Approximation Algorithms for Unit Disk Graphs

Erik Jan van Leeuwen

institute of information and computing sciences, utrecht university

technical report UU-CS-2004-066

www.cs.uu.nl

Approximation Algorithms for Unit Disk Graphs[*]

Erik Jan van Leeuwen

Institute for Information and Computing Sciences, Utrecht University,
Padualaan 14, 3584 CH Utrecht, the Netherlands.
E-mail: `erikjan@cs.uu.nl`

Abstract

Mobile ad hoc networks are frequently modeled by unit disk graphs. We consider several classical graph theoretic problems on unit disk graphs (Maximum Independent Set, Minimum Vertex Cover, and Minimum (Connected) Dominating Set), which are relevant to such networks.

We propose two new notions for unit disk graphs: thickness and density. The thickness of a graph is the number of disk centers in any width 1 slab. If the thickness of a graph is bounded, then the considered problems can be solved in polynomial time. We prove this both indirectly by presenting a relation between unit disk graphs of bounded thickness and the pathwidth of such graphs, and directly by giving dynamic programming algorithms. This result implies that the problems are fixed-parameter tractable in the thickness.

We then consider unit disk graphs of bounded density. The density of a graph is the number of disk centers in any 1-by-1 box. We present a new approximation scheme for the considered problems, which uses the bounded thickness results mentioned above and the so called shifting technique. We show that the scheme is an asymptotic FPTAS and that this result is optimal, in the sense that no FPTAS can exist (unless P=NP). The scheme for Minimum Connected Dominating Set is the first FPTAS$^\infty$ for this problem. The analysis that is applied can also be used to improve existing results, which among others implies the existence of an FPTAS$^\infty$ for MCDS on planar graphs.

1 Introduction

Mobile ad hoc networks are the next generation in communication networks. Devices can enter or leave the network anytime, devices can be mobile, and no centralized control points or base stations are necessary. This flexibility makes mobile ad hoc networks very interesting for the consumer and military market, but also makes them more complex than many existing wireless networks (such as GSM). The distinct properties of mobile ad hoc networks have given rise to various new problems and challenges. These can be very practical (such as message routing) and theoretical.

This paper focusses on the more theoretical aspects of mobile ad hoc networks and considers a graph model for such networks. As will be shown, a mobile ad hoc network can be naturally modeled as a so called *(unit) disk graph*. Each node in such a graph has a disk around it containing all points reachable by that node. The intersections of these disks then

[*]This research was partially supported by the Netherlands Organisation for Scientific Research NWO (project *Treewidth and Combinatorical Optimisation*).

determine the edges of the graph. As these graphs have a nice geometric interpretation, we may hope that classical graph theoretic problems that are relevant for mobile ad hoc networks (such as Maximum Independent Set and Minimum (Connected) Dominating Set) are easier to solve or approximate than they are on general graphs. We show that by making some realistic assumptions about this geometric interpretation, various new properties and algorithms can be found, which improve on known results.

1.1 Preliminaries

Unit disk graphs are a special kind of (geometric) intersection graphs. Hence, we first define these graphs.

Definition 1.1 *Let S be a set of geometric objects. Then the graph $G = (V, E)$, where each vertex corresponds to an object in S and two vertices are connected by an edge if and only if the two corresponding objects intersect, is called an* intersection graph*. The graph G is said to be* realized by *S.*

In this definition, tangent objects are assumed to intersect. We can now formally define (unit) disk graphs. Denote by $c_i \in \mathbb{R}^2$ the center and by r_i the radius of a disk D_i.

Definition 1.2 *A graph G is a* disk graph *if and only if there exists a set of disks $D = \{D_i \,|\, i = 1, \ldots, n\}$, such that G is the intersection graph of D. The set of disks is called a* disk representation *of G.*

A disk graph can be given without its disk representation. In this case, it is assumed each vertex knows the vertices adjacent to it. However, knowing a disk representation can help to more efficiently solve problems on disk graphs.

Definition 1.3 *A graph G is a* unit disk graph *if and only if G is a disk graph and the radii of a set of disks realizing G are equal.*

Usually the common radius is 1, but often it is assumed to be $\frac{1}{2}$. Note that any common radius can be obtained by scaling D appropriately.

We use the following notation.

Notation 1.4 *Let $G = (V, E)$ be a disk graph with disk representation $D = \{D_i \,|\, i = 1, \ldots, n\}$. For a vertex $v \in V$, we denote the corresponding disk in D by D_v, its center by c_v, and its radius by r_v. For a disk center c_i of some disk in D, we denote the vertex $v \in V$ corresponding to that disk by $v(c_i)$.*

Observe that (unit) disk graphs are a good model for mobile ad hoc networks. Each node of the network corresponds to a disk center and the transmission range of a node corresponds to the radius of the disk. In a unit disk graph, all nodes are assumed to have the same transmission range.

We also define the various approximation scheme types considered in this paper.

Definition 1.5 *Let P be a maximization (minimization) problem. Then an algorithm A is a*

- polynomial-time approximation scheme (PTAS) *for P if and only if for any instance x of P and for any (fixed) $\epsilon > 0$, $A(x, \epsilon)$ runs in time polynomial in $|x|$ and delivers a feasible solution with value $SOL_{x,\epsilon}$, such that $SOL_{x,\epsilon} \geq (1 - \epsilon)OPT_x$ ($SOL_{x,\epsilon} \leq (1 + \epsilon)OPT_x$).*

- fully polynomial-time approximation scheme (FPTAS) *for P if and only if for any instance x of P and for any $\epsilon > 0$, $A(x, \epsilon)$ runs in time polynomial in $|x|$ and $\frac{1}{\epsilon}$ and delivers a feasible solution with value $SOL_{x,\epsilon}$, such that $SOL_{x,\epsilon} \geq (1 - \epsilon)OPT_x$ ($SOL_{x,\epsilon} \leq (1 + \epsilon)OPT_x$).*
- asymptotic fully polynomial-time approximation scheme (FPTAS$^\infty$) *for P if and only if for any instance x of P and for any $\epsilon > 0$, $A(x, \epsilon)$ runs in time polynomial in $|x|$ and $\frac{1}{\epsilon}$ and delivers a feasible solution with value $SOL_{x,\epsilon}$. If $|x| > c_\epsilon$, then also $SOL_{x,\epsilon} \geq (1 - \epsilon)OPT_x$ ($SOL_{x,\epsilon} \leq (1 + \epsilon)OPT_x$).*

Finally, we define fixed-parameter tractability, following Downey and Fellows [19].

Definition 1.6 *Let $\langle x, k\rangle$ be an instance of a parameterized problem P, with parameter $k \in \mathbb{N}$. Then P is* fixed-parameter tractable (FPT) *if and only if there exists an algorithm that delivers a feasible solution for $\langle x, k\rangle$ in running time $O(f(k)\,poly(|x|))$ for all x, where f is an arbitrary function and $poly(|x|)$ is an arbitrary polynomial in $|x|$.*

Note that a fixed-parameter tractable problem can be solved in polynomial time, given any fixed k.

1.2 Problem definitions

We consider various classical optimization problems on graphs, relevant to (unit) disk graph models of mobile ad hoc networks.

Definition 1.7 *Let $G = (V, E)$ be a graph. A set $S \subseteq V$ is an* independent set *if and only if there are no $u, v \in S$, such that $(u, v) \in E$. A set $S \subseteq V$ is a* vertex cover *if and only if for each $(u, v) \in E$ it holds that $u \in S$ or $v \in S$.*

Observe that an independent set is the complement of a vertex cover (and vice versa) [22]. Furthermore, we are usually looking for a maximum independent set and a minimum vertex cover. An independent set is maximum if and only if there is no independent set of greater size. A similar definition holds for minimum vertex cover.

In the context of mobile ad hoc networks, an independent set of a (unit) disk graph can be seen as a set of nodes that can transmit simultaneously without signal interferences. Vertex covers are mostly interesting from a theoretical point of view.

Definition 1.8 *Let $G = (V, E)$ be a graph. A set $S \subseteq V$ is a* dominating set *if and only if for each vertex v either $v \in S$ or there exists a vertex $u \in S$ for which $(u, v) \in E$.*

Definition 1.9 *Let $G = (V, E)$ be a graph. A set $S \subseteq V$ is a* connected dominating set *if and only if S is a dominating set and the subgraph of G induced by S ($G[S] = (S, (S \times S) \cap E)$) is connected.*

A dominating set in a mobile ad hoc network can be seen as a set of emergency transmitters capable of reaching every node in the network, or as central nodes in node clusters. A connected dominating set can be used as a backbone for easier and faster communications. The problem is to find a minimum (connected) dominating set.

1.3 Previous work

All problems mentioned above are NP-hard for general graphs (see Garey and Johnson [22]). Since (unit) disk graphs are a restricted class of graphs with a nice geometric interpretation, one might hope that these problems are better solvable. Unfortunately, Clark, Colbourn, and Johnson [16] proved these problems to be NP-hard on (unit) disk graphs as well. Therefore, most research has focussed on approximation algorithms.

For all considered problems on unit disk graphs, simple constant factor approximation algorithms exist. Minimum Independent Set for instance can be approximated within a factor of 5 using a simple greedy algorithm [34]. This algorithm simply chooses an arbitrary vertex to put in the independent set, removes that vertex and its neighbors from the graph and then repeats. If the leftmost vertex (i.e. the vertex corresponding to the leftmost disk center) is chosen instead, the algorithm improves to a 3-approximation algorithm [34]. On general disk graphs, choosing the vertex with the smallest disk radius results in a 5-approximation algorithm [34]. Marathe *et al.* [34] have a detailed description of these algorithms. They also propose constant factor approximation algorithms for Minimum Vertex Cover and Minimum (Connected) Dominating Set on unit disk graphs. Malesińska [33] gives an approximation algorithm for Minimum Vertex Cover on general disk graphs.

Agarwal and Mustafa [2] provide a more general approach and consider the intersection graph of a set S of convex 2D objects. If κ is the size of the maximum independent set of this graph, their algorithm returns an independent set of size $(\kappa/(2\log(2n/\kappa)))^{\frac{1}{3}}$ in $O(n^3+\tau(S))$ time, where $\tau(S)$ is the time necessary to compute the left- and rightmost point of each object and test which objects intersect. Clearly, a set of disks is a set of convex 2D objects and hence their result also holds for (unit) disk graphs.

Several polynomial time approximation schemes exist as well [14, 21, 26, 35, 37, 38]. Most of these schemes have in common that they use the so called *shifting technique*. This is a general technique, independently discovered by Baker [6] and Hochbaum and Maass [25]. The basic idea is the following. A set of regularly spaced separators is used to decompose the problem into smaller, easier solvable subproblems. The solutions of the subproblems are merged to form a solution to the global problem. This is repeated for several placements of the separator set. The best solution over these placements is then selected as an approximation of the optimum. Moving the separator set can be regarded as shifting the set through the problem. Hence the name 'shifting technique'.

Since its discovery, the shifting technique has been used to solve various problems [1, 4, 21, 23, 26, 27, 30]. In the context of (unit) disk graphs, Matsui [35] and Hunt *et al.* [26] presented the first PTAS's using the shifting technique. Matsui gives a PTAS for Maximum Independent Set on unit disk graphs. Hunt *et al.* [26] propose a different PTAS for Maximum Independent Set and use similar ideas to construct a PTAS for Minimum Vertex Cover and Minimum Dominating Set on unit disk graphs. Erlebach, Jansen, and Seidel [21] extend these ideas to give a PTAS for Maximum Independent Set and Minimum Vertex Cover on general disk graphs. Chan [14] presents a PTAS for Maximum Independent Set on the intersection graph of a set of fat objects. Under the used definition, a set of disks is fat. Hence the presented scheme is a PTAS for Maximum Independent Set on disk graphs.

Nieberg, Hurink, and Kern [37, 38] give a robust PTAS for Maximum Independent Set and Nieberg and Hurink [39] a robust PTAS for Minimum Dominating Set on unit disk graphs for which no disk representation is given. A robust algorithm on unit disk graphs solves the problem correctly for every unit disk graph. For graphs that are not unit disk graphs, the algorithm may either produce the correct output for the problem, or provide a certificate that the input is not a unit disk graph.

The relation between the fixed-parameter tractability of some problems and the possibility of creating PTASs and FPTAS$^{\infty}$s exploiting this has been used before, most notably by Baker [6] for planar graphs. Demaine and Hajiaghayi [18] look at a more general class of graphs, namely minor-closed graphs of locally bounded treewidth. Hunt *et al.* [26] consider λ-precision unit disk graphs in which the distance between any two disk centers is at least λ. We will discuss these results in more detail in Section 4.

Finally, we consider results for the Minimum Connected Dominating Set. Marathe *et al.* [34] present a 10-approximation algorithm. A significant amount of work has been done on distributed approximation algorithms for minimum connected dominating sets, because of their practical use as a communication backbone. Wan, Alzoubi, and Frieder [43] present an $O(n \log n)$ message and $O(n)$ time complexity algorithm with approximation factor 8. This is later improved by Cardei *et al.* [13] to $O(n\, deg)$ message complexity, where deg is the maximum degree of the graph. Butenko *et al.* [12] demonstrate a heuristic which outperforms most approximation algorithms in practice. Cheng *et al.* [15] gave the first known PTAS for Minimum Connected Dominating Set on unit disk graphs. Demaine and Hajiaghayi [18] use the same proof-technique to give a PTAS on planar graphs and an almost-PTAS on minor closed graphs of locally bounded treewidth.

1.4 Organization

This paper is organized as follows. In Section 2, we propose a new notion for unit disk graphs, called thickness. The thickness of a unit disk graph is the maximum number of disk centers in a width 1 slab. Given a decomposition of the graph into slabs of bounded thickness, we show that Maximum Independent Set, Minimum Dominating Set, and Minimum Connected Dominating Set can be solved optimally in polynomial time. We prove this both directly by using a slab decomposition, and indirectly by demonstrating a relation between unit disk graphs of bounded thickness and the pathwidth of such graphs.

In Section 3, these techniques are applied in a new approximation algorithm for the studied optimization problems. We again introduce a new notion for unit disk graphs, called density, which is the maximum number of disk centers in a 1-by-1 grid square. We propose a new approximation scheme for unit disk graphs of bounded density, which improves on existing algorithms. Then we provide a discussion section to see how these schemes relate to existing results.

2 Thickness

Throughout this paper, we assume we are given a unit disk graph $G = (V, E)$ with a known disk representation $D = \{(c_i, r_i) \mid i = 1, \ldots, n\}$. Furthermore, all disks in D are assumed to have radius $\frac{1}{2}$.

The thickness of a unit disk graph[1] is determined by a disk representation of that graph. To define the thickness of a representation, we first need the notion of a slab decomposition.

[1]The notion we use differs from existing notions of thickness in graph theory, such as described by Harary [24] and Eppstein [20].

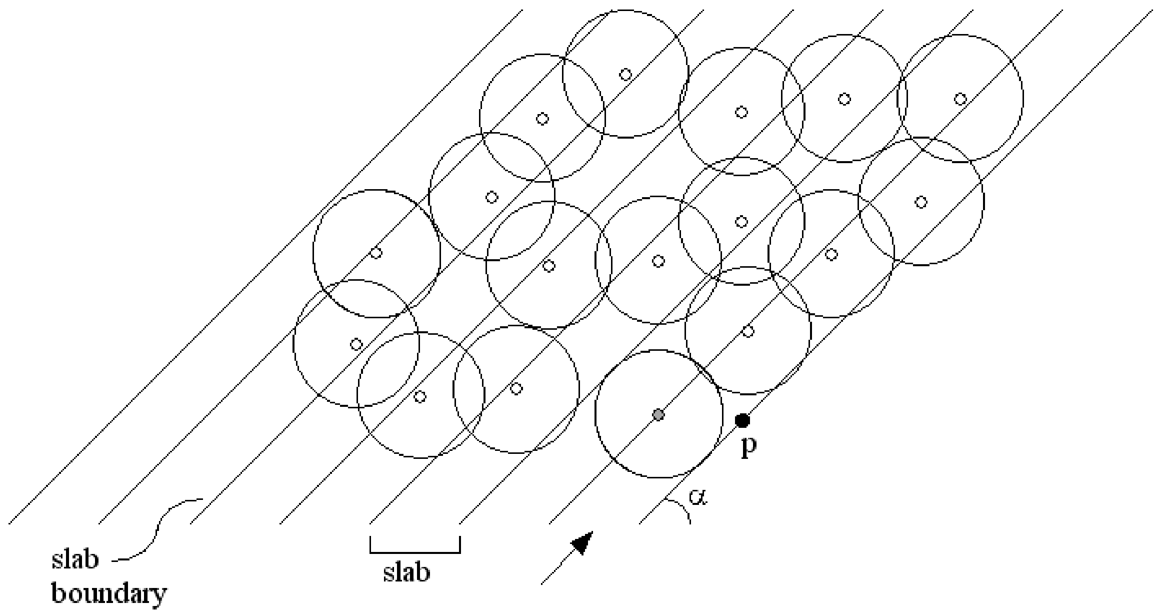

Figure 1: An arbitrary set of disks decomposed into slabs. The angle with the x-axis is α and one slab boundary goes through p. The grey disk center is on a slab boundary and hence in the right slab (marked with an arrow).

Definition 2.1 *Let $0 \leq \alpha < \pi$ be an angle and p an arbitrary point in the plane. Partition the plane using an infinite set of parallel lines, such that the distance between two neighboring lines is* 1*, each line intersects the x-axis at angle α, and (exactly) one line goes through p. The lines are called* slab boundaries*, and the area between any two neighboring slab boundaries is called a* slab*. The* width *of each slab is* 1*.*

Observe that the partitioning of the plane imposed by the slabs remains the same after a rotation of π around p. Hence $0 \leq \alpha < \pi$ is sufficient.

Definition 2.2 *Given a partition of the plane into slabs, a disk (c_i, r_i) is considered to be* in *a given slab if its center c_i is between the two slab boundaries defining the slab. If a center lies on a slab boundary, then the disk is considered to belong to the slab to the right of the slab boundary. The slabs decompose D into mutually exclusive, but collectively exhaustive subsets. This decomposition is called the* slab decomposition $s = \langle \alpha, p \rangle$ *of D.*

See Figure 1 for a visual explanation of this definition.

Definition 2.3 *A slab decomposition of a unit disk graph G is the slab decomposition of any disk representation D of G. Furthermore, let $Y_1, Y_2, \ldots, Y_b$ be subsets of V, such that Y_j contains those vertices corresponding to disk centers of D in the j-th non-empty slab of s.*

In other words, $Y_1, \ldots, Y_b$ contain the vertices of the $b \leq n$ non-empty slabs of s. For convenience, we assume there also exist three 'dummy' slabs Y_{-1}, Y_0, and Y_{b+1}, which are all empty. Observe that vertices in Y_j $(1 \leq j \leq b)$ can only have edges to vertices in Y_{j-1}, Y_j, and Y_{j+1}, as the disks have diameter 1 and slabs have width 1. Also note that $Y_1 \cup \cdots \cup Y_b = V$ and $Y_j \cap Y_k = \emptyset$ for all $1 \leq j, k \leq b$ $(j \neq k)$.

Using slab decompositions, we can define thickness.

Definition 2.4 *The* thickness $t(D, s)$ *of a slab decomposition s of D is the maximum number of disk centers of D in any slab of s. The* thickness $t(G, s)$ *of a slab decomposition s of a unit disk graph G with disk representation D is equal to $t(D, s)$.*

For example, the thickness of the slab decomposition shown in Figure 1 is 5.

In the above definitions, we have specified a slab decomposition by a tuple $\langle \alpha, p\rangle$, where p is an arbitrary point in the plane. We can however 'move' the decomposition to the right until there is at least one disk center that intersects a slab boundary. Clearly, this move does not affect the thickness of the decomposition. Hence, for our purposes, we can fully specify a slab decomposition by a tuple $\langle \alpha, c_s\rangle$, where c_s is the center of a disk in D.

Also note that given a slab decomposition $\langle \alpha, c_s\rangle$, we can easily obtain an equivalent decomposition in which the slab boundaries are perpendicular to the x-axis. To ensure the thickness remains the same, we apply the following transformation matrix to the centers of all disks.

$$T_\alpha = \mathsf{rotate}(\tfrac{1}{2}\pi - \alpha) = \begin{pmatrix} \cos(\frac{1}{2}\pi - \alpha) & -\sin(\frac{1}{2}\pi - \alpha) & 0 \\ \sin(\frac{1}{2}\pi - \alpha) & \cos(\frac{1}{2}\pi - \alpha) & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{1}$$

We now consider the set of all slab decompositions with some angle α, denoted by s^α. Using the specification proposed above, we derive that $s^\alpha = \{\langle \alpha, c_i\rangle \mid c_i \text{ is a disk center in } D\}$.

Definition 2.5 *The thickness t^α of D is defined as $t^\alpha(D) = \max_{s \in s^\alpha} t(D, s)$. Futhermore, let α_{min} be an angle such that for all angles α, $t^{\alpha_{min}}(D) \leq t^\alpha(D)$.*

The definition of the thickness t^α states that it is the maximum thickness of all slab decompositions having angle α. Then α_{min} is the angle for which this maximum thickness is minimal.

Definition 2.6 *Let $s^* = \langle \alpha^*, c_s^*\rangle$ be a slab decomposition of D such that for all (other) slab decompositions s of D, $t(D, s^*) \leq t(D, s)$. Let $t^* = t(D, s^*)$.*

This definition defines t^* as the *minimum thickness* of any slab decomposition of D. An interesting proposition regarding t^* and $t^{\alpha_{min}}$ is the following.

Proposition 2.7 $t^{\alpha_{min}} \leq 2t^*$

Proof: Consider $s^* = \langle \alpha^*, c_s^*\rangle$. For a slab decomposition $s \in s^{\alpha^*}$, any slab of s can overlap with at most 2 slabs of s^*. Hence $t(D, s) \leq 2t(D, s^*)$, and thus $t^{\alpha^*} \leq 2t^*$. The proposition now follows from the definition of $t^{\alpha_{min}}$. $\square$

The thickness of a slab decomposition of a graph can be computed in polynomial time. We simply rotate the slab decomposition and the disk centers such that the slab boundaries are perpendicular to the x-axis. Furthermore, we translate such that one slab boundary intersects the origin. Then it is straightforward to count the number of disk centers in each slab. The minimum thickness t^* and t^α can also be computed in polynomial time by exhaustively enumerating all relevant positions for a slab decomposition [29].

2.1 Relation to pathwidth

Given a slab decomposition of minimum thickness, we can prove an upper bound on the pathwidth of a unit disk graph. For completeness, we first define path decompositions and pathwidth.

Definition 2.8 (Robertson and Seymour [40]) *A* path decomposition *of a graph $G = (V, E)$ is a sequence $(X_1, X_2, \ldots, X_p)$ of subsets of V (called* bags*) such that*

1. $\bigcup_{1 \leq i \leq p} X_i = V$,
2. *for all $(v, w) \in E$, there is an i ($1 \leq i \leq p$) such that $v, w \in X_i$,*
3. *for all i, j, k with $1 \leq i < j < k \leq p$: $X_i \cap X_k \subseteq X_j$.*

The width *of a path decomposition $(X_1, X_2, \ldots, X_p)$ is $\max_{1 \leq i \leq p} |X_i| - 1$.*

Informally, we can rephrase the requirements of a path decomposition as follows: the bags collectively exhaust V, for each edge $e \in E$ there must be a bag that contains both endpoints of e, and for each vertex $v \in V$, if $v \in X_i$ and $v \in X_k$ for $i < k$, then v must also be in X_j for each j with $i < j < k$. Each graph $G = (V, E)$ trivially has at least one path decomposition, namely (V).

Definition 2.9 *The* pathwidth *of a graph $G = (V, E)$ is the minimum width of any path decomposition of G.*

Now let G again be a unit disk graph with disk representation D.

Lemma 2.10 *Given an angle α ($0 \leq \alpha < \pi$), there exists a path decomposition of G of width at most $2t^\alpha - 1$ and consisting of at most $2n$ bags. If no edge $(u, v) \in E$ exists for which $|T_\alpha(c_u)^x - T_\alpha(c_v)^x| = 1$, then there exists a path decomposition for G of width at most $t^\alpha - 1$ and consisting of at most $2n$ bags.*

Proof: Consider a slab of width 1 intersecting the x-axis at angle α. Position the slab such that it only contains the leftmost disk center c_l, i.e. $T_\alpha(c_l)^x$ is minimal (see Figure 2). Set $X_1 = \{c_l\}$ and $j = 2$.

Now 'slide' the slab to the right (i.e. in direction $\alpha - \frac{1}{2}\pi$) until a disk center c_i intersects the left or right boundary of the slab. If c_i intersects the right boundary, set $X_j = X_{j-1} \cup \{v(c_i)\}$ and $j \leftarrow j + 1$. If c_i intersects the left boundary, set $X_j = X_{j-1} \backslash \{v(c_i)\}$ and $j \leftarrow j + 1$. If disk centers intersect the right and left boundary simultaneously, we treat disk centers intersecting the right boundary first. This ensures that for edges of length 1, there will be an X_j containing both endpoints. We continue sliding the slab to the right until all vertices have been in the slab, ending with $j = p + 1$ for some p.

We claim that $(X_1, \ldots, X_p)$ is a path decomposition of G. Requirements 1 and 2 of Definition 2.8 are obviously satisfied. Because we slide the slab to the right, insert a vertex when it intersects the right boundary, and only remove it after it intersects the left boundary of the slab, the third requirement is also fulfilled. Hence $(X_1, \ldots, X_p)$ is a path decomposition of G.

The width of $(X_1, \ldots, X_p)$ is $\max_{1 \leq j \leq p} |X_j| - 1$. To determine the width, consider a position of the slab in which its left boundary intersects some disk center c_i. Then the size of

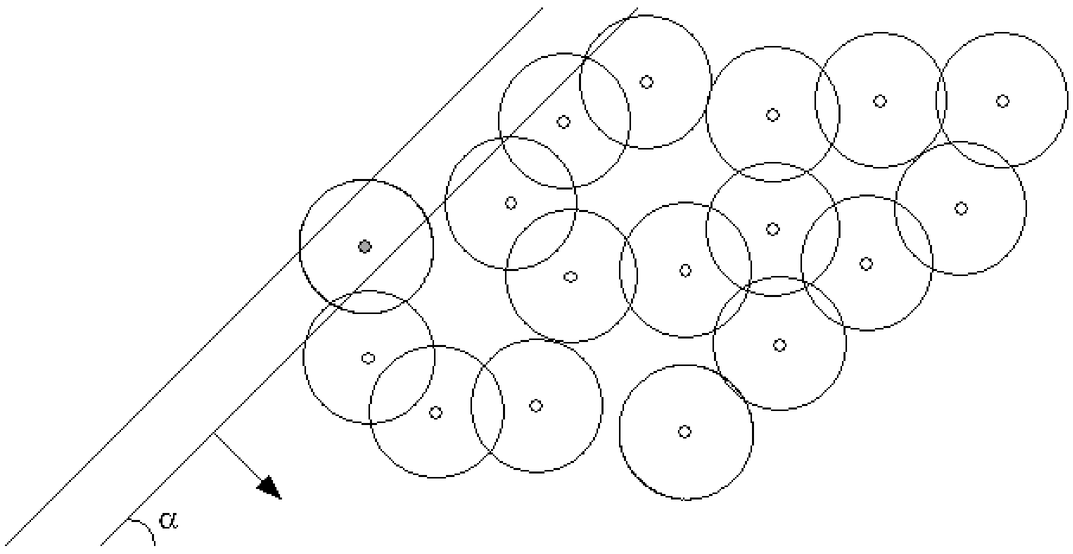

Figure 2: The grey disk center is the leftmost disk center. The slab is positioned such that it only contains this disk center. The arrow shows the direction in which to slide the slab.

the corresponding bag X_j at this position of the slab is at most $t(D, \langle\alpha, c_i\rangle)$ plus the number of disk centers intersecting the right boundary of the slab. Thus the size of X_j is at most $2t(D, \langle\alpha, c_i\rangle)$. Then

$$\max_{1\leq j\leq p} |X_j| - 1 \leq \max_{1\leq i\leq n} 2t(D, \langle\alpha, c_i\rangle) - 1 = 2t^\alpha - 1$$

Hence the width of $(X_1, \ldots, X_p)$ is at most $2t^\alpha - 1$. Because each vertex enters and leaves the bags once, the number of bags is at most $2n$.

Note that this analysis is tight. A trivial example in which the width of any path decomposition is at least $2t^\alpha - 1$ can be obtained by putting t^α disk centers on the left boundary of the slab, and t^α disk centers on the right boundary at distance 1 (see Figure 3). Clearly, such an example can indeed be constructed and has a clique containing $2t^\alpha$ vertices. Hence the pathwidth must be at least $2t^\alpha - 1$.

Now consider the case for which there exists no edge $(u, v) \in E$ for which $|T_\alpha(c_u)^x - T_\alpha(c_v)^x| = 1$. Observe that the example of Figure 3 cannot occur. Furthermore, during the construction of $X_1, \ldots, X_p$, we know that for any disk center intersecting the right boundary, the corresponding vertex cannot have an edge to a vertex corresponding to a disk center that intersects the left boundary. Therefore, while sliding the slab to the right, we can treat disk centers intersecting the left boundary before those intersecting the right boundary, without violating requirement 2 on path decompositions. It is easy to see that the resulting path decomposition now has width at most $t^\alpha - 1$ and consists of at most $2n$ bags. $\square$

A corollary of this lemma is that $2t^{\alpha_{min}} - 1$ is an upper bound on the pathwidth of G. If α_{min} is such that no edge $(u, v) \in E$ exists for which $|T_{\alpha_{min}}(c_u)^x - T_{\alpha_{min}}(c_v)^x| = 1$, then $t^{\alpha_{min}} - 1$ is an upper bound on the pathwidth of G.

Using the sliding slab technique proposed above, a path decomposition as indicated in Lemma 2.10 can be constructed in $O(n \log n)$ time [29].

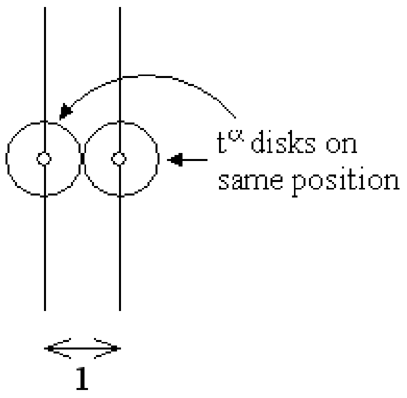

Figure 3: The two disk centers are repeated t^α times at the same position. This induces a clique of $2t^\alpha$ vertices. Hence the pathwidth of induced graph is at least $2t^\alpha - 1$.

2.2 Solving problems

We have shown how to transform an optimum slab decomposition into a path decomposition. A path decomposition can be used to solve many classical optimization problems on graphs. This includes, but is not limited to, the problems focussed on in this paper. If the pathwidth of a graph is pw, then Maximum Independent Set and Minimum Vertex Cover can be solved in $O(2^{pw}n)$ time, Minimum Dominating Set in $O(3^{pw}n)$ time, and Minimum Connected Dominating set in $O(pw^{pw}n)$ time [3, 11, 18, 42].

These algorithms, however, are designed for general graphs, and we are considering unit disk graphs. Therefore there might be more efficient algorithms. In this section, we show that for unit disk graphs, a slab decomposition can be used to solve the optimization problems directly, without first creating a path decomposition. This results in less complex and faster algorithms. To demonstrate the advantages of a slab decomposition of a unit disk graph, we show how to use it to solve Maximum Independent Set, Minimum Dominating Set and Minimum Connected Dominating Set optimally.

2.2.1 Maximum Independent Set

To solve optimization problems using a slab decomposition, we will use dynamic programming. Recall from Definition 2.3 that Y_j contains those vertices corresponding to disk centers of D in the j-th non-empty slab of s. For the maximum independent set problem, we can use the following 'subproblem optimality principle'.

Proposition 2.11 *For some j $(1 \le j \le b+1)$, let W_j be a subset of Y_j. If there exists a maximum independent set IS of $Y_0 \cup \cdots \cup Y_j$ such that $IS \cap Y_j = W_j$, then there exists a maximum independent set IS' of $Y_0 \cup \cdots \cup Y_{j-1}$ such that $IS' \cup W_j = IS$ and IS' is independent of W_j.*

Because a vertex in Y_j can only have edges to vertices in Y_{j-1}, Y_j, or Y_{j+1}, the requirement in this proposition that IS' is independent of W_j implies that $W_{j-1} = IS' \cap Y_{j-1}$ must be independent of W_j. This leads to Algorithm 2.1 for computing a maximum independent set of G.

```
1.   Set $size_0(\emptyset) = 0$ and $solution_0(\emptyset) = \emptyset$
2.   for  $j \leftarrow 1$ to $b+1$
3.   do   for each  $W_j \subseteq Y_j$
4.        do   $size_j(W_j) = -\infty$; $solution_j(W_j) = \emptyset$
5.             for each  $W_{j-1} \subseteq Y_{j-1}$
6.             do   if    $W_j \cup W_{j-1}$ is an independent set
7.                  then if    $|W_j| + size_{j-1}(W_{j-1}) > size_j(W_j)$
8.                       then  $size_j(W_j) \leftarrow |W_j| + size_{j-1}(W_{j-1})$
9.                             $solution_j(W_j) \leftarrow W_j \cup solution_{j-1}(W_{j-1})$
10.                      fi
11.                 fi
12.            od
13.       od
14.  od
15.  return $(size_{b+1}(\emptyset), solution_{b+1}(\emptyset))$
```

Algorithm 2.1: SlabDecompositionMIS$(G, Y_1, \dots, Y_b)$

Lemma 2.12 *Algorithm 2.1 computes a maximum independent set of a unit disk graph with thickness t in $O(t^2 2^{2t} n)$ time.*

Proof: We prove the lemma by induction on the value of $size_j$ and $solution_j$. The value of $size_j(W_j)$ will be the size of a maximum independent set IS of $Y_0 \cup \dots \cup Y_j$ such that $IS \cap Y_j = W_j$. This independent set is stored in $solution_j(W_j)$. If no such independent set exists, then we store $size_j(W_j) = -\infty$ and $solution_j(W_j) = \emptyset$. Note that this can occur if and only if W_j itself is not an independent set. Hence checking if $size_j(W_j) = -\infty$ is equivalent to checking whether W_j is an independent set.

Trivially, $size_0$ and $solution_0$ are correctly computed by the algorithm. Now inductively assume that for some $j \geq 1$, $size_{j-1}$ and $solution_{j-1}$ have been correctly computed by the algorithm. Consider the set Y_j and let W_j be an arbitrary subset of Y_j. As was already noted, vertices in Y_j can only have edges to vertices in Y_{j-1}, Y_j, or Y_{j+1}. So if $W_j \cup W_{j-1}$ is an independent set for some $W_{j-1} \subseteq Y_{j-1}$, then clearly $W_j \cup solution_{j-1}(W_{j-1})$ must be an independent set for $Y_0 \cup \dots \cup Y_j$. As the algorithm will remember the subset W_{j-1} of Y_{j-1} for which $W_j \cup W_{j-1}$ is an independent set and $|W_j| + size_{j-1}(W_{j-1})$ is maximum, by the subproblem optimality principle (Proposition 2.11), $size_j(W_j)$ and $solution_j(W_j)$ will be correct for j. Thus by induction, we have proved that $size_j$ and $solution_j$ are correctly computed.

It is easy to see that $size_{b+1}(\emptyset)$ is the size of the maximum independent set of G and that this set is stored in $solution_{b+1}(\emptyset)$. Hence the algorithm correctly computes a maximum independent set of G.

The time bound is easy to prove. Because $b \leq n$, the outer **for**-loop is executed at most $n+1$ times. For each of the at most 2^t subsets of Y_j and for each of the at most 2^t subsets of Y_{j-1}, the algorithm checks for independence. This can be done in $O(t^2)$ time, as each subset contains at most t vertices. Assuming table look-ups can be done in constant time, the resulting total running time of the algorithm is $O(t^2 2^{2t} n)$. □

Because the complement of a maximum independent set of a graph G is a minimum vertex cover of G, we immediately have the following corollary.

Corollary 2.13 *A minimum vertex cover of a unit disk graph with thickness t can be computed in $O(t^2 2^{2t} n)$ time.*

2.2.2 Minimum Dominating Set

For the minimum dominating set problem on unit disk graphs we can use a similar approach as in the previous section. For some set $W \subseteq V$, we denote the vertices in Y_j dominated by W as $D^j(W)$. Furthermore, given a dominating set DS, we denote DS restricted to slab j ($DS \cap Y_j$) by A_j, the vertices in slab j dominated by A_j or A_{j-1} ($D^j(A_j) \cup D^j(A_{j-1})$) by B_j, and the vertices dominated by A_{j+1} ($D^j(A_{j+1})$) by C_j. We will always ensure that A_j, B_j, and C_j are mutually exclusive, but collectively exhaust Y_j.

We will use the following subproblem optimality principle.

Proposition 2.14 *For some j ($1 \leq j \leq b+1$), let A_j, B_j be subsets of Y_j ($A_j \cap B_j = \emptyset$). If there exists a minimum dominating set $DS \subseteq Y_{-1} \cup \cdots \cup Y_j$ for $Y_{-1} \cup \cdots \cup Y_{j-1} \cup B_j$ such that $DS \cap Y_j = A_j$, then there exists a minimum dominating set $DS' \subseteq Y_{-1} \cup \cdots \cup Y_{j-1}$ with $A_{j-1} = DS' \cap Y_{j-1}$, such that $DS' \cup A_j = DS$, $D^j(A_{j-1}) \supseteq B_j - D^j(A_j)$, and DS' dominates $(Y_{-1} \cup \cdots \cup Y_{j-2}) \cup (Y_{j-1} - A_{j-1} - D^{j-1}(A_j))$.*

A straightforward implementation would use exhaustive enumeration on both A_j and B_j (and of course A_{j-1}). Then the running time would be approximately $O(t^2 2^{3t} n)$, which makes it slower than the algorithm developed in the previous paragraph. To improve on this naive algorithm, we observe that Proposition 2.14 states that $D^j(A_{j-1})$ can be a superset of $B_j - D^j(A_j)$. This observation can be exploited, such that we can use exhaustive enumeration on just A_j and A_{j-1}, followed by a post-processing step with exhaustive enumeration on A_j and B_j. This is shown in Algorithm 2.2.

Lemma 2.15 *Algorithm 2.2 computes a minimum dominating set of a unit disk graph with thickness t in $O(t^2 2^{2t} n)$ time.*

Proof: We prove this again by induction on the value of $size_j$ and $solution_j$. The value of $size_j(A_j, B_j)$ will be the size of a minimum dominating set $DS \subseteq Y_{-1} \cup \cdots \cup Y_j$ of $Y_{-1} \cup \cdots \cup Y_{j-1} \cup B_j$ such that $DS \cap Y_j = A_j$. This dominating set DS is stored in $solution_j(A_j, B_j)$. If no such dominating set exists, then we store $size_j(A_j, B_j) = \infty$ and $solution_j(A_j, B_j) = \emptyset$.

Trivially, $size_0$ and $solution_0$ are correctly computed by the algorithm. Now inductively assume that for some $j \geq 1$, $size_{j-1}$ and $solution_{j-1}$ have been correctly computed by the algorithm. Consider the set Y_j and let A_j be an arbitrary subset of Y_j. Let A_{j-1} be an arbitrary subset of Y_{j-1}. Then let $B_j = D^j(A_j) \cup D^j(A_{j-1}) - A_j$ and set $C_{j-1} = D^{j-1}(A_j) - A_{j-1}$. Now let $B_{j-1} = Y_{j-1} - A_{j-1} - C_{j-1}$ be the remaining vertices of slab $j-1$.

By induction, $size_{j-1}(A_{j-1}, B_{j-1})$ must be the size of a minimum dominating set $DS_{j-1} \subseteq Y_{-1} \cup \cdots \cup Y_{j-1}$ for $Y_{-1} \cup \cdots \cup Y_{j-2} \cup B_{j-1}$ with $Y_{j-1} \cap DS_{j-1} = A_{j-1}$, or it has value ∞. If $size_{j-1}(A_{j-1}, B_{j-1})$ has value ∞, then by definition no such dominating set exists, and we do not have to update any data structures. Otherwise, $solution_{j-1}(A_{j-1}, B_{j-1}) \cup A_j$ must be a dominating set for $Y_{-1} \cup \cdots \cup Y_{j-1} \cup B_j$. Hence if $size_{j-1}(A_{j-1}, B_{j-1}) + |A_j| < size_j(A_j, B_j)$,

```
1.  Set $size_0(\emptyset,\emptyset) = 0$ and $solution_0(\emptyset,\emptyset) = \emptyset$
2.  for $j \leftarrow 1$ to $b+1$
3.  do  for each $A_j \subseteq Y_j$
4.      do  for each $B_j \subseteq Y_j - A_j$
5.          do  $size_j(A_j, B_j) = \infty$
6.              $solution_j(A_j, B_j) = \emptyset$
7.          od
8.      od
9.      for each $A_j \subseteq Y_j$
10.     do  for each $A_{j-1} \subseteq Y_{j-1}$
11.         do  Let $B_j = D^j(A_j) \cup D^j(A_{j-1}) - A_j$,
                  $C_{j-1} = D^{j-1}(A_j) - A_{j-1}$, and
                  $B_{j-1} = Y_{j-1} - A_{j-1} - C_{j-1}$
12.             if    $size_{j-1}(A_{j-1}, B_{j-1}) \neq \infty$ and
                      $|A_j| + size_{j-1}(A_{j-1}, B_{j-1}) < size_j(A_j, B_j)$
13.             then  $size_j(A_j, B_j) = |A_j| + size_{j-1}(A_{j-1}, B_{j-1})$
14.                   $solution_j(A_j, B_j) = A_j \cup solution_{j-1}(A_{j-1}, B_{j-1})$
15.             fi
16.         od
17.     od
18.     for each $A_j \subseteq Y_j$
19.     do  for each $B_j \subseteq Y_j - A_j$ (in order of descending $|B_j|$)
20.         do  for each $v \in B_j$
21.             do  if    $size_j(A_j, B_j) < size_j(A_j, B_j \backslash \{v\})$
22.                 then  $size_j(A_j, B_j \backslash \{v\}) \leftarrow size_j(A_j, B_j)$
23.                       $solution_j(A_j, B_j \backslash \{v\}) \leftarrow solution_j(A_j, B_j)$
24.                 fi
25.             od
26.         od
27.     od
28. od
29. return $(size_{b+1}(\emptyset,\emptyset), solution_{b+1}(\emptyset,\emptyset))$
```

Algorithm 2.2: SlabDecompositionMDS$(G, Y_1, \ldots, Y_b)$

we found a (smaller) dominating set for $Y_{-1} \cup \cdots \cup Y_{j-1} \cup B_j$ and can update $size_j(A_j, B_j)$ and $solution_j(A_j, B_j)$. This ensures we minimize $size_j(A_j, B_j)$.

After we have finished the computation for all subsets A_{j-1} of Y_{j-1}, we have a lot of values of $size_j$ and $solution_j$. Now consider $size_j(A_j, B_j)$ for some $B_j \subseteq Y_j - A_j$. Using the observation we made earlier, we deduce that $D^j(solution_j(A_j, B_j))$ can be a superset of $B_j - D^j(A_j)$. In other words, it never hurts to dominate more of Y_j than just $B_j - D^j(A_j)$, as long as the size of the dominating set does not increase. This implies that for an arbitrary set $B'_j \subseteq Y_j - A_j$, $size_j(A_j, B'_j)$ should have a value if and only if there exists a set $B_j \supseteq B'_j$ for which a value for $size_j(A_j, B_j)$ has been computed.

Obviously, it would be rather inefficient to enumerate all possible supersets of each B'_j. Therefore we turn the argument around. If for some $B_j \subseteq Y_j - A_j$, $size_j(A_j, B_j)$ is an improvement over $size_j(A_j, B_j \backslash \{v\})$ for some $v \in B_j$, then we rather use $solution_j(A_j, B_j)$ to dominate $Y_{-1} \cup \cdots \cup Y_{j-1} \cup B_j \backslash \{v\}$. Hence we update $size_j(A_j, B_j \backslash \{v\})$ and $solution_j(A_j, B_j \backslash \{v\})$. To ensure the correctness of the computation, we process the B_j's in order of descending $|B_j|$.

According to the subproblem optimality principle (Proposition 2.14), this must result in (the size of) a minimum dominating set $DS \subseteq Y_{-1} \cup \cdots \cup Y_j$ of $Y_{-1} \cup \cdots \cup Y_{j-1} \cup B_j$ such that $DS \cap Y_j = A_j$. This holds for any $A_j \subseteq Y_j$. Thus by induction, we have proved that $size_j$ and $solution_j$ are correctly computed.

It is easy to see that $size_{b+1}(\emptyset, \emptyset)$ is the size of a minimum dominating set of G and that this set is stored in $solution_{b+1}(\emptyset, \emptyset)$. Hence the algorithm correctly computes a minimum dominating set of G.

The time bound is easy to prove. Because $b \leq n$, the outer **for**-loop is executed at most $n+1$ times. For each of the at most 2^t subsets A_j of Y_j and for each of the at most 2^t subsets A_{j-1} of Y_{j-1}, the algorithm computes the sets of Y_j and Y_{j-1} dominated by A_j and A_{j-1}. This can be done in $O(t^2)$ time. Hence the first phase of the algorithm costs $O(t^2 2^{2t} n)$ time.

For the second phase, we observe there are

$$\sum_{i=0}^{t} \binom{t}{i} 2^{t-i}$$

possible combinations for $A_j \subseteq Y_j$ and $B_j \subseteq Y_j - A_j$. Using the binomial theorem, we deduce

$$\sum_{i=0}^{t} \binom{t}{i} 2^{t-i} = \sum_{i=0}^{t} \binom{t}{i} 2^{t-i} 1^i = 3^t \ .$$

For each $v \in B_j$, we might update the values of $size_j(A_j, B_j)$ and $solution_j(A_j, B_j)$. This can be done in $O(t^2)$ time. As $3^t \leq 2^{2t}$, the second phase of the algorithm costs $O(t^2 2^{2t} n)$ time as well. Assuming table look-ups can be done in constant time, the resulting total running time of the algorithm is $O(t^2 2^{2t} n)$. □

2.2.3 Minimum Connected Dominating Set

To solve the minimum connected dominating set problem for unit disk graphs of bounded thickness, we build on the solution of the minimum dominating set problem. We start again by looking for a subproblem optimality principle. We observe that the subset of a minimum connected dominating set on slabs 1 to j is not necessarily connected. Therefore we need to define another problem, which can be solved slab by slab, but at the end still results in a minimum connected dominating set.

The problem we will solve is the minimum partial connected dominating set problem.

Definition 2.16 *For each j ($1 \leq j \leq b+1$), a set $pCDS_j \subseteq Y_{-1} \cup \cdots \cup Y_j$ is a* partial connected dominating set *of $Y_{-1} \cup \cdots \cup Y_{j-1} \cup B_j$ with $B_j \subseteq Y_j$ if and only if $pCDS_j$ is a dominating set for $Y_{-1} \cup \cdots \cup Y_{j-1} \cup B_j$ and either $C \cap Y_j \neq \emptyset$ for each connected component C of $pCDS_j$, or $j \geq b$ and $pCDS_j$ is connected.*

Note that the definition enforces that $pCDS_{b+1}$ is a connected dominating set. Because Y_{b+1} is empty, $C \cap Y_j = \emptyset$ for any connected component C of $pCDS_{b+1}$. Therefore $pCDS_{b+1}$ must be connected and thus it is a connected dominating set. Vica versa, for any connected dominating set CDS of G, the set $CDS \cap (Y_{-1} \cup \cdots \cup Y_j)$ is a partial connected dominating set of $Y_{-1} \cup \cdots \cup Y_{j-1}$, for each $1 \leq j \leq b+1$.

The *minimum partial connected dominating set problem* is to compute a partial connected dominating set for G of minimum size. Observe that this problem is equivalent to computing a minimum connected dominating set of G. However, because partial connected dominating sets are less strict, the minimum partial connected dominating set problem can be solved slab by slab. The important property of a partial connected dominating set $pCDS_j$ we will use is the fact that any connected component of $pCDS_j$ intersects the j-th slab. In a slab-by-slab dynamic programming algorithm, this ensures the constructed set will become a connected dominating set. However, remembering the connected components of each considered partial connected dominating set is challenging.

To simplify the exposition of the solution to this challenge, we introduce some new notions. For each connected component C_i ($1 \leq i \leq K_j$) of a partial connected dominating set, we denote $C_i \cap Y_j$ by A_j^i and let $A_j = A_j^1 \cup \cdots \cup A_j^{K_j}$. We observe that the sets $A_j^1, \ldots, A_j^{K_j}$ are mutually exclusive and also not connected to each other. This implies that a connected component of A_j cannot be 'distributed' over two sets A_j^i and A_j^l ($1 \leq i < l \leq K_j$). Hence each set A_j^i must be the union of one or more connected components of A_j. We call $A_j = A_j^1 \cup \cdots \cup A_j^{K_j}$ the *front* of the partial connected dominating set.

Consider partial connected dominating sets $pCDS_j$ and $pCDS_{j-1}$. They are called *compatible* if and only if $pCDS_{j-1} = pCDS_j \cap (Y_{-1} \cup \cdots \cup Y_{j-1})$. If $pCDS_j$ and $pCDS_{j-1}$ are compatible, this implies that each connected component of $pCDS_{j-1}$ is either still a connected component in $pCDS_j$, or has merged with one or more other connected components of $pCDS_{j-1}$ to form a connected component of $pCDS_j$. But then there must be a relation between the front of $pCDS_j$ and the front of $pCDS_{j-1}$. So let $A_j = A_j^1 \cup \cdots \cup A_j^{K_j}$ and $A_{j-1} = A_{j-1}^1 \cup \cdots \cup A_{j-1}^{K_{j-1}}$ be the front of respectively $pCDS_j$ and $pCDS_{j-1}$. The two fronts are called *compatible* if and only if either $j = b+1$ and $K_{j-1} \leq 1$, or $j \leq 1$, or $j = b$, $K_{j-1} = 1$, and $A_j = \emptyset$, or each A_{j-1}^i is connected to exactly one A_j^k, with $1 \leq i \leq K_{j-1}$ and $1 \leq k \leq K_j$. We now give a formal proof of the relation between the compatibility of the fronts and the compability of the partial connected dominating sets.

Proposition 2.17 *The fronts of $pCDS_j$ and $pCDS_{j-1}$ are compatible if $pCDS_j$ and $pCDS_{j-1}$ are compatible.*

Proof: Suppose $pCDS_j$ and $pCDS_{j-1}$ are compatible. If $j \leq 1$, then the two fronts are trivially compatible. If $j = b+1$, then $Y_j = \emptyset$ and thus $pCDS_j = pCDS_{j-1}$. Because $pCDS_j$ must be connected, $pCDS_{j-1}$ must also be connected. This implies that $K_{j-1} \leq 1$. Hence the two fronts are compatible.

If $j = b$ and $pCDS_j = pCDS_{j-1}$, then $A_j = \emptyset$. Furthermore, $pCDS_j$ must be connected and thus $pCDS_{j-1}$ consists of exactly one connected component C. Because $pCDS_{j-1}$ is a partial connected dominating set, $C \cap Y_{j-1} \neq \emptyset$. Hence $K_{j-1} = 1$. Therefore the two fronts are compatible.

If $j = b$ and $pCDS_j \neq pCDS_{j-1}$, then $A_j \neq \emptyset$. If $pCDS_j$ is connected or $C \cap Y_j \neq \emptyset$ for each connected component C of $pCDS_j$, then each A^i_{j-1} is connected to at least one A^k_j, with $1 \leq i \leq K_{j-1}$ and $1 \leq k \leq K_j$.

If $j < b$, then $C \cap Y_j \neq \emptyset$ for each connected component C of $pCDS_j$ and thus each A^i_{j-1} is connected to at least one A^k_j as well, with $1 \leq i \leq K_{j-1}$ and $1 \leq k \leq K_j$.

Now we note that if an A^i_{j-1} connects to A^k_j and A^l_j, with $1 \leq i \leq K_{j-1}$ and $1 \leq k < l \leq K_j$, then A_j would not be a correct front, because A^k_j and A^l_j should have been joined. Therefore each A^i_{j-1} is connected to exactly one A^k_j, with $1 \leq i \leq K_{j-1}$ and $1 \leq k \leq K_j$. Hence the two fronts must be compatible. $\square$

Proposition 2.18 *Let $pCDS_{j-1}$ be a partial connected dominating set with front $A_{j-1} = A^1_{j-1} \cup \cdots \cup A^{K_{j-1}}_{j-1}$ and let $A_j = A^1_j \cup \cdots \cup A^{K_j}_j$ be a possible front compatible with A_{j-1}. Then $pCDS_j = pCDS_{j-1} \cup A_j$ is a partial connected dominating set compatible with $pCDS_{j-1}$ if $pCDS_j$ is a dominating set for $Y_{-1} \cup \cdots \cup Y_{j-1} \cup B_j$, for some set $B_j \subseteq Y_j$.*

Proof: Suppose $pCDS_j$ is a dominating set for $Y_{-1} \cup \cdots \cup Y_{j-1} \cup B_j$, for some set $B_j \subseteq Y_j$. If $j = b+1$ and $K_{j-1} \leq 1$, then $pCDS_{j-1}$ must be connected and $A_j = \emptyset$. Hence $pCDS_j$ is a partial connected dominating set. If $j \leq 1$, then $C \cap Y_j \neq \emptyset$ for each connected component C of $pCDS_j$ is certainly true. So $pCDS_j$ is a partial connected dominating set. If $j = b$, $K_{j-1} = 1$, and $A_j = \emptyset$, then $pCDS_{j-1}$ is connected. Then $pCDS_j$ must also be connected and thus is a partial connected dominating set. If each A^i_{j-1} is connected to exactly one A^k_j, with $1 \leq i \leq K_{j-1}$ and $1 \leq k \leq K_j$, then trivially $C \cap Y_j \neq \emptyset$ for each connected component C of $pCDS_j$. Therefore $pCDS_j$ must be a partial connected dominating set. The compatibility of $pCDS_j$ with $pCDS_{j-1}$ follows straightforwardly. $\square$

In a dynamic programming algorithm, we are clearly only interested in compatible partial connected dominating sets. Hence we can use the following subproblem optimality principle.

Proposition 2.19 *For some j $(1 \leq j \leq b+1)$, let $A_j = A^1_j \cup \cdots \cup A^{K_j}_j$ and B_j be subsets of Y_j $(A_j \cap B_j = \emptyset)$. If there exists a minimum partial connected dominating set $pCDS_j \subseteq Y_{-1} \cup \cdots \cup Y_j$ for $Y_{-1} \cup \cdots \cup Y_{j-1} \cup B_j$ such that $(A^1_j, \ldots, A^{K_j}_j)$ is the front of $pCDS_j$, then there exists a minimum partial connected dominating set $pCDS_{j-1} \subseteq Y_{-1} \cup \cdots \cup Y_{j-1}$ with $A_{j-1} = A^1_{j-1} \cup \cdots \cup A^{K_{j-1}}_{j-1}$ the front of $pCDS_{j-1}$, such that $pCDS_{j-1} \cup A_j = pCDS_j$ (i.e. $pCDS_j$ and $pCDS_{j-1}$ are compatible) , $D^j(A_{j-1}) \supseteq B_j - D^j(A_j)$, and $pCDS_{j-1}$ dominates $(Y_{-1} \cup \cdots \cup Y_{j-2}) \cup (Y_{j-1} - A_{j-1} - D^{j-1}(A_j))$.*

Recall that $D^j(S)$ is the subset of Y_j dominated by S.

Proposition 2.17 and Proposition 2.18 have shown that we can use the compatibility of the fronts to test the compatibility of the partial connected dominating sets themselves. This observation can be exploited in the dynamic programming algorithm.

As mentioned before, we will adapt the algorithm for Minimum Dominating Set. We use the same enumeration strategy, but now we also have to consider all possible compatible fronts. So given subsets A_j and A_{j-1} of respectively Y_j and Y_{j-1}, we have to enumerate all compatible fronts $A_j^1, \ldots, A_j^{K_j}$ and $A_{j-1}^1, \ldots, A_{j-1}^{K_{j-1}}$, such that $A_j = A_j^1 \cup \cdots \cup A_j^{K_j}$ and $A_{j-1} = A_{j-1}^1 \cup \cdots \cup A_{j-1}^{K_{j-1}}$. We already noted that each set A_{j-1}^i (A_j^l) of a front must be the union of one or more connected components of A_{j-1} (A_j). This reduces to amount of enumeration work.

A good strategy would be to simply enumerate all possible partitions of the connected components of A_j and A_{j-1}.

Definition 2.20 *Let S be a finite set. Then $S_1, \ldots, S_p$ is called a* partition *of S if and only if $S_i \cap S_j = \emptyset$ for each $1 \le i < j \le p$ and $\bigcup_{i=1}^p S_i = S$.*

In other words, a partition of S is a decomposition of S into mutually exclusive, but collectively exhaustive subsets. This is exactly what we need. Unfortunately, the number of possible partitions of an m-element set is ϖ_m, the m-th Bell number (named for E.T. Bell [9, 10]). Lovász [32] proves that ϖ_m is approximately $m^{-1/2}[\lambda(m)]^{m+1/2}e^{\lambda(m)-m-1}$, where $\lambda(m)$ is a function such that $\lambda(m)\ln\lambda(m) = m$ [44]. Hence ϖ_m is $\Theta((\frac{m}{\log m})^m)$. As $|A_j| \le t$ and $|A_{j-1}| \le t$, this could imply a running time of $O(2^{2t}(\frac{t}{\log t})^{2t}n)$ for the dynamic programming algorithm. While this is polynomial if t is $O(\frac{\log n}{\log\log n})$, the running time is far of the $O(t^2 2^{2t} n)$ algorithms we have seen before.

To reduce the running time, we observe that we are only interested in so called non-crossing partitions of the connected components of A_j and A_{j-1}.

Definition 2.21 *Let S be a finite set and $\prec$ a partial ordering on these elements. Then $S_1, \ldots, S_p$ is a* non-crossing partition *of S if and only if $S_1, \ldots, S_p$ is a partition of S and for any i, j ($1 \le i, j \le p, i \ne j$) and any $a, b \in S_i$ and $c, d \in S_j$, $a \prec c \prec b \prec d$ is false.*

Here we have a finite set of vertices. Given two vertices u, v, we define $\prec$ such that $u \prec v$ if and only if $c_u^y < c_v^y$ or $c_u^y = c_v^y$ and $c_u^x < c_v^x$. We observe that for any two distinct connected components C and C' of A_j, either $u \prec v$ for each $u \in C, v \in C'$, or $v \prec u$ for each $u \in C, v \in C'$. If this would not be true, C and C' would be connected, which contradicts that C and C' are distinct connected components.

Lemma 2.22 *Given the above partial ordering $\prec$, it is sufficient to consider all non-crossing partitions of the connected components of A_j and A_{j-1}.*

Proof: Figure 4 shows four connected components of A_j. Under the partial ordering $\prec$, $a \cup c, b \cup d$ would be a crossing partition of these connected components (as shown in the figure). Then, in any partial connected dominating set of $Y_{-1} \cup \cdots \cup Y_j$ with front $a \cup c, b \cup d$, there must be at least one path from a to c and at least one path from b to d. Any pair of such paths must intersect. Hence the front $a \cup c, b \cup d$ is equivalent to the front $a \cup b \cup c \cup d$, which is non-crossing. Clearly, this holds for any crossing partition of any four connected components of A_j. We observe that any partition of three or less connected components can never be crossing. Therefore it suffices to consider only non-crossing partitions. □

Non-crossing partition were first considered by Becker [7, 8]. See Simion [41] for numerous applications of such partitions. Following Becker [8] and Kreweras [28], the number of non-crossing partitions of an m-element set is C_m, the m-th Catalan number. It is well known that C_m is $O(\frac{4^m}{m\sqrt{m}})$ [45].

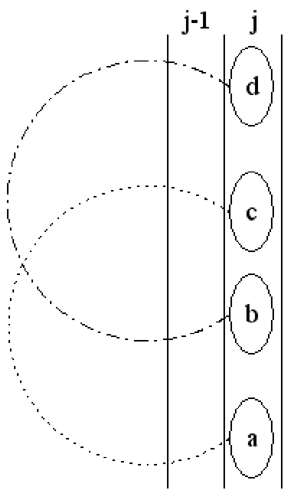

Figure 4: Connected components a, b, c, and d are partitioned in a crossing manner. Any path between a and c must intersect any path between b and d.

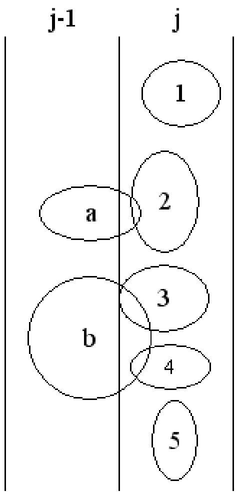

Figure 5: Non-crossing partition a, b of the connected components of A_{j-1} induces the non-crossing partition $1, 2, 3 \cup 4, 5$ of the connected components of A_j. Partition $a \cup b$ would induce $1, 2 \cup 3 \cup 4, 5$. Connected components 1 and 5 should never be in the same set of the partition, because they are not connected with vertices of the partial connected dominating set in slabs -1 to j.

Finally, suppose we are given a non-crossing partition $A^1_{j-1}, \dots, A^{K_{j-1}}_{j-1}$ of the connected components of A_{j-1} as a front for $pCDS_{j-1}$. Then this partition induces a unique compatible front of A_j. In the induced partition $A^1_j, \dots, A^{K_j}_j$, all connected components of A_j connected to an A^i_{j-1} $(1 \leq i \leq K_{j-1})$ are in the same A^l_j, for some $1 \leq l \leq K_j$, and for each connected component of A_j not connected to a vertex in A_{j-1}, there is an A^l_j containing just the vertices of that connected component (see Figure 5). The reverse also holds, i.e. each non-crossing partition of A_j induces a unique compatible non-crossing partition of A_{j-1}. We observe that all relevant non-crossing partitions of A_{j-1} will already be stored in the table maintained by the dynamic programming algorithm. Therefore we do not actually have to explicitely enumerate all non-crossing partitions, but can simply look them up in the table.

All these observations are used in Algorithm 2.3.

Lemma 2.23 *Algorithm 2.3 computes a minimum connected dominating set of a unit disk graph with thickness t in $O(t^2 2^{4t} n)$ time.*

Proof: As before, we prove this by induction on the value of $size_j$ and $solution_j$. The value of $size_j(A_j, B_j, (A^1_j, \dots, A^{K_j}_j))$ will be the size of a minimum partial connected dominating set $pCDS_j \subseteq Y_{-1} \cup \dots \cup Y_j$, such that $pCDS_j$ dominates $Y_{-1} \cup \dots \cup Y_{j-1} \cup B_j$, $pCDS_j \cap Y_j = A_j$, and $(A^1_j, \dots, A^{K_j}_j)$ is the front of $pCDS_j$. This partial connected dominating set is stored in $solution_j(A_j, B_j, (A^1_j, \dots, A^{K_j}_j))$. If such a partial connected dominating set does not exist, then $size_j(A_j, B_j, (A^1_j, \dots, A^{K_j}_j)) = \infty$ and $solution_j(A_j, B_j, (A^1_j, \dots, A^{K_j}_j)) = \emptyset$.

Trivially, $size_0$ and $solution_0$ are correctly computed by the algorithm. Now inductively assume that for some $j \geq 1$, $size_{j-1}$ and $solution_{j-1}$ have been correctly computed by the algorithm. Consider the set Y_j and let A_j be an arbitrary subset of Y_j. Let A_{j-1} be an arbitrary subset of Y_{j-1}.

Next, let $(A^1_{j-1}, \dots, A^{K_{j-1}}_{j-1})$ be an arbitrary, non-crossing partition of the connected components of A_{j-1}. If there is a set A^i_{j-1} $(1 \leq i \leq K_{j-1})$ not connected to a vertex of A_j, then no partition of A_j can be a compatible front. Therefore we skip to the next non-crossing partition of A_{j-1}.

Otherwise, let $(A^1_j, \dots, A^{K_j}_j)$ be the non-crossing partition of the connected components of A_j induced by $(A^1_{j-1}, \dots, A^{K_{j-1}}_{j-1})$. Then let $B_j = D^j(A_j) \cup D^j(A_{j-1}) - A_j$ and set $C_{j-1} = D^{j-1}(A_j) - A_{j-1}$. Now let $B_{j-1} = Y_{j-1} - A_{j-1} - C_{j-1}$ be the remaining vertices of slab $j-1$.

By induction, $size_{j-1}(A_{j-1}, B_{j-1}, (A^1_{j-1}, \dots, A^{K_{j-1}}_{j-1}))$ must either be the size of a minimum partial connected dominating set $pCDS_{j-1} \subseteq Y_{-1} \cup \dots \cup Y_{j-1}$, such that $pCDS_{j-1}$ dominates $Y_{-1} \cup \dots \cup Y_{j-2} \cup B_{j-1}$, $pCDS_{j-1} \cap Y_{j-1} = A_{j-1}$, and $(A^1_{j-1}, \dots, A^{K_{j-1}}_{j-1})$ is the front of $pCDS_{j-1}$, or it has value ∞. If $size_{j-1}(A_{j-1}, B_{j-1}, (A^1_{j-1}, \dots, A^{K_{j-1}}_{j-1}))$ has value ∞, then by definition no such partial connected dominating set exists, and we do not have to update any data structures.

Otherwise, by their construction, the two fronts $(A^1_j, \dots, A^{K_j}_j)$ and $(A^1_{j-1}, \dots, A^{K_{j-1}}_{j-1})$ are compatible. Using Proposition 2.18, we know that $solution_{j-1}(A_{j-1}, B_{j-1}, (A^1_{j-1}, \dots, A^{K_{j-1}}_{j-1})) \cup A_j$ must be a partial connected dominating set for $Y_{-1} \cup \dots \cup Y_{j-1} \cup B_j$ with front $(A^1_j, \dots, A^{K_j}_j)$. Hence if $size_{j-1}(A_{j-1}, B_{j-1}, (A^1_{j-1}, \dots, A^{K_{j-1}}_{j-1})) + |A_j| < size_j(A_j, B_j, (A^1_j, \dots, A^{K_j}_j))$ or partition $(A^1_{j-1}, \dots, A^{K_{j-1}}_{j-1})$ has not yet been considered for this A_j and B_j, then we update

```
1.  Set $size_0(\emptyset,\emptyset,\emptyset) = 0$ and $solution_0(\emptyset,\emptyset,\emptyset) = \emptyset$
2.  for $j \leftarrow 1$ to $b+1$
3.  do  for each $A_j \subseteq Y_j$
4.      do  for each $A_{j-1} \subseteq Y_{j-1}$
5.          do  for each possible front $(A^1_{j-1}, \ldots, A^{K_{j-1}}_{j-1})$
6.              do  if   there is an $A^i_{j-1}$ not connected to a vertex of $A_j$
7.                  then skip
8.                  Let $B_j = D^j(A_j) \cup D^j(A_{j-1}) - A_j$,
                     $C_{j-1} = D^{j-1}(A_j) - A_{j-1}$,
                     $B_{j-1} = Y_{j-1} - A_{j-1} - C_{j-1}$, and
                     $(A^1_j, \ldots, A^{K_j}_j)$ the front induced by $(A^1_{j-1}, \ldots, A^{K_{j-1}}_{j-1})$
9.                  if   $size_{j-1}(A_{j-1}, B_{j-1}, (A^1_{j-1}, \ldots, A^{K_{j-1}}_{j-1})) \neq \infty$ and
                         $|A_j| + size_{j-1}(A_{j-1}, B_{j-1}, (A^1_{j-1}, \ldots, A^{K_{j-1}}_{j-1})) <$
                           $size_j(A_j, B_j, (A^1_j, \ldots, A^{K_j}_j))$
10.                 then $size_j(A_j, B_j, (A^1_j, \ldots, A^{K_j}_j)) =$
                           $|A_j| + size_{j-1}(A_{j-1}, B_{j-1}, (A^1_{j-1}, \ldots, A^{K_{j-1}}_{j-1}))$
11.                      $solution_j(A_j, B_j, (A^1_j, \ldots, A^{K_j}_j)) =$
                           $A_j \cup solution_{j-1}(A_{j-1}, B_{j-1}, (A^1_{j-1}, \ldots, A^{K_{j-1}}_{j-1}))$
12.                 fi
13.             od
14.         od
15.     od
16.     for each $A_j \subseteq Y_j$
17.     do  for each $B_j \subseteq Y_j - A_j$ (in order of descending $|B_j|$)
18.         do  for each possible front $(A^1_j, \ldots, A^{K_j}_j)$
19.             do  for each $v \in B_j$
20.                 do  if   $size_j(A_j, B_j, (A^1_j, \ldots, A^{K_j}_j)) <$
                               $size_j(A_j, B_j \backslash \{v\}, (A^1_j, \ldots, A^{K_j}_j))$
21.                     then $size_j(A_j, B_j \backslash \{v\}, (A^1_j, \ldots, A^{K_j}_j)) =$
                               $size_j(A_j, B_j, (A^1_j, \ldots, A^{K_j}_j))$
22.                          $solution_j(A_j, B_j \backslash \{v\}, (A^1_j, \ldots, A^{K_j}_j)) =$
                               $solution_j(A_j, B_j, (A^1_j, \ldots, A^{K_j}_j))$
23.                     fi
24.                 od
25.             od
26.         od
27.     od
28. od
29. return $(size_{b+1}(\emptyset,\emptyset,\emptyset), solution_{b+1}(\emptyset,\emptyset,\emptyset))$
```

Algorithm 2.3: SlabDecompositionMpCDS$(G, Y_1, \ldots, Y_b)$

$size_j(A_j, B_j, (A_j^1, \ldots, A_j^{K_j}))$ and $solution_j(A_j, B_j, (A_j^1, \ldots, A_j^{K_j}))$.

From the proof of Lemma 2.15, we know that the computed dominating set will indeed have minimum size for each A_j, B_j, and $(A_j^1, \ldots, A_j^{K_j})$. Thus, using the subproblem optimality principle (Proposition 2.19), the algorithm computes (the size of) a minimum partial connected dominating set $pCDS_j \subseteq Y_{-1} \cup \cdots \cup Y_j$, such that $pCDS_j$ dominates $Y_{-1} \cup \cdots \cup Y_{j-1} \cup B_j$, $pCDS_j \cap Y_j = A_j$, and $(A_j^1, \ldots, A_j^{K_j})$ is the front of $pCDS_j$. This holds for each A_j, B_j, and $(A_j^1, \ldots, A_j^{K_j})$. Therefore, by induction, we have proved that $size_j$ and $solution_j$ are correctly computed.

As mentioned earlier, a partial connected dominating set for $Y_{-1} \cup \cdots \cup Y_{b+1}$ is a minimum connected dominating set of G. Thus $size_{b+1}$ will be the size of a minimum connected dominating set of G and this set is stored in $solution_{b+1}$. Hence the algorithm correctly computes a minimum connected dominating set of G.

Given our preceding discussions, the time bound is relatively easy to prove. Because $b \leq n$, the outer **for**-loop is executed at most $n + 1$ times. For each of the at most 2^t subsets A_j of Y_j, for each of the at most 2^t subsets A_{j-1} of Y_{j-1}, and for each of the at most C_t non-crossing partitions of the connected components of A_{j-1}, the induced partition of the connected components of A_j and the sets dominated by A_j and A_{j-1} must be computed. This can be done in $O(t^2)$ time. Furthermore, we must check if a partition $(A_j^1, \ldots, A_j^{K_j})$ of A_j has been considered before. Using a balanced search tree, this can be done in $O(\log C_t) = O(t)$ time. Since C_t is $O(\frac{4^t}{t\sqrt{t}})$ and all relevant partitions of A_{j-1} are stored in the maintained table, this part of the algorithm costs $O(\sqrt{t}2^{4t}n)$ time.

In the post-processing phase, we can also use the partitions of A_j stored in the table. There are at most 3^t combinations of $A_j \subseteq Y_j$ and $B_j \subseteq Y_j - A_j$. Given these sets and a non-crossing partition of A_j, all computations and updates take at most $O(t^2)$ time. Therefore this part requires $O(t^2 2^{(2+\log 3)t} n)$ time. Because $2^{(2+\log 3)t} \leq 2^{4t}$, the total running time of the algorithm is $O(t^2 2^{4t} n)$. □

2.2.4 Computational consequences

Consider Lemma 2.12, Corollary 2.13, Lemma 2.15, and Lemma 2.23. We can summarize the results of this section as follows.

Theorem 2.24 *Let $G = (V, E)$ be a unit disk graph with known disk representation D and minimum thickness t^*. Then Maximum Independent Set, Minimum Vertex Cover, and Minimum Dominating Set can be solved in $O({t^*}^2 2^{2t^*} n)$ time. Minimum Connected Dominating Set can be solved in $O({t^*}^2 2^{4t^*} n)$ time.*

Because the running times are of the form $O(f(t^*)\, n)$, we have the following corollary.

Corollary 2.25 *Maximum Independent Set, Minimum Vertex Cover, and Minimum (Connected) Dominating Set are fixed parameter tractable (in t^*) for unit disk graphs with a known disk representation.*

Proof: This follows immediately from Theorem 2.24 and Definition 1.6. □

Note that the running times also imply that polynomial time algorithms for the problems exist if $t^* = t^*(n) \leq c \log n$ for some constant $c > 0$. As this will become important in the subsequent section, we formulate this in the following theorem.

Theorem 2.26 *If the minimum thickness $t^* = t^*(n)$ of a unit disk graph $G = (V, E)$ with known disk representation D is bounded by $c \log n$ for some constant $c > 0$, then Maximum Independent Set, Minimum Vertex Cover, and Minimum Dominating Set can be solved in $O(c^2 n^{2c+1} \log^2 n)$ time. Minimum Connected Dominating Set can be solved in $O(c^2 n^{4c+1} \log^2 n)$ time.*

3 Density

If the thickness of a unit disk graph is large, for instance in the order of $\sqrt{n}$, then the running time of the algorithms discussed in the previous section becomes non-polynomial.

In these scenarios, we have to resort to approximation algorithms to obtain a polynomial time algorithm. Therefore we present a new approximation scheme for optimization problems like Maximum Independent Set on unit disk graphs. This approximation scheme will use the shifting technique and the algorithms from the previous section as subroutines. We introduce a new, realistic, and relevant notion for unit disk graphs called *density*. The new approximation scheme improves on existing algorithms by focussing on unit disk graphs of bounded density.

Before we describe the new scheme, we introduce the definitions and relevant algorithms for the notion of density on unit disk graphs[2].

3.1 Preliminaries

As in the previous section, we consider unit disk graphs with a known disk representation. Each disk is assumed to have radius $\frac{1}{2}$. We first define a grid decomposition and describe the density of such a decomposition. Then we prove that the studied optimization problems remain NP-hard on unit disk graphs of bounded density.

3.1.1 Definitions

The density of a unit disk graph is related to the thickness of a unit disk graph. Density can be seen as a two-dimensional generalization of thickness. As such, we begin by defining grid decompositions, similar to the slab decompositions defined in Section 2.

Definition 3.1 *Let $0 \leq \alpha < \frac{1}{2}\pi$ be an angle and p an arbitrary point in the plane. Partition the plane using an infinite grid, such that each grid square has width and height 1, the vertical lines of the grid intersect the x-axis at angle α, and the top-left corner of exactly one grid square is on p. We call the horizontal and vertical lines defining the grid the* horizontal *and* vertical grid boundaries. *A* grid square *is the area between two neighboring horizontal and two neighboring vertical grid boundaries.*

Observe that the partitioning of the plane imposed by the grid remains the same after a rotation of $\frac{1}{2}\pi$ around p. Hence $0 \leq \alpha < \frac{1}{2}\pi$ is sufficient.

Definition 3.2 *Given a grid as described above, a disk (c_i, r_i) is considered to be* in *a grid square if its center c_i is between the two horizontal and between the two vertical grid boundaries defining the square. If a center is on a vertical (horizontal) grid boundary, then the disk is*

[2]The notion we use differs from the existing notion of density in graph theory as introduced by Miller and Vavasis [36].

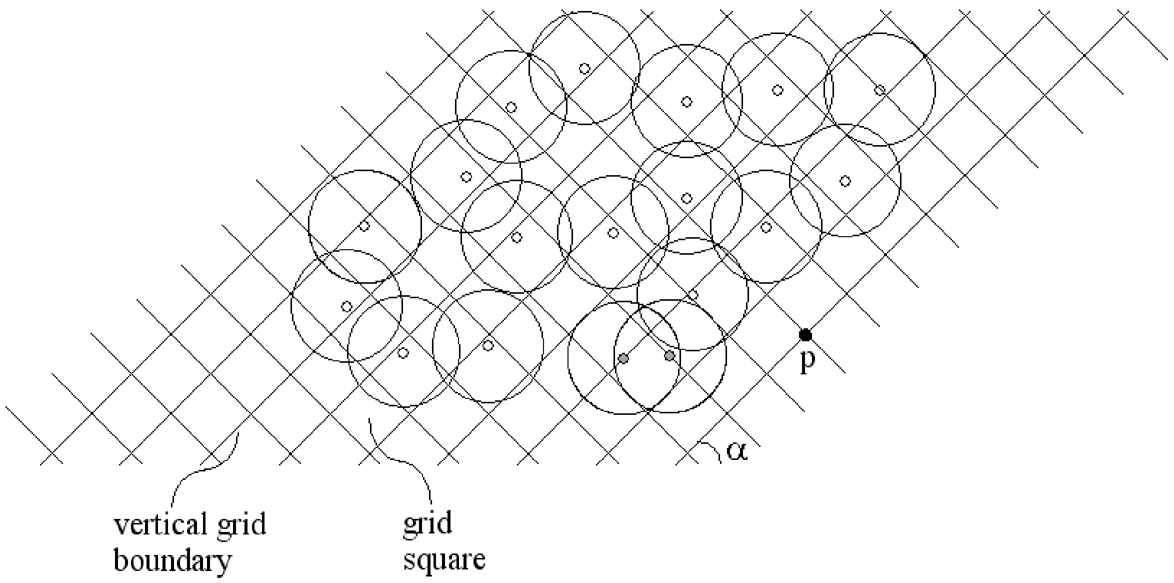

Figure 6: An arbitrary set of disks decomposed by a grid. The angle of the vertical grid boundaries is α and the top-left corner of exactly one grid square is on p. Note that the grey disk centers are both in the same grid square.

considered to be in the grid square to the right of (below) the boundary. The grid squares decompose D into mutually disjoint, collectively exhaustive subsets. This decomposition is called a grid decomposition *$g = \langle \alpha, p \rangle$ of D. A grid decomposition g of a unit disk graph G is the grid decomposition of a disk representation of G.*

See Figure 6 for a visual explanation of this definition. Using grid decompositions, we can define density.

Definition 3.3 *The* density *$d(D, g)$ of a grid decomposition g of D is defined as the maximum number of disk centers of D in any grid square of g. The density of a grid decomposition of a unit disk graph G is $d(G, g) = d(D, g)$, where D is a disk representation of G.*

The density of the grid decomposition in Figure 6 for example is 2.

In the previous section, we refined the specification of thickness by 'replacing' the point p by a disk center in D. Here we can apply a similar trick. Consider a grid decomposition $\langle \alpha, p \rangle$. First we move the vertical boundaries to the right, until a vertical boundary intersects a disk center c_v. Then we move the horizontal boundaries down, until a horizontal boundary intersects a disk center c_h. Note that c_v could be equal to c_h. This move of the grid clearly does not affect the density of the resulting decomposition. Hence, for our purposes, we can fully specify a grid decomposition by a three-tuple $\langle \alpha, c_v, c_h \rangle$, where c_v and c_h are disk centers in D.

Also note that given a grid decomposition $\langle \alpha, c_v, c_h \rangle$, we can easily obtain an equivalent decomposition in which the vertical boundaries are perpendicular to the x-axis. To ensure the density remains the same, we apply transformation matrix T_α (see Equation 1) to the centers of all disks. For simplicity, we will often use a decomposition where a top-left corner of a grid square corresponds to the origin. Because c_v must be on vertical boundary and c_h must be on a horizontal boundary, $(T_\alpha(c_v)^x, T_\alpha(c_h)^y)$ must be the top-left corner of a grid square. Hence we set

$$T_g \quad = \quad \mathsf{translate}(-(T_\alpha(c_v)^x, T_\alpha(c_h)^y)) \cdot T_\alpha = \begin{pmatrix} 1 & 0 & -T_\alpha(c_v)^x \\ 0 & 1 & -T_\alpha(c_h)^y \\ 0 & 0 & 1 \end{pmatrix} \cdot T_\alpha \tag{2}$$

Clearly, the application of T_g ensures the top-left corner of a grid square is on the origin, while the density remains the same.

Definition 3.4 *Let $g^* = \langle \alpha^*, c_v^*, c_h^* \rangle$ be a grid decomposition (where c_v^* and c_h^* are disk centers in D), such that for all grid decompositions g, $d(D, g^*) \leq d(D, g)$. Denote $d^* = d(D, g^*)$.*

Observe that d^* is the minimum density of a grid decomposition of D.

The density of a given grid decomposition and the minimum density can be computed in polynomial time using similar methods as used when computing (minimum) thickness [29].

3.1.2 Complexity

Maximum Independent Set, Minimum Vertex Cover, and Minimum Dominating Set are NP-hard for unit disk graphs. This was proved by Clark, Colbourn, and Johnson [16] using a reduction from the same problems on planar graphs of degree 3 and 4. By adapting their proof, we are able to prove the following theorem [29].

Theorem 3.5 *Maximum Independent Set and Minimum Vertex Cover are NP-hard for unit disk graphs of density* 1*. Minimum Dominating Set is NP-hard for unit disk graphs of density* 2*.*

This theorem straightforwardly implies that Maximum Independent Set and Minimum Vertex Cover are NP-hard for unit disk graphs of arbitrary density and that Minimum Dominating Set is NP-hard for unit disk graphs of density at least 2.

Minimum Connected Dominating Set was proved NP-hard for unit disk graphs by Lichtenstein [31] using a reduction from 3SAT. The instances of Connected Dominating Set constructed in this proof have density 3. Hence Minimum Connected Dominating Set is NP-hard for unit disk graphs of density at least 3.

We now prove no FPTAS exists for the considered problems, unless P=NP.

Theorem 3.6 *No FPTAS exists for Maximum Independent Set, Minimum Vertex Cover, or Minimum (Connected) Dominating Set on unit disk graphs of bounded density, unless P=NP.*

Proof: We use the notion of a polynomially bounded optimization problem. Ausiello *et al.* [5] define an optimization problem to be *polynomially bounded* if there exists a polynomial p, such that for any instance x of the problem and for any feasible solution SOL_x of x, $SOL_x \leq p(|x|)$. Informally, a problem is polynomially bounded if the size of any feasible solution is polynomial in the size of the input. Ausiello *et al.* then prove that no NP-hard polynomially bounded optimization problem admits an FPTAS, unless P=NP. Clearly, Maximum Independent Set, Minimum Vertex Cover, and Minimum (Connected) Dominating Set are polynomially bounded optimization problems. Furthermore, they are NP-hard on unit disk graphs of bounded density. Hence no FPTAS can exist unless P=NP. □

3.2 A new approximation scheme

We present a new approximation scheme based on the notion of density introduced in the previous section. We first describe the general idea of the algorithm, and then show how to apply it to Maximum Independent Set, Minimum Vertex Cover, and Minimum (Connected) Dominating Set.

3.2.1 Shifting strip decompositions

Let $G = (V, E)$ be a unit disk graph with known disk representation D, containing disks of radius $\frac{1}{2}$. Furthermore, let g be some grid decomposition of D with density $d = d(D, g)$ and let T_g be a transformation matrix as defined in equation 2.

The main idea of the new scheme is to use the shifting technique to decompose the set of disks into subsets of bounded thickness, and then apply the algorithms described in Section 2.2 to each subset. Recall that the idea behind the shifting technique is to use a set of regularly spaced separators to decompose the problem into smaller, easier solvable subproblems, and then combine the solutions of these subproblems to a solution of the global problem. This is repeated for several placements of the separator set, and the best solution found is then output as an approximation of the optimum.

In this approximation scheme, the separators will be a set of horizontal lines, which partition the plane into horizontal *strips*. The lines are called *strip boundaries* and are of the form $y = j$, where $j \in \mathbb{Z}$. The *height* of a strip is equal to the distance between the two strip boundaries defining the strip.

Definition 3.7 *Consider a strip defined by two horizontal lines $y = j$ and $y = l$ ($j < l$ and $j, l \in \mathbb{Z}$). Then a disk center c_i is* in *the strip if and only if $j < T_g(c_i)^y \leq l$.*

The decomposition of the disk centers induced by the strips is called a *strip decomposition.* Observe that a strip decomposition also induces a decomposition of the graph, such that each strip corresponds to the induced subgraph $G[V'] = (V', (V' \times V') \cap E)$, where $V' \subseteq V$ is the set of vertices corresponding to the disk centers in the strip.

Now choose some polynomially bounded function $f(n)$, such that $f(n) \geq d$ for all $n \in \mathbb{N}$. Then we space the strip boundaries such that the thickness t of the set of disks in each strip satisfies $f(n) \leq t < f(n)+d$. Because the thickness of each strip is between $f(n)$ and $f(n)+d$, the strip decomposition consists of at most $\lceil \frac{n}{f(n)} \rceil$ strips. As the density is d, the height of a strip must be at least $\frac{f(n)}{d}$ (see Figure 7). Such a strip decomposition can be constructed straightforwardly, as proved in the following lemma.

Lemma 3.8 (van Leeuwen [29]) *Given a set of disks D with density d, a set of strip boundaries such that the thickness t of the set of disks in each strip satisfies $f(n) \leq t < f(n)+d$ can be computed in $O(n \log n)$ time. The strip boundaries in the returned set are sorted by y-coordinate.*

Using this lemma, we can ensure that the thickness of each strip in the computed strip decomposition is bounded by $f(n) + d$. By choosing $f(n)$ appropriately, we can apply the algorithms of Section 2.2 and solve the considered problem optimally for (the disks of) each strip in polynomial time. Then we combine the solution for each strip to a solution of the

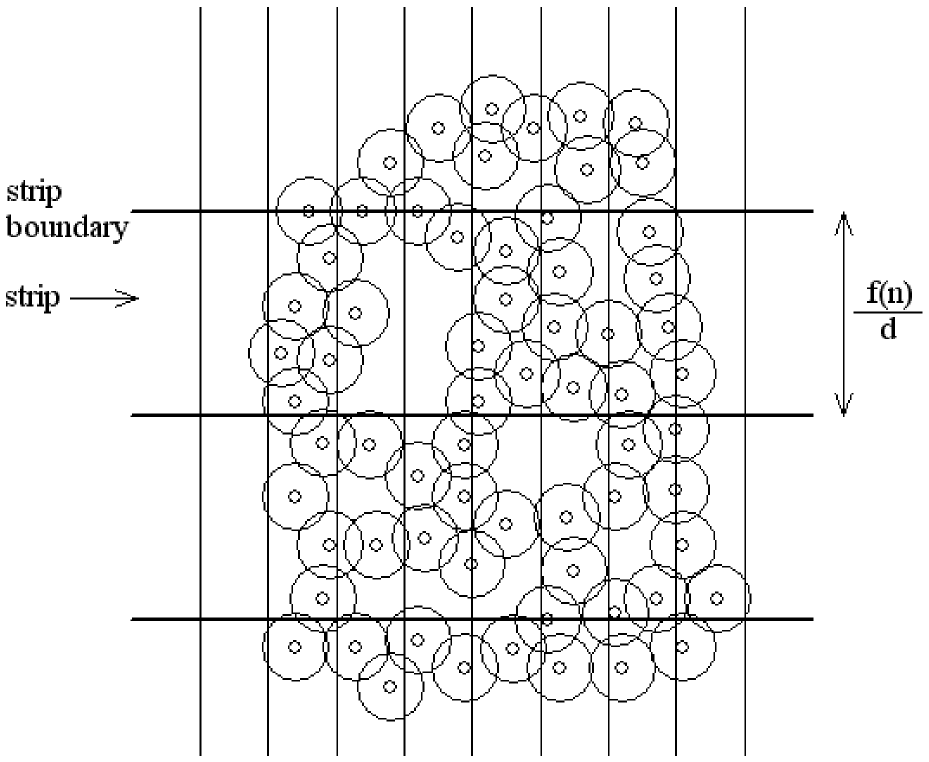

Figure 7: A strip decomposition of an arbitrary set of disks. Because the thickness t of each strip satisfies $f(n) \leq t < f(n) + d$, the height of each strip must be at least $\frac{f(n)}{d}$.

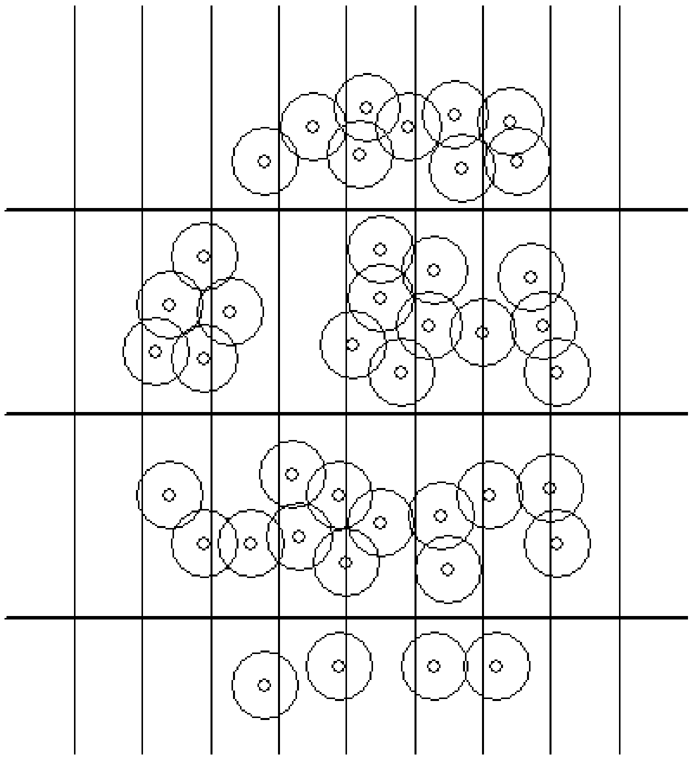

Figure 8: The set of disks which remain from the disks in Figure 7 after removing all disks intersecting a strip boundary. Clearly, the disks in a strip are independent of the disks in other strips.

global problem. By repeating this for several placements of the strip boundaries, we obtain an approximation of the optimum.

While the general idea remains the same for each problem, the details differ significantly. Therefore we describe how this idea can be applied to Maximum Independent Set, Minimum Vertex Cover, and Minimum (Connected) Dominating Set in the next paragraphs.

3.2.2 Maximum Independent Set

Consider a strip decomposition as described above. We observe that the (disks in the different) strips are not necessarily independent. Hence the combination of independent sets of each strip might not be an independent set and thus not a solution to the global problem. Therefore we remove all disks intersecting a strip boundary. A disk (c_i, r_i) is considered to intersect a strip boundary $y = j$ $(j \in \mathbb{Z})$ if $j - r_i \leq T_g(c_i)^y < j + r_i$. The result is a set of independent subsets of disks (see Figure 8).

Now let k be an integer $(0 < k \leq \frac{f(n)}{d})$. Denote by D_a $(0 \leq a \leq k-1)$ the set of mutually exclusive subsets of D obtained by partitioning D using horizontal strips as described before, but with the strip boundaries shifted down by a. Observe that each shift can increase the thickness t of a strip by at most d. Hence, if D_0 is the initial decomposition with thickness t such that $f(n) \leq t < f(n) + d$, then D_a has strips of thickness at most $f(n) + (a+1)d$. As $a \leq k - 1 \leq \frac{f(n)}{d} - 1$, the thickness of a strip will be at most $f(n) + kd \leq 2f(n)$.

Because the thickness of every strip is at most $2f(n)$, we can use Lemma 2.12 to show that we can compute a maximum independent set of each strip in $O(f(n)^2 2^{4f(n)} n)$ time. Denote the union of the maximum independent sets thus computed in all strips of D_a by IS_a. Clearly IS_a is a maximum independent set for D_a. Let IS_{max} be such that $|IS_{max}| = \max_{0 \le a \le k-1} |IS_a|$.

Lemma 3.9 *IS_{max} is at least a $(1 - \frac{1}{k})$-approximation of a maximum independent set of D.*

Proof: Let IS^* be a maximum independent set of D. Because the diameter of each disk in D is 1, no disk can be intersected by a strip boundary for more than one value of a. So if IS^*_a is the set of disks of IS^* in D_a, then $\sum_{a=0}^{k-1} |IS^*| - |IS^*_a| \le |IS^*|$. But then there must be a value of a such that $|IS^*| - |IS^*_a| \le |IS^*|/k$, and thus $|IS^*_a| \ge (1 - \frac{1}{k})|IS^*|$.

Now we observe that, since IS_a is a maximum independent set for D_a, $|IS_a| \ge |IS^*_a|$. This holds for any a, but for the value of a for which $|IS^*_a| \ge (1 - \frac{1}{k})|IS^*|$ in particular. Hence for this value of a we have

$$|IS_a| \ge |IS^*_a| \ge \left(1 - \frac{1}{k}\right) |IS^*|.$$

Because $|IS_{max}| = \max_{0 \le a \le k-1} |IS_a|$, we clearly also have $|IS_{max}| \ge (1 - \frac{1}{k})|IS^*|$. □

Observe that theoretically, the best attainable approximation factor is $1 - \frac{1}{\frac{f(n)}{d}} = 1 - \frac{d}{f(n)}$. This is because we assume for analytical purposes that the density of each grid square is (close to) d. In practice however, the density of most grid squares can be (significantly) lower than d. In this case, the height of a strip may be larger than $\frac{f(n)}{d}$ and larger values of k can be allowed for the same value of $f(n)$. This can lead to better approximation factors.

Lemma 3.10 *IS_{max} can be computed in $O(kn^2 f(n) 2^{4f(n)} + kn \log n)$ time.*

Proof: For each $0 \le a \le k-1$, there are two main components in the computation. First we need to compute D_a, and then we must compute a maximum independent set for D_a.

As a pre-processing step, we compute a sorted set $B \subseteq \{y = j \mid j \in \mathbb{Z}\}$ of strip boundaries, such that every strip has thickness between $f(n)$ and $f(n) + d$. According to Lemma 3.8, this costs $O(n \log n)$ time.

Given B, we can easily compute D_a by removing a disk (c_i, r_i) from D if $j - r_i \le T_g(c_i)^y + a < j + r_i$, for some $(y = j) \in B$. Because B is sorted and $|B| \le \lceil \frac{n}{f(n)} \rceil \le n$, we can check in $O(\log n)$ time if a disk must be removed. Thus takes $O(n \log n)$ time in total for each D_a.

As observed earlier, computing a maximum independent set for each strip can be done in $O(f(n)^2 2^{4f(n)} n)$ time. Thus we spend $O(f(n) 2^{4f(n)} n^2)$ time for all up to $\frac{n}{f(n)}$ strips. Maintaining the maximum IS_a takes no more than $O(n)$ time. This brings the total running time for all $0 \le a \le k-1$ to $O(kn^2 f(n) 2^{4f(n)} + kn \log n)$. □

By choosing $f(n)$ appropriately, the running time of the algorithm described above is polynomial in n and k.

Corollary 3.11 *If we choose $f(n) = \frac{1}{4} \log n$, then IS_{max} can be computed in time polynomial in n and k.*

Proof: If $f(n) = \frac{1}{4} \log n$, then it follows from the previous lemma that the computation of IS_{max} takes $O(kn^3 \log n + kn \log n)$, which is $O(kn^3 \log n)$. This obviously is polynomial in n and k. □

If we now set $k = \lceil \frac{1}{\epsilon} \rceil$, it is tempting to conclude from Lemma 3.9 and Corollary 3.11 that we have obtained an FPTAS for the maximum independent set problem. However, this does not take into account that k can be at most $\frac{f(n)}{d} = \frac{\log n}{4d}$. Hence we get a $(1-\epsilon)$-approximation, but with $\frac{4d}{\log n}$ as the lowest possible value of ϵ for a given n. This means that any value for ϵ can be obtained, provided $\frac{d}{f(n)} = \frac{4d}{\log n}$ becomes smaller than any desired value for n large enough. This is certain to be the case if $d = d(n) = o(\log n)$. Therefore we have the following theorem.

Theorem 3.12 *There exists an FPTAS$^\infty$ for the maximum independent set problem on unit disk graphs of bounded density, i.e. of density $d = d(n) = o(\log n)$.*

3.2.3 Extending to Minimum Vertex Cover and Minimum Dominating Set

For Minimum Vertex Cover and Minimum Dominating Set, we can follow the same approach as for Maximum Independent Set. There are however some subtle differences between solving the former two and the latter.

The most important difference is that we should not remove disks intersecting a strip boundary. Otherwise it could be that the union of the solutions for the strips is not a solution to the global problem. If we consider Minimum Vertex Cover for example, the edges crossing a strip boundary would not be covered. To ensure that the combined solutions do form a solution to the global vertex cover problem, the subset of D for a strip will consist of those disks that are either completely contained in the strip or intersect a strip boundary. Thus, given a strip boundary $y = j$ ($j \in \mathbb{Z}$), a disk (c_i, r_i) will be in the strip above the boundary if $j - r_i \leq T_g(c_i)^y$ and in the strip below the boundary if $T_g(c_i)^y < j + r_i$ (see Figure 9). Note that this implies that some disks will be in two strips. Only then can we be certain that all edges will be covered by a vertex cover.

For Minimum Dominating Set, we must take similar measures. We observe that a dominating set for the disks in a strip can contain any disk outside the strip that intersects a disk in the strip. Therefore a subset of D for a strip will consist of those disks for which either the disk center is in the strip or the disk intersects a disk in the strip. Thus, given a strip boundary $y = j$ ($j \in \mathbb{Z}$), a disk (c_i, r_i) will be in the strip above the boundary if $j - 2r_i \leq T_g(c_i)^y$ and in the strip below the boundary if $T_g(c_i)^y < j + 2r_i$ (see Figure 9).

Let $f(n) \geq 2d$ be some function and let k be an integer ($0 < k \leq \frac{f(n)}{d}$). Denote by D_a ($0 \leq a \leq k-1$) the set of subsets of D obtained by partitioning D using horizontal strips as defined above, where the strips have thickness between $f(n)$ and $f(n) + d$ and boundaries are shifted down by a. After adding disks close to strip boundaries (i.e. within distance $\frac{1}{2}$ and 1 from the boundary for respectively Minimum Vertex Cover and Minimum Dominating Set), we observe that the thickness of each strip in D_0 is at most $f(n) + 3d$. Each shift can increase the thickness by at most d. Hence the thickness of each subproblem in D_a is at most $f(n) + (a+3)d$. Since $a \leq k - 1 \leq \frac{f(n)}{d} - 1$, the thickness of each strip is at most

$$f(n) + \left(\frac{f(n)}{d} - 1 + 3\right) d = 2f(n) + 2d \leq 3f(n).$$

Therefore, by Corollary 2.13 and Lemma 2.15, we are able to compute a minimum vertex cover and a minimum dominating set for each strip in D_a in $O(f(n)^2 2^{6f(n)} n)$ time.

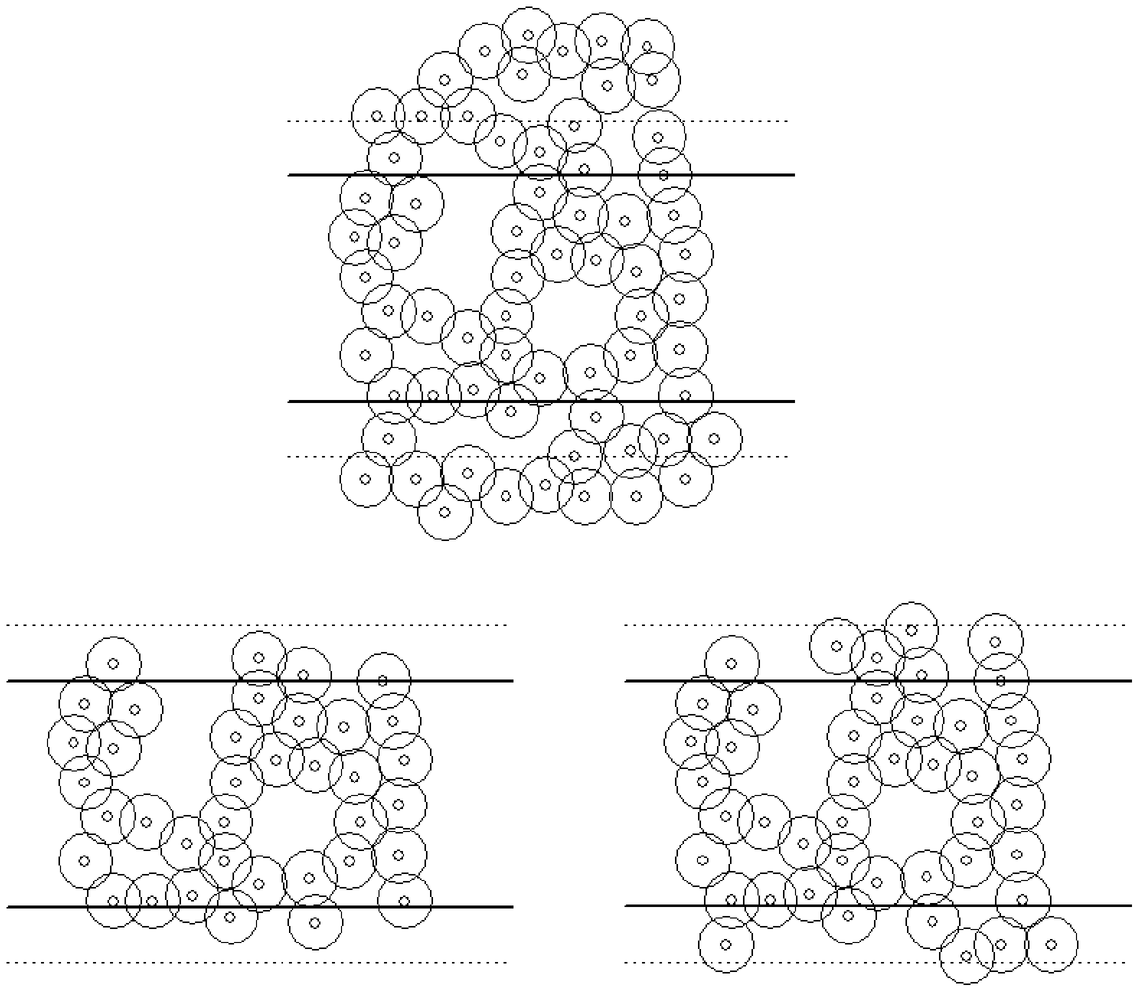

Figure 9: Above is an arbitrary set of disks and one strip. The dotted lines are at distance 1 from the strip boundaries. Below and to the left, we show the disks in the strip for the minimum vertex cover problem. A disk must be completely contained between the dotted lines to be in the strip. Below and to the right, we show the disks in the strip for the minimum dominating set problem. The disk center of a disk must be between the dotted lines for the disk to be in the strip.

Denote the union of the vertex covers for the strips of D_a by VS_a. Clearly, VS_a is a vertex cover for D. Let VS_{min} be such that $|VS_{min}| = \min_{0 \leq a \leq k-1} |VS_a|$. Then we can prove the following approximation factor.

Lemma 3.13 *VS_{min} is at least a $(1 + \frac{1}{k})$-approximation of a minimum vertex cover of D.*

Proof: Let VS^* be a minimum vertex cover of D. Let VS^*_a denote the collection of disks of VS^* in each of the subsets of D in D_a. Observe that some disks can be contained twice in VS^*_a, because of the overlap of the strips at the boundaries. Because the diameter of each disk is 1, no disk can intersect a strip boundary for more than one value of a. Hence $\sum_{a=0}^{k-1}(|VS^*_a| - |VS^*|) \leq |VS^*|$. But then there must be a value of a such that $|VS^*_a| - |VS^*| \leq |VS^*|/k$ and thus $|VS^*_a| \leq (1 + \frac{1}{k})|VS^*|$.

Now we observe that, since VS_a is a minimum vertex cover for each of the strips in D_a, $|VS_a| \leq |VS^*_a|$. This holds for any a, and for the value of a for which $|VS^*_a| \leq (1 + \frac{1}{k})|VS^*|$ in particular. Hence, for this value of a, we have $|VS_a| \leq |VS^*_a| \leq (1 + \frac{1}{k})|VS^*|$. Then by definition also $|VS_{min}| \leq (1 + \frac{1}{k})|VS^*|$. □

For Minimum Dominating Set, we note that if a disk center c_i is outside the strip, but intersects a disk in the strip, $v(c_i)$ can only be used as a dominator, and as such does not need to be dominated. We can easily change Algorithm 2.2 for Minimum Dominating Set on unit disk graphs of bounded thickness to take this into account. Now let DS_a be the union of the dominating sets for the strips of D_a computed by this changed algorithm. Clearly, DS_a is a dominating set for D. Furthermore, let DS_{min} be such that $|DS_{min}| = \min_{0 \leq a \leq k-1} |DS_a|$.

Lemma 3.14 *DS_{min} is at least a $(1 + \frac{2}{k})$-approximation of a minimum dominating set of D.*

Proof: We observe that no disk (c_i, r_i) can be within a distance of $2r_i = 1$ of a strip boundary for more than two values of a. We can then apply similar arguments as in the proof of Lemma 3.13 to prove an approximation factor of $(1 + \frac{2}{k})$. □

We now consider the running time of both schemes.

Lemma 3.15 *VS_{min} (DS_{min}) can be computed in $O(kn^2 f(n) 2^{6f(n)} + kn \log n)$ time.*

Proof: We use exactly the same algorithms and arguments as in Lemma 3.10. For Minimum Vertex Cover and Minimum Dominating Set however, we spend $O(f(n)^2 2^{6f(n)} n)$ time for each strip. This brings the total running time to $O(kn^2 f(n) 2^{6f(n)} + kn \log n)$ time. □

We again choose $f(n)$ appropriately to get a running time polynomial in n and k.

Corollary 3.16 *If we choose $f(n) = \frac{1}{6} \log n$, then VS_{min} (DS_{min}) can be computed in time polynomial in n and k.*

Proof: If $f(n) = \frac{1}{6} \log n$, then according to Lemma 3.15, VS_{min} (DS_{min}) can be computed in $O(kn^3 \log n + kn \log n)$, which is $O(kn^3 \log n)$. This is polynomial in n and k. □

We observe that if $f(n) = \frac{1}{6} \log n$, then we can obtain a $(1 - \epsilon)$-approximation with $\frac{6d}{\log n}$ as the lowest possible value of ϵ for a given n. As before however, if $d = d(n) = o(\log n)$, we have an FPTAS$^\infty$.

Theorem 3.17 *There exists an FPTAS$^\infty$ for the minimum vertex cover and minimum dominating set problems on unit disk graphs of bounded density, i.e. of density $d = d(n) = o(\log n)$.*

3.2.4 Minimum Connected Dominating Set

The minimum connected dominating set problem is a much harder problem than the problems considered thusfar, because the set of vertices in a solution must not only dominate all vertices in the graph, but the subgraph induced by the solution vertices must also be connected. Before we give a description of the algorithm, we prove the property of a (connected) dominating set that we will exploit in the analysis later on.

Proposition 3.18 *Let $G = (V, E)$ be a connected graph and S an arbitrary dominating set of G. If $G[S]$ has n_{cc} connected components, then there exists a connected dominating set for G of size at most $|S| + 2n_{cc} - 2$.*

Proof: We prove this by induction on the number of connected components of $G[S]$. If $n_{cc} = 1$, then $G[S]$ is connected and S is a connected dominating set of size $|S|+2n_{cc}-2 = |S|$. So assume $n_{cc} > 1$ and for any dominating set S' such that $G[S']$ has at most $n_{cc}-1$ connected components, there exists a connected dominating set of size at most $|S'| + 2n_{cc} - 4$. Because S is a dominating set, there must be two vertices $v, w \in S$ such that v and w are in different connected components of $G[S]$ and there is a path P from v to w in $G[V - S \cup \{v, w\}]$ containing at most two vertices. Then we can construct dominating set $S'' = S \cup P$. Observe that S'' is a dominating set of G and has at most $n_{cc} - 1$ connected components. From the induction hypotheses, we know that there exists a connected dominating set for G of size at most $|S''| + 2n_{cc} - 4$. This is at most $(|S| + 2) + 2n_{cc} - 4 = |S| + 2n_{cc} - 2$. Using induction, we prove the proposition. □

The FPTAS$^\infty$ uses a strip partitioning similar to the one used with the minimum dominating set problem. There we observed that a dominating set for the disks of a strip can contain any disk outside the strip that intersects a disk in the strip. Hence we added this extra set of disks to the disks of a strip. For Minimum Connected Dominating Set, we must add the extra set of disks to ensure connectivity (see Lemma 3.19). To prove a relation between the minimum connected dominating set and the computed approximation, we also add the disks intersecting the extra set (see Lemma 3.20). This implies that we add disks within distance 2 of a strip boundary to the strip. In other words, given a strip boundary $y = j$ ($j \in \mathbb{Z}$), a disk (c_i, r_i) will be in the strip above the boundary if $j - 4r_i \le T_g(c_i)^y$ and in the strip below the boundary if $T_g(c_i)^y < j + 4r_i$ (see Figure 10).

Now let $f(n) \ge 4d$ be some function and let k be an integer ($3 < k \le \frac{f(n)}{d}$). Denote by D_a ($0 \le a \le k - 1$) the set of subsets of D obtained by partitioning D as described above, where the strip boundaries are shifted down by a. We denote the set of disks in the b-th strip of D_a by D_a^b ($b \in \mathbb{Z}$).

We introduce the following names and notation to denote relevant pieces of a strip. Let $y = j$ and $y = l$ ($j, l \in \mathbb{Z}, j < l$) be the two strip boundaries defining an arbitrary strip in D_a. We call the area between $y = l + 1$ and $y = j - 1$ the *interior* of the strip. The area between $y = l + 2$ and $y = l + 1$ is called the *upper exterior* and the area between $y = j - 1$ and $y = j - 2$ the *lower exterior* of the strip. The combination of the upper and the lower exterior of a strip is simply referred to as the *exterior*. The area between $y = l + 1$ and $y = l - 1$ is called the *upper boundary area* and the area between $y = j - 1$ and $y = j + 1$ is the *lower boundary area*. Both boundary areas consist of an upper and a lower part (see Figure 11).

Observe that the thickness of any strip in D_0 (including the disks in the boundary area and the exterior of the strip) is at most $f(n) + 5d$. Hence the thickness of any strip in D_a is

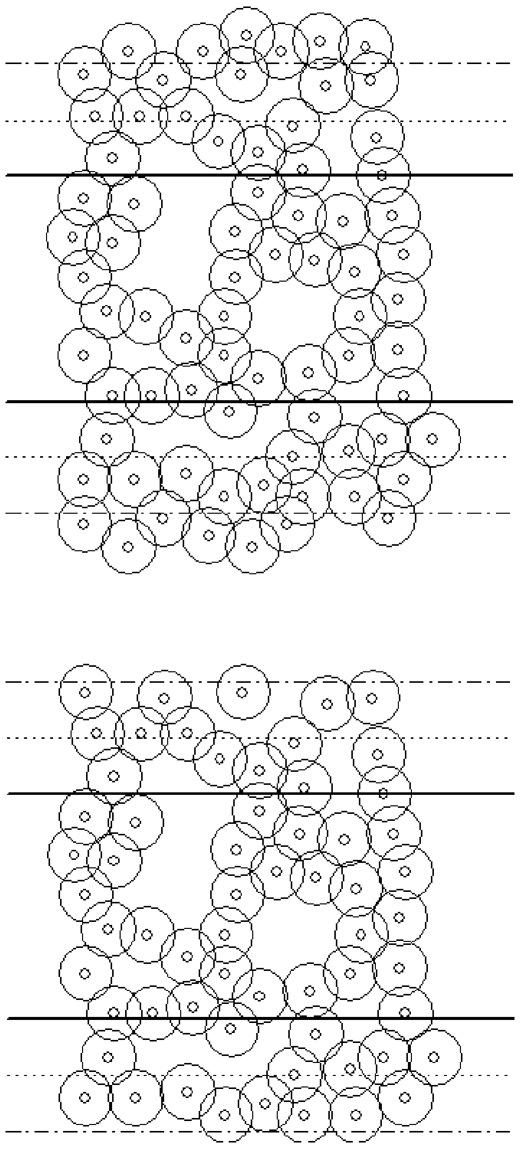

Figure 10: Above is an arbitrary set of disks and one strip. The dotted lines are at distance 1 from the strip boundaries, and the dotted-striped lines at distance 2. Below, we show the disks in the strip for the minimum connected dominating set problem. The disk center of a disk must be between the dotted-striped lines for the disk to be in the strip.

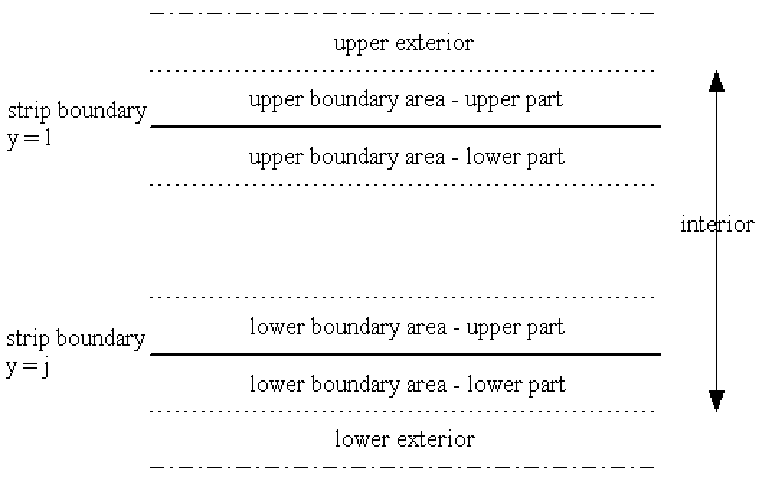
upper exterior
strip boundary y=1
upper boundary area - upper part
upper boundary area - lower part
interior
strip boundary y=j
lower boundary area - upper part
lower boundary area - lower part
lower exterior

Figure 11: The different pieces of a strip and their names.

at most $f(n) + (a+5)d$. Since $a \leq k-1 \leq \frac{f(n)}{d} - 1$, the thickness of a strip is at most

$$f(n) + \left(\frac{f(n)}{d} - 1 + 5\right) d = 2f(n) + 4d \leq 3f(n).$$

Using Lemma 2.23 and the fact that the thickness of any strip in D_a is at most $3f(n)$, we can compute a minimum connected dominating set for each connected component in the interior of each strip in D_a in $O(f(n)^2\, 2^{12f(n)}\, n)$ time. Each such minimum connected dominating set may use the vertices in the exterior of the strip.

Denote the union of the minimum connected dominating sets of each connected component in the b-th strip of D_a by CDS_a^b ($b \in \mathbb{Z}$) and let $CDS_a = \bigcup_{b\in\mathbb{Z}} CDS_a^b$. We claim that CDS_a is a connected dominating set for G.

Lemma 3.19 *For each a ($0 \leq a \leq k-1$), CDS_a is a connected dominating set for G.*

Proof: Trivially, CDS_a is a dominating set for G. It remains to prove the connectivity of CDS_a. So assume CDS_a is not connected. Consider an arbitrary pair of vertices $v, w \in CDS_a$, such that v and w are in two different connected components X and Y of CDS_a and there is a path P from v to w in $G[V - CDS_a \cup \{v,w\}]$ containing at most two vertices. At least one such pair of vertices must exist, because CDS_a is a dominating set and (by assumption) has at least two connected components.

Suppose v and w are in the interior same strip, i.e. $v, w \in \text{interior}(D_a^b)$ for some $b \in \mathbb{Z}$. If $P \not\subseteq \text{interior}(D_a^b)$, then, as P connects v and w and contains at most two vertices, v and w must be in the boundary area of D_a^b and the vertices of P are in the exterior or the boundary area of D_a^b. But then either $(\{v,w\} \cup P) \subseteq \text{interior}(D_a^{b+1})$ or $(\{v,w\} \cup P) \subseteq \text{interior}(D_a^{b-1})$. So without loss of generality, we can also assume that $P \subseteq \text{interior}(D_a^b)$. Then there must be a connected component of $\text{interior}(D_a^b)$ containing v and w (and P). Because CDS_a contains a connected dominating set for each connected component of $\text{interior}(D_a^b)$, v and w must be connected in CDS_a. But then X and Y are connected, which is a contradiction.

Therefore v and w must be in the interiors of different, but neighboring strips, i.e. $v \in \text{interior}(D_a^b)$ and (without loss of generality) $w \in \text{interior}(D_a^{b-1})$. This implies that P is in the shared boundary area of D_a^b and D_a^{b-1} and contains exactly two vertices (see Figure 12). Let

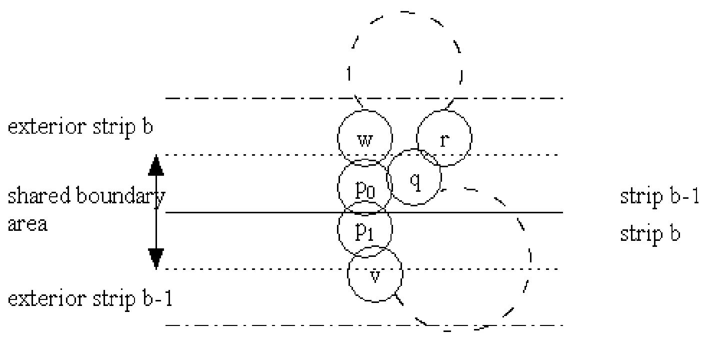

Figure 12: The thick line is the boundary between strips b and $b-1$. If v and w are in the interiors of different strips, then P must intersect the boundary area and consist of exactly two vertices, denoted p_0 and p_1. Vertex p_0 is in the upper part of the shared boundary area and is thus connected to w.

p_0 be the vertex of P in the upper part of the shared boundary area. Observe that p_0 and v are in the same connected component C of interior(D_a^b). Because CDS_a contains a connected dominating set for each connected component of interior(D_a^b), p_0 must be dominated by a vertex $q \in D_a^b$, such that q and v are connected in CDS_a.

Since p_0 is in the upper part of the shared boundary area, q must be either in the exterior of D_a^b or in the shared boundary area. Either way, $q \in$ interior(D_a^{b-1}). Furthermore, q and w are in the same connected component of interior(D_a^{b-1}). As CDS_a contains a connected dominating set of each connected component of interior(D_a^{b-1}), there must be a vertex $r \in D_a^{b-1}$ dominating q, such that r and w are connected in CDS_a. But then v and w are connected in CDS_a (via q and r). This is a contradiction. Hence CDS_a must be connected and thus a connected dominating set. $\square$

Next we link the size of CDS_a^b to the size of the minimum connected dominating set.

Lemma 3.20 *Let OPT be a minimum connected dominating set for G. Then for each a ($0 \leq a \leq k-1$) and for each strip b, $|CDS_a^b| \leq |OPT \cap D_a^b| + 2|OPT \cap \text{exterior}(D_a^b)|$.*

Proof: Let $OPT_a^b = OPT \cap D_a^b$. Clearly, OPT_a^b is a dominating set for the interior of D_a^b. However, $G[OPT_a^b]$ may consist of several connected components. We will show that by adding $2|OPT \cap \text{exterior}(D_a^b)|$ vertices, we can connect these components, such that we obtain a connected dominating set for each connected component in the interior of D_a^b, possibly using vertices in the exterior of D_a^b.

Consider some connected component C of D_a^b. Because OPT is connected, it must hold for each connected component X of $OPT_a^b \cap C$ that $X \cap \text{exterior}(D_a^b) \cap C \neq \emptyset$. Hence the number of connected components of $OPT_a^b \cap C$ is at most $|OPT \cap \text{exterior}(D_a^b) \cap C|$. Since $OPT_a^b \cap C$ is a dominating set for interior$(D_a^b) \cap C$, it follows from Proposition 3.18 that there exists a connected dominating set of size at most $|OPT_a^b \cap C| + 2|OPT \cap \text{exterior}(D_a^b) \cap C|$ for interior$(D_a^b) \cap C$. This holds for all connected components of D_a^b.

As the connected components of D_a^b are mutually exclusive, there exists a connected dominating set of size at most $|OPT_a^b| + 2|OPT \cap \text{exterior}(D_a^b)|$ for each connected component in the interior of D_a^b. This connected dominating set possibly uses vertices in the exterior of D_a^b. Because CDS_a^b is a *minimum* connected dominating set for the each connected component in the interior of D_a^b, possibly using vertices in the exterior of D_a^b, $|CDS_a^b| \leq |OPT \cap D_a^b| + 2|OPT \cap \text{exterior}(D_a^b)|$. $\square$

Let CDS_{min} be such that $|CDS_{min}| = \min_{0 \le a \le k-1} |CDS_a|$. Then we can prove the following lemma.

Lemma 3.21 *CDS_{min} is at least a $(1 + \frac{8}{k})$-approximation of a minimum connected dominating set of D.*

Proof: Let OPT be a minimum connected dominating set of D. Then

$$\begin{aligned} k\,|CDS_{min}| &\le \textstyle\sum_{a=0}^{k-1} |CDS_a| \\ &\le \textstyle\sum_{a=0}^{k-1} \sum_b |CDS_a^b| \\ &\le \textstyle\sum_{a=0}^{k-1} \sum_b |OPT \cap D_a^b| + 2|OPT \cap \text{exterior}(D_a^b)| \\ &= \Big(\textstyle\sum_{a=0}^{k-1} \sum_b |OPT \cap D_a^b|\Big) + \Big(2 \textstyle\sum_{a=0}^{k-1} \sum_b |OPT \cap \text{exterior}(D_a^b)|\Big). \end{aligned}$$

Observe that no disk can be within distance 2 of a strip boundary for more than four values of a. Hence $\sum_{a=0}^{k-1} \sum_b |OPT \cap D_a^b| \le (k+4)|OPT|$. Similary, no disk can be in the exterior of a strip for more than two values of a. Therefore $\sum_{a=0}^{k-1} \sum_b |OPT \cap \text{exterior}(D_a^b)| \le 2|OPT|$. Then

$$\begin{aligned} k\,|CDS_{min}| &\le \Big(\textstyle\sum_{a=0}^{k-1} \sum_b |OPT \cap D_a^b|\Big) + \Big(2 \textstyle\sum_{a=0}^{k-1} \sum_b |OPT \cap \text{exterior}(D_a^b)|\Big) \\ &\le (k+4)|OPT| + 2(2|OPT|) \\ &= (k+8)|OPT|. \end{aligned}$$

Hence $|CDS_{min}| \le (1 + \frac{8}{k})|OPT|$. The lemma follows. □

Lemma 3.22 *CDS_{min} can be computed in $O(kn^2 f(n)\, 2^{12f(n)} + kn \log n)$ time.*

Proof: We use the same algorithms and arguments as in Lemma 3.10. We spend at most $O(f(n)^2\, 2^{12f(n)}\, n^2)$ time for each strip. Hence the total running time is $O(kn^2 f(n)\, 2^{12f(n)} + kn \log n)$. □

We again choose $f(n)$ appropriately to get a running time polynomial in n and k.

Lemma 3.23 *If we choose $f(n) = \frac{1}{12} \log n$, then CDS_{min} can be computed in time polynomial in n and k.*

Proof: If $f(n) = \frac{1}{12} \log n$, then according to Lemma 3.22, CDS_{min} can be computed in $O(kn^3 \log n + kn \log n)$, which is $O(kn^3 \log n)$. This is polynomial in n and k. □

If $f(n) = \frac{1}{12} \log n$, then an obtain a $(1 - \epsilon)$ approximation of the minimum connected dominating set problem with $\frac{96d}{\log n}$ as the lowest possible value of ϵ for a given n. If however $d = d(n) = o(\log n)$, we have an FPTAS$^\infty$.

Theorem 3.24 *There exists an FPTAS$^\infty$ for the minimum connected dominating set problem on unit disk graphs of bounded density, i.e. of density $d = d(n) = o(\log n)$.*

Now we again consider the results by Cheng *et al.* [15] and Demaine and Hajiaghayi [18]. The basic idea of their results is to obtain some partitioning of the graph into mutually exclusive pieces and then apply the shifting technique. For Cheng *et al.*, these pieces are squares in the plane, while Demaine and Hajiaghayi use the adjacent layers of a breadth-first search tree. The essential ingredient in both results is that, to prove a relation between the minimum connected dominating set restricted to a piece of the graph and the optimum solution computed by the algorithm for each piece, extra vertices are necessary. Cheng *et al.* increase the size of the square by $\Theta(\log n)$ and Demaine and Hajiaghayi add $\Theta(\log n)$ extra layers to each piece. Because of these extra vertices, the algorithm of Cheng *et al.* has a running time of $O(n^{O(\frac{1}{\epsilon}\log\frac{1}{\epsilon})})$ and Demaine and Hajiaghayi obtain a $O(n^{O(\frac{1}{\epsilon}\log\frac{1}{\epsilon}\log\log n)})$ running time for minor-closed graphs of locally bounded treewidth and $O(n^{O(\frac{1}{\epsilon})})$ for planar graphs.

What our analysis has shown (in Lemma 3.20), is that this extra $\log n$ factor is in fact unnecessary. We can easily prove lemma's similar to Lemma 3.19 and 3.20 for minor-closed graphs of locally bounded treewidth, planar graphs, and general unit disk graphs. Such lemma's would provide an instant improvement to the running times of the Cheng *et al.* and Demaine and Hajiaghayi algorithms. More importantly, the almost-PTAS for minor-closed graphs of locally bounded treewidth and the PTAS for planar graphs by Demaine and Hajiaghayi improve to an FPTAS$^\infty$.

4 Discussion

The idea of using a fixed-parameter tractable problem on graphs with a special property to construct an FPTAS$^\infty$ for more general graphs has been considered before. Hunt *et al.* [26] consider unit disk graphs of λ-precision, meaning that the distance between any two disk centers is at least λ. Hunt *et al.* show that in such a graph, a slab of height k has bounded treewidth. Using a dynamic programming argument, a tree decomposition of bounded width can be constructed. Combined with the shifting technique, Hunt *et al.* give FPTAS$^\infty$s for Maximum Independent Set, Minimum Vertex Cover, and Minimum Dominating Set. Minimum Connected Dominating Set was not considered. We observe that any unit disk graph of λ-precision has density $\Theta(\frac{1}{\lambda^2})$. The reverse is not necessarily true. Hence our results are a generalization of the results by Hunt *et al.* Furthermore, we have shown that using a tree decomposition is not necessary, and we can in fact suffice with a much simpler path decomposition.

Even before the results of Hunt *et al.*, Baker [6] proposed FPTAS$^\infty$s for optimization problems on planar graphs. Baker used the observation that a planar graph can be build from several k-outerplanar pieces. It is well known that k-outerplanar graphs have treewidth at most $3k-1$. Hence by applying the shifting technique to obtain k-outerplanar pieces of a planar graph, an FPTAS$^\infty$ can be constructed for Maximum Independent Set, Minimum Vertex Cover, Minimum Dominating Set, and other problems. Baker did not give a solution for Minimum Connected Dominating Set.

Demaine and Hajiaghayi [18] recently provided a deep investigation of the connection between fixed-parameter tractable problems and FPTAS$^\infty$s and PTASs. They show that a whole range of problems (so called bidimensional problems) possessing specific properties can be solved on minor-closed graphs of locally bounded treewidth. The problems that possess these properties include Maximum Independent Set, Minimum Vertex Cover, and Minimum Dominating Set. Demaine and Hajiaghayi give FPTAS$^\infty$s for these problems. Because planar graphs are minor-closed and have locally bounded treewidth, this approach is a generalization of the results by Baker. In additition to the FPTAS$^\infty$s, Demaine and Hajiaghayi also give an almost-PTAS for Minimum Connected Dominating Set on minor-closed graphs of locally bounded treewidth and a PTAS for that problem on planar graphs.

The results of Demaine and Hajiaghayi can unfortunately not be applied to unit disk graphs of bounded density. As we will show, such graphs have locally bounded treewidth, but they are not minor-closed.

First we define the notions of locally bounded treewidth and minor-closed graphs.

Definition 4.1 *Consider some graph $G = (V, E)$ and let $(u, v) \in E$. If we* contract *edge $(u, v) \in E$, we remove u and v from the graph, add a new vertex uv and connect it to all neighbors of u and v in the original graph. Let G' be a graph obtained from G by performing some number of edge contractions. Then G' is a* minor *of G.*

Definition 4.2 *Let $\mathcal{C}$ be a class of graphs. Then $\mathcal{C}$ is said to be* minor-closed *if and only if for each minor G' of a graph $G \in \mathcal{C}$, $G' \in \mathcal{C}$.*

A good example of a class of graphs that is minor-closed are the planar graphs.

Definition 4.3 *The* r-neighborhood *of a vertex $v \in V$, denoted by $N^r(v)$, is defined as the set of vertices reachable from v in at most r steps. The* local treewidth *$ltw^r(v)$ of a vertex v is the treewidth of $N^r(v)$. The* local treewidth *$ltw^r(G)$ of a graph $G = (V, E)$ is $\max_{v \in V} ltw^r(v)$. A graph G has* locally bounded treewidth *if and only if $ltw^r(G) \leq f(r)$, for some function $f(r)$.*

The function $f(r)$ can be arbitrary. There are however several graph classes for which $f(r)$ is linear in r, i.e. they have *linear local treewidth.* This includes planar graphs, bounded-genus graphs, single-crossing-minor-free graphs, and apex-minor-free graphs[3] [17]. Note that these classes are all minor-closed.

We now add another type of graphs to the list and show that unit disk graphs of bounded density have locally bounded treewidth.

Lemma 4.4 *A unit disk graph of bounded density has locally bounded treewidth.*

Proof: Consider an arbitrary unit disk graph of bounded density d. Let v be an arbitrary vertex of that graph. Then all disks in the r-neighborhood of D_v lie within an $2r+1$-by-$2r+1$ square. The number of disks in this square is at most $(2r + 1)^2 d$. This implies a bound of $(2r + 1)^2 d$ on the treewidth. This bound can be improved by using the sliding slab argument of Lemma 2.10 on the square. Then we obtain a bound of $(2r + 1)d$ on the pathwidth of the disks in the square. This implies a bound of $(2r + 1)d$ on the local treewidth. □

Hence unit disk graphs of bounded density have linear local treewidth. However, unit disk graphs of bounded density are not minor-closed.

Lemma 4.5 *The $K_{3,2}$ is not a unit disk graph.*

Proof: We prove this by contradiction. So suppose we have some set of unit disks realizing the $K_{3,2}$. Denote the three mutually independent disks by A, B, and C, and the other two disks by X and Y.

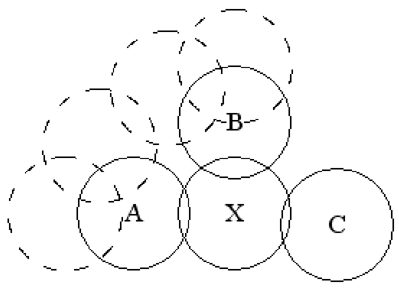

Figure 13: The dotted disks indicate possible positions of disk Y when positioned on or between lines $\overline{BX}$ and line $\overline{AX}$. Disk X shields Y from intersecting the third disk C.

Without loss of generality, we consider disk B and assume A is the first disk of $\{A, C\}$ we encounter when rotating line $\overline{BX}$ counter-clockwise around the center of X and C the second (see Figure 13).

Because X and Y are independent, the distance between their centers must be larger than 1. If Y is on line $\overline{BX}$ and on the same side of X as B, then Y cannot intersect A or C, because B does not intersect A or C. Similarly, if Y is on line $\overline{AX}$ and on the same side of X as A, then Y cannot intersect B or C, because A does not intersect B or C. Hence if Y is between line $\overline{BX}$ and line $\overline{AX}$, Y cannot intersect C. Intuitively, X shields Y from intersecting C (see Figure 13).

Using similar arguments, we can prove that if Y is positioned on or between $\overline{AX}$ and $\overline{CX}$, then Y cannot intersect B and if Y is positioned on or between $\overline{BX}$ and $\overline{CX}$, then Y cannot intersect A. Because the union all the considered areas contains all possible positions for Y, we know that Y cannot intersect A, B, and C simultaneously. However, in the $K_{3,2}$ disk representation, Y does. This is a contradiction. Therefore there exists no disk representation realizing the $K_{3,2}$. □

From this proof, the following corollary follows straightforwardly.

Corollary 4.6 *The $K_{3,x}$ is not a unit disk graph, for each $x \geq 2$.*

Using Lemma 4.5, we can prove that the class of unit disk graphs is not minor-closed.

Lemma 4.7 *The class of unit disk graphs is not minor closed.*

Proof: Construct a $K_{2,2}$ unit disk graph and connect two non-adjacent disks with a path P (see Figure 14). This graph has a $K_{3,2}$ minor, which can be obtained by contracting the edges on P until P contains only one vertex. However, following Lemma 4.5, the $K_{3,2}$ is not in the class of unit disk graphs. Hence the class of unit disk graphs is not minor-closed. □

By looking closely at the set of disks in Figure 14, we can observe that this set has density 1. Therefore the unit disk graphs of bounded density are not minor-closed as well.

Lemma 4.8 *The unit disk graphs of bounded density d are not minor-closed, for each $d \geq 1$.*

Because the unit disk graphs of bounded density are not minor-closed, the results of Demaine and Hajiaghayi [18] cannot be applied directly to these graphs.

[3]A single crossing graph can be drawn in the plane with at most one crossing. An apex graph is a graph where the removal of one vertex leaves a planar graph.

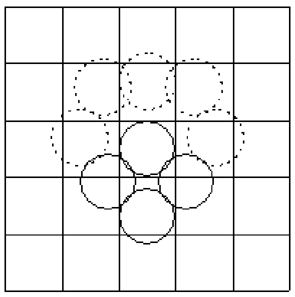

Figure 14: A disk representation realizing a $K_{2,2}$ where two non-adjacent vertices are connected by a path P. The disks of this path are dotted.

5 Conclusion

This paper has shown that Maximum Independent Set, Minimum Vertex Cover, and Minimum (Connected) Dominating Set on unit disk graphs are fixed-parameter tractable in the thickness of the graph. Such graphs have pathwidth bounded by the thickness. This allows known algorithms for path decomposition of bounded width to be applied. Furthermore, we have given algorithms that work directly on the used slab decompositions. This improves the running time of the algorithms for unit disk graphs of bounded thickness. In particular for Minimum Connected Dominating Set, we can improve from $O(t^2(2t)^{2t}n)$ to $O(t^2 2^{4t}n)$.

We then used these algorithms to construct asymptotic FPTASs for the considered problems on unit disk graphs. This under the realistic assumption of bounded density. These results are optimal, in the sense no FPTAS exists for Maximum Independent Set, Minimum Vertex Cover, or Minimum (Connected) Dominating Set on unit disk graphs of bounded density (unless P=NP). Furthermore, the schemes are an improvement over existing approximation algorithms if the density is bounded. The scheme for Minimum Connected Dominating Set is (to our knowledge) the first asymptotic FPTAS for this problem. The analysis to obtain this result can be extended to unit disk graphs, planar graphs, and minor-closed graphs of locally bounded treewidth. Combined with the results from Cheng *et al.* [15] and Demaine and Hajiaghayi [18], we can obtain improved time bounds for the existing PTAS on unit disk graphs and new asymptotic FPTASs for planar graphs and minor-closed graphs of locally bounded treewidth.

Early results seem to indicate that an asymptotic FPTAS also exists for Maximum Independent Set on disk graphs of bounded density. Future work is aimed at strengthening these results and applying the new schemes to several problems not directly related to mobile ad hoc networks.

Acknowledgements

The author would like to thank dr. Hans Bodlaender for many helpful suggestions and discussions.

References

[1] Agarwal, P.K., Kreveld, M. van, Suri, S., "Label Placement by Maximum Independent Set in Rectangles", *Computational Geometry: Theory and Applications* **11** 3-4 (1998), pp. 209–218. Cited on p. 4.

[2] Agarwal, P.K., Mustafa, N.H., "Independent Set of Intersection Graphs of Convex Objects in 2D" in Hagerup, T., Katajainen, J. (eds.) *Proceedings of the 9th Scandinavian Workshop on Algorithm Theory (SWAT 2004)*, Lecture Notes in Computer Science **3111**, Springer-Verlag, Berlin, 2004, pp. 127–137. Cited on p. 4.

[3] Alber, J., Niedermeier, R., "Improved Tree Decomposition Based Algorithms for Domination-like Problems" in Rajsbaum, S. (ed.) *Proceedings of the 5th Latin American Symposium on Theoretical Informatics*, Lecture Notes in Computer Science **2286**, Springer-Verlag, Berlin, 2002, pp. 613–628. Cited on p. 10.

[4] Arora, S., "Polynomial Time Approximation Schemes for Euclidean Traveling Salesman and Other Geometric Problems", *Journal of the ACM* **45** 5 (1998), pp. 753–782. Cited on p. 4.

[5] Ausiello, G., Creszenzi, P., Gambosi, G., Kann, V., Marchetti-Spaccamela, A., Protasi, M., *Complexity and Approximation - Combinatorial Optimzation Problems and Their Approximability*, Springer-Verlag, Berlin, 1999. Cited on p. 24.

[6] Baker, B.S., "Approximation Algorithms for NP-Complete Problems on Planar Graphs", *Journal of the ACM* **41** 1 (1994), pp. 153-180. Cited on p. 4, 5, 37.

[7] Becker, H.W., "Rooks and Rhymes", *Mathematics Magazine* **22** 1 (September – October 1948), pp. 23–26. Cited on p. 17.

[8] Becker, H.W. "Planar Rhyme Schemes", *Bulletin of the American Mathematical Society* **58** (1952), pp. 39. Cited on p. 17.

[9] Bell, E.T., "Exponential Polynomials", *The Annals of Mathematics* **35** 2 (April 1934), pp. 258–277. Cited on p. 17.

[10] Bell, E.T., "The Iterated Exponential Integers", *The Annals of Mathematics* **39** 3 (July 1938), pp. 539–557. Cited on p. 17.

[11] Bodlaender, H.L., "A Tourist Guide through Treewidth", *Acta Cybernetica* **11** 1–2 (1993), pp. 1–22. Cited on p. 10.

[12] Butenko, S., Cheng, X., Oliviera, C.A.S., Pardalos, P.M., "A New Heuristic for the Minimum Connected Dominating Set Problem on Ad Hoc Wireless Networks", to appear in *Cooperative Control and Optimization*, Kluwer Academic Publishers, 2004, pp. 61–73. Cited on p. 5.

[13] Cardei, M., Cheng, X., Cheng, X., Du, D.-Z., "Connected Domination in Multihop Ad Hoc Wireless Networks" in Caulfield, H.J., Chen, S.-H., Cheng, H.-D., Duro, R.J., Honavar, V., Kerre, E.E., Lu, M., Romay, M.G., Shih, T.K., Ventura, D., Wang, P.P., Yang, Y. (eds.) *Proceedings of the 6th Joint Conference on Information Science (JCIS 2002)*, JCIS / Association for Intelligent Machinery, 2002, pp. 251–255. Cited on p. 5.

[14] Chan, T.M., "Polynomial-time Approximation Schemes for Packing and Piercing Fat Objects", *Journal of Algorithms* **46** 2 (February 2003), pp. 178–189. Cited on p. 4.

[15] Cheng, X., Huang, X., Li, D., Wu, W., Du, D.-Z., "A Polynomial-Time Approximation Scheme for the Minimum Connected Dominating Set in Ad Hoc Wireless Networks", *Networks* **42** 4 (December 2003), pp. 202–208. Cited on p. 5, 36, 40.

[16] Clark, B.N., Colbourn, C.J., Johnson, D.S., "Unit Disk Graphs", *Discrete Mathematics* **86** 1–3 (1990), pp. 165–177. Cited on p. 4, 24.

[17] Demaine, E., Hajiaghayi, "Equivalence of Local Treewidth and Linear Local Treewidth and its Algorithmic Applications" in: *Proceedings of the 15th ACM-SIAM Symposium on Discrete Algorithms (SODA 2004)*, Society for Industrial and Applied Mathematics, 2004, pp. 840–849. Cited on p. 38.

[18] Demaine, E.D., Hajiaghayi, M., "Bidimensionality: New Connections between FPT Algorithms and PTASs" in: *Proceedings of the 16th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2005)*, to appear. Cited on p. 5, 10, 36, 37, 39, 40.

[19] Downey, R.G., Fellows, M.R., *Parameterized Complexity*, Springer-Verlag, New York, 1999. Cited on p. 3.

[20] Eppstein, D., "Separating Thickness from Geometric Thickness" in Goodrich, M.T., Kobourov, S.G. (eds.) *Proceedings of the 10th International Symposium on Graph Drawing (GD 2002)*, Lecture Notes in Computer Science **2528**, Springer-Verlag, Berlin, 2002, pp. 150–161. Cited on p. 5.

[21] Erlebach, T., Jansen, K., Seidel, E., "Polynomial-time Approximation Schemes for Geometric Graphs" in *Proceedings of the 12th ACM–SIAM Symposium on Discrete Algorithms (SODA 2001)*, Society for Industrial and Applied Mathematics, 2001, pp. 671–679. Cited on p. 4.

[22] Garey, M.R., Johnson, D.S., *Computers and Intractability, A Guide to the Theory of NP-Completeness*, W.H. Freeman and Company, San Francisco, 1979. Cited on p. 3, 4.

[23] Glaßer, C., Reith, S., Vollmer, H., "The Complexity of Base Station Positioning in Cellular Networks" in Rolim, J.D.P., Broder, A.Z., Corradini, A., Gorrieri, R., Heckel, R., Hromkovic, J., Vaccaro, U., Wells, J.B. (eds.) *Proceedings of the ICALP 2000 Satellite Workshops*, Workshop on Approximation and Randomized Algorithms in Communication Networks, Carleton Scientific, Waterloo, Canada, 2000, pp. 167–178. Cited on p. 4.

[24] Harary, F., *Graph Theory*, Addison-Wesley Publishing Company, Reading, Massachusetts, 1969. Cited on p. 5.

[25] Hochbaum, D.S., Maass, W., "Approximation Schemes for Covering and Packing Problems in Image Processing and VLSI", *Journal of the ACM* **32** 1 (1985), pp. 130–136. Cited on p. 4.

[26] Hunt III, D.B., Marathe, M.V., Radhakrishnan, V., Ravi, S.S., Rosenkrantz, D.J., Stearns, R.E., "NC-Approximation Schemes for NP- and PSPACE-Hard Problems for

Geometric Graphs", *Journal of Algorithms* **26** 2 (1998), pp. 238–274. Cited on p. 4, 5, 37.

[27] Kreveld, M. van, Strijk, T., Wolff, A., "Point Labeling with Sliding Labels", *Computational Geometry: Theory and Applications* **13** 1 (1999), pp. 21–47. Cited on p. 4.

[28] Kreweras, G., "Sur les Partitions Non Croisées d'un Cycle", *Discrete Mathematics* **1** 4 (February 1972), pp. 333–350. Cited on p. 17.

[29] Leeuwen, E.J. van, *Optimization Problems on Mobile Ad Hoc Networks – Algorithms for Disk Graphs*, Master's Thesis INF/SCR-04-32, Institute of Information and Computing Sciences, Utrecht University, 2004. Cited on p. 7, 9, 24, 25.

[30] Li, X.-Y., Wang, Y., "Simple Heuristics and PTASs for Intersection Graphs in Wireless Ad Hoc Networks" in *Proceedings of the 6th International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications (DIALM '02)*, pp. 62–71, ACM Press, 2002. Cited on p. 4.

[31] Lichtenstein, D., "Planar Formulae and Their Uses", *SIAM Journal on Computing* **11** 2 (August 1982), pp. 329–343. Cited on p. 24.

[32] Lovász, L., *Combinatorial Problems and Exercises*, 2nd edition, North-Holland, Amsterdam, 1993. Cited on p. 17.

[33] Malesińska, E., *Graph-Theoretical Models for Frequency Assignment Problems*, PhD Thesis, Technical University of Berlin, Berlin, 1997. Cited on p. 4.

[34] Marathe, M.V., Breu, H., Hunt III, H.B., Ravi, S.S., Rosenkrantz, D.J., "Simple Heuristics for Unit Disk Graphs", *Networks* **25** (1995), pp. 59–68. Cited on p. 4, 5.

[35] Matsui, T., "Approximation Algorithms for Maximum Independent Set Problems and Fractional Coloring Problems on Unit Disk Graphs", in Akiyama, J. Kano, M., Urabe, M. (eds.) *Japan Conference on Discrete and Computational Geometry*, Lecture Notes in Computer Science **1763**, Springer-Verlag, Berlin, 1998, pp. 194–200. Cited on p. 4.

[36] Miller, G.L., Vavasis, S.A., "Density Graphs and Separators" in *Proceedings of the Second Annual ACM–SIAM Symposium on Discrete Algorithms (SODA 1992)*, Society for Industrial and Applied Mathematics, 1992, pp. 331–336. Cited on p. 22.

[37] Nieberg, T., Hurink, J.L., Kern, W., *A New PTAS for Maximum Independent Sets in Unit Disk Graphs*, Memorandum No. 1688, Department of Applied Mathematics, University of Twente, Enschede, September 2003. Cited on p. 4.

[38] Nieberg, T., Hurink, J.L., Kern, W., "A Robust PTAS for Maximum Weight Independent Sets in Unit Disk Graphs" in *Proceedings of the 30th International Workshop on Graph-Theoretic Concepts in Computer Science (WG 2004)*, to appear. Cited on p. 4.

[39] Nieberg, T., Hurink, J.L., *A PTAS for the Minimum Dominating Set Problem in Unit Disk Graphs*, Memorandum No. 1732, Department of Applied Mathematics, University of Twente, Enschede, 2004. Cited on p. 4.

[40] Robertson, N., Seymour, P.D., “Graph Minors. I. Excluding a Forest”, *Journal of Combinatorial Theory, Series B* **35** (1983), pp. 39–61. Cited on p. 8.

[41] Simion, R., “Noncrossing Partitions”, *Discrete Mathematics* **217** 1–3 (28 April 2000), pp. 367–409. Cited on p. 17.

[42] Telle, J.A., Proskurowski, A., “Algorithms for Vertex Partitioning Problems on Partial k-Trees”, *SIAM Journal on Discrete Mathematics* **10** 4 (1997), pp. 529–550. Cited on p. 10.

[43] Wan, P-J., Alzoubi, K.M., Frieder, O., “Distributed Construction of Connected Dominating Set in Wireless Ad Hoc Networks”, *Proceedings of IEEE Infocom 2002, Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies*, 2002, Volume 3, pp. 1597–1604. Cited on p. 5.

[44] Weisstein, E.W., “Bell Number”, *MathWorld*, `http://mathworld.wolfram.com/BellNumber.html`. Cited on p. 17.

[45] Weisstein, E.W. *et al.*, “Catalan Number”, *MathWorld*, `http://mathworld.wolfram.com/CatalanNumber.html`. Cited on p. 17.