm{tw}(T) = 1$ for any tree T). The treewidth and the relaxed treewidth of a graph are related though.

Proposition 5.1.5 *For any graph* G*,* $\frac{1}{2}(\mathrm{tw}(G) + 1) \leq \mathrm{rtw}(G) \leq \mathrm{tw}(G) + 1$ *and* $\frac{1}{2}(\mathrm{pw}(G) + 1) \leq \mathrm{rpw}(G) \leq \mathrm{pw}(G) + 1$*. Moreover, these bounds are tight.*

Proof: Let (T, X) be a relaxed tree decomposition of a graph G. For each edge $(t, t') \in E(T)$, replace it by a two-edge path with new vertex tt' and let $X_{tt'} = X_t \cup X_{t'}$. This yields a tree decomposition of G of width at most twice the width of (T, X) minus one. This bound is tight, as $\mathrm{rtw}(K_n) = \lceil n/2 \rceil$ and $\mathrm{tw}(K_n) = n - 1$ for any $n > 0$.

Any tree decomposition of width k is also a relaxed tree decomposition of width $k + 1$. This bound is tight, as exhibited by the following graph. Let $k \geq 2$ and let $U = \{u_1, \ldots, u_{k+1}\}$ and $W = \{w_1, \ldots, w_{k+1}\}$ be disjoint sets of $k + 1$ vertices each. Add edges to make U a clique and for each $w_i \in W$ add edges to all vertices $u_j \in U \backslash \{u_i\}$. That is, for each $i \in \{1, \ldots, k + 1\}$, (u_i, w_i) is a nonedge. Call the resulting graph G.

Clearly, $\mathrm{tw}(G) = k$ and $\mathrm{rtw}(G) \leq k + 1$. Suppose for sake of contradiction that G has a relaxed tree decomposition (T, X) of width at most k. Since $|U| = k + 1$, no bag can contain all vertices of U. As U is a clique, there must be bags X_t and $X_{t'}$ such that $(t, t') \in E(T)$ and $U \subseteq X_t \cup X_{t'}$. Moreover, there must be vertices $u_i, u_j \in U$ such that $u_i \in X_t - X_{t'}$ and $u_j \in X_{t'} - X_t$. This implies that $W - \{w_i, w_j\} \subseteq X_t \cup X_{t'}$. But then $|X_t \cup X_{t'}| \geq 2k$ and thus $X_t \cap X_{t'} = \emptyset$. Assume w.l.o.g. that $|X_t \cap U| \leq |X_{t'} \cap U|$. Then $w_j \in X_t \cup X_{t'}$ and thus $|X_t \cup X_{t'}| \geq 2k + 1$, a contradiction.

The same arguments hold for (relaxed) path decompositions. □

The definition of relaxed tree- and pathwidth is reminiscent of the definition of strong treewidth by Seese [239].

Definition 5.1.6 *A* strong tree decomposition (T, X) *is a relaxed tree decomposition where the bags are pairwise disjoint. The* strong treewidth $\mathrm{stw}(G)$ *of a graph* G *is the minimum width over all strong tree decompositions of* G.

The definitions of the notions of *strong path decomposition* and *strong pathwidth* $\mathrm{spw}(G)$ easily follow.

Proposition 5.1.7 *On any graph* G, $\mathrm{rtw}(G) \leq \mathrm{stw}(G)$, $\mathrm{tw}(G) \leq 2{\cdot}\mathrm{stw}(G){-}1$, $\mathrm{rpw}(G) \leq \mathrm{spw}(G)$, *and* $\mathrm{pw}(G) \leq 2 \cdot \mathrm{spw}(G) - 1$.

Proof: The bound $\mathrm{rtw}(G) \leq \mathrm{stw}(G)$ is immediate from Definition 5.1.6. The bound $\mathrm{tw}(G) \leq 2{\cdot}\mathrm{stw}(G){-}1$ follows from Proposition 5.1.5 and was also proved by Seese [239]. The inequalities for strong pathwidth follow similarly. □

It is not clear yet what relation exists between the strong treewidth of a graph and its (relaxed) treewidth, although a large gap might exist.

Proposition 5.1.8 *Let* G *be a wheel graph with* $k(k+2)$ *spokes for* $k \geq 2$. *Then* $\mathrm{tw}(G) = \mathrm{rtw}(G) = 3$, *but* $\mathrm{stw}(G) = k + 1$.

Proof: One can easily show that $\mathrm{tw}(G), \mathrm{rtw}(G) \leq 3$. As G has a K_4-minor, $\mathrm{tw}(G) \geq 3$. A cycle has relaxed treewidth 2 and from this $\mathrm{rtw}(G) \geq 3$.

To prove that $\mathrm{stw}(G) = k + 1$, consider any strong tree decomposition (T, X) of G. Let X_h be the bag containing the hub of G. Any other vertex of G must be either in X_h or a bag X_t with $(h, t) \in E(T)$. Consider any edge $(u, v) \in E(G)$ that is not a spoke and for which there is no bag containing both u and v. Then w.l.o.g. $u \in X_h$ and $v \in X_t$ for $t \neq h$. Call (u, v) a *crossing edge* of t. Clearly, any bag $t \neq h$ has at least two (unique) crossing edges. Moreover, any vertex in X_h can be incident with at most two crossing edges. As the hub is not incident with any crossing edge, $|V(T)| \leq |X_h|$. But then the width of (T, X) is greater than k by simple counting. A strong tree decomposition with k bags of $k+1$ vertices each and a central bag X_h with k vertices and the hub has width $k + 1$. Hence $\mathrm{stw}(G) = k + 1$. □

Finally, we give an important property of relaxed tree decompositions, similar to a result by Kloks [168, p. 149] for tree decompositions.

Lemma 5.1.9 *Let* (T, X) *be a relaxed tree decomposition of width* w *of a graph* G. *Then there exists a relaxed tree decomposition* (T', X') *of width* w *of* G *such that any vertex in* T' *has degree at most* 3.

Proof: If $t \in V(T)$ has degree larger than three, split the set of neighbors of t into two sets (N_1 and N_2) of cardinality at least two. Remove t from T and replace it by two vertices t', t'', connected by an edge. Let $X_{t'} = X_{t''} = X_t$. Now connect all vertices in N_1 to t' and all vertices in N_2 to t''. Clearly the result is a relaxed tree decomposition of G. Iteratively apply the above splitting process to obtain the requested relaxed tree decomposition. □

5.2 Thickness

The reason for studying strong and relaxed path and tree decompositions is that there is a natural way to bound the width of such decompositions on unit disk graphs. We capture the width in a geometric notion, called thickness. Below we define the thickness of a unit disk graph and give a relation to strong and relaxed pathwidth.

Assume that we are given a unit disk graph G with a known representation $\mathcal{D} = \{\mathcal{D}(v) = (c_v, r_v) \mid v \in V(G)\}$, where $c_v \in \mathbb{R}^2$ is the center of the disk corresponding to vertex v and $r_v = 1/2$ is its radius.

The thickness of a unit disk graph is determined by a slab decomposition of a representation of that graph. Given an angle α ($0 \leq \alpha < \pi$) and a point $p \in \mathbb{R}^2$, partition the plane using an infinite set of parallel lines (called *bars*) such that the distance between any two consecutive lines is precisely 1, the lines are parallel and intersect the x-axis at angle α, and (exactly) one line goes through p. The area within distance $1/2$ of a bar is called a *slab*. A disk is said to be *in* a slab if its center is contained in the interior or lies on the left boundary of the slab. The parallel lines thus induce a partition of $\mathcal{D}$, which we call a *slab decomposition* (α, p) of $\mathcal{D}$. This in turn induces a decomposition of $V(G)$ into pairwise disjoint, but collectively exhaustive subsets $Y_1, \ldots, Y_k$ such that Y_j contains the vertices corresponding to the disks contained in the j-th nonempty slab from the left.

Definition 5.2.1 *The* thickness *of a slab decomposition (α, p) of $\mathcal{D}$ is the maximum number of disks of $\mathcal{D}$ in any slab of the decomposition, i.e. it is* $\max_{1 \leq j \leq k}\{|Y_j|\}$.

For any fixed angle α ($0 \leq \alpha < \pi$), the *min-thickness* $t^*_\alpha(\mathcal{D})$ is the minimum thickness of any slab decomposition (α, p) over all $p \in \mathbb{R}^2$. Similarly, the *max-thickness* $\bar{t}_\alpha(\mathcal{D})$ is the maximum thickness of any slab decomposition (α, p) over all $p \in \mathbb{R}^2$.

Definition 5.2.2 *The* thickness $t^*(\mathcal{D})$ *is the minimum min-thickness $t^*_\alpha(\mathcal{D})$ over all angles α ($0 \leq \alpha < \pi$). The* minimax thickness $\bar{t}(\mathcal{D})$ *is the minimum max-thickness $\bar{t}_\alpha(\mathcal{D})$ over all angles α ($0 \leq \alpha < \pi$).*

The thickness and the minimax thickness can both be computed in polynomial time by exhaustively enumerating all relevant angles and points [258].

There is an easy relation between the thickness of $\mathcal{D}$ and its minimax thickness, given by the following proposition.

Proposition 5.2.3 *$t^*(\mathcal{D}) \leq \bar{t}(\mathcal{D}) \leq 2 \cdot t^*(\mathcal{D})$. Moreover these bounds are tight.*

Proof: The inequality $t^*(\mathcal{D}) \leq \bar{t}(\mathcal{D})$ holds by definition. For any angle α, a slab of one slab decomposition with angle α can overlap at most two of any other. Hence $\bar{t}_\alpha(\mathcal{D}) \leq 2 \cdot t^*_\alpha(\mathcal{D})$ for any angle α ($0 \leq \alpha < \pi$) and thus $\bar{t}(\mathcal{D}) \leq 2 \cdot t^*(\mathcal{D})$. The tightness of these inequalities is demonstrated by respectively two disjoint disks and two disks with intersecting interiors. □

The definition of minimax thickness already hints at a relation to the pathwidth of a unit disk graph.

Theorem 5.2.4 *The pathwidth of a unit disk graph with representation $\mathcal{D}$ is at most $\bar{t}(\mathcal{D}) - 1$.*

Proof: Let α be an angle such that $\bar{t}_\alpha(\mathcal{D}) = \bar{t}(\mathcal{D})$. Let B be the set of all straight lines parallel to a line intersecting the x-axis at angle α. For any $b \in B$, define X_b as the set of disks intersecting b. Let B' be any minimal subset of B such that $\{X_b \mid b \in B'\} = \{X_b \mid b \in B\}$. By definition, $\max_{b \in B'}\{|X_b|\} \leq \bar{t}_\alpha(\mathcal{D})$. Moreover, by ordering the bars in B' from left to right, the sets X_b with $b \in B'$ induce a path decomposition. □

This implies that the relaxed pathwidth is at most $\bar{t}(\mathcal{D})$. One can however improve on this bound by considering the strong pathwidth of a unit disk graph instead of its pathwidth.

Theorem 5.2.5 *The strong pathwidth of a unit disk graph with representation $\mathcal{D}$ is at most $t^*(\mathcal{D})$.*

Proof: Consider any slab decomposition of thickness $t^*(\mathcal{D})$ and let $Y_1, \ldots, Y_k$ be the induced partition of $V(G)$. Since the slabs have width 1 and two unit disks intersect if and only if the distance between their centers is at most 1, $Y_1, \ldots, Y_k$ is a strong path decomposition of G of width $t^*(\mathcal{D})$. □

Corollary 5.2.6 *The relaxed pathwidth of a unit disk graph with representation $\mathcal{D}$ is at most $t^*(\mathcal{D})$.*

Observe that following Proposition 5.2.3 this bound is always better, possibly by as much as a factor of 2, then the bound that followed from Theorem 5.2.4. Hence it makes sense to consider algorithms for relaxed path decompositions, as we might attain better worst-case running times than when using the path decomposition given by Theorem 5.2.4.

5.3 Algorithms on Strong, Relaxed Tree Decompositions

We have shown how to transform an optimum slab decomposition into a strong and a relaxed path decomposition. However, as far as we know, no algorithms exist for problems like Maximum Independent Set and Minimum (Connected) Dominating Set that make use of such a decomposition. We develop these algorithms in this section and prove that we can match or improve on the running times attained for path decompositions. In fact, the given algorithms apply to relaxed tree decompositions and thus can also be applied in a more general setting.

5.3.1 Maximum Independent Set and Minimum Vertex Cover

The idea behind the algorithm is similar to the idea behind the algorithm on tree decompositions by Telle and Proskurowski [249]. Let G be any graph and (T, X) a relaxed tree decomposition of G. Fix a vertex $r \in V(T)$ and root the tree T at r. Define a function size as follows. If $u \in V(T)$ has no children, then for any independent set $A_u \subseteq X_u$,

$$\mathsf{size}_u(A_u) = 0.$$

For all $u \in V(T)$ with children $u_1, \ldots, u_\rho$, let for any independent set $A_u \subseteq X_u$,

$$\mathsf{size}_u(A_u) = \sum_{i=1}^{\rho} \max_{A_{u_i}} \Big\{ |A_{u_i}| + \mathsf{size}_{u_i}((A_u \cup A_{u_i}) \cap X_{u_i}) \Big\},$$

where the maximum is over all independent sets $A_{u_i} \subseteq X_{u_i} - X_u - N(A_u)$.

For any $u \in V(T)$, let T_u denote the subtree of T rooted at u.

Lemma 5.3.1 *The cardinality of a maximum independent set of graph G is $\max_{A_r}\{\mathsf{size}_r(A_r)+|A_r|\}$. The maximum is over all independent sets $A_r \subseteq X_r$.*

Proof: We claim that for any $u \in V(T)$ and for any $A_u \subseteq X_u$, $\mathsf{size}_u(A_u)$ is the maximum cardinality of any set $I \subseteq \bigcup_{t \in V(T_u)\setminus\{u\}} X_t - X_u$ for which $I \cup A_u$ is an independent set, or $-\infty$ if no such set exists.

We prove this by induction. If u has no children, then $\bigcup_{t \in V(T_u)\setminus\{u\}} X_t = \emptyset$ and the claim is immediate from the definition of size_u. So suppose that u has children $u_1, \ldots, u_\rho$ and that the claim holds for $u_1, \ldots, u_\rho$. For any $i \in \{1, \ldots, \rho\}$ and any independent set $A_u \subseteq X_u$, it follows by induction that

$$\max_{A_{u_i}} \Big\{ |A_{u_i}| + \mathsf{size}_{u_i}((A_u \cup A_{u_i}) \cap X_{u_i}) \Big\},$$

where the maximum is over all independent sets $A_{u_i} \subseteq X_{u_i} - X_u - N(A_u)$, is the maximum cardinality of any set $I_i \subseteq \bigcup_{t \in V(T_{u_i})} X_t - X_u$ for which $I_i \cup A_u$ is an independent set, or $-\infty$ is no such set exists. Furthermore, $\bigcup_{t \in V(T_{u_i})} X_t - X_u$ and $\bigcup_{t \in V(T_{u_j})} X_t - X_u$ for any $1 \leq i < j \leq \rho$ are not connected. Hence $\mathsf{size}_u(A_u)$ is indeed as claimed.

The lemma is immediate from the proof of the claim. □

It is also easy to compute (the cardinality of) a maximum independent set using size. Let w be the width of (T, X), let $M = \max_{(t,t') \in E(T)} |X_t \cup X_{t'}|$, and let $n = |V(G)|$.

Theorem 5.3.2 *One can compute (the cardinality of) a maximum independent set of G in $\mathrm{O}(nw\,2^M)$ time.*

Proof: This follows from the definition of size and from Lemma 5.3.1. □

If (T, X) is a strong or a relaxed tree decomposition, then M is at most twice the width of the decomposition. If (T, X) is an ordinary tree decomposition of width w, then we can assume that $M \leq w + 1$ [168, p. 149].

Corollary 5.3.3 *One can compute (the cardinality of) a maximum independent set of G in $\mathrm{O}(nw\,2^{2\cdot\mathrm{stw}(G)})$, $\mathrm{O}(nw\,2^{2\cdot\mathrm{rtw}(G)})$, or $\mathrm{O}(nw\,2^{\mathrm{tw}(G)})$ time, assuming the appropriate graph decomposition is given.*

Recall that $\mathrm{rtw}(G) \leq \mathrm{tw}(G) \leq 2 \cdot \mathrm{rtw}(G) - 1$. For Maximum Independent Set, it is therefore more beneficial to have a tree decomposition.

The same holds for Minimum Vertex Cover. It is well-known that I is an independent set of a graph G if and only if $V(G) - I$ is a vertex cover of G.

Theorem 5.3.4 *One can compute (the cardinality of) a minimum vertex cover of G in $\mathrm{O}(nw\,2^{2\cdot\mathrm{stw}(G)})$, $\mathrm{O}(nw\,2^{2\cdot\mathrm{rtw}(G)})$, or $\mathrm{O}(nw\,2^{\mathrm{tw}(G)})$ time, assuming the appropriate graph decomposition is given.*

5.3.2 Minimum Dominating Set

We give two algorithms for Minimum Dominating Set. First, we present a simple algorithm using a strong path decomposition. It has a running time of $\mathrm{O}(nw\,2^{3\cdot\mathrm{spw}(G)})$ and is crucial to the approximation schemes of Chapter 6. We then consider a more complex algorithm, which can use any relaxed tree decomposition as input and has running time $\mathrm{O}(nw\,2^{2\cdot\mathrm{rtw}(G)})$ or $\mathrm{O}(nw\,3^{\mathrm{pw}(G)})$.

A Simple Algorithm

Let G be any graph and (T, X) a strong path decomposition of G. We can view the bags as being in sequence and number them accordingly $X_1, \ldots, X_p$. Observe that for any i, the vertices in X_i can only be dominated by vertices in X_{i-1} (if $i > 1$), X_i, or X_{i+1} (if $i < p$). This idea can be exploited as follows. Define for any $A_1 \subseteq X_1$ and $A_2 \subseteq X_2$,

$$\mathsf{size}_1(A_1, A_2) = \begin{cases} 0 & \text{if } A_1 \cup A_2 \text{ dominates } X_1 \\ \infty & \text{otherwise.} \end{cases}$$

Define for any $i = 2, \ldots, p-1$, any $A_i \subseteq X_i$, and any $A_{i+1} \subseteq X_{i+1}$,

$$\begin{aligned} \mathsf{size}_i(A_i, A_{i+1}) \quad = \quad & \min\{|A_{i-1}| + \mathsf{size}_{i-1}(A_{i-1}, A_i) \mid \\ & A_{i-1} \subseteq X_{i-1} \text{ and } A_{i-1} \cup A_i \cup A_{i+1} \text{ dominates } X_i\}. \end{aligned}$$

Moreover, for any $A_p \subseteq X_p$,

$$\begin{aligned} \mathsf{size}_p(A_p) \quad = \quad & \min\{|A_{p-1}| + \mathsf{size}_{p-1}(A_{p-1}, A_p) \mid \\ & A_{p-1} \subseteq X_{p-1} \text{ and } A_{p-1} \cup A_p \text{ dominates } X_p\}. \end{aligned}$$

Lemma 5.3.5 *The cardinality of a minimum dominating set of graph G is $\min_{A_p}\{\mathsf{size}_p(A_p) + |A_p|\}$, where the minimum is over all $A_p \subseteq X_p$.*

Proof: We claim that for any $i \in \{1, \ldots, p-1\}$, any $A_i \subseteq X_i$, and any $A_{i+1} \subseteq X_{i+1}$, $\mathsf{size}_i(A_i, A_{i+1})$ is the cardinality of a smallest set $R \subseteq \bigcup_{j=1}^{i-1} X_j$ such that $R \cup A_i \cup A_{i+1}$ dominates $\bigcup_{j=1}^{i} X_j$, or ∞ if no such set exists. Applying induction, this follows immediately from the definition of size. Hence for any $A_p \subseteq X_p$, $\mathsf{size}_p(A_p)$ is the cardinality of a smallest set $R \subseteq \bigcup_{j=1}^{p-1} X_j$ such that $R \cup A_p$ dominates $\bigcup_{j=1}^{p} X_j$, or ∞ if no such set exists. As G has a dominating set, $\min_{A_p}\{\mathsf{size}_p(A_p) + |A_p|\}$ is the cardinality of a minimum dominating set. □

Theorem 5.3.6 *One can compute (the cardinality of) a minimum dominating set of G in $\mathrm{O}(nw\,2^{3w})$ time, given a strong path decomposition of width w.*

Clearly this algorithm extends to strong tree decompositions, although this increases the running time of the algorithm. It can even be extended to relaxed tree decompositions at a higher cost. Here we chose to keep the algorithm and its description simple. We will only use it on a strong path decomposition. Moreover, a much faster algorithm exists on relaxed tree decompositions.

A Better Algorithm

We improve on the running time of the above algorithm by extending it to relaxed tree decompositions. We apply an idea similar to the one used in the algorithm by Alber et al. [6, 8].

Let G be any graph and (T, X) a relaxed tree decomposition of G. By Lemma 5.1.9, we may assume that any vertex of T has degree at most three. Fix a vertex $r \in V(T)$ which has degree at most one and root the tree T at r. Define a function size as follows. If $u \in V(T)$ has no children, then for any $A_u \subseteq X_u$ and any $B_u \subseteq X_u - A_u$,

$$\mathsf{size}_u(A_u, B_u) = \begin{cases} 0 & \text{if } B_u \subseteq N(A_u) \\ \infty & \text{otherwise.} \end{cases}$$

If $u \in V(T)$ has children $u_1, \ldots, u_\rho$, then for any $A_u \subseteq X_u$ and $B_u \subseteq X_u - A_u$,

$$\begin{aligned} &\mathsf{size}_u^i(A_u, B_u) \\ &\quad = \min\Big\{|A_{u_i}| + \mathsf{size}_{u_i}\Big((A_u \cup A_{u_i}) \cap X_{u_i},\ X_{u_i} - X_u - N[A_u \cup A_{u_i}]\Big) \ \Big|\ \\ &\qquad\qquad A_{u_i} \subseteq X_{u_i} - X_u, B_u \subseteq N(A_{u_i} \cup A_u)\Big\} \end{aligned}$$

$$\mathsf{size}_u(A_u, B_u) = \min_{(B_1, \ldots, B_\rho) \text{ partitioning } B_u} \left\{\sum_{i=1}^{\rho} \mathsf{size}_u^i(A_u, B_i)\right\}.$$

Lemma 5.3.7 *The cardinality of a minimum dominating set of graph G is $\min_{A_r \subseteq X_r}\{\mathsf{size}_r(A_r, X_r - A_r) + |A_r|\}$.*

Proof: It suffices to prove the following. We claim that for any $u \in V(T)$, any $A_u \subseteq X_u$, and any $B_u \subseteq X_u - A_u$, $\mathsf{size}_u(A_u, B_u)$ is the cardinality of a smallest set $R \subseteq \bigcup_{t \in V(T_u)} X_t - X_u$ such that $R \cup A_u$ dominates $B_u \cup \big(\bigcup_{t \in V(T_u)} X_t - X_u\big)$, or ∞ if no such set exists.

If $u \in V(T)$ has no children, then this is immediate from the definition of size. So suppose that $u \in V(T)$ has children $u_1, \ldots, u_\rho$ and the claim holds for each child. Let $A_u \subseteq X_u$, $B_u \subseteq X_u - A_u$. Then $\mathsf{size}^i_u(A_u, B_u)$ is the cardinality of a smallest set $R_i \subseteq \bigcup_{t \in V(T_{u_i})} X_t - X_u$ such that $R_i \cup A_u$ dominates $B_u \cup (\bigcup_{t \in V(T_{u_i})} X_t - X_u)$, or ∞ if no such set exists. Observe that $\bigcup_{t \in V(T_{u_i})} X_t - X_u$ and $\bigcup_{t \in V(T_{u_j})} X_t - X_u$ are disjoint for any $1 \leq i < j \leq \rho$. Hence using its definition, $\mathsf{size}_u(A_u, B_u)$ is indeed as claimed. □

A trivial way to compute the function size would be to follow its definition. However, the running time would be $\mathrm{O}(nw\,(2^w 3^w + 5^w))$, where w is the width of the decomposition. One can improve on this naive analysis to get a better worst-case running time. Let again $M = \max_{(t,t') \in E(T)} |X_t \cup X_{t'}|$.

Theorem 5.3.8 *One can compute (the cardinality of) a minimum dominating set of graph G in $\mathrm{O}(nw\,(2^M + 4^w))$ time.*

Proof: If $u \in V(T)$ has no children, then size_u is computable in $\mathrm{O}(w3^w)$ time.

Consider some $u \in V(T)$ with children $u_1, \ldots, u_\rho$. We compute size^i_u in two phases. First, enumerate all $A_u \subseteq X_u$ and $A_{u_i} \subseteq X_{u_i} - X_u$. Set $B_u = N(A_u \cup A_{u_i}) \cap X_u$. If this set B_u was not encountered before or if $|A_{u_i}| + \mathsf{size}_{u_i}(\ldots)$ is smaller than the previous value of $\mathsf{size}^i_u(A_u, B_u)$, then set

$$\mathsf{size}^i_u(A_u, B_u) = |A_{u_i}| + \mathsf{size}_{u_i}((A_u \cup A_{u_i}) \cap X_{u_i}, X_{u_i} - X_u - N[A_u \cup A_{u_i}]).$$

This takes $\mathrm{O}(w\rho\, 2^M)$ time.

There is no guarantee that we see all values $B_u \subseteq X_u - A_u$ for a given A_u. However, we do see all 'maximal' values. Note that the following inequality should hold: $\mathsf{size}^i_u(A_u, B'_u) \leq \mathsf{size}^i_u(A_u, B_u)$ for any $A_u \subseteq X_u$ and any $B'_u \subseteq B_u \subseteq X_u - A_u$. So enumerate all $A_u \subseteq X_u$ and $B_u \subseteq X_u - A_u$ with $B_u \neq \emptyset$ *in order*, meaning that if $B'_u \subseteq B''_u$, then B''_u is considered before B'_u. Now update size^i_u as follows. For any $x \in B_u$, let

$$\mathsf{size}^i_u(A_u, B_u \backslash \{x\}) = \min\{\mathsf{size}^i_u(A_u, B_u \backslash \{x\}), \mathsf{size}^i_u(A_u, B_u)\},$$

where we assume that $\mathsf{size}^i_u(A_u, B_u) = \infty$ or $\mathsf{size}^i_u(A_u, B_u \backslash \{x\}) = \infty$ if B_u or $B_u \backslash \{x\}$ respectively was not considered in the first step. This takes $\mathrm{O}(w\, 3^w)$ time. Moreover, we have now correctly computed size^i_u.

To compute size_u, recall that any vertex of T has degree at most three and that r is a leaf of T. Hence $\rho \leq 2$. Now by enumerating all $A_u \subseteq X_u$, $B_1 \subseteq X_u - A_u$, $B_2 \subseteq X_u - A_u - B_2$, and letting $B_u = B_1 \cup B_2$, it is easy to compute size_u. This takes $\mathrm{O}(4^w)$ time. The theorem follows. □

Observe that if a relaxed path decomposition of width w is given, then $\rho \leq 1$, and the described algorithm runs in $O(nw\,(2^M + 3^w))$ time. If a strong path decomposition of width w is given, this implies a running time of $O(nw\,2^{2w})$, which is a factor of 2^w faster than the previous algorithm.

Corollary 5.3.9 *One can compute (the cardinality of) a minimum dominating set of graph G in $O(nw\,2^{2\cdot \mathrm{rtw}(G)})$ or $O(nw\,3^{\mathrm{pw}(G)})$ time, assuming the appropriate graph decomposition is known.*

The running time for path decompositions follows from the fact that given a path decomposition of width w, one can assume that $M \leq w + 1$ [168].

Minimum Dominating Set is a clear case where having a relaxed tree decomposition is preferable to having an ordinary tree decomposition. As $\mathrm{rtw}(G) \leq \mathrm{tw}(G)$, and possibly $\mathrm{rtw}(G) = \frac{1}{2} \cdot \mathrm{tw}(G) + \frac{1}{2}$, the worst case running time of $O(nw\,2^{2\cdot \mathrm{rtw}(G)})$ is preferable to the $O(n\,2^{2\cdot \mathrm{tw}(G)})$ algorithm by Alber et al. [6, 8]. When considering path decompositions, we improve if $\mathrm{rpw}(G) \leq (\frac{1}{2}\log 3) \cdot \mathrm{pw}(G)$, where $\frac{1}{2}\log 3 \approx 0.792$.

5.3.3 Minimum Connected Dominating Set

We develop a technique for solving the minimum connected dominating set problem that augments both proposed algorithms for Minimum Dominating Set. Hence we effectively obtain two algorithms.

The technique is based on the following general ideas. Let G be a graph and (T, X) a relaxed tree decomposition of G. Root the tree T at some fixed vertex $r \in V(T)$. Given $X \subseteq V(G)$, a set $W \subseteq V(G)$ is *partially connected* (with respect to X) if each connected component of W intersects X. This definition is crucial because of the following easy fact. Let $u \in V(T)$ and let G_u be the graph induced by $\bigcup_{t \in V(T_u)} X_t$. Then for any connected dominating set $W \subseteq V(G)$, $W \cap V(G_u)$ is partially connected with respect to $W \cap X_u$ if $W \cap X_u \neq \emptyset$. This suggests that during the dynamic programming, we should construct partially connected dominating sets.

Suppose that $W \subseteq V(G_u)$ is partially connected with respect to $W \cap X_u$ for some $u \in V(T)\backslash\{r\}$. Then W induces an equivalence relation $\sim_{W,u}$ on the connected components of $W \cap X_u$, namely $C \sim_{W,u} C'$ for connected components C, C' of $W \cap X_u$ if and only if there is a C–C' path in $G[W]$. Let $(u, v) \in E(T)$ such that v is closer to r than u. Given a subset $A_u \subseteq X_u$, an equivalence relation $\sim$ on the connected components of A_u, and a subset $A_v \subseteq X_v - X_u$, we say that A_v is *compatible to* $(A_u, \sim)$ if for any equivalence class $\mathcal{C}$ of $\sim$, there is a connected component $C \in \mathcal{C}$ such that there is a C–A_v path in $G[C \cup A_v]$. Then given a set $W \subseteq V(G_u)$ that is partially connected with respect to $W \cap X_u$ and any $A_v \subseteq X_v - X_u$, $W \cup A_v$ is partially connected with respect to A_v if and only if A_v is compatible to $(W \cap X_u, \sim_{W,u})$. Note that given a compatible set A_v, $\sim_{W \cup A_v, v}$ is uniquely determined by $\sim_{W,u}$. We say that $\sim_{W,u}$ *determines* $\sim_{W \cup A_v, v}$.

We can now use the above ideas as follows. First, we adapt the simple algorithm given in Paragraph 5.3.2. Suppose that (T, X) is a strong path decomposition, i.e. the bags are numbered in sequence, $X_1, \ldots, X_p$. Define a function size as follows. Let $\sim_{\mathrm{id}}$ denote the identity equivalence relation, i.e. $C \sim_{\mathrm{id}} C'$ if and only if $C = C'$, and let $\sim_{\mathrm{all}}$ denote the total equivalence relation, i.e. $C \sim_{\mathrm{all}} C'$ for any C, C'. Then for any $A_1 \subseteq X_1$, any $A_2 \subseteq X_2$, and any equivalence relation $\sim$ on the connected components of $G[A_1]$ such that A_2 is compatible to $(A_1, \sim)$,

$$\mathsf{size}_1(A_1, A_2, \sim) = \begin{cases} 0 & \text{if } A_1 \cup A_2 \text{ dominates } X_i \text{ and } \sim \equiv \sim_{\mathrm{id}} \\ \infty & \text{otherwise.} \end{cases}$$

For any $i \in \{2, \ldots, p-1\}$, any $A_i \subseteq X_i$, any $A_{i+1} \subseteq X_{i+1}$, and any equivalence relation $\sim$ on the connected components of $G[A_i]$ such that A_{i+1} is compatible to $(A_i, \sim)$,

$$\begin{aligned} \mathsf{size}_i(A_i, A_{i+1}, \sim) \quad = \quad & \min \big\{ |A_{i-1}| + \mathsf{size}_{i-1}(A_{i-1}, A_i, \sim') \mid \\ & \qquad A_{i-1} \subseteq X_{i-1}, A_i \text{ compatible to } (A_{i-1}, \sim'), \\ & \qquad\quad A_{i-1} \cup A_i \cup A_{i+1} \text{ dominates } X_i, \\ & \qquad\qquad \text{and } \sim' \text{ determines } \sim \big\}. \end{aligned}$$

Finally, for any $A_p \subseteq X_p$,

$$\begin{aligned} \mathsf{size}_p(A_p) \quad = \quad & \min \big\{ |A_{p-1}| + \mathsf{size}_{p-1}(A_{p-1}, A_p, \sim') \mid \\ & \qquad A_{p-1} \subseteq X_{p-1}, A_p \text{ compatible to } (A_{p-1}, \sim'), \\ & \qquad\quad A_{p-1} \cup A_p \text{ dominates } X_p, \\ & \qquad\qquad \text{and } \sim' \text{ determines } \sim, \text{ where } \sim \equiv \sim_{\mathrm{all}} \big\}. \end{aligned}$$

For the analysis, we define $q(G, T, X)$ as the maximum number of equivalence relations one needs to consider during the algorithm for any subset $A_u \subseteq X_u$ for any $u \in V(T)$.

Lemma 5.3.10 *Given a set $\{c_0, \ldots, c_k\}$, there is a data structure for storing equivalence relations of this set that uses space the number of stored relations times $\mathrm{O}(k \log k)$ and where insert, update, and find cost $\mathrm{O}(k \log k)$ time.*

Proof: Observe that for any equivalence relation we can always assume that c_h is in the j-th equivalence class for some $j \leq h + 1$. Now consider the following tree. Level h of the tree corresponds to c_h. A node on level h has $h+1$ children, where the j-th child is used to describe that c_{h+1} is in equivalence class j. A path in the tree (from the root to a leaf) thus provides a complete description of an equivalence relation. One can use level $k+1$ to store any value associated with this equivalence relation.

To quickly find the j-th child of a node, we use a balanced binary search tree to store the children. It follows immediately from this description that insert, update, and find each cost $\mathrm{O}(k \log k)$ time. The tree has $k + 1$ levels and on each level, one has a search tree of height $\mathrm{O}(\log k)$ to navigate.

Of course, one does not store the entire tree, but keeps only those paths that correspond to a stored relation. Each path costs $O(k \log k)$ space to maintain. As each node uses a balanced binary search tree to store its children, insert, update, and find still take $O(k \log k)$ time. The lemma follows. □

Theorem 5.3.11 *One can compute (the cardinality of) a minimum connected dominating set of graph G in $O(n \cdot w \log w \cdot q(G,T,X) \cdot 2^{3w})$ time, given a strong path decomposition (T,X) of width w.*

Proof: It immediately follows from the proof of Lemma 5.3.5 that the set attaining $\min_{A_p \subseteq X_p}\{\mathsf{size}_p(A_p) + |A_p|\}$ is a dominating set of G. Moreover, the compatibility constraints in the definition of size ensure that this set is connected. Hence it is easily shown that $\min_{A_p \subseteq X_p}\{\mathsf{size}_p(A_p) + |A_p|\}$ is the cardinality of a minimum connected dominating set of G.

To compute size, we use the data structure described in the above lemma. We only maintain those equivalence relations $\sim$ which are actually realizable, that is, for which $\mathsf{size}_i(A_i, A_{i+1}, \sim) \neq \infty$. Hence for fixed $A_i \subseteq X_i$, $A_{i+1} \subseteq X_{i+1}$, we have a data structure requiring $O(w \log w \cdot q(G,T,X))$ space with $O(w \log w)$ time insert, update, and find operations.

It follows that size_1 can be computed in $O(w \log w \cdot 2^{2w})$ time. In computing size_i for $i > 1$, we deviate slightly from the recursive formula. For any $A_{i-1} \subseteq X_{i-1}$, $A_i \subseteq X_i$, and $A_{i+1} \subseteq X_{i+1}$ such that $A_{i-1} \cup A_i \cup A_{i+1}$ dominates X_i, we consider all relations $\sim'$ for which $\mathsf{size}_{i-1}(A_{i-1}, A_i, \sim')$ is stored. Then we find the equivalence relation $\sim$ determined by $\sim'$ and check whether A_{i+1} is compatible to $(A_i, \sim)$. We then store $\mathsf{size}_i(A_i, A_{i+1}, \sim) = |A_{i-1}| + \mathsf{size}_{i-1}(A_{i-1}, A_i, \sim')$ if $\sim$ was not encountered before, or update this value if necessary. One computes size_p similarly.

Observe that during this process, it is not necessary to explicitly enumerate all equivalence relations. The relevant relations may be obtained from the tables. Thus in the analysis of the running time, it suffices to bound the maximum number of equivalence relations, which is $q(G,T,X)$. Hence for any $i > 1$, we spend $O(w \log w \cdot q(G,T,X) \cdot 2^{3w})$ time. The theorem follows. □

In a similar manner, one can adapt the second algorithm of Section 5.3.2.

Theorem 5.3.12 *One can compute (the cardinality of) a minimum connected dominating set of graph G in $O(n \cdot w \log w \cdot q(G,T,X) \cdot (2^M + 4^w))$ time, given a relaxed tree decomposition (T,X) of width w.*

It remains to bound $q(G,T,X)$. For an arbitrary graph G and an arbitrary relaxed tree decomposition (T,X) of width w, $q(G,T,X)$ is bounded by the number of different equivalence relations on a set of cardinality w. In other words, it is bounded by the number of distinct partitions of a w-element set.

Definition 5.3.13 *Given a set S, $S_1, \ldots, S_p$ is a* partition *of S if $S_i \neq \emptyset$ for any $1 \leq i \leq p$, $S = \bigcup_{i=1}^{p} S_i$, and for any $1 \leq i < j \leq p$, $S_i \cap S_j = \emptyset$.*

The number of partitions of an w-element set is equal to the w-th Bell number, ϖ_w, named for E.T. Bell [29, 30]. We give an upper bound.

Proposition 5.3.14 *$\varpi_w \leq (\frac{w}{\ln w})^w$ for $w \geq 2$.*

Proof: Since $x \leq e^{x-1}$ for any $x \in \mathbb{R}$, we have that $ex \leq e^x$. Then for any k,

$$\begin{aligned} e \cdot \frac{k \ln w}{w} &\leq e^{\frac{k \ln w}{w}} && \Rightarrow \\ \left(\frac{ke \ln w}{w}\right)^w &\leq e^{k \ln w} && \Rightarrow \\ k^w &\leq \left(\frac{w}{e \ln w}\right)^w w^k. \end{aligned}$$

Hence following Dobiński's formula [88] (see also Wilf [271]),

$$\varpi_w = \frac{1}{e} \sum_{k=0}^{\infty} \frac{k^w}{k!} \leq \left(\frac{w}{e \ln w}\right)^w \sum_{k=0}^{\infty} \frac{w^k}{k!} = \left(\frac{w}{e \ln w}\right)^w e^w = \left(\frac{w}{\ln w}\right)^w.$$

The bound follows. □

Corollary 5.3.15 *One can compute (the cardinality of) a minimum connected dominating set of graph G in $\mathrm{O}(n \cdot w \log w \cdot (\frac{w}{\ln w})^w 2^{3w})$ time, given a strong path decomposition (T, X) of width w.*

Corollary 5.3.16 *One can compute (the cardinality of) a minimum connected dominating set of graph G in $\mathrm{O}(n \cdot w \log w \cdot (\frac{w}{\ln w})^w (2^M + 4^w))$ time, given a relaxed tree decomposition (T, X) of width w.*

It can be easily verified that all algorithms in this section also apply to the weighted case of the problems.

5.4 Unit Disk Graphs of Bounded Thickness

Recall from Theorem 5.2.5 that the strong pathwidth of a unit disk graph with representation $\mathcal{D}$ is at most $t^*(\mathcal{D})$. The results in the previous section yield the following theorem.

Theorem 5.4.1 *Let G be a unit disk graph with representation $\mathcal{D}$ and let $t = t^*(\mathcal{D})$. Then Maximum Independent Set, Minimum Vertex Cover, and Minimum Dominating Set can be solved in $\mathrm{O}(nt\,2^{2t})$ time. Minimum Connected Dominating Set can be solved in $\mathrm{O}(n \cdot t \log t \cdot (\frac{t}{\ln t})^t 2^{2t})$ time.*

The theorem implies that if the thickness is bounded, say by a constant, then these problems can be solved in polynomial time.

One can improve on the worst-case analysis of the algorithm for Minimum Connected Dominating Set by using that the given strong path decomposition

comes from a strip decomposition. So let (T, X) be such a strong path decomposition. Instead of bounding $q(G, T, X)$ by ϖ_w, we will show that many relations are in fact the same. Hence we can use so-called noncrossing partitions to bound $q(G, T, X)$ instead of general partitions.

Definition 5.4.2 *Let S be a finite set and $\preceq$ a total order of its elements. Then $S_1, \ldots, S_p$ is a* noncrossing partition *of S if $S_1, \ldots, S_p$ is a partition of S and for any $1 \leq i, j \leq p$ with $i \neq j$, any $a, c \in S_i$, and any $b, d \in S_j$, $a \preceq b \preceq c \preceq d$ is false.*

Noncrossing partitions were first considered by Becker [27, 28]. Refer to Simion [240] for numerous applications of such partitions. By Becker [28] and Kreweras [183], the number of noncrossing partitions of a w-element set is C_w, the w-th *Catalan number*. It is well-known that $C_w \approx 4^w$. If the cardinality of a set can be bounded by a polynomial in C_w, it is said to have *Catalan structure*. Catalan structure was considered before in the context of Minimum Connected Dominating Set, for example on planar graphs [84]. These ideas were subsequently generalized by Dorn, Fomin, and Thilikos [90, 91] to more general graph classes (see also Section 6.5). This however is the first application to unit disk graphs.

Let G be a unit disk graph with representation $\mathcal{D}$ and (T, X) a strong path decomposition induced by a strip decomposition through Theorem 5.2.5. The bags are numbered in sequence $X_1, \ldots, X_p$. We exhibit the existence of Catalan structure. We start with an easy bound. Without loss of generality, assume that no two disk centers have the same y-coordinate. Then the following is a total order. For any two vertices $u, v \in V(G)$, let $u \preceq v$ if and only if $u = v$ or the disk center of $\mathcal{D}(u)$ lies below the disk center of $\mathcal{D}(v)$.

Lemma 5.4.3 *For any $1 \leq i \leq p$, let $W \subseteq \bigcup_{j=1}^{i} X_j$ be partially connected with respect to $A_i := W \cap X_i$ and for any $u, v \in A_i$, $u \sim v$ if and only if $u = v$ or there is a u–v path P_{uv} in W such that $P_{uv} \cap A_i = \{u, v\}$. If $a \sim c$ and $b \sim d$ for distinct $a, b, c, d \in A_i$ such that $a \preceq b \preceq c \preceq d$, then a, b, c, and d are connected in W.*

Proof: As $a \preceq b \preceq c \preceq d$, P_{ac} and P_{bd} must cross, meaning there are adjacent vertices $w, x \in P_{ac}$ and adjacent vertices $y, z \in P_{bd}$ such that the line segment $\overline{c_w c_x}$ intersects the line segment $\overline{c_y c_z}$. As G is a unit disk graph, both segments have length at most 1 and c_w is within distance 1 of c_y or c_z, or c_x is within distance 1 of c_y or c_z. Thus at least one of the edges (w, y), (w, z), (x, y), or (x, z) must be in $E(G)$. Hence a, b, c, and d are connected in W. □

Observe that a relation $\sim$ on the vertices of A_i as in the above lemma induces an equivalence relation $\sim_{W,i}$ on the connected components of $W \cap X_i$ as needed for the algorithm for Minimum Connected Dominating Set. In fact, one can construct a surjective map from equivalence relations $\sim$ to $\sim_{W,i}$. Hence if we bound the number of equivalence relations $\sim$, we have bounded the number of equivalence relations $\sim_{W,i}$.

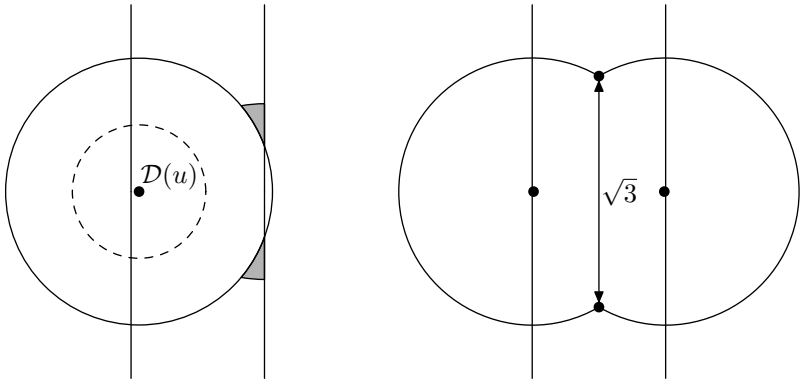

Figure 5.1: In the left figure, the disk $\mathcal{D}(u)$ is drawn dashed. The solid circle indicates the area in which the centers of $\mathcal{D}(v)$ and $\mathcal{D}(w)$ cannot lie, or they would intersect $\mathcal{D}(u)$. Then the centers of $\mathcal{D}(v)$ and $\mathcal{D}(w)$ have to be within the shaded areas. In the right figure, the area in which the centers of $\mathcal{D}(a)$ and $\mathcal{D}(b)$ cannot lie is indicated.

Theorem 5.4.4 *One can compute (the cardinality of) a minimum connected dominating set of a unit disk graph G in $O(n \cdot t \log t \cdot 2^{5t})$ time, where $t = t^*(D)$ is the thickness of a representation $\mathcal{D}$ of G.*

Proof: From Lemma 5.4.3, it follows that $\sim$ induces a noncrossing partition on the vertices of A_i. Hence we can bound $q(G, T, X)$ by C_t. The theorem now follows from Theorem 5.3.11. □

Using Theorem 5.3.12, we can give the following slightly better result.

Theorem 5.4.5 *One can compute (the cardinality of) a minimum connected dominating set of a unit disk graph G in $O(n \cdot t \log t \cdot 2^{4t})$ time, where $t = t^*(D)$ is the thickness of a representation $\mathcal{D}$ of G.*

It is possible to improve on this analysis if the number of connected components of A_i is smaller than $|A_i|$. Instead of applying noncrossing partitions to the vertices, we apply them to (parts of) connected components.

By rotating if necessary, we may assume that the slab boundaries of the underlying slab decomposition are parallel to the y-axis. Let $i \in V(T)$. Define the *y-range* $\bar{y}(S)$ of a connected subset $\emptyset \neq S \subseteq X_i$ as the range between the smallest and the largest y-coordinate of any disk center of S. The *contour* of S is the perimeter of the union of the disks of S. Consider the connected components of $G[A_i]$ for some $A_i \subseteq X_i$. Two connected components are said to *interact* if their y-ranges intersect.

Proposition 5.4.6 *No three connected components can pairwise interact.*

Proof: If three connected components C, C', and C'' pairwise interact, their y-ranges pairwise intersect and thus the y-ranges have a common y-coordinate. Without loss of generality, this common coordinate corresponds to the y-coordinate $y(u)$ of a disk center for some $u \in C$. Then $y(u) \in \bar{y}(C')$ and $y(u) \in \bar{y}(C'')$. Consider C' and let v be the vertex in C' such that $y(v)$ is minimal under the condition that $y(v) \geq y(u)$. Similarly, let $w \in V(C')$ such that $y(w)$ is maximal, but $y(w) \leq y(u)$. Define $a, b \in C''$ in a similar way. By symmetry, we may assume that the center of $\mathcal{D}(u)$ lies on or to the left of the middle of the slab. Then the centers of $\mathcal{D}(v)$ and $\mathcal{D}(w)$ lie in the shaded area of Figure 5.1. In the best case, u lies on the left boundary of the slab and $y(v) = y(u) + \epsilon$, $y(w) = y(u) - \epsilon$ for infinitesimally small $\epsilon > 0$. But then the centers of $\mathcal{D}(a)$ and $\mathcal{D}(b)$ must be at distance at least $\sqrt{3} > 1$, which contradicts that $\mathcal{D}(a)$ and $\mathcal{D}(b)$ intersect. □

Corollary 5.4.7 *If $G[A_i]$ has c connected components, then at most $c - 1$ pairs of connected components interact.*

Proof: Consider the (interval) graph H induced by the y-ranges of the connected components of $G[A_i]$. By Proposition 5.4.6, no three intervals of H pairwise intersect. Hence H is a forest, where each vertex corresponds to a connected component and each edge to an interaction. Since H is a forest, the number of edges of H, and thus the number of pairwise interacting connected components is at most $c - 1$. □

This corollary will prove to be useful in counting arguments later on.

Following the proof of Proposition 5.4.6 and as the contours of no two connected components intersect, we can say of two interacting connected components C and C' that C is *in front* of C', if (when their y-ranges overlap) the contour of C lies to the left of the contour of C'.

Suppose that two connected components C and C' interact and C is in front of C'. Now split the vertices of C' into three parts: those vertices u for which $y(u) > \bar{y}(C)$, for which $y(u) \in \bar{y}(C)$, and for which $y(u) < \bar{y}(C)$. Iteratively decompose the connected components this way. What remains are *blocks* of vertices. Let $\mathcal{B}$ denote the set of all blocks. By construction, for any two distinct blocks $B, B' \in \mathcal{B}$, either $\bar{y}(B) \cap \bar{y}(B') = \emptyset$, $\bar{y}(B) \subseteq \bar{y}(B')$, or $\bar{y}(B) \supseteq \bar{y}(B')$. Note that the vertices of a block are connected. If $\bar{y}(B) \subseteq \bar{y}(B')$ for two blocks B and B', then B is *occluded* by B'. We may also call B' the *occluder* of B.

Consider the following total order $\preceq$ on the blocks. For $B, B' \in \mathcal{B}$, let $B \preceq B'$ if and only if $B = B'$, $\bar{y}(B) < \bar{y}(B')$, or $\bar{y}(B) \subseteq \bar{y}(B')$. Let $W \subseteq \bigcup_{j=1}^{i} X_j$ be such that W is partially connected with respect to $A_i := W \cap X_i$. Define $\sim$ on $\mathcal{B}$ such that $B \sim B'$ if and only if $B = B'$ or there is a B–B' path in $(W - A_i) \cup B \cup B'$. We claim that $\sim$ induces a noncrossing partition on $\mathcal{B}$ with respect to $\preceq$.

Lemma 5.4.8 *Consider distinct $B_a, B_b, B_c, B_d \in \mathcal{B}$ such that $B_a \sim B_c$, $B_b \sim B_d$, and $B_a \preceq B_b \preceq B_c \preceq B_d$. Then $B_a \sim B_b$.*

Proof: We aim to show that $\bar{y}(B_a) < \bar{y}(B_b) < \bar{y}(B_c) < \bar{y}(B_d)$. The lemma then follows from Lemma 5.4.3. Let (B, B') be an arbitrary pair of (B_a, B_b), (B_b, B_c), (B_c, B_d). By assumption, $B \preceq B'$ and $B \not\sim B'$. We show that $\bar{y}(B) < \bar{y}(B')$. Using the definition of $\preceq$ and since $B \neq B'$, it suffices to prove that assuming $\bar{y}(B) \subseteq \bar{y}(B')$ leads to a contradiction.

So suppose that $\bar{y}(B) \subseteq \bar{y}(B')$. By the construction of the blocks, this implies that B' occludes B. But then $N(B) \cap X_{i-1} \subseteq N(B') \cap X_{i-1}$. Hence any path in W to or from B must contain a vertex not in B neighboring a vertex in B'. As there is such a path, $B \sim B'$, a contradiction.

It follows that $\bar{y}(B_a) < \bar{y}(B_b) < \bar{y}(B_c) < \bar{y}(B_d)$ and thus that $B_a \sim B_b$ by using Lemma 5.4.3. □

Observe that one can construct a surjective map from relations $\sim$ on the blocks to relations $\sim_{W,i}$ on connected components of A_i. Hence it suffices to bound the number of relations $\sim$ on the blocks.

Theorem 5.4.9 *One can compute (the cardinality of) a minimum connected dominating set of a unit disk graph G with representation $\mathcal{D}$ in $\mathrm{O}(n \cdot c \log c \cdot 2^{3c} 2^{3t})$ time, where $t = t^*(\mathcal{D})$ and c is the largest number of connected components in any $A_i \subseteq X_i$ in the strong path decomposition (T, X) corresponding to the slab decomposition supporting $t^*(\mathcal{D})$.*

Proof: It follows from Corollary 5.4.7 that the number of blocks is at most $3(c-1)$. Lemma 5.4.8 showed that $\sim$ induces a noncrossing partition of $\mathcal{B}$ with respect to $\preceq$. Hence the number of relations $\sim$ is at most $C_{|\mathcal{B}|} \leq C_{3(c-1)}$ and we can bound $q(G, T, X)$ by $C_{3(c-1)}$. The theorem immediately follows from Theorem 5.3.11. □

Because $c \leq t$, the running time of the above algorithm is $\mathrm{O}(n \cdot t \log t \cdot 2^{6t})$ in the worst case, which is worse than the running time guaranteed by Theorem 5.4.4. We will see in the next chapter however that there are cases where $c \ll t$.

The results developed above also apply to the algorithm of Theorem 5.3.12 and we thus improve on the running time of Theorem 5.4.9 as follows.

Theorem 5.4.10 *One can compute (the cardinality of) a minimum connected dominating set of a unit disk graph G with representation $\mathcal{D}$ in $\mathrm{O}(n \cdot c \log c \cdot 2^{3c} \cdot 2^{2t})$ time, where $t = t^*(\mathcal{D})$ and c is the largest number of connected components in any $A_i \subseteq X_i$ in the strong path decomposition (T, X) corresponding to the slab decomposition supporting $t^*(\mathcal{D})$.*

This gives a worst-case running time of $\mathrm{O}(n \cdot t \log t \cdot 2^{5t})$.