Technical Report

Department of Computer Science
and Engineering
University of Minnesota
4-192 EECS Building
200 Union Street SE
Minneapolis, MN 55455-0159 USA

TR 03-021

A Distributed Approximation for Minamum Connecting Dominated Sets in Wireless Networks

Manki Min, Raghuram Lanka, Xiao Huang, and Ding-zhu Du

May 02, 2003

A Distributed Approximation for Minimum Connected Dominating Sets in Wireless Networks

Manki Min* Raghuram Lanka* Xiao Huang* Ding-Zhu Du*

Abstract

A wireless ad-hoc network is a wireless, multihop network without an infrastructure. And due to the limited resources of hosts in the network, the resource utilization becomes more critical. As a result, construction of virtual backbones in wireless ad-hoc networks gets more attractive. In this paper, we present a distributed approximation scheme for minimum connected dominating sets in unit-disk graphs which can serve as a virtual backbone in the network. Our algorithm has the performance ratio of 7 and the message complexity of $O(n \log n)$, which are the best among the schemes in the literature.

*Department of Computer Science and Engineering, University of Minnesota, Minneapolis, MN 55455, USA. E-mail:{min,rlanka,xiao,dzd}@cs.umn.edu

1 Introduction

A wireless ad-hoc network is a wireless, multihop network in which each node communicates with other nodes using only shared channels such as wireless radio waves. And wireless medium has, by nature, limited transmitting power which leads to the limitation of transmission distance. This property makes it feasible to model the network topology in unit-disk graphs. Since there is no infrastructure such as backbones to exploit, the resource utilization is much more inefficient than wired, infrastructured networks. Moreover, since each node has very limited resources, the resource utilization becomes a more important issue in ad-hoc networks.

A connected dominating set is a set of nodes such that every node in the set can be interconnected and every node that is not in the set is a neighbor of a node in the set. In other words, such a connected dominating set can act as a backbone in a wired, infrastructured network. Wan et al proposed a distributed algorithm to construct a minimum connected dominating set in unit-disk graphs which has the performance ratio of at most 8 and the message complexity of $O(n \log n)$ [10]. Meanwhile, Cardei et al also proposed a distributed algorithm with the performance ratio of 8 and the message complexity of $O(n\Delta)$, where Δ is the maximum degree [1]. And recent works toward a minimum connected dominating set in [6], [11], [9], [3] have non-constant performance ratios and message complexities greater than $O(n \log n)$.

In this paper, we present a new distributed algorithm to construct a minimum connected dominating set in unit-disk graphs with a performance ratio of 7 and the message complexity of $O(n \log n)$. Our algorithm uses a Steiner tree approximation to interconnect the dominating set. Chen et al proposed a 3-approximation for the problem of the Steiner tree with minimum number of Steiner points where each edge in the tree is bounded by a constant [2]. Since the maximal independent set, which is a dominating set, that can be found by Wan et al's algorithm has some interesting properties, we can use Steiner tree approach to interconnect the dominating set. Those properties will be described later in detail.

The remaining of this paper is organized as follows: Section 2 briefly describes the problem of the Steiner tree with minimum number of Steiner points in unit-disk graphs and show how we can use Chen et al's result in our algorithm. In section 3, we first describe the problem of the minimum connected dominating set in unit-disk graphs and then explain the interesting properties of the maximal independent set that can be found by Wan et al's algorithm. We present our 7-approximation scheme and its simulation results in section 4 and 5, respectively and then our conclusions in section 6.

2 ST-MNSP-in-UDG

Given a unit-disk graph and a set of vertices, called terminals, the Steiner tree with minimum number of Steiner points in unit-disk graphs problem (ST-MNSP-in-UDG) is to find a Steiner tree interconnecting all terminals with minimum number of Steiner points, while a pair of terminals can be interconnected if and only if the distance of the two terminals is at most 1.

Chen et al [2] showed that the Steiner tree problem with minimum number of Steiner points (STP-MSP) has a 3-approximation scheme. And in the process of their proof, they showed that a minimum spanning tree approach has a performance ratio close to 3. In this section, we show that ST-MNSP-in-UDG also has an approximation scheme with a performance ratio close to 3.

Lemma 1 *ST-MNSP-in-UDG is NP-hard.*

Proof. Lin and Xue [8] showed that the Steiner tree problem with minimum number of Steiner points and bounded edge-length (STP-MSPBEL) is NP-hard. STP-MSPBEL can be described as follows: Given a set of terminals in a two-dimensional Euclidean plane, and a positive constant R, STP-MSPBEL asks for a Steiner tree interconnecting all terminals with minimum number of Steiner points where every edge in the tree has a length at most R. In ST-MNSP-in-UDG, the Steiner tree interconnects all terminals with minimum number of Steiner points and every edge in the tree has a length at most 1. Hence ST-MNSP-in-UDG

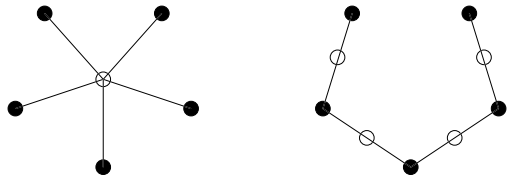

Figure 1: Steiner tree and steinerized MST

can be considered as an instance of STP-MSPBEL in which R is 1. Since STP-MSPBEL is NP-hard, ST-MNSP-in-UDG is also NP-hard. □

Moreover, we can obtain the same properties of a shortest optimal solution for ST-MNSP-in-UDG as those of a shortest optimal solution for STP-MSP.
Any shortest optimal solution for ST-MNSP-in-UDG has the following properties:
(1) No two edges cross each other.
(2) Two edges meeting at a vertex form an angle of at least 60°.
(3) If two edges form an angle of exactly 60°, then they have the same length.
As a result, there always exists a shortest optimal Steiner tree T^* for ST-MNSP-in-UDG such that every vertex in T^* has degree at most five. Since it is hard to find an optimal Steiner tree, we find a minimum spanning tree and steinerize the tree by adding Steiner points on edges longer than 1 as described in Figure 1. The resulting tree will be called a *steinerized* minimum spanning tree, SMST in short. Chen et al [2] showed that an SMST is an approximation for STP-MSP with a performance ratio close to 3.

Lemma 2 *Let T^* be a shortest optimal solution for ST-MNSP-in-UDG and T be an SMST. And let S^* and S be the number of Steiner points in T^* and T, respectively. Then $|S| \leq 3|S^*| + 1$.*

Proof. Chen et al proved the above Lemma for the problem STP-MSP. Since the problem ST-MNSP-in-UDG is an instance of the problem STP-MSP where the bound of every edge is 1 instead of an arbitrary constant value, ST-MNSP-in-UDG also preserves the above

property. □

Note that it is merely a minimum spanning tree with Steiner points on long edges that the above approximation scheme finds. In our approximation scheme, the given points as an input to ST-MNSP-in-UDG has some important properties which will be described in next section.

3 CDS-in- DG

Given a unit-disk graph and a set of vertices, the minimum connected dominating set problem, MCDS in short, is to find a subset of vertices with a minimum cardinality to dominate all vertices in the unit-disk graph.

It has been known that MCDS is NP-hard in unit-disk graphs [5]. Currently, implemented best known approximation for MCDS in unit-disk graphs has performance ratio of 8 [10] [1]. Wan et al [10] presented a distributed construction of a connected dominating set in wireless ad hoc networks. In their algorithm, they first find a maximal independent set R and then interconnect the maximal independent set. They showed that R has an interesting property. The below lemma from [10] describes the property.

Lemma 3 *Any pair of complementary subsets of R are separated by exactly two hops.*

By definition, any two nodes in a maximal independent set are separated by at least two hops. Wan et al showed that we can always find a two-hop path between any pair of complementary subsets of R. Now consider a minimum spanning tree T for R. Since R is an independent set, any edge in T must have a distance greater than 1. In terms of SMST, this means that any edge in T must have at least one Steiner points on it. A spanning tree for R constructed using only those two-hop paths as edges will have minimum number of Steiner points since any edge in any spanning tree for R must have a distance at least 1. This implies that every edge of a minimum spanning tree of R has a length between 1 and 2 and hence every edge has exactly one Steiner point on it.

And using the above property, we can compute the upper bound for $|R|$ as in the following Lemma.

Lemma 4 *Let C^* be an optimal solution for MCDS in a unit-disk graph. Then $|R| \leq 4|C^*| - 1$.*

Proof. Figure 2 illustrates the relationship between R and C^*. White nodes and black nodes denote nodes in C^* and R, respectively. In (b), (c) and (d), threaded lines and dashed lines denote distances less than or equal to one and greater than one, respectively.
Figure 2 (a) shows that any pair of black nodes must have an angle greater than 60° with their white common neighbor. Note that $\frac{a}{\sin\alpha} = \frac{b}{\sin\beta} = \frac{c}{\sin\gamma}$. Consider a triangle consisting of two black nodes and one white node at each three angle point, as in Figure 2 (a). Suppose $\alpha \leq 60^\circ$ and $b \geq c$. Since $\alpha+\beta+\gamma = 180^\circ$, $120^\circ \leq \beta+\gamma < 180^\circ$. Since $b \geq c$, $\sin\beta \geq \sin\gamma$, so $\gamma \leq \beta \leq 180^\circ - \gamma$. Now we have $\gamma \leq \beta$ and $\beta < 180^\circ - \gamma$. Since $120^\circ \leq \beta+\gamma \leq 2\beta$, $\beta \geq 60^\circ$. Now we have two cases: (i) $\beta \leq 90^\circ$. In this case, clearly $\sin\alpha \leq \sin\beta$. Hence $a \leq b$. (ii) $\beta > 90^\circ$. In this case, $\beta < \beta+\gamma = 180^\circ - \alpha$. Hence we get $a < b$. Therefore, we can conclude that if $\alpha \leq 60^\circ$, then $a \leq b$.
Now consider a single white node and two connected white nodes. As in Figure 2 (b) and (c), if we divide the whole neighborhood of a single white node and two connected white nodes equally by 60°, we get 6 and 8 black neighbors, respectively. As a result, we can conclude that a single white node can have at most 5 black neighbors and two connected white nodes can have at most 7 black neighbors.
Now we compute the upper bound for $|R|$. Since C^* is a dominating set, every node in R must have a neighbor in C^*. Since C^* is connected, any node in C^* has at least one neighbor in C^*, as in Figure 2 (d). v can have at most 4 black neighbors which are adjacent only to v and not to u. Now consider any preorder traversal of an arbitrary spanning tree for C^*. Let $c_1, \ldots, c_k$ be the traversal of C^*. Partition R into $R_1, \ldots, R_k$, where $R_i = \{v \in R : d(c_i, v) \leq 1 \text{ and } d(c_j, v) > 1, j < i\}$. Since c_1 and c_2 are connected, $|R_1| + |R_2| \leq 7$. For any c_k, $k \geq 3$, $|R_k| \leq 4$. Therefore $|R| = \sum_{i=1}^{k} |R_i| \leq 7 + 4(k-2) = 4k - 1$. □

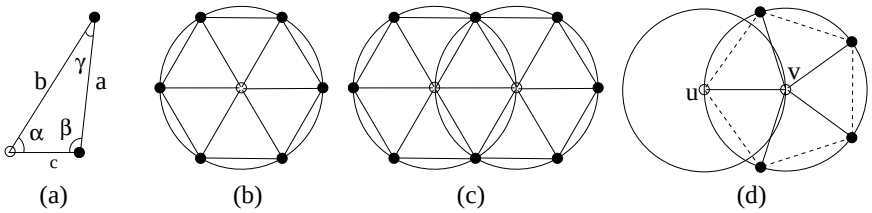

Figure 2: R and C^*

In this paper, as an application of Lemma 2, we show a polynomial-time 7-approximation and compare it with other implemented approximations by computational experiments.

4 Approximation scheme

The basic idea of our approximation is to combine the two approximations from [2] and [10]. First, using the MIS construction in [10], find a maximal independent set R. Note that every maximal independent set is a dominating set. Next, in order to interconnect R, we find an SMST T for R. Since R is a dominating set, T contains a dominating set and T is connected. Up to now, T is not a connected dominating set, but it has some important properties.

Lemma 5 *Let T^* be an optimal solution for ST-MNSP-in-UDG on input R and S^* the set of Steiner points in T^*. Let C^* be an optimal solution for MCDS on the given terminals. Then*

$$|S^*| \leq |C^*|$$

Proof. Since C^* is a dominating set, every node in R must be either in C^* or a neighbor of a node in C^*. This implies that the tree $T_{R\cup C^*}$ spanning $R\cup C^*$ is a Steiner tree for R and all the Steiner points lie in C^*. If a node v in R is in C^*, then v cannot be a Steiner point in the tree $T_{R\cup C^*}$. Hence the number of Steiner points in $T_{R\cup C^*}$ is at most $|C^*| - |R \cap C^*|$. Since T^* is an optimal solution, $|S^*| \leq |C^*| - |R \cap C^*|$. Therefore $|S^*| \leq |C^*|$. □

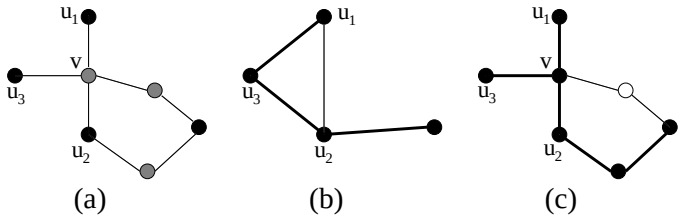

Figure 3: SMST and CDS

4.1 Transformation of SMST into CDS

As said in the beginning of this section, what we found is an SMST T for the maximal independent set R while what we want to find is a connected dominating set for the given terminals. We can transform T into a connected dominating set as follows: Lemma 3 implies that in T, every edge has exactly one Steiner point on it and that for each edge, there must be a node which interconnects both ends of the edge. So we can always find the set S' of nodes such that $|S'|$ is less than or equal to the number of Steiner points in T and the graph induced by $R \cup S'$ is connected. Next, we describe how to find such a set S'.

Let G be the given graph. Let V_G be the set of gray nodes which have at least two black neighbors. Now consider two graphs G_1 and G_2. G_1 is a subgraph of G induced by $R \cup V_G$ and $G_2 = \{R, E_2\}$ is a graph s.t. $(a, b) \in E_2 \iff a$ and b have a 2-hop paths in G. Then to find a Steiner minimum tree for R from G_1, in which Steiner points are chosen from V_G, is equivalent to find an SMST for R from G_2. This is illustrated in Figure 3. In this figure, (b) represents an SMST and (c) repretents a Steiner minimum tree. And the Steiner minimum tree for R makes a CDS for G. Note that the number of nodes in S' is at most the number of edges in an SMST. In our procedure, we use a fragment-incremental approach similar to the procedure to find a minimum-weight spanning tree in [7]. The procedure in [7] is carried on by each black node, but our procedure is carried on by each gray node which represents a 2-hop path between two black nodes.

Based on the relationship between an SMST and an CDS, we form a CDS by merging black fragments, which are individual black nodes initially, with a selection of a gray node.

And a black node selects a gray node based on the following rules: 1. A black node selects a gray neighbor which will form the largest fragment. 2. A gray node which will form a loop in the SMST should not be selected. The first rule is to find a minimum number of gray nodes. The second rule guarantees that as long as the resulting graph is connected, it forms a tree in the SMST topology. To enforce the second rule, if the current fragment of a black node already has all the black neighbors of a gray neighbor, then the gray neighbor is marked as white and does not participate in the procedure any more. And our procedure will finally output a single fragment. Suppose the output forms more than one fragment, say F_1 and F_2. By Lemma 3, initially there should be v_1, v_2 in F_1, F_2, respectively, such that v_1 and v_2 has a common gray neighbor v_3. Since F_1 and F_2 are not merged, v_3 mush be marked as white during the procedure. But in order to mark a gray node as white, all the black neighbors of the gray node must be already in a fragment of v_1 or v_2, say F_1, which implies that v_2 is already in F_2. As a result, F_1 and F_2 are actually a single fragment.

4.2 Local optimization

During the procedure, some black nodes in R can be removed without breaking the connectivity or the dominating set property of the resulting graph. We can reduce the number of dominators by removing those redundant black nodes. Note that if a black node is redundant during the procedure, it will also be redundant after the procedure. Here, we concentrate on identifying redundant black nodes in R after the procedure for interconnecting R. Hereafter we use **black** *nodes* to denote nodes in R.

After interconnecting R, some gray nodes turn into black and others turn into white and every **black** node is in the same fragment. If a gray node turns into black, then it means that the node is necessary to connect **black** nodes. In other words, white nodes may not connect **black** nodes and must be dominated by some black node. Hence, in a black node's point of view, its **black** neighbor v is redundant if it's connected to every other black neighbor of v. In this case, v's role in CDS is just for dominating, refer to Figure 4 (a). Also, in a white node's point of view, its **black** neighbor v is redundant if it can be

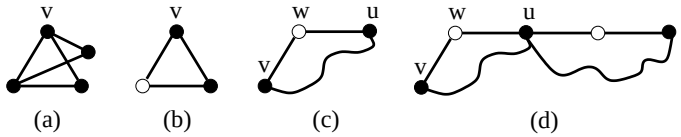

Figure 4: Redundancy of a black node v

covered by another black node, as in (b). The above will be the initial filtering of trivially redundant **black** nodes. Among the remaining **black** nodes, some may not contribute to the graph's connectivity. Now we can further identify redundant **black** nodes from those **black** nodes by looking at dominating set property. Define *loop-forming neighbors* as white neighbors of a **black** node v which can be covered by another **black** node u, as in Figure (c). Note that every other white neighbor will be covered by a black node anyway. For every pair (v, u) of **black** nodes, there are always three possible cases: 1) {loop-forming neighbors of v} $\subset$ {loop-forming neighbors of u} as in (d), or 2) {loop-forming neighbors of v} = {loop-forming neighbors of u} as in (c), or 3) none of the above.

For the case 1), we can remove v safely since every neighbor of v will be dominated by another black node. For the case 2), we can remove only one of v or u since every loop-forming neighbor of v or u needs at least one of v or u to cover it. And for the case 3), we don't remove v nor u since removing any of v or u may break the dominating set. We will deploy this local optimization scheme in our algorithm which will be described in the next subsection.

4.3 Algorithm

In this subsection, our algorithm for interconnecting **black** nodes will be described. As stated earlier, we distinguish **black** nodes and black nodes, but initially, **black** nodes and black nodes are the same. Our algorithm consists of five phases: initial phase, gray phase, black phase, update phase, final phase.

Figure 5 (a) and (b) are state diagrams for gray and **black** nodes, respectively. In gray phase, only gray nodes multicast request messages to its **black** neighbors. In black phase,

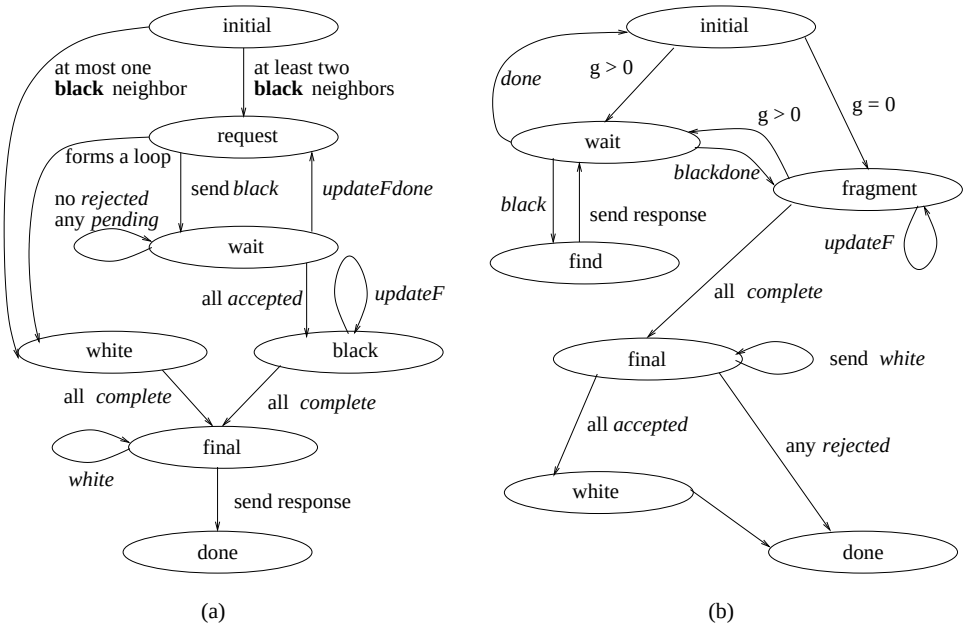

Figure 5: State diagram of gray and **black** nodes

only **black** nodes send response messages to its gray neighbors. In update phase, **black** nodes who receives a *blackdone* message initiate the update of its fragment and messages are sent through black nodes to other members in its fragment.

Every node maintains local variables fid, $flevel$. Every **black** node maintains local variables g. fid and $flevel$ of a node denotes the representative node's id and level of its fragment, which is the set of **black** nodes that are connected. $flevel$ are updated in the same way as in [7]. g of a **black** node counts the number of its gray neighbors. Initially, every **black** node has itself as its fid and 0 as its $flevel$. And every gray has null fid and 0 as its $flevel$.

Initial phase:

Every node broadcasts its color and its **black** neighbor list to its neighbors. Note that every gray node has at most 5 **black** neighbors. If a gray node has at most one **black** neighbor, it marks itself as white and multicasts to its **black** neighbors a *done* message. In the following gray and black phases, the communication happens only between a gray node

and its **black** neighbors. In the update phase, the communication happens only between a black node and its **black** neighbors.

Gray, black and update phase:

Upon receiving a *done* message, the **black** node decreases g by 1. If a gray node has at least one **black** neighbor that has a different fid or $flevel$ from other **black** neighbors' fid and $flevel$, it broadcasts a *black* message including its fid and $flevel$. Otherwise, it marks itself as white and broadcasts a *done* message. A **black** node waits until it gets all messages, either *black*, *done*, or *blackdone* from its gray neighbors and decides which gray node should be marked as black as follows. Among the gray neighbors who sent a *black* message, the one which has the largest $flevel$ will be selected. If two grey neighbors have the same $flevel$, then select the one with more **black** neighbors. If tie happens, the one with a higher degree wins and if still tie happens, the one with a higher id wins. The **black** node sends back to the selected gray neighbor an *accepted* message. And the **black** node sends back *pending* messages to the remaining gray neighbors. If a **black** node receives *blackdone*, it initiates the update phase and multicasts *updateF* message including $flevel$ and fid to its black or **black** neighbors. Whenever a node receives *updateF*, it updates its $flevel$ and fid and relay the message to its black or **black** neighbors. When a node does not have neighbor to relay the message, it multicasts *updateFdone* and this message is sent back to the initiator. A gray node waits until it gets all responses from all of its **black** neighbors. If all received messages are *accepted*, the gray node marks itself as black and updates its $flevel$ and fid and broadcasts a *blackdone* message which including its $flevel$ and fid. If the gray node receives at least one *rejected* message, it marks itself as white and broadcasts a *done* message. Otherwise, the gray node waits for *updateFdone* messages from all of its **black** neighbors. After that, the gray node broadcasts *black* again. If a **black** node receives *blackdone* message from its gray neighbor, it updates its $flevel$ and fid to those of the gray neighbor and decreses g by 1. Whenever a **black** node has zero as g and receives *updateFdone*, it broadcasts *completed*. The gray, black and update pahses ends when g of every **black** node reaches zero and every update of fragment is done. After the gray,black

and update phase, every node participates in the final phase.

Final phase:

This phase is for local optimization. **black** node broadcasts a *white* message including its black neighbor list. After this broadcast, every white node knows whether it is a loop-forming neighbor of its **black** neighbor or not. Each white node sends to its **black** neighbor a *LFN* message saying if it's a loop-forming neighbor of the **black** neighbor or not. Now every **black** node maintains a loop-forming neighbor list and multicasts to white neighbors a *white* message including the list. White nodes and black nodes select redundant **black** nodes based on the property in subsection 4.2. After the selection, the nodes sends *accepted* to the selected **black** nodes and *rejected* to the rest. If a **black** node receives all *accepted* or is a leaf node, it marks itself as white and broadcasts *finalized*. Otherwise, it only broadcasts *finalized*.

Lemma 6 *The above procedure has a message complexity of $O(n \log n)$, where n is the number of hosts in the network.*

Proof. We use notations V, V_B to denote the set of hosts in the networks and the set of **black** hosts in the networks. Note that any gray node at any step has at most five **black** neighbors. In the initial phase, every node will broadcast its color and some gray nodes may send *done* messages to its **black** neighbors, so there will be at most $6|V| = O(n)$ messages. Messages in the gray, black and update phase can be counted as follows: We can use the similar argument as in [7] since we followed their scheme to construct a minimum spanning tree. The only difference is that we do not use test messages which can count up to $2E$, where E is the number of edges, and instead we use $O(n)$ messages between **black** nodes and gray nodes to decide whether a gray node will be rejected or not. And each white node occurs only when fragment increments. Since the fragment level can be up to $\log n$, the number of messages used for rejection will be $O(n \log n)$. So, the total number of messages in those phases will be $O(n \log n)$. In the final phase, every **black** nodes broadcast a message and its neighbors send back messages, so there will be at most $|V_B| = O(n)$ messages.

Therefore, in total, the message complexity will be $O(n \log n)$. □

Now we are ready to present our 7-approximation algorithm for MCDS. We first elect a leader using the algorithm in [4]. Then construct an MIS using the algorithm in [10]. And construct a CDS interconnecting the MIS using the above procedure.

Theorem 1 *The set of black nodes formed by the above algorithm is a 7-approximation for MCDS. And the message complexity is* $O(n \log n)$.

Proof. Let $V(T)$ be the set of black nodes formed by the above algorithm and S be the set of Steiner points in T and C^* be the optimal solution of MCDS for the set of given terminals. Then,

$$\begin{aligned} |V(T)| &= |R| + |S| \\ &\leq |R| + 3|S^*| + 1 \ldots \text{by Lemma 2} \\ &\leq 4|C^*| - 1 + 3|S*| + 1 \ldots \text{by Lemma 4} \\ &\leq 4|C^*| - 1 + 3|C^*| + 1 \ldots \text{by Lemma 5} \\ &= 7|C^*| \end{aligned}$$

Therefore $V(T)$ is a 7-approximation for MCDS.

Leader election requires $O(n \log n)$ messages and so does MIS construction. Also, our procedure to construct a CDS requires $O(n \log n)$ messages. Therefore, the algorithm requires $O(n \log n)$ messages. □

5 Simulation

We implemented three algorithms, Wan's [10], Cardei's [1] and ours, and compared the simulation results. In this section, Wan's algorithm, Cardei's algorithm and our algorithm will be denoted as WAN, CARDEI and MIN.

WAN and CARDEI are the best schemes with constant performance ratios in literature. Both of them have performance ratios of 8. WAN has a message complexity of $O(\log n)$ and CARDEI has a message complexity of $O(n \cdot \Delta)$, where Δ is the maximum degree.

In our simulation, N hosts are randomly generated in a $X \times Y$ square units space. Here, the graph is generated to be connected. N is chosen from 20, 50, 100 and X, which is the same as Y, is chosne from 100, 1000, 1000 and the transmission range R is chosen from 25, 50. We ran the three algorithms 100 times on different sets of parameters. Table 1 shows the simulation results.

The simulation result shows that our algorithm outperforms both WAN and CARDEI for every combination of parameters. This is because of our local optimization which effectively removes redundant black nodes from the dominating set. The results before the optimization are not included in the table, but we found that our local optimization reduces at least 15% of black nodes. Figure 6 depicts each run for the sets of parameters (N, X, R). (a) shows the runs for (20, 100, 25), (b) for (50, 1000, 50) and (c) for (100, 10000, 50).

6 Conclusions

In this paper, we presented a 7-approximation for the MCDS problem. Our performance ratio of 7 is the best among published approximations. And it has the same message complexity of $O(n \log n)$ as that in [10]. Our approximation handles the problem when every host is stable and has the same transmission range. Our future work is to study the same problem for an environment where hosts are moving and/or transmission ranges differ from hosts to hosts.

References

[1] M. Cardei, X. Cheng, X. Cheng and D.-Z. Du, Connected Domination in Multihop Ad Hoc Wireless Networks, 6^{th} *International Conference on Computer Science and Informatics (CS&I 2002)*, March 2002, North Carolina, USA

[2] D. Chen, D.-Z. Du, X.-D. Hu, G.-H. Lin, L. Wang and G. Xue, Approximations for Steiner trees with minimum number of Steiner points, *Theoretical Computer Science*, 262 (2001), 83-99.

[3] X. Cheng, X. Huang, D. Li and D.-Z.- Du, Polynomial-Time Approximation Scheme for Minimum Connected Dominating Set in Ad Hoc Wireless Networks, *Technical Report TR02-003, Computer Science and Engineering Department, Univ. of Minnesota*, 2002

[4] I. Cidon and O. Mokryn, Propagation and Leader Election in a Multihop Broadcast Environment, *Proceedings of* 12^{th} *International Symposium on DIStributed Computing (DISC98)*, 104-119.

[5] B. N. Clark, C. J. Colbourn and D. S. Johnson, Unit Disk Graphs, *Discrete Mathematics*, 86 (1990), 165-177.

[6] B. Das and V. Bharghavan, Routing in Ad-Hoc Networks Using Minimum Connected Dominating Sets, *Proceedings of International Conference on Communication(ICC97)*, 376-380

[7] R. G. Gallager, P. A. Humblet and P. N. Spiram A Distributed Algorithm for Minimum-Weight Spanning Trees, *ACM Transactions on Programming Languages and Systems*, 5 (1983), 66-77.

[8] G.-H. Lin and G. L. Xue, Steiner tree problem with minimum number of Steiner points and bounded edge-length, *Inform. Process. Lett.*, 69 (1999), 53-57.

[9] I. Stojmenovic, M. Seddigh and J. Zunic, Dominating sets and neighbor elimination based broadcasting algorithms in wireless networks, *Proceedings of IEEE Hawaii International Conference on System Sciences*, January 2001.

[10] P.-J. Wan, K. M. Alzoubi and O. Frieder, Distributed Construction of Connected Dominating Set in Wireless Ad Hoc Networks, *Proceedings of* 21^{st} *Annual Joint Conference of the IEEE Computer and Communications Societies (Infocom02)*, 3 (2002), 1597-1604.

[11] J. Wu and H. L. Li, On calculating connected dominating set for efficient routing in ad hoc wireless networks, *Proceedings of the* 3^{rd} *ACM international workshop on Discrete algorithms and methods for mobile computing and communications*, 1999, 7-14.

N	X	R	Avg. Deg.	Max. Deg.	WAN	CARDEI	MIN	vs WAN	vs CARDEI
20	100	25	6.096	9.96	8.3	7.44	5.6	67.47%	75.27%
20	100	50	9.882	15.22	5.06	4.34	2.95	58.30%	67.97%
50	100	25	10.0588	18.04	15.78	13.38	11.36	71.99%	84.90%
50	100	50	24.1028	38.57	6.12	5.29	3.8	62.09%	71.83%
100	100	25	17.0504	29.66	19.16	15.78	14.65	76.46%	92.84%
100	100	50	47.9818	78.14	6.8	5.92	4.66	68.53%	78.72%
20	1000	25	4.926	8.47	10.16	9.19	7.24	71.26%	78.78%
20	1000	50	5.173	9	9.82	9.02	7.02	71.49%	77.83%
50	1000	25	6.8388	13.14	21.82	19.24	16.3	74.70%	84.72%
50	1000	50	7.1024	13.88	21.16	19.12	15.9	75.14%	83.16%
100	1000	25	8.9088	18.72	37.9	33.08	29.63	78.18%	89.57%
100	1000	50	9.0824	18.53	36.96	31.73	28.34	76.68%	89.32%
20	10000	25	4.746	8.27	10.3	9.59	7.63	74.08%	79.56%
20	10000	50	4.809	8.55	10.08	9.22	7.23	71.73%	78.42%
50	10000	25	6.6852	13.06	22.6	19.81	16.85	74.56%	85.06%
50	10000	50	6.548	12.9	22.68	20.14	17.03	75.09%	84.56%
100	10000	25	8.536	18.1	39.12	33.84	30.2	77.20%	89.24%
100	10000	50	8.6036	18.16	38.72	33.88	30.16	77.89%	89.02%

Table 1: Comparison of results

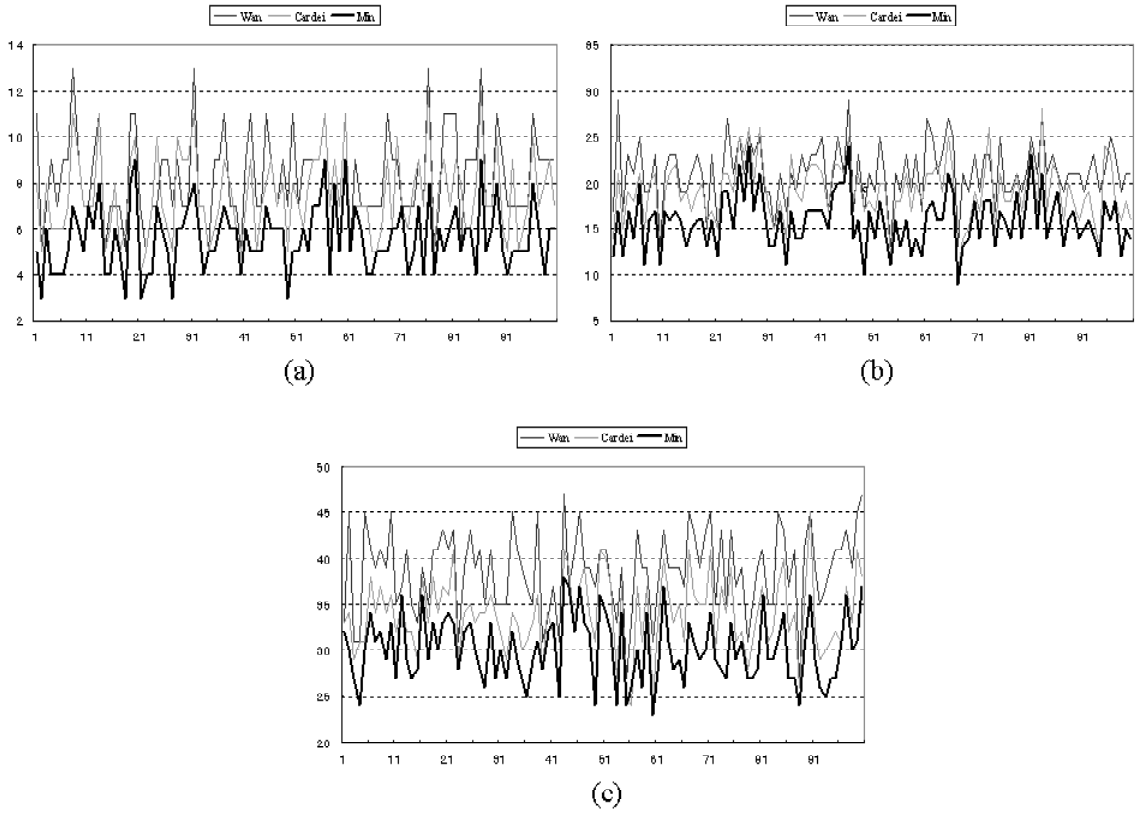

Figure 6: Simulation runs