A Dominating and Absorbent Set in a Wireless Ad-hoc Network with Different Transmission Ranges

Myung Ah Park, James Willson
Dept. of Computer Science
The University of Texas at Dallas
Richardson, TX 75083-0688 USA
mpark, jwillson@utdallas.edu

Chen Wang
Dept. of Computer Science and Technology
Tsinghua University
Beijing, China 100084
wc00@mails.thu.edu.cn

My T. Thai
Dept. of Computer Science and Engineering
University of Florida
Gainesville, FL 32611-6120 USA
mythai@cise.ufl.edu

Weili Wu, Andras Farago
Dept. of Computer Science
The University of Texas at Dallas
Richardson, TX 75083-0688 USA
weiliwu, farago@utdallas.edu

ABSTRACT

Unlike a cellular or wired network, there is no base station or network infrastructure in a wireless ad-hoc network, in which nodes communicate with each other via peer communications. In order to make routing and flooding efficient in such an infrastructureless network, Connected Dominating Set (CDS) as a virtual backbone has been extensively studied. Most of the existing studies on the CDS problem have focused on unit disk graphs, where every node in a network has the same transmission range. However, nodes may have different powers due to difference in functionalities, power control, topology control, and so on. In this case, it is desirable to model such a network as a disk graph where each node has different transmission range.

In this paper, we define Minimum Strongly Connected Dominating and Absorbent Set (MSCDAS) in a disk graph, which is the counterpart of minimum CDS in unit disk graph. We propose a constant approximation algorithm when the ratio of the maximum to the minimum in transmission range is bounded. We also present two heuristics and compare the performances of the proposed schemes through simulation.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design - *Wireless Communication*

General Terms

Algorithms, Performance

Keywords

Dominating set, absorbent set, disk graph, wireless ad-hoc network

MobiHoc'07, September 9–14, 2007, Montréal, Québec, Canada.

1. INTRODUCTION

A wireless ad-hoc network is an infrastructureless network where nodes communicate with each other via peer-to-peer communications through single hop or multihops. Due to this characteristic, wireless ad-hoc networks have been widely deployed in many application areas such as military operations, disaster relief, and environmental monitoring where it is difficult to install base stations or a physical backbone in the network.

A CDS is commonly used as the virtual backbone of a wireless ad-hoc network for efficient routing, broadcasting and collision avoidance protocols. Given an undirected graph $G = (V, E)$, a subset $V' \subseteq V$ is a CDS of G if for each node $u \in V$, u is either in V' or there exists a node $v \in V'$ such that $(u, v) \in E$ and the subgraph induced by V', i.e., $G(V')$, is connected. It is desirable to maintain the size of a CDS as small as possible since control overhead in the network can be significantly reduced. The Minimum CDS (MCDS) problem has been studied intensively in Unit Disk Graph (UDG), in which each node has the same transmission range. The MCDS problem in UDG has been shown to be NP-hard.

However, nodes in a network may have different powers due to difference in functionalities, power control to alleviate collisions, topology control to achieve a certain level of connectivity and so on. For example, in a clustered network, cluster heads or gateway nodes might have higher power than other nodes. On the other hand, in a certain power control scheme, a node enlarges or shrinks its transmission range according to a measured frequency in collisions. Likewise, in some topology-controlled networks, each node may adjust its transmission range to maintain a certain number of neighbors in order to achieve a good spatial reuse. Such an adjustment of transmission range depends on node distribution in a network.

In such cases, a wireless ad hoc network can be modeled as a disk graph(DG) rather than a UDG. The nodes in V are located in the Euclidean plane and each node $v_i \in V$ has a transmission range $r_i \in [r_{min}, r_{max}]$. There exists a directed edge $(v_i, v_j) \in E$ if and only if $d(v_i, v_j) \leq r_i$, where $d(v_i, v_j)$ denotes the Euclidean distance between v_i and v_j. Such a graph is called a *Disk Graph*. An edge (v_i, v_j) is unidirectional if $(v_i, v_j) \in E$, but $(v_j, v_i) \notin E$. An edge (v_i, v_j) is bidirectional if both (v_i, v_j) and (v_j, v_i) are in E, i.e., $d(v_i, v_j) \leq min\{r_i, r_j\}$. In other words, the edge (v_i, v_j) is bidirectional if v_i is in the disk D_j centered at v_j with radius r_j and v_j is in the disk D_i centered at v_i with radius r_i. Fig. 1

gives an example of a DG representing a network. A dotted circle indicates the transmission range of a node and a directed edge represents a unidirectional link in G, while a undirected edge represent a bidirectional link.

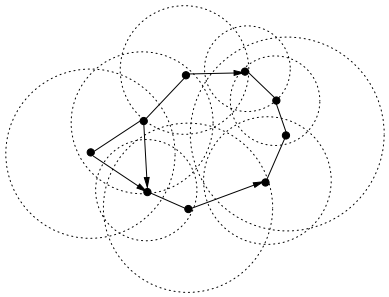

Figure 1: A disk graph representing a network

Wu and Dai [7] [1] extended the concept of dominating set in UDG to the one in DG, which is called a Dominating and Absorbent Set(DAS). Given a *directed* graph $G = (V, E)$, a subset $D \subseteq V$ is a dominating set of G if for any vertex $u \in V$, $u \in D$ or there exists $v \in D$ such that $(v, u) \in E$. A vertex in D is called a dominant, while a vertex in $V - D$ is called a dominated node. Also, a set $A \subseteq V$ is an absorbent set of G if, for any vertex $u' \in V$, $u' \in A$ or there exists $v' \in A$ such that $(u', v') \in E$. A vertex in A is called an absorbent, while a vertex in $V - A$ is called an absorbed node. A vertex set is a DAS if it is both a dominating set and an absorbent set. For a vertex u, the dominating neighbor set of u is defined as $\{v \in V | (v, u) \in E\}$, while the absorbent neighbor set of u is defined as $\{w \in V | (u, w) \in E\}$. Unlike in UDG, these two sets might be different in DG. A digraph G is *strongly connected* if, for any pair of two vertices, (u, v), there exists a directed path from u to v and from v to u as well. Throughout the paper, we assume that G is strongly connected.

Like CDS in UDG, we want to make the size of DAS as small as possible. Minimum Strongly Connected Dominating and Absorbent Set(MSCDAS) problem is defined as follows: Given a directed disk graph $G = (V, E)$, find a subset $S \subseteq V$ with minimum size, such that S is a DAS and the subgraph induced by S is strongly connected. The MSCDAS problem is NP-hard since the MCDS problem in UDG is NP-hard and UDG is a special case of DG.

In this paper, we propose a constant approximation algorithm for MSCDAS problem when the ratio of the maximum to the minimum in transmission range is bounded. We also present heuristics and evaluate the proposed schemes through simulation. The simulation results show that our algorithms produce significantly smaller size of SCDAS compared to existing algorithms in [7]. This phenomenon appears especially under dense network environments, which is the usual case in sensor networks. The remainder of this paper is organized as follows. Section 2 describes the related research work on the MSCDAS problem in (U)DGs. We discuss a constant approximation algorithm and give its theoretical analysis in Section 3. We present two heuristics and discuss the performance evaluation of the proposed schemes in Section 4 and 5, respectively. Finally, in Section 6, we conclude this paper.

2. RELATED WORK

The MCDS problem has been studied extensively in UDGs which model homogeneous wireless sensor networks. The two components of most of existing algorithms for the MCDS problem are constructing a Maximal Independent Set(MIS) and connecting the nodes in the constructed MIS using extra nodes if necessary. [6] proposed a distributed algorithm with approximation ratio of 8. The algorithm starts with an arbitrary rooted spanning tree and run a ranking process originated from a root in order to give a total ordering of nodes in a given network. Once the ranking process is done, it constructs a MIS from the root at the spanning tree by a color-marking process based on the ranking of nodes. All nodes in the MIS are colored black and others in grey after this phase. At the last phase, a dominating tree is constructed to connect all the nodes in the MIS. Forming the dominating tree starts from a grey node which is neighbored to the root at the spanning three and has the largest number of black nodes. Grey nodes joining the tree in this phase are colored blue. Eventually, the internal nodes (black or blue) of the dominating tree will become an output of the CDS. Later, Li *et al.* proposed an improved algorithm for connecting black nodes in MIS and achieved the approximation ratio of 6.8 [4]. The authors add blue nodes in a greedy fashion. More specifically, the algorithm considers the current connected components consisting of black and blue nodes at each iteration and colors in blue a grey node connecting the largest number of connected components. Thai *et al.* extended the work of [4] to undirected disk graphs by considering different transmission ranges [5].

None of the above works assumed a directed disk graph as their network models. However, nodes in a network may have different powers due to difference in functionalities, power control to alleviate collisions, topology control to achieve a certain level of connectivity and so on. In this case, a wireless ad hoc network can be modeled as a directed disk graph(DG) rather than any type of undirected graph. A node u in a given directed DG can communicate directly with node v but node v might not be able to communicate directly with node u.

In *directed* graphs, very few related works have been done. In [7], Wu presented a localized algorithm to construct a strongly connected DAS. The localized algorithm is an extension of marking process in UDG. The basic idea of the extended marking process is simple; whenever a node finds that any two neighbors have no direct edge, it marks itself as a member of DAS. In consequence of the above procedure, the resulting DAS is the set of all intermediate nodes along a shortest path between any pair of nodes. Even if the algorithm is simple, the extensive marking process does not guarantee a performance bound. In [1], the authors extended their previous work of [7]. However, the proposed algorithm has the same limitation in performance bound as their previous work.

3. O(1) APPROXIMATION ALGORITHM

In this section, we introduce the *Dominating-Absorbent Spanning Trees (DAST) Algorithm* to approximate the MSCDAS problem. We then give the theoretical analysis of its approximation ratio based on the geometric characteristics of disk graphs. Let us begin this section with notations that will be used throughout this paper.

For an arbitrary vertex $v \in V$, let $N^-(v)$ be the set of its incoming neighbors, i.e., $N^-(v) = \{u \mid (u, v) \in E\}$. Likewise, let $N^+(v)$ be the set of its outgoing neighbors, i.e., $N^+(v) = \{u \mid (v, u) \in E\}$

3.1 Algorithm Description

In the DAST algorithm, we construct an outgoing spanning tree and an incoming spanning tree rooted at an arbitrary node r from a given graph G, and then output the non-leaf nodes of the two trees as a SCDAS. More specifically, we pick an arbitrary node r as the root and construct DT and AT from the root node by calling the subroutine ConsTree. The subroutine **ConsTree** produces

a rooted outgoing spanning tree of the input graph, i.e., the root has an outgoing path to every vertex on the tree. Reversing edges of the graph and calling **ConsTree** again gives a rooted incoming tree. Thus, every pair of nodes can communicate with each other through the root. Finally, we output all the non-leaf nodes of the two trees as a SCDAS.

Algorithm 1 Dominating-Absorbent Spanning Tree(DAST)

```
1: INPUT: A directed disk graph G = (V, E)
2: OUTPUT: A strongly connected dominating and absorbent set
   S
3: arbitrarily select a node r in V
4: DT = ConsTree((V, E), r)
5: reverse all directed edges in E and form a new edge set E'
6: AT = ConsTree((V, E'), r)
7: return DT ∪ AT
8:
9: Subroutine ConsTree(V,E,r):
10: color all nodes WHITE
11: D ← {r}
12: while D contains at least one WHITE node do
13:    select any WHITE node v from D and color it BLACK
14:    for each WHITE node u in N+(v) do
15:       color u GRAY
16:       for each WHITE nodes s in N+(u) do
17:          if s ∉ D then
18:             D ← D ∪ {s}
19:             parent(s) = u
20:          end if
21:       end for
22:    end for
23:    if v ≠ r then
24:       color parent(v) BLUE
25:       D ← D − {v}
26:    end if
27: end while
28: return all BLACK and BLUE nodes
```

3.2 The Correctness of the Algorithm

In order to prove the correctness of the algorithm, we first prove two lemmas on the subroutine **ConsTree**.

LEMMA 1. *The set of BLACK nodes produced by* **ConsTree** *forms a dominating set of the input graph.*

PROOF. Assuming that the graph G is strongly connected, we first claim that no vertex remains WHITE at the end of **ConsTree**. Suppose that some vertex x remains WHITE. Then $x \notin D$ since **ConsTree** does not terminate until there is no WHITE node left in D. Because of the strong connectivity of G , there must exist a path from r to x. According to the subroutine, no vertex in the path can be marked with any color other than WHITE. Otherwise, x will not be WHITE in some following iteration. But it is easy to see that r is colored BLACK in the first iteration. The contradiction leads to the first claim. Consequently, all the nodes are categorized into three classes according to their colors; BLACK, BLUE, and GRAY. For each gray node, its black dominant is determined explicitly when the gray node has been colored. Since every blue node must be gray before it becomes BLUE, it also has a black dominant. Therefore we conclude that the set of black nodes is a dominating set of the input graph. □

LEMMA 2. *For each vertex $v \in G$, there is an outgoing path from r to v after running* **ConsTree***, which consists of BLACK or BLUE intermediate nodes and v itself.*

PROOF. A GRAY node must have an incoming edge from its BLACK dominant by Lemma 1 and *Line* 13-15. So, it is sufficient to prove that the lemma holds for each BLUE or BLACK node. We want to prove this by induction on each iteration of the WHILE loop. The basis case is trivial, in which r turns into BLACK. Assume that the lemma holds for all BLUE and BLACK nodes generated in the first k iterations. Let us consider $k+1$ iteration. A new BLUE node v_e is always generated from a GRAY node. This implies that v_e must have an incoming edge from some BLACK node v_b ,which was turned into BLACK at some iteration i where $i < k+1$. By the induction hypothesis, there exists an outgoing path from r to v_b using only BLACK and BLUE nodes as its intermediate nodes. By linking this outgoing path to an edge (v_b, v_e), we can construct an outgoing path from r to the new BLUE node v_e. On the other hand, a new BLACK node at the $k+1$ iteration is generated from a WHITE node, say, v_w. v_w must have been found by an incoming edge from a GRAY node v_g at some iteration i, where $i < k+1$ (*Line* 15 - 19). This implies that the following is true at the beginning of $k+1$ iteration; (i) the parent of v_w is v_g (ii) there exists an outgoing path from r to v_g satisfying the intermediate node condition by the induction hypothesis. Note that v_g turns into BLUE at the end of $k+1$ iteration (*Line* 22). Therefore, we get an outgoing path from r to the new BLACK node satisfying the intermediate node condition at $k+1$ iteration. □

Now, we are ready to conclude the correctness of the DAST algorithm from the above two lemmas.

THEOREM 1. *The DAST algorithm computes a SCDAS of G.*

PROOF. From Lemma 1, we know that DT and AT are respectively a dominating set and an absorbent set of G. Let u and v be a pair of vertices in $DT \cup AT$. By Lemma 2, there exist a path from u to r whose intermediate nodes belong to AT and a path from r to v whose intermediate nodes belong to DT. By linking these two paths, we get a path from u to v such that all intermediate nodes are in $DT \cup AT$. This also holds for a path from v to u. Thus the correctness of the DAST algorithm is proved. □

3.3 Approximation Ratio

In this section, we want to claim that the DAST algorithm guarantees constant approximation ratio if the ratio of the maximum to minimum in transmission range is bounded. In order to prove our claim, we first introduce a concept underlying in our proof.

Definition An *Independent Subset (IS)* of a directed graph $G = (V, E)$ is a vertex set $I \subset V$ which satisfies: for any $u, v \in I$, at least one of (u, v) and (v, u) is not in E.

Simply, any two vertices in an IS do not have bidirectional edges between them. This leads to the following property of an IS.

LEMMA 3. *For any two nodes u, v in an IS, the distance between u and v, $dist(u, v)$ is greater than r_{min}.*

PROOF. For two nodes, u and v, $dist(u, v) \leq r_{min}$ implies that both (u, v) and (v, u) are in E, which leads to a contradiction. □

The framework of our proof is to bound the size of any IS by using this minimum distance property of an IS. This method is similar to the one in [2]. We will see that the BLACK nodes returned by DT or AT actually forms an IS. Furthermore, it is easy to see that,

in the output of DT or AT, the number of BLUE nodes cannot be more than the number of BLACK nodes since a BLUE node is set as a parent of a BLACK node. Combining all these properties together, we prove that the DAST algorithm guarantees constant approximation ratio. Let us start with the property of the BLACK nodes returned by DT or AT.

LEMMA 4. *The set of BLACK nodes returned by DT or AT is an IS of G.*

PROOF. Let us assume that there exists a pair of BLACK nodes, u and v such that both (u, v) and (v, u) are in E. Without loss of generality, we can assume that u is marked with BLACK before v. Then, v must be WHITE when u becomes BLACK at some iteration k. Since $v \in N^+(u)$ by the assumption, v must be marked with GRAY at the iteration k. Once v has been colored GRAY, its color can change only to BLUE(*Line* 24), and BLUE nodes cannot change color. This implies that v will not be colored BLACK at any following iteration, leading to a contradiction. So, there is no pair of BLACK nodes that has bidirectional edges between them. □

LEMMA 5. *In a disk graph $G = (V, E)$, the size of any IS S is upper bounded by*

$$2.4(k + \frac{1}{2})^2 opt + 3.7(k + \frac{1}{2})^2$$

where $k = r_{max}/r_{min}$ and opt is the size of the optimal solution of the SCDAS problem.

PROOF. Let OPT^* and opt^* be an optimal strongly connected dominating set and its size. Let OPT and opt be an optimal SCDAS and its size. It is easy to see that $opt^* \leq opt$. Since OPT^* is a dominating set, any node in G must lie in the area A^* covered by OPT^*. This implies that any IS must lie in A^*. Note that a node in an IS may lie on the boundary of A^*. By Lemma 3, we know that all the disks centered at nodes in S with radius $r_{min}/2$ are disjoint. Thus the size of any IS is bounded by the maximum number of disks with radius $r_{min}/2$ packed in the area covered by the extension with radius $r_{min}/2$ for each node in OPT^*.

Let v_i, $1 \leq i \leq opt^*$ be the nodes in OPT^* and v_o be a dominated node, i.e., a node in G but not in OPT^*. Let D'_i be a disk centered at v_i with radius r_{max} and D'_o be a disk centered at v_o with radius $\frac{r_{min}}{2}$. Clearly, for a given disk D_i, there exists a disk D_j intersecting with D_i where $1 \leq i, j \leq opt^*$. Let L be the set of disks L_i with radius $(r_{max} + r_{min}/2)$ centered at v_i. Hence, all disks D'_i and D'_o must be contained in the union of the disks L_i. Each disk L_i is added as follows. At each iteration i, add a disk L_i centered at v_i such that there exists a node $v_j \in \{v_1, ..., v_{i-1}\}$ such that $d(v_i, v_j) \leq r_{max}$. The node v_j exists since all nodes in OPT are connected. The newly covered area A_i is bounded by two arcs of disk L_i and L_j in Fig. 2, where $d(v_i, v_j) = r_{max}$. Note that in Fig. 2, the disk L_j was added before the disk L_i, i.e., $j < i$. Let $\alpha = \angle X v_j v_i$ and $c = r_{max} + \frac{r_{min}}{2}$, we have:

$$\begin{aligned} A_i &\leq \text{area of } L_i - 2 \text{ area of the } \angle X v_j Y \text{ sector of } L_j \\ &\quad + \text{area of the diamond } X v_j Y v_i \\ &\leq \pi c^2 - 2\alpha c^2 + r_{max}\sqrt{c^2 - (\frac{r_{max}}{2})^2} \\ &\leq c^2(\pi - \frac{2\pi}{3}) + c r_{max} \\ &\leq c^2(\frac{\pi}{3} + 1) \end{aligned}$$

Hence the total area A covered by L is at most

$$c^2(\frac{\pi}{3} + 1) \cdot opt^* + c^2\pi$$

Note that $d(v_i, v_j) \geq r_{min}$ where v_i, v_j are in independent set S. Hence all disks with radius $\frac{r_{min}}{2}$ centered at nodes in S are disjoint. Now we compute how many such disks are in A. We

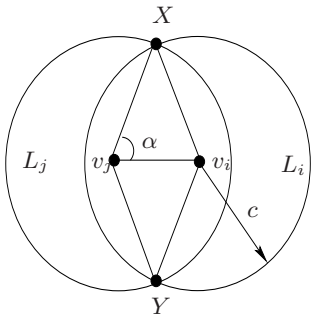

Figure 2: On the proof of the size relationship between a DAS and a SCDS

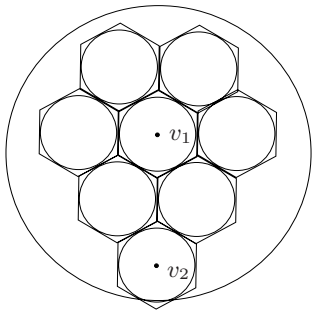

Figure 3: The densest packing of unit disks

know that the densest packing of unit disks in the plane is attained by a hexagonal lattice. For each disk d_i with radius $\frac{r_{min}}{2}$ centered at a vertex v_i where v_i is in the independent set S, place a regular hexagon of *width* r_{min} as shown in Fig. 3. Each hexagon has an area of $\sqrt{3}/2 \cdot r_{min}^2$. For example, in Fig. 3, the disk d_1 uses an area of at least $\sqrt{3}/2 \cdot r_{min}^2$.

Notice that a disk nearby the boundary might not use all its area. For example, in Fig. 3, the hexagon of the disk d_2 centered at v_2 has one part outside of disk D which is the biggest disk in Fig. 3. That part has an area of

$$(\frac{\sqrt{3}}{2})r_{min}^2 - (\frac{r_{min}}{2})^2\pi)/6$$

Hence each disk with radius of $r_{min}/2$ can use an area of:

$$(\frac{\sqrt{3}}{2}r_{min}^2 - (\frac{\sqrt{3}}{2}r_{min}^2 - \frac{\pi r_{min}^2}{4})/6) \geq .85 r_{min}^2$$

Therefore, the size of S is bounded by:

$$\begin{aligned} |S| &\leq \frac{\text{Total Area A}}{.85 r_{min}^2} \\ &\leq \frac{opt^*(c^2(\frac{\pi}{3} + 1)) + c^2\pi}{.85 r_{min}^2} \\ &\leq opt \frac{1 + \pi/3}{.85}\left(\frac{r_{max}}{r_{min}} + \frac{1}{2}\right)^2 + \frac{\pi}{.85}\left(\frac{r_{max}}{r_{min}} + \frac{1}{2}\right)^2 \\ &\leq 2.4\left(\frac{r_{max}}{r_{min}} + \frac{1}{2}\right)^2 opt + 3.7\left(\frac{r_{max}}{r_{min}} + \frac{1}{2}\right)^2 \end{aligned}$$

□

THEOREM 2. *The DAST algorithm produces a SCDAS with its size bounded by*

$$9.6(k+\frac{1}{2})^2 opt + 14.8(k+\frac{1}{2})^2$$

where $k = r_{max}/r_{min}$.

PROOF. Let C denote the SCDAS obtained from the DAST algorithm. Let $BLACK_{DT}$ and $BLUE_{DT}$ be the black and blue nodes in DT, respectively. Also, let $BLACK_{AT}$ and $BLUE_{AT}$ be the black and blue nodes in AT, respectively. When we add a BLACK node, we may add a BLUE node, and a BLUE node can be added at no other time.

Thus we know that $BLUE_{DT} \leq BLACK_{DT}-1$ and $BLUE_{AT} \leq BLACK_{AT} - 1$. Finally, from Lemma 3, 4 and 5, we can conclude:

$$\begin{aligned} |C| &\leq BLUE_{DT} + BLACK_{DT} + BLUE_{AT} + BLACK_{AT} \\ &\leq 2 \times (BLACK_{DT} + BLACK_{AT}) \\ &\leq 4 \times (2.4(k+\frac{1}{2})^2 opt + 3.7(k+\frac{1}{2})^2) \\ &\leq 9.6(k+\frac{1}{2})^2 opt + 14.8(k+\frac{1}{2})^2 \end{aligned}$$

□

COROLLARY 1. *If the ratio of the maximum to the minimum in transmission range is bounded, the DAST algorithm guarantees constant approximation ratio.*

4. HEURISTICS

In this section, we propose two efficient heuristics for MSCDAS problem, namely greedy spider contraction algorithm and greedy strongly connected component merging algorithm. The two algorithms are similar in terms of their structures, that is, first, find a DAS and add nodes in a greedy manner to make it connected. In fact, the two algorithms use the same subroutine to find a DAS. Thus, we begin with introducing the algorithm to find a DAS.

4.1 Finding Dominating and Absorbent Set

As shown in Fig. 2, the algorithm **FDAS** consists of two stages - construction of a DS and construction of an AS. It then outputs the union of the DS and AS. More specifically, in the phase of finding a DS, all nodes in the network are initially uncolored. At each iteration, an uncolored node u is selected and colored BLACK. Moreover, any uncolored node with an incoming edge from u, if any, is colored GRAY. When the uncolored node u is selected, we consider two criteria, namely, *random selection* and *highest degree first*. In the random selection policy, a node is randomly chosen among all uncolored nodes. In the highest degree first policy, a node with the highest vertex degree is chosen among all uncolored nodes. In this paper, the *degree* of a node is defined by the sum of the number of incoming edges and the number of outgoing edges. The node u which has just been colored BLACK is put into S, which will become a DAS. Let us assume that there is no uncolored node at the beginning of the i-th iteration. Then, S constructed so far forms a dominating set. Since there is no uncolored node in the network at the start of the i-th iteration, a given node $w \in V$ is either BLACK or GRAY. If there is a GRAY node which is not dominated by any BLACK node, this contradicts Line 11 where a node turns into GRAY only when a BLACK node dominates it.

Finding an AS consists of two parts, pre-processing and main processing. Once we get the list of BLACK nodes from the dominating set, we check if any dominated node (a GRAY node) is absorbed as well by a BLACK node. By doing this, we can avoid adding extra BLACK nodes unnecessarily in finding AS. If there exist such dominated nodes, we color them WHITE. Now, we have three different colored nodes in the network, that is, BLACK, GRAY, and WHITE. A BLACK node is in the DS described above. A GRAY node is dominated, but not yet absorbed by the DS. A WHITE node is dominated and absorbed as well by the DS

Once finishing the pre-processing, but having still GRAY nodes left, we choose a GRAY node in a greedy manner so that coloring the GRAY node BLACK can bring as many absorbed nodes as possible. We repeat this procedure until there is no GRAY node in the network. It is obvious that the final S forms a DAS with the reasoning similar to the proof of DS discussed above.

Algorithm 2 Find a Dominating and Absorbent Set (FDAS)

```
1: INPUT: A directed graph G = (V, E)
2: OUTPUT: A dominating and absorbent set S
3: /* initially, all nodes are uncolored */
4: /* find a dominating set */
5: S ← ∅
6: while there exits an uncolored node do
7:    find an uncolored node u ∈ V with policy 1 or 2 and color
      u in black
8:    S ← S ∪ {u}
9:    for v is an outgoing adjacent node of u do
10:      if v is an uncolored node then
11:         color v GRAY
12:      end if
13:   end for
14: end while
15: /* find an absorbent set */ STATE /* pre-processing */
16: for u is a BLACK node do
17:   for v' : an incoming adjacent node of u do
18:      if v' : GRAY then
19:         color v' WHITE
20:      end if
21:   end for
22: end for
23: /* main processing */
24: while there exists a GRAY node do
25:   find a GRAY node, w, with the most incoming edges from
      GRAY nodes
26:   color w BLACK
27:   S ← S ∪ {w}
28:   color the related GRAY nodes WHITE
29: end while
30: Return S
```

4.2 Greedy Spider Contraction Algorithm

In this section, we propose a heuristic called Greedy Spider Contraction Algorithm(G-SCA). The G-SCA algorithm finds a DAS first, then makes the DAS connected via extra nodes if necessary. In order to get a DAS, the G-SCA runs Algorithm **FDAS** described in the previous section. Before explaining how the G-SCA makes its DAS connected, let us introduce the definitions that will be used in the G-SCA.

Definition The directed Steiner tree with Minimum Steiner Nodes (DSMSN) problem is defined as follows: Given a directed graph $G = (V, E)$, a root node $r \in V$ and a set of nodes $S \subseteq V$ called *terminals*, find a directed tree $T = (V', E')$ rooted at r such that it spans all the terminals in S and $|V' - S|$ is minimum.

We call a node $w \in V' - S$ a *Steiner node* and T a *directed Steiner tree with minimum Steiner nodes*. The DSMSN problem is NP-hard since a node-weighted Steiner tree problem with node weight of 1 is a special case of DSMSN problem and it is known as NP-hard [3].

Definition Given a directed graph $G = (V, E)$, a root node $r \in V$ and a vertex subset $S \subset V$, a subgraph $T \subseteq G$ is said to be inconnected to a node r with regard to S if for every node $v \in S$, there exists a path from v to r.

Definition Given a directed graph $G = (V, E)$, a root node $r \in V$ and a vertex subset $S \subset V$, a subgraph $T \subseteq G$ is said to be outconnected from a node r with regard to S if for every node $v \in S$, there exists a path from r to v.

Consider S as a DAS returned by Algorithm **FDAS** and a node $r \in V$ as a root. Note that S is a set of BLACK nodes at end of Algorithm **FDAS**. Suppose that we have an inconnected tree to a node r from a every node in S and an outconnected tree from the node r to every node in S. Then the union of the inconnected and outconnected trees contains a path between any two nodes in S. In this way, we get a strongly connected subgraph that contains a DAS. This is the underlying framework of the G-SCA. However, our goal is to get a strongly connected subgraph containing a DAS where the number of additional nodes is as small as possible. In order to achieve that goal, we use a greedy method using a concept which is defined as follows.

Definition Given a directed graph $G = (V, E)$ and a root $s \in V$, a *directed spider* is an outconnected subtree rooted at s satisfying the following conditions: (1) all the other nodes except the root in the tree are NON-WHITE and (2) there exists a directed path in the tree from the root to every node in the tree.

Note that when a DAS is returned by Algorithm **FDAS**, there are two types of nodes in the network, that is, WHITE and BLACK. Let us call a directed spider rooted at a WHITE node v a v-spider. For a leaf node u in a directed spider, a directed path from v or r to u is called a *leg*. Note that for a v-spider, all the other nodes except v in a leg are NON-WHITE. As long as its DAS is not strongly connected, the G-SCA finds a directed spider with the largest number of NON-WHITE nodes, color the root of the spider BLUE, and contracts it. The contracting operation works as follows:

Definition *Contracting operation:* The contraction of a v-spider performs in the following way:

- Step 0 : start from $i = 1$
- Step 1 : for each undeleted node u at level i in the spider, check if the link (v, u) is bidirectional or unidirectional

  (1) add an unidirectional edge (v, w_o) for each $w_o \in N^+(u)$ such that $(v, w_o) \notin E$.

  (2) if (v, u) is bidirectional, add an unidirectional edge (w_i, v) for each $w_i \in N^-(u)$ such that $(w_i, v) \notin E$.

  (3) if (v, u) is unidirectional, add an unidirectional edge (w_i, w_o) for each $w_i \in N^-(u)$ such that $(w_i, w_o) \notin E$ for $w_o \in N^+(u)$.
- Step 2 : delete u
- Step 3 : repeat the above procedure for all the levels in the spider.

The contracting operation described above preserves connectivity between any two nodes. Suppose that there is a path $p = (u, v_1, v_2, \cdots, v_k, w)$ from node u to node w in the current graph, and the node v_i has been deleted. Then, in a new graph resulting from contracting operation, $p' = (u, v_1, v_2, \cdots, v_{i-1}, v_{i+1}, \cdots, v_k, w)$ still forms a path between u and w.

Now, we are ready to introduce our heuristic for the DSMSN problem, which is shown in Algorithm **SpanByDSpider**.

LEMMA 6. *Algorithm* **SpanByDSpider** *produces an outconnected tree rooted at r spanning all BLACK nodes*

PROOF. From the definition of spider contraction, note that once a directed spider contraction occurs, the root of the spider becomes a BLUE node. Moreover, all the other nodes in the spider are deleted. Since we delete all the incoming edges to the root r, a v-spider contraction containing r can happen only if the root of the v-spider is r. This implies that the root r must be the only one NON-WHITE node when the *WHILE* loop exits. Note that a BLUE node u is an abstraction of outconnected subtree rooted at u which contains only BLUE and BLACK nodes. When *WHILE* loop exits, there cannot exist a BLACK node in the network. Thus there exists an outconnected tree at r containing only and all BLUE and BLACK nodes. □

Algorithm 3 SpanByDSpider(G, r, S)

1: **INPUT**: Graph $G = (V, E)$, a root r, a set of BLACK nodes S
2: **OUTPUT**: An outconnected tree T rooted at r spanning all the nodes in S
3: delete all the incoming edges to r
4: $T \leftarrow \emptyset$;
5: **while** the number of NON-WHITE nodes in $G > 1$ **do**
6: find a directed spider with the largest number of NON-WHITE nodes
7: contract the selected spider
8: color the root of the spider BLUE.
9: update G
10: **end while**
11: construct T from the set of NON-WHITE nodes

Now, we are ready to describe the G-SCA algorithm which is shown in Algorithm 4. The G-SCA algorithm consists of two stages. At the first stage, the G-SCA constructs its DAS S using Algorithm **FDAS** At the second stage, Algorithm **SpanBySpider** is deployed to construct a strongly connected DAS.

Choose $s \in S$ with the largest transmission range among all the nodes in S. Note that at the beginning of Line 4, all the nodes in S are BLACK and all the nodes in $V - S$ are WHITE. By calling the algorithm ***SpanByDSpider***(G, s, S'), we construct an outconnected tree, T_s^o, rooted at s. In order to get an inconnected tree rooted at s, we reverse all the edges in the original graph G to construct a new graph G' and then call the algorithm ***SpanByDSpider***(G', s, S') to obtain a tree $T_s^{i'}$. By reversing all the edges in $T_s^{i'}$ back to their original directions, we can construct an inconnected tree T_s^i rooted at s. Last, taking the union of these two trees, T_s^o and T_s^i, produces a strongly connected subgraph containing a DAS since it has paths from s to s' and from s' to s for any given node $s' \in S$.

4.3 Greedy Strongly Connected Component Merging Algorithm

As mentioned earlier, the Greedy Strongly Connected component Merging algorithm (G-CMA) consists of two stages, finding a

Algorithm 4 Greedy Spider Contraction Algorithm (G-SCA)

INPUT: A directed graph $G = (V, E)$
OUTPUT: A SCDAS
run **FDAS**
choose node $s \in S$ with the largest transmission range
$S' = S - \{s\}$
$T_s^o = SpanByDSpider(G, s, S')$
construct $G' = (V, E^T)$, where $E^T = \{e' | e' = (v, u) \text{where} (u, v) \in E\}$
$T_s^{i'} = SpanByDSpider(G', s, S')$
construct T_s^i by reversing all the edges in $T_s^{i'}$
$H = T_s^o \cup T_s^i$
return all nodes in H

DAS and making it connected via extra nodes. More specifically, in the first stage, the G-CMA gets a DAS by running Algorithm **FDAS**. In the second stage, two strongly connected components (SCCs) of BLACK nodes are continually merged by using a shortest path between them until there is only one SCC of BLACK nodes in the network. The main steps of the G-CMA algorithm are described in Algorithm 5

Algorithm 5 Greedy Strongly Connected Component Merging Algorithm (G-CMA)

INPUT: A directed graph $G = (V, E)$
OUTPUT: A SCDAS
$H \leftarrow \emptyset$
run **FDAS**
$H \leftarrow S \cup H$
let $\mathcal{C}$ be the current collection of strongly connected components of BLACK nodes
while $|\mathcal{C}| > 1$ **do**
  for every pair (i, j) where $C_i, C_j \in \mathcal{C}$ and $i \neq j$ **do**
    find a minimum cost path, P_f to connect C_i to C_j
    find a minimum cost path, P_b to connect C_j to C_i
    let $Cost_{ij}$ be the number of distinct WHITE nodes along the forward path P_f and the backward path, P_b
  end for
  find a pair of (i', j') with the minimum $Cost_{ij}$ among all the pairs of (i, j)
  color BLACK distinct WHITE nodes, $w_1, w_2, \cdots w_k$ along the paths P_f and P_b
  $H \leftarrow H \cup \{w_1, w_2, \cdots w_k\}$
end while
Return H

The G-CMA algorithm constructs a SCDAS which is a subgraph H of G. After the first stage, H contains a DAS returned by Algorithm 2. If the DAS is not strongly connected, The G-CMA continues to find two SCCs which can be merged by the minimum cost until there is only one SCC in the network. At each iteration, the G-CMA considers every pair of SCCs in the network and calculates the merging cost for that pair. When two SCCs, C_i and C_j are considered for merging, we need to consider a forward path to connect C_i and C_j and a backward path to connect C_j and C_i since G is a directed graph. If G is a UDG, a forward path is also a backward path. However, this is not necessarily the case in DG. The G-CMA algorithm first finds a forward path with the minimum cost. In our algorithm, the cost of a path is defined as the number of WHITE intermediate nodes along the path. If there is more than one minimum cost forward path, one of such forward paths is arbitrarily chosen. Once a minimum cost forward path is discovered, the G-CMA assumes all intermediate nodes along that forward path as BLACK nodes and then finds a backward path with a minimum cost. Then, among all the pairs of SCCs, the G-CMA selects a pair of SCCs with minimum merging cost. One may get a cheaper path by considering all possible combinations of forward and backward paths for each pair of SCCs.

5. NUMERICAL RESULTS

In this section, we evaluate the schemes for MSCDAS problem through simulation which are proposed in the previous section. More specifically, we investigate the impact of the network density and the ratio of transmission range on the performance of each scheme in terms of the size of SCDAS. The network density can be varied by either changing the number of nodes in a given area or increasing the area size for a fixed number of nodes. In our simulation, we consider both ways to get different network density. For each run, we generate a network where nodes are randomly distributed, and choose the transmission range of each node randomly for a given range of powers. Once a network is generated, we check whether or not it is strongly connected. If not, the generated network is thrown away. Otherwise, it is considered as one instance. Repeating this procedure, we get 1000 instances of networks for each performance measure and take the average value for each point in simulation results. Since Wu's localized marking algorithm is the only one available in the literature with which our proposed schemes can be compared, we describe Wu's algorithm in detail as below.

For a given network $G = (V, E)$, every node is initially unmarked and has a unique id. A node broadcasts the information of its neighbors Whenever a node $u \in V$ discovers that (w, u), $(u, v) \in E$, but $(w, v) \notin E$, the node u marks itself as a member of SCDAS. Furthermore, two additional rules are applied in order to reduce the size of SCDAS. Before we explain the two rules, recall the definition of $N^-(u)$ and $N^+(u)$ for a given node u defined in Section 3. The former is the set of incoming neighbors of node u, while the latter is the set of outgoing neighbors of node u. In *Rule 1*, a marked node u unmarks itself whenever the *marked* node u finds that for some node $v \in V$, $N^-(u) - \{v\}$ and $N^+(u) - \{v\}$ are covered by $N^-(v)$ and $N^+(v)$, respectively and $id(u) < id(v)$. *Rule 2* is an extension of *Rule 1* to a triple of nodes. More specifically, a marked node u unmarks itself whenever all the following conditions are satisfied: (1) $N^-(u) - \{v, w\} \subseteq N^-(v) \cup N^-(w)$, (2) $N^+(u) - \{v, w\} \subseteq N^+(v) \cup N^+(w)$, (3) $id(u) = \min\{id(u), id(v), id(w)\}$

In our simulation results, we represent the performance of Wu's algorithm as *LOCAL*. *R-SCA* and *H-SCA* indicates the performances of the G-SCA with random selection policy and with highest degree first policy, respectively.

5.1 Effect of network density

5.1.1 Varying the number of nodes

To evaluate the performance of the proposed algorithms under the different number of nodes, we randomly deploy N nodes in a fixed area of $1000m \times 1000m$. N varies from 10 to 130 with an increment of 10. In other words, we vary network density from 10^{-5} to 1.3×10^{-4}. Each node randomly chooses its transmission range from the range of $[200m, 600m]$.

As shown in Fig. 4, the LOCAL performs the worst among all the schemes considered in our simulation as the number of node increases. Among the proposed centralized algorithms, the G-CMA does the best while R-SCA performs the worst. As the number

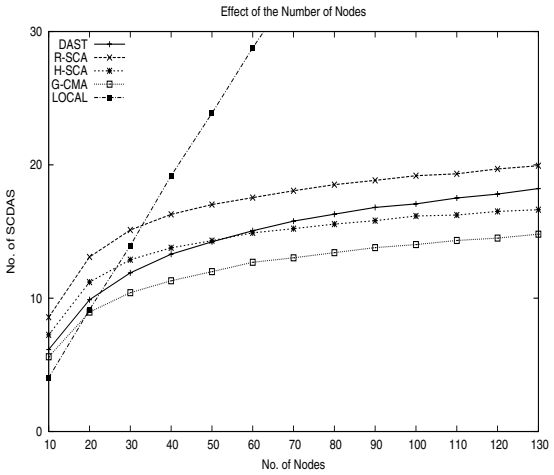

Figure 4: Effect of number of nodes

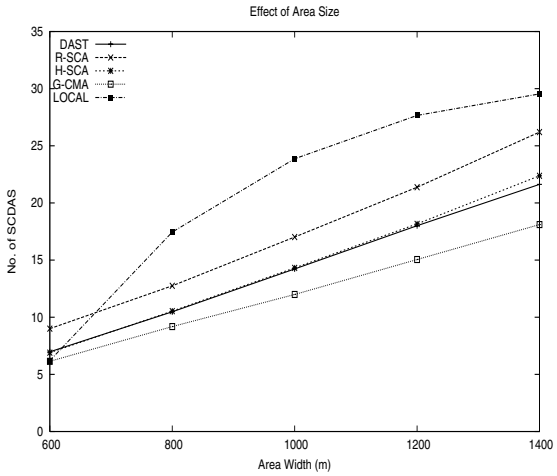

Figure 5: Effect of area size

of nodes increases, the size of SCDAS increases linearly with the LOCAL, while it tends to saturate with our centralized algorithms. More specifically, the LOCAL chooses as a SCDAS about 50% of the total number of nodes N in the network at the range of [10, 60] of N. On the other hand, the G-CMA finds its SCDAS with steadily decreasing percentage. For example, The size of SCDAS returned by the G-CMA is about 50%, 20% and 10% at $N = 20, 60$ and 130, respectively. Note that the differences between the LOCAL and the G-CMA in the size of SCDAS becomes very distinct as the network grows. This phenomenon is the same for the LOCAL versus the R-SCA (or H-SCA). The reason why the performance of the LOCAL does not improve as the network becomes dense is that *Rule 1* and *Rule 2* can be hardly satisfied when the number of nodes increases in a fixed area, while the power range remains the same.

It is also noticeable that the H-SCA performs better than the R-SCA. This implies that node selection method in finding a DAS have a significant impact on the resulting size of SCDAS. *highest degree first* policy always performs better than *random selection* policy for the Algorithm SCA. At a low network density, the difference in performance of the R-SCA and the H-SCA is slight. However, as the network density grows, H-SCA performance significantly better than the R-SCA. For example, the sizes of SCDAS produced by the R-SCA and the H-SCA are 8.85 and 8.16 $N = 10$, respectively, while 25.44 and 27.28 at $N = 90$. Intuitively, if a network is sparse, most of nodes in the network will be included in a DAS irrespective of random selection policy or highest degree first policy. If the network becomes dense, we have a room to use the benefit of node spatial distribution. So, highest-degree-first policy performs better than random selection policy.

One can observe that the DAST algorithm performs almost the same as the H-SCA. For a network density up to $N = 40$, the DAST algorithm is slightly better than the H-SCA. Beyond that, however, the performance of the H-SCA gets better than the DAST. Moreover, the difference of performance becomes distinct as the number of nodes increases. Compared to the G-CMA, the DAST algorithm obtains 120%, 136% and 130% of the size of SCDAS achieved by the G-CMA at $N = 40, 60$ and 130, respectively.

5.1.2 Varying area size

In this section, we vary the area size rather than the number of nodes in the network in order to evaluate the performance of the proposed algorithms under the different network density. We randomly deploy 50 nodes in the network. Each node randomly choose its transmission range from the range of $[200m, 600m]$. The area width varies from $600m$ to $1400m$. This corresponds to the network density from about 1.4×10^{-4} to about 2.5×10^{-5}. As shown in Fig. 5, the G-CMA performs the best among all the schemes considered in our paper. More specifically, the G-CMA obtains its SCDASs with about $52.6\%, 50.2\%$ and 61.2% of the size of SCDSs achieved by LOCAL at the area width $800, 1000$ and 1400, respectively. This corresponds to $18.3\%, 23.9\%$ and 36.2% of the total number of nodes in the network. Unlike varying the number of nodes in a given area mentioned in the previous section, the size of SCDAS with the LOCAL tends to saturate, while those of all centralized algorithms tend to increase as the area width becomes larger. We observe that, at the area width of 600, the performance of the LOCAL is as good as the G-CMA. In fact, with the area width of 600, the network becomes close to a complete graph with the parameters set as mentioned above where a node can communicate with all the other nodes. It is obvious that any localized algorithm will work well with a complete graph. The LOCAL chooses as its SCDASs about $12.4\%, 34.9\%, 47.8\%, 55.3\%$ and 59.08% of N at the area width of $600, 800, 1000, 1200$ and 1400, respectively.

Similarly in the previous section where the number of nodes varies in a given area, SCA with *random selection* policy performs the worst among all proposed schemes, at all area width considered in this paper. R-SCA obtains its SCDAS with about $29\%, 18.9\%$ and 17.1% more nodes than H-SCA at the area width of 600, 1000 and 1400, respectively. Rather than using random selection policy, when *highest degree first* policy is used, we observe that the size of SCDAS can be significantly reduced. Very interestingly, when the area size varies, the performance of H-SCA is the same as that of the DAST with slight difference at area width of 1400. Both algorithms achieve the size of SCDAS about $21\%, 36\%$ and 44.7% at the area width of $800, 1200$ and 1400, respectively. One can also observe that the difference in performance of the G-CMA and other schemes become distinct as the area size grows. This implies that strongly connected component merging technique can save unnecessary nodes much more efficiently than any other scheme when the network is sparse.

5.2 Effect of Transmission Range Ratio

In this section, we investigate the effect of transmission range ratio, k. In the previous section, we prove that the size of SCDAS produced by the DAST algorithm is bound by a function which is proportionate to k^2. Let us assume that the number of nodes and their locations in a given area are fixed. Consider the following two cases of transmission range; In case 1, all nodes have the same transmission range R. On the other hand, in case 2, 50% of nodes has a transmission range of $R_1 = \frac{2}{3} \cdot R$ and 50% of nodes have a transmission range of $R_2 = 2R_1$. Note that the average transmission range is the same for both cases. Now, assume that the network constructed by each case is strongly connected. Which case has a higher chance to have a larger DAS? Intuitively, the answer is the second case. It is because, in order for a node with a small transmission range to reach a node in DAS within one hop, the more BLACK nodes are needed. With the same reasoning, one can expect that if the value of k is larger, so is the size of SCDAS.

In the first experiment, we deploy 50 nodes randomly in the area of $1000m \times 1000m$. In order to vary the value of k, we fix $1000m$ for the maximum transmission range, while vary the minimum transmission range from $200m$ to $1000m$ for an integer value of $k \in [1, 5]$. Interestingly, the LOCAL performs the best with the low values of k. By noticing that the graph grows to a complete graph as the value of k becomes lower, a node can communicate with any other node directly without any intermediate nodes. So the size of SCDAS produced by the LOCAL becomes smaller. The intuitive reason why our centralized algorithms performs worse for lower values of k is that they finds SCDAS in multiple phases, which is possible to lead to larger sizes of SCDASs than the LOCAL in a certain network situation. However, one can observe that the performance of the LOCAL sharply degrades as k grows. Wu's localized algorithm obtains its SCDASs with $2.2\%, 11.9\%, 25.3\%$ and 32.3% of N for $k = 2, 3, 4$ and 5, respectively.

As shown in the Fig. 6, the H-SCA algorithm performs the best among the proposed algorithms for every value of k considered in the simulation. More specifically, it produces its SCDASs with $7.7\%, 13.7\%, 17\%$ and 19.6% of N for $k = 2, 3, 4$ and 5, respectively. From the figure, one can observe that the performance of the DAST algorithm is as good as the H-SCA algorithm for the entire interval of k. Even though we do not show in the figure, the R-SCA algorithm performs the worse than the DAST algorithm. Similarly to the results in the previous sections, it shows again that the random selection of nodes in drawing DAS performs badly, which is even worse than the simple pruning algorithm, the DAST. By setting a DAS via greedy choice of nodes with *highest degree first* policy, however, one can improve the performance of SCA significantly.

From the figure, one can observe that increase in the size of SCDAS obtained by each algorithm gets slow down as the value of k increases. It is also prominently noticeable that the G-CMA algorithm performs the best when $k = 1$. However, as k increases, the performance of the G-CMA becomes the worst. For instance, when k increases from 2 to 3, the size of SCDAS produced by the G-CMA sharply increases. From our experiment, we find that the G-CMA algorithm performs as bad as the R-SCA for higher values of k. Unlike the case where network density varies, the G-CMA does not perform efficiently when the transmission range ratio varies. In other words, the G-CMA algorithm is sensitive to the transmission range ratio with the parameters described above. We also investigate the performances of the proposed schemes in a larger network. At this time, we deploy 100 nodes randomly in the area of $1200m \times 1200m$. In order to vary the value of k, we select $1200m$ for the maximum transmission range, while vary the minimum transmission range from $200m$ to $1200m$ for an integer value of $k \in [1, 6]$. Unlike the first experiment, the G-CMA algorithm performs the best among all the schemes for $k \in [2, 6]$ as shown in the Fig. 7. It produces as the size of SCDAS $5.18\%, 14.1\%, 20.3\%$ and 22.6% of N with $k = 1, 3, 5$ and 6, respectively. Similarly to the case with $N = 50$, the performance of the LOCAL is the best when $k = 1$, i.e, when the network becomes a complete graph. When k becomes 2, however, the size of SCDAS achieved by the LOCAL increases very steeply, which is more than 30 times the one for $k = 1$. In fact, one can observe that the sizes of SCDAS by the LOCAL is about four times of the ones by the G-CMA for all values of $k \in [2, 6]$. We can conclude that the performance of the LOCAL is sensitive to the transmission range ratio. As the power levels available in a network becomes more various, the performance of the LOCAL degrades very sharply.

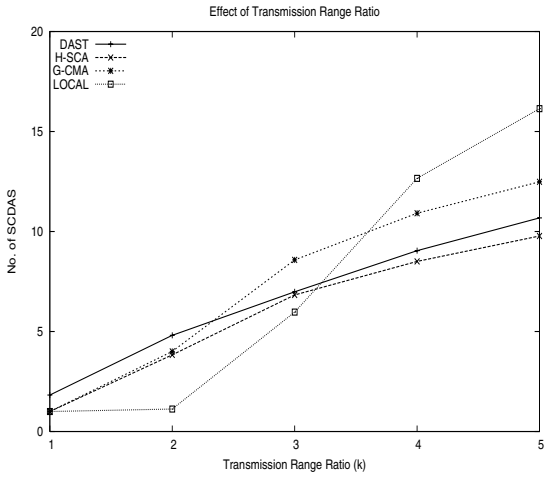

Figure 6: Effect of transmission range ratio, N=50

One can also observe that the DAST is as good as the G-CMA when $k = 1$, i.e., the network becomes homogeneous where each node have the same transmission range. However, the difference in performance of the DAST and the G-CMA becomes very distinct as k grows. For example, the size of SCDAS achieved by the DAST is $1\%, 3.7\%$ and 5.6% more than the one by the G-CMA at $k = 2, 4$, and 6, respectively. Even though we do not show in the figure, it is observed that, unlike the case with $N = 50$, the R-SCA performs similar to the H-SCA at the low values of k. As k increases, however, the H-SCA performs significantly better than the R-SCA. For example, when $k = 6$, the difference between the performance of the R-SCA and the H-SCA becomes 5.1%. Note that when the heterogeneity in transmission range grows, i.e., the value of k grows, the size of SCDAS achieved by each scheme also becomes larger. However, one can observe that every scheme tends to saturate as k increases.

6. CONCLUSIONS

In this paper, we discuss the Minimum Strongly Connected Dominating and Absorbent Set (MSCDAS) problem in a heterogeneous wireless ad-hoc network, where each node may have a different transmission range. The MSCDAS problem is the counterpart of MCDS problem in a homogeneous wireless ad-hoc network. We propose an O(1) approximation algorithm called DAST and give a theoretical analysis based on geometric properties of a disk graph. We also present two heuristics for MSCDAS problem, namely G-

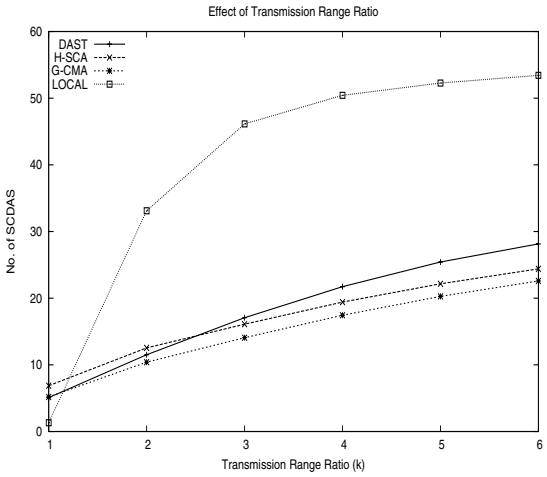

Figure 7: Effect of transmission range ratio, N=100

SCA algorithm and G-CMA algorithm. The G-SCA algorithm is based on greedy spider contraction, while the G-CMA algorithm is based on merge of strongly connected components with minimum cost. Through simulation, we investigate the performance of the proposed schemes under various network environments such as varying the total number of nodes in the network, the area size, and the transmission range ratio. The simulation results show that, overall, the G-CMA algorithm performs the best among all proposed schemes. However, when transmission range ratio varies with $N = 50$ in our experiment, the performance of the G-CMA algorithm becomes worse than the DAST. So, one needs to select an appropriate scheme according to network parameters.

7. ACKNOWLEDGMENTS

This work was supported in part by National Science Foundation under grant CCF0627233 and by the National Basic Research Program of China Grant 2007CB807900, 2007CB807901, the Hi-Tech Research and Development Program of China Grant 2006AA10Z216 and the National Natural Science Foundation of China Grant 60553001, 60604033.

8. REFERENCES

[1] F. Dai and J. Wu. An extended localized algorithm for connected dominating set formation in ad hoc wireless networks. *IEEE Transactions on Parallel and Distributed Systems*, 15(10):902–920, October 2004.

[2] S. Funke, A. Kesselman, U. Meyer, , and M. Segal. Simple improved distributed algorithm for minimum cds in unit disk graphs. *ACM Transactions on Sensor Networks*, 2(3):444–454, August 2006.

[3] S. Guha and S. Khuller. Improved methods for approximating node weighted steiner trees and connected dominating sets. *Information and Computation*, 150(1):57–74, April 1999.

[4] Y. Li, M. T. Thai, F. Wang, C.-W. Yi, P.-J. Wan, and D.-Z. Du. On greedy construction of connected dominating sets in wireless networks. *Wireless Communications and Mobile Computing (WCMC)*, 5(8):927–932, July 2005.

[5] M. T. Thai, F. Wang, D. Liu, S. Zhu, and D.-Z. Du. Connected dominating sets in wireless networks with different transmission ranges. *IEEE Transactions on Mobile Computing*, 6(7), July 2007.

[6] P.-J. Wan, K. M. Alzoubi, and O. Frieder. Distributed construction on connected dominating set in wireless ad hoc networks. *Mobile Networks and Applications*, 9(2):141–149, April 2004.

[7] J. Wu. An extended dominating-set-based routing in ad hoc wireless networks with unidirectional links. *IEEE Transactions on Parallel and Distributed Systems*, 13(9):866–881, September 2002.