COORDINATED ENERGY CONSERVATION IN AD HOC NETWORKS

Selma Ikiz, Vinit A. Ogale and Vijay K. Garg
{ikiz, ogale, garg }@ece.utexas.edu
Dept of Electrical and Computer Engineering
The University of Texas at Austin
Austin TX 78712

ABSTRACT

This paper presents a new power conservation scheme for multi-hop ad hoc networks. A virtual backbone consisting of special nodes (coordinators) is used for the power saving algorithm and routing. We present a new distributed algorithm for constructing a connected dominating set (CDS) that is used to construct and maintain the virtual backbone of the network. Our scheme includes a message history based variable sleeping time for the non-coordinators. Simulations indicate that our scheme results in better power conservation than other practical schemes discussed in the literature if the network has a sparse message density.

`Keywords`: Ad-hoc network, energy efficient routing, virtual backbone formation

1 INTRODUCTION

An ad hoc network is a collection of wireless mobile hosts without any fixed infrastructure. In such a network each host can act as an intermediary and forward packets to the next hop in order to reach the final destination. Ad hoc mobile networks have far reaching applications due to their suitability for rapid deployment and inherent robustness.

Ongoing research addresses issues like routing, network management, QoS, Media Access Control (MAC) protocols, topology management, mobility and security [1][2]. Routing is a fundamental issue for any network and, not surprisingly, is a very active topic of research in ad hoc networks [1][3][4][5][6][7][8][9]. Also, in most mobile ad hoc networks, power seems to be a major constraining factor [1][3]. Hence energy conservation is one of the key issues in any protocol or algorithm for ad hoc networks.

In this paper we focus on energy conserving dynamic backbone based routing technique for ad hoc networks[3][7]. A backbone based scheme involves partitioning the network into coordinators and non-coordinators. The coordinator nodes are responsible for the routing within the network and hence need to be active. The set of the coordinator nodes constitutes the *backbone* of the network. In contrast, the non-coordinator nodes are responsible for only the packets sent by them or addressed to them and are allowed to enter very low power consuming *sleep* states. This can drastically increase the life of the non-coordinators. Since coordinator nodes can not sleep, it is important to dynamically update the backbone according to the power remaining in each node, if we wish to increase the overall life of the network. Our paper makes two important contributions. First, we present a new algorithm for maintaining and constructing the backbone. Secondly, we propose a power saving protocol, which allows the nodes to sleep for varying amounts of time depending on the message history of that node. Our simulations show that the proposed scheme results up to 50% power saving compared to existing schemes.

2 BACKGROUND AND RELATED WORK

The backbone formation can be considered to be a problem of determining a *connected dominating set* (CDS) in ad hoc networks. A *dominating set* (DS) of a graph G is a subset V_s of the vertex set V such that each node that does not belong to the subset V_s is adjacent to a node in V_s. A connected dominating set is a dominating set which induces a connected subgraph.

We know that the task of finding a *minimum connected dominating set* (MCDS) in an ad hoc network is NP-hard [10]. Fortunately, in an ad hoc network with high chances of link failures and topology changes due to mobility, constructing a minimum CDS is not the aim. Some redundancy in the backbone is desirable to increase reliability of the network and mitigate the backbone maintenance.

Current work in this field includes different algorithms to construct a CDS [1][11][12]. In [1] a scheme which factors power in construction of a CDS is proposed. However, in this scheme the nodes forming the backbone are not changed periodically. As a result the coordinators spend more energy and die out much sooner than the other nodes.

Chen *et.al.* [3] suggest a protocol (*Span*) to form a CDS and change the coordinator nodes periodically. Span adaptively selects coordinators to form a backbone. It has several rules based on the node's remaining energy level and number of neighbors for coordinator announcement and withdrawal. It also assumes periodic broadcasting of `HELLO` messages that contain the node's status, its neighbors and each neighbor's status. Span uses an approach similar to the 802.11 ad hoc power-saving mode (PSM) that uses periodic beacons to synchronize nodes in the network. This synchronization is done at `MAC` layer. In the 802.11 PSM, beacon periods are divided into two time slots; the advertisement time and the advertised packet transfer time. Moreover, Span increases energy saving by adding another window only for non-coordinator nodes. For such nodes, there are three windows inside a beacon; `ATIM`, `NATIM`, and the rest. `ATIM` is used to transfer packet advertisements from node to node and coordinator to node. New advertisement window (`NATIM`) is used for packet transfers between these nodes. The remaining time is used by coordinators for packet transfer among themselves.

In this paper our scheme extends and improves Span. We use a new CDS selection algorithm presented in the next section. We also propose a new scheme which enables inactive nodes to sleep for longer periods thus improving the overall network life.

3 NETWORK MODEL

In a typical ad hoc wireless network, each node can have a different transmission radius resulting in unidirectional links between nodes. For convenience, we simplify the underlying network topology by disregarding all the unidirectional links. Hence we consider two nodes to be connected if and only if there exists a bidirectional link between them. The topology of such a network can be modeled as a unit-disk graph (UDG) [11]. Yet, a mobile object is positioned in a 3D world [6]. Hence, we model the underlying topology as a unit-sphere graph (USG). This is a realistic model, if the network topology is unknown and the nodes have identical functions. Generally in such cases, unidirectional links are not useful. The simplified 3D topology for the network is shown in Fig 1. Each node represents a vertex and we draw an edge between two nodes if and only if a bidirectional wireless link exists between the nodes. Note that the network topology is not static and may change frequently as nodes move, die or join the network.

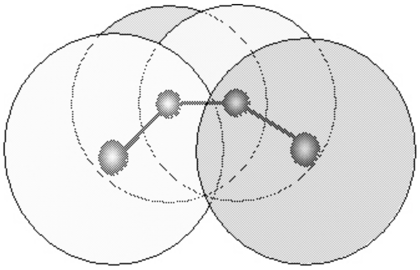


Figure 1. Model the topology of wireless ad hoc network by unit-sphere graphs.

4 VIRTUAL BACKBONE FORMATION ALGORITHM

In this section we present our virtual backbone formation algorithm. Nodes apply the rules of the algorithm locally to form a CDS, which forms the virtual backbone of the network.

4.1 NOTATION

Let us represent the network as a connected graph $G = (V, E)$. The backbone consists of a smaller graph $G' \subset G$ such that G' remains connected. Let :

1. $N(v)$ be the *open neighbor set* of vertex v, i.e., $N(v) \equiv \{u \mid \{v, u\} \in E\}$,
2. $n(v)$ be the cardinality of $N(v)$
3. $p(v)$ be the power metric defined as $p(v) = E_r/E_t$ where E_t is the maximum amount of energy available at the node and E_r is the remaining energy at the node,
4. $ID(v)$ be the unique node identifier of *v*.

We construct a tuple $T(v) \equiv \langle N(v), p(v), n(v), ID(v)\rangle$. To compare two such tuples we use the following rule; $T(v) \prec T(u)$ if and only if $N(v) \subset N(u)$, or $N(v) \equiv N(u)$ and we compare the other elements lexicographically. The two tuples are equal if and only if all their elements are equal.

Clusters : Let v be a coordinator, then we say $c.v =$ `true`, and $N(v)$ is the *cluster* of v.

Each node in the network periodically sends a `HELLO` message containing a list of its neighbors. Each node maintains a list of its neighbors and each neighbor's neighbor list. The nodes in the neighbor table are marked as either coordinators or non-coordinators.

4.2 ACTIVITY I: INITIATE COORDINATOR SELECTION

4.2.1 Rule 1

If any two neighbors of node *v* are not connected, then node v declares itself as the coordinator. This algorithm is similar to the algorithm proposed by Wu and Li in [3][1].

4.3 ACTIVITY II : REDUCE THE BACKBONE SIZE

4.3.1 Rule 1

A coordinator node *v* withdraws as a coordinator and sends `NLC` (No Longer a Coordinator) to its neighbors if there exist a node u i.e. $T(v) \prec T(u)$.

4.3.2 Rule 2

For the coordinator node *v* ::

1. Node v sends `RTW` (Request to Withdraw) to all neighboring coordinators including itself if :
   (a) all coordinator neighbors of node *v* are connected to each other directly or through a coordinator, and
   (b) all the non-coordinator neighbors of *v* are connected to at least one other coordinator other than *v*.
2. If node *v* receives `PTW` (Permission to Withdraw) from all neighboring coordinators, then node *v* withdraws as coordinator and sends `NLC` (No Longer a Coordinator) to all neighbors including itself.
3. If node *v* receives `RTW` from one or more nodes, then it waits till the end of the round and sends `PTW` to the node that has the minimum T value.

4.4 ACTIVITY III : POWER-BASED COORDINATOR RE-SELECTION

A coordinator consumes more than thrice the energy consumed by a sleeping node [3]. Hence it is likely that the coordinators will die much before the non-coordinators, with the potential consequence of partitioning the network. To prevent this and maximize the overall network life, our algorithm allows a coordinator *v* to withdraw if its power metric $p(v)$ is lower than the power metric of its neighboring non-coordinator nodes.
For a coordinator node *v* ::

1. If the power metric *p* of all non-coordinator neighbors of node *v* is least 15% of $p(v)$ and all coordinators of *v* are connected to each other *directly or through another node*, then *v* sends a `RCR` (Request for Coordinator Re-selection) message to all the nodes on the alternative path and also to each of its neighbors including itself.
2. If node *v* receives a `RCR` from node *u*, then node v
   - waits till the end of the round for all `RCR`s,
   - sends `VTC` (Volunteer To be a Coordinator) to the coordinator with the minimum power metric,
   - sends a `CRI` to all other nodes, and
   - announces itself as the coordinator.
3. If node *v* receives a `VTC` from all its neighbors, it withdraws as a coordinator. Otherwise, on receiving a `CRI` it sends a `WCRR` (Withdraw Coordinator Re-selection Request) to all neighbors.
4. On receiving a `WCRR` a node, which had become a coordinator in response to the `RCR`, withdraws.

Each coordinator in the network periodically checks if it can withdraw by applying Activity III. Thus the algorithm tries to balance out the energy consumption amongst all the nodes in the network.

5 EXAMPLE

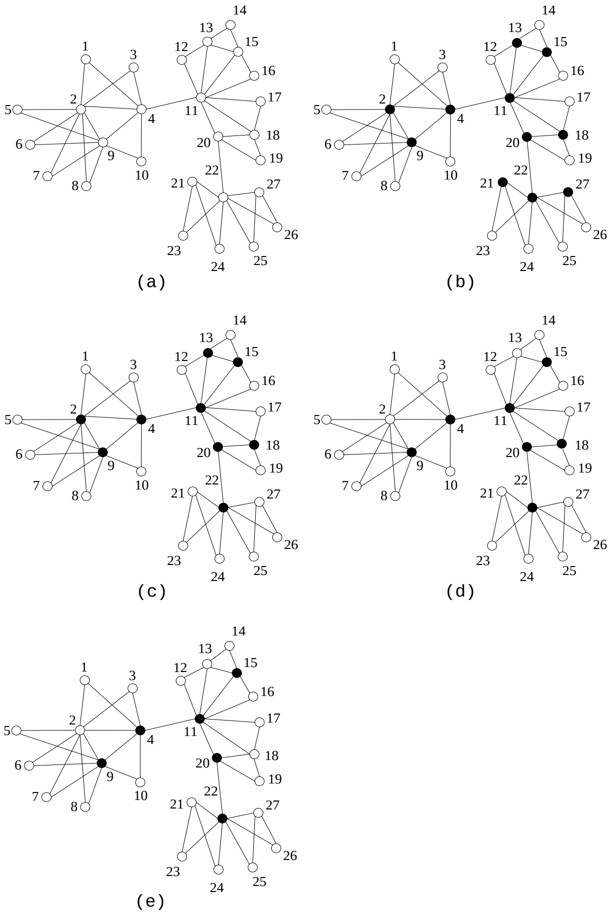
Fig.2 shows an example of using the proposed marking algorithm. Since each node keeps track of all its neighbors, it broadcasts its neighbor list and their states periodically. After this information exchange phase, every node will have information on all nodes with a radius of two units. In Fig.2 (a), node 1 will not mark itself as a coordinator node since two of its neighbor is directly connected. However, node 2 will mark itself as a coordinator seeing that node 5 and node 4 does not have a connection. Fig.2 (b) shows the resultant graph of the second phase. These coordinator nodes form a connected dominating set, but not the minimal one. After applying Rule 1 at the third phase, node 21 and 27 will withdraw from being a coordinator, and will be unmarked as shown in Fig.2(c). At phase four, coordinator nodes form their clusters, which are simply just their neighbors. Node 2 will check if all its coordinator neighbors are connected, and all its non-coordinator neighbors are connected to one more coordinator. After seeing that its coordinator neighbors, node 4 and node 9, are connected, and all its other neighbors have a coordinator other than itself, node 2 will decide to withdraw. Similarly, node 9, 13, 15, and 18 will decide to withdraw, and broadcast withdraw request. Assuming energy levels are equal, node 2 will get permission from node 9, while node 9 cannot get a permission from node 2. For node 13, 15, 18 and 19, node 11 will give permission to just one of them. Therefore, after the first round, graph will be as shown in Fig.2(d). At the beginning of the second round, node 9 will see that it cannot not decide to withdraw since node 8 does not have another coordinator. Likewise node 15 cannot decide. On the other hand, node 18 will still insist at withdrawing, and will get the permissions. Fig.2(e) shows the final graph.

6 THE APPROXIMATION FACTOR OF CDS

Peng, Khaled and Ophir in [11] reinvestigate CDS algorithms in [4][5][13] and [1] and establish an approximation factor for each one of them. By using their approach, we show that our proposed algorithm has an approximation factor of $\frac{n}{4}$ [14]. Peng *et.al.* show that their algorithm has an approximation factor of 8 in 2D environment. In 3D environment, we show that their approximation factor is also $\frac{n}{4}$ [14].

7 PROOF OF VALIDITY

Lemma 1 *The coordinators decided by Activity I form a connected dominating set if the underlying network is connected.*



(a) (b) (c) (d) (e)

Figure 2. An example of marking process

Proof: (Outline) Assume that the coordinators do not form a CDS. Therefore, there exist two coordinators connected by a non-coordinator node. Now Activity I requires such a node to be a coordinator. Therefore the claim is true. ■

Claim 1 *In Activity II, Rule 2, if node* v *withdraws then no other coordinator in the same or adjoining cluster can withdraw in the same round i.e. if $c.v \wedge c.u \wedge (N(v) \cap N(u) = \Phi$ is true in this step, then $c.v \vee c.u$ is true in the next step.*

Proof: Rule 2 requires a coordinator to get a `PTW` message from all neighbors before withdrawing. Each neighbor can give a `PTW` to exactly one node in a round. Hence the claim follows. ■

Lemma 2 *The connectivity of the dominating set does not change due to Activity II.*

Proof: We show that if there exists a CDS before applying Activity II, then the connectivity is maintained after applying Activity II.
CASE 1: A node *v* withdraws due to Rule 1 implies that there exists a node u i.e. $(T(v) \prec T(u) \wedge c.v \wedge c.u)$. Therefore node *u* has a link to all the nodes connected by *v*. Since both cannot withdraw in the same round, the connectivity is maintained.
CASE 2: A node *v* withdraws due to Rule 2. After applying Rule 2, the connectivity of the network could change if

1. there is a non-coordinator neighbor of v that has no coordinator, or
2. there is no path between two coordinators in CDS.

The algorithm eliminates the first possibility, because the coordinator *v* withdraws only if all the non-coordinator neighbors are connected to at least one other coordinator. Node *v* also checks if all its coordinator neighbors are connected to each other directly or through an other coordinator. Hence, the backbone cannot be partitioned due to the withdrawal of *v*. ■

Lemma 3 *The connectivity of the dominating set in the network does not change due to Activity III.*

Proof: This proof is similar to the proof of Lemma 2. ■

Theorem 1 *Activities I, II and III applied in sequence result in the formation of a CDS in a connected graph.*

Proof: This follows from Lemmas 1, 2 and 3. ■

Theorem 2 *The distributed algorithm proposed in this paper have an approximation factor of $\frac{n}{4}$, and $O(m)$ message complexity in 3D environment, where n is the number of nodes and m is the number of links in the network.*

Proof: For every activity in the algorithm messages are exchanged locally, and no message spurs any other. Hence, the message complexity is $O(m)$. Proof for the approximation factor, refer to [14]. ■

8 VARIABLE SLEEPING TIME BASED ON HISTORY

Our approach is based on Span's `NATIM` success, however, it furthers its effect by taking packet delivery history into account. We assume a virtual backbone is formed by coordinator nodes, and non-coordinator nodes are allowed to turn off their radio receivers. In Span's design `NATIM` is constant. To increase the power saving, we propose this value to be variable with an upper bound. The second improvement is non-coordinator nodes should use a two bit history for sleeping time. When a node observes two consecutive beacons without any packet advertisement, it decides to sleep through the next beaconing period. Fig.3 shows its transition graph.

Whenever a coordinator does not get a reply back after two consecutive ATIM windows, it removes the neighbor node from its neighbor table and clears its buffer. This process is the same for an immediate neighbor node, except it sends the packet through a coordinator.

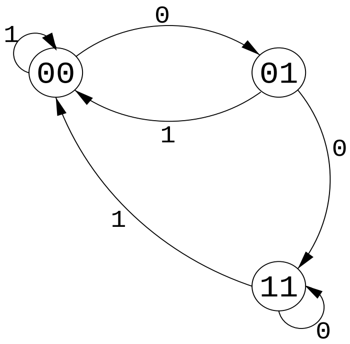


Figure 3. The state transition graph.

8.1 SIMULATION

Receiver nodes are the ones that are affected most from this approach. Sender nodes are somewhat affected. Coordinator nodes are least affected, since from their perspective nothing changes.
Therefore we simulate the Span, 802.11 PSM, and our algorithm for only a receiver node with different packet receiving probabilities. We also calculate the possible minimum value, i.e. receiver is awake only when there is a message transfer, which is the absolute minimum power requirement. We use the same beacon, ATIM and NATIM values given to be optimal in Span simulations [3]. A packet is fixed to 128 bytes, and number of packets per beacon period is limited to 100. The simulator randomly selects when the node receives the packets in the NATIM window. When the node receives the last packet destined to it, it gets into sleeping state without waiting the end of NATIM window. For power consumption, we use the values as shown in Table 1.

Table 1: Power consumption of the Cabletron 802.11 network card in the Tx (tansmit), Rx (receive), Idle, and sleeping modes [3].

Tx	Rx	Idle	Sleeping
1400mW	1000mW	830mW	130mW

Simulation results indicate that employing variable NATIM window and sleeping time ameliorates the power saving of a receiver Fig.4. It is a significant improvement over the 802.11 PSM, and a sensible improvement over Span. The advantages of our scheme are pronounced when the network has a high node density and low packet arrival probabilities.

8.2 WORST CASE ANALYSIS OF THE ALGORITHM

The worst case for our algorithm arises when a receiver has to stay awake till the end of the NATIM window. In this section, we only compare our algorithm with SPAN by calculating the expected sleeping time of a receiver at a beaconing period. Here we show that our algorithm's performance is equal to Span's in the worst case.

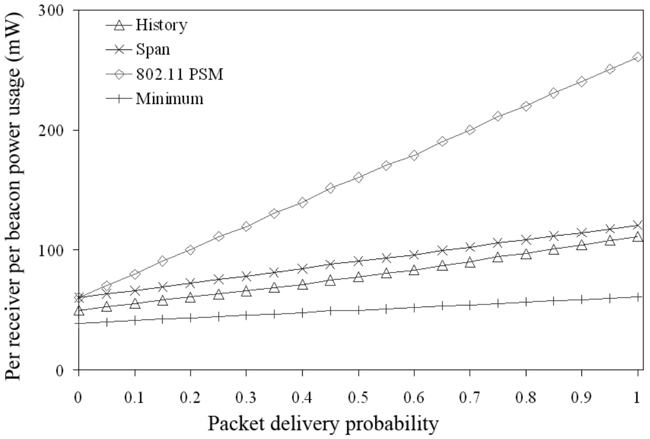


Figure 4. Comparison of energy consumption for a receiver.

Let's first define the notation. At the n^{th} beacon for a receiver,

1. A_n represents the arrival of a packet,
2. X_n represents the sleeping time,
3. S_n represents the state, where state = *sleep*, *awake*,
4. $P[A_n = 1] = p$ and $[P[A_n = 1] = 1 - p$, where p is the probability that a packet arrives,

Then, sleeping time formula for our algorithm is as follows;

$$X_n = \begin{cases} \text{beacon-NATIM}, & \text{if } A_n = 1 \wedge S_n = awake; \\ \text{beacon-ATIM}, & \text{if } A_n = 0 \wedge S_n = awake;; \\ \text{beacon}, & \text{otherwise}; \end{cases}$$

Refer to [14], for a detailed calculation of the formulas. The sleeping time formula for Span is as follows;

$$X_n = \begin{cases} \text{beacon - NATIM}, & \text{if } A_n = 1 \\ \text{beacon - ATIM}, & \text{otherwise}; \end{cases}$$

For our algorithm the expected sleeping time formula for a given packet arrival probability p is;

$$\begin{aligned} E[X_n] = & \ beacon \\ & - \frac{p(2-p)}{p^3 - 2p + 2} NATIM \\ & - \frac{2(1-p)^2}{p^3 - 2p + 2} ATIM \end{aligned} \quad (1)$$

And, for Span the formula is;

$$E[X_n] = beacon - p.NATIM - (1-p)ATIM \quad (2)$$

Fig.5 shows the graph of the expected sleeping times of a receiver according to SPAN and our algorithm for various packet arrival probabilities.

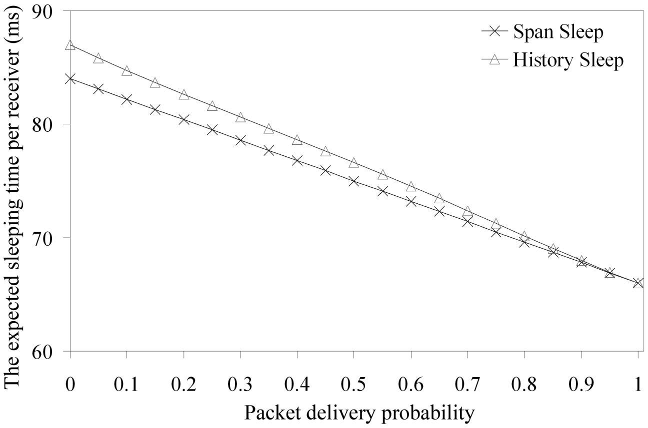


Figure 5. Comparison of expected sleeping time in a beacon.

Sender nodes are affected by our algorithm, if

1. the immediate sender node cannot get an `HELLO` message from the receiver node, and
2. it has a packet to the receiver.

Then the receiver has to wake up during the following beacon to check if the neighbor is still there. The overhead of this process is the receiver has to spend at most one more ATIM time. The expected power usage value of this overhead is 1.38 mW per beacon (a detailed calculation can be seen in [14].

9 CONCLUSION

In this paper we have extended Span's energy saving mode by using history and variable NATIM time. Although, using CDS improves both the network's and node's life span, the gap between sleeping energy consumption and idle mode need some more research.

Wu and Li [1] proposed a distributed algorithm for approximating connected dominating sets in ad hoc networks that also appears to preserve the capacity. In a later paper Wu *et.al.* [12] enhance their algorithm by adding new rules and refining the existing ones. Our algorithm, however, elects fewer coordinators because it actively checks the redundancy locally whenever there is a local topological change. Therefore, in this paper, we also have established a better CDS forming algorithm for ad hoc networks. Using a similar method to [11], we established an approximation factor for our algorithm.

A further research topic could be focusing on more in-depth simulation under different settings to see the effects of CDS construction algorithm in conjunction with variable sleeping time.

REFERENCES

[1] Wu, J., and Gao, M. "On Calculating Power-Aware Connected Dominating Sets for Efficient Routing in Ad Hoc Wireless Networks", in Proc. of the 30th Annual International Conference On Parallel Processing,Valencia, Spain. Sept. 2001.

[2] A. Penttinen, "Research On Ad Hoc Networking: Current Activity And Future Directions", http://keskus.hut.fi/opetus/s38030/k02/Papers/13-Aleksi.pdf

[3] B. Chen, K. Jamieson, H. Balakrishnan, R. Morris, "Span: An energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks", in Proc. ACM International Conference on Mobile Computing and Networking, Italy, July 2001

[4] B. Das, V. Bharghavan, "Routing in ad-hoc networks using minimum connected dominating sets", in IEEE International Conference on Communications, Montreal, Canada. June 1997.

[5] B. Das, E. Sivakumar, and V. Bhargavan, "Routing in ad-hoc networks using a spine", in IEEE International Conferenceon Computer Communications and Networks, Las Vegas, 1997.

[6] G. Kao, T. Fevens and J. Opatrny, "Position-Based Routing on 3-D Geometric Graphs in Mobile Ad Hoc Networks", Proceedings of the 17th Canadian Conference on Computational Geometry (CCCG'05), pp.88-91.

[7] D.B. Johnson and D.A. Maltz, "Dynamic Source Routing in Ad Hoc Wireless Networks", Mobile Computing, Kluwer Academic Publishers, 1996.

[8] C. Perkins, "Ad Hoc On Demand Distance Vector (AODV) Routing", draft-ietf-manet-aodv-00.txt, November 1997.

[9] Y. Xu, J. Heidemann, and D. Estrin, "Geography-informed Energy Conservation for Ad Hoc Routing", in MobiCom'2001.

[10] B. N. Clark, C. J. Colbourn, and D. S. Johnson, "Unit disk graphs", Discrete Mathematics, V86, pp. 165-177, 1990.

[11] P. Wan, K.M. Alzoubi, O. Frieder, "Distributed Construction of Connected Dominating Set in in Wireless Ad Hoc Networks", in Proc. INFOCOM, 2002, pp. 1597-1604.

[12] J. Wu, M. Gao, I. Stojmenovic, "On calculating power-aware connected dominating sets for efficient routing in ad hoc wireless networks", Journal of Communication and Networks, March 2002.

[13] I. Stojmenovic, M. Seddigh, J. Zunic, "Dominating sets and neighbor elimination-based broadcasting algorithms in wireless networks", IEEE Transactions on Parallel and Distributed Systems, 2002, pp. 14-25.

[14] S. Ikiz, V. Ogale, and V.K. Garg, "A Energy Conservation Algorithm for Mobile Ad Hoc Networks", Technical Report TR-PDS-2005–02, PDSL, ECE Dept. Univ. of Texas at Austin.