Selecting Forwarding Neighbors in Wireless Ad Hoc Networks

Gruia Călinescu
CS Department
Illinois Institute of Technology
Chicago, IL 60616
calinesc@cs.iit.edu

Ion Măndoiu
CS Department
University of California at Los Angeles
Los Angeles, CA 90095
mandoiu@cc.gatech.edu

Peng-Jun Wan
CS Department
Illinois Institute of Technology
Chicago, IL 60616
wan@cs.iit.edu

Alexander Zelikovsky*
CS Department
Georgia State University
Atlanta, GA 30303
alexz@cs.gsu.edu

ABSTRACT

Broadcasting is a fundamental operation which is frequent in wireless ad hoc networks. A simple broadcasting mechanism, known as *flooding*, is to let every node retransmit the message to all its 1-hop neighbors when receiving the first copy of the message. Despite its simplicity, flooding is very inefficient and can result in high redundancy, contention, and collision. One approach to reducing the redundancy is to let each node forward the message only to a small subset of 1-hop neighbors that cover all of the node's 2-hop neighbors. In this paper, we propose two practical heuristics for selecting the minimum number of forwarding neighbors: an $O(n \log n)$ time algorithm that selects at most 6 times more forwarding neighbors than the optimum, and an $O(n^2)$ time algorithm with an improved approximation ratio of 3, where n is the number of 1- and 2-hop neighbors. The best previously known algorithm, due to Bronnimann and Goodrich [2], guarantees $O(1)$ approximation in $O(n^3 \log n)$ time.

1. INTRODUCTION

Wireless ad hoc networks can be flexibly and quickly deployed for many applications such as automated battlefield, search and rescue, and disaster relief. Unlike wired networks or cellular networks, no wired backbone infrastructure is installed in wireless ad hoc networks. A communication session is achieved either through a single-hop radio transmission if the communication parties are close enough, or through relaying by intermediate nodes otherwise. In this paper, we assume that all nodes in a wireless ad hoc network are distributed in a two-dimensional plane and have an equal maximum transmission range of one unit.

Broadcasting is a fundamental networking operation in wireless ad hoc networks. It is widely and frequently performed in many networking tasks such as paging a particular host, sending an alarm signal, and finding a route to a particular host [1][6][13]. A simple broadcasting mechanism, known as *flooding*, is to let every node retransmit the message to all its 1-hop neighbors when receiving the first copy of the message. Despite its simplicity, flooding has a serious drawback, known as the *broadcast storm* [12]. First, because the radio propagation is omnidirectional and a physical location may be covered by the transmission ranges of several nodes, many retransmissions are redundant. Second, heavy contention could exist because retransmitting nodes are probably close to each other. Third, collisions are more likely to occur because the RTS/CTS dialogue is inapplicable and the timing of retransmissions is highly correlated.

The following simple technique was recently exploited [9][14] to reduce redundant retransmissions: By virtue of beaconing, each node maintains a local topology of its 2-hop neighborhood, and relays the message only to a small subset of 1-hop neighbors which cover (in terms of radio range) all nodes that are two hops away. The subset of 1-hop neighbors selected by each node is referred to as *forwarding set* [14] or *multipoint relaying set* [9]. In this paper we consider the problem of finding a forwarding set of minimum size.

Minimum Forwarding Set Problem: Given a source A, let $\mathcal{D}$ and $\mathcal{P}$ be the sets of 1- and 2-hop neighbors of A. Find a minimum-size subset $\mathcal{F}$ of $\mathcal{D}$ such that every node in $\mathcal{P}$ is within the coverage area of at least one node from $\mathcal{F}$.

1.1 Previous work

Jacquet et al. [9] and Sinha et al. [14] considered the Minimum Forwarding Set problem assuming no knowledge of the geographic location of the nodes. In this case, the Minimum Forwarding Set problem is essentially the well-studied Set Cover problem. Not surprisingly, the heuristic proposed in [9] is a translation of Chvátal's greedy algorithm [3] for Set Cover, and thus guarantees an approximation factor of $O(\log m)$, where m is the maximum neighborhood size. The greedy algorithm iteratively selects a 1-hop neighbor covering the maximum number of 2-hop neighbors not yet covered, and terminates when all 2-hop neighbors have been covered. The greedy algorithm does not take into account the geometric properties of the Minimum Forwarding Set problem, and in fact Figure 1 shows a family of instances for which the size of the solution found by the greedy algo-

*Partially supported by NSF Grant CCR-9988331 and a CRDF Grant.

DIAL M 2001 7/01 Rome, Italy

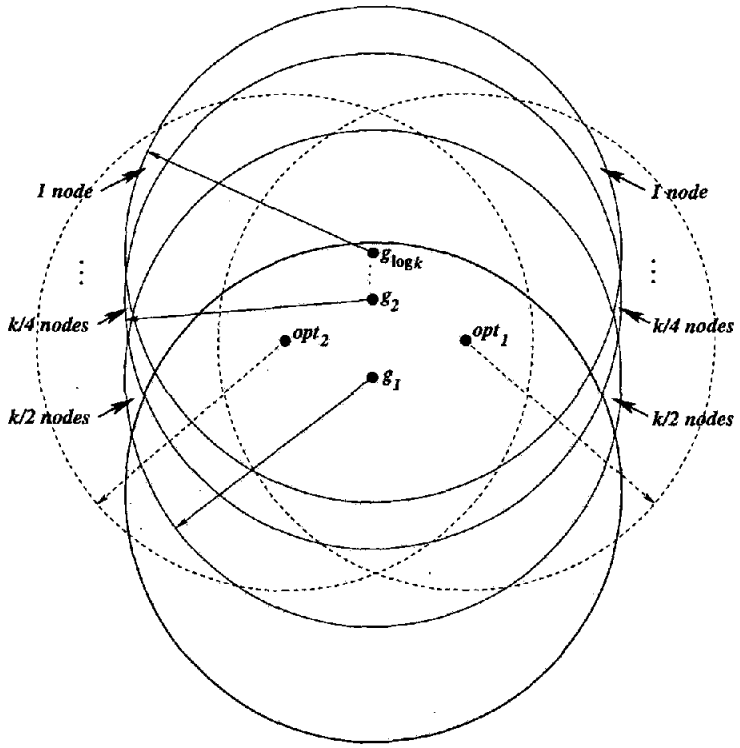

Figure 1: Instance for which the size of the solution computed by the greedy algorithm, $\{g_1, \ldots, g_{\log k}\}$, is larger than the optimum solution, $\{opt_1, opt_2\}$, by a logarithmic factor.

rithm is larger than the optimum by a logarithmic factor.

Under the assumption that the nodes in the wireless network are distributed in a two-dimensional plane and each node has unit transmission range, the topology of the network is modeled as a *unit-disk graph* [4]. In this graph, there is an edge between two nodes if and only if their distance is at most one. The Minimum Forwarding Set problem for a given source node s asks for a minimum size set of 1-hop neighbors of s *dominating* 2-hop neighbors of s in the unit-disk graph. The related Dominating Set problem in unit-disk graphs [4] asks for a subset of nodes dominating (i.e., adjacent to) all the other nodes. The Dominating Set problem in unit-disk graphs is NP-hard [4] but admits a PTAS [8]. The Minimum Forwarding Set problem does not reduce to the Dominating Set problem in unit-disk graphs since dominators are restricted to the set of 1-hop neighbors.

The Minimum Forwarding Set problem is also related to the Unit-Disk Cover problem [7], which asks for the minimum number of unit disks covering a given set of points in the plane. The Unit-Disk Cover problem is also NP-hard [4] and admits a PTAS [7]. Since in the Unit-Disk Cover problem disk centers can be chosen arbitrarily in the plane, the algorithms for this problem do not apply to the Minimum Forwarding Set problem where disks must be centered at 1-hop neighbors only.

The Minimum Forwarding Set problem is a special case of the NP-Hard Disk Cover problem [2], which asks for a minimum size subset of a given set of disks covering a given set of points. The complexity of Minimum Forwarding Set problems is not known. A constant-ratio approximation algorithm for Disk Cover, and therefore also for Minimum Forwarding Set, was given by Bronnimann and Goodrich [2] However, their algorithm – which is a special case of a sophisticated algorithm for spaces with bounded VC-dimension – has impractical running-time and its proven approximation ratio is a very large constant.

1.2 Our contributions

- A 6-approximation algorithm for the Minimum Forwarding Set problem running in $O(n \log n)$ time, where n is the total number of 1- and 2-hop neighbors.
- A 3-approximation algorithm for the Minimum Forwarding Set problem running in $O(n^2)$ time.
- An exact $O(n^2)$ time, and a 2-approximation $O(n \log n)$ time algorithm for the special case of the Minimum Forwarding Set problem when all 2-hop neighbors are in the same quadrant with respect to the source node.
- A constant-factor approximation for the Minimum Disk Cover problem with disks of the same radius, based on rounding the optimal solution of a linear programming relaxation.
- An experimental study of the proposed algorithms for the Minimum Forwarding Set problem.

The paper is organized as follows. In next section we reformulate the Minimum Forwarding Set problem in geometric

Algorithm 1: 1-Hop Disk Cover

Input: Unit-disk A, set of unit disks $\mathcal{D}$ centered inside A, set of points $\mathcal{P}$ outside A such that $\mathcal{P} \subseteq \cup\{D \in \mathcal{D}\}$
Output: Subset $\mathcal{F} \subseteq \mathcal{D}$ such that $\mathcal{P} \subseteq \cup\{D \in \mathcal{F}\}$

1. Partition the exterior of A into four quadrants Q_1–Q_4 by two orthogonal lines, not containing points in $\mathcal{P}$, through the center of A (see Figure 2).
2. For $q = 1, \ldots, 4$, compute a disk cover, $\mathcal{F}_q$, for the points in $\mathcal{P} \cap Q_q$.
3. Output $\mathcal{F} = \mathcal{F}_1 \cup \mathcal{F}_2 \cup \mathcal{F}_3 \cup \mathcal{F}_4$.

terms, give a high-level algorithm based on decomposition into quadrants, and establish basic geometric properties of the partitioned sets of 1- and 2-hop neighbors. In Section 3 we describe a 2-approximation $O(n \log n)$ time algorithm for covering 2-hop neighbors in a quadrant. An exact $O(n^2)$ time algorithm for the same problem is described in Section 4. In Section 5 we give an extension of our techniques to the Disk Cover problem of [2]. We present preliminary experimental results comparing the proposed algorithms for the Minimum Forwarding Set problem in Section 6 and conclude in Section 7.

2. PARTITION BASED ALGORITHM

Throughout this paper a *unit disk*, or just *disk* for short, refers to a closed disk of radius 1. The boundary of a region R of the Euclidean plane is denoted by ∂R, e.g., the boundary circle of a disk D is denoted by ∂D. Under the assumption that each network node has unit transmission range, we reformulate the Minimum Forwarding Set problem as follows.

1-Hop Disk Cover Problem: Given a unit-disk A, a set $\mathcal{D}$ of unit disks centered inside A, and a set of points $\mathcal{P}$ outside A such that $\mathcal{P} \subseteq \cup\{D \in \mathcal{D}\}$, find a minimum-size subset $\mathcal{F}$ of $\mathcal{D}$ such that $\mathcal{P} \subseteq \cup\{D \in \mathcal{F}\}$.

Our high-level algorithm (Algorithm 1) partitions the points of $\mathcal{P}$ according to the four quadrants defined by two orthogonal lines through the center of A, and then independently solves the 1-Hop Disk Cover problem for each quadrant. The union of these four disk covers is then a disk cover for all the points in $\mathcal{P}$. As usual, the approximation ratio of an algorithm $\mathcal{A}$ for a minimization problem Π is the supremum, over all instances of Π, of the ratio between the output value of $\mathcal{A}$ and the optimal value. The following theorem relates the approximation ratio of Algorithm 1 to the approximation ratio that can be guaranteed for the 1-Hop Disk Cover restricted to points in a single quadrant.

THEOREM 1. *If disk covers $\mathcal{F}_q$ computed in Step 2 are within a factor of α of optimum, then Algorithm 1 has an approximation ratio of at most 3α for the 1-Hop Disk Cover problem.*

Proof. Let OPT be the optimal set of disks, and denote by OPT_q, $q = 1, 2, 3, 4$, the subset of disks in OPT having centers in the q^{th} sector of disk A. The key observation is that points in the quadrant Q_q cannot be covered by disks in $OPT_{q+2 (\text{mod } 4)}$. Therefore, points in $\mathcal{P} \cap Q_1$ must be covered by disks in $OPT_4 \cup OPT_1 \cup OPT_2$, and thus, by the assumption that $\mathcal{F}_q$'s are within a factor of α of the respective optimum solutions,

$$|\mathcal{F}_1| \leq \alpha(|OPT_4| + |OPT_1| + |OPT_2|).$$

Similarly,

$$\begin{aligned} |\mathcal{F}_2| &\leq \alpha(|OPT_1| + |OPT_2| + |OPT_3|), \\ |\mathcal{F}_3| &\leq \alpha(|OPT_2| + |OPT_3| + |OPT_4|), \\ |\mathcal{F}_4| &\leq \alpha(|OPT_3| + |OPT_4| + |OPT_1|). \end{aligned}$$

Thus, the output of the algorithm has size

$$\begin{aligned} & |\mathcal{F}_1| + |\mathcal{F}_2| + |\mathcal{F}_3| + |\mathcal{F}_4| \\ \leq\ & 3\alpha(|OPT_1| + |OPT_2| + |OPT_3| + |OPT_4|) \\ =\ & 3\alpha|OPT|. \end{aligned}$$

■

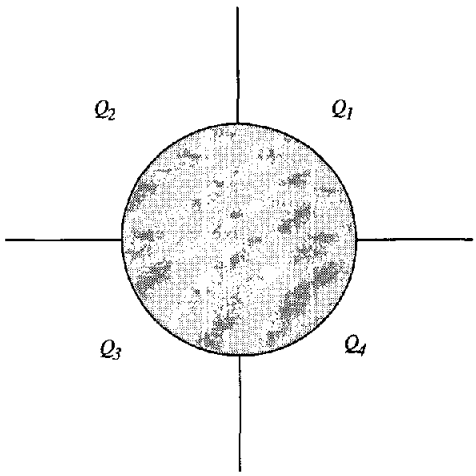

Figure 2: The four quadrants in Algorithm 1.

We will show that $\alpha = 2$ can be achieved in $O(n \log n)$ time (see Section 3), and $\alpha = 1$ can be achieved in $O(n^2)$ time (see Section 4). Hence, Algorithm 1 achieves an approximation factor of 6, respectively 3, within the same time bounds. It is natural to ask if these approximation ratios can be improved by partitioning the set of points according to $k < 4$ equal sectors defined by half-lines starting at the center of A. The proof of Theorem 1 can be generalized to show that partitioning into k sectors gives an approximation ratio of $(\lceil k/2 \rceil + 1)\alpha$ for the 1-Hop Disk Cover problem if the disk cover for each sector is approximated within a factor of α. Thus, using decomposition into 3 equal sectors does not lead to an approximation ratio better than that obtained by decomposition into quadrants. Improvements using decomposition into 2 equal sectors are possible provided that we can find an algorithm for covering the points in a 180° sector with an approximation ratio of less than 3/2. The ideas used in Section 4 to solve exactly the problem for a quadrant do not extend to 180° sectors, since these lack the

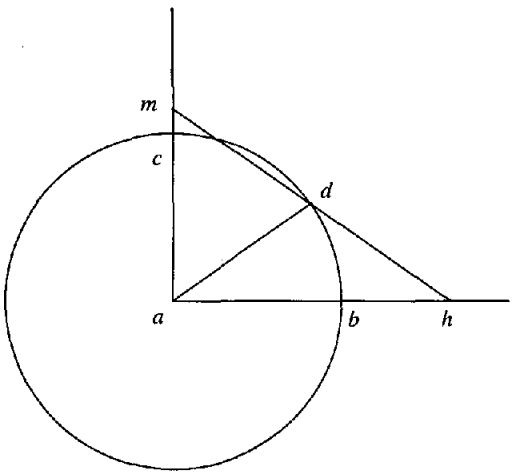

Figure 3: The extreme configuration in the proof of Lemma 2(b).

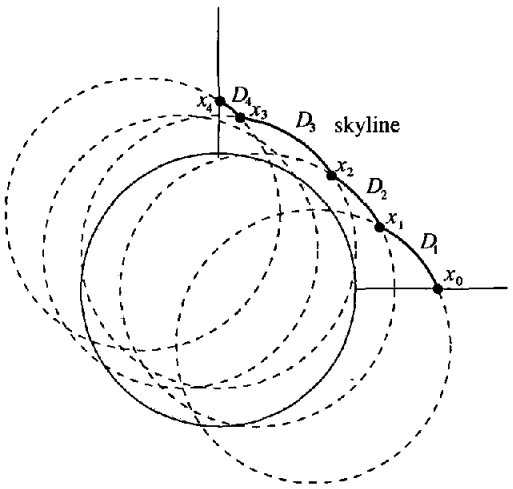

Figure 4: The skyline of a set of disks in a quadrant.

second of the essential topological properties established for the quadrants in the following lemma.

LEMMA 2. *Let Q be an exterior quadrant of A, $J = \partial D$ be its border, and $\mathcal{D}$ be a set of disks intersecting the interior of Q. Then:*
(a) For any disk $D \in \mathcal{D}$, $|\partial D \cap J| = 2$.
(b) For any two disks $D, D' \in \mathcal{D}$, $|\partial D \cap \partial D' \cap Q| \leq 1$.
(c) No two disks in $\mathcal{D}$ are tangent in Q.

Proof. Without loss of generality, we may assume that the unit-disk A is centered at the origin and that Q is defined by the positive x- and y-axes. Then, the boundary of the quadrant Q, J, consists of the two half-lines from $(0,1)$ to $(0,\infty)$, and from $(1,0)$ to $(1,\infty)$, together with a quarter-circle of ∂A. Let a, b, c be the points with coordinates $(0,0)$, $(1,0)$, and $(0,1)$, respectively. We will use $\widehat{bc}$ to denote the quarter-circle of A enclosed in J.

(a) Since every $D \in \mathcal{D}$ has non-empty intersection with the interior of Q, every circle ∂D has at least two intersection points with J. The closed simple Jordan curve ∂D and the infinite simple Jordan curve J must intersect an even number of times (unless they are tangent, but this cannot happen), and thus cannot intersect three times. Thus, to complete the proof of part **(a)** we need to show that ∂D does not intersect J four or more times.

Let d denote the center of disk D. Then $0 < |da| \leq 1$, since d is inside A. Note that ∂D can intersect the x-axis in at most two points, of which only one can have x-coordinate bigger than 1. Similarly, D can intersect the y-axis in at most two points, of which only one can have y-coordinate bigger than 1. Furthermore, D intersects $\widehat{bc}$ at most once. Indeed, when two unit-circles with centers within distance of at most 1 intersect, the two intersection points are at least $2\pi/3$ apart on each of the circles, and hence a quarter-circle may contain only one of them.

(b) Assume, for a contradiction, that D and D' are two distinct disks in $\mathcal{D}$ that intersect at points h and l, with both h and l in $Q \cup J$. Let d and d' be the centers of D and D', respectively. We will change the configuration a bit, to obtain a more extreme case. First, translate d, d', h, and l to the right until d or d' hits $\widehat{bc}$, and assume, by symmetry, that d is on $\widehat{bc}$. We still have $h, l \in R \cup J$. Assume also that h is to the right of the point l. Now start rotating the rhombus $hd'ld$ clockwise around d until h hits either the x-axis or $\widehat{bc}$, whichever happens first (see Figure 3). This procedure also keeps d' inside the unit-disk A and l in Q. Let m be the point where the line hd intersects the y-axis. As $|d'h| = |dh| = 1$, d' must lie in the same side of the line hm as a. As the angle $\widehat{ham} \leq \frac{\pi}{2}$, m must be within the diameter of the unit-disk centered at d that contains h. Therefore $|dm| \leq 1$. Thus l, which is in Q, must be outside the triangle ahm, and consequently d' must be on the other side of the line hm than a, which is a contradiction.

(c) Let D and D' be two disks from $\mathcal{D}$. Then ∂D and $\partial D'$ cannot be tangent from the interior since they have the same radius. If ∂D and ∂D are tangent from the exterior, then the distance between their centers is 2, and the common point can only be the origin a, which is not in Q. ■

3. FAST GEOMETRIC DISK COVERING IN A QUADRANT

In this section we give a fast 2-approximation algorithm for the 1-Hop Disk Cover problem with all points of $\mathcal{P}$ coming from an exterior quadrant Q of unit disk A.

The *skyline* $S = (x_0, x_1, \ldots, x_k)$ of $\mathcal{D}$ is the upper envelope of $Q \cap (\cup\{D \in \mathcal{D}\} \cup A)$ (see Figure 4). The skyline consists of arcs $\widehat{x_{i-1}x_i}$ on the border of disks $D_i \in \mathcal{D} \cup \{A\}$, $i = 1, \ldots, k$, such that $x_0 \in \partial Q \cap \partial D_1$, $x_i \in \partial D_{i-1} \cap \partial D_i$ $(i = 1, \ldots, k-1)$, and $x_k \in \partial D_k \cap \partial Q$. The algorithm (Algorithm 2) starts by computing the skyline S with x_i's numbered in counter-clockwise order, i.e., with polar coordinates (ρ_i, r_i) of points x_i satisfying $\rho_0 \leq \rho_1 \leq \rho_2 \leq \cdots \leq \rho_k$. As established in Lemma 8 below, the skyline disks D_i covering a point $p \in \mathcal{P}$

Algorithm 2: Geometric 1-Hop Disk Covering in a quadrant

Input: Unit-disk A, set of unit disks $\mathcal{D}$ centered inside A, set of points $\mathcal{P}$ in the exterior quadrant Q of A such that $\mathcal{P} \subseteq \cup\{D \in \mathcal{D}\}$
Output: Subset $\mathcal{F} \subseteq \mathcal{D}$ such that $\mathcal{P} \subseteq \cup\{D \in \mathcal{F}\}$

1. Find the skyline $S = (x_0, x_1, x_2, \ldots, x_k)$ of $\mathcal{D}$, where the polar coordinates of x_i are (ρ_i, r_i) and $\rho_0 \leq \rho_1 \leq \rho_2 \leq \cdots \leq \rho_k$. Let D_i be the disk containing arc $\overset{\frown}{x_{i-1}x_i}$.
2. For each $p \in \mathcal{P}$ with polar coordinates (ρ, r), find the interval $[D_{first(p)}, D_{last(p)}]$ of skyline disks D_i that cover p, via three binary searches:
   (a) find $i \in \{1, \ldots, k\}$, such that $\rho \in [\rho_{i-1}, \rho_i]$
   (b) $first(p) \leftarrow \min\{j : 1 \leq j \leq i, p \in D_j\}$
   (c) $last(p) \leftarrow \max\{j : i \leq j \leq k, p \in D_j\}$
3. Using the greedy algorithm, find the minimum set $\mathcal{F}$ of disks D_i hitting each interval $[D_{first(p)}, D_{last(p)}]$, $p \in \mathcal{P}$.
4. Output $\mathcal{F}$

form an interval in the sequence $D_1, \ldots, D_k$. The algorithm computes these intervals for each point of $\mathcal{P}$, them outputs a minimum size set $\mathcal{F}$ of skyline disks D_i hitting all intervals. Clearly, the hitting set $\mathcal{F}$ computed by Algorithm 2 is a disk cover for the points in $\mathcal{P}$. Furthermore, we have:

THEOREM 3. *Algorithm 2 runs in $O(n \log n)$ time, and has an approximation ratio of 2 for the 1-Hop Disk Cover problem in a quadrant.*

Theorems 1 and 3 immediately give:

COROLLARY 4. *Combined with Algorithm 2, Algorithm 1 runs in $O(n \log n)$ time and has an approximation ratio of 6 for the Minimum Forwarding Set problem.*

The rest of the section is devoted to the proof of Theorem 3.

LEMMA 5. *A point $q \in Q$ belongs to a disk $D \in \mathcal{D}$ if and only if the half-line L from the center a of A through a point q intersects $\partial D \cap Q$ at a point q' such that q belongs to the segment $[a, q']$.*

Proof. Every disk $D \in \mathcal{D}$ contains a. Thus, the segment $[a, q']$ is fully contained in D, and every point of L outside of this segment is in the exterior of D. ■

LEMMA 6. *If point $p \in \mathcal{P}$ has polar coordinates (ρ, r) such that $\rho \in [\rho_{i-1}, \rho_i]$, then $p \in D_i$.*

Proof. Follows immediately from Lemma 5. ■

LEMMA 7. *Let D_1, D_2, D_3 be three disks of $\mathcal{D}$ appearing in this order in the skyline of $\{D_1, D_2, D_3\}$. Then $D_1 \cap D_3 \cap Q \subseteq D_2 \cap Q$.*

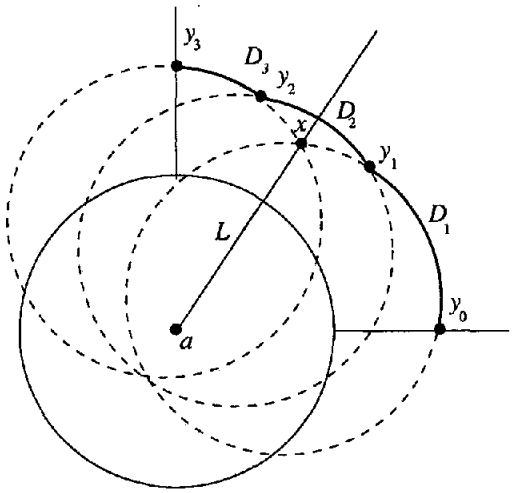

Figure 5: The skyline of $\{D_1, D_2, D_3\}$ in Lemma 7.

Proof. Assume that $D_1 \cap D_3 \cap Q \neq \emptyset$, and let $S' = (y_0, y_1, y_2, y_3)$ be the skyline of $\{D_1, D_2, D_3\}$ (see Figure 5). Since $y_1 = \partial D_1 \cap \partial D_2 \cap Q$, $y_2 = \partial D_2 \cap \partial D_3 \cap Q$, and $y_1, y_2 \notin D_1 \cap D_3$, Lemma 2(b) implies that $\partial D_2 \cap \partial(D_1 \cap D_3 \cap Q) = \emptyset$.

To complete the proof, it suffices to show that D_2 contains some point of $D_1 \cap D_2 \cap Q$. Let $x = \partial D_1 \cap \partial D_2 \cap Q$, and let L be the half-line from a through x. Since $a \in D_1$, L intersects ∂D_1 exactly once, at x. Thus, L does not intersect the arc $\overset{\frown}{y_0y_1}$ of the skyline. Similarly, L does not intersect $\overset{\frown}{y_2y_3}$. It follows that L intersects $\overset{\frown}{y_1y_2}$, and, by Lemma 5, $x \in D_2$. ■

The following is a straightforward corollary of Lemma 7:

LEMMA 8. *For every $p \in \mathcal{P}$, the skyline disks D_i covering p form an interval $[D_{first(p)}, D_{last(p)}]$ in the sequence $D_1, \ldots, D_k$.*

LEMMA 9. *The optimum cover of $\mathcal{P}$ with disks from the set $\{D_1, \ldots, D_k\}$ of skyline disks contains at most 2 times more disks than the optimum cover of $\mathcal{P}$ with disks from $\mathcal{D}$.*

Proof. It suffices to prove that, for every $D \in \mathcal{D}$, $D \cap Q$ is covered by at most two skyline disks. Furthermore, since Lemma 5 implies that any set of disks covering $\partial D \cap Q$ fully covers $D \cap Q$, we only need to show that $\partial D \cap Q$ is covered by at most two skyline disks.

Let d_1 and d_2 be the two points of intersection of ∂D with the boundary of the central disk A. By Lemma 2(b), any skyline disk D_i intersecting $D \cap Q$ contains at least one of the points d_1 and d_2. The key observation is that, for any two skyline disks D_i and D_j both containing d_1 (or both containing d_2), the arc $\partial D \cap \partial D_i \cap Q$ is contained in the arc $\partial D \cap \partial D_j \cap Q$ or vice versa. Therefore the minimal set of skyline disks covering $\partial D \cap Q$ has at most two disks. ■

Proof of Theorem 3. The approximation ratio of Algorithm 2 follows from Lemma 9. Step 1 of the algorithm can

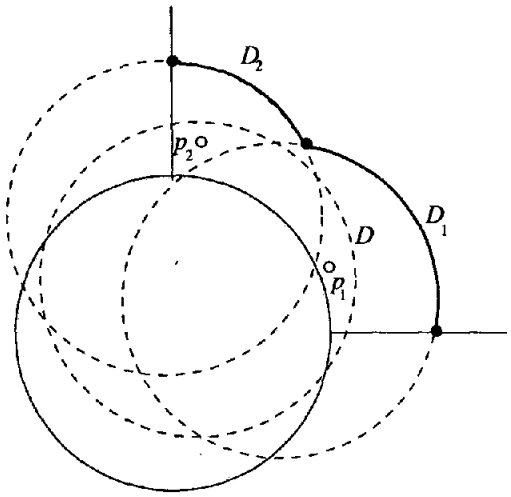

Figure 6: Tight example for the approximation ratio of Algorithm 2.

be implemented in $O(n \log n)$ time using, e.g., an adaptation of the divide-and-conquer algorithm in [11] for computing the Manhattan skyline. The binary searches in Step 2 also take $O(n \log n)$ time. Finally, the minimum set of points hitting a set of intervals can be computed in linear in $O(n)$ time by the following simple greedy algorithm: sort (using counting sort) the intervals according to the right endpoint, and then repeatedly pick the rightmost point of the first (in the sorted order described above) interval not yet hit. The work is constant per interval: to check if an interval is hit, we only have to compare the leftmost endpoint of the interval with the rightmost selected element. ■

Remark. The approximation ratio of 2 in Theorem 3 is tight: Figure 6 gives an instance when the optimum disk cover consisting of skyline disks has size 2, while there is a single disk covering the two points of $\mathcal{P}$.

4. EXACT COMBINATORIAL DISK COVERING IN A QUADRANT

In this section we give an $O(n^2)$ exact algorithm (Algorithm 3) for the 1-Hop Disk Cover problem with all points of $\mathcal{P}$ coming from an exterior quadrant Q of unit disk A. The algorithm is based on careful removal of disks combinatorially covered by their neighbors. Below, we say that a disk $D \in \mathcal{D}$ is *combinatorially covered* (or just *covered*) by a set of disks if each point in $\mathcal{P} \cap D$ belongs to the union of these disks.

Algorithm 3 sorts all disks with respect to the positions of the points of intersection with the boundary of the quadrant Q. The main step of the algorithm, called *2-refinement*, traverses the disks in sorted order recursively dropping disks covered by their immediate neighbors on each side. For the purpose of simplifying both the algorithm and the proof of correctness, two dummy disks are added as the first and last disks in the sorted sequence before 2-refinement. Each dummy disk covers one private (not covered by any other disk) dummy point which ensures that the dummy disks are always part of any disk cover.

Algorithm 3: Combinatorial Disk Covering

Input: The set of unit disks $\mathcal{D}$, $|\mathcal{D}| = m$, centered inside the unit disk A, set of points $\mathcal{P}$ in the quadrant Q outside A such that $\mathcal{P} \subseteq \cup\{D \in \mathcal{D}\}$
Output: Minimum size subset $\mathcal{F} \subseteq \mathcal{D}$ such that $\mathcal{P} \subseteq \cup\{D \in \mathcal{F}\}$

1. For each $D_i \in \mathcal{D}$ find l_i and r_i, the two points of intersection between the boundaries ∂D_i with ∂Q. We assume that $l_j < r_j$ in a fixed orientation of ∂Q. Renumber the disks in $\mathcal{D}$ such that either $l_i < l_{i+1}$ or $l_i = l_{i+1}$ and $r_i < r_{i+1}$ for every $i = 1, \ldots, m-1$.
2. Add at the beginning and at the end of $\mathcal{D}$ dummy disks D_0 and D_{m+1} each containing a private dummy point (i.e., a point covered only by D_0, respectively D_{m+1}) and not covering any other point of $\mathcal{P}$.
3. **Combinatorial 2-refinement:**
   Initialize a stack S with D_0 and D_1
   For $i = 2, \ldots, m+1$ do
   While $top(S)$ is covered by the disk under $top(S)$ together with the disk D_i, pop the stack S
   Push D_i on the stack S.
4. Remove the dummy disks D_0 and D_{m+1}.
5. Output the set $\mathcal{F}$ of disks from the stack S.

THEOREM 10. *Algorithm 3 gives an optimal solution for the 1-hop Disk Cover problem in a quadrant and can be implemented in $O(n^2)$ time.*

Theorems 1 and 10 imply:

COROLLARY 11. *Combined with Algorithm 3, Algorithm 1 runs in $O(n \log n)$ time and has an approximation ratio of 3 for the Minimum Forwarding Set problem.*

In the rest of the section, we will prove Theorem 10. Without loss of generality, we assume that the dummy disks D_0 and D_{m+1} are part of the input. Under this assumption any disk cover should contain D_0 and D_{m+1}.

The proof is organized as follows. We first introduce the important topological notion of Δ-configuration and discuss its properties.

We first show that, @@@@@@@@@@ with one exception, at any moment during the execution of the algorithm, no disk in the stack covers any another disk in the stack.

Then we show that the disks in the stack never form Δ-configurations.

Next we show that, in the maximum-area optimum disk cover, every two consecutive optimum disks combinatorially cover all disks between them. Finally, we show that the set of disks remaining in the stack after 2-refinement and the maximum-area optimal disk cover are interleaved, and hence have the same size.

We say that (i,j,k), $0 \leq i < j < k \leq m = 1$, form a Δ-*configuration* if the three arcs $\partial D_i \cap Q$, $\partial D_j \cap Q$ and $\partial D_k \cap Q$ intersect pairwise, and the walk along $\partial D_j \cap Q$ starting from l_j towards r_j meets $\partial D_k \cap Q$ no later than $\partial D_i \cap Q$ (see Figure 7). A simple topological argument shows that if (i,j,k) form a Δ-configuration, then $D_j \cap Q \subseteq (D_i \cup D_k)$. Also, if $(D_i \cap D_k) \setminus D_j$ is nonempty, either (i,j,k) form a Δ-configuration, or D_i covers D_j, or D_k covers D_j.

We start with a simple property of the algorithm.

LEMMA 12. *At any moment, in the stack, no disk is covered by another, with the possible exception that the topmost disk. If the topmost disk is covered by another disk, then it covered by the disk right under it in the stack.*

Proof. Consider the first moment a disk D_i covers a disk D_j, with D_j not the topmost element of the stack. Then D_i is the topmost element of the stack.

If D_j is right under D_i, then D_j should have been popped from S before D_i is pushed. Otherwise, let $D_q \neq D_j$ be the disk right under D_i in the stack. If walking on $\partial D_q \cap Q$ starting at l_q towards r_q, we do not meet $\partial D_j \cap Q$, then $D_q \subseteq D_j \subseteq D_i$, and D_q should have been popped from S before D_i is pushed. If walking on $\partial D_q \cap Q$ starting at l_q towards r_q, we meet $\partial D_j \cap Q$ after meeting $\partial D_i \cap Q$, then $D_q \setminus D_i \subseteq D_j \setminus D_i$ and therefore $D_q \subseteq D_i$, a contradiction as before. If walking on $\partial D_q \cap Q$ starting at l_q towards r_q, we meet $\partial D_j \cap Q$ no later than $\partial D_i \cap Q$, then $D_j \subseteq (D_i \cap D_j) \subseteq D_q$, contradicting the fact that this is the first moment the lemma does not hold.

Now let D_j be the topmost element of the stack, and assume D_i covers D_j. Let D_q be the disk right under D_j in the stack. If $D_q = D_i$, we are done.

If $D_q \subseteq D_i$, by induction we can assume that D_t, the disk right under D_q in the stack, also covers D_q. Then D_q should have been popped from S before D_i is pushed,

If $D_q \not\subseteq D_i$, then the walk on $\partial D_q \cap Q$ starting at l_q towards r_q meets $\partial D_i \cap Q$ no later than meeting $\partial D_j \cap Q$. Indeed. if the walk on $\partial D_q \cap Q$ starting at l_q towards r_q meets $\partial D_j \cap Q$ before $\partial D_i \cap Q$, then $D_q \setminus D_i \subseteq D_j \setminus D_i = \emptyset$, contradicting $D_q \not\subseteq D_i$. But then $D_j \subseteq D_i \cap D_j \subseteq D_q$, and we are done. ■

LEMMA 13. *The stack S never contains a Δ-configuration.*

Proof. Assume for a contradiction that there is a Δ-configuration on the stack S. Let (i,j,k) be a Δ-configuration with the smallest k, and, among these, choose the one with the largest i. We will show that D_j is right under D_k in the stack, and D_i is right under D_j in the stack, but in this case D_j should have been popped from S before D_k is pushed, a contradiction.

Assume first that D_j is right under D_k in the stack, and let D_q, $q \neq i$, be the disk under D_j in the stack. By Lemma 12,

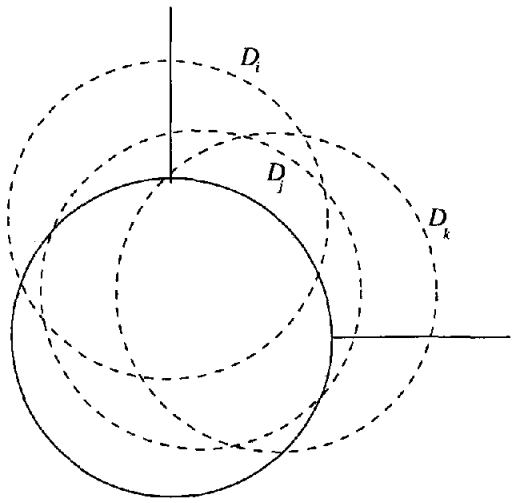

Figure 7: The Δ-configuration.

$D_q \not\subseteq D_i$ and $\partial D_q \cap Q$ must intersect $\partial D_i \cap Q$. If the walk on $\partial D_q \cap Q$ starting at l_q towards r_q meets $\partial D_i \cap Q$ before $\partial D_j \cap Q$, then D_q and D_k cover D_j, and therefore D_j should have been popped from S before D_k is pushed. If the walk on $\partial D_q \cap Q$ starting at l_q towards r_q meets $\partial D_j \cap Q$ no later than $\partial D_i \cap Q$, then (i,q,j) form an earlier Δ-configuration, a contradiction.

Now assume D_q, $q \neq j$, is right under D_k in the stack. If D_q is covered by either D_i or D_j, then by Lemma 12 D_q is covered by the disk under it in the stack, and therefore D_q should have been popped from S before D_k is pushed. So $\partial D_q \cap Q$ intersects both $\partial D_i \cap Q$ and $\partial D_j \cap Q$. If the walk on $\partial D_q \cap Q$ starting at l_q towards r_q meets $\partial D_j \cap Q$ before $\partial D_i \cap Q$, then (i,j,q) form an earlier Δ-configuration, a contradiction. If the walk on $\partial D_q \cap Q$ starting at l_q towards r_q meets $\partial D_i \cap Q$ before $\partial D_j \cap Q$, then it must also meet $\partial D_k \cap Q$ before $\partial D_j \cap Q$, and therefore (j,q,k) form a Δ-configuration, contradicting the fact that (i,j,k) is the Δ-configuration with $k-i$ minimum.

So in all cases we obtain a contradiction. ■

LEMMA 14. *Let* $\mathrm{OPT} = \{D_{t_1}, D_{t_2}, \ldots, D_{t_{opt}}\}$ *be the maximum-area optimum disk cover, where* $0 = t_1 < t_2 < \ldots < t_{opt}$. *Then, for every* $t_i < q < t_{i+1}$, $i = 0, \ldots, opt-1$, D_q *is combinatorially covered by* $D_{t_i} \cup D_{t_{i+1}}$.

Proof. Assume by contradiction that there is a disk $D_q \notin OPT$, with $t_i < q < t_{i+1}$ such that D_q is not covered by D_{t_i} and $D_{t_{i+1}}$ (see Figure 8). Let $p \in D_q \setminus (D_{t_i} \cup D_{t_{i+1}})$. W.l.o.g., assume that $p \in D_{t_j}$ for $j < i$. Then (t_j, t_i, q) form a Δ-configuration, and $D_{t_i} \subseteq D_{t_j} \cup D_q$. Hence, $OPT' = (OPT \setminus D_{t_i}) \cup D_q$ geometrically covers $\cup\{D \in OPT\}$ and is itself an optimum disk cover. The contradiction follows from the fact that $\cup\{D \in OPT\}$ is a strict subset of $\cup\{D \in OPT'\}$. Indeed, D_q is not combinatorially covered by the other disks in OPT' and hence participates in the skyline of OPT'. Let s be the skyline arc of D_q. The interior of

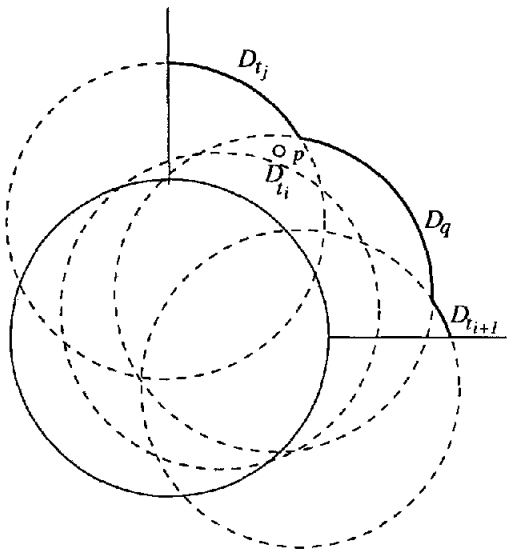

Figure 8: Replacing D_{t_i} with D_q increases the area of the optimum disk cover.

s has no points in common with the disks in $OPT' \setminus D_q = OPT \setminus D_{t_i}$. Furthermore, since D_{t_i} is fully covered by OPT', the interior of s has no points in common with D_{t_i}. Thus, the interior of s is in $(\cup\{D \in OPT'\}) \setminus (\cup\{D \in OPT\})$. ■

LEMMA 15. $|\mathcal{F} \cap \{D_{t_i}, D_{t_i+1}, \ldots, D_{t_{i+1}-1}\}| \leq 1$ *for any* $i = 0, \ldots, opt$.

Proof. Assume that just after t_{i+1} is pushed on the stack, $\mathcal{F} \cap \{D_{t_i}, D_{t_i+1}, \ldots, D_{t_{i+1}-1}\} \supseteq \{D_x, D_y\}$, with $x < y$ and $y - x$ maximum possible.

We first show that D_y is covered by D_x and $D_{t_{i+1}}$. Indeed, Lemma 14 implies that D_y is covered by D_{t_i} and $D_{t_{i+1}}$. If $x = t_i$, we are done. Otherwise, D_{t_i} is covered by D_x and D_v, where D_v is the disk under D_x in the stack, therefore, D_y is covered by D_v, D_x and $D_{t_{i+1}}$. If D_y has a point in $D_v \setminus D_x$, then (v, x, y) is a Δ-configuration, a contradiction to Lemma 13. Thus D_y is covered by D_x and $D_{t_{i+1}}$.

Let D_z be the disk under D_y in the stack. We will show that D_y is covered by D_z and $D_{t_{i+1}}$ and thus D_y should have been removed from the stack before $D_{t_{i+1}}$ is pushed on the stack. Indeed, D_y is covered by D_x and $D_{t_{i+1}}$ and if D_y has a point in $D_x \setminus D_z$, then (x, z, y) is a Δ-configuration, a contradiction to Lemma 13. ■

Proof of Theorem 10. The approximation factor follows from Lemma 15. Algorithm 3 can be implemented to run in $O(n^2)$ time, where $n = m + p$ is the number of centers, m, plus the number of points, p. Indeed, Step 1 needs $O(m \log m)$ time, and each of the m iterations in combinatorial 2-refinement can be implemented in $O(p)$ time by traversing all points in $\mathcal{P}$. ■

5. THE GENERAL MINIMUM DISK COVER PROBLEM

In this section we describe a constant-factor approximation algorithm for the following

Minimum Disk Cover Problem. Given a set of unit disks $\mathcal{D}$ and a set of points $\mathcal{P}$ in the Euclidean plane, find a minimum-size subset $\mathcal{F} \subseteq \mathcal{D}$, such that $\mathcal{P} \subseteq \cup\{D \in \mathcal{F}\}$.

This problem is NP-Hard since it contains as a special case Dominating Set in unit-disk graphs, a problem shown to be NP-Hard in [4]. A polynomial-time algorithm with constant approximation ratio for Minimum Disk Cover was first provided by [2].

If we can obtain a constant ratio for covering an equilateral triangle with sizes equal to 1, we can obtain a constant ratio for the whole plane, by tiling the plane into triangles and separately covering all the triangles, and using the fact that one disk in the optimum can only cover points in a constant number of triangles.

Let ABC be such a triangle. If no point of $\mathcal{P}$ is in the triangle, there is nothing to be done. Also, if there is a disk $D \in \mathcal{D}$ whose center is in the triangle, then D covers all the triangle. So, in the following, we assume all the points are in the triangle, and all the centers of disks in $\mathcal{D}$ are outside the triangle.

The algorithm has four phases:

1. After removing those disks that do not intersect the triangle, partition the remaining disks into three sets $\mathcal{D}_1$, D_2, and $\mathcal{D}_3$, such that all the centers of the disks in D_1 are on the other side of the line AB than C, all the centers of the disks in D_2 are on the other side of the line BC than A, and all the centers of the disks in D_3 are on the other side of the line AC than B. If a disk could be put in more than one D_i, pick one arbitrarily.

2. For $i = 1, \ldots, 3$, let $Q_i = B$ be the triangle and let J_i be the line which separates the centers of the disks of $\mathcal{D}_i$ from the interior of the triangle. Find the skyline as in Section 3, and compute $\mathcal{F}_i$, the set of disks containing some arc of the skyline.

3. Write the natural Integer Programming formulation involving only the disks in $\mathcal{F}_1 \cup \mathcal{F}_2 \cup \mathcal{F}_3$. Solve the Linear Programming relaxation.

4. Round the linear programming optimum to an integer solution, as described in Subsection 5.1.

Later we prove Theorem 16, which claims that the algorithm described above has approximation ratio at most 6 for the problem of covering the points inside the triangle.

First, we note that Lemma 2 holds easily when J_i is a straight line. For each $\mathcal{F}_i$, Lemma 8 also holds. Let $\mathcal{F} = \mathcal{F}_1 \cup \mathcal{F}_2 \cup \mathcal{F}_3$, and assume $\mathcal{F}$ is sorted with $\mathcal{F}_1$ (which is sorted) followed by the sorted $\mathcal{F}_2$, and followed by the sorted

$\mathcal{F}_3$. Lemma 9 also holds, and therefore $\mathcal{F}$ contains a solution at most twice *opt*, the size of an optimum solution.

5.1 Rounding

We use the natural IP, with variables x_D, for $D \in \mathcal{F}$:

$$\text{minimize} \sum_{D \in \mathcal{F}} x_D$$

$$\text{subject to} \sum_{D\,:\,P \in D} x_D \geq 1 \quad \forall P \in \mathcal{P} \tag{1}$$

$$x_D \in \{0,1\} \quad \forall D \in \mathcal{D}. \tag{2}$$

Let LP be the linear programming relaxation of IP, obtain by replacing the constraints 2 by

$$x_D \geq 0 \quad \forall D \in \mathcal{D}. \tag{3}$$

Let Z^*_{IP} the value of the IP optimum. As argued above, we have $Z^*_{IP} \leq 2\ opt$.

Let y be a (fractional) solution to LP. For a point $P \in B$, the set of disks covering it consists of at most three intervals, say I^P_1, I^P_2, and I^P_3. For one of the three intervals, which we call simply I^P, we have: $\sum_{D \in I^P} y_D \geq 1/3$.

Consider the following integer program, which we call IP', with variables x_D, for $D \in \mathcal{F}$:

$$\text{minimize} \sum_{D \in \mathcal{F}} x_D$$

$$\text{subject to} \sum_{D \in I^P} x_D \geq 1 \quad \forall P \in \mathcal{P} \tag{4}$$

$$x_D \in \{0,1\} \quad \forall D \in \mathcal{D}. \tag{5}$$

Let LP' be the linear programming relaxation of IP'. The matrix of IP' is totally unimodular (see [5], page 200), and $3y$ is a solution to LP'. Therefore IP' has a solution of size at most $3\sum_{D \in \mathcal{F}} y_D$, and an optimum for IP' can be found easily by the greedy algorithm, as described at the end of the proof of Theorem 3. Now, if y is an optimum solution to LP, then $\sum_{D \in \mathcal{F}} y_D \leq Z^*_{IP} \leq 2\ opt$, and therefore the solution found by the greedy algorithm has size at most $6opt$.

Rounding consists of finding for each point $P \in B$ the interval I^P, and then using the greedy algorithm to hit each I^P with elements of $\mathcal{F}$. In conclusion, we proved:

THEOREM 16. *The algorithm described in this section has approximation ratio at most 6 for the covering points inside an equilateral triangle with sizes equal to 1 with unit-disks from a fixed set $\mathcal{D}$.*

Since a single disk from the optimum solution can cover points in at most 17 triangles of the tiling we conclude

COROLLARY 17. *There is a 102-approximation algorithm for the Minimum Disk Cover problem.*

6. EXPERIMENTAL RESULTS

We compared the two algorithms proposed for the Minimum Forwarding Set problem on random instances generated as follows. The polar coordinates for the specified number of 1- and 2-hop neighbors of a source node placed at the origin were generated by choosing for each point the angle uniformly between $[0, 2\pi)$ and the radius uniformly from the interval $(0, 1]$ for 1-hop neighbors and uniformly from the interval $(1, 2]$ for 2-hop neighbors. The two algorithms were then applied to the instance obtained by deleting all 2-hop neighbors not covered by any 1-hop neighbor. The algorithms were implemented as Java applets. As expected, the geometric algorithm is much faster than combinatorial one, by up to two orders of magnitude in our experiments, which were run on a Pentium II 300MHz PC.

Table 1 reports the average results over 100 instances generated for each instance size. We remark that, although the geometric disk covering has a worst case approximation guarantee twice larger than that of the combinatorial disk covering algorithm, on the random instances used in our experiments its solution is larger on the average by only 17–44%.

7. CONCLUSIONS

In this paper we presented a geometric $O(n \log n)$ 6-approximation algorithm and a combinatorial $O(n^2)$ 3-approximation algorithm for selecting forwarding neighbors in wireless ad-hoc networks, significantly improving both the running time and the approximation ratio of the best previously known algorithm. An extension of our method can be used to obtain an alternative constant-ratio polynomial-time algorithm for the Minimum Disk Cover problem.

We mention that Theorem 10 is true in the following more general setting. Let J be an infinite simple Jordan curve which separates the plane into exactly two regions, and let B_i one of these two regions. Let all points $\mathcal{P}$ be in B_i, and each D_j be a region bordered by a simple closed Jordan curve $\partial\mathcal{D}_j$. Also, each $\partial\mathcal{D}_j$ intersects the infinite curve J in exactly two points, and, for any two disks D_j and D_k, $\partial\mathcal{D}_j \cap \partial\mathcal{D}_k \cap B_i$ has at most one point. Moreover, whenever two of the curves above intersect, they cross each other. Then the combinatorial disk covering algorithm 3 solves the covering problem exactly.

Refinement techniques can also be applied to a fractional solution to the natural linear program LP to obtain a rounding procedure with a ratio of 2 when $\mathcal{P}$ and the centers of $\mathcal{D}$ are separated by a straight line. Then, as in Section 5, it follows that the linear program LP has constant integrality ratio for the general problem. However, when the disks in $\mathcal{D}$ are weighted, we do not know the integrality ratio of the corresponding integer and linear programs. The linear program is given below:

$$\text{minimize} \sum_{D \in \mathcal{D}} w_D x_D$$

Instance Size		Geometric Algorithm		Combinatorial Algorithm		Relative
# 1-Hop	# 2-Hop	Skyline	Solution	1-Refined	Solution	Error
6000	2000	44	36	197	29	24%
3000	1000	40	28	141	24	17%
2000	1000	38	27	137	22	23%
1000	5000	29	23	93	16	44%

Table 1: Comparison of the 1-Hop Disk Covers produced by the geometric and combinatorial algorithms.

$$\text{subject to} \sum_{D\,:\,P\in D} x_D \geq 1 \quad \forall P \in \mathcal{P} \tag{6}$$

$$x_D \geq 0 \quad \forall D \in \mathcal{D} \tag{7}$$

8. ACKNOWLEDGMENTS

We thank Yuchen Wu for his help with the implementation of the proposed algorithms.

9. REFERENCES

[1] J. Broch, D. B. Johnson, and D. A. Maltz, The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks, IETF Internet Draft, draft-ietf-manet-dsr-05.txt, March 2001.

[2] H. Bronnimann and M.T. Goodrich, Almost Optimal Set Covers in Finite VC-Dimension. *Proc. 10th ACM Symp. on Computational Geometry (SCG)*, 1994, 293-302.

[3] V. Chvátal. A greedy heuristic for the set-covering problem. *Mathematics of Operation Research*, 4(3):233–235, 1979.

[4] B. N. Clark, C. J. Colbourn, and D. S. Johnson, "Unit Disk Graphs", *Discrete Mathematics*, 86:165–177, 1990.

[5] M. Grötschel, L. Lovász, and A. Schrijver *Geometric Algorithms and Combinatorial Optimization*, second edition, Springer-Verlag, 1991.

[6] Z. J. Haas, M. R. Pearlman, and P. Samar. The Interzone Routing Protocol (IERP) for Ad Hoc Networks, IETF Internet Draft, draft-ietf-manet-zone-ierp-00.txt, January 2001.

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

[8] 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.

[9] P. Jacquet, P. Muhlethaler, A. Qayyum, A. Laouiti, L. Viennot, and T. Clausen, Optimized Link State Routing Protocol, IETF Internet Draft, draft-ietf-manet-olsr-04.txt, March 2001.

[10] M. V. Marathe, H. Breu, H. B. Hunt III, S. S. Ravi and D. J. Rosenkrantz, "Simple Heuristics for Unit Disk Graphs", *Networks*, Vol. 25, 1995, pp. 59–68.

[11] B.M.E. Moret and H.D. Shapiro. *Algorithms from P to NP, Volume I: Design and Efficiency.* Benjamin/Cummings, 1991.

[12] S.-Y. Ni, Y.-C. Tseng, Yuh-Shyan Chen, and J.-P. Sheu, The Broadcast Storm Problem in a Mobile Ad Hoc Network", *Proceedings of the fifth annual ACM/IEEE international conference on Mobile computing and networking*, 1999, Pages 151 - 162.

[13] C. E. Perkins, E. M. Royer, and S. Das, Ad Hoc On Demand Distance Vector (AODV) Routing, IETF Internet Draft, draft-ietf-manet-aodv-08.txt, March 2001.

[14] P. Sinha, R. Sivakumar, and B. Vaduvur, Enhancing Ad Hoc Routing with Dynamic Virtual Infrastructures, to appear in *IEEE INFOCOM* 2001.