In: Proceedings of the Twelfth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'01), pp. 671–679. © SIAM 2001.

Polynomial-Time Approximation Schemes for Geometric Graphs

Thomas Erlebach* Klaus Jansen† Eike Seidel†

Abstract

A disk graph is the intersection graph of a set of disks with arbitrary diameters in the plane. For the case that the disk representation is given, we present polynomial-time approximation schemes (PTASs) for the maximum weight independent set problem (selecting disjoint disks of maximum total weight) and for the minimum weight vertex cover problem in disk graphs. These are the first known PTASs for $\mathcal{NP}$-hard optimization problems on disk graphs. They are based on a novel recursive subdivision of the plane that allows applying a shifting strategy on different levels simultaneously, so that a dynamic programming approach becomes feasible. The PTASs for disk graphs represent a common generalization of previous results for planar graphs and unit disk graphs. They can be extended to intersection graphs of other "disk-like" geometric objects (such as squares or regular polygons), also in higher dimensions.

1 Introduction

Intersection graphs are graphs whose vertices are represented by sets such that two vertices are adjacent if and only if the corresponding sets have a non-empty intersection. They have been studied by many authors [8, 4, 18]. We are interested in approximation algorithms for $\mathcal{NP}$-hard optimization problems on intersection graphs of geometric objects, in particular approximation algorithms for independent set and vertex cover. Two prominent applications of geometric intersection graphs are frequency assignment in cellular networks [16] and map labeling [1].

The goal of the *maximum weight independent set* problem (MWIS) is to compute, for a given set of geometric objects with certain weights, a subset of disjoint (non-overlapping) objects with maximum total weight. The goal of the *minimum weight vertex cover* problem (MWVC) is to compute a subset of the given objects with minimum total weight such that, for any two intersecting objects, at least one of the objects is contained in the subset. MIS and MVC refer to the unweighted versions of these problems. We obtain polynomial-time approximation schemes for MWIS and MWVC in the intersection graphs of disks, squares, or other "disk-like" objects, also in higher dimensions.

1.1 Preliminaries.

For a set V of geometric objects, the corresponding *intersection graph* is the undirected graph with vertex set V and an edge between two vertices if the corresponding objects intersect.

Assume that we are given a set $\mathcal{D} = \{D_1, \ldots, D_n\}$ of n disks in the plane, where D_i has diameter d_i, center $c_i = (x_i, y_i)$, and weight w_i. For a subset $U \subseteq \mathcal{D}$, $w(U)$ denotes the sum of the weights of the disks in U. Disks D_i and D_j *intersect* if $dist(c_i, c_j) \leq (d_i + d_j)/2$, where $dist(p_1, p_2)$ denotes the Euclidean distance between two points p_1 and p_2 in the plane. A *disk graph* (sometimes also called a *circle graph*) is the intersection graph of a set of disks. We assume that the input to our algorithms is the set $\mathcal{D}$ of disks, not only the corresponding intersection graph. This is an important distinction, because determining for a given graph whether it is a disk graph is known to be $\mathcal{NP}$-complete [15], and hence no efficient method is known for computing a disk representation if only the intersection graph is given.

Interestingly, every planar graph is a *coin graph*, i.e., the intersection graph of a set of interior-disjoint disks [14]. Therefore, the class of disk graphs properly contains the class of planar graphs.

For a given set $\mathcal{D}$ of disks in the plane, we let $OPT_{IS}(\mathcal{D})$ and $OPT_{VC}(\mathcal{D})$ denote the total weight of an optimal solution for MWIS and MWVC, respectively. An algorithm is a ρ-approximation algorithm for MWIS if it runs in polynomial time and always computes an independent set of total weight at least $\frac{1}{\rho} OPT_{IS}(\mathcal{D})$. An algorithm is a ρ-approximation algorithm for MWVC if it runs in polynomial time and always computes a vertex cover of total weight at most $\rho OPT_{VC}(\mathcal{D})$. An algorithm is a *polynomial-time approximation scheme* (PTAS) for MWIS if it takes an additional parameter $\varepsilon > 0$ and always computes an independent set of total weight at least $\frac{1}{1+\varepsilon} OPT_{IS}(\mathcal{D})$, where the running-time is polynomial in the size of the representation of $\mathcal{D}$ for

*Computer Engineering and Networks Laboratory, Dept. of Electrical Engineering, ETH Zürich, CH-8092 Zürich, Switzerland. Email: erlebach@tik.ee.ethz.ch

†Institute for Computer Science and Applied Mathematics, Christian-Albrechts-University of Kiel, Olshausenstr. 40, D-24098 Kiel, Germany. Email: {kj|ese}@informatik.uni-kiel.de. Supported in part by EU Research Training Network ARACNE, HPRN-CT-2000-00112.

fixed $\varepsilon > 0$. A PTAS for MWVC is defined analogously. If an algorithm is a ρ-approximation algorithm, the algorithm is also said to have approximation ratio ρ.

Note that the complement of an independent set is a vertex cover, and vice versa. Therefore, we have $OPT_{IS}(\mathcal{D}) = w(\mathcal{D}) - OPT_{VC}(\mathcal{D})$. Nevertheless, taking the complement of the solution output by a ρ-approximation algorithm for MWIS does in general not provide a ρ-approximation for MWVC, and vice versa.

In general graphs with n vertices, there cannot be a polynomial-time approximation algorithm for MWIS with approximation ratio $n^{1-\varepsilon}$ for any $\varepsilon > 0$ unless $\mathcal{NP}$ = co-RP [9]. MWVC is MAX SNP-hard in general graphs and cannot be approximated within a constant smaller than 7/6 unless $\mathcal{P} = \mathcal{NP}$ [10]. For MWVC in general graphs, a 2-approximation algorithm is known [3]. For intersection graphs of geometric objects, better approximation ratios are often possible.

1.2 Related Work on Disk Graphs. For unit disk graphs (intersection graphs of disks with equal diameter), MWIS and MWVC remain $\mathcal{NP}$-hard [5], but PTASs exist for MWIS, MWVC, and the minimum dominating set problem [12]. For intersection graphs of disks with arbitrary diameters, the best previously known approximation algorithms achieve approximation ratio 5 for MIS [17] and $\frac{3}{2}$ for MWVC [16]. In [12] and [16], the question was raised whether a PTAS exists for disk graphs. As the class of disk graphs contains the class of unit disk graphs and the class of planar graphs, a PTAS for disk graphs would generalize the results of [12] and [2]. This paper resolves this question by presenting PTASs for MWIS and MWVC in disk graphs.

A PTAS for the fractional chromatic number problem on disk graphs, using our PTAS for MWIS as a subroutine, was obtained in [13].

1.3 Related Work on Map Labeling. Map labeling refers to a family of tasks concerning the placement of labels on a map. Often it is assumed that the features to be labeled are points and that the labels can be modeled as rectangles (just take the bounding box of the label text). For each feature, there are certain admissible positions of the labeling rectangle: for example, the rectangle must be placed so that the feature coincides with a corner of the rectangle. Rectangles can be assigned weights that represent the importance of including that label on the map. Then it is meaningful to study the problem of maximizing the total weight of labeled features subject to the constraint that different labels must not overlap. This is just the maximum weight independent set problem in the intersection graph of a set of axis-aligned rectangles.

Agarwal et al. [1] give an $O(\log n)$-approximation algorithm for MIS in an intersection graph of n axis-aligned rectangles. Their algorithm can be adapted to the weighted case in a straightforward way, thus yielding approximation ratio $O(\log n)$ also for MWIS. For the special case that all rectangles have the same height (which is meaningful if the labels are text labels of a certain font size), they obtain a PTAS.

Doddi et al. [6] consider *sliding labels* (a point can lie anywhere on the boundary of its label) and assume that labels may be placed in any orientation. They provide constant-factor approximation algorithms for maximizing the size of the labels (assuming that all features must be labeled and that all labels are circles or squares with identical size). These were recently improved in [7]. In [6], bicriteria approximation algorithms that label a $(1-\varepsilon)$-fraction of all features with labels whose size if at least a $\frac{1}{1+\varepsilon}$-fraction of the optimal size are also discussed. These algorithms use a PTAS for MWIS as a subroutine. In [6], it is mentioned that their algorithms can be extended to the case of non-uniform squares if the ratio between the size of the largest square and the smallest square is bounded by a constant. Using our new PTAS for MWIS in the intersection graphs of squares, their algorithms can now be extended to labeling with non-uniform squares where this ratio is arbitrary.

Van Kreveld et al. [19] investigate the question of how many features in a map can be labeled if different restrictions on the rectangular labels are enforced (feature must be at a corner of the rectangle vs. sliding rectangles). They present a practical 2-approximation algorithm and a PTAS for maximizing the total weight of the labeled features in the case of sliding labels if all rectangles have the same height.

An up-to-date bibliography of publications on map labeling can be found on the web [20].

1.4 Our Results. In this paper we present PTASs for MWIS and MWVC in the intersection graphs of "disk-like" objects. In §2 and §3, we present the details of the PTASs for MWIS and MWVC in disk graphs. In §4, we discuss how our approach can be extended to other geometric objects (such as squares or regular polygons) and to higher dimensions. We mention an open problem in §5.

Our PTASs are based on a sophisticated use of the *shifting strategy* [11] that was previously employed, among other results, for obtaining PTASs for various optimization problems in planar graphs [2] and unit disk graphs [12]. We partition the given disks into levels according to their diameters and use a novel recursive

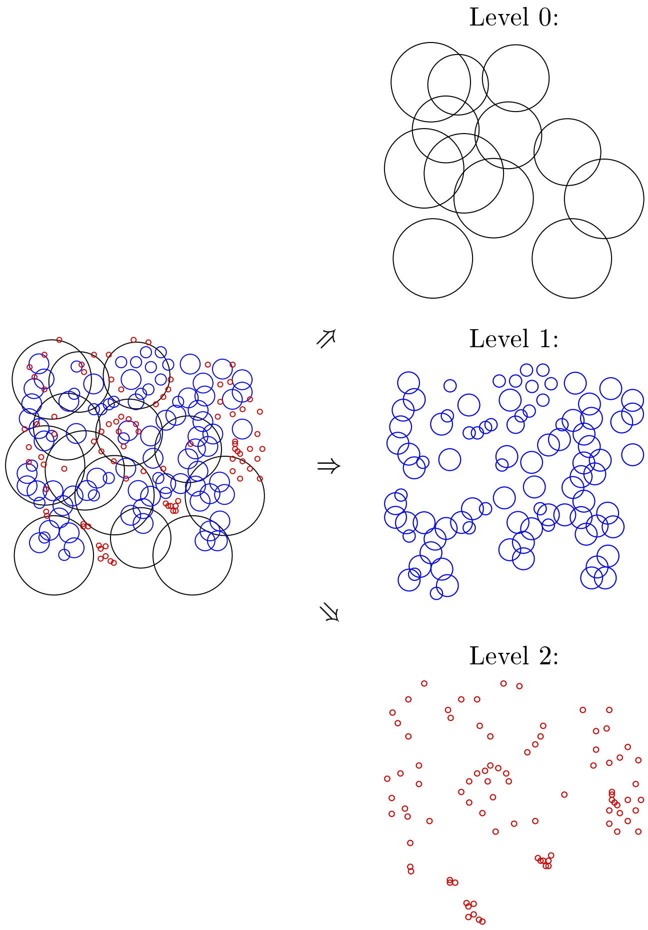

Figure 1: Partitioning the disks into levels.

subdivision of the plane that allows us to apply the shifting strategy on all levels simultaneously.

We outline the basic idea of the PTAS for MWIS. The plane is partitioned into squares on each level and some of the disks are removed from the input so that different squares on the same level yield independent subproblems with respect to all disks that are on this level or on a level with disks of smaller diameter. Furthermore, at most a constant number of disks with larger diameter can be disjoint and intersect a square on the current level. Hence, all such sets of disks can be enumerated in polynomial time for each square, and a dynamic programming approach becomes feasible. The details are given in the next section.

2 A PTAS for Independent Set in Disk Graphs

Let $k > 1$ be a fixed positive integer. Scale all disks so that the largest disk has diameter 1. Let $d_{\min}$ be the diameter of the smallest disk. Let $\ell = \lfloor \log_{k+1}(1/d_{\min}) \rfloor$. We partition the given set $\mathcal{D}$ of disks into $\ell + 1$ levels. For $0 \le j \le \ell$, level j consists of all disks D_i with diameter d_i satisfying $(k+1)^{-j} \ge d_i > (k+1)^{-(j+1)}$. Note that the disk with diameter $d_{\min}$ is on level ℓ.

An example with three levels is sketched in Fig. 1.

2.1 Subdividing the Plane. For each level j, $0 \le j \le \ell$, we impose a grid on the plane that consists of lines that are $(k+1)^{-j}$ apart from each other. The v-th vertical line, $-\infty < v < \infty$, is at $x = v(k+1)^{-j}$. The h-th horizontal line, $-\infty < h < \infty$, is at $y = h(k+1)^{-j}$. We say that the v-th vertical line has index v and that the h-th horizontal line has index h. Furthermore, we say that a disk D_i with center (x_i, y_i) and diameter d_i *hits* a vertical line at $x = a$ if $a - d_i/2 < x_i \le a + d_i/2$. Similarly, we say that D_i hits a horizontal line at $y = b$ if $b - d_i/2 < y_i \le b + d_i/2$. Intuitively, a disk hits a line if it intersects that line, except if it only touches the line from the left or from below. Note that every disk can hit at most one horizontal line and at most one vertical line on its level.

Let $0 \le r, s < k$ and consider the vertical lines whose index modulo k equals r and the horizontal lines whose index modulo k equals s. We say that these lines are *active* for (r, s). Figure 2 illustrates the horizontal grid lines and active lines on two consecutive levels.

Define $\mathcal{D}(r, s)$ to be the set of disks that is obtained from $\mathcal{D}$ by deleting all disks that hit a line that is on the same level as the disk and that is active for (r, s). See Fig. 3 for an example.

In the following, we write OPT as a short-hand for OPT_{IS}.

LEMMA 2.1. *For at least one pair (r, s), $0 \le r, s < k$, we have $OPT(\mathcal{D}(r, s)) \ge (1 - \frac{1}{k})^2 OPT(\mathcal{D})$.*

Proof. Let $S^* \subseteq \mathcal{D}$ be any set of disjoint disks with total weight $OPT(\mathcal{D})$. For $0 \le r < k$, let S_r^* be the set of all disks in S^* that hit a vertical line on their level whose index modulo k is r. As the sets S_r^* are disjoint, the weight of at least one of them must be at most a $\frac{1}{k}$-fraction of the weight of S^*. For this set S_r^*, let $T^* = S^* \setminus S_r^*$ and note that the weight of T^* is at least $(1 - \frac{1}{k}) OPT(\mathcal{D})$.

For $0 \le s < k$, let T_s^* be the set of all disks in T^* that hit a horizontal line on their level whose index modulo k is s. The weight of at least one of these sets T_s^* must be at most a $\frac{1}{k}$-fraction of the weight of T^*. For this set T_s^*, let $U^* = T^* \setminus T_s^*$. Note that $U^* \subseteq \mathcal{D}(r, s)$ and the weight of U^* is at least $(1 - \frac{1}{k})^2 OPT(\mathcal{D})$. □

The algorithm considers all k^2 possible values for r and s such that $0 \le r, s < k$. For each possibility, an optimal independent set in $\mathcal{D}(r, s)$ is computed using dynamic programming. Among the k^2 sets obtained in this way, the one with largest weight is output.

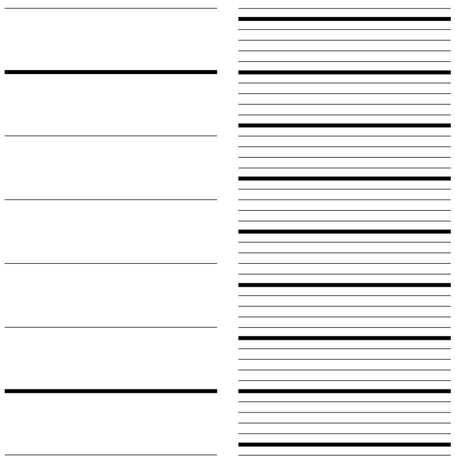

Figure 2: Horizontal grid lines on level j (left-hand side) and level $j+1$ (right-hand side) for $k=5$. Active lines are drawn bold.

By Lemma 2.1, this set has total weight at least $(1-\frac{1}{k})^2 OPT(\mathcal{D})$. Therefore, the algorithm achieves approximation ratio $(1+\frac{1}{k-1})^2$. As k gets larger, the approximation ratio gets arbitrarily close to 1.

It remains to show how an optimal independent set in $\mathcal{D}(r,s)$ can be computed using dynamic programming and that the running-time of the algorithm is polynomial in the size of the input for a fixed value of k.

2.2 Dynamic Programming. Let $0 \le r,s < k$. Consider one particular level j, $0 \le j \le \ell$. The lines on level j that are active for (r,s) partition the plane into squares. More precisely, for consecutive active vertical lines at $x = a_1$ and $x = a_2$ and consecutive active horizontal lines at $y = b_1$ and $y = b_2$, one square $\{(x,y) \mid a_1 < x \le a_2, b_1 < y \le b_2\}$ is obtained. We refer to these squares on level j as *j-squares*.

By definition of $\mathcal{D}(r,s)$, every disk in $\mathcal{D}(r,s)$ that is on level j is completely contained in some j-square. Furthermore, we have the following lemma about the relationship between squares on different levels.

LEMMA 2.2. *For any j, $0 \le j < \ell$, every $(j+1)$-square is completely contained in some j-square.*

Proof. We prove the lemma by showing that every line that is active for (r,s) on level j is also active for (r,s) on level $j+1$. Figure 2 illustrates this claim for the horizontal lines: note that the active lines on level j are active on level $j+1$ also after the active lines are "shifted" up or down on their respective levels.

Without loss of generality, consider only the horizontal lines. Let $y = h(k+1)^{-j}$ be an active horizontal line on level j. This means that $h \bmod k = s$. This line

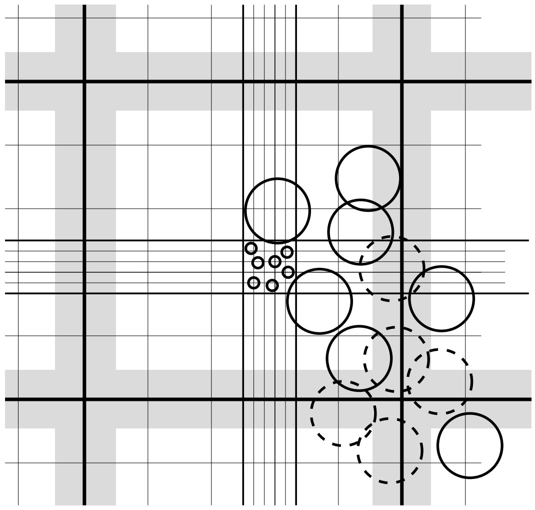

Figure 3: Example of grid and active lines on level j (coarse grid) and on level $j+1$ (fine grid) for $k=5$. The big disks have level j, the small disks have level $j+1$. All disks shown have the maximum possible diameter on their level. Active lines are drawn bold. Disks that hit active lines are drawn dashed. Note that a disk on level j can hit an active line only if its center is in the strip along that active line that is drawn shaded.

is identical to the line $y = h(k+1)(k+1)^{-(j+1)}$, which is on level $j+1$ and has index $h(k+1)$. Obviously, $h(k+1) \bmod k = h \bmod k = s$. Hence, the line is also active on level $j+1$. $\square$

Next, we show that the number of disjoint disks of level at most j that can intersect a j-square is bounded by a constant. Figure 3 can serve as an illustration of this fact: only $O(k^4)$ disjoint disks whose diameter is larger than that of the small disks can intersect the j-square shown in the figure.

LEMMA 2.3. *Let S be some j-square and let $I \subseteq \mathcal{D}$ be a set of disjoint disks such that each disk in I has level at most j and intersects S. Then there is a constant C (depending only on k) such that $|I| \le C$.*

Proof. Let S' be a square that consists of S and a strip of width $(k+1)^{-j}$ surrounding S. As the disks in I have level at most j, they have diameter at least $(k+1)^{-(j+1)}$ and, therefore, area at least $\pi((k+1)^{-(j+1)}/2)^2$. Furthermore, each disk in I occupies area at least $\pi((k+1)^{-(j+1)}/2)^2$ within S': disks of diameter $(k+1)^{-(j+1)}$ that intersect S must be completely contained in S', and larger disks that intersect S occupy an area of S' that is even larger. The area of S' is $((k+2)(k+1)^{-j})^2$. Therefore, we

```
TableS ← ∅;
R ← all disks in D(r,s) of level ≤ j intersecting S;
for all J ⊆ R such that |J| ≤ C do
    if the disks in J are disjoint then
        X ← {D ∈ J | D has level j};
        for all child squares S' of S do
            I' ← {D ∈ J | D intersects S'};
            X ← X ∪ TableS'(I');
        od;
        I ← {D ∈ J | D has level smaller than j};
        if TableS(I) is undefined or
           w(X) > w(TableS(I)) then
            TableS(I) ← X;
        fi
    fi
od
```

Figure 4: Computing Table_S for a relevant j-square S.

must have:

$$|I| \le \frac{((k+2)(k+1)^{-j})^2}{\pi((k+1)^{-(j+1)}/2)^2} = \frac{4}{\pi}(k+2)^2(k+1)^2 = O(k^4)$$

Hence, we can choose $C = O(k^4)$. □

Now we are ready to discuss the dynamic programming algorithm for computing an optimal independent set in $\mathcal{D}(r,s)$. Call a j-square S *relevant* if $\mathcal{D}(r,s)$ contains at least one disk of level j that is contained in S. For a relevant j-square S and a relevant j'-square S', $j' > j$, we say that S' is a *child* or *child square* of S (and S is a *parent* of S') if S' is contained in S and if there is no relevant j''-square S'', $j' > j'' > j$, such that S' is contained in S'' and S'' is contained in S.

The algorithm processes all relevant squares in order of non-increasing levels. While a j-square S is processed, a table Table_S is computed. For every set I of disjoint disks of level smaller than j that intersect S, the table entry $\mathit{Table}_S(I)$ is a maximum-weight set of disjoint disks of level at least j that are contained in S and disjoint from the disks in I. In order to compute the table Table_S, all sets J containing at most C disks (where C is the constant from Lemma 2.3) of level at most j that intersect S are enumerated. For each such set that consists of disjoint disks, the optimal way of extending the set by adding disks of larger levels that are contained in S is computed by looking up the respective sets stored in the tables at children of S. The pseudocode for the computation of Table_S is shown in Fig. 4. An example of the table lookups performed for a square S and a set J of two disjoint disks intersecting S is sketched in Fig. 5.

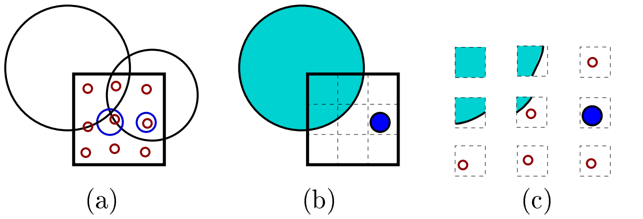

Figure 5: Example of table lookups for a square S at level j. (a) shows 13 disks in $\mathcal{D}(r,s)$ that intersect S: 2 disks on smaller levels, 2 disks on level j, and 9 disks on level $j+1$. (b) displays a set J of two disjoint disks of level at most j. (c) illustrates the 9 table lookups that are performed for this square and this set J if $k = 2$ and all subsquares on level $j+1$ are relevant.

When the tables for all relevant squares have been computed, the algorithm takes the union of the sets $\mathit{Table}_S(\emptyset)$ for all relevant squares S that do not have a parent. We claim that the result is a maximum weight independent set in $\mathcal{D}(r,s)$.

Now we prove the correctness of the algorithm.

Lemma 2.4. *Assume that the algorithm has processed a relevant j-square S and that I is a set of disjoint disks of smaller levels than j that intersect S. Then $\mathit{Table}_S(I)$ has the following properties:*

(a) $\mathit{Table}_S(I) \subseteq \mathcal{D}(r,s)$ consists of disks that are contained in S and have level at least j.

(b) $I \cup \mathit{Table}_S(I)$ is an independent set.

(c) $w(\mathit{Table}_S(I))$ is maximum among all sets that satisfy (a) and (b).

Proof. The proof is by induction on the number of squares processed by the algorithm. Initially, the claim of the lemma holds vacuously. Now assume that the algorithm processes a relevant j-square S, and let I be a set of disjoint disks of smaller levels than j that intersect S. Let I^* be a maximum-weight set of disks that are contained in S and whose level is at least j such that $I \cup I^*$ is an independent set. Let $w^* = w(I^*)$, and let J^* be the disks in I^* of level j. Then the set $I \cup J^*$ is one of the sets J enumerated in the outer loop of the algorithm in Fig. 4. Furthermore, every disk in $I^* \setminus J^*$ is contained in a relevant child square of S, and the relevant child squares of S are disjoint. By the inductive hypothesis, the tables computed at children of S are correct. Therefore, in the iteration with $J = I \cup J^*$, the algorithm obtains a set X of total weight w^* and we conclude $w(\mathit{Table}_S(I)) \ge w^*$. On the other hand, the child squares of S are disjoint, and so all table entries computed by the algorithm are indeed sets of disjoint disks. In particular, $I \cup \mathit{Table}_S(I)$ is an independent set. This shows $w(\mathit{Table}_S(I)) \le w^*$ and, therefore, $w(\mathit{Table}_S(I)) = w^*$. The algorithm computes correct entries for Table_S, and the induction goes through. □

We observe that all disks in $\mathcal{D}(r,s)$ are contained in some relevant square without parent, and that all relevant squares without parent are disjoint. By Lemma 2.4, this shows that the algorithm indeed computes an optimal independent set in $\mathcal{D}(r,s)$. It remains to consider the running-time.

LEMMA 2.5. *For a fixed $k > 0$, the running-time of the algorithm is polynomial.*

Proof. As there are n disks, there can be at most n relevant squares. The relevant squares and their forest structure (the links from every relevant square to its children) can be computed in polynomial time. For each relevant square S, the algorithm shown in Fig. 4 must enumerate $O(n^C)$ sets and perform, for each set, $O(n)$ lookups at children of S. Each lookup takes polynomial time. The whole algorithm is executed for k^2 possible values of (r,s). As $C = O(k^4)$, the overall running-time is $k^2 n^{O(k^4)}$. □

We summarize our result in the following theorem.

THEOREM 2.1. *There is a polynomial-time approximation scheme for the maximum weight independent set problem in disk graphs, provided that a disk representation of the graph is given.*

3 A PTAS for Vertex Cover in Disk Graphs

In this section we describe a PTAS for MWVC. The partitioning of the disks into levels and the subdivision of the plane into squares at each level is the same as for MWIS. Again, all values of r and s such that $0 \le r, s < k$ are considered in turn. Note that the definition of the squares on each level depends on r and s. Contrary to the PTAS for MWIS, disks that are not completely contained in some square on their level are not removed; instead, these disks are considered in all squares on their level that they intersect (there are at most four such squares). Now a j-square S is called relevant if $\mathcal{D}$ contains a disk of level j that intersects S.

Consider some j-square S and let Y^S denote the set of all disks in $\mathcal{D}$ that intersect S. Let $Y^S_{<j}$ be the set of disks in Y^S that have level smaller than j. Define $Y^S_{\le j}$, $Y^S_{=j}$, $Y^S_{\ge j}$, and $Y^S_{>j}$ analogously.

We say that a set $Z \subseteq Y^S$ is a *pseudo-cover* of S if, for any two disks $D_1, D_2 \in Y^S$ that intersect in S (i.e., $D_1 \cap D_2 \cap S \ne \emptyset$), Z contains D_1 or D_2 (or both). Note that a pseudo-cover of S need not contain D_1 or D_2 if the intersection of D_1 with D_2 is outside S. We observe that, for any two disks in $\mathcal{D}$ that intersect, there exists a relevant square in which they intersect.

```
Table_S ← ∅;
Y^S_{≤j} ← all disks in D of level at most j intersecting S;
for all J ⊆ Y^S_{≤j} such that |J| ≤ C do
    if the disks in J are disjoint then
        Q ← Y^S_{≤j} \ J;
        X ← {D ∈ Q | D has level j};
        for all child squares S' of S do
            P' ← {D ∈ Q | D intersects S'};
            X ← X ∪ Table_{S'}(P');
        od;
        P ← {D ∈ Q | D has level smaller than j};
        if Table_S(P) is undefined or
          w(X) < w(Table_S(P)) then
            Table_S(P) ← X;
        fi
    fi
od
```

Figure 6: Modified algorithm for MWVC.

For any pseudo-cover Z of S, call $Y^S_{<j} \cap Z$ the *projection* of Z onto $Y^S_{<j}$ (and similarly for $Y^S_{\le j}$). While processing S, the algorithm considers only pseudo-covers Z of S for which the projection of Z onto $Y^S_{\le j}$ equals $Y^S_{\le j} \setminus I$ for some independent set $I \subseteq Y^S_{\le j}$. As the number of independent sets in $Y^S_{\le j}$ is bounded by $O(n^C)$ by Lemma 2.3, we can enumerate all of them and, by taking the complements, the corresponding projections of pseudo-covers of S onto $Y^S_{\le j}$ in polynomial time.

As in the PTAS for MWIS, all relevant squares are processed in a bottom-up fashion. At a relevant j-square S, the projections of pseudo-covers of S onto $Y^S_{\le j}$ that are complements of independent sets are enumerated. During this enumeration process, a table Table_S is constructed by looking up the respective entries stored in tables at children of S. The entry $\mathit{Table}_S(P)$, for a projection $P \subseteq Y^S_{<j}$ of a pseudo-cover of S onto $Y^S_{<j}$, is a set $X \subseteq Y^S_{\ge j}$ such that $P \cup X$ is a pseudo-cover of S. Unlike in the PTAS for MWIS, the entries $\mathit{Table}_S(P)$ will not be optimal (minimum-weight) such sets X in general, but still good enough for our purposes.

Figure 6 shows the pseudocode for computing Table_S once the tables $\mathit{Table}_{S'}$ have been computed for all children S' of S. Again, C is the constant from Lemma 2.3. In the end, the algorithm outputs the union of the sets $\mathit{Table}_S(\emptyset)$, taken over all relevant squares S that do not have a parent. We claim that this is a

$(1 + \frac{6}{k})$-approximation of the minimum weight vertex cover.

In order to analyze this algorithm, we present several lemmas.

LEMMA 3.1. *The algorithm outputs a vertex cover of $\mathcal{D}$.*

Proof. Let S be some j-square and let Y^S be the set of all disks in $\mathcal{D}$ that intersect S. Recall that $Y^S_{<j}$ denotes the set of disks in Y^S whose level is smaller than j, and similarly for $Y^S_{\leq j}$, $Y^S_{=j}$, $Y^S_{\geq j}$, and $Y^S_{>j}$. For every set I of disjoint disks in $Y^S_{<j}$, we show that the table entry $\mathit{Table}_S(P)$ for $P = Y^S_{<j} \setminus I$ is indeed a set $X \subseteq Y^S_{\geq j}$ such that $P \cup X$ is a pseudo-cover of S. The proof is by induction on the number of relevant squares processed by the algorithm.

Assume that the entry $\mathit{Table}_S(P)$ is modified by the algorithm of Fig. 6 in an iteration of the outer loop corresponding to a certain set $J \subseteq Y^S_{\leq j}$. Let $Q = Y^S_{\leq j} \setminus J$. Let $Q = Q_{=j} \cup Q_{<j}$, where $Q_{=j}$ and $Q_{<j}$ are the disks in Q of level equal to j and of level smaller than j, respectively. Note that $P = Q_{<j}$.

Consider a child S' of S with level $j' > j$. Let $Q^{S'}$ be the subset of Q containing all disks that intersect S'. By the inductive hypothesis, $\mathit{Table}_{S'}(Q^{S'})$ is a set $X_{S'} \subseteq Y^{S'}_{\geq j'}$ such that $Q^{S'} \cup X_{S'}$ is a pseudo-cover of S'. Obviously, the set assigned to $\mathit{Table}_S(P)$ is the set

$$X = Q_{=j} \cup \bigcup_{S' \text{ child of } S} X_{S'}.$$

Clearly, we have $X \subseteq Y^S_{\geq j}$. It remains to show that $Z = Q_{<j} \cup X$ is a pseudo-cover of S. Note that Z contains all sets $Q^{S'} \cup X_{S'}$ for children S' of S; hence, Z is a pseudo-cover of every child S' of S.

Now consider any two disks D_1 and D_2 that intersect in S. If D_1 or D_2 has level greater than j, then D_1 and D_2 intersect in at least one child S' of S; thus D_1 or D_2 is in Z, because Z is a pseudo-cover of S'. If both D_1 and D_2 have level at most j, then at least one of them is not contained in J (as J is an independent set) and, therefore, contained in Z. This shows that Z is a pseudo-cover of S.

In the end, the algorithm outputs a union of pseudo-covers in all relevant squares without parent. As every intersection is contained in some relevant square, this proves that the algorithm always outputs a vertex cover of $\mathcal{D}$. $\square$

Now let C^* be any optimal vertex cover of $\mathcal{D}$. Every disk hits at most one vertical line on its level and at most one horizontal line on its level. For any pair of values r and s, $0 \leq r, s < k$, let $C^*(r, s)$ be the set of all disks in C^* that hit a vertical line on their level whose index modulo k equals r or a horizontal line on their level whose index modulo k equals s. Obviously, there exist values of r and s such that the total weight of $C^*(r, s)$ is at most $\frac{2}{k}$ times the weight of C^*. Consider the subdivision of the plane that results from this choice of r and s.

Let $\mathcal{R}$ be the set of all relevant squares. For any j-square S, let $C(S)$ denote the disks in C^* that intersect S and that are on level j. Note that $C^* = \bigcup_{S \in \mathcal{R}} C(S)$.

LEMMA 3.2. *We have $\sum_{S \in \mathcal{R}} w(C(S)) \leq (1 + \frac{6}{k}) w(C^*)$.*

Proof. The only disks in C^* that are counted more than once in the sum on the left-hand side are those that intersect several squares on their level. However, any disk can intersect at most 4 squares on its level. Furthermore, only disks in $C^*(r, s)$ can intersect more than one square on their level. Thus, only disks of total weight $w(C^*(r, s)) \leq \frac{2}{k} w(C^*)$ are counted multiple times, and each disk is counted at most four times on the left-hand side, while it is counted once in $w(C^*)$. Thus $\sum_{S \in \mathcal{R}} w(C(S))$ can exceed $w(C^*)$ by at most $3\frac{2}{k} w(C^*)$, establishing the claimed inequality. $\square$

LEMMA 3.3. *Let A be the vertex cover output by the algorithm. Then $w(A) \leq \sum_{S \in \mathcal{R}} w(C(S))$.*

Proof. For any relevant j-square S, let C^S be the set of all disks in C^* that intersect S. Define $C^S_{<j}$, $C^S_{\leq j}$, $C^S_{=j}$, $C^S_{\geq j}$, and $C^S_{>j}$ as usual. Recall that $C(S) = C^S_{=j}$. We claim that, after Table_S has been computed, we have the following:

$$w(\mathit{Table}_S(C^S_{<j})) \leq \sum_{S' \prec S} w(C(S')) \tag{3.1}$$

Here, $S' \prec S$ means that S' is a relevant square that is contained in S. Note that $S \prec S$.

The proof is by induction on the number of relevant squares that were processed by the algorithm. Assume that the algorithm is about to process the relevant square S and that (3.1) holds for all squares that were processed before S. In one of the iterations of the outer loop in Fig. 6, we have $Q = C^S_{\leq j}$. In this iteration, the algorithm will look up the sets $\mathit{Table}_{S'}(C^{S'}_{\leq j})$ at all children S' of S. Assume that S' is a j'-square. As S' is a child of S, we have $C^{S'}_{\leq j} = C^{S'}_{<j'}$. By the inductive hypothesis, we know that $w(\mathit{Table}_{S'}(C^{S'}_{\leq j})) \leq \sum_{S'' \prec S'} w(C(S''))$. The set X computed in this iteration of the loop has weight at most

$$\sum_{S' \text{ child of } S} w(\mathit{Table}_{S'}(C^{S'}_{\leq j})) + w(C(S)) \leq \sum_{S' \prec S} w(C(S')).$$

Thus, we have shown that (3.1) is satisfied also for S. Finally, let $\mathcal{R}_0$ be the set of all relevant squares without parent. The algorithm outputs $\bigcup_{S\in\mathcal{R}_0} Table_S(\emptyset)$ as a solution. For any relevant j-square $S \in \mathcal{R}_0$, we have $C^S_{<j} = \emptyset$. Thus we get that $\sum_{S\in\mathcal{R}_0} w(Table_S(\emptyset)) \le \sum_{S\in\mathcal{R}} w(C(S))$. $\square$

Combining these lemmas, we obtain the following theorem.

THEOREM 3.1. *There is a polynomial-time approximation scheme for the minimum weight vertex cover problem in disk graphs, provided that a disk representation of the graph is given.*

4 Extension to Other Geometric Graphs

In the previous sections we have presented PTASs for MWIS and MWVC in the intersection graphs of disks with arbitrary diameters. Our approach does not make use of any specific properties of disks, it is only required that the given geometric objects can be partitioned into levels such that only a constant number of disjoint objects of level at most j can intersect a square on level j. Therefore, the same approach can be used for other geometric objects such as squares or regular polygons. We can also deal with rectangles if the ratio between the height and the width does not differ by more than a constant factor between different rectangles, because it suffices to scale the input along one axis so that the resulting rectangles are approximately squares.

Furthermore, the approach works for geometric objects in any d-dimensional space provided that d is a constant. The space is partitioned into d-dimensional cubes on each level, and instead of removing objects (in case of MWIS) that hit certain horizontal or vertical lines, we remove objects that hit certain hyperplanes.

More specifically, we obtain PTASs for MWIS and MWVC in the intersection graphs of geometric objects in d dimensions if the following conditions are satisfied (possibly after appropriate scaling):

- d is bounded by a constant.
- For each object i, a d-dimensional ball B_i that contains i can be determined in polynomial time.
- One can decide in polynomial time whether two of the given objects intersect and whether an object intersects an arbitrary cube.
- Let level j contain all given objects whose balls B_i have diameter d_i satisfying $(k+1)^{-j} \ge d_i > (k+1)^{-(j+1)}$. Then the number of disjoint objects of level at most j that can intersect a d-dimensional cube with side length $k(k+1)^{-j}$ must be bounded by a constant (for fixed k).

5 An Open Problem

Our approximation schemes use a partitioning of the plane into squares on each level, and this works only if the ratio of the height to the width is roughly the same for all given geometric objects. It is not clear whether the approach can be extended to the intersection graphs of arbitrary axis-parallel rectangles, for example. For computing a maximum independent set among n given rectangles, the best known approximation algorithm has approximation ratio $O(\log n)$ [1]. An open problem is to devise an approximation algorithm with constant approximation ratio or even a PTAS for this problem, or to provide evidence that MWIS in intersection graphs of arbitrary rectangles is substantially harder to approximate than in intersection graphs of squares or disks.

References

[1] P. K. Agarwal, M. van Kreveld, and S. Suri. Label placement by maximum independent set in rectangles. *Computational Geometry: Theory and Applications*, 11(3–4):209–218, 1998.

[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] R. Bar-Yehuda and S. Even. A local-ratio theorem for approximating the weighted vertex cover problem. *Annals of Discrete Mathematics*, 25:27–45, 1985.

[4] A. Brandstädt, V. B. Le, and J. P. Spinrad. *Graph Classes: A Survey*, volume 3 of *Monographs on Discrete Mathematics and Applications*. SIAM, 1999.

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

[6] S. Doddi, M. V. Marathe, A. Mirzaian, B. M. E. Moret, and B. Zhu. Map labeling and its generalizations. In *Proceedings of the 8th Annual ACM–SIAM Symposium on Discrete Algorithms SODA'97*, pages 148–157, 1997.

[7] S. Doddi, M. V. Marathe, and B. M. E. Moret. Point set labeling with specified positions. In *Proceedings of the 16th Annual ACM Symposium on Computational Geometry (SoCG'00)*, pages 182–190, 2000.

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

[9] J. Håstad. Clique is hard to approximate within $n^{1-\epsilon}$. In *Proceedings of the 37th Annual Symposium on Foundations of Computer Science FOCS'96*, pages 627–636, 1996.

[10] J. Håstad. Some optimal inapproximability results. In *Proceedings of the 29th Annual ACM Symposium on Theory of Computing STOC'97*, pages 1–10, 1997.

[11] 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.

[12] 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.

[13] K. Jansen and L. Porkolab. Polynomial time approximation schemes for preemptive resource constrained scheduling and fractional weighted colorings of geometric graphs. Unpublished manuscript, 2000.

[14] P. Koebe. Kontaktprobleme der konformen Abbildung. *Berichte über die Verhandlungen der Sächsischen Akademie der Wissenschaften, Leipzig, Math.-Phys. Klasse*, 88:141–164, 1936.

[15] J. Kratochvíl. Intersection graphs of noncrossing arc-connected sets in the plane. In *Proceedings of the Symposium on Graph Drawing (GD'96)*, LNCS 1190, pages 257–270, 1997.

[16] E. Malesińska. *Graph-Theoretical Models for Frequency Assignment Problems*. PhD thesis, Technische Universität Berlin, 1997.

[17] 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.

[18] T. A. McKee and F. R. McMorris. *Topics in Intersection Graph Theory*, volume 2 of *Monographs on Discrete Mathematics and Applications*. SIAM, 1999.

[19] M. van Kreveld, T. Strijk, and A. Wolff. Point labeling with sliding labels. *Computational Geometry: Theory and Applications*, 13(1):21–47, 1999.

[20] A. Wolff and T. Strijk. The map-labeling bibliography, 2000. http://www.math-inf.uni-greifswald.de/map-labeling/bibliography/.