Theoretical Computer Science 410 (2009) 756–765

Contents lists available at ScienceDirect

Theoretical Computer Science

journal homepage: www.elsevier.com/locate/tcs

A $5+\epsilon$-approximation algorithm for minimum weighted dominating set in unit disk graph

Decheng Dai [*,1], Changyuan Yu [1]

Institute for Theoretical Computer Science, Tsinghua University, Beijing, 100084, PR China

ARTICLE INFO

Article history:
Received 23 July 2008
Received in revised form 8 October 2008
Accepted 5 November 2008
Communicated by D.-Z. Du

Keywords:
Weighted unit disk graph
Dominating set
Approximation algorithm

ABSTRACT

We study the minimum weight dominating set problem in weighted unit disk graph, and give a polynomial time algorithm with approximation ratio $5+\epsilon$, improving the previous best result of $6+\epsilon$ in [Yaochun Huang, Xiaofeng Gao, Zhao Zhang, Weili Wu, A better constant-factor approximation for weighted dominating set in unit disk graph, J. Comb. Optim. (ISSN: 1382-6905) (2008) 1573–2886. (Print) (Online)]. Combining the common technique used in the above mentioned reference, we can compute a minimum weight connected dominating set with approximation ratio $9+\epsilon$, beating the previous best result of $10+\epsilon$ in the same work.

1. Introduction

In graph theory, the dominating set problem is one of the most famous problems, since it was studied from the 1950s onwards. In a given graph $G=(V,E)$, a node $v\in V$ dominates itself and all its neighbors $v\cup N(v)$ and a set of nodes $D\in V$ dominates $\{v\cup N(v)|v\in D\}$. When D dominates V, we say D is a dominating set of G. The minimum dominating set problem (MDS) is to find the smallest dominating set. In a weighted graph $G=(V,E,W)$, W is a mapping $V\mapsto R^*$ and every node is assigned a non-negative weight $W(v)$. The goal of the minimum weighted dominating set problem (MWDS) is to find the minimum weighted subset of nodes to dominate V.

A connected dominating set $D\in V$ is a dominating set of G whose reduced graph $(D,\{(u,v)\in E|u,v\in D\})$ is a connected graph. In the minimum connected dominating set problem (MCDS), the goal is to find the smallest connected dominating set. In the weighted graph we define the minimum weighted connected dominating set problem (MWCDS) in an obvious and similar way.

In 1979, Garey proved that MDS is *NP*-hard [7]. In 1984, Bar-Yehuda and Moran showed that minimum dominating set is polynomially equivalent to the set cover problem [8]. In 1999, Feige proved that $(1-o(1))\ln n$ is a threshold below which set cover cannot be approximated efficiently, unless $NP\subset DTIME[n^{O(\log\log n)}]$ [10]. Two years after that, Vazirani showed that there is no polynomial approximation algorithms can achieve an approximation ratio better than $O(\log n)$ [9]. Besides these hardness results, Guha and Khuller [11] gave an $O(\log n)$-approximation algorithm for MCDS in 1999.

1.1. Dominating set in unit disk graphs

Most problems become easier when they are restricted to planar graphs, because planar graphs not only satisfy the triangular inequality but also fix distances and angles in the graph between points. In our paper we consider MWDS and

* Corresponding author. Tel.: +86 13488787340.
E-mail addresses: ddc02@mails.tsinghua.edu.cn (D. Dai), yucy05@mails.tsinghua.edu.cn (C. Yu).

[1] Supported by the National Natural Science Foundation of China Grant (60553001), and the National Basic Research Program of China Grant (2007CB807900, 2007CB807901).

0304-3975/$ – see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2008.11.015

MWCDS in a more special graph: *Unit Disk Graph*. A unit disk graph [6] is the intersection graph of a family of unit circles in the Euclidean plane. In the unit disk graph, each vertex is associated with a unit circle in the plane and two vertices are adjacent if and only if the corresponding circles are intersected.

Routing in wireless ad-hoc networks is a most direct motivation to study dominating sets in unit disk graphs. [12] proposed the dominating sets for the construction of routing backbones in 2002. In wireless networks, sensors collect data and communicate to each other. Every senor is considered as a node in the graph. There is an edge between two nodes if and only if one sensor can cover another in its range and they can communicate with each other. The unit disk graph is the most simple model for sensor networks in which we consider all sensors are the same. Since messages are collected to key sensors and transmitted around the sensor network, a smaller connected dominating set gives a more energy-efficient routing in a wireless netowrk [13].

For unit disk graphs, there is some hardness results for MWDS since 1990. Clark et al. [6] showed that MDS in UDG is *NP-hard* in 1990, while earlier, Lichtenstein [3] proved that MWDS is *NP-hard* in UDG. Several algorithms were also presented in unit disk graphs. Marathe et al. [4] gave a constant-factor approximation algorithm for MDS and MCDS in UDG in 1995. Three years after that, Hunt et al. [5] gave a PTAS for MDS and MCDS. For the weighted unit disk graph, every disk has a non-negative weight. There are some recent results for this kind of graph. Ambühl et al. [1] gave the first constant-approximation algorithm for MWDS in UDG, and Huang et al. [2] improved their approximation ratio from 72 to $6 + \epsilon$. Meanwhile, Huang et al. [2] also gave a $10 + \epsilon$ approximation algorithm for MWCDS in UDG, improving the previous 89 approximation [1].

1.2. Our results

In this paper we present a 5-approximation algorithm for MWDS problem in unit disk graphs. The framework of the algorithm is divided into two phases, which is similar to [1,2]. In the first phase, we divide the plane into constant size pieces and reduce the MWDS into polynomial number of special cover problems in a constant-bounded area. A constant-bounded area means all the points/disks in the plane can be bounded by a constant-size square. The special cover problem means covering a set of points located in a constant size square by a minimum-weight set of double-size unit disks. Assuming the size of every square is a constant k, we only lose a small factor of $\frac{1}{k}$ in this reduction.

In the second phase , we give a new dynamic programming techniques to achieve a 5-approximation results for the divided problems and merging with the first phase, we achieve a $5 + \epsilon$ approximation algorithm for MWDS. In this phase, we improve the previous 6-approximation algorithm given in [2]. Compared to the algorithm in [2] by dividing every square into larger strips such that every disk intersects with fewer strips. Combining with the techniques in [2], which gives a 4-approximation ratio to connect the dominating set, our algorithm yields a $5 + \epsilon + 4 = 9 + \epsilon$ approximation algorithm for MWCDS in UDG.

1.3. Paper structure

We organize our paper as follows. In Section 2 we discuss the 2-StripWMDS problem and give a dynamic programming algorithm for this problem. In Section 3, we state the $5 + \epsilon$ algorithm for MWDS and prove its approximation ratio, which is our main result. Finally, Section 4 gives some conclusions and open problems for the approximation algorithms in unit disk graph.

2. Solving MWDS over a pair of strips

In this section we mainly discuss a dynamic programming algorithm for solving the minimum weight dominating set problem over a pair of adjacent strips in the plane. Before we give the algorithm, we start with some definitions about the dominating set problem.

2.1. Definitions

Definition 2.1 (*Minimum Weighted Dominating Set Problem in Unit Disk Graph*). We have n nodes in the plane, denoted by V. Each node i has a pair of coordinates (x_i, y_i), and is associated with a disk D_i, whose radius is 1 and weight is w_i (also denoted by $w(D_i)$). We refer the disks and nodes to the same thing. For any set S of nodes, its weight is defined as the total weight of the nodes in this set, i.e. $w(S) = \sum_{i \in S} w_i$. For any nodes $i, j \in V$, there exists an edge between them if and only if $dist(i, j) \leq 1$, where $dist(i, j)$ is the Euclidian distance, i.e. $dist(i, j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$. We denote the set of edges as E. When a pair of nodes i, j have an edge between them, we also say that i can be dominated or covered by j, or D_j alternatively. A subset $S \subseteq V$ is called a dominating set if for any $i \in V$, either $i \in S$ or there exists $j \in S$ such that $(i, j) \in E$. The objective of the Minimum Weighted Dominating Set Problem is to find a subset S, such that S is a dominating set with minimum weight. This problem is also denoted by **MWDS**.

Definition 2.2 (2-*StripMWDS*). We have n nodes, each associated with a weighted unit disk, and two adjacent strips, $T_1 = \{(x, y) | y_1 \leq y \leq y_2\}$ and $T_2 = \{(x, y) | y_2 \leq y \leq y_3\}$, where $y_1 < y_2 < y_3$. We use $S_{ka}, S_{kb}, k = 1, 2$ to denote the set of disks, whose centers are above the strip T_k and below the strip T_k, respectively. Assume that all the nodes in T_k

can be covered by the disks only from $S_k = S_{ka} \bigcup S_{kb}$. The objective is to find subsets of disks, $C_k \subset S_k$, $k = 1, 2$, such that $C = C_1 \bigcup C_2$ has minimum weight and nodes in T_k can be covered by disks only from C_k.

The following concept is very useful in our proofs.

Definition 2.3 (*Low-dominate*). For any given vertical line $l = \{(x, y)|x = x_0\}$ and disks D, D', disk D low-dominates D' on the line l if and only if one of the following conditions is satisfied,

- Both D and D' intersect with line $x = x_0$ and D has the lower intersection point;
- D intersects the line $x = x_0$ but D' doesn't;
- Neither D nor D' intersect the line $x = x_0$ but both of them are in the left hand side of $x = x_0$ and $x_D > x_{D'}$(or $x_D = x_{D'} \wedge y_D < y_{D'}$).

The concept of **up-dominate** can be defined similarly. By definition, the relation "low-dominate" and "up-dominate" satisfy the transitivity properties, that is, if D_1 low(up)-dominates D_2 and D_2 low(up)-dominates D_3 then D_1 low(up)-dominates D_3.

Given any instance of 2-**StripWMDS**, for any $D \in S_{ka}$ (or S_{kb}) and line $l : x = c$, we use $S_{ka}(D, l)$ (or$S_{kb}(D, l)$) to denote the set of all the disks from S_{ka} (or S_{kb}), which are low(up)-dominated by D on the line l.

2.2. Algorithm

For the 2-**StripMWDS** problem, we have the following dynamic programming algorithm to give the optimal solution in polynomial time. For each $i = 1, \ldots, n$, $k = 1, 2$, each $D_k \in S_{ka}$, $D'_k \in S_{kb}$, we define $C(i, D_1, D'_1, D_2, D'_2)$ as an optimal solution(as well as the optimal value) satisfying the following conditions,

1. Each node $p_j, j \leq i$, can be covered by some disk in $C(i, D_1, D'_1, D_2, D'_2)$, whose center is not in the strip where p_j lies.
2. $D_1, D'_1, D_2, D'_2 \in C(i, D_1, D'_1, D_2, D'_2)$;
3. For $k = 1, 2$, D_k low-dominates all the disks in $C(i, D_1, D'_1, D_2, D'_2) \bigcap S_{ka}$, and D'_k up-dominates all the disks in $C(i, D_1, D'_1, D_2, D'_2) \bigcap S_{kb}$.

If a set of disks only satisfies the above conditions but not the optimal solution, we call it a *feasible solution*. By calculating dynamically, the algorithm tries all feasible solutions, takes the optimal one, and maintains a table containing all the possible $C(i, D_1, D'_1, D_2, D'_2)$.

Algorithm 1: Solve the MWDS over a pair of strips.
Step 1: Label the nodes in the pair of strips $T = T_1 \bigcup T_2$ from left to right, and from up to down. So there are $p_1, \ldots p_n$ in T, and for any $i < j$, we have either $x_i < x_j$ or $(x_i = x_j) \wedge (y_i > y_j)$.
Step 2: Maintain a table containing all the possible $C(i, D_1, D'_1, D_2, D'_2)$, and calculate them according to Lemma 2.2.
Step 3: Output the minimum solution of $C(n, D_1, D'_1, D_2, D'_2)$, i.e. in the n-th column of the table, where the minimum is taken over all possible $D_1 \in S_{1a}, D'_1 \in S_{1b}, D_2 \in S_{2a}, D'_2 \in S_{2b}$.

2.3. Proofs

We first prove a lemma about the disk dominating property on a line. This is useful in proving the correctness of the above algorithm.

Lemma 2.1. *Given a strip and two lines $L_2 = \{(x, y)|x = x_2\}$, $L_3 = \{(x, y)|x = x_3\}$, where $x_2 < x_3$, D,D' two disks whose centers are above the strip. If D low-dominates D' on L_2, D' low-dominates D on L_3, then any node P in the strip with $x_p \leq x_2$ covered by D' can also be covered by D. (See Fig. 1 for example.)*

Proof. Assume that P is covered by D' but not by D. We must have $x_P < x_2$, since D low-dominates D' on line L_2. Then D' must intersect line $L_1 : x = x_P$ at some point B_1 in the strip. In line L_3, D' low dominates D, so if D' intersects with L_3, let B_3 denote the lowest intersection point. Otherwise, let B_3 denote point at which disk D' is tangent with L_3 if we move L_3 to left. Since the line segment B_1B_3 is lying in the disk D', D' must also intersect line L_2 at some point B_2 in the strip. Now we consider the disk D, since it dominates D' on line L_2, we know that it should intersect L_2 at some point A with $y_A \leq y_{B_2}$.

Since disk D has its center above the line $y = y_2$, and D cannot have intersection point below B_1, B_3 with line L_1, L_3 respectively, we know that the circle D must intersect the line segment $\overline{B_1B_3}$, say E, F. Circle D is the circumcircle of the triangle ΔAEF, so we have

$$R_D = \frac{|\overline{EF}|}{2 \sin \angle EAF}.$$

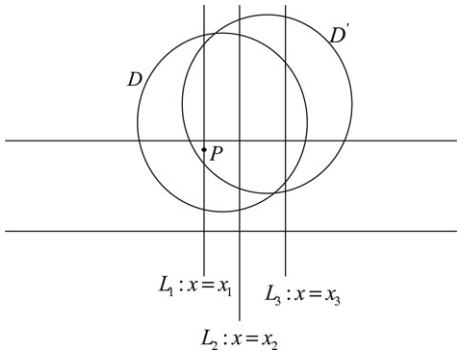

Fig. 1. Example of low-dominating transition.

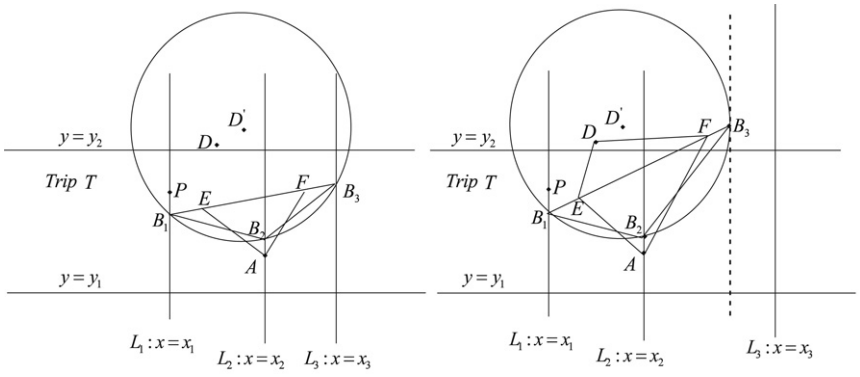

Fig. 2. Proof of dominating transition.

Similarly, we have

$$R_{D'} = \frac{|\overline{B_1B_3}|}{2\sin\angle B_1B_2B_3}.$$

Since the centers of D, D' are both lying above $\overline{B_1B_3}$, we know that $\angle EAF$, $\angle B_1B_2B_3$ are both larger than 90°. Now from $\angle EAF < \angle B_1B_2B_3$ and $|\overline{EF}| < |\overline{B_1B_3}|$, we obtain $R_D < R_{D'}$, which contradicts with the fact that all the disks have radius 1. (Fig. 2) □

Now we state the method of computing $C(i, D_1, D'_1, D_2, D'_2)$ in step 2 and its correctness by the following lemma. In the statement of this lemma we use the function $I(A)$: $I(A) = 1$ when A is a true statement, otherwise $I(A) = 0$.

Lemma 2.2. *We can compute all $C(i, D_1, D'_1, D_2, D'_2)$ as follows.*

- Case 1*: Let $C(i, D_1, D'_1, D_2, D'_2) = +\infty$, if one of the following happens:*
  - *p_i cannot be covered by any disk from D_1, D'_1, D_2, D'_2 with its center not in the strip where p_i lies;*
  - *$\exists k \in \{1, 2\}$, such that D_k cannot low-dominate $\{D_1, D_2\} \bigcap S_{ka}$ or D'_k cannot up-dominate $\{D'_1, D'_2\} \bigcap S_{kb}$.*
- Case 2*: let $C(i, D_1, D'_1, D_2, D'_2) = \sum_{k=1}^{2} \left(w(D_k) + w(D'_k)\right) - I(D_1 = D_2)w(D_1) - I(D'_1 = D'_2)w(D'_1)$, if $i = 1$ and one of D_1, D'_1, D_2, D'_2 covers p_1.*
- Case 3*: Use the following recursive relation to compute $C(i, D_1, D'_1, D_2, D'_2)$, if it falls into neither Case* 1 *nor Case* 2*.*

$$C\left(i, D_1, D'_1, D_2, D'_2\right) = \min_{\substack{D_{ks} \in S_{ka}(D_k, L_i) \\ D'_{ks} \in S_{kb}(D'_k, L_i)}} \left\{ C\left(i-1, D_{1s}, D'_{1s}, D_{2s}, D'_{2s}\right) + \sum_{k=1}^{2} \left(I\left(D_k \notin \{D_{1s}, D_{2s}\}\right) w(D_k) + I\left(D'_k \notin \{D'_{1s}, D'_{2s}\}\right) w(D'_k) \right) \right\}$$

Proof. In Case 1, if a quadruple (D_1, D'_1, D_2, D'_2) doesn't satisfy the second condition, it is impossible to find a solution for $C(i, D_1, D'_1, D_2, D'_2)$ satisfies to its definition. Now consider the case when this quadruple satisfies the second condition but

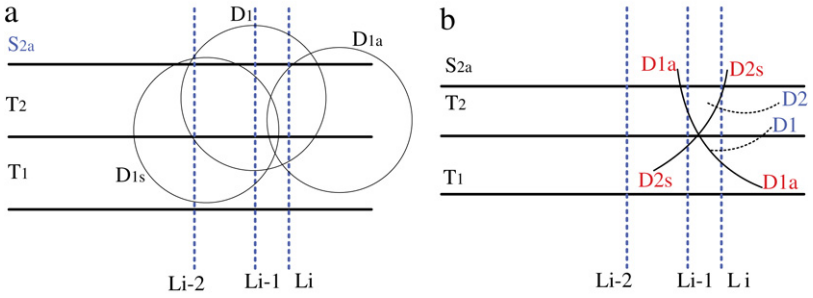

Fig. 3. Idea of the proof of low-dominating relations.

not the first one. W.L.O.G. , assume that p_i locates in T_1. Consider a disk $\tilde{D}$ not in these four disks, because $\tilde{D}$ is either low-dominated by D_1 or up-dominated by D'_1, it is impossible for $\tilde{D}$ to cover p_i. So in this case p_i isn't dominated by any disk and there doesn't exist a solution for $C(i, D_1, D'_1, D_2, D'_2)$.

In Case 2, it is the trivial case and the quadruple (D_1, D'_1, D_2, D'_2) must be the only optimal solution, so the algorithm takes it as the initial step of the whole dynamic programming.

For Case 3, we prove the equation as follows. We first prove $LHS \leq RHS$ by showing every solution enumerated in *RHS* is a feasible solution to $C(i, D_1, D'_1, D_2, D'_2)$. *LHS* is no more than *RHS* because *LHS* is optimal among all feasible solutions. In the second part we prove $LHS \geq RHS$. Actually, we want to prove in *RHS*, the corresponding optimal solution $C(i-1, D_{1s}, D'_{1s}, D_{2s}, D_{2s})$ doesn't contain D_k, when $D_k \notin \{D_{1s}, D_{2s}\}$. However we proved it in the opposite view. In the proof we divide the *LHS* into two parts C_{i-1} and something else, then constructing a feasible solution C_{i-1} to $C(i-1, D_{1s}, D'_{1s}, D_{2s}, D'_{2s})$ which doesn't contain D_k when $D_k \notin \{D_{1s}, D_{2s}\}$. Because *RHS* takes the minimum among all equations in this form, so $LHS \geq RHS$.

(1) To prove the Left hand side(LHS) is no larger than the right hand side(RHS), we first fix any $D_{1s}, D'_{1s}, D_{2s}, D'_{2s}$, and a corresponding optimal solution $C\left(i-1, D_{1s}, D'_{1s}, D_{2s}, D'_{2s}\right)$, which achieves the minimum of RHS. Consider the set of disks $C_i = C\left(i-1, D_{1s}, D'_{1s}, D_{2s}, D'_{2s}\right) \bigcup \{D_1, D'_1, D_2.D'_2\}$. We prove that C_i satisfies the following conditions:

1. Each node $p_j, j \leq i$, can be covered by some disk in C_i, with center not in the strip where p_j lies.
2. $D_1, D'_1, D_2, D'_2 \in C_i$;
3. For $k = 1, 2, D_k$ low-dominates all the disks in $C_i \bigcap S_{ka}$, and D'_k up-dominates all the disks in $C_i \bigcap S_{kb}$.

The first two conditions hold obviously by the construction of C_i and the assumption of Case 3. For the third condition, we prove D_k low-dominates $C_i \bigcap S_{ka}$, and D'_k up-dominates $C_i \bigcap S_{kb}$ can be proved similarly.

Viewing two strips as a single strip, we have D_2 low-dominates D_{2s} on L_i, and D_{2s} low-dominates $C\left(i-1, D_{1s}, D'_{1s}, D_{2s}, D'_{2s}\right) \bigcap S_{2a}$ on L_{i-1}. Assume that D_2 cannot low-dominate D_{2a} on L_i, and $D_{2a} \in C\left(i-1, D_{1s}, D'_{1s}, D_{2s}, D'_{2s}\right) \bigcap S_{2a}$. Then we must have D_{2a} low-dominates D_2, hence low-dominates D_{2s} on L_i too. So by Lemma 2.1, we know that any point in the strip and to the left of L_{i-1} covered by D_{2a} must be covered by D_{2s}, too. Now, if we delete D_{2a} from $C\left(i-1, D_{1s}, D'_{1s}, D_{2s}, D'_{2s}\right)$, it is still a feasible solution under the three conditions. This contradicts the fact that $C\left(i-1, D_{1s}, D'_{1s}, D_{2s}, D'_{2s}\right)$ is optimal. By the assumption of Case 3, we also have D_2 low-dominates $\{D_1, D_2\} \bigcap S_{2a}$ on L_i. Therefore, D_2 low-dominates $C_i \bigcap S_{2a}$ on L_i.

We are only left to prove D_1 low-dominates $C_{i-1} \cap S_{1a}$ on L_i. Suppose there exists a disk D_{1a} which is not low-dominated by D_1 on L_i. The center of D_{1a} must be in strip T_2 or above strip T_2(we denote this area as S_{2a}).

1. D_{1a} in T_2: As the assumption, D_{1a} can not be chosen to dominate any disk in T_2 when the center of D_{1a} is in T_2, it can be used to dominate the disks in T_1 only. In other words, any node to the left of L_{i-1} in T_2 must be covered by some other disk in $C(i, D_1, D'_1, D_2, D'_2)$. By applying the similar argument for D_{2a}, any node in T_1 which is to the left of L_{i-1} and covered by D_{1a} is covered by D_{1s}. So we conclude that D_{1a} can be removed to achieve a better optimal solution than $C(i-1, D_{1s}, D'_{1s}, D_{2s}, D'_{2s})$, which draws a contradiction.
2. D_{1a} in S_{2a}: Now we consider the situation when the center of D_{1a} is located in S_{2a}. Using the same technique we can show that any disk in T_1 covered by D_{1a} is covered by some other disk in $C(i, D_1, D'_1, D_2, D'_2)$. So are the disks in T_2 as follows. In L_i, D_{1a} low-dominates D_1, D_1 low-dominates D_2(by definition, it is impossible for D_2 to low-dominate D_1, so D_1 must low-dominates D_2), and D_2 low-dominates D_{2s}, so D_{1a} low-dominates D_{2s} in L_i. Because in L_{i-1}, D_{2s} low-dominates D_{1a}, by Lemma 2.1 we conclude that every node to the left of L_{i-1} in T_2 covered by D_{1a} can also be covered by D_{2s}. D_{1a} can be removed to achieve a better solution for $C(i-1, D_{1s}, D'_{1s}, D_{2s}, D'_{2s})$, which is a contradiction.

The idea of the analysis above is showed in Fig. 3, (a) shows the case D_{1a} in T_2 and (b) shows the case D_{1a} above T_2 (For continence, in (b) we draw only pieces of the disks but not whole disks). As showed in Fig. 3(b), on L_i, D_{1a} low-dominates D_1 low-dominates D_2 low-dominates D_{2s} on $T_1 \cup T_2$, but D_{2s} low-dominates D_{1a} on L_{i-1} on T_1. In this case Lemma 2.1 is used

for T_2. So we prove that D_1 low-dominates $C_i \bigcap S_{1a}$. That is to say, for any $D_{1s}, D'_{1s}, D_{2s}, D'_{2s}$, the corresponding C_i is a feasible solution under the conditions in the definition of $C\left(i, D_1, D'_1, D_2, D'_2\right)$, hence we have

$$weight(C_i) \geq C\left(i, D_1, D'_1, D_2, D'_2\right)$$

And taking the minimum of all $weight(C_i)$ over all possible $D_{1s}, D'_{1s}, D_{2s}, D'_{2s}$ we have

$$RHS \geq C\left(i, D_1, D'_1, D_2, D'_2\right)$$

(2) For the other direction, we show $LHS \geq RHS$. We fix the optimal solution C_i in LHS, and prove it can be decomposed into the form of RHS. For this decomposition, we find out a quadruple $(D_{1s}, D'_{1s}, D_{2s}, D_{2s})$, generate a set of disks $C_{i-1} \subseteq C_i$ and proved C_{i-1} is a feasible solution to $C(i-1, D_{1s}, D'_{1s}, D_{2s}, D_{2s})$. By showing $RHS = \min(w(C_{i-1}) + w(C_i - C_{i-1}))$ we finish the proof in this direction. Fix any optimal solution $C\left(i, D_1, D'_1, D_2, D'_2\right)$, we prove it can be decomposed into the form of RHS. We first construct a set of disks C_{i-1} as the following steps:

- if D_k low-dominates $C\left(i, D_1, D'_1, D_2, D'_2\right) \bigcap S_{ka}$ in L_{i-1}, let $D_{ks} = D_k$; Otherwise select D_{ks} in $C\left(i, D_1, D'_1, D_2, D'_2\right) \bigcap S_{ka}$ such that it dominates all the other disks in $C\left(i, D_1, D'_1, D_2, D'_2\right) \bigcap S_{ka}$;
- if D'_k up-dominates $C\left(i, D_1, D'_1, D_2, D'_2\right) \bigcap S_{kb}$ in L_{i-1}, let $D'_{ks} = D'_k$; Otherwise select D'_{ks} in $C\left(i, D_1, D'_1, D_2, D'_2\right) \bigcap S_{kb}$ such that it dominates all the other disks in $C\left(i, D_1, D'_1, D_2, D'_2\right) \bigcap S_{kb}$;
- $C_{i-1} = \left(C\left(i, D_1, D'_1, D_2, D'_2\right) \setminus \{D_1, D'_1, D_2, D'_2\}\right) \bigcup \{D_{1s}, D'_{1s}, D_{2s}, D'_{2s}\}$.

By this construction, the generation of C_{i-1} takes two steps. First we find $D_{1s}, D'_{1s}, D_{2s}, D'_{2s}$ and we remove D_k if it is not contained in $\{D_{1s}, D_{2s}\}$. So C_{i-1} doesn't contains D_k if $D_k \notin \{D_{1s}, D_{2s}\}$ and we have the following equation.

$$C\left(i, D_1, D'_1, D_2, D'_2\right) = w(C_{i-1}) + \sum_{k=1}^{2} \left(I\left(D_k \notin \{D_{1s}, D_{2s}\}\right) w(D_k) + I\left(D'_k \notin \{D'_{1s}, D'_{2s}\}\right) w(D'_k) \right).$$

We now prove that C_{i-1} is a feasible solution under the three conditions stated in the definition of $C\left(i-1, D_{1s}, D'_{1s}, D_{2s}, D'_{2s}\right)$.

According to our construction, the second and the third condition hold. For the first condition, similar to the proof as above, we see that if any D_k or D'_k is removed, then any nodes $p_j, j \leq i-1$, covered by D_k can still be covered by a disk which is in C_{i-1} and has center not in the strip where p_j is.

Since C_{i-1} is a feasible solution, we have

$$w(C_{i-1}) \geq C\left(i-1, D_{1s}, D'_{1s}, D_{2s}, D'_{2s}\right),$$

then we have

$$C\left(i, D_1, D'_1, D_2, D'_2\right) \geq RHS. \quad \square$$

The proof to Case 3 is divided into two parts. The first part shows $LHS \leq RHS$ by proving RHS is a feasible solution which is no better than the optimal solution(LHS). In the second part we show $LHS \geq RHS$.

To sum up, we have the following theorem for Algorithm 1.

Theorem 2.1. *Algorithm* 1 *gives an optimal solution for* 2*-stripsMWDS problem, and has a polynomial running time.*

Proof. The table maintained by the algorithm has size $O(n^5)$, and computing each item need $O(n^4)$, so the total running time is still polynomial.

Since each optimal solution *opt* would induce D_1, D'_1, D_2, D'_2 on L_n, so it is one feasible solution in the table maintained by the algorithm at least. By Lemma 2.2, Algorithm 1 will output the optimal solution in the table, and it must be an optimal solution. $\square$

3. 5-approximation algorithm for MWDS

First, we describe the main idea of the algorithm for solving MWDS in the whole plane. We follow the framework in [2].

1. Step 1: **(Double partition)** First, partition the plane into blocks, each with size $K\mu \times K\mu$, where $\mu = \frac{\sqrt{2}}{2}$ and K is a large integral constant. Second, partition each block into K^2 squares, each with size $\mu \times \mu$ (Fig. 4).
2. Step 2: Solve MWDS in each block, using the algorithm for MWDS over two strips.
3. Step 3: Combine solutions for each block and obtain a solution.

3.1. Solving MWDS in block

Since each block B contains K^2 squares, say $S_{ij}, i, j \in [K]$, let V_{ij} denote the set of disks in S_{ij}. We assume that K is an even integer. Each block also contains K strips, $T_1^x, \ldots, T_K^x$, where T_k^x consists of squares $S_{kj}, j \in [K]$. We use T_k^y to denote the strip, which consists of squares $S_{ik}, i \in [K]$.

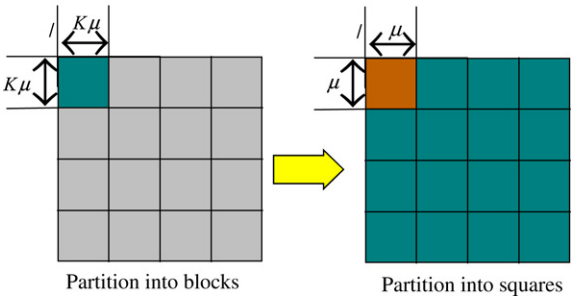

Fig. 4. Double partition.

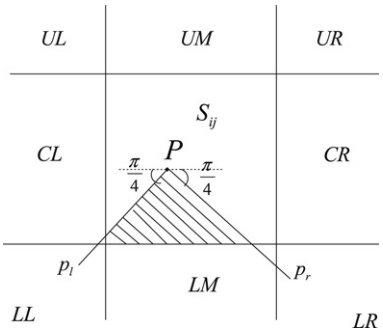

Fig. 5. Neighbors of S_{ij} and region P_{LM}.

We first describe the idea of the algorithm for solving MWDS in block. Notice that the first step in the following is not a real step in our algorithm, since we actually do not know the optimal solution. However, this step comes first in the logic of proving approximation ratio and is very important to help us understand the intuition of our algorithm for MWDS in block.

1. **(Dominating Pattern)** Fix any optimal solution *opt* for the original MWDS in the plane. Consider squares in the block B. For each square S_{ij}, if there is no disk in $opt \bigcap V_{ij}$, assign each unmarked node in V_{ij} to some disk from *opt* that can cover it and mark it; otherwise, select a disk D from $opt \bigcap V_{ij}$, and mark all the nodes covered by D. After we process all the squares in block B, we obtain a dominating pattern for *opt*. In such a pattern, for each square S_{ij}, its corresponding dominating set is either a disk from inside S_{ij}, or a group of disks outside S_{ij}.
2. **(Guessing the pattern)** For each square S_{ij}, if it is covered by a disk inside it, we can guess this disk by enumerating all possible disks; otherwise, we have the lemmas in [2], which says that we can use up to 4 nodes to separate nodes in V_{ij} into two groups, nodes can be covered by disks only from the Up and Down region of the square, and nodes can be covered by disks only from the Left and Right region of the square. Thus, we can still guess the right dominating pattern in *opt* by enumerating.
3. **(Solving MWDS over pairs of strips)** Once we guess a pattern, we decompose the problem into problems in trips. We solve 2-StripMWDS for strips $T^x_{2k-1} \bigcup T^x_{2k}$, $k = 1, \ldots, K/2$. Similarly, we solve 2-StripMWDS for strips $T^y_{2k-1} \bigcup T^y_{2k}$, $k = 1, \ldots, K/2$. We combine all the solutions for these 2-StripMWDS problems, and obtain a dominating set for the block.
4. **(Repairing strips)** Do step 3 again, but this time we solve 2-StripMWDS for strips $T^x_{2k} \bigcup T^x_{2k+1}$, $T^y_{2k} \bigcup T^y_{2k+1}$, $k = 1, \ldots, K/2$. Here T^x_0, $T^y_0 T^x_{2k+1}$, T^y_{2k+1} can be viewed as strips with no nodes to be covered. We take the minimum of the two solutions for the block under each pattern. We then output the minimum solution over all possible enumerating patterns.

We first state the lemmas in [2], which can separate nodes in V_{ij} into two groups, hence help us fix a dominating pattern by up to 4 nodes. First we introduce some notations used in [2]. We divide the neighbor parts of S_{ij} into eight regions UL, UM, UR, CL, CR, LL, LM, LR as shown in Fig. 3. Assume the four lines forming S_{ij} are $x = x_1, x = x_2, y = y_1, y = y_2$. We also use $Left = UL \bigcup CL \bigcup LL$, $Right = UR \bigcup CR \bigcup LR$, $Up = UL \bigcup UM \bigcup UR$, $Down = LL \bigcup LM \bigcup LR$ (Figs. 5 and 6).

Lemma 3.1 ([2]). *Suppose $p \in V_{ij}$ is a disk in S_{ij}, which can be dominated by a disk $D \in LM$. We draw two lines p_l and p_r, which has slope $1, -1$ respectively. Then the shadow P_{LM} can also be dominated by D. Similar results can be hold for shadow P_{UM}, P_{CL} and P_{CR}, which can be defined with a rotation.*

We give an alternative proof here, which is simpler and shows more intuition of P_{LM}'s construction.

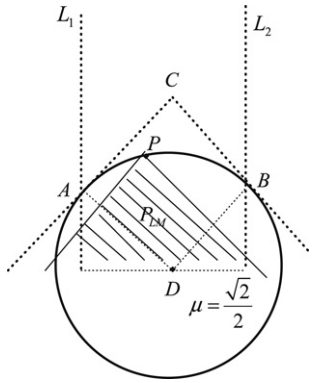

Fig. 6. Proof of dominating nodes in P_{LM}.

Proof. We draw two vertical lines L_1, L_2, which has a distance of μ to the center of disk D. Since $D \in LM$, we know that the square S_{ij} must be between them, hence p must be on the arc $\widehat{AB}$ or in the region below it. Since CA, CB are both tangent with disk D, and having slope 1, -1 respectively, we can easily see that wherever p is, the region P_{LM} must be all covered in the disk D. □

Then, we give the definition of sandglass and another lemma which can use up to 4 nodes to separate the nodes in V_{ij} into two groups covered only by disks from $Up \bigcup Down$, $Left \bigcup Right$ respectively.

Definition 3.1 (*[2] Sandglass*). If C is a dominating set for square S_{ij}, and $C \bigcap V_{ij} = \emptyset$, then there must exist a subset $V_M \subset V_{ij}$, which can only be covered by disks from UM and LM(we can set $V_M = \emptyset$ if no such disk exists). Choose $V_{LM} \subset V_M$ the disks that can be covered by disks from LM, draw p_l and p_r for each $p \in V_{LM}$. Choose the leftmost p_l and the rightmost p_r and form a shadow. Symmetrically, choose V_{UM} and form a shadow with leftmost and rightmost lines. The union of the two shadows form a "sandglass" region $Sand_{ij}$ of S_{ij}.

Lemma 3.2 (*[2]*). *Suppose C is a dominating set for S_{ij}, and $Sand_{ij}$ is chosen in the above way. Then any disks located in $Sand_{ij}$ can be dominated by disks only from a neighboring region $Up \bigcup Down$, and disks located in $S_{ij} \setminus Sand_{ij}$ can be dominated by disks only from a neighboring region $Left \bigcup Right$.*

So formally, we have the following algorithm for solving MWDS in a block. The algorithm is based on Algorithm 1 from [2], and our new ideas fall into two parts. First, instead of calculating optimal dominating set for each strip, we deal with a pair of two strips one time using Algorithm 1. Second, we combine strips into pairs in two ways, solve the problem twice and select the better solution.

Algorithm 2**:** solve MWDS in a block.
Step 1: For each S_{ij}, select its sandglass or select a disk $d \in V_{ij}$.
Step 2: If $d \in V_{ij}$ is selected, then remove d and all disks dominated by d.
Step 3: For each pair of strips T^x_{2k-1} and T^x_{2k}, $k = 1, \ldots, K/2$, calculate its optimal dominating set for the union of disks in the sandglasses of squares in this pair of strips.
Step 4: For each pair of strips T^y_{2k-1} and T^y_{2k}, $k = 1, \ldots, K/2$, calculate its optimal dominating set for the remaining disks not covered by Step 3.
Step 5: Repeat Step 3, 4 again, but this time we combine strips T^x_{2k} and T^x_{2k+1} into a pair(T^y_{2k} and T^y_{2k+1} respectively), $k = 1, \ldots, K/2$.

Fix any optimal solution opt for the original MWDS problem. Assume that there are m blocks, say $B_1, \ldots, B_m$. Recall that, for the solution opt, we can fix a dominating pattern ϕ. For a block B_i, denote the disks that are used in pattern ϕ to dominate disks in B_i as opt_i^{ϕ}. Then we have the following lemma used in proving the approximation ratio.

Lemma 3.3. *For any block B_i, Algorithm 2 can find a dominating set for B_i with weight no more than $5w(opt_i^{\phi})$.*

Proof. According to the definition of opt_i^{ϕ}, for any square in block B_i, it is either dominated by a disk in this square, or dominated by some disks not in it. So during the enumeration process in Algorithm 2, once the dominating pattern ϕ is guessed correctly by the algorithm, for any disk that is used in pattern ϕ to dominate the square containing this disk, it is selected and then deleted by the algorithm in step 1, hence is only used once. The optimal solution for each pair of strips found by our algorithm is bounded by the feasible solution given by opt_i^{ϕ}. So, if we replace all the optimal solutions for each 2-StripMWDS by the feasible solutions given by opt_i^{ϕ} in pattern ϕ, we obtain an upper bound for the weight of the solution found by Algorithm 2.

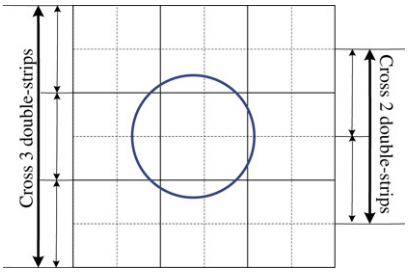

Fig. 7. A disk cannot be counted 3 times in both ways of pairing strips.

Now, we consider how many times each disk in opt_i^{ϕ} can be counted. Consider the case when we calculating MWDS for pairs of horizontal strips first. As showed in Fig. 7, if a disk is used 3 times in Step 3, it is used at most twice in step 5; otherwise(it is used twice in Step 2), it is used at most 3 times in Step 5. So totally, it is used 5 times in the horizontal strips. For the horizontal strips the analysis is the same. By adding the horizontal and vertical strips up, for any disk, it could be counted at most 10 times totally. The two solutions for block B_i together have weight no more than $10w(opt_i^{\phi})$, so our algorithm gives a solution with weight no more than $5w(opt_i^{\phi})$ when it guess the pattern ϕ correctly. Since the algorithm enumerates all possible dominating patterns, and takes the minimum solution, the lemma holds. □

3.2. Solving MWDS for the whole plane

We use the framework of [2] to construct our algorithm for MWDS in UDG. To summarize, we state the algorithm as following,

Algorithm 3**:** Solve MWDS for the whole plane.
Step 1: Partition the whole plane into blocks of size $K\mu \times K\mu$, then partition each block into squares with size $\mu \times \mu$, where $\mu = \frac{\sqrt{2}}{2}$.
Step 2: Calculate MWDS for each block that contains disks and merge the solutions together to form a solution for the whole plane.
Step 3: Move each block two squares to the right, and two squares to the top of the original block.
Step 4: Repeat Step 2 for this new partition to update the solution if any better solution is found.
Step 5: Repeat Step 3 for $K/2$ times, and output the final solution.

Theorem 3.1. *For any constant ϵ, by setting $K = O(\frac{1}{\epsilon})$, Algorithm* 3 *always outputs a dominating set with weight bounded by $(5+\epsilon)OPT$, where OPT is the optimum.*

Proof. The proof of Theorem 3.1 follows the proof given in [2]. The main idea is that each disk in the optimal solution *opt* can only be counted 5 times in most region of a block, however may be counted more in the boundary region of a block. So when we shift the whole block many times, for any disk in *opt*, it would be counted at most 5 times in most positions. □

Similarly, combining the technique used in [2], we can turn the dominating set found by Algorithm 3 into a connected dominating set by adding some disks, and the approximation ratio adds 4.

Theorem 3.2. *Our algorithm* 3*, together with the algorithm used in* [2] *to connect a dominating set, gives a polynomial algorithm with approximation ratio* $9+\epsilon$.

4. Conclusion and open problems

In this paper, we present a new technique of dynamic programming, which can give the optimal solution for the MWDC problem over two strips(compare to only for one strip in [2]). One possible direction for the future research is whether this technique can be further extended to solve more strips, hence reduce the approximation ratio. Another interesting open problem would be the weighted dominating set problem in the case when disks have a different radius.

Acknowledgements

We thank Professor Dingzhu Du for introducing us the problem and some useful discussions. The authors were supported by the National Natural Science Foundation of China Grant 60553001 and the National Basic Research Program of China Grant 2007CB807900, 2007CB807901.

References

[1] Christoph Ambühl, Thomas Erlebach, Matús Mihalák, Marc Nunkesser, Constant-factor approximation for minimum-weight (connected) dominating sets in unit disk graphs, in: Proc. APPROX-RANDOM, 2006, pp. 3–14.
[2] Yaochun Huang, Xiaofeng Gao, Zhao Zhang, Weili Wu, A better constant-factor approximation for weighted dominating set in unit disk graph, J. Comb. Optim. (ISSN: 1382-6905) (2008) 1573–2886. (Print) (Online).
[3] David Lichtenstein, Planar formulae and their uses, SIAM J. Comput. 11 (2) (1982) 329–343.
[4] M.V. Marathe, H. Breu, H.B. Hunt III, S.S. Ravi, D.J. Rosenkrantz, Simple heuristics for unit disk graphs, Networks, 25, pp. 59–86.
[5] H.B. Hunt III, M.V. Marathe, V. Radhakrishman, S.S. Ravi, D.J. Rosenkrantz, R.E. Stearns, NC-approximation schemes for NP- and PSPACE-hard problems for geometric graphs, J. Algorithms 26 (2) (1998) 238–274.
[6] B.N. Clark, C.J. Colbourn, D.S. Johnson, Unit disk graphs, Discrete Math. 86 (1–3) (1990) 165–177.
[7] M.R. Garey, David S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, W. H. Freeman, ISBN: 0-7167-1044-7.
[8] R. Bar-Yehuda, S. Moran, On approximation problems related to the independent set and vertex cover problem, Discrete Appl. Math. 9 (1984) 1–10.
[9] V.V. Vazirani, Approximation Algorithms, Springer, 2001.
[10] Uriel Feige, A Threshold of ln n for Approximating Set Cover, in: Proc. 28th ACM Symposium on Theory of Computing, 1996, pp. 314–318.
[11] S. Guha, S. Khuller, Improved methods for approximation node weighted Steiner trees and connected dominating sets, Inform. Comput. 150 (1) (1999) 57–74.
[12] Khaled M. Alzoubi, Pengjun Wan, Ophir Frieder, Message-optimal connected dominating sets in mobile ad hoc networks, in: Proceedings of the 3rd ACM Interational Symposium on Mobile Ad Hoc Networking and Computing, June 9–11, 2002, pp. 157–164.
[13] Jie Wu, Hailan Li, On calculating connected dominating set for efficient routing in ad hoc wireless networks, in: Proceedings of the 3rd International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications, Seattle, Washington, USA, August 20, 1999, pp. 7–14.