Constant-factor approximation for minimum-weight (connected) dominating sets in unit disk graphs

Christoph Ambühl* Thomas Erlebach† Matúš Mihal'ák‡ Marc Nunkesser§

June 30, 2006

Abstract

For a given graph with weighted vertices, the goal of the minimum-weight dominating set problem is to compute a vertex subset of smallest weight such that each vertex of the graph is contained in the subset or has a neighbor in the subset. A unit disk graph is a graph in which each vertex corresponds to a unit disk in the plane and two vertices are adjacent if and only if their disks have a non-empty intersection. We present the first constant-factor approximation algorithm for the minimum-weight dominating set problem in unit disk graphs, a problem motivated by applications in wireless ad-hoc networks. The algorithm is obtained in two steps: First, the problem is reduced to the problem of covering a set of points located in a small square using a minimum-weight set of unit disks. Then, a constant-factor approximation algorithm for the latter problem is obtained using enumeration and dynamic programming techniques exploiting the geometry of unit disks. Furthermore, we show how to obtain a constant-factor approximation algorithm for the minimum-weight connected dominating set problem in unit disk graphs.

Our techniques also yield a constant-factor approximation algorithm for the weighted disk cover problem (covering a set of points in the plane with unit disks of minimum total weight) and a 3-approximation algorithm for the weighted forwarding set problem (covering a set of points in the plane with weighted unit disks whose centers are all contained in a given unit disk).

1 Introduction

The dominating set problem is a classical optimization problem on graphs. For a given undirected graph $G = (V, E)$, a subset $D \subseteq V$ of its vertices is called a *dominating set* if every vertex in V is contained in D or has a neighbor in D. A vertex in D is called a *dominator*. A dominator *dominates* itself and all its neighbors. The goal of the *minimum dominating set problem* (MDS) is to compute a dominating set of smallest size. In the weighted version, the *minimum-weight dominating set problem* (MWDS), each vertex of the input graph is associated with a weight, and the goal is to compute a dominating set of minimum weight.

A dominating set $D \subseteq V$ is called a *connected dominating set* in the graph $G = (V, E)$ if the subgraph induced by D is connected. The minimum connected dominating set problem (MCDS) and minimum-weight connected dominating set problem (MWCDS) are defined in the obvious way.

*Department of Computer Science, University of Liverpool, e-mail: christoph@csc.liv.ac.uk
†Department of Computer Science, University of Leicester, e-mail: te17@mcs.le.ac.uk
‡Department of Computer Science, University of Leicester, e-mail: mm215@mcs.le.ac.uk
§Institute of Theoretical Computer Science, ETH Zürich, e-mail: mnunkess@inf.ethz.ch

For general graphs, MDS (and therefore MWDS) is $\mathcal{NP}$-hard [8]. Furthermore, MDS for general graphs is known to be equivalent to the *set cover* problem, implying that it can be approximated within a factor of $O(\log n)$ for graphs with n vertices using a greedy algorithm (see, e.g., [15]), but no better unless all problems in $\mathcal{NP}$ can be solved in $n^{O(\log \log n)}$ time [7]. Approximation ratio $O(\log n)$ can also be achieved for the weighted set cover problem and thus for MWDS. The best known approximation ratio for MWCDS in general graphs is $O(\log n)$ as well [9].

In this paper, we are concerned with MWDS and MWCDS in a special class of graphs: *unit disk graphs*. A unit disk graph is a graph in which each vertex is associated with a (topologically closed) unit disk in the plane and two vertices are adjacent if and only if the corresponding disks have a non-empty intersection. We are interested in efficient approximation algorithms. An algorithm for MDS (or MWDS) is called a *ρ-approximation algorithm*, and has *approximation ratio* ρ, if it runs in polynomial time and always outputs a dominating set whose size (or total weight) is at most a factor of ρ larger than the size (or total weight) of the optimal solution. The definitions for MCDS and MWCDS are analogous. A *polynomial-time approximation scheme* (PTAS) is a family of approximation algorithms with ratio $1 + \varepsilon$ for every constant $\varepsilon > 0$.

A major motivation for studying (connected) dominating sets in unit disk graphs comes from routing in wireless ad-hoc networks, where dominating sets have been proposed for the construction of routing backbones (see, e.g., [1]). Each node of the graph models a wireless device, and two nodes are connected by an edge if they are close enough to receive each other's transmissions. A message that is broadcast by all nodes of a dominating set will be received by all nodes of the network. Therefore, a small connected dominating set is an energy-efficient routing backbone. Recent work has emphasized that ad-hoc networks are often heterogeneous as different nodes have different capabilities. Therefore it is meaningful to assign weights to the nodes (giving small weight to nodes that have a large remaining battery life, for example) and aim to determine a (connected) dominating set of small weight [16]. Thus, one arrives at the MWDS and MWCDS problems in unit disk graphs.

Clark et al. [6] have proved that MDS is $\mathcal{NP}$-hard for unit disk graphs. Lichtenstein [12] has shown that MCDS is $\mathcal{NP}$-hard for unit disk graphs. Constant-factor approximation algorithms for MDS and MCDS in unit disk graphs were given by Marathe et al. [13]. For MDS in unit disk graphs, a PTAS was presented by Hunt et al. [11], based on the shifting strategy [2, 10]. These algorithms, however, do not extend to the weighted version. In particular, the PTAS is heavily based on the fact that the optimal dominating set for unit disks in a $k \times k$ square has size at most $O(k^2)$ and can thus be found in polynomial time using complete enumeration if k is a constant. In the weighted case, there is no such bound on the size of an optimal (or near-optimal) solution, as an optimal solution may consist of a large number of disks with tiny weight. For MCDS in unit disk graphs, a PTAS was presented in [5]. For the special case of unit disk graphs with bounded density, asymptotic fully polynomial-time approximation schemes (with running time polynomial in $\frac{1}{\varepsilon}$ and in the size of the input, but achieving ratio $1 + \varepsilon$ only for large enough inputs) were presented for MDS and MCDS in [14].

Wang and Li [16] give distributed algorithms for MWDS and MWCDS in unit disk graphs that achieve approximation ratio $O(\min\{\log \Delta, \sigma\})$, where Δ is the maximum degree of the graph and σ is the ratio of the maximum weight to the minimum weight of a disk. Note that these approximation ratios are not better than the known ratios for general graphs in the worst case.

1.1 Our results

In this paper, we present the first constant-factor approximation algorithms for MWDS and MWCDS in unit disk graphs. Our algorithm for MWDS solves the problem in two steps. First, we reduce MWDS in unit disk graphs to the problem of covering a set of points that are located in a small square using a minimum-weight set of unit disks. In the reduction we lose only a constant factor in the approximation ratio. Then, we present a constant-factor approximation algorithm for the latter problem using enumeration and dynamic programming techniques exploiting the geometry of unit disks. To solve the MWCDS problem, we first compute an $O(1)$-approximation for the MWDS problem and then use an approach based on a minimum spanning tree calculation to add disks to the solution in order to make the dominating set connected.

We also show that our techniques yield a constant-factor approximation algorithm for the weighted disk cover problem for unit disks and a 3-approximation algorithm for the special case of the forwarding set problem (see Section 5 for a definition of this problem).

The remainder of the paper is structured as follows. Our top-level approach to solving MWDS, which consists of breaking the problem into subproblems in small squares, is presented in Section 2. In Section 3, we show how the subproblem can be reduced to a special disk cover problem and give a constant-factor approximation algorithm for the latter problem. We also describe how this implies a constant-factor algorithm for the general weighted disk cover problem with unit disks. Section 4 shows how we can make a dominating set connected while incurring a cost that is bounded by a constant factor times the cost of the optimal connected dominating set. In Section 5, we apply our techniques to obtain a 3-approximation algorithm for the forwarding set problem. Finally, we give our conclusions and mention some open problems in Section 6.

2 Algorithm for minimum-weight dominating sets

Let an instance of MWDS in unit disk graphs be given by a set $\mathcal{D}$ of weighted unit disks in the plane. The weight of disk $d \in \mathcal{D}$ is denoted by $w_d \geq 0$. Each disk has radius 1 and is specified by the coordinates of its center. For $U \subseteq \mathcal{D}$, we write $w(U)$ for $\sum_{d \in U} w_d$.

Our algorithm uses a parameter $\mu < 1$; we can set $\mu = 0.999$. We partition the plane into squares of side length μ. The square S_{ij}, for $i, j \in \mathbb{Z}$, contains all points (x, y) with $i\mu \leq x < (i+1)\mu$ and $j\mu \leq y < (j+1)\mu$.

For a square S_{ij} that contains at least one disk center, let $\mathcal{D}_{ij}$ be the set of disks in $\mathcal{D}$ whose center is in S_{ij}. Let $N(\mathcal{D}_{ij})$ denote the set of all disks in $\mathcal{D} \setminus \mathcal{D}_{ij}$ that intersect a disk in $\mathcal{D}_{ij}$. We consider a subproblem to be solved for each square S_{ij} that can be stated as follows: Find a minimum-weight set of disks in $\mathcal{D}_{ij} \cup N(\mathcal{D}_{ij})$ that dominates all disks in $\mathcal{D}_{ij}$. Let OPT_{ij} denote an optimal solution to the subproblem for square S_{ij}. In Section 3, we will present an algorithm that outputs a solution U_{ij} for the subproblem satisfying $w(U_{ij}) \leq 2 \cdot w(\mathrm{OPT}_{ij})$. In the end, we output the union of all sets U_{ij} that we have computed. It is clear that this yields a dominating set.

Theorem 1 *There is a constant-factor approximation algorithm for the minimum weight dominating set problem in unit disk graphs.*

Proof. The algorithm described above outputs a dominating set U of weight at most $\sum w(U_{ij})$. Here and in the following, the summation is over all squares S_{ij} that contain at least one disk center. As we will present a 2-approximation algorithm to solve each subproblem in Section 3, we have $w(U_{ij}) \leq 2 \cdot w(\mathrm{OPT}_{ij})$. Let OPT denote an optimal dominating set for the whole instance. Let $\mathrm{OPT}[S_{ij}] = \mathrm{OPT} \cap (\mathcal{D}_{ij} \cup N(\mathcal{D}_{ij}))$. Note that $\mathrm{OPT}[S_{ij}]$ is a feasible solution to the subproblem for square S_{ij} and therefore we have $w(\mathrm{OPT}_{ij}) \leq w(\mathrm{OPT}[S_{ij}])$.

We get $w(U) \leq \sum w(U_{ij}) \leq 2 \sum w(\mathrm{OPT}_{ij}) \leq 2 \sum w(\mathrm{OPT}[S_{ij}])$. The sum $\sum w(\mathrm{OPT}[S_{ij}])$ adds the costs of solutions $\mathrm{OPT}[S_{ij}]$ for all squares S_{ij} that contain at least one disk center. Note that a disk d in OPT can be in $\mathrm{OPT}[S_{ij}]$ only if its center is in S_{ij} or it intersects a disk with center in

S_{ij}. Therefore, the distance between the center of d and the square S_{ij} is at most 2. Consequently, there are only $O(1/\mu^2)$ squares S_{ij} such that d can be in $\mathrm{OPT}[S_{ij}]$. More precisely, all such squares must be fully contained in a disk of radius $2 + \sqrt{2}\mu$ around the center of d, and for $\mu = 0.999$ that disk can contain at most $\lfloor (2 + \sqrt{2}\mu)^2\pi/\mu^2 \rfloor = 36$ such squares. This means that the number of times each disk in OPT contributes its weight to $\sum w(\mathrm{OPT}[S_{ij}])$ is bounded by 36. We get $\sum w(\mathrm{OPT}[S_{ij}]) \leq 36 \cdot w(\mathrm{OPT})$ and, thus, $w(U) \leq 2 \sum w(\mathrm{OPT}[S_{ij}]) \leq 72 \cdot w(\mathrm{OPT})$. $\square$

3 Solving the subproblem for a small square

In this section we present a 2-approximation algorithm for the following problem: Given a $\mu \times \mu$ square S_{ij}, where $\mu < 1$, and the set of disks $\mathcal{D}_{ij} \cup N(\mathcal{D}_{ij})$, compute a minimum-weight set of disks that dominates all disks in $\mathcal{D}_{ij}$.

Let OPT_{ij} denote the set of disks in an optimal solution for the problem. In the following, we will often write that the algorithm "guesses" certain properties of OPT_{ij}. Such guesses are to be interpreted as follows: The algorithm tries all possible choices for the guess (there will be a polynomial number of such choices) and computes a solution for each choice. In the end, the algorithm outputs the solution of minimum weight among all solutions found in this way. Some guesses may not lead to feasible solutions; such guesses are discarded. In the analysis, we concentrate on the solution in which the algorithm makes the right guess about OPT_{ij}. It then suffices to show that the solution the algorithm finds for that guess is a constant-factor approximation of the optimum, because the solution output by the algorithm in the end will be at least as good as the one it finds for that guess.

First, the algorithm guesses the largest weight w of a disk in OPT_{ij}. Note that there are at most n possible values for this guess (where n is the number of disks in the instance). If there is a disk of weight at most w in $\mathcal{D}_{ij}$, the algorithm simply outputs that disk as the solution (note that the disk has its center in S_{ij} and therefore dominates all other disks in $\mathcal{D}_{ij}$), and this solution is optimal. If there is no disk of weight at most w in $\mathcal{D}_{ij}$, we know that OPT_{ij} consists entirely of disks in $N(\mathcal{D}_{ij})$ of weight at most w. In this case, we first discard all disks from $N(\mathcal{D}_{ij})$ that have weight larger than w and arrive at the following problem: Find a set of disks of minimum weight from $N(\mathcal{D}_{ij})$ that dominates all disks in $\mathcal{D}_{ij}$. A disk d_1 from $N(\mathcal{D}_{ij})$ dominates a disk d_2 from $\mathcal{D}_{ij}$ if and only if the distance of the centers of d_1 and d_2 is at most 2. Therefore, we can increase the radius of the disks in $N(\mathcal{D}_{ij})$ from 1 to 2 and reduce the radius of the disks in $\mathcal{D}_{ij}$ from 1 to 0 and obtain an equivalent problem: If $\mathcal{D}'$ denotes the set containing the enlarged version of the disks in $N(\mathcal{D}_{ij})$ and $\mathcal{P}$ denotes the set of centers of the disks in $\mathcal{D}_{ij}$, we need to find a minimum-weight subset of the disks in $\mathcal{D}'$ that covers all points in $\mathcal{P}$. Furthermore, we can renormalize the setting so that the disks in $\mathcal{D}'$ have radius 1. The renormalized square S is now a $\delta \times \delta$ square, with $\delta = \mu/2 < 1/2$. Therefore, the problem to be solved can be stated as follows:

> **Disk cover in a small square:** Given a set $\mathcal{P}$ of points in a $\delta \times \delta$ square S, where $\delta < 1/2$, and a set $\mathcal{D}'$ of weighted unit disks, find a minimum-weight subset of $\mathcal{D}'$ that covers all points in $\mathcal{P}$.

In the following subsection, we will present a 2-approximation algorithm for this problem. In view of the discussion above, this implies that we have a 2-approximation algorithm for the problem of computing a minimum-weight set of disks that dominates all disks in $\mathcal{D}_{ij}$ for a given $\mu \times \mu$ square S_{ij}, and this is the ingredient that we needed in the previous section to obtain the constant-factor approximation algorithm for MWDS in unit disk graphs.

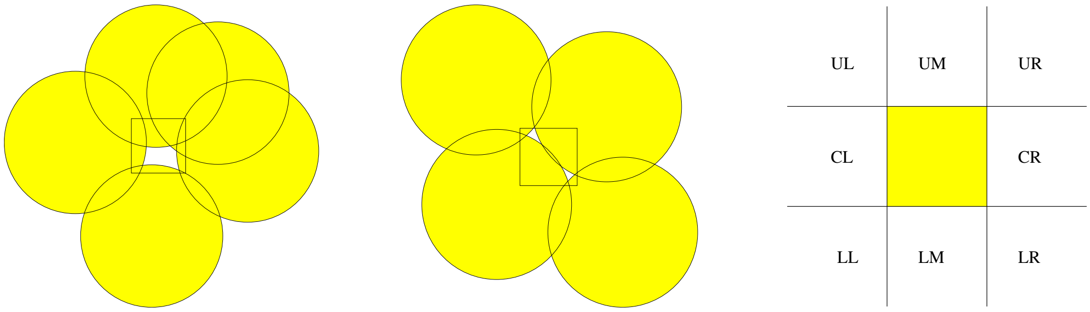

Figure 1: One-hole solution (left), many-hole solution (middle), naming of regions (right)

3.1 Algorithm for disk cover in a small square

We are given a set $\mathcal{P}$ of points in a $\delta \times \delta$ square S and a set $\mathcal{D}'$ of n weighted unit disks, and we want to find a minimum-weight subset of $\mathcal{D}'$ that covers all points in $\mathcal{P}$. Let OPT' denote a set of disks constituting an optimal solution to this problem.

Let C be the area covered by the union of the disks in OPT'. A *hole* of OPT' is defined to be a topological component of $S \setminus C$. Intuitively, if S was a glass window and the disks in OPT' were to cover parts of this window, the holes would be the connected regions where one can still see through the window.

Definition 1 OPT' *is a* one-hole solution *if it has exactly one hole and each disk in* OPT' *forms part of the boundary of that hole (and that part consists of more than* 1 *point).* OPT' *is a* many-hole solution *if it has at least two holes.*

Definition 1 is illustrated in Fig. 1. If OPT' is neither a one-hole solution nor a many-hole solution, it must be of one of the following types: Either OPT' has no hole at all, or it has one hole but not all disks in OPT' form part of the boundary of the hole. If OPT' does not have a hole, we can delete one disk d from OPT' (and remove all points in d from $\mathcal{P}$) to obtain a solution with at least one hole. If OPT' has one hole but not all disks are on the boundary of the hole, let d' be a disk that is not on the boundary of the hole. If we delete d' from OPT' (and the corresponding points from $\mathcal{P}$), we have at least two holes and arrive at a many-hole solution. Therefore, OPT' can always be converted into a one-hole or many-hole solution by deleting at most two disks.

The algorithm guesses whether OPT' is a one-hole solution or a many-hole solution. If OPT' is neither of these, the algorithm also guesses this and additionally guesses the one or two disks that need to be removed from OPT' (and added to the solution computed by the algorithm) in order to obtain a one-hole or many-hole solution. Hence, we can assume that OPT' is a one-hole or many-hole solution and that the algorithm has guessed correctly which of the two is the case. In each of the two cases, we will encounter subproblems that can be solved by dynamic programming, as stated in the following lemma.

Lemma 1 *Let $\mathcal{P}$ be a set of points located in a strip between the horizontal lines $y = y_1$ and $y = y_2$ for some $y_1 < y_2$. Let $\mathcal{D}$ be a set of weighted unit disks with centers above the line $y = y_2$ (upper disks) or below the line $y = y_1$ (lower disks). Furthermore, assume that the union of the disks in $\mathcal{D}$ contains all points in $\mathcal{P}$. Then a minimum-weight subset of $\mathcal{D}$ that covers all points in $\mathcal{P}$ can be computed in polynomial time.*

Proof. A solution consists of some upper disks and some lower disks. All upper disks in the solution intersect the line $y = y_2$, and all lower disks the line $y = y_1$. We view the upper halfplane bounded by $y = y_2$ and the lower halfplane bounded by $y = y_1$ as special cases of disks (with weight 0). For a set $\mathcal{U}$ of upper disks and a point $p \in \mathcal{P}$ with x-coordinate x_p, we say that an upper disk $u \in \mathcal{U}$ is *active* at x_p if its lowest intersection point with the vertical line $x = x_p$ has the smallest y-coordinate among all lowest intersection points of disks $u' \in \mathcal{U}$ with that line. If there are two or more active upper disks at $x = x_p$ by this definition, we consider only the one with leftmost center. For lower disks, active disks are defined similarly (i.e., having an intersection point with $x = x_p$ of largest y-coordinate). For a given solution and a given x-coordinate x_p, there is one active upper disk and one active lower disk at x_p (and each of these could also be the respective halfplane, as mentioned above). The algorithm computes a table T_p for every point $p \in \mathcal{P}$, in order of non-decreasing x-coordinates. For ease of presentation, we assume that no two points have the same x-coordinate. Let $p_1, p_2, \ldots, p_k$ denote the points of $\mathcal{P}$ in order of increasing x-coordinates. For an upper disk u and a lower disk d, the table entry $T_{p_i}(u, d)$ denotes the optimal weight of a solution that covers all points from p_1 up to p_i and has u and d as the active upper and lower disk, respectively, at x_{p_i}. (If u and d do not cover p_i, we say that u, d is not feasible for p_i and set the table entry to ∞.) The table T_{p_1} can be initialized by setting $T_{p_1}(u, d) = w_u + w_d$ for all pairs of disks u and d that cover p_1. Once the tables for $p_1, \ldots, p_{i-1}$ have been computed, the table entries $T_{p_i}(u, d)$ for all feasible disks u and d for p_i can be computed as follows:

$$T_{p_i}(u, d) = \min\{T_{p_{i-1}}(u', d') + [u \neq u'] \cdot w_u + [d \neq d'] \cdot w_d \mid u', d' \text{ feasible for } p_{i-1}\}$$

Here, the term $[u \neq u']$ is 1 if $u \neq u'$, and 0 otherwise (and similarly for $[d \neq d']$). Intuitively, the equation is based on the observation that an optimal solution covering $p_1, \ldots, p_i$ with u and d as active disks for x_{p_i} can be obtained by adding u and d to an optimal solution corresponding to some $T_{p_{i-1}}(u', d')$, where the weight of u or d needs to be added only if x_{p_i} is the first x-coordinate for which u or d is active. The correctness of the calculation in the case of unit disks follows from the fact that an upper or lower disk can be active in the solution only for points in $\mathcal{P}$ that are consecutive (except if the disk is actually the lower or upper halfplane mentioned above, but these special disks have weight 0 and therefore do not cause problems if their weight is added each time they become active). The weight of an optimal solution for the disk cover problem can be found by locating the minimum value $T_{p_k}(u, d)$ among all feasible disks u, d for p_k. The solution itself can be found using standard bookkeeping techniques. □

In the following two subsections, we deal with the one-hole case and the many-hole case, respectively.

3.1.1 One-hole solutions

Assume that OPT' is a one-hole solution. The boundary of the hole is formed by disks from OPT' and, potentially, some parts from sides of the square S (we view the latter as special kinds of disks with weight 0 and infinite radius, i.e., halfplanes, and do not treat them explicitly in the following). All disks in OPT' have their centers outside S. Using the lines that are the extensions of the sides of S, we can partition the plane outside S into 8 regions in the natural way (see also Fig. 1): upper left region (UL), upper middle region (UM), upper right region (UR), central right region (CR), lower right region (LR), lower middle region (LM), lower left region (LL), and central left region (CL). The upper region (U) is the union of UL, UM and UR, and similarly for the lower region (L).

If we follow the boundary of the hole in counterclockwise direction, we will encounter disks with center in CL, then disks with center in L, then disks with center in CR, then disks with center in U.

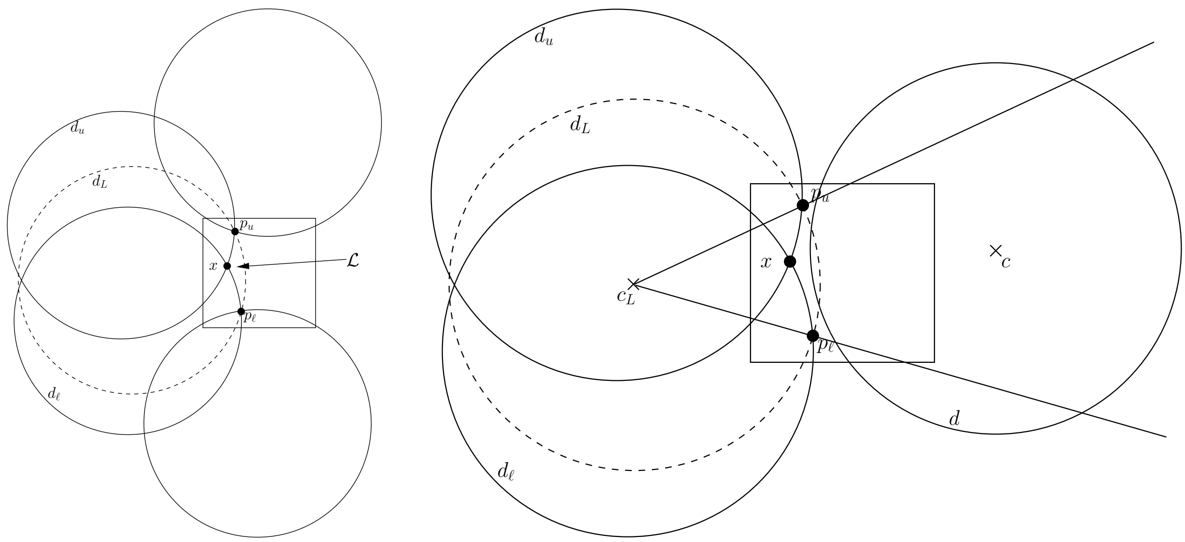

Figure 2: The region $\mathcal{L}$ is defined by parts of the boundaries of disk d_L, drawn dashed, and disks d_u and d_ℓ (left). A disk d with center not in CL from OPT' intersecting $\mathcal{L}$ must have its center in the cone of two halflines starting at the center c_L of d_L and passing through p_u and p_ℓ, respectively (right)

The points on the boundary that are in the intersection of two consecutive disks on the boundary are called *corners*. Each corner is *determined* by two disks (the disks on whose boundaries it lies).

Among all corners that are determined by at least one disk whose center is in CL, let p_ℓ denote the one with the smallest y-coordinate and let p_u denote the one with the largest y-coordinate. Let p'_ℓ and p'_u be defined analogously with respect to CR. (The case where no part of the boundary of the hole is created by disks with center in CL or CR is easier and is not treated in detail here.) The algorithm guesses the corners p_ℓ, p_u, p'_ℓ and p'_u and the pairs of disks determining them. As there are only $O(n^2)$ pairs of disks, the number of potential guesses is polynomial.

Let d_L be the unit disk that has p_ℓ and p_u on the boundary and has its center to the left of the line $\overline{p_\ell p_u}$. Note that in general d_L is not a disk that is part of the input of the problem. Let d_ℓ and d_u be the disks from OPT' that have their center in CL and contain p_ℓ and p_u, respectively, on the boundary. Let x be the intersection point of the boundaries of d_ℓ and d_u that is closer to S. Let $\mathcal{L}$ be the connected region that is delineated by the boundary of d_L between p_u and p_ℓ, and by the boundary of d_ℓ between x and p_ℓ, and by the boundary of d_u between p_u and x. See Fig. 2 (left) for an illustration.

Lemma 2 *The only disks in* OPT' *that intersect* $\mathcal{L}$ *have their center in CL or in the union of UR, CR and LR. Furthermore, no disk from* OPT' *with center in CL can cover a point outside* $\mathcal{L}$ *that is not already covered by* d_u *or* d_ℓ*.*

Proof. As p_u and p_ℓ are on the boundary of the hole, no disk in OPT' can contain p_u or p_ℓ in its interior. Hence, any disk d from OPT' that intersects $\mathcal{L}$ must either have its center to the left of the line $\overline{p_\ell p_u}$ and intersect the parts of the boundaries of d_ℓ and d_u that define $\mathcal{L}$, or it must have its center to the right of the line $\overline{p_\ell p_u}$ and intersect the boundary of $\mathcal{L}$ twice on the part that is also a boundary of d_L. In the former case, the y-coordinate of the center of d must lie between the y-coordinates of the centers of d_ℓ and d_u, and hence d must have its center in CL. (To see this, consider the disk d' that is obtained from d by shifting it horizontally to the right until it first contains p_u or p_ℓ on its boundary; observe that the disk d_u can be rotated around p_u until it becomes identical to d', with its center continuously moving downward; the same argument can be applied to the disk d_ℓ and shows that the center of d' must have larger y-coordinate than the center of d_ℓ. By the same argument, we also have that c_L must lie in CL.) In the latter case, the center c of d must lie in the cone of points between the halflines starting at the center c_L of d_L and passing through p_ℓ and p_u, respectively, see Fig. 2 (right). We want to show that c cannot be in UM or LM. Assume for a contradiction that c is in UM (the case for LM is similar). The slope of the line connecting c_L and p_u is at most $\delta/\sqrt{1-\delta^2}$. Therefore, the largest y-coordinate of a point in the intersection of the cone and UM is bounded by $y_{p_u} + \delta^2/\sqrt{1-\delta^2}$, so the distance between p_u and any point in that intersection is at most $\delta/\sqrt{1-\delta^2}$ (see Fig. 3 for an illustration). Hence, for $\delta < \sqrt{2}/2$ (and we even have $\delta < 1/2$), a unit disk with

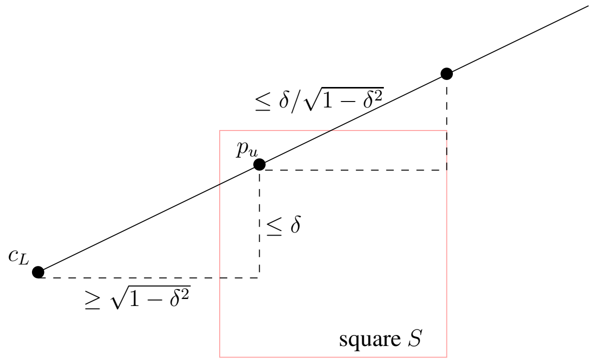

Figure 3: Any disk with center in the cone and in UM contains point p_u, for $\delta < \sqrt{2}/2$

center in that intersection must contain p_u. Thus, c cannot be in UM, as d would then contain p_u in its interior. Similarly, we get that c cannot be in LM. Furthermore, c clearly cannot be in UL or LL, as it must be to the right of p_u. Hence, we have shown that c must be in the union of UR, CR and LR.

We have shown that the only disks in OPT' that intersect $\mathcal{L}$ have their center in CL or in the union of UR, CR and LR. It remains to show that no disk from OPT' with center in CL can cover a point outside $\mathcal{L}$ that is not already covered by d_u or d_ℓ. Let d' be a disk from OPT' with center in CL. All disks from OPT' are on the boundary of the hole, and p_u and p_ℓ are the topmost and lowest corners, respectively, that are determined by at least one disk with center in CL. Therefore, d' must appear on the boundary of the hole between p_u and p_ℓ. This implies that $d' \setminus (d_u \cup d_\ell)$ consists of one region that is contained in $\mathcal{L}$ and a second region that is outside the square S (and cannot contain any points from $\mathcal{P}$). This establishes the claim. $\square$

Similar to $\mathcal{L}$, we can define a region $\mathcal{R}$ with respect to CR, p'_ℓ and p'_u, and the analogue of Lemma 2 holds for $\mathcal{R}$.

Let $\mathcal{P}'$ be the set of points that is obtained from $\mathcal{P}$ by removing the points that are contained in one of the disks determining the four corner points guessed by the algorithm. For the points in $\mathcal{P}' \cap (\mathcal{L} \cup \mathcal{R})$, we can compute an optimal disk cover using Lemma 1 (rotated by 90°), since the points are contained in the vertical strip containing S and the only disks that need to be considered for covering them have their center to the left or to the right of the strip. The remaining points in $\mathcal{P}'$ can only be covered by disks with center in U or in L by OPT', hence we can again compute an optimal disk cover for them using Lemma 1. If we output the union of the two disk covers, we have computed a 2-approximation to the overall disk cover problem in this square.

3.1.2 Many-hole solutions

Now we consider the case that OPT' is a many-hole solution. There must be two disks $d_1, d_2 \in \mathrm{OPT}'$ such that $S \setminus (d_1 \cup d_2)$ consists of two disjoint regions and each of these two regions contains a hole of

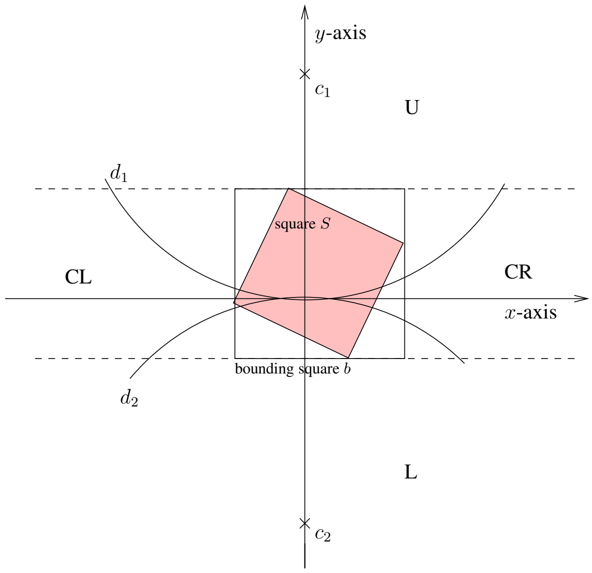

Figure 4: New coordinate system for the many-hole case

OPT'. (As a special case, d_1 or d_2 could be any halfplane that touches a side of S but does not contain S; in this case, we would have a single disk from OPT' that intersects the square in such a way that two holes are created.) We use a new coordinate system in which the y-axis contains the centers c_1 and c_2 of d_1 and d_2, respectively, and the intersection points of the boundaries of d_1 and d_2 are on the x-axis. Let b be the smallest axis-parallel square containing the (rotated) square S. Let δ' be the side length of b. Note that $\delta' \leq \delta\sqrt{2} < \sqrt{2}/2$. See Fig. 4 for an illustration. As for the one-hole case, we partition the plane outside b into regions UL, UM, UR, CR, LR, LM, LL, CL, and we define regions U and L as before.

The disks d_1 and d_2 create two holes in S; we refer to the left hole as LH, and to the right hole as RH. Because OPT' is a superset of $\{d_1, d_2\}$, OPT' may contain more than two holes, but all the holes in OPT' are contained in either LH or RH.

We begin with some observations: It is clear that for a point with coordinates (x, y) that is contained in LH or RH, we have $|y| < 1 - \sqrt{1 - x^2}$. Furthermore, for any two points such that one is from LH and one from RH, the y-distance between the points is at most $1 - \sqrt{1 - \alpha^2}$, where α is the x-distance between the points. This follows from the following computation. Assume the first point has coordinates $(-\alpha_1, y_1)$ and the second point (α_2, y_2), for some $\alpha_1, \alpha_2 \geq 0$. We have $\alpha_1 + \alpha_2 = \alpha \leq \delta'$, and the y-distance of the points is bounded by $|y_1| + |y_2| \leq 1 - \sqrt{1 - \alpha_1^2} + 1 - \sqrt{1 - (\alpha - \alpha_1)^2}$. We find that this expression is maximized at $\alpha_1 = \alpha$ and $\alpha_1 = 0$, giving an upper bound of $1 - \sqrt{1 - \alpha^2}$.

Lemma 3 *In* OPT'*, no disk with center in the union of UR, CR and LR (in the union of UL, CL and LL) can intersect LH (RH).*

Proof. We consider the case of LH and the union of UR, CR and LR only. The other case is symmetric. For brevity, let R denote the union of UR, CR and LR.

Assume, for a contradiction, that there is a disk d in OPT' that has its center c in R and intersects LH. We will show that d covers RH completely, contradicting our assumption that OPT' is a many-hole solution with at least one hole in RH.

Assume that d does not cover RH completely. Then there must be a point q in RH that is not contained in d. Furthermore, as d does not cover LH completely, there must be a point p in LH that

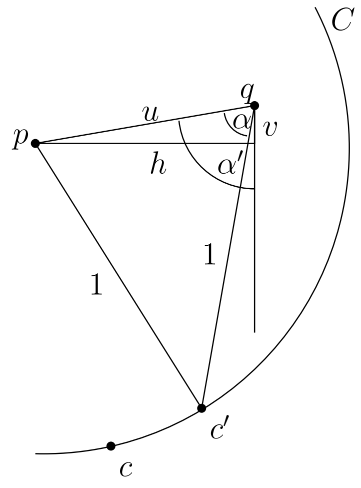

Figure 5: No disk from the union of UR, CR and LR can intersect LH

lies on the boundary of d. Assume without loss of generality that q is not below p. As d contains p but not q and its center is to the right of q, c must lie below q. Moreover, c lies on a circle C of radius 1 with center p, because d has p on its boundary. See Fig 5. Consider point c' on C at distance 1 from q and below q (observe that c' exists because p and q are at most $\sqrt{2}/2$ apart). Let α denote the angle of lines $\overline{qp}$ and $\overline{qc'}$ and let α' denote the angle of line $\overline{qp}$ and the negative y-axis. Observe that both α and α' are at most 180°. Because c lies left of c' on circle C, it is enough to show that $\alpha \leq \alpha'$, i.e., $\cos\alpha \geq \cos\alpha'$, i.e., c is not in R (a contradiction). Let u, h and v denote the distance, the horizontal distance and the vertical distance of p and q. Then $\cos\alpha = u/2$ and $\cos\alpha' = v/u$. We have $v \leq 1 - \sqrt{1-h^2}$, i.e., $1 - h^2 \leq (1-v)^2$, i.e., $2v \leq v^2 + h^2 = u^2$, i.e., $v/u \leq u/2$, which concludes the proof. $\square$

Our approach to the weighted disk cover problem in the many-hole case can now be outlined as follows. We will show that LH contains a region $\mathcal{L}$ such that points in $\mathcal{L}$ can be covered only by disks with center in CL by OPT'. Let $\mathcal{P}' \subseteq \mathcal{P}$ be the points in LH that are not in $\mathcal{L}$ and are not already covered by the disks the algorithm guesses to define $\mathcal{L}$. We will show that points in $\mathcal{P}'$ can only be covered by disks with center in U or L. The same approach will be applied to RH. This breaks the problem into two independent subproblems: covering points in $\mathcal{L}$ and in the corresponding region of RH using disks with center in CL or CR, and covering the remaining points using disks with center in U or L. Each of the two subproblems can be solved optimally by dynamic programming (Lemma 1). Since the subproblems are independent, the union of their optimal solutions gives an optimal solution to the disk cover problem in the many-hole case.

In the following we discuss this solution approach for points from $\mathcal{P}$ that are in LH in more detail. The arguments for RH are symmetric. Lemma 3 shows that no disk with center in the union of UR, CR and LR can intersect LH. We distinguish the following three cases concerning disks with center in CL that are contained in OPT':

1. OPT' does not contain any disk with center in CL.

2. OPT' contains one disk with center in CL.

3. OPT' contains two or more disks with center in CL.

The algorithm guesses which of the three cases holds for OPT'. In the first case we have that all points in LH are covered by disks with center in U or L by OPT'. In the second case, we have to additionally guess the disk d with center in CL that is in OPT'. The remaining points in LH (those that are not covered by d) can then again only be covered by disks with center in U or L by OPT'.

It remains to deal with the third case, where OPT' contains two or more disks with center in CL. We can show that in this case, all disks from OPT' with center in CL must lie on the boundary of the same hole.

Assume that the boundary of the solution contains a part of a disk with center in CL (as we will show in Lemma 7, this is always the case). Let p_u and p_ℓ be the corners with largest and smallest y-coordinate, respectively, among all corners of holes that are determined by at least one disk with center in CL. Let d_u and d_ℓ be the disks with center in CL that determine p_u and p_ℓ, respectively, and let d_L be the disk with center to the left of p_u and p_ℓ that contains p_u and p_ℓ on the boundary. (Note that d_L is in general not a disk that is part of the input.) By our assumption p_u and p_ℓ are distinct.

The algorithm guesses these points and the disks determining them. The disks d_u, d_ℓ and d_L define a region $\mathcal{L}$ in the same way as in the one-hole case, see Fig. 2 (left). The region $\mathcal{L}$ contains points from $\mathcal{P}$ that are covered in OPT' using disks with center in CL (see Lemma 4). It follows that these disks form a boundary between p_u and p_ℓ of the same hole (Lemma 5). And, because every disk from OPT' with center in CL must be on this boundary (Lemma 6), the disks with center in CL cannot cover any other point outside $\mathcal{L}$.

The points in LH that are not in $\mathcal{L}$ must be therefore covered by disks with center in U or L. Similarly, the points in RH can be split into those that can only be covered by disks with center in CR and those that can be covered by disks with center in U or L. We create two subproblems that can be solved by dynamic programming (Lemma 1): one subproblem for the points to be covered by disks with center in U or L, and one subproblem for the points to be covered by disks with center in CL or CR. For both subproblems, there is a (vertical or horizontal) strip that contains all the points to be covered while all disks have their centers outside that strip. As OPT' cannot contain a disk that covers points from both subproblems, the union of the optimal solutions to the two subproblems actually gives an optimal disk cover for the square.

Lemma 4 OPT' *does not contain any disk with center outside CL that intersects* $\mathcal{L}$ *in LH.*

Proof. Let d be a disk that has its center c outside CL and that intersects $\mathcal{L}$ in LH. Since c is not in CL, c has to be to the right of the line $\overline{p_\ell p_u}$ and d intersects d_L twice between the points p_u and p_ℓ. We claim that c must be in the union of UR, CR and LR. This follows by the same arguments as in the proof of Lemma 2 (which are applicable since $\delta' \leq \sqrt{2}\delta < \sqrt{2}/2$). However, Lemma 3 shows that no disk with center in the union of UR, CR and LR intersects LH, and therefore no such disk can intersect $\mathcal{L}$ in LH. □

Lemma 5 p_u *and* p_ℓ *lie on the boundary of the same hole in* OPT'*, and the boundary between* p_u *and* p_ℓ *is formed by disks with center in CL (or parts of sides of* S*) only.*

Proof. Observe that from the construction of the region $\mathcal{L}$ we have that all disks with center in CL that are on the boundary of the solution can appear only in the region $\mathcal{L}$. Since no other disk than those with center in CL can intersect $\mathcal{L}$ (Lemma 4), the results follows. □

In the following, we define ℓ_u and ℓ_d to be the horizontal lines that contain the upper and lower side of b, respectively.

Lemma 6 *If there is a disk with center in CL that lies on the boundary of a hole in* OPT'*, then all disks with center in CL lie on the boundary of a hole in* OPT'.

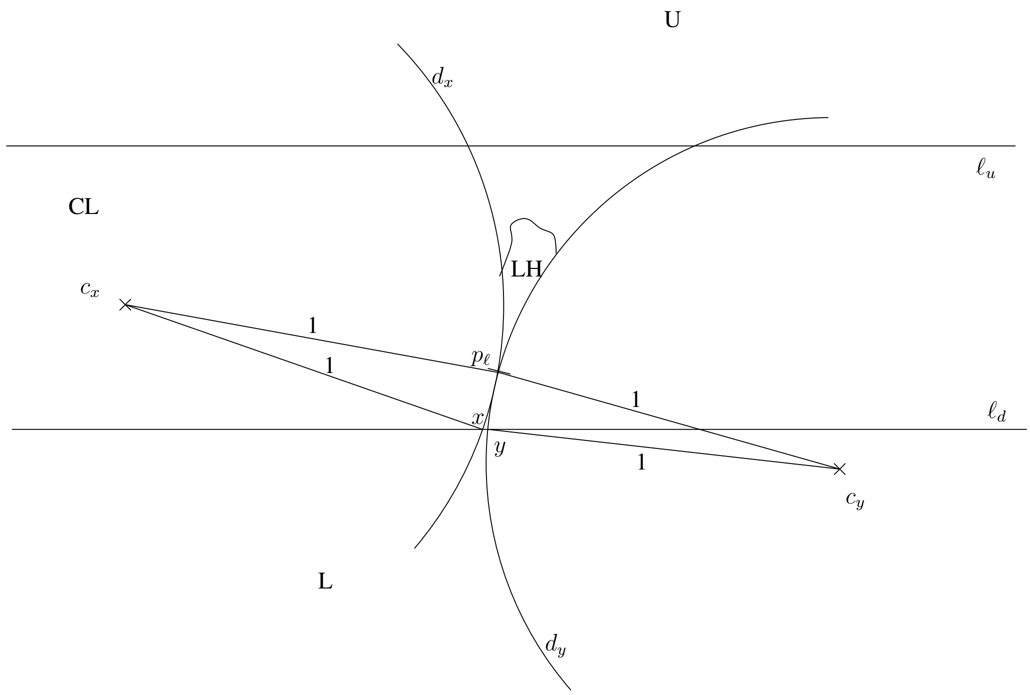

Figure 6: Setting in the many-hole case where a disk with center in CL lies on the boundary

Proof. For a contradiction, assume that $d_l' \in \mathrm{OPT}'$ has its center in CL but does not lie on the boundary of any hole in OPT'. Let T denote the set of disks in OPT' that have their centers in CL. Consider the boundary B_T formed by disks in T inside b. Disk d_l' must lie on B_T. Let q be a corner point of B_T formed by d_l'. Recall that p_u and p_ℓ are the highest and lowest corner points of OPT' that are determined by a disk with center in CL. This means that p_u and p_ℓ are the highest and lowest points on B_T that are also on the boundary of a hole in OPT'. q cannot be between p_u and p_ℓ on B_T by Lemma 5. Therefore, let us assume that q lies below p_ℓ. (If q lies above p_u, the arguments are analogous.)

Let d_x with center in CL and d_y with center in L be the disks that determine p_ℓ. (Observe that d_y has to exist; if p_ℓ is created as an intersection of d_x and ℓ_d, then q cannot be below p_ℓ.) Let c_x and c_y be their respective centers, see Fig. 6. Let x be the rightmost intersection point of the boundary of d_x and ℓ_d. Let y be the leftmost intersection point of the boundary of d_y and ℓ_d. The only area that is left uncovered by disks d_x and d_y and that could be covered by d_l' lies in the triangular region with corner points p_ℓ, x and y. That region is nonempty if y lies to the right of x. We claim that if $x = y$, then c_y lies in R, the union of UR, CR and LR. Thus if y moves to the right of x, then c_y moves to the right as well, i.e., c_y stays in R, which is a contradiction to Lemma 3. We are left to prove the claim. Assume that $x = y$. Let h be the horizontal distance of c_y and p_ℓ (see Fig. 7 for an illustration). Observe that h is equal to the horizontal distance of c_x and x (for this observe that line $\overline{c_x p_\ell}$ is parallel to line $\overline{x c_y}$). Therefore, $h \geq \sqrt{1 - \delta'^2}$. For $\delta' \leq \sqrt{2}/2$ we have $h \geq \delta'$ and therefore c_y lies in R. $\square$

Lemma 7 *There can be at most one disk $d_\ell \in \mathrm{OPT}'$ that has its center in CL and is not on the boundary of* OPT'.

Proof. Assume for contradiction that there are at least two disks with center in CL that do not appear on the boundary of OPT'. Lemma 6 implies that then all disks with center in CL do not appear on

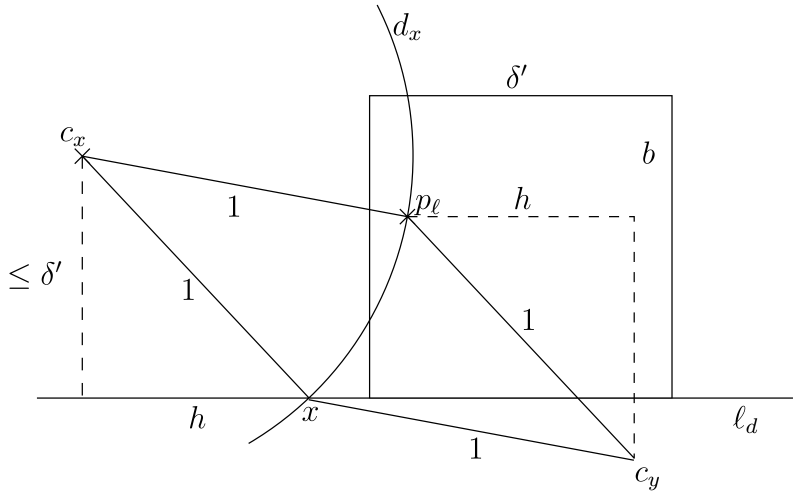

Figure 7: The shown setting is impossible: Center c_y must lie to the right of b because $h > \delta'$

the boundary of OPT'. Now we proceed similarly as in the proof of Lemma 6. Let T denote the set of disks in OPT' that have their center in CL. Consider the boundary B_T formed by disks from T inside b (and observe that all disks from T appear on B_T). Let d_x and d_z be two adjacent disks on B_T (with common corner point q), and assume without loss of generality that d_x is above d_z, i.e., the center c_x of d_x is above the center c_z of d_z, see Fig. 8. We can assume that q is in LH, because otherwise one of the two disks d_x and d_z would be redundant. Because no disk with center in CL is on the boundary of OPT', point q must be covered by some disk with center in U or L (note that Lemma 3 shows that q cannot be covered by a disk with center in the union of UR, CR and LR). Assume that q is covered by a disk d_y with center c_y in L (the case when c_y is in U is analogous). Let x and z be the rightmost intersections of the disks d_x and d_z, respectively, with line ℓ_d. Points that are covered by d_z and not by d_x must be in the triangular region qxz. We show that the triangular region qxz is covered by d_y, and therefore d_z could be removed from OPT', a contradiction to the fact that OPT' is an optimum solution. Disk d_y does not cover the entire triangular region qxz if and only if y, the leftmost intersection of d_y and ℓ_d, is to the right of x. (Note that the rightmost intersection of d_y and ℓ_d is always to the right of x, if d_y has center in L.) If d_y has q on its boundary (i.e., the distance of c_y and q is 1), we can argue similarly as in Lemma 6 (q takes the role of p_ℓ) and show that d_y must have its center c_y in the union of UR, CR, LR. It is easy to observe that if the distance from c_y to q is less than 1, c_y remains in the union of UR, CR and LR, which contradicts Lemma 3. $\square$

In summary, we have shown that in both the one-hole case and the many-hole case we can obtain a 2-approximation (in the many-hole case, even an optimal solution) of the minimum-weight disk cover for the given $\delta \times \delta$ square S. Furthermore, all other cases (no holes, or one hole with not all disks on the boundary of the hole) can be reduced to one of these cases by guessing one or two disks in the optimal solution. Therefore, we obtain a 2-approximation algorithm for the problem of computing a minimum-weight disk cover in a small square.

3.2 Algorithm for general weighted disk cover with unit disks

We remark that our result on disk cover in a small square also implies a constant-factor approximation algorithm for the general weighted disk cover problem with unit disks (i.e., given a set of points and a set of weighted unit disks, find a minimum-weight set of disks that covers all the points). We

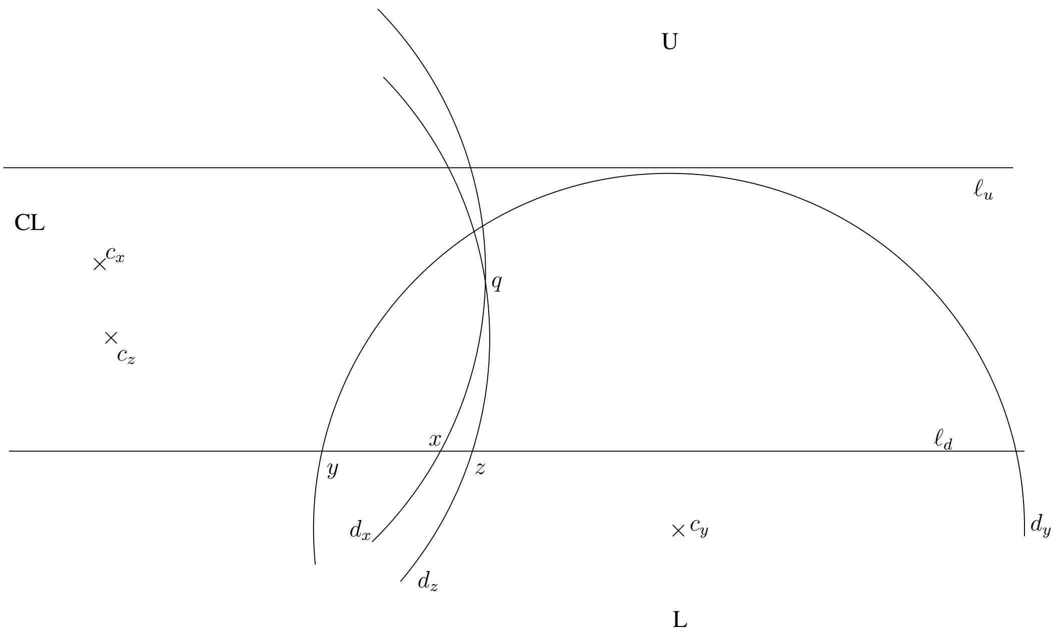

Figure 8: Setting for the many-hole case where no disk with center in CL is on the boundary

can simply partition the plane into $\delta \times \delta$ squares and compute an approximate disk cover for each square. Then we output the union of all computed disk covers as the solution. As a disk from the optimal solution can be used to cover points in at most $O(1/\delta^2)$ different $\delta \times \delta$ squares, we lose only a factor of $O(1/\delta^2)$ in the approximation ratio by solving the problem for each square separately. More precisely, for every disk d, any square containing a point covered by d must be fully contained in a disk of radius $1 + \sqrt{2}\delta$ around the center of d, and hence there are at most $\lfloor \pi(1 + \sqrt{2}\delta)^2/\delta^2 \rfloor = 36$ such squares (for $\delta = 0.999/2$). Since our algorithm for disk cover in one square has approximation ratio 2, the overall approximation ratio of our algorithm for the weighted disk cover problem with unit disks is 72. Previously, constant-factor approximation algorithms were known only for the unweighted case of the disk cover problem [3, 4]. Moreover, the approximation ratio 72 of our algorithm improves also the approximation ratio 102 for the unweighted case from [4].

4 Connecting the dominating set

In this section we consider the problem of adding disks to a given dominating set in order to produce a connected dominating set. We present an algorithm that solves this problem by adding disks of total weight at most $O(w^*)$, where w^* denotes the optimal weight of a connected dominating set for the given set of weighted unit disks. Note that the problem of connecting up a dominating set is a special case of the node-weighted Steiner tree problem; for general graphs, the best known approximation ratio for the latter problem is logarithmic in the size of the graph [9].

Let $\mathcal{D}$ be a set of weighted unit disks, and let $U \subseteq \mathcal{D}$ be a dominating set. Let G denote the unit disk graph corresponding to the disks in $\mathcal{D}$, and assume that G is connected (otherwise, G cannot have a connected dominating set). The vertex set of a connected component of $G[U]$ (the subgraph of G induced by U) is called a *cluster* of U. We create an auxiliary graph H. The vertices of H correspond to the clusters of U. For every path of length at most 3 in G that connects a vertex in one cluster c_1 of U to a vertex in another cluster c_2 of U and whose one or two internal vertices are not in U, we add an edge between c_1 and c_2 to H. The weight of the edge is the sum of the weights of the disks corresponding to the one or two internal vertices of the path. Note that H can have parallel edges. Next, we compute a minimum spanning tree T in H. (The proof of the theorem below shows that H is a connected graph.) Finally, we connect the dominating set U by adding all disks that correspond to internal vertices of the paths in G that correspond to the edges of T.

Theorem 2 *Let $\mathcal{D}$ be a set of weighted unit disks and U be a dominating set. Let w^* be the weight of a minimum-weight connected dominating set for $\mathcal{D}$. There is an efficient algorithm that computes a set U' of disks such that $U \cup U'$ is a connected dominating set and $w(U') \leq 17w^*$.*

Proof. We show that the auxiliary graph H contains a spanning tree T' of weight at most $17w^*$. This implies that H is connected. Furthermore, the weight of the set U' of disks that the algorithm adds to U is at most the weight of the minimum spanning tree, and the weight of the minimum spanning tree is upper bounded by the weight of T'. Therefore, we get $w(U') \leq 17w^*$.

It remains to show how to construct a spanning tree T' of H with weight at most $17w^*$. Let U^* be an optimal connected dominating set, $w(U^*) = w^*$. Let C be an arbitrary non-empty set of clusters of U, but not the set of all clusters of U. Let $\bar{C}$ be the set of the remaining clusters of U. We claim that G must contain a path π from a vertex in some cluster in C to a vertex in some cluster in $\bar{C}$ such that π contains at most two internal vertices and has the property that all its internal vertices are in $U^* \setminus U$. (Note that such a path π corresponds to an edge in H.) To prove the claim, we argue as follows. Let x be an arbitrary vertex in a cluster in C, and y an arbitrary vertex in a cluster in $\bar{C}$. As U^* is a connected dominating set, there must be a path p in G from x to y all of whose internal vertices are in U^*. Let x' be the last vertex on p that is not in U and that is dominated by a vertex x'' in a cluster in C. Note that such a vertex x' must exist. Furthermore, x' or the vertex y' after x' on p must be dominated by a vertex y'' in a cluster in $\bar{C}$. Therefore, we obtain the desired path as x'', x', y'' or x'', x', y', y''.

Now we can create a spanning tree of H as follows. We start with a tree consisting of a single vertex of H (corresponding to some cluster of U) and grow the tree by repeatedly finding a path π in G that connects a vertex from a cluster in the tree to a vertex in a cluster not in the tree and has the properties discussed above. The claim above shows that such a path must exist. We can thus grow the tree by adding the edge in H that corresponds to the path π. This is repeated until we have a spanning tree T'.

The weight of each edge in the spanning tree T' corresponds to the weight of the internal vertices (which are in U^*) of a path of length at most 3 that connects different clusters of U. Furthermore, a vertex (disk) d of U^* can contribute to at most 17 edges of H: Whenever d contributes to the weight of an edge, it is an internal vertex of a path that connects two clusters of U whose closest disks have (graph-theoretic) distance at most 2 from it. However, the set of disks at distance at most 2 from d can contain at most 18 disjoint disks (see e.g. [16]) and therefore at most 18 disks from different clusters of U. As the spanning tree can contain at most 17 edges between these 18 clusters, we obtain that d contributes its weight to at most 17 edges of the spanning tree T'. Consequently, $w(T') \leq 17w^*$. □

Together with Theorem 1, we obtain the following corollary.

Corollary 1 *There is a constant-factor approximation algorithm for the minimum-weight connected dominating set problem in unit disk graphs.*

The approximation ratio of the algorithm of Corollary 1 is at most $72 + 17 = 89$.

5 A 3-approximation algorithm for minimum-weight forwarding sets

In this section we consider the *minimum-weight forwarding set problem* (MWFS). In this problem, we are given a distinguished unit disk d_o (the source disk), a set of weighted unit disks $\mathcal{D}$ with centers in d_o, and a set of points $\mathcal{P}$ in the plane outside d_o (but contained in the union of the disks in $\mathcal{D}$). The goal is to find a minimum-weight subset $\mathcal{D}'$ of $\mathcal{D}$ such that every point in $\mathcal{P}$ is covered by at least one disk from $\mathcal{D}'$. In the unweighted version of the problem (MFS), all disks have weight 1. Obviously, the problem is a special case of the disk cover problem.

MFS and MWFS arise in wireless ad-hoc networks in the context of the efficient implementation of flooding [4]. Flooding is a broadcasting mechanism where each node forwards the message to all its neighbors. This leads to many redundant messages. A more efficient implementation is obtained by letting each node forward the message only to a subset of the one-hop neighbors (the forwarding set) that covers all the two-hop neighbors. If the wireless network is modeled as a unit disk graph, the problem of determining a smallest (or minimum-weight) forwarding set for a node is just MFS (or MWFS) as defined above.

Calinescu et al. [4] devised a 3-approximation algorithm for MFS. We combine our ideas from Section 3 with their approach and obtain a 3-approximation algorithm for MWFS, the weighted version of the problem. The 3-approximation algorithm ALG for the unweighted case from [4] partitions the points in $\mathcal{P}$ according to the four quadrants defined by two orthogonal lines through the center o of disk d_o, and then independently solves the covering problem for each quadrant. The union of these four disk covers is then a disk cover for all the points in $\mathcal{P}$. Clearly, the same approach can be applied to the weighted case as well.

Lemma 8 ([4]) *If an α-approximation algorithm is used for the (weighted) covering problem in each quadrant, the approximation ratio of ALG is at most 3α.*

The proof of this lemma is based on the observation that each disk in $\mathcal{D}$ can cover points from $\mathcal{P}$ in at most three quadrants.

We present an optimal algorithm for the weighted covering problem of points in one quadrant, thus obtaining a 3-approximation algorithm for MWFS.

Lemma 9 *There is a polynomial-time optimal algorithm for the minimum-weight forwarding set problem if the points $\mathcal{P}$ lie in one quadrant only.*

Proof. For a disk $d \in \mathcal{D}$, let $w(d)$ denote the weight of the disk. Let Q be the quadrant in which the points $\mathcal{P}$ lie. Let $Q' = Q - d_o$ be the *external* quadrant, i.e., the quadrant without the disk d_o. Observe that in any optimal solution $\mathcal{D}'$, each disk $d \in \mathcal{D}'$ appears in Q' on the boundary of the solution (where the boundary of the solution means the boundary of the union of the disks from $\mathcal{D}'$). To see this, consider a point $p \in \mathcal{P}$. Let $\overline{op}$ be the half-line starting at o and passing through p. Because p is covered by $\mathcal{D}'$, the half-line must intersect the boundary of $\mathcal{D}'$ at some point o' that lies beyond the point p. Let $d' \in \mathcal{D}'$ be the disk that contains o'. We say that disk d' is *active* at p. Observe that o' is in Q'. Because every disk from $\mathcal{D}$ contains o, the disk d' contains the whole segment of $\overline{op}$ between o and o'. Thus the disk d' contains p as well. Therefore, if there is a disk $d \in \mathcal{D}'$ that is not on the boundary of the solution, we could remove the disk and keep the points covered, a contradiction to $\mathcal{D}'$ being an optimal solution.

In [4] it was proved that in Q' the boundaries of any two disks from $\mathcal{D}$ can intersect in at most one point. This implies that each disk from the optimal solution appears on the boundary of $\mathcal{D}'$ in Q' exactly once (i.e., there is exactly one continuous part of the disk on the boundary).

We present a dynamic programming approach for the covering problem of points in Q (which is an adaptation of the dynamic programming from Lemma 1). Assume that the points in $\mathcal{P}$ are ordered according to their polar coordinates (with the reference center being o). The algorithm computes a table with entries $T_i(d_j)$ that store the cost of an optimal solution for covering the points $p_1, \ldots, p_i$ in such a way that d_j is active at p_i. If there is no solution for which d_j is active at p_i, we set the entry to ∞. Computing $T_1(d_j)$ is straightforward: We set $T_1(d_j) = w(d_j)$ if d_j covers p_1, otherwise we set $T_1(d_j) = \infty$. For a disk d_j that covers p_i, we can compute the cost $T_i(d_j)$ using

$$T_i(d_j) = \min_k \{T_{i-1}(d_k) + [j \neq k] \cdot w(d_j) \ \mid d_j \text{ active at } p_i \text{ for the solution of } T_{i-1}(d_k)\}.$$

Here, the term $[j \neq k]$ is 1 if $j \neq k$, and 0 otherwise. The correctness of the computation is justified by the fact that each disk can be active only for points in $\mathcal{P}$ that are consecutive. The weight of the optimal solution can be obtained as the minimum of the table entries $T_n(d)$, where n is the number of points in $\mathcal{P}$ and d ranges over all disks in $\mathcal{D}$ that cover p_n. The optimal solution itself can be obtained using standard bookkeeping techniques. □

Theorem 3 *There is a* 3*-approximation algorithm for the minimum-weight forwarding set problem.*

6 Conclusion

We have presented the first constant-factor approximation algorithms for MWDS and MWCDS in unit disk graphs. Our techniques also yield a constant-factor approximation algorithm for the weighted disk cover problem with unit disks, and a 3-approximation algorithm for the weighted forwarding set problem.

Our algorithms for MWDS and MWCDS make use of the geometry of the disks and therefore require the disk representation as part of the input. It would be interesting to determine whether similar approximation ratios can be achieved if the disk representation of the unit disk graph is not given as part of the input.

As our algorithm for MWDS partitions the plane into small squares and solves the problem for each square separately, it is suitable for a distributed implementation. However, the running time of our algorithm, although polynomial, is quite large, since the algorithm must try all possibilities for the many guesses it makes about the optimal solution. Furthermore, the approximation ratio is not a very small constant. It would be interesting to design an improved algorithm for MWDS that is faster and achieves a better approximation ratio. An interesting question is whether MWDS or MWCDS even admit a PTAS for unit disk graphs. Finally, it would be interesting study whether MDS or MWDS admit approximation algorithms with ratio better than $\Omega(\log n)$ on general disk graphs.

References

[1] K. Alzoubi, P.-J. Wan, and O. Frieder. Message-optimal connected dominating sets in mobile ad hoc networks. In *Proceedings of the 3rd ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc 2002)*, pages 157–164, 2002.

[2] B. S. Baker. Approximation algorithms for NP-complete problems on planar graphs. *Journal of the ACM*, 41(1):153–180, 1994. Extended abstract published in the proceedings of FOCS'83, pp. 265–273, 1983.

[3] H. Brönnimann and M. T. Goodrich. Almost optimal set covers in finite VC-dimension. *Discrete & Computational Geometry*, 14(4):463–479, 1995.

[4] G. Calinescu, I. Mandoiu, P.-J. Wan, and A. Zelikovsky. Selecting forwarding neighbors in wireless ad hoc networks. *Mobile Networks and Applications*, 9(2):101–111, 2004.

[5] X. Cheng, X. Huang, D. Li, W. Wu, and D.-Z. Du. A polynomial-time approximation scheme for the minimum-connected dominating set in ad hoc wireless networks. *Networks*, 42(4):202–208, 2003.

[6] B. N. Clark, C. J. Colbourn, and D. S. Johnson. Unit disk graphs. *Discrete Mathematics*, 86:165–177, 1990.

[7] U. Feige. A threshold of $\ln n$ for approximating set cover. In *Proceedings of the 28th Annual ACM Symposium on Theory of Computing (STOC'96)*, pages 314–318, 1996.

[8] M. R. Garey and D. S. Johnson. *Computers and Intractability. A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company, New York-San Francisco, 1979.

[9] S. Guha and S. Khuller. Improved methods for approximating node weighted Steiner trees and connected dominating sets. *Information and Computation*, 150(1):57–74, 1999.

[10] D. S. Hochbaum and W. Maass. Approximation schemes for covering and packing problems in image processing and VLSI. *Journal of the ACM*, 32(1):130–136, 1985.

[11] H. B. Hunt III, M. V. Marathe, V. Radhakrishnan, S. S. Ravi, D. J. Rosenkrantz, and R. E. Stearns. NC-Approximation schemes for NP- and PSPACE-hard problems for geometric graphs. *Journal of Algorithms*, 26(2):238–274, 1998.

[12] D. Lichtenstein. Planar formulae and their uses. *SIAM Journal on Computing*, 11(2):329–343, 1982.

[13] M. V. Marathe, H. Breu, H. B. Hunt III, S. S. Ravi, and D. J. Rosenkrantz. Simple heuristics for unit disk graphs. *Networks*, 25:59–68, 1995.

[14] E. J. van Leeuwen. Approximation algorithms for unit disk graphs. In *Proceedings of the 31st International Workshop on Graph-Theoretic Concepts in Computer Science (WG'05)*, LNCS 3787, pages 351–361, 2005.

[15] V. V. Vazirani. *Approximation Algorithms*. Springer, 2001.

[16] Y. Wang and X.-Y. Li. Distributed low-cost backbone formation for wireless ad hoc networks. In *Proceedings of the 6th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc 2005)*, pages 2–13, 2005.