



A Weighted-Density Connected Dominating Set Data Gathering Algorithm for Wireless Sensor Networks

Larry King

Clemson University, Clemson, SC 29634, USA

E-mail: larryfking3@gmail.com

Natarajan Meghanathan (Corresponding author)

Department of Computer Science, Jackson State University

P. O. Box 18839, 1400 John R. Lynch Street, Jackson, MS 39217, USA

Tel: 01-601-979-3661 E-mail: natarajan.meghanathan@jsums.edu

This research is funded by the U.S. National Science Foundation through grant (CNS-0851646) entitled: "REU Site: Undergraduate Research Program in Wireless Ad hoc Networks and Sensor Networks."

Abstract

We propose a weighted-density connected dominating set (wDCDS) based data gathering algorithm for wireless sensor networks. The wDCDS is constructed using the weighted-density of a sensor node, which is defined as the product of the number of neighbors available for the node and the fraction of the initially supplied energy available for the node. A data gathering tree (wDCDS-DG tree) rooted at the wDCDS Leader (the node with the largest available energy) is formed by considering only the nodes in the wDCDS as the intermediate nodes of the tree. The leader node forwards the aggregated data packet to the sink. The wDCDS and wDCDS-DG tree are dynamically reconstructed for each round of data gathering. Simulation studies reveal that the wDCDS-DG tree yields a significantly larger network lifetime, lower delay and lower energy consumption per round compared to the density-only CDS and energy-only CDS based data gathering trees.

Keywords: Connected Dominating Set, Density, Energy, Data Gathering Tree

1. Introduction

A wireless sensor network is a distributed system of smart sensor nodes that collect data about the ambient environment and propagate the data back to one or more control centers called 'sinks', which access the data. A sensor node typically has limited battery charge, computing capability and memory capacity. The transmission range of a sensor node is the distance within which the signals emanating from the node can be received with appreciable signal strength. Wireless sensor networks have limited bandwidth as the sensor nodes within the transmission range of each other share the communication medium. As the sink is normally fixed and located far away from the sensor network field, direct transfer of the collected data to the sink is not a viable solution from both energy as well as bandwidth point of view. This motivates the need for data gathering algorithms that can be effectively and efficiently run at the sensor nodes to combine the data and send only the aggregated data (that is a representative of the data collected from all the sensor nodes) to the sink. Throughout this paper, the terms 'data aggregation', 'data fusion' and 'data gathering' are used interchangeably. They mean the same.

Data gathering algorithms typically proceed in rounds, wherein during each round, data from all the sensor nodes are collected and aggregated, and then forwarded to the sink. The communication structures normally used for such data aggregation are clusters (Heinzelman, Chandrakasan & Balakrishnan, 2004), grid (Luo, Ye, Cheng, Lu & Zhang, 2005), chain (Lindsey, Raghavendra & Sivalingam, 2002), connected dominating set (CDS) (Meghanathan, 2009) and trees (Lindsey, Raghavendra & Sivalingam, 2001). Meghanathan (2009) proposed an energy-based algorithm to construct the CDS (called the ECDS) of the underlying sensor network for every round of communication and also to construct a data gathering tree (ECDS-DG tree) based on the ECDS. The ECDS strategy prefers to include nodes that have a relatively high energy as intermediate nodes of the data gathering tree (i.e., used for data collection, aggregation and forwarding);

whereas, nodes with relatively lower energy are used only as leaf nodes of the tree (i.e., for data collection and forwarding only). Simulation studies (Meghanathan, 2009) revealed that the ECDS based algorithm yields larger network lifetime than that observed with the classical algorithms such as the cluster-based LEACH (Low-Energy Adaptive Clustering Hierarchy; Heinzelman, Chandrakasan & Balakrishnan, 2004) and the chain-based PEGASIS (Power-Efficient Gathering in Sensor Information Systems; Lindsey, Raghavendra & Sivalingam, 2002). Throughout this paper, we define the network lifetime as the time of first node failure due to exhaustion of battery charge.

Even though, the ECDS-based algorithm leads to a significant improvement in the network lifetime, the algorithm does not intend to minimize the number of nodes that form the CDS. We conjecture (this has been also vindicated in our simulations) that the overall delay and energy consumption could be further lowered by attempting to reduce the number of nodes that are part of the CDS. This necessitates the need to consider the number of neighbors per node also as a criterion to construct the CDS. In this paper, we explore the use of a weighted-density based approach to construct the CDS (referred to as wDCDS) of the underlying sensor network for every round of data communication and to construct a data gathering tree based on the wDCDS. The weighted-density of a sensor node is defined as the product of the number of neighbors of the node and the fraction of the initially supplied energy that is available for the node. The wDCDS algorithm prefers nodes with a relatively larger weighted-density for inclusion in the CDS. As illustrated in the simulation results, the wDCDS approach not only minimizes the delay and energy consumption, it also yields a larger network lifetime compared to the ECDS based approach.

In addition to the wDCDS approach, we also explore the use of a density-only approach for constructing the CDS (referred to as the DCDS). For constructing the DCDS, we prefer nodes with a larger number of neighbors over nodes with relatively less number of neighbors. Since the number of neighbors for a sensor node does not change with time (we consider the network is functional only until the time of first node failure), the DCDS remains the same for every round of communication. Hence, nodes with more neighbors are likely to get exhausted earlier, leading to a lower network lifetime. The DCDS algorithm is merely used as a baseline to illustrate how much the network lifetime for a data gathering algorithm could be improved by considering both the density as well as the energy of the nodes.

Once a CDS (based on the density, energy or the weighted-density) is determined for a particular round, the principle behind the construction of the data gathering tree is common to all the three data gathering algorithms. The Leader node is the node with the largest available energy and it is responsible for transmitting the aggregated data packet to the sink. A data gathering tree rooted at the Leader node is formed by considering only the nodes in the CDS as the intermediate nodes of the tree. The non-CDS nodes form the leaf nodes of the tree. The upstream node of an intermediate CDS node in the data gathering tree is the closest CDS node that is also relatively closer to the Leader node.

The rest of the paper is organized as follows: Section 2 reviews the well-known data gathering algorithms based on different communication structures. Section 3 presents the algorithm to construct the wDCDS and the wDCDS-DG tree. Section 4 presents an overview of the algorithm to construct the DCDS-DG and ECDS-DG trees. Section 5 presents a simulation study of the three data gathering trees and discusses the performance results observed. Section 6 concludes the paper.

2. Literature Review

We now briefly review the data gathering algorithms proposed based on different communication structures. The algorithms discussed are: LEACH (clusters), PEGASIS (chain), Improved PEGASIS for CDMA systems (trees), ECDS-DG (connected dominating set) and TTDD (grid).

2.1 Low Energy Adaptive Clustering Hierarchy (LEACH)

The basic idea behind the LEACH algorithm (Heinzelman, Chandrakasan & Balakrishnan, 2004) is that the whole network is divided into clusters and a cluster-head is used for each. A sensor selects the cluster-head closer to it and directly transmits the data to the cluster-head. The cluster-head aggregates all the data collected in its cluster and transmits the aggregated data to the sink. The role of the high-energy consuming cluster-head position is rotated among all the sensor nodes in the network. If P is the percentage of nodes that can be cluster heads, LEACH ensures that a sensor node is elected as cluster head exactly once within every $1/P$ rounds of data communication. The optimal number of clusters (i.e., the cluster-heads) that will reduce the overall energy consumption in a given system depends on several parameters, such as the network topology, the percentage of cluster heads and the relative costs of computation and communication.

2.2 Power-Efficient Gathering in Sensor Information Systems (PEGASIS)

PEGASIS (Lindsey, Raghavendra & Sivalingam, 2002) forms a chain of the sensor nodes and uses this chain as the basis for data aggregation. The chain is formed using a greedy approach, starting from the node farthest to the sink. The nearest node to this node is included as the next node in the chain. This procedure is continued until all the nodes are included in the chain. A node can be in the chain at only one position. For every round, the leader node is responsible for forwarding the aggregated data to the sink. Once the leader node is selected and notified by the sink node, each node

on both sides of the chain (with respect to the leader node), receives and transmits the aggregated data to the next node in the chain, until the data reaches the leader node.

2.3 Improved PEGASIS for CDMA Systems

PEGASIS incurs a huge delay, especially for Time Division Multiple Access (TDMA) systems, as data moves across the complete chain of sensor nodes, one node at a time, before transmitted to the sink. An improved version of PEGASIS (Lindsey, Raghavendra & Sivalingam, 2001) has been proposed for Code Division Multiple Access (CDMA) systems, where there can be simultaneous communication between any pair of nodes if each node is assigned unique CDMA code and each node knows the CDMA code for communication with every other node. The chain formed using the greedy distance-based heuristic is still used as the basis for data aggregation. A round of data aggregation and transmission is comprised of $\log N$ levels, where N is the number of nodes in the network. Each node transmits the data to a close neighbor in a given level of the hierarchy. Nodes that receive data at a given level are the only nodes that rise to the next level. In order to lower the delay, data is aggregated simultaneously using as many pairs as possible at each level.

2.4 Energy-aware Connected Dominating Set-based Data Gathering (ECDS-DG) Algorithm

The ECDS-DG algorithm (Meghanathan, 2009) for every round of data communication works as follows: The leader node for a round is the node with the largest available energy. The leader node becomes the root node of the data gathering tree. An energy-aware connected dominating set (ECDS) is constructed and the leader node is the first node to be included in the ECDS. Only nodes that have a relatively larger available energy are included in the ECDS. Every node in the network is either in the ECDS or is a neighbor of a node in the ECDS. Once the ECDS construction phase is completed, the ECDS-DG tree is constructed. The leaf nodes of the ECDS-DG tree are the non-ECDS nodes. The ECDS nodes form the intermediate nodes of the ECDS-DG tree. The upstream node of a leaf node is the closest ECDS node. The upstream node of an ECDS node i is another ECDS node j such that j is closer to i and also relatively closer to the leader node compared to i . The leader node could itself be the upstream node for certain intermediate ECDS nodes. An intermediate ECDS node aggregates data collected from all of its downstream leaf nodes and downstream ECDS nodes, if any exists, and forwards the aggregated data to its upstream ECDS node in the ECDS-DG tree. This procedure is repeated until the aggregated data reaches the leader node, which then aggregates the data received with its own data and forwards the aggregated data to the sink.

2.5 Two-Tier Data Dissemination (TTDD) Algorithm

The TTDD algorithm (Luo, Ye, Cheng, Lu & Zhang, 2005) lets each source sensor node of the data to proactively construct a grid structure such that the sensor nodes at the grid points (called dissemination nodes) forward the data from the source to the sink node. The sink node within a grid, issues a query for the data and the query is routed by the sensors within the grid to the dissemination node for the grid. The query is further propagated only by the dissemination nodes and the source now responds back through the reverse path of the dissemination nodes. Considerable overhead would be involved in establishing the grid structure for each source sensor node. The dissemination nodes at the grid points are bound to run out of battery power quickly. A variant of TTDD called the Energy Efficient Data Dissemination (EEDD) algorithm (Zhou, Xiang and Wang, 2006) divides the entire sensor field into virtual grids of size $R_{trans}/2\sqrt{2}$, where R_{trans} is the transmission range of a sensor node. Each grid has a grid head, most likely to be the node with the largest energy among the nodes in the grid. The grid heads are responsible for forwarding the data from the source node to the sink. The grid heads have to be frequently changed in order to maintain fairness for each sensor node. As a result, more latency will be incurred in propagating the data from a source to the sink.

3. Weighted-Density based Connected Dominating Set and Data Gathering Algorithm

A Connected Dominating Set (CDS) of a graph is the connected sub-graph of G such that for every vertex v in G , v is either in the CDS or is a neighbor of a node in the CDS. The problem of determining a Minimum Connected Dominating Set (MCDS) is critical for data aggregation in wireless sensor networks because a MCDS helps to aggregate data from all the nodes in the network such that each node is involved in only one transmission and reception and the data gets aggregated at a minimum number of intermediate nodes before being forwarded to the sink. The problem of determining a MCDS is NP-complete (Cormen, 2001).

3.1 Assumptions

- (a) We assume all the sensor nodes are CDMA (Code Division Multiple Access) enabled so that we can achieve parallel communication between any pair of sensor nodes as and when desired. Such an assumption has also been made in other well-known data aggregation algorithms such as LEACH and PEGASIS.
- (b) We assume that the underlying network graph that is used to obtain the CDS is a unit disk graph constructed assuming each sensor node has a fixed transmission range R . A link exists between two nodes in the unit disk

graph if and only if the physical distance between the two nodes is less than or equal to R . This assumption prevents the non-CDS leaf nodes of the data gathering tree from incurring higher transmission costs.

- (c) We assume that a sensor node can do transmission power control, if desired. In other words, a sensor node can communicate with any other node (even to nodes outside the transmission range) in the network. Transmission power control is done only by the nodes that are part of the CDS (i.e., the intermediate nodes of the data gathering tree) and not by the non-CDS leaf nodes.
- (d) For data aggregation, we assume that every upstream node uses a particular CDMA code to communicate with all its immediate downstream nodes. The upstream node broadcasts a time schedule for data transmission to all its immediate downstream nodes. A downstream node sends its data to the upstream node according to the slots provided in the time schedule. Note that such TDMA (Time Division Multiple Access) – based communication between every upstream node and its immediate downstream nodes can occur in parallel using the unique CDMA codes chosen by each of the upstream nodes.

3.2 wDCDS-DG Algorithm

The algorithm (pseudo code in Figure 1) is executed for each round of data aggregation. The weighted-density of a sensor node (equation (1)) at any time instant is the product of the number of neighbors of the sensor node and the fraction of the initial energy currently available at the node during that time instant. The first node (*Start Node* in the pseudo code of Figure 1) to be included in the wDCDS is the node with the largest weighted-density. If more than one node has the largest weighted-density, the tie is broken arbitrarily. The algorithm is executed in two stages: In the first stage, we form a CDS (called wDCDS) of the entire network by preferring nodes with relatively larger weighted-density to be part of the wDCDS. In the second stage, we form a data aggregation tree rooted at the Leader node and by involving only nodes that are part of the wDCDS. The sensor node that has the largest available energy during a round is selected as the Leader node for the round. The pseudo code of the algorithm is illustrated in Figure 1.

$$\text{Weighted-density}(u) = \# \text{Neighbors}(u) * \frac{\text{Energy}(u)}{\text{Initial-Energy}} \dots\dots\dots (1)$$

Note that the term $\text{Energy}(u)/\text{Initial-Energy}$ in the above equation represents the fraction of the initial energy currently available at node u .

We maintain four data structures:

- (i) wDCDS-List – includes all the nodes that are part of the wDCDS
- (ii) Uncovered-Nodes-List – includes all the nodes that are not covered by a node in the wDCDS-List
- (iii) Covered-Nodes-List – includes nodes that are either in the wDCDS-List or covered by a node in wDCDS-List
- (iv) Priority-Queue – includes nodes that are in the Covered-Nodes-List and are probable candidates for addition to the wDCDS-List. This list is sorted in the decreasing order of the weighted-density of the nodes. A dequeue operation on the queue returns the node with the largest weighted-density.

The wDCDS is primarily constructed as follows: The *Start Node* is the first node to be added to the wDCDS-List. As a result of this, all the neighbors of the *Start Node* are said to be covered: removed from the Uncovered-Nodes-List and added to the Covered-Nodes-List and to the Priority-Queue. If both the Uncovered-Nodes-List and the Priority-Queue are not empty, we dequeue the Priority-Queue to extract a node s that has the largest weighted-density and is not yet in the wDCDS-List. If there is at least one neighbor node u of node s that is yet to be covered, all such nodes u are removed from the Uncovered-Nodes-List and added to the Covered-Nodes-List and to the Priority-Queue; node s is also added to the wDCDS-List. If all neighbors of node s are already covered, then node s is not added to the wDCDS-List. The above procedure is repeated until the Uncovered-Nodes-List gets empty or the Priority-Queue gets empty. If the Uncovered-Nodes-List gets empty, then all the nodes in the network are covered. If the Priority-Queue gets empty and the Uncovered-Nodes-List has at least one node, then the underlying network is considered to be disconnected.

Note that the *Start Node* and the *Leader node* are selected with different criteria. The reason is that as the *Leader node* has to spend relatively larger amount energy to transmit the aggregated data to the faraway sink, it is better to rotate this role among the nodes in the wDCDS. Also, note that as the *Start Node* for every round is randomly chosen among the nodes that have the largest weighted-density, the constituent nodes of the wDCDS are more likely to be different for different rounds. Both these strategies are aimed at increasing the network lifetime.

Once the wDCDS is formed, the wDCDS-DG tree is constructed as follows: The upstream node for every non-wDCDS leaf node v is the wDCDS node that is closest to v . Once the upstream nodes for all the leaf nodes are determined, we determine the upstream-nodes for all the wDCDS nodes, except the *Leader*. The upstream node for a wDCDS node v is the closest wDCDS node u that is also relatively closer to the *Leader node*. The downstream node list of the wDCDS

node is updated as and when the node is selected as the upstream node of a leaf node or another wDCDS node. Figure 7 illustrates the working of the wDCDS-DG algorithm through an example.

3.3 Delay of the wDCDS-DG Tree

The delay per round for data aggregation in the wDCDS-DG tree is computed in terms of the maximum number of time units incurred at the Leader node to collect and aggregate data from all of its immediate downstream nodes. As communication between an upstream node and its downstream nodes in the wDCDS-DG tree occur sequentially using TDMA, every intermediate node in the tree has to wait to collect the data from all of its immediate downstream nodes before aggregating and forwarding the data to its upstream node. The overall delay (time units) for data aggregation and transmission is one plus the delay incurred at the Leader node.

4. Density-only and Energy-only based Connected Dominating Set and Data Gathering Algorithms

The Density-based Connected Dominating Set and Data Gathering (DCDS-DG) tree algorithm uses a heuristic, called the *d-MCDS* heuristic, proposed by Meghanathan and Farago (2008), to determine the Density-based Connected Dominating Set (DCDS). The pseudo code is illustrated in Figure 2. We first include in the DCDS, the *Start Node*, which is the node with the largest number of neighbors. If there is a tie, one of the contending nodes is randomly chosen and included in the DCDS. The heuristic proceeds further by only considering the nodes that have the maximum number of uncovered neighbors. Once the DCDS is formed, the Leader node is selected to be the node with the largest energy level among the nodes in the DCDS. The DCDS-DG tree is constructed similar to the procedure adopted for a wDCDS-DG tree.

The algorithm to determine the Energy-based Connected Dominating Set (ECDS) and the ECDS-based Data Gathering (ECDS-DG) tree is similar to that of the wDCDS-DG algorithm except that the Start Node is the node with the largest energy level and the Priority-Queue is maintained in the decreasing order of the energy level of the nodes. The Leader node of a round is the node with the largest available energy among the nodes in the ECDS. For more information on ECDS-DG, the interested reader is referred to (Meghanathan, 2009).

5. Simulations

We evaluated the performance wDCDS-DG, DCDS-DG and ECDS-DG in a discrete event simulator developed in Java. The DCDS-DG and wDCDS-DG algorithms are newly implemented for this research paper and we used the ECDS-DG code recently developed by its author (Meghanathan, 2009). The network considered is a square network of dimensions 100m x 100m. The location of the sink node is varied between the following three values: (0, 0), (50, 50) and (50, 300). The transmission range per node adopted for forming the unit disk graph is varied from 15m to 50m (i.e., 15% to 50% of the side of the square network considered for our simulations). Note that the transmission range limits only the distance between two neighboring nodes in the CDS and the distance between a non-CDS node and a CDS node. However, the CDS-based data aggregation tree (for all the three algorithms studied) may involve an upstream node and downstream node that need not be neighbors in the CDS. In such cases, a sensor node conducts transmission power control (i.e., vary the transmission range) depending on the distance to the receiver node. We assume all the sensor nodes are CDMA-enabled so that there can be simultaneous communication between any pair of sensor nodes.

5.1 Energy Consumption Model

The energy consumption model is the commonly used first order radio model (Rappaport, 2002) that has also been used in several previous work (Heinzelman, Chandrakasan & Balakrishnan, 2004; Lindsey, Raghavendra & Sivalingam, 2001; Lindsey, Raghavendra & Sivalingam, 2002). According to this model, the energy expended by a radio to run the transmitter or receiver circuitry is $E_{elec} = 50$ nJ/bit and $\epsilon_{amp} = 100$ pJ/bit/m² for the transmitter amplifier. The radios are turned off when a node wants to avoid receiving unintended transmissions. The energy lost in transmitting a k -bit message over a distance d is given by: $E_{TX}(k, d) = E_{elec} * k + \epsilon_{amp} * k * d^2$. The energy lost in receiving a k -bit message is $E_{RX}(k) = E_{elec} * k$. The energy lost for data fusion is 5 nJ/ bit/ message.

The energy lost per round is the sum of the energy lost at all the nodes for the transmission, reception and fusion of the data. The leaf nodes of the data gathering tree do not lose energy to receive or fuse the data, but lose energy to transmit data to their upstream CDS node. Every CDS node, including the Leader node, loses energy to receive, aggregate data from each of the intermediate downstream nodes and to forward the aggregated data to its upstream node in the tree. Note that the sink node can be considered as an upstream node of the Leader node.

5.2 Performance Metrics

The performance metrics considered are: (i) Network lifetime, measured as the number of rounds the network sustains before the first sensor node dies due to exhaustion of battery charge, (ii) Delay (in terms of the number of time units) per round of data aggregation and transmission to the sink, (iii) Energy lost per round and (iv) Energy * Delay per round. A lower value for the energy*delay per round for an algorithm indicates that the algorithm very well balances the tradeoff between energy and delay. The results reported in Figures 4 through 6 and 8 through 13 are obtained for 1000

trials of the appropriate algorithms for each value of the transmission range per node. For each trial, the initial energy supplied to every sensor node is 1J. The delay per round is independent of the sink locations, as it is a measure of only the number of time units needed for data aggregation. Hence, the results for delay per round illustrated in Figure 3 are the values obtained for all the sink locations.

5.3 Performance Results and Discussion

Overall, the following observations can be made: When the sink is located far away from the sensor network field, the wDCDS-DG algorithm significantly outperforms both the DCDS-DG and ECDS-DG algorithms for all values of the transmission range per node. When the sink is located either in the corner or in the center of the network field, the wDCDS-DG algorithm performs better than ECDS-DG up to moderate values of the transmission range. As the transmission range per node gets high, the wDCDS-DG algorithm starts performing poor. Nevertheless, the wDCDS-DG algorithm is better than the DCDS-DG algorithm for all values of the transmission range per node.

The DCDS and wDCDS based data gathering trees yield a significantly lower delay (refer Figure 3) compared to that of the ECDS based data gathering tree. As the transmission range per node increases, the difference between the delay incurred per round for a DCDS or wDCDS based data gathering tree and an ECDS based data gathering tree increases. The delay per round for a DCDS or wDCDS based data gathering tree is about 75% and 40% of the delay per round for an ECDS based data gathering tree at transmission range per node values of 15m and 50m respectively. There is no significant difference between the delay per round incurred by the DCDS and wDCDS based data gathering trees.

If the sink is located far away from the sensor network field, the wDCDS-DG tree yields the largest network lifetime (refer Figure 4) for each value of the transmission range per node. This can be also attributed to the relatively lower energy incurred per round of data communication. If the sink is located in the corner or the center of the network field (refer Figures 5 and 6), the wDCDS-DG tree still yields a larger network lifetime, but only up to moderate values of the transmission range. As the transmission range gets high, the network lifetime decreases as the energy consumed per round (refer Figures 8, 9 and 10) increases. Nevertheless, the wDCDS-DG tree incurs the lowest energy consumed per round for all the scenarios. Because of the lowest energy per round and lowest delay per round, the wDCDS-DG tree also incurs the lowest energy*delay per round (refer Figures 11, 12 and 13). This indicates that the weighted-density based approach very well balances the tradeoff between energy and delay compared to the density-only or the energy-only based approaches.

For the DCDS-DG tree, the energy consumed per round (refer Figures 8, 9 and 10) increases from low to a significantly high value as the transmission range per node increases. The density-based approach for determining the CDS has the smallest number of constituent nodes and the corresponding data gathering tree incurs the least height. Because of the involvement of less number of nodes in the data gathering tree, data transmissions from nodes located in the different regions of the network have to go through a relatively longer distance. Nodes that are part of the DCDS are repeatedly exhausted and die earlier, leading to a lower network lifetime. Also, there is a significant increase in the energy consumed per round because of the long-distance transmissions, especially when the transmission range per node gets high. The ECDS based approach incurs higher energy per round, though improves the network lifetime compared to the DCDS-based approach. As a result, most of the nodes in the network lose relatively more energy per round, resulting in premature node failures. The weighted-density based approach achieves the correct balance between the height of the tree and the number of immediate downstream nodes per intermediate node of the tree. Thus, the wDCDS-DG tree incurs the lowest energy*delay per round of data aggregation.

6. Conclusions

The high-level contribution of this paper is the development of a weighted-density based connected dominating set data gathering (wDCDS-DG) tree algorithm for wireless sensor networks. The weighted-density of a node for a round is defined as the product of the number of neighbors of the node and the fraction of the energy currently available at the node during the round. The weighted-density is used as the metric to form the wDCDS. Only nodes that have a relatively larger weighted-density are considered for inclusion in the wDCDS. The wDCDS-DG tree yields a longer network lifetime, lower delay and lower energy consumption per round when compared to a density-only CDS based data gathering (DCDS-DG) tree and an energy-only CDS based data gathering (ECDS-DG) tree. Since the wDCDS-DG tree performs better than the ECDS-DG tree, the wDCDS-DG algorithm can also be considered to perform better than the classical LEACH and PEGASIS algorithms, which are outperformed by ECDS-DG. (Meghanathan, 2009). Simulation results illustrate that when the sink is located far away from the sensor network field (which is more often the case), wDCDS-DG can yield significantly larger lifetime than ECDS-DG. If the sink is located either in the corner or in the center of the network field, then wDCDS-DG yields larger lifetime up to moderate values of transmission range per node and as the transmission range gets high, ECDS-DG starts yielding slightly larger lifetime.

References

Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C. (2001). *Introduction to Algorithms*. (2nd ed.) The MIT Press.

- Heinzelman, W., Chandrakasan, A., and Balakrishnan, H. (2004). *Energy-Efficient Communication Protocols for Wireless Microsensor Networks*. Paper presented at the Hawaaiian International Conference on Systems Science, USA.
- Lindsey, S., Raghavendra, C., and Sivalingam, K. M. (2002). Data Gathering Algorithms in Sensor Networks using Energy Metrics. *IEEE Transactions on Parallel and Distributed Systems*, 13 (9), 924-935.
- Lindsey, S., Raghavendra, C., and Sivalingam, K. M. (2002). *Data Gathering in Sensor Networks using the Energy*Delay Metric*. Paper presented at the 15th International Parallel and Distributed Processing Symposium, USA.
- Luo, H., Ye, F., Cheng, J., Lu, S., and Zhang, L. (2005). TTDD: Two-Tier Data Dissemination in Large-Scale Wireless Sensor Networks. *Wireless Networks*, 11 (1-2), 161-175.
- Meghanathan, N. (2009). *An Algorithm to Determine Energy-aware Connected Dominating Set and Data Gathering Tree for Wireless Sensor Networks*. Paper presented at the 2009 International Conference on Wireless Networks, USA.
- Meghanathan, N., and Farago, A. (2008). On the Stability of Paths, Steiner Trees and Connected Dominating Sets in Mobile Ad hoc Networks. *Elsevier Ad hoc Networks*, 6 (5), 744 – 769.
- Rappaport, T. S. (2002). *Wireless Communications: Principles and Practice*. (2nd ed.). Prentice Hall.
- Zhou, Z., Xiang, X., and Wang, X. (2006). *An Energy Efficient Data Dissemination Protocol in Wireless Sensor Networks*. Paper Presented at the International Symposium on a World of Wireless, Mobile and Multimedia Networks, USA.

Input: Graph $G = (V, E)$, where V is the vertex set and E is the edge set

Initial-energy // Initial energy supplied to each node

Auxiliary Variables and Functions:

wDCDS-List, Covered-Nodes-List, Uncovered-Nodes-List, Priority-Queue, Start Node, Leader

// Weighted-density for a node $u \in V$

$\text{weighted-density}(u) = \#Neighbors(u) * (\text{energy}(u) / \text{Initial-energy})$

$\text{StartNode} = \left\{ u \mid \text{Max}_{u \in V} [\text{weighted-density}(u)] \right\}$

Output: *wDCDS-DG* Tree Information

- *Upstream-Node*(v) for every vertex $v \in V - \{Leader\}$
- *Downstream-Nodes*(u) for every vertex $u \in V$

Initialization:

wDCDS-List = {*Start Node*}; *Priority-Queue* = {*Start Node*}; *Covered-Nodes-List* = {*Start Node*}

Uncovered-Nodes-List = $V - \{Start Node\}$

Begin *wDCDS-DG* Construction

while (*Uncovered-Nodes-List* $\neq \Phi$ and *Priority-Queue* $\neq \Phi$) **do**

 node $s = \text{Dequeue}(\text{Priority-Queue})$ // Extracts the node with the largest weighted-density

$\text{alreadyCovered} = \text{true}$ // to test whether all neighbors of node s have already been covered or not

for all node $u \in \text{Neighbors}(s)$ **do**

if ($u \in \text{Uncovered-Nodes-List}$) **then**

$\text{alreadyCovered} = \text{false}$

$\text{Uncovered-Nodes-List} = \text{Uncovered-Nodes-List} - \{u\}$

$\text{Covered-Nodes-List} = \text{Covered-Nodes-List} \cup \{u\}$

$\text{Priority-Queue} = \text{Priority-Queue} \cup \{u\}$

end if

end for

if ($\text{alreadyCovered} = \text{false}$) **then**

```

    wDCDS-List = wDCDS-List U {s}
  end if
end while

Leader = {u | Max_{u ∈ DCDS-List} [energy(u)]}
∀v ∉ wDCDSList,
Upstream-Node(v) = {ū = u | Min_{u ∈ wDCDS-List} [distance(v, u)]}
DownstreamNodes(ū) = DownstreamNodes(ū) ∪ {v}

∀v ∈ wDCDSList - {Leader},
UpstreamNode(v) = {ū = u | [Min_{u ∈ wDCDSList} [distance(v, u)]] AND [distance(v, Leader) > distance(u, Leader)]}
DownstreamNodes(ū) = DownstreamNodes(ū) ∪ {v}

End wDCDS-DG Construction

```

Figure 1. Pseudo Code for the wDCDS-DG Algorithm

Input: Graph $G = (V, E)$, where V is the vertex set and E is the edge set

Auxiliary Variables and Functions:

DCDS-List, Covered-Nodes-List, Uncovered-Nodes-List, Priority-Queue, StartNode, Leader

// *StartNode* is the node with the largest number of neighbors

Output: DCDS-DG Tree Information

- *Upstream-Node(v)* for every vertex $v \in V - \{Leader\}$
- *Downstream-Nodes(u)* for every vertex $u \in V$

Initialization:

DCDS-List = {*StartNode*}; *Priority-Queue* = {*StartNode*}; *Covered-Nodes-List* = {*StartNode*}

Uncovered-Nodes-List = $V - \{StartNode\}$

Begin DCDS-DG Construction

```

while (Uncovered-Nodes-List ≠ ∅ and Priority-Queue ≠ ∅) do
  node s = Dequeue(Priority-Queue) // Extracts the node with the largest number of neighbors that
  are not yet covered
  DCDS-List = DCDS-List U {s}
  for all node u ∈ Neighbors(s) do
    if (u ∈ Uncovered-Nodes-List) then
      Uncovered-Nodes-List = Uncovered-Nodes-List - {u}
      Covered-Nodes-List = Covered-Nodes-List U {u}
      Priority-Queue = Priority-Queue U {u}
    end if
  end for
end while

```


end while

$$Leader = \left\{ u \mid \underset{u \in DCDS-List}{Max} [energy(u)] \right\}$$

$\forall v \notin DCDSList,$

$$UpstreamNode(v) = \left\{ \bar{u} = u \mid \underset{u \in wDCDS-List}{Min} [distance(v, u)] \right\}$$

$$DownstreamNodes(\bar{u}) = DownstreamNodes(\bar{u}) \cup \{v\}$$

$\forall v \in DCDSList - \{Leader\},$

$$UpstreamNode(v) = \left\{ \bar{u} = u \mid \underset{u \in wDCDS-List}{Min} [distance(v, u)] \right\} AND [distance(v, Leader) > distance(u, Leader)]$$

$$DownstreamNodes(\bar{u}) = DownstreamNodes(\bar{u}) \cup \{v\}$$

End DCDS-DG Construction

Figure 2. Pseudo Code for the DCDS-DG Algorithm

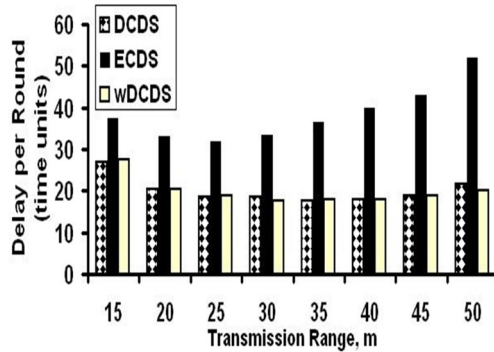


Figure 3. Average Delay per Round

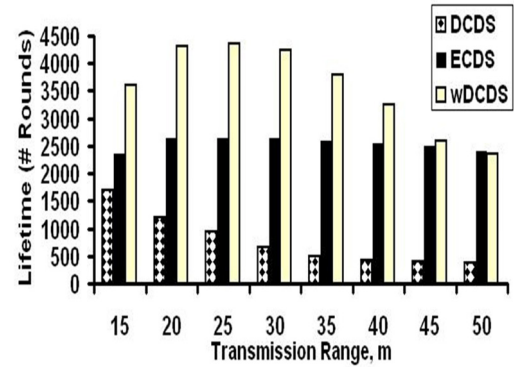


Figure 4. Lifetime per Round [Sink at (50, 300)]

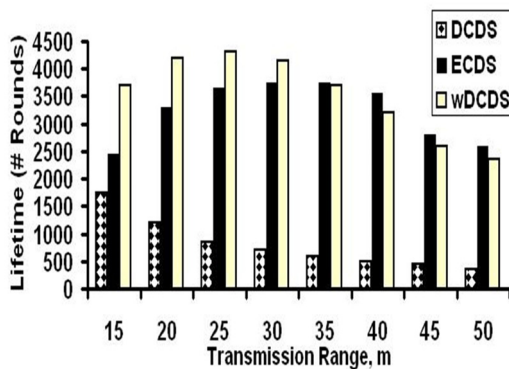


Figure 5. Lifetime per Round [Sink at (50, 50)]

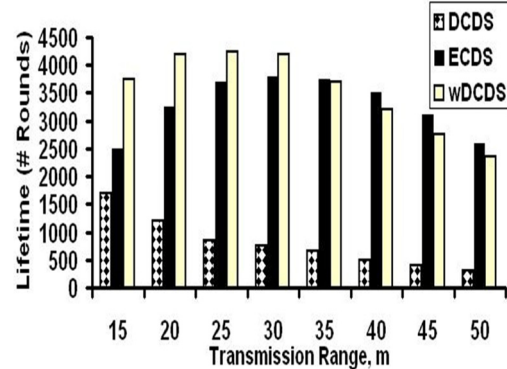
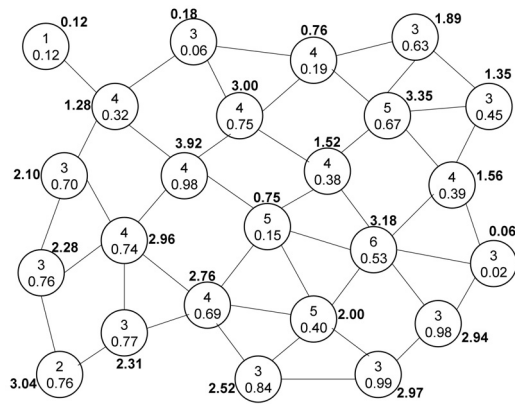
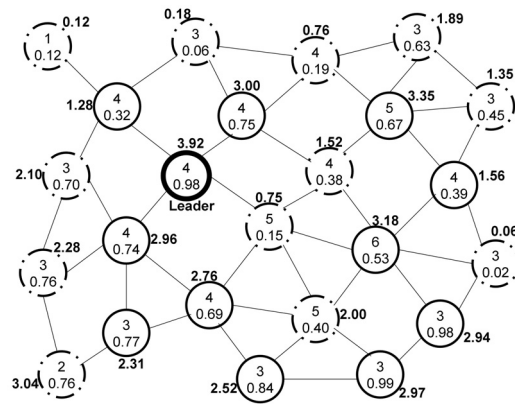


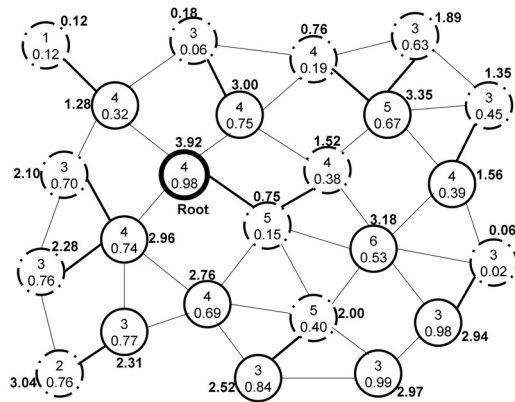
Figure 6. Lifetime per Round [Sink at (0, 0)]



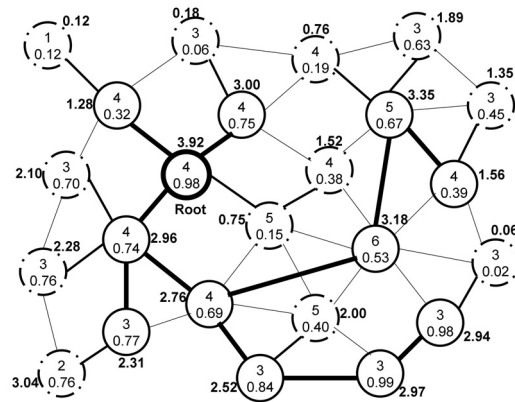
(Initial Network, Before ECDS Construction)



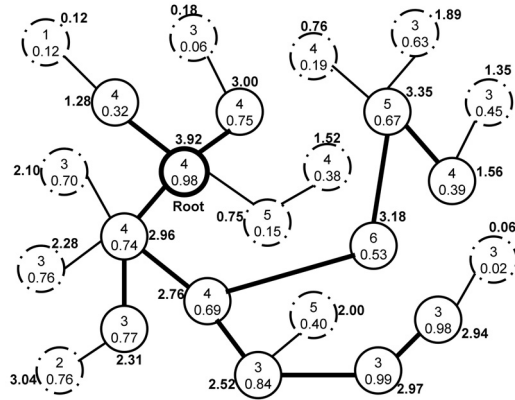
(After ECDS Construction)



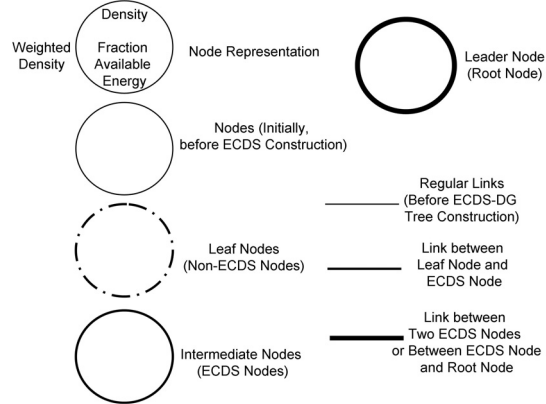
(Formation of Links between Leaf Nodes and ECDS Nodes)



(Formation of Links between Two ECDS Nodes or Between an ECDS Node and the Root Node)



ECDS-DG Tree



Legend for the Figures

Figure 7. Example for the Execution of the wDCDS-DG Algorithm

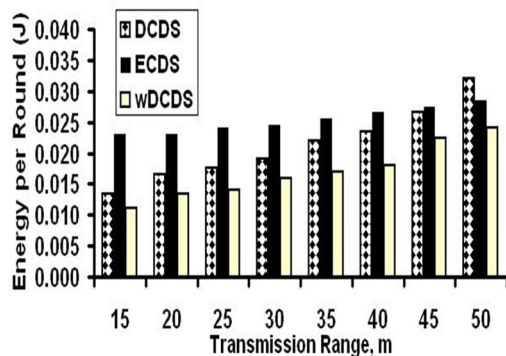


Figure 8. Energy Consumed per Round
[Sink at (50, 300)]

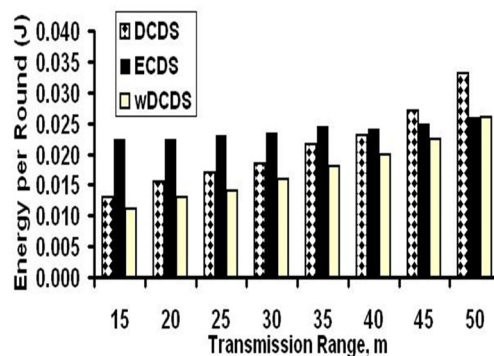


Figure 9. Energy Consumed per Round
[Sink at (50, 50)]

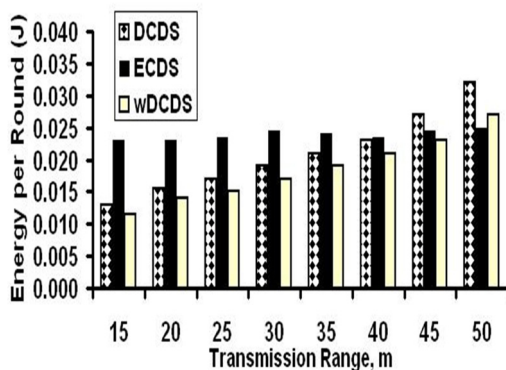


Figure 10. Energy Consumed per Round
[Sink at (0, 0)]

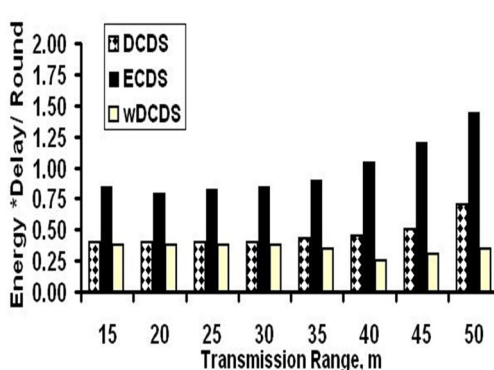


Figure 11. Energy * Delay per Round
[Sink at (50, 300)]

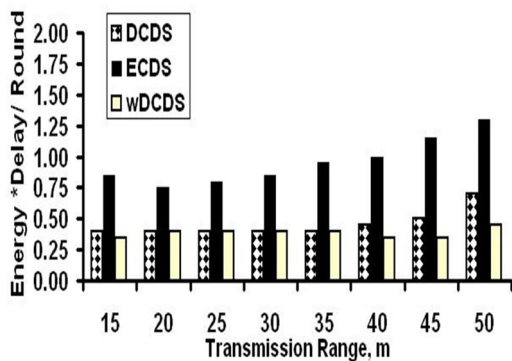


Figure 12. Energy * Delay per Round
[Sink at (50, 50)]

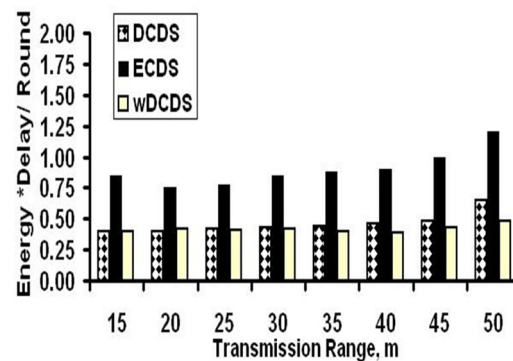


Figure 13. Energy * Delay per Round
[Sink at (0, 0)]