# Incremental Construction of $k$-Dominating Sets in Wireless Sensor Networks

Mathieu Couture, Michel Barbeau,
Prosenjit Bose and Evangelos Kranakis

School of Computer Science
Carleton University
September 18 2006

**Abstract**

Given a graph $G$, a $k$-dominating set of $G$ is a subset $S$ of its vertices with the property that every vertex of $G$ is either in $S$ or has at least $k$ neighbors in $S$. We present a new incremental local algorithm to construct a $k$-dominating set. The algorithm constructs a monotone family of dominating sets $D_1 \subseteq D_2 \ldots \subseteq D_i \ldots \subseteq D_k$ such that each $D_i$ is an $i$-dominating set. For unit disk graphs, the size of each of the resulting $i$-dominating sets is at most six times the optimal.

## 1 Introduction

An *ad hoc network* is a special type of wireless network where no node has a priori knowledge about the other nodes. Constructing and maintaining a structure allowing nodes to communicate with each other is one of the main challenges of ad hoc networks. Sensor networks are a specific type of ad hoc networks dedicated to a specific task: sensing (light, temperature, humidity, etc.). The dominating set structure helps ad hoc and sensor networks to perform routing. In sensor networks, dominating sets also help the sensing task itself. Since nodes located close to each other sense similar values, only a dominating set of the nodes is needed to monitor an area. This helps prolonging the network's lifetime by turning off the nodes that are not in the dominating set, thereby extending the battery life of these nodes.

Sensor networks typically contain more nodes and each node has less memory than in general ad hoc networks. Therefore, it is important to design algorithms with low memory complexity. An algorithm is *local* if the information needed by a node to perform its computation only concerns its direct neighbors. The amount of memory needed by each node to execute a local algorithm only depends on the number of its neighbors and not on the network size.

Sensor nodes are more error-prone than nodes in general ad hoc networks. They have limited energy resources and need to be periodically redeployed by adding new nodes to the network. The fact that they are error-prone calls for *fault-tolerance* in the design of algorithms for such networks.

We address the problem of locally constructing $k$-dominating sets on unit disk graphs. Unit disk graphs are structures used to model ad hoc and sensor networks. A $k$-dominating set of a graph is a subset of its nodes where each node of the graph is either in the $k$-dominating set or has at least $k$ neighbors in the $k$-dominating set. Kuhn *et al.* [15] introduced the idea of exploiting clique properties to construct a $k$-dominating set from a 1-dominating set. The *performance ratio* of a $k$-dominating set algorithm is defined as the ratio of the size of the $k$-dominating set it produces over the size of an optimal (minimum) $k$-dominating set. Kuhn *et al.* [15] claim that the proposed algorithm has an expected performance ratio of $O(1)$ for unit disk graphs. However, in Section 5 we provide a counter-example to one of the claims in their proof of Theorem 5.7

We generalize dominating set algorithms based on the idea of maximal independent sets [2, 19, 25] to obtain $k$-dominating sets. A subset $S$ of the nodes of a graph $G$ is said to be *independent* if it does not contain two adjacent nodes. It is *maximal* if it does not have a proper independent superset. It is straightforward to show that a maximal independent set is also a dominating set. Our algorithm is local and, on unit-disk graphs, has a deterministic performance ratio of six. It is not position-aware, which means that nodes do not need to know their coordinate in the plane. It also constructs the $k$-dominating set incrementally. More specifically, it constructs a monotone family of dominating sets $D_1 \subseteq D_2 \ldots \subseteq D_i \ldots \subseteq D_k$ such that each $D_i$ is an $i$-dominating set. Incremental construction of $k$-dominating sets is helpful when redeploying sensor networks. When senor nodes in the $k$-dominating set run out of batteries or experiment failure for diverse reasons, the $k$-dominating set has to be reconstructed. With an incremental algorithm, reconstruction of a $k$-dominating set can be done by keeping the current dominators.

The $k$-dominating set problem has been addressed by Dai and Wu [6] and Kuhn *et al.* [15]. However, our algorithm the only one which has a constant deterministic performance ratio. It is also the only one to provide an explicit incremental construction. The algorithm presented in [6] elects all nodes at the same time. It does not allow to augment a $j$-dominating set to a $k$-dominating with $k \geq j$. Also, their algorithm offers probabilistic guarantees both on the performance ratio and the correctness of the output. The algorithm presented in [15] needs to decide about a 1-dominating set which later on plays a crucial role. Should nodes in that 1-dominating set experiment failure (i.e. running out of batteries during the execution of the algorithm), the whole algorithm could fail. Such a failure is likely to happen when reconstructing the $k$-dominating set after network redeployment. In such situation, it may happen that nodes from the original 1-dominating set have drained their batteries. Also, the algorithm presented in [15] requires nodes to either know their geographic position or to be able to modify their communication range as the algorithm runs and offers an expected performance ratio. Our algorithm is not position-aware, do not have any requirement on the ability to dynamically modify the communication range and can construct the $k$-dominating set from another $j$-dominating set with $j \leq k$.

The rest of this paper is organized as follows: in Section 2, we review related work on dominating sets. In Section 3, we present our algorithm. In Section 4, we analyze the performance ratio of our algorithm. In Section 5, we compare it with the one proposed by Kuhn *et al.* [15]. In Section 6, we present some simulation results. We draw conclusions in Section 7.

## 2 Related Work

A subset $S$ of the vertices of a graph $G$ is *dominating* $G$ if every node of $G$ is either in $S$ or has at least one neighbor in $S$. The dominating set problem consists of finding a dominating set of minimum size. It is a special case of the set-cover problem [13]. In the set-cover problem, the input is a set of sets $\mathcal{S}$, and the output is a subset $\mathcal{S}'$ of $\mathcal{S}$ such that the union of the sets in $\mathcal{S}'$ is the same as the union of the sets in $\mathcal{S}$. Once again, $\mathcal{S}'$ should have minimum size. Set-cover is one of the first problems that have been shown to be NP-complete [13]. The dominating set problem itself has been shown to be NP-complete by a reduction from the vertex-cover problem. Johnson [12] proposed a greedy approximation algorithm to the set-cover problem (and thus to the dominating set problem). Its performance ration is $H(\Delta)$, where $\Delta$ is the size of the largest set in $S$ and $H$ is the harmonic function. We are not aware of any better approximation algorithm.

Alber *et al.* [1] proposed a way to reduce the size of the dominating set decision problem. More explicitly, they proposed a polynomial time algorithm which transforms an instance of the decision problem *does a given graph $G$ have a dominating set of size $k$?* to another smaller instance of the same problem. For the special case of planar graphs, the graph output by their algorithm is guaranteed to have at most $335k$ nodes, which is independent of the size of the original graph. However, this algorithm does not construct an actual dominating set.

Ad hoc and sensor networks are often modeled using a unit disk graph. A unit disk graph is a graph where vertices are points in the plane and there is an edge between two points within unit distance. The dominating set problem on unit disk graphs is NP-complete [4, 20].

The $k$-dominating set problem [6, 15, 21] is a variant of the dominating set problem. A subset $S$ of the vertices of a graph $G$ is $k$-dominating if every node of $G$ is either in $S$ or has at least $k$ neighbors in $S$. A node is said to be $k$-dominated by $S$ if it is either in $S$ or has at least $k$ neighbors in $S$. The $k$-dominating set problem, as addressed in this paper, is the following:

**Problem 2.1** *Given a graph $G$, find a $k$-dominating set of $G$ whose size is minimum.*

Other variants of this problem are the distance-$k$ dominating set problem [16], max $k$-cover problem [8] and connected dominating set problem. In the distance-$k$ dominating set problem, the goal is to find a subset $S$ of the nodes of a graph $G$ such that every node of $G$ is either in $S$ or at most $k$ hops away from a node which is in $S$. In the max $k$-cover problem, the goal is to find a subset of size $k$ of $S$ that covers the maximum number of nodes. In the connected dominating set problem, one needs to find a dominating set that is also connected.

Note that there is often confusion between the distance-$k$ version of the problem and the one studied in this paper. For both problems, there are papers in which they are referred to as *the* $k$-dominating set problem [16, 21].

It also happens that a $k$-dominating set is defined as a dominating set of size $k$ [1]. The $k$-dominating set problem discussed in the current paper has also been referred as the *k-fold* dominating set problem [15].

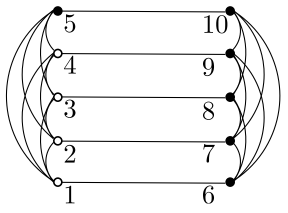

Figure 1: Performance ratio of $\frac{n}{4}$ for the algorithm presented in [28].

The connected dominating set problem has been originally addressed by Guha and Khuller [10]. They proposed three approximation algorithms. Two algorithms are based on growing a tree. The third algorithm consists of growing a forest. The algorithms have a performance ratio of $O(\log n)$, where $n$ is the number of vertices in the graph. Das and Bharghavan [7] implemented these non-local algorithms in a distributed framework.

Wu and Li [27] introduced a local algorithm based on the notion of replacement path: *If all of your neighbors form a clique, then do not elect yourself as a dominator. Otherwise, elect yourself as a dominator.* Wan *et al.* [25] showed that the worst case performance ratio of that algorithm is at least $\frac{n}{2}$. Figure 1 depicts the worst case. Nodes 1 to 5 and 6 to 10 form two cliques, and facing nodes (nodes with same $y$-coordinate) are connected. In that case, there is a connected dominating set of size 2 (nodes 1 and 6), but the algorithm would elect every node as a dominator. On the other hand, two nodes are connected by a path of dominators having a minimum number of hops [27]. Wu and Li [28] further improved the algorithm by observing that if a node $u$ has two connected neighbors $v$ and $w$ such that the neighborhood of $u$ is included in the union of the neighborhoods of $v$ and $w$, and if the identifier of $u$ is smaller then the identifiers of $v$ and $w$, then $u$ should not be elected as a dominator. The proof in [25] adjusts easily to show that the worst case performance ratio in that case is at least $\frac{n}{4}$. Again in Figure 1, all the black nodes get elected while only two nodes suffice. Instead of using only two connected neighbors, Dai and Wu [5] then proposed to use $k$ connected neighbors. They called this improved version of the algorithm *Rule k*. Hansen and Schmutz [11] analyzed the expected size of the produced connected dominating set. Finally, the *Rule k* algorithm has been further generalized by Wu and Dai [26] in the form of a generic coverage condition. The generic coverage condition is the following: *If for any two of your neighbors $v$ and $w$, there exists a replacement path between $v$ and $w$ such that all nodes on that path have identifiers higher than yours, then do not elect yourself a as dominator. Otherwise, elect yourself as a dominator.* The reason why it is more general than *Rule k* is that there is no bound on the size of the replacement path and nodes on the replacement path need not be neighbors of node $u$.

Stojmenovic *et al.* [24] proposed to improve the algorithm presented in [27] by giving some priority to nodes having higher degree. Another stream of research in connected dominating sets uses the notion of independent set. Marathe *et al.* [19] pointed out that using maximal independent sets in unit disk graphs could provide a simple way to approximate a minimum dominating set within a factor of five. Alzoubi *et al.* [2] and Wan *et al.* [25] showed that a similar strategy can lead to a performance ratio of 8 in the case of connected dominating sets. Cardei *et al.* [3] obtained similar results. Li *et al.* [17, 18] successively reduced the performance ratio to $5.8 + \ln 4$ and $4.8 + \ln 5$, respectively.

To the best of our knowledge, not much work has been done on the $k$-dominating set problem. The work we are aware of are from Dai and Wu [6], Moscibroda and Wattenhofer [21] and Kuhn *et al.* [15]. Dai and Wu [6] proposed three different algorithms. The first one is randomized and gives a $k$-connected $k$-dominating set with high probability. The second algorithm is another generalization of Wu *et al.* [29]. It is deterministic and is proven to always give a $k$-connected $k$-dominating set. In both cases, the decision is based on local or $k$-local information, which means that a node needs to be aware of the nodes that are within $k$ hops. The third algorithm proceeds as follows: randomly create a $k$-coloring of the graph and then apply any existing connected dominating set algorithm on each color. Moscibroda and Wattenhofer [21] have addressed the problem of finding $k$ *disjoint* dominating sets (the union of which makes a $k$-dominating set). Their algorithms are based on coloring schemes. Note that the problem they addressed is slightly different since the parameter $k$ is not given as an input. Instead, they find the largest $k$ such that it is possible to find $k$ disjoint dominating sets. Kuhn *et al.* [15] proposed two algorithms. The first one is for general graphs and uses a distributed rounding scheme. The second one is for unit disk graphs and works in two phases. The first phase reuses an algorithm from Gao *et al.*

[9] to construct a 1-dominating set. In the second phase, the 1-dominators select the other dominators. The authors give a proof that the expected performance ratio of their algorithm is $O(1)$.

## 3 Algorithm

Alzoubi *et al.* [2] and Wan *et al.* [25] addressed construction of a *connected* dominating set. Their algorithm consists of two phases. The first phase constructs a maximal independent set. A maximal independent set is also a dominating set. In this section, we generalize that first phase of the algorithm presented in [2, 25] to obtain a $k$-dominating set. Our generalization augments a $(k-1)$-dominating set in order to obtain a $k$-dominating set. More specifically, we want to construct a monotone $k$-dominating family.

**Definition 3.1** *A* $k$-dominating family *is a sequence $D_1, D_2, \dots, D_k$ of subsets of vertices of the unit disk graph such that for all $i = 1, 2, \dots, k$, $D_i$ is an $i$-dominating set. A* monotone $k$-dominating family *is a $k$-dominating family with the additional property that the sequence of dominating sets is monotonically increasing under inclusion, i.e. $D_1 \subseteq D_2 \subseteq \cdots \subseteq D_k$.*

The key idea of our algorithm is that we first construct a 1-dominating set by constructing a maximal independent set. Then, we construct a maximal independent set of the nodes that are not 2-dominated, which gives a 2-dominating set. We repeat the procedure until we have a $k$-dominating set. The construction of each dominating set is similar to the approach in [2, 25].

We now present an overview of our algorithm. Every node has a unique identifier. In initialization phase, each node sends its identifier to its immediate neighbors. After initialization, two types of messages are used: JOIN$(id, i)$ and GIVE-UP$(id, i)$, where $id$ is the identifier of the sending node and $i = 1 \dots k$ identifies a round. These messages are only sent to immediate neighbors. The JOIN$(id, i)$ message means that the sender joins the $j$-dominating sets for $j = i \dots k$. Such a node is said to be *marked* in round $i$. The GIVE-UP$(id, i)$ message means that the sender is excluded of the $i$-dominating set. After transmitting a JOIN message, the sender remains silent. A node is said to be a *candidate* for round $i$ if it is not part of the $(i-1)$-dominating set and it has never sent the GIVE-UP$(id, i)$ message. Following the completion of the initialization phase, every node that has an identifier lower than the ones of all its immediate neighbors sends

**Algorithm 1** DOMINATING SET($id, N, k$)

**Input:** $id$, the node identifier
$N$, the list of the neighbors identifiers
$k$, the required number of dominators for a non-dominating node

**Output:** $dominator$, a boolean indicating whether the node is a dominator

**Local Variables:** $round$, the current round
$candidate$, a lookup table indicating whether or not a node $n$ is a candidate to be a dominator in round $r$ (all initial values are **true**)

```
 1: dominator ← false
 2: $round \leftarrow 1$
 3: if $id < \min(N)$ then
 4:     dominator ← true
 5:     send JOIN($id, 1$)
 6:     exit
 7: end if
 8: while $round \leq k$ do
 9:     receive $message$
10:     if $message$ is JOIN($n, r$) then
11:         send GIVE-UP($id, round$)
12:         $round \leftarrow round + 1$
13:         for $i = r$ to $k$ do
14:             $candidate[n, i] \leftarrow$ false
15:         end for
16:     end if
17:     if $message$ is GIVE-UP($n, r$) then
18:         $candidate[n, r] \leftarrow$ false
19:     end if
20:     if $id < \min\{n \in N | candidate[n, round]\}$ then
21:         $dominator \leftarrow$ true
22:         $round \leftarrow k + 1$
23:         send JOIN($id, round$)
24:         exit
25:     end if
26: end while
```

the JOIN($id, 1$) message. The rest of the algorithm is message driven. Algorithm 1 specifies how each node should behave. Note that different nodes may execute simultaneously different rounds.

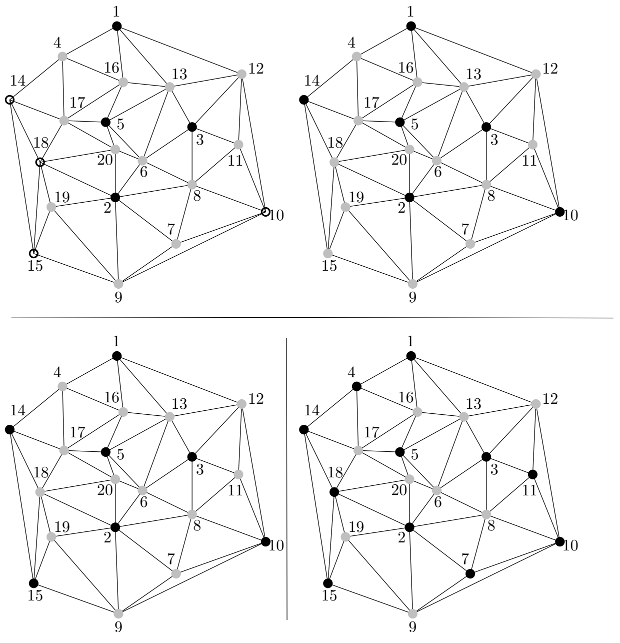

Figure 2: Marking Process Example for $k = 1$ (above) and $k = 2$ and 3 (below).

Figure 2, illustrates the marking process for $k = 1$, 2 and 3. Nodes in black are dominators. Nodes in grey are $k$-dominated. Nodes in white ate not $k$-dominated. For $k = 1$, nodes 1, 2, 3 and 5 have the smallest identifier among their (0-dominated) neighbors and thus declare themselves dominators. Initially, node 10 can not declare itself a dominator because of nodes 7, 8 and 9. However, after node 2 has declared itself a dominator, nodes 7, 8 and 9 become 1-dominated. Node 10 is then allowed to declare itself a dominator. The same reasoning applies to nodes 1 and 14. The 1-dominating set is then $\{1, 2, 3, 5, 10, 14\}$. For $k = 2$, there is only one new dominator, node 15. For $k = 3$, the new dominators are nodes 4, 7, 11 and 18.

## 4 Theoretical Properties

In this section, we give an overview of the theoretical properties of our algorithm. We first show that our algorithm computes a valid $k$-dominating set and a monotone $k$-dominating family. Then, we analyze the worst case performance ratio of our algorithm. In the latter part, we follow the general idea of Kuhn *et al.* [15]. More precisely, we first show that no unit disk can contain more than a given number of dominators (i.e. $5k$). Then, we use properties of $k$-dominating sets to show that this leads to a constant performance ratio.

**Proposition 4.1** *Let $S_i$ be the set of nodes that are marked in rounds $j = 0 \dots i$ of Algorithm 1. Then $S_i$ is an $i$-dominating set.*

*Proof:* We proceed by induction on $i$. To make the things simpler, we say that $i$ ranges from 0 to $k$. The round zero chooses the empty set as a 0-dominating set. The base case is trivial, since all nodes of the graph have at least zero neighbors in the empty set. For the induction case, We have to show that if $S_i$ is a valid $i$-dominating set, then $S_{i+1}$ is also valid $(i+1)$-dominating set.

In order to do this, we proceed by contradiction. Let $n_1$ be a node that is not $(i+1)$-dominated by $S_{i+1}$. This means that it has not sent a $\textsc{join}(id, i+1)$ message. Consequently, it must have a neighbor $n_2$ with a lower identifier that is still a candidate for round $i+1$ (line 20), meaning that it is not $(i+1)$-dominated either (line 11 in $n_2$, and 17 in $n_1$). Since $n_2$ is not $(i+1)$-dominated, by the same reasoning, there must have a node $n_3$ that is not $(i+1)$-dominated and has a lower identifier than the one of $n_2$. This process allows to construct a path $n_1, n_2, \dots, n_j, \dots, n_k$ such that none of the $n_j$ is $(i+1)$-dominated, $id(n_1) > id(n_2) > \dots > id(n_j) > \dots > id(n_k)$, and $n_k$ do not have any neighbor with a lower identifier that is not $(i+1)$-dominated. Then, $n_k$ should have elected himself as a dominator, contradicting the fact that it is not $(i+1)$-dominated. Therefore, every node is $(i+1)$-dominated, which completes the inductive case. □

In order to claim that Algorithm 1 produces a monotone $k$-dominating family, we also have to show the monotonicity property.

**Proposition 4.2** *For $i = 1 \dots k$, let $S_i$ be defined as above. Then for all $i = 0 \dots k-1$, $S_i \subseteq S_{i+1}$.*

*Proof:* The monotonicity property claimed in the statement of the proposition is true by construction. That is, let $n \in S_i$. Then, it has been marked in some round $j \leq i < i+1$ and by definition of $S_{i+1}$, we have $n \in S_{i+1}$. $\square$

For unit disk graphs, we prove an upper bound on the cardinality of the $k$-dominating set computed by Algorithm 1. We first need to show that at each round, the elected nodes form an independent set.

**Proposition 4.3** *In any given round, the nodes marked by Algorithm 1 form an independent set.*

*Proof:* Suppose that in the same round, two adjacent nodes $n_1$ and $n_2$ declare themselves dominators. Without loss of generality, suppose $n_1$ has a lower identifier than $n_2$. This means that as long as $n_1$ did not send a give-up message, $n_2$ can not elect itself a dominator. But since $n_1$ never sends such a message (no node sends both a give-up and a join message), $n_2$ can never declare itself a dominator. This means that no two adjacent nodes can declare themselves dominators. $\square$

When $k = 1$, for unit disk graphs, we can use the above property to show that the set of marked nodes is not larger than five times the size of an optimal dominating set [19]. For $k > 1$, it is not the case that the set of elected nodes form an independent set and the proof is a little more complicated.

**Proposition 4.4** *Let $G = (V, E)$ be a unit disk graph, $C$ be a unit disk and $S \subseteq V$ be the set of nodes marked by Algorithm 1. Then $|S \cap C| \leq 5k$.*

*Proof:* By proposition 4.3, $S$ is the union of $k$ independent sets. Since no unit disk can contain more than 5 independent nodes [19], $S \cap C$ can not contain more than $5k$ nodes. $\square$

**Proposition 4.5** *Let $G = (V, E)$ be a graph, $S$ a subset of $V$, $t$ an integer and $OPT_k = \{v_1, \ldots, v_{|OPT_k|}\}$ an optimal $k$-dominating set of $G$. If $|S| > t|OPT_k|$, then there is at least one node $v \in OPT_k$ such that $|N(v) \cap S| > k(t-1)$, where $N(v)$ is the set formed by $v$ and its neighbors.*

*Proof:* Let $S'$ be $S \backslash OPT_k$. Since $|S'| \geq |S| - |OPT_k| > t|OPT_k| - |OPT_k|$, we have $|S'| > (t-1)|OPT_k|$. For each $v_i \in OPT_k$, define $S_i$ as $N(v_i) \cap S'$.

Since each node in $S'$ is adjacent to at least $k$ nodes in $OPT_k$, we have that

$$\sum_{i=1}^{opt_k} |S_i| \geq k|S'| > k(t-1)|OPT_k|$$

Therefore, by the pigeonhole principle, one of the $S_i$ contains more than $k(t-1)$ nodes. The result follows from the fact that $S_i \subseteq N(v_i) \cap S$. $\square$

The two last propositions thus allow us to prove our performance ratio:

**Theorem 4.6** *Let $G = (V, E)$ be a unit disk graph, $S \subseteq V$ the set of nodes marked by Algorithm 1 and $OPT_k$ an optimal $k$-dominating set. Then $|S| \leq 6|OPT_k|$. In other words, the performance ratio is not greater than six.*

*Proof:* Suppose $|S| > 6|OPT_k|$. By proposition 4.5, there is at least one node $v \in V$ such that $|N(v) \cap S| > 5k$. But this contradicts proposition 4.4, and therefore $|S| \leq 6|OPT_k|$. $\square$

As stated before, for $k = 1$ it is known that no independent set can be larger than five times the size of an optimal dominating set [19]. We then have a leap from five to six when we generalize the maximal independent set algorithm for dominating sets to $k$-dominating sets. At first sight, it may be a bit surprising but that leap actually comes directly from the definition of a $k$-dominating set: the only nodes which need to have at least $k$ neighbors in the dominating set are the ones that are not in the dominating set. Nodes that are in the $k$-dominating set do not need to have $k$ neighbors in the dominating set.

As we shall see in the next section, it is that property which motivates our concerns about the proof of Kuhn *et al.* [15] for the performance ratio of their algorithm. We did not verify whether this has a major impact on their results, but we do not believe this is the case.

We now show that for any $k$, there exists graphs for which our algorithm has a performance ratio of five. It is an open question whether or not it is possible to close the gap between five and six. Before, we need the following lemma:

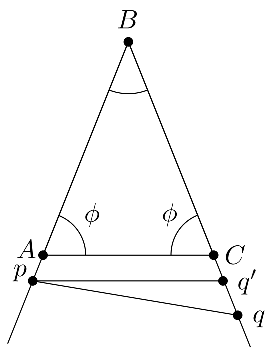

Figure 3: Lemma 4.7.

**Lemma 4.7** *Let $\triangle ABC$ be an isosceles triangle such that $\angle BAC = \angle ACB = \phi$, $p$ be a point located on the line $AB$ such that $A$ is between $p$ and $B$, and $q$ be a point located on the line $CB$ such that $C$ is between $q$ and $B$. Then $|pq| > |AC|$.*

*Proof:* If $|pB| = |qB|$, then $\triangle pBq$ is similar to $\triangle ABC$, and $|pB| > |AB|$ implies $|pq| > |AC|$. Suppose now that $|pB| < |qB|$, and let $q'$ be the point located on the line $CB$ such that $C$ is between $q'$ and $B$ and $|q'B| = |pB|$. By the first case, $|pq'| > |AC|$. Now, since $\triangle ABC$ is isosceles, $\phi < \frac{\pi}{2}$, and since $\angle pq'q = \pi - \phi$, we have that $\angle pq'q > \frac{\pi}{2}$. Therefore, $\angle pq'q$ is the largest angle of $\triangle pq'q$, meaning that its opposite side, $pq$, is the largest side. In particular, we have $|pq| > |pq'| > |AC|$. The case where $|pB| > |qB|$ is identical, which completes the proof of the lemma. $\square$

**Proposition 4.8** *The worst case performance ratio of Algorithm 1 is at least five.*

*Proof:* For $k = 1$ and $n = 6$, place five nodes equally spaced on the boundary of a circle of radius 1, and place one other node in the center of that circle. Since the circle has radius 1, the center node shares an edge with all the other nodes. Also, since the distance between every pair of nodes on the circle is at least $2\sin\frac{\pi}{5} > 1$, there is no other edge in the unit disk graph. In the remainder of the proof, this basic structure will be referred to as a *star*, the node placed in the center of the circle will be referred to as *the center* of the star and the five nodes on the boundary of the circle will be referred to as the *branches* of the star. The center of a star form a dominating set of the whole star. However, if the center happens to be given a higher identifier than one of the branches, all branches would be marked as dominators, leading to a performance ratio of five.

Figure 4 depicts how to connect several stars to build cases with $n$ as large as desired. More precisely, we show how to construct examples of size $14 + 8m$, for any given $m$ (on Figure 4, $m = 1$). The construction goes as follows: place $m + 2$ stars on a horizontal line such that their centers are placed at $x$-coordinates $0, 3, 6, \ldots, 3(m + 1)$ and no branch lies on the horizontal line. Since the circles in which the stars are inscribed are at distance at least 1 from each other, the only edges of the graph so far are the ones linking the branches of the stars to their centers. All that remains is to connect the graph. In order to do so, add nodes on the intersection of the inscribing circles with the horizontal line. These nodes will be referred to as *bridging nodes*. Since the centers of two consecutive stars are at distance 3 from each other, the two bridging nodes between them are at distance 1 from each other. Therefore, there is an edge between two bridging nodes which are between the centers of two consecutive stars, making the whole network connected. To see how the performance ratio of five can be reached, notice that the star centers form a dominating set of size $m + 2$. However, since the set of all branches form an independent set, it could be that those nodes would be marked as dominators, leading to a dominating set of size $5(m + 2)$, which gives a performance ratio of at least five.

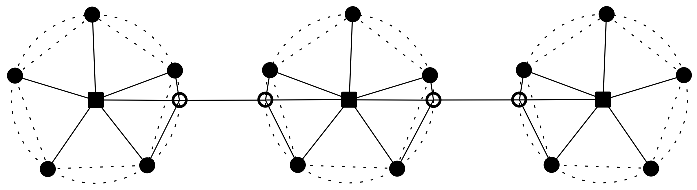

Figure 4: Lower bound of five for Algorithm 1.

For $k > 1$, Figure 5 shows how to generalize the star structure. The goal is to map to each node of the star a set of $k$ nodes such that:

1. nodes mapped to the center share an edge with every other node and

2. nodes mapped to the branches only share edges with nodes mapped to the center and nodes mapped to the same branch.

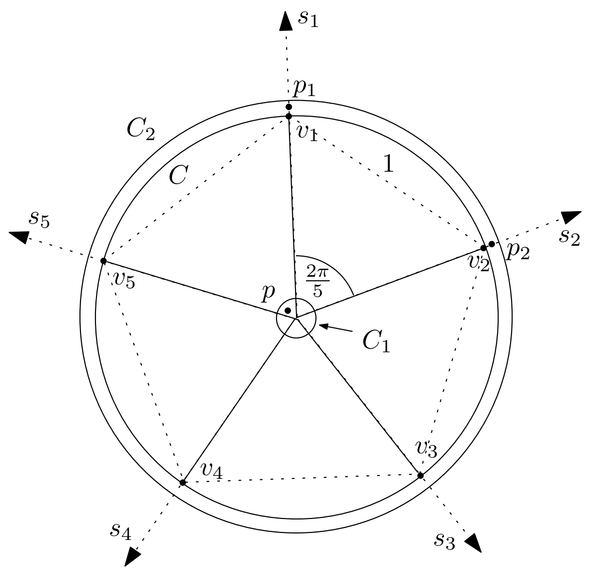

Figure 5: Widget for $k > 1$.

In order to achieve this, draw a regular pentagon having side length of 1. Let $C$ be the inscribing circle of that pentagon and $r = \frac{1}{2\sin(\frac{\pi}{5})}$ be the radius of $C$. Now, let $C_1$ and $C_2$ respectively be the circles having the same center as $C$ and radii $r_1 = \frac{1-r}{2}$ and $r_2 = r + r_1$. For each vertex $v_i$ of the pentagon ($i$ from 1 to 5), let $s_i$ be the half-line from the center of $C$ through $v_i$. Now, let $p$ be a point located inside $C_1$, and $p_1$ and $p_2$ be two points located on some $s_i$ and $s_j$ ($i \neq j$), between $C_2$ and $C$. Then, Lemma 4.7 tells us that

$$|p_1, p_2| > |v_1, v_2| \geq 1$$

and from the triangle inequality, we have

$$|p, p_1| \leq r_1 + r_2 = 2(\frac{1-r}{2}) + r = 1.$$

Similarly, $|p, p_2| \leq 1$. The construction we need is then the following: place $k$ points inside $C_1$ and $k$ points on each of the $s_i$ between $C$ and $C_2$. We call the result of that construction a *generalized star*. The points located inside $C_1$ form a $k$-dominating set, but the algorithm may mark all nodes located on the $s_i$. Since there are $5k$ such points, the performance ratio is five in that case. To construct a lower bound example with $k > 1$ for large $n$, we link the generalized stars in a similar fashion as for the case $k = 1$. □

## 5 Comparison with Kuhn et al.'s algorithm

To the best of our knowledge, Kuhn *et al.* [15] proposed the only algorithm producing a $k$-dominating set with probability 1. However, their performance ratio is not deterministic. In this section, we compare the two algorithms. The most important result of [15] about the algorithm they proposed for unit disk graphs is Lemma 5.6, which states that in a disk of radius $\frac{1}{2}$, their algorithm elects at most $O(k)$ leaders in expected case. In the deterministic case, our algorithm elects at most exactly $k$ leaders.

**Proposition 5.1** *Let $G = (V, E)$ be a unit disk graph, $C$ a disk of radius $\frac{1}{2}$ and $S \subseteq V$ the set of nodes marked by Algorithm 1. Then $|S \cap C| \leq k$.*

*Proof:* The proof goes the same way as for proposition 4.4. By proposition 4.3, $S$ is the union of $k$ independent sets. Since the maximum number of independent nodes a unit disk can contain is 1, $S \cap C$ can not contain more than $k$ nodes. $\square$

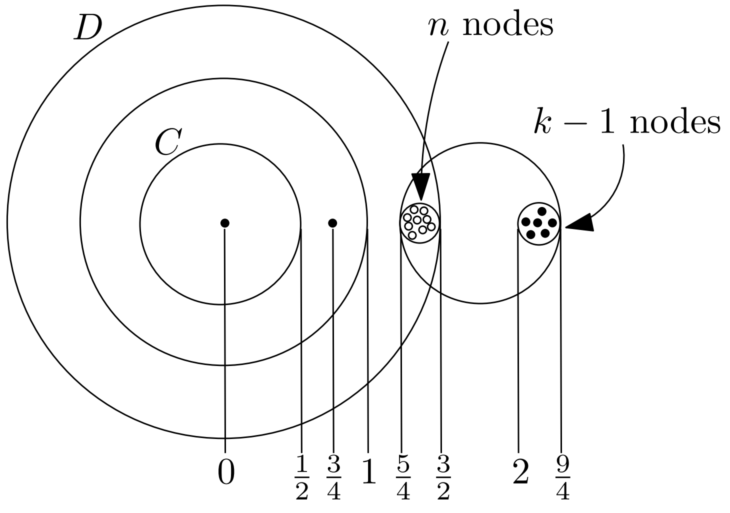

Figure 6: Counter-example to the proof of Theorem 5.7 of [15].

Kuhn *et al.* [15] actually showed that the performance ratio of the proposed algorithm is $O(1)$. However, their proof uses the fact that, in order to $k$-dominate the nodes in a disk $C$ of radius $\frac{1}{2}$, every optimal $k$-dominating set must elect at least $k$ dominators in a disk of radius $\frac{3}{2}$ having the same center as $C$. However, this is not always the case. Figure 6 shows how to construct a counter-example for any $k \geq 2$ (on the figure, $k = 7$). Black nodes are the dominators and white nodes are the non-dominators. The node at $\frac{3}{4}$ is only there to make the network connected. The only node in $C$ is 7-dominated but only two of the $n + 2$ nodes in $D$ are selected as dominators.

## 6 Simulation Results

We have generalized an existing independent set-based algorithm [2, 19, 25] in order to incrementally construct a $k$-dominating set. We have chosen to generalize this specific algorithm because it is local and has constant performance ratio. By simulation, we compare our algorithm with $k$-generalized versions of other available algorithms. Stojmenovic *et al.* [24] suggested the following heuristic to improve the independent set algorithm of [2, 19, 25]: instead of ordering the nodes according to their identifier, order them according to their degree first and then their identifier. Higher priority is granted to nodes having higher degree. The performance ratio is still at most five, but the case of Figure 4 is avoided. However, it has not been proven that the performance ratio is actually improved. For the $k$-dominating set problem, it is not desirable to favor higher degree nodes. The reason is that nodes having degree less than $k$ cannot have $k$ dominating neighbors, so they must necessarily be in the $k$-dominating set.

Selecting nodes of higher degree is the same idea that is behind the greedy set-cover algorithm. The greedy set-cover algorithm first favors nodes that dominate the largest number of nodes not yet dominated. Although this is a global selection criterion, it still has to be examined. At first sight, since it does not have constant performance ratio (we said in the first paragraph of Section 2 that it is $H(\Delta)$, where $\Delta$ is the maximum degree of a node in the network and $H$ is the harmonic function), one would believe that it would not perform as well as our algorithm. However, it turns out that in order to have $H(\Delta) > 5$, we need $\Delta$ to be at least 83, and to reach six, we need $\Delta$ to be at least 226. Since it is not likely to have nodes having that many neighbors in real situations, this algorithm still deserves attention.

In this section, we discuss simulation results comparing Algorithm 1 with $k$-generalized versions of both the algorithm presented in [24] and the greedy algorithm. We also compare it with the greedy construction of a maximal

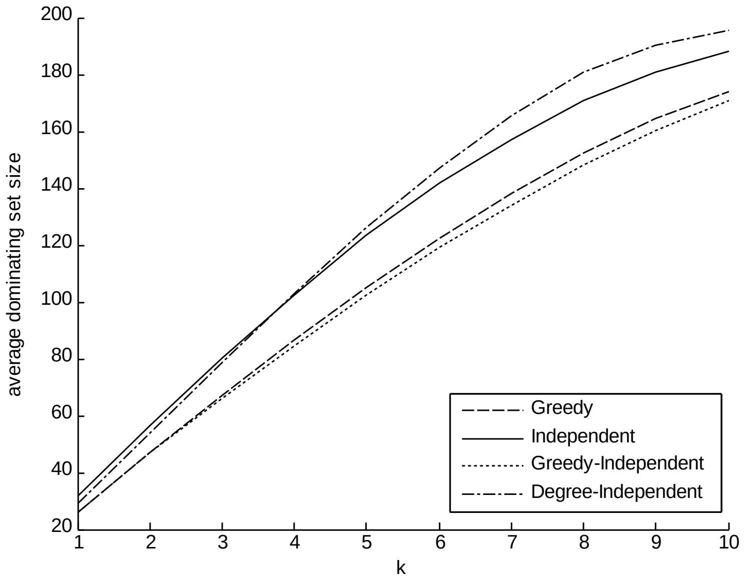

Figure 7: Average dominating set size for 200 nodes.

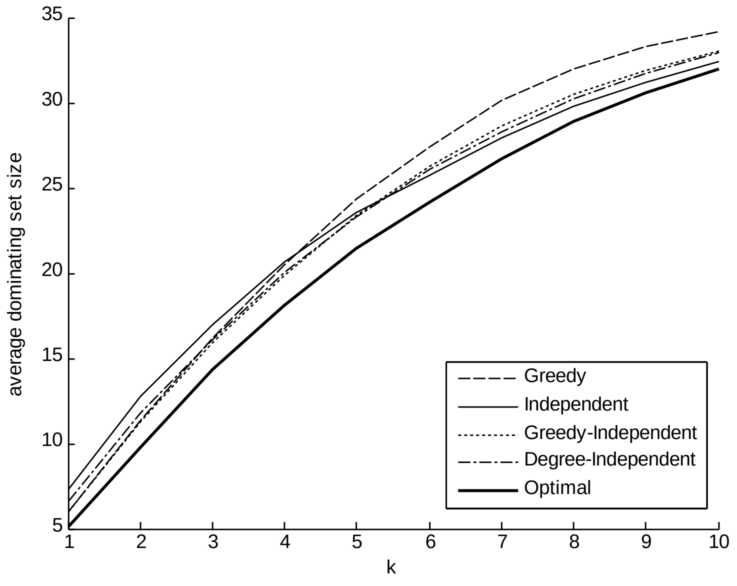

Figure 8: Average dominating set size for 35 nodes.

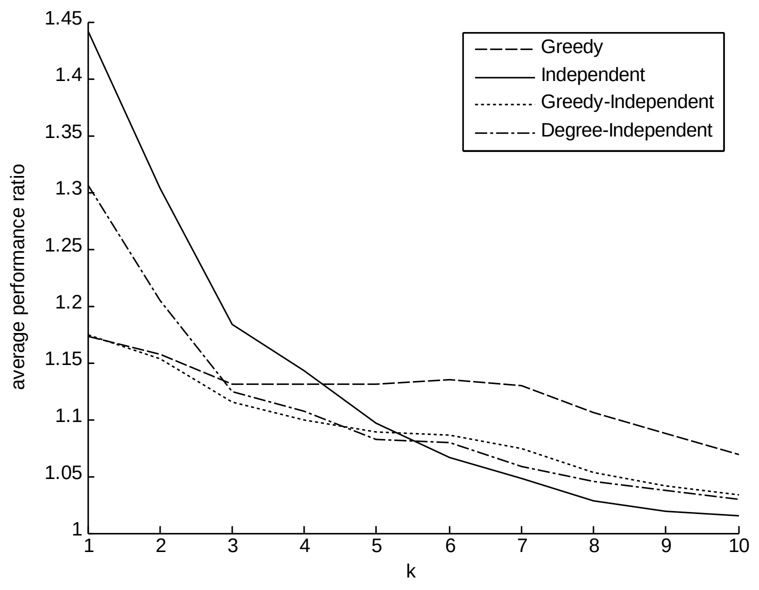

Figure 9: Average performance ratio for 35 nodes.

| alg / $k$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| greedy | 26.03 | 47.17 | 67.3 | 86.69 | 105.01 | 122.39 | 138.29 | 152.32 | 164.32 | 174.21 |
| ind | 31.72 | 54.49 | 73.36 | 90.1 | 104.46 | 117.87 | 130.27 | 141.59 | 151.92 | 161.33 |
| gr-ind | 26.07 | 47.17 | 66.5 | 85.03 | 101.54 | 117.09 | 131.34 | 143.91 | 154.76 | 164.16 |
| deg-ind | 29.26 | 50.82 | 69.49 | 86.46 | 101.76 | 116.43 | 129.82 | 141.73 | 152.85 | 162.62 |

Table 1: Simulation results for 200 nodes.

independent set. The $k$-generalized versions of those algorithms work the same way we generalized the maximal independent set algorithm: for $k = 1$, we run the standard algorithm on all nodes. For $k \geq 2$, we run the standard algorithm on nodes that are not yet $k$-dominated. We ran our simulations 200 times for networks of 200 nodes. We have chosen a communication range such that with high probability, the network is connected. According to Penrose [22, 23], for any integer $k \geq 0$ and real constant $c$, if the nodes have identical radius $r$ given by the formula:

$$r = \sqrt{\frac{\ln n + k \ln \ln n + \ln(k!) + c}{n\pi}}$$

then the network is $(k + 1)$-connected with probability $e^{-e^{-c}}$ as $n$ goes to

| alg / $k$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| greedy | 5.98 | 11.35 | 16.19 | 20.46 | 24.3 | 27.43 | 30.13 | 31.99 | 33.3 | 34.17 |
| ind | 7.35 | 12.78 | 16.93 | 20.67 | 23.57 | 25.78 | 27.95 | 29.76 | 31.19 | 32.45 |
| gr-ind | 5.99 | 11.31 | 15.95 | 19.89 | 23.4 | 26.25 | 28.68 | 30.46 | 31.87 | 33.03 |
| deg-ind | 6.66 | 11.82 | 16.09 | 20.04 | 23.26 | 26.09 | 28.25 | 30.26 | 31.74 | 32.9 |
| optimal | 5.1 | 9.81 | 14.31 | 18.09 | 21.48 | 24.18 | 26.68 | 28.93 | 30.61 | 31.96 |

Table 2: Simulation results for 35 nodes.

infinity. For $n = 200$, choosing $k = 1$ and $c = 5$, we then obtain that for a radius of $r \approx 0.138$, the network is 2-connected with probability 0.99.

Figure 7 and Table 1 show the simulation results we have obtained. The algorithm which performed the best is the one in which we greedily constructed an independent set. Not far behind is the greedy algorithm. It worth noting that even if those algorithms perform slightly better, neither of the two are local. This is because the greedy choice of the next node to be marked is based on a global criteria. For the two local algorithms, it is interesting to note that the one using the ordered pair degree-id only performs better for small values of $k$ (5 and less). After that, it is the one simply based on identifiers which performs better. With a 95% certainty, the expected values of the size of the dominating sets was at most $\pm 0.67$ node.

Unfortunately, Figure 7 and Table 1 do not show the optimal solution. Since the dominating set problem is NP-complete, only exponential time algorithms are known to solve the problem. This is why only small instances of the problem can be addressed by simulation. Figure 8 and Table 2 compare the same algorithms with the optimal solution for a network of 35 nodes. We ran over 200 simulation cases. In that case, with a 95% certainty, the actual expected values of the dominating sets size was at most $\pm 0.28$ nodes. Figure 9 shows the average performance ratio we obtained for each algorithm. For small values of $k$, the two global greedy algorithms are the best, followed by the local algorithm granting priority to high degree nodes. The algorithm simply based on identifiers performs the worst. However, as $k$ grows, the result change completely. The algorithm simply based on identifiers becomes the best, and the basic greedy algorithm becomes the worst. The algorithm constructing independent sets by granting priority to high degree nodes performs slightly better than the greedy construction of an independent set.

## 7 Conclusion

In this paper, we have introduced a new algorithm to address the $k$-dominating set problem. Our algorithm has a deterministic performance ratio of six. The previously best algorithm had an expected performance ratio of $O(k)$ for an unspecified constant [15]. We have shown that the size of the $k$-dominating set our algorithm produces may be five times larger than the optimal one. However, it is an open issue whether or not the gap between five and six can be closed. The expected performance ratio is also unknown.

Simulation results have shown that in some cases, the $k$-generalized version of the greedy dominating set algorithm performs better than ours. Besides their worst-case performance ratio, an other important difference between the greedy dominating set algorithm and ours is that one is global while the other is local. We believe that differences between the performance of local and global algorithms is an interesting research avenue. Important work in that field has been done by Kuhn [14].

## Acknowledgment

The authors graciously acknowledge the financial support received from the following organizations: Natural Sciences and Engineering Research Council of Canada (NSERC) and Mathematics of Information Technology and Complex Systems (MITACS).

## References

[1] J. Alber, M. R. Fellows, and R. Niedermeier, Polynomial-time data reduction for dominating set. *J. ACM*, **51(3)**:363–384, 2004.

[2] K. M. Alzoubi, P.-J. Wan, and O. Frieder, Message-optimal connected dominating sets in mobile ad hoc networks. In *MobiHoc '02: Proceedings of the 3rd ACM international symposium on Mobile ad hoc networking & computing*, pp. 157–164, ACM Press, New York, NY, USA, 2002.

[3] M. Cardei, X. Cheng, X. Cheng, and D.-Z. Du, Connected domination in multihop ad hoc wireless networks. In *Proc. the 6th Interna-*

*tional Conference on Computer Science and Informatics (CS&I'2002)*, Durham, NC, USA, 2002.

[4] B. N. Clark, C. J. Colbourn, and D. S. Johnson, Unit disk graphs. *Discrete Math.*, **86(1-3)**:165–177, 1990.

[5] F. Dai and J. Wu, Distributed dominant pruning in ad hoc networks. In *Proc. Intl. Conf. on Communications (ICC)*, Anchorage, AK, 2003.

[6] F. Dai and J. Wu, On constructing k-connected k-dominating set in wireless networks. In *IPDPS*, IEEE Computer Society, 2005.

[7] B. Das and V. Bharghavan, Routing in ad-hoc networks using minimum connected dominating sets. In *ICC (1)*, pp. 376–380, 1997.

[8] U. Feige, M. M. Halldorsson, G. Kortsarz, and A. Srinivasan, Approximating the domatic number. *SIAM J. Comput.*, **32(1)**:172–195, 2003.

[9] J. Gao, L. Guibas, J. Hershberger, L. Zhang, and A. Zhu, Discrete mobile centers. In *SCG '01: Proceedings of the seventeenth annual symposium on Computational geometry*, pp. 188–196, ACM Press, New York, NY, USA, 2001.

[10] S. Guha and S. Khuller, Approximation algorithms for connected dominating sets. In *ESA '96: Proceedings of the Fourth Annual European Symposium on Algorithms*, pp. 179–193, Springer-Verlag, London, UK, 1996.

[11] J. C. Hansen and E. Schmutz, The expected size of the rule k dominating set. *CoRR*, **cs.DM/0408067**, 2004.

[12] D. S. Johnson, Approximation algorithms for combinatorial problems. In *STOC '73: Proceedings of the fifth annual ACM symposium on Theory of computing*, pp. 38–49, ACM Press, New York, NY, USA, 1973.

[13] R. M. Karp, Reducibility among combinatorial problems. In R. E. Miller and J. W. Thatcher, eds., *Complexity of Computer Computations*, pp. 85–103, Plenum Press, 1972.

[14] F. Kuhn, The Price of Locality: Exploring the Complexity of Distributed Coordination Primitives. In *PhD Thesis, ETH Zurich, Diss. ETH No. 16213*, 2005.

[15] F. Kuhn, T. Moscibroda, and R. Wattenhofer, Fault-Tolerant Clustering in Ad Hoc and Sensor Networks. In *26th International Conference on Distributed Computing Systems (ICDCS), Lisboa, Portugal*, 2006.

[16] S. Kutten and D. Peleg, Fast distributed construction of small k-dominating sets and applications. *J. Algorithms*, **28(1)**:40–66, 1998.

[17] Y. Li, M. T. Thai, F. Wang, C.-W. Yi, P. Wan, and D.-Z. Du, On greedy construction of connected dominating sets in wireless networks. Tech. Rep. 04-048, University of Minnesota - Computer Science and Engineering, 2004.

[18] Y. Li, M. T. Thai, F. Wang, C.-W. Yi, P. Wan, and D.-Z. Du, On greedy construction of connected dominating sets in wireless networks. In *Wireless Communications and Mobile Computing (WCMC)*, vol. 5, pp. 927–932, 2005.

[19] M. Marathe, H. Breu, H. Ravi, and D. Rosenkrantz, Simple heuristics for unit disk graphs. 1995.

[20] S. Masuyama, T. Ibaraki, and T. Hasegawa, The computational complexity of the $m$-center problems on the plane. *The Transactions of the Institute of Electronics and Communication Engineers of Japan*, **64E**:57–64, 1981.

[21] T. Moscibroda and R. Wattenhofer, Maximizing the lifetime of dominating sets. In *IPDPS '05: Proceedings of the 19th IEEE International Parallel and Distributed Processing Symposium (IPDPS'05) - Workshop 12*, p. 242.2, IEEE Computer Society, Washington, DC, USA, 2005.

[22] M. D. Penrose, The longest edge of the random minimal spanning tree. *The Annals of Applied Probability*, **7(2)**:340–361, 1997.

[23] M. D. Penrose, On k-connectivity for a geometric random graph. *Random Struct. Algorithms*, **15(2)**:145–164, 1999.

[24] I. Stojmenovic, M. Seddigh, and J. Zunic, Dominating sets and neighbor elimination-based broadcasting algorithms in wireless networks. *IEEE Trans. Parallel Distrib. Syst.*, **13(1)**:14–25, 2002.

[25] P.-J. Wan, K. M. Alzoubi, and O. Frieder, Distributed construction of connected dominating set in wireless ad hoc networks. *Mob. Netw. Appl.*, **9(2)**:141–149, 2004.

[26] J. Wu and F. Dai, A generic distributed broadcast scheme in ad hoc wireless networks. *IEEE Trans. Comput.*, **53(10)**:1343–1354, 2004.

[27] J. Wu and H. Li, On calculating connected dominating set for efficient routing in ad hoc wireless networks. In *DIALM '99: Proceedings of the 3rd international workshop on Discrete algorithms and methods for mobile computing and communications*, pp. 7–14, ACM Press, New York, NY, USA, 1999.

[28] J. Wu and H. Li, A dominating-set-based routing scheme in ad hoc wireless networks. *Telecommunication Systems*, **18(1–3)**:13–36, 2001.

[29] W. Wu, H. Du, X. Jia, Y. Li, C.-H. Huang, and D.-Z. Du, Minimum connected dominating sets and maximal independent sets in unit disk graphs. Tech. Rep. 04-047, Department of Computer Science and Engineering, University of Minnesota, 2004.