01 Apr 2010

Energy Constrained Dominating Set for Clustering in Wireless Sensor Networks

Julia Albath

Mayur Thakur

Sanjay Kumar Madria
Missouri University of Science and Technology, madrias@mst.edu

Recommended Citation

J. Albath et al., "Energy Constrained Dominating Set for Clustering in Wireless Sensor Networks," *Proceedings of the 2010 24th IEEE International Conference on Advanced Information Networking and Applications (AINA)*, Institute of Electrical and Electronics Engineers (IEEE), Apr 2010.
The definitive version is available at https://doi.org/10.1109/AINA.2010.14

Energy Constrained Dominating Set for Clustering in Wireless Sensor Networks

Julia Albath
Missouri University of Science and Technology
Department of Computer Science
Rolla, MO
julia.albath@acm.org

Mayur Thakur
Google Inc.
Mountain View, CA
mayurthakur@google.com

Sanjay Madria
Missouri University of Science and Technology
Department of Computer Science
Rolla, MO
madrias@mst.edu

***Abstract*—Using partitioning in wireless sensor networks to create clusters for routing, data management, and other protocols has been proven as a way to ensure scalability and to deal with sensor network shortcomings such as limited communication ranges and energy. Choosing a cluster head within each cluster is important because cluster heads use additional energy for their responsibilities and that burden needs to be carefully passed around. Many existing protocols either choose cluster heads randomly or use nodes with the highest remaining energy. We introduce the energy constrained minimum dominating set (ECDS) to model the problem of optimally choosing cluster heads with energy constraints. We propose a distributed algorithm for the constrained dominating set which runs in $\mathcal{O}(\log n \log \Delta)$ rounds with high probability. We experimentally show that the distributed algorithm performs well in terms of energy usage, node lifetime, and clustering time and, thus, is very suitable for wireless sensor networks.**

I. Introduction

A wireless sensor network consists of a large number of small sensors with low-power transceivers. These sensors are an effective tool for gathering data for a variety of purposes, such as border protection, surveillance of forests for fire, and tracking of animal movements. The data collected by each sensor is communicated via a multi-hop path in the network to a single processing center, the base station. The base station uses all reported data to determine the characteristics of the environment or detect an event.

Communication via the on-board radio is the most expensive operation of the sensor nodes [1]. In radio communications, the signal strength decreases proportional to the square of the propagation distance [2]. In other words, to have the same signal strength reach twice the distance, four times the amount of energy is required. Protocols such as LEACH [3], and those described in [4] and [5] reduce energy consumption and increase the lifetime of the network. The basic idea in these protocols is to cluster sensors into groups and to choose a cluster head such that sensors communicate only to their cluster head. The cluster heads then communicate the aggregated information to the processing center. Clustering has been shown to greatly reduce power consumption, is easily scalable, and is robust in face of node failures [3]. A good clustering scheme takes into account one or more of the following: communication range, number and type of sensors, geographical location, and remaining energy [6]. Clustering and proper cluster head selection in order to maximize the lifetime of the network are important considerations when designing protocols and algorithms for sensor networks [7].

A sensor network can be expressed as a graph $G = (V, E)$, where each of the vertices represents a sensor node and there is an edge between two vertices if their corresponding sensor nodes are within each other's communication range. A dominating set of a graph $G = (V, E)$ is a subset $V' \subseteq V$ such that each $x \in V - V'$ has a neighbor in V'. The assignment of nodes to cluster heads is often modeled as a dominating set (DS) problem [8]. The minimum dominating set problem is NP-complete for general graphs [9] and remains NP-complete for planar graphs, unit disk graphs, bi-partite graphs, and chordal graphs, but it does admit a Polynomial Time Approximation Scheme (PTAS) for planar graphs and unit disk graphs [10]. The dominating set problem models the optimization problem of finding a small number of cluster heads.

Clustering in sensor networks and in Mobile Ad-Hoc Networks (MANET) benefits from using a dominating set approach. The dominating set approach leads to better clustering because dominating set based clustering can be executed in a constant number of rounds [11]. A DS based approach works because every node in the network is either a dominating node or is only one hop from a dominating node [12]. Single-hop communications within clusters is appropriate because most nodes will be close to their cluster head and their links are of good quality [13].

Cluster heads spend additional energy on message transmission, so a small set of cluster heads might not be optimal from a network survivability standpoint. For instance, using a dominating set as the set q of cluster heads comes with the disadvantage that the network might lose a few cluster heads and become fragmented fairly soon. Consider the graph shown in Figure 1. Each node starts with the same amount of energy (7 units) (Figure 1(a)) and let's assume that one unit is used for each receive or send and the nodes in the dominating set combine all received data into one outgoing message. The optimal dominating set is one node (Figure 1(b)), but the network becomes disconnected after only one time step. On the other hand a slightly non-optimal dominating set using the heuristic "Don't give a cluster head more than three nodes"

1550-445X/10 $26.00 © 2010 IEEE
DOI 10.1109/AINA.2010.14

results in a network that survives two time steps as shown in Figures 1(c) and 1(d) (the shaded nodes represent the cluster heads).

A wireless sensor has many constraints, energy being one of the important ones. Other constraints include bandwidth, storage and computational abilities. A wireless sensor network needs to consider these and other constraints when choosing cluster heads and assigning nodes to clusters. For example, each node sends one packet per round to its cluster head. The length of a round is limited, this limits the number of nodes a cluster head can support. Additionally, a cluster head has to store received messages until they are combined at the end of each round, this also limits how many nodes a cluster head can support. For this work, we have chosen to concentrate on the limited energy available to each sensor and the natural limitations that puts on the size of each cluster.

Motivated by the above examples, we introduce the energy constrained minimum dominating set of a graph in order to achieve these objectives. The contributions of this paper are summarized as follows:

- We introduce and define the Energy Constrained Dominating Set (ECDS) Problem in Section II.
- We describe the problem of developing clusters so that the energy consumption during each round of processing is minimized without exhausting the available resources of any given node. In the energy constrained DS problem, we are given integer constraints on each node that denote the maximum number of relay links a node can handle if it is chosen as a cluster head. The objective is to minimize the size of the cluster head set subject to the constraint that no node has more work than it can handle and that every node is either in the constrained DS or one hop from such a node. Since the clustering algorithm typically runs repeatedly (for example, when nodes move or cluster heads die), we give a practical distributed algorithm in Section IV.
- We prove that the distributed algorithm runs in $\mathcal{O}(\log n \log \Delta)$ rounds with high probability, where Δ is the maximum degree of a node in the graph. We provide the proof in Section V.
- We support our theoretical analysis with extensive simulations using TOSSIM. We compared the performance of the distributed ECDS algorithm to the HEED algorithm [14]. HEED selects cluster heads according to residual energy and with node proximity to neighbors or node degree. Our protocol uses local information about the connectivity of each node and the connectivity of its neighbors in addition to the residual energy to decide which node should become a cluster head. ECDS takes less time and fewer rounds to cluster the network, allowing more messages to reach the base station. For the scenarios in our study, ECDS clustering takes 3.5 rounds or less, compared to 4.5 rounds or less for HEED. The number of cluster heads is as expected and the number of nodes in each cluster remains steady. Our algorithm results in very few single node clusters. The number of cluster heads, the size of the clusters and the number of clusters which contain only the cluster head are much better in ECDS then in HEED. The lifetime of the sensor network, measured as time of first node death and time of last node death, is better in ECDS than in HEED. While the overall energy consumption is slightly higher for ECDS, when considering that ECDS produces more useful data, the energy consumption per message is much lower. The results of said simulation are available in Section VI.

II. Definitions and Notations

This section describes the notations used in the rest of the paper and defines the dominating set, network clustering, and energy constrained connected dominating set.

Definition 2.1: For a graph G and a subset S of the vertex set $V(G)$, denote by $N_G[S]$ the set of vertices in G which are in S or adjacent to a vertex in S. If $N_G[S] = V(G)$, then S is said to be a *dominating set of* G.

Definition 2.2: Given an undirected graph $G = (V, E)$, and, for each $v_i \in V(G)$, a constraint $r(v_i) \in \mathbb{N}$, the energy-constrained dominating set (ECDS) of G is a pair (S, C), where C is an assignment from $x \in S$ to $V_x \subseteq V$ such that (a) $\{V_x \mid x \in S\}$ is a partition of V, (b) for each $x \in S$, $x \in V_x \subseteq N_G[\{x\}]$, and (c) for each $x \in S$, $||V_x|| \leq r(x) + 1$. In the definition of ECDS, we assume that when a node is selected as a cluster head, it includes itself in the cluster. (See part (b) of the definition.) Also note from the "+1" in condition (c) that we allow a node to cover itself for free. That is, the constraint $r(x)$ for x denotes the maximum number of nodes that x can cover in addition to itself.

ECDS is also related to, but different from, the Network Clustering problem [15]. ECDS has a constraint parameter that is not present in Network Clustering. Also, the clusters must form a partition in ECDS, where as they may overlap in the Network Clustering problem. The general dominating set can be described as a constrained dominating where each constraint is equal to n, the number of nodes in the graph. It trivially follows that the constrained minimum dominating set is NP-complete. The minimum dominating set in the general form has approximation algorithms of within $1 + \log \|V\|$. Since the constrained dominating set problem is a special case of the general dominating set problem, no improvement on these bounds will be possible.

In Wireless Sensor Networks (WSN) coverage generally means ensuring that the entire area has proper sensor distribution to ensure even sensing. In this work, we define coverage as one node's ability to handle the relaying of messages to and from other nodes in its cluster.

III. Related Work

In [16], cluster heads are chosen so that the energy consumption over the entire network is even, ensuring that the network lives as long as possible. A node will chose a cluster head to ensure the overall energy consumption in the entire

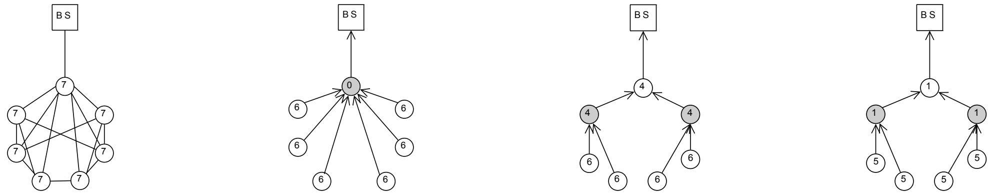

Fig. 1. Energy constraint clustering example: 1(a) Original Network 1(b) Without constraints after one round 1(c) With constraints after one round,1(d) With constraints after two rounds

network is even. Our algorithm, on the other hand, requires only local information about the topology and residual energy.

In [17] each vertex is assigned a weight, as well as a capacity, and the goal is to minimize the sum of the weights without exceeding the capacity of any vertex. The authors provide a $(2+\epsilon)\,OPT$ approximation algorithm.

A randomized distributed algorithm that runs in $O(\log n \log \Delta + 1)$ rounds and where the size of the dominating set obtained is, with high probability, within $O(\log n)$ of the optimal, is presented in [18]. Our distributed algorithm is based on this algorithm and extends its ideas to vertices with constraints and applies it to wireless sensor networks.

A fast, distributed algorithm is presented in [19]. It is used to compute a small k-dominating set D (for any fixed k) and its induced graph partition. (which breaks the graph into radius k clusters centered around the vertices of D). The time complexity of the algorithm is $O(k\ \log^* n)$, where $\log^*$ is the inverse Ackermann function.

In [20], a series of approximation algorithms for finding a small, weakly-connected dominating set (WCDS) in a given graph is presented for use in clustering mobile ad hoc networks. The main contribution of the work is a completely distributed algorithm for finding small WCDS. Our work focuses on wireless sensor networks and creates connected dominating sets.

The connected minimum dominating set is considered in [21]. The authors provide two approximation algorithms which achieve approximation factors of $2H(\Delta)+2$ and $H(\Delta)+2$ where Δ is the maximum degree in the graph and H is the harmonic function.

IV. Distributed Algorithm

This algorithm is a modification of the local randomized greedy (LRG) algorithm from [18]. The LRG algorithm is a modification of the distributed version of the greedy algorithm of [22]. To enable comparison, we first informally describe the LRG algorithm and then describe our modification to the algorithm.

The LRG algorithm proceeds in rounds. At the start of a round, each node that is not already in the dominating set computes whether it wants to be a candidate dominator in that round. Candidacy is determined by letting only nodes that can cover a large number of nodes be candidates. These nodes must have a large number of neighbors remaining. Note that a node can be covered by multiple candidates. A node defines its *support* as the number of candidates that cover it. Each node is selected for the dominating set with a probability that is the inverse of the median of supports of all nodes that it covers. Once a node is selected, its neighbors are considered “covered”. The round has ended; if uncovered nodes remain, another round starts.

The reason for using the median of supports follows: If we pick all the candidates, then we might pick too many nodes for the dominating set. If we pick only one candidate, then we may require too many rounds.

We will now describe our modification of the LRG algorithm and show an effective $(\mathcal{O}(\log n \log \Delta))$ randomized distributed algorithm for ECDS. In this version, the algorithm is required to obey the constraints *in expectation*. In particular, for each node u, $E[\#$ nodes covered by $u \mid u$ is selected as a cluster head$] \leq r(u)$. In fact, our algorithm obeys the constraint in expectation in an even stricter sense. Note that in the above formulation, it is possible for certain sets of cluster heads to grossly violate the constraints. For example, the above constraint allows an algorithm to have the following behavior: Whenever the algorithm outputs the set $\{v_1\}$ as the cluster head, all the constraints are violated grossly. Our algorithm does not have this undesirable behavior. In fact, our algorithm obeys the following: Let u be an arbitrary node and let $U \subseteq V - \{u\}$. Then, $E[\#$ nodes covered by $u | U \bigcup \{u\}$ is selected as a cluster head$] \leq r(u)$. This basically says that the nodes obey the constraints in expectation independent of one another.

The following issues must be handled:

1) What is the support of a node?
2) Given that we select a node x to be a dominator, how do we select which nodes to cover/dominate from among the neighbors of x.

To address issue 1, we say that the *constrained span* $c(x)$ of a node x at a given step in our algorithm is the smaller of the following two quantities: the number of uncovered neighbors of x and the constraint of x. (The set of neighbors of a node x includes x and all nodes with which x shares a high quality communication link/edge). Let x be a candidate and let y be a node that is adjacent to x. The *out-support* $s_{out}(x)$ of x is the ratio of the constrained span $c(x)$ to the number of uncovered neighbors of x. For example, if a candidate x has 5 uncovered neighbors and the constraint of x is 3, then $c(x) = 3$, $s_{out}(x) = 3/5$. The out-support of x is the fractional support a node gives to each of its neighbors. The *in-support*

$s_{in}(y)$ of y is the sum of the out-supports of each neighbor of y. Thus, the in-support of a node is the total support a node will get if all its neighbors are dominators and each gives fractional support to all the neighbors. Roughly speaking, the larger a node's in-support, the larger the probability that it will be covered in a randomly chosen dominating set.

How should we decide which nodes to select as dominators? Certainly, selecting all nodes would be overkill. Consider a node x whose neighbors y_1, y_2, $\ldots$, y_k have in-supports (in increasing order) $s_{in}(1) \leq s_{in}(2) \leq \ldots \leq s_{in}(k)$. Clearly, y_k "needs" node x as a dominator at most as much as y_{k-1} needs x because $s_{in}(k-1) \leq s_{in}(k)$. Similarly, y_{k-1} needs x at most as much as y_{k-2} needs x, and so on. Thus, to decide whether we want to select x as a dominator (2 above), we use the inverse of the median of $s_{in}(i)$'s. More specifically, we select a candidate x with probability equal to the inverse of the median of the in-supports of the neighbors of x.

The complete algorithm, the weighted local randomized greedy (WLRG) algorithm, is described in Algorithm 2.

INPUT: Graph $G = (V, E)$, constraint $r(v_i)$ on vertices
OUTPUT: Subset $D \subseteq V$, set of currently chosen vertices

1. **Span calculation:** Compute the constrained span $c(x)$ by computing the minimum of the constraint and the number of uncovered neighbors of x. Also, compute $\hat{c}(x)$, the rounded constrained span as the smallest power of 2 that is at least as much as $c(x)$.
2. **Candidate selection:** Compute whether $\hat{c}(x)$ is at least as much as the rounded constrained span of each node within a distance of 2 from x. If so, x is a candidate.
3. **Constrained out-support calculation:** If x is a candidate, compute the constrained out-support of x as follows: If $c(x) = 0$, let $s_{out}(x) = 0$. Else,
$$s_{out}(x) = \frac{c(x)}{||N(x) - C||}.$$
Note that $||N(x) - C||$ is the number of uncovered neighbors of x.
4. **Constrained in-support calculation:** If x is an uncovered node, let $A(x)$ be the set of neighbors of x that are candidates. Compute the constrained in-support $s_{in}(x)$ of x as
$$s_{in}(x) = \sum_{y \in A(x)} s_{out}(y).$$
5. **Dominator selection:** If x is a candidate, find the median m of $\{s_{in}(y) \mid y \in N(x) - C\}$. Let $p = 1/m$. With probability p, add x to D.
6. **Neighbor selection:** If x is selected, add x to D, and for each neighbor $y \in N(x) - C$, select y with probability $s_{out}(x)$ and add it to V_x. Set $C = \bigcup_{x \in D} V_x$.
7. Go to the next round.

Fig. 2. Weighted Local Randomized Greedy Algorithm

Explanatory notes on the algorithm: Let $D = C = \emptyset$. D will denote the set of nodes selected to be in the dominating set. C will denote the set of nodes already covered by the dominators. Also, the set of neighbors of x with which x shares a good communication link are determined using the received signal strength indicator(RSSI). RSSI is inversely proportional to the signal strength. This allows nodes to communicate only with other nodes to which there is a strong connection. Fewer retransmissions will be required to achieve a successful transmission over such links. While there is only a weak correlation between RSSI and node distance, the link quality does impact the amount of energy required for communications.

- An intuitive way to think about $s_{in}(y)$ is the following: Suppose all candidate nodes were made dominators. Suppose also that each dominator x selected $c(x)$ neighbors—the maximum number of nodes that x can dominate—at random from its neighbors. Then, $s_{in}(x)$ is the expected number of dominators that cover the uncovered node x.
- A candidate whose uncovered neighbors all have large s_{in}'s intuitively need not be selected as a dominator, because its neighbors will likely get covered by *other* nodes. On the other hand, if we only select very few dominators, then the algorithm will run for many rounds. This is the intuition for selecting a dominator with probability equal to the median of the inverse of s_{in}'s.

We can show the the algorithm described above (with a slight modification) returns a dominating set that obeys the constraints with high probability (whp). The number of rounds is $\mathcal{O}(\log n \log \Delta)$ (Δ is the maximum constrained degree) whp.

V. Analysis of the Distributed Algorithm WLRG

WLRG (Weighted Local Randomized Greedy) is described in Section IV. We now show that WLRG terminates in $\mathcal{O}(\log n \log \Delta)$ rounds with high probability.

Theorem 5.1: WLRG on a graph $G = (V, E)$ terminates in $\mathcal{O}(\log n \log \Delta)$ where n is the number of nodes and Δ is $\max\{\min(t(v), d(v)) \mid v \in V\}$, where $t(v)$ is the constraint on v and $d(v)$ is the degree of v.

We will now give the proof of this result. The structure of this proof closely follows the analysis of LRG [18]. In fact, since ECDS is a generalization of the dominating set problem, WLRG is a generalization of LRG. The key difference between our analysis and the analysis of LRG is that (a) we need a notion of *partial coverage* and (b) we need to incorporate in our analysis the neighbor selection step, a step that is not present in the LRG algorithm.

Let $G = (V, E)$ be the sensor node graph. In the proof, we will focus on a round (say the ith round) of WLRG. Let C be the set of nodes covered in an earlier round. Let $H = (V', E')$ be the subgraph of G such that V' is the union of all candidate nodes X (as defined by the candidate selection step) and all uncovered nodes Y adjacent to some $x \in X$, and E' consists of edges $(u, v) \in E$ where u is a candidate and v is an uncovered node.

Lemma 5.2: (Equivalent to Lemma 3.1 of [18].) All candidates in a connected component of H have the same rounded span.

Proof: Let v_1 and v_2 be two candidates in a connected component of H. Consider a path p from v_1 to v_2 in H. Then there cannot be two consecutive nodes in p such that both are non-candidates. (This is because at least one endpoint of each edge in H is a candidate node.) Since any two candidates within a distance of 2 must have the same rounded span, we have that all candidates that lie on p have the same rounded span. And it follows that all candidates in a connected component of H have the same rounded span. ■

We will now show using a potential function argument that WLRG terminates in $\mathcal{O}(\log n \log \Delta)$ rounds with high probability. We define the potential at the start of a round as follows: Let m be the maximum rounded span of any node at the start of a round. Define Φ as

$$\Phi = \sum_{v:\hat{c}(v)=m} c(v).$$

Lemma 5.3: (Equivalent to Lemma 3.2 of [18].) Let Φ_i and Φ'_i be the potentials at the beginning and end of round i. There is a $d > 0$ such that $E[\Phi'_i] \leq d\Phi_i$.

Note that the potential at the start of round $i+1$ might not be the same as the potential at the end of round i because the underlying graph changes due to some nodes being covered in round i.

Recall that X is the set of candidates. For each candidate v, let $U(v)$ denote the set of uncovered neighbors of v. Sort the elements of $U(v)$ in nonincreasing order of their in-supports $s_{in}()$'s. Let $T(v)$ (respectively, $B(v)$) denote the set of the first $\lceil ||U(v)||/2 \rceil$ (last $\lceil ||U(v)||/2 \rceil$) elements of $U(v)$. For a candidate v and a node $u \in U(v)$, we say that v is a *top dominator* for u if $u \in T(v)$. The probability that a top dominator v of u is selected is $1/m$, where m is the median of $\{s_{in}(y) \mid y \in U(v)\}$. Since $u \in T(v)$, $1/m \geq 1/s_{in}(u)$.

For an uncovered node u in H, we say that u is a *top heavy* node if at least $s_{in}(u)/4$ of its in-support comes from candidates that are top dominators for u. An uncovered node is *bottom heavy* if it is not top heavy.

Lemma 5.4: If u is top heavy, then the probability that u is covered in this round by a top dominator of u is at least $1 - e^{-1/4}$.

Proof: Let $P_c(u)$ be the probability that u is covered in this round by a top dominator. Then, the probability that u is not covered in this round by a top dominator is $1 - P_c(u)$. Since u is not covered if none of the top dominators adjacent to u cover u, we can write this probability as:

$$\Pi_{v \in X: u \in T(v)} P[u \text{ is not covered by } v].$$

We will upper bound this term.

Let $P_d(v)$ be the probability that v is picked to be a dominator in this round. If u is not covered by v, then exactly one of the following events happen:

- v is not picked to be a dominator (with probability $1 - P_d(v)$ or
- v is picked to be a dominator (with probability $P_d(v)$) and yet v does not cover u (with probability $1 - s_{out}(v)$)

Thus,

$$P[u \text{ is not covered by } v] = (1 - P_d(v)) + P_d(v)(1 - s_{out}(v)),$$

which simplifies to $1 - P_d(v)s_{out}(v)$. As shown above, if $u \in T(v)$, then $P_d(v) \geq 1/s_{in}(u)$. Thus,

$$\Pi_{v \in X: u \in T(v)}(1 - P_d(v)s_{out}(v)) \leq \Pi_{v \in X: u \in T(v)}(1 - \frac{s_{out}(v)}{s_{in}(u)}).$$

Define $x_v = \frac{s_{out}(v)}{s_{in}(u)}$.

Note that since u is top heavy, it follows from definition, that

$$\Sigma_{v \in X: u \in T(v)} s_{out}(v) \geq \frac{s_{in}(u)}{4}.$$

Thus,

$$\Sigma_{v \in X: u \in T(v)} x_v \geq \frac{1}{4}.$$

Let there be n elements in the set $\{v \in X \mid u \in T(v)\}$.

$$\Pi_{v \in X: u \in T(v)}(1 - x_v) \leq (1 - \frac{1}{4n})^n \leq e^{1/4}.$$

Since $1 - P_c(u) \leq e^{1/4}$, it follows that $P_c(u) \geq 1 - e^{1/4}$. ■

Consider an arbitrary edge $(v, u) \in E'$. (Recall that E' is the set of edges (v, u) in H such that v is a candidate and u is an uncovered node.) This edge can be one of four types:

1) v is a top dominator for u and u is top heavy (call this set of edges E_{tt}),
2) v is a top dominator for u and u is bottom heavy (call this set of edges E_{tb}),
3) v is a bottom dominator for u and u is top heavy (call this set of edges E_{bt}), or
4) v is a bottom dominator for u and u is bottom heavy (call this set of edges E_{bb}).

Let $S_{tt} = \sum_{(v,u) \in E_{tt}} s_{out}(v)$. Similarly, define S_{tb}, S_{bt}, and S_{bb}. Let S be the sum, over all edges (v, u) such that v is a candidate and u is an uncovered node in H, of $s_{out}(v)$.

Note that $E_{tt} \cap E_{bt}$ or $E_{bt} \cap E_{bb}$ might not be empty because a node v can be both a top and a bottom dominator for a node u. Certainly, though, $E_{tt} \cap E_{tb} = E_{bt} \cap E_{bb} = \emptyset$.

Lemma 5.5: (equivalent to Lemma 3.4 of [18].) Let S_{tt} and S be as defined above. Then,

$$S_{tt} \geq (1/3)S.$$

Proof: Consider a bottom heavy node u.

$$\sum_{v \in X: u \in B(v)} s_{out}(v) < \frac{s_{in}(u)}{4}.$$

Thus,

$$\sum_{v \in X: u \in B(v)} s_{out}(v) \geq \frac{3s_{in}(u)}{4}.$$

Thus,

$$\sum_{v \in X: u \in B(v)} s_{out}(v) > 3 \sum_{v \in X: u \in T(v)} s_{out}(v).$$

If we sum both sides of the above inequality over all bottom heavy nodes, we have that $S_{bb} \geq 3S_{tb}$. We also know that $S_{bb} \leq (1/2)S$. Thus, $S_{tb} \leq (1/6)S$. Now,

$$S_{tt} + S_{tb} \geq (1/2)S.$$

Thus, $S_{tt} \geq (1/2 - 1/6)S = (1/3)S$. ■

We can now use these results to prove Lemma 5.3 and also Theorem 5.1 in exactly the same manner as [18]. The only difference between the two proofs is that in our proof Δ is $\max\{\min(t(v), d(v)) \mid v \in V\}$, a global upper bound on the constrained span for any node in any round, while in [18] Δ is the maximum degree of the graph, also a global upper bound on the span of a node in the graph.

VI. Experiments

In order to test the distributed clustering algorithm, we implemented the algorithm in TinyOS and ran simulations in TOSSIM [23]. We compared the ECDS algorithm against the HEED algorithm [7] and used a random topology for each simulation. We choose HEED for comparison because it has been proven as a reliable algorithm, which can be implemented in a WSN. We ran a simulated 15 minutes for network sizes of 30, 45, 60, and 75 nodes. We set the initial energy and the cost per action in such a way that a 15 minute simulation would provide adequate data for analysis. A simulation of 15 minutes with 75 nodes runs up to 24 hours, thus we limited ourselves to short simulated time frames. Our algorithm is independent of the routing protocol used, but for our experiments we use the Surge multi-hop application that is part of TinyOS. HEED also uses the Surge multi-hop routing protocol. Each node generates a reading every 20 seconds. The cluster heads aggregate the readings. Surge uses a link estimation and parent selection (LEPS) mechanism to determine multi-hop routes. All traffic received at each node is monitored and used to update the internal neighbor table. The neighbor table tracks all neighbors and selects the next hop based on shortest path semantics. The default destination is the base station.

We use a credit-point system for updating the mote energy budget as used with iHEED [24]. ECDS and HEED use energy for tasks such as sending and receiving and points are deducted proportionally to the actual amount of energy used. Each node starts with the same amount of points and for each send/receive an amount proportional to the size of the message is deducted. For the simulation, we set parameters corresponding to the initial energy and the cost per action. In an implementation, this data should be read from the sensors hardware. In our implementation, cluster heads receive many more messages from nodes in their cluster than it sends messages to nodes in its cluster. All messages are sent with the same power level, therefore we do not consider the distance when determining the cost of each send/receive [25]. For ECDS, the initial energy allows a constraint of 20. During each 15 minute simulation, periodic re-clustering was performed. Whenever the network re-clusters, the constraint is updated and is based on the energy available at each node. For each network size, the experiments were repeated 30 times.

We measured the size of the dominating set and compared it to the expected size of the dominating set for each round, which allowed us to show that the algorithm performs as expected. We measured the number of rounds the algorithm executed until the entire network was clustered. We compared the time of the first node's deaths to the last node's death. Having all nodes die at approximately the same time provides the most useful WSN. Additionally, we measured the time it took for the entire network to cluster. A fast clustering algorithm ensures a useful WSN.

A. Cluster Generation

In a distributed environment it is important to evaluate how long it takes for a clustering protocol to finish. There are two measurements for WSN : time and the number of rounds of execution. Figure 3(a) shows the average number of rounds to cluster the network for various sizes. An ideal distributed clustering algorithm will cluster in a constant number of rounds. Both the ECDS and the HEED algorithm execute in a constant number of rounds, but the ECDS algorithm finished in fewer rounds. The algorithm depends on the routing information obtained from the (independent) routing protocol. This routing information may not be complete, especially in the earlier rounds. Incomplete routing information will exclude some nodes from joining a cluster at each round. Similar behavior can be seen in Figure 3(b), which shows the average time it took for the networks to cluster. Clearly, the number of rounds and the time are related and both are important measurements. An algorithm that runs over several short rounds may still outperform an algorithm that runs in a constant number of long rounds. Again, it is important that an algorithm takes a constant amount of time, no matter the size of the network. Both the ECDS and the HEED algorithm take a constant amount of time, but the ECDS algorithm is faster.

B. Cluster Goodness

Our algorithm uses a randomized, probabilistic approach. At each round, the sum of the probabilities is equal to the number of expected cluster heads. Figure 3(c) shows the average expected number of cluster heads versus the average actual number of cluster heads for each network size. For all networks, the average number of expected cluster heads is close to the average actual number of cluster heads, indicating that our algorithm performs as expected. Figure 4(a) shows the average size of the dominating set. The dominating set is the number of cluster heads selected for each simulation run. Each node starts with the same amount of energy, an amount that can support up to 20 nodes in a cluster. Another important consideration is the number of nodes assigned to each cluster. Scalability is improved when clusters are of similar size regardless of network size. Figure 4(b) shows the average number of nodes in each cluster. In ECDS the number of nodes assigned to a cluster remains relatively constant, while the size decreases for HEED. Not only the number of nodes in each cluster and the number of clusters matter, but also how many of those clusters are single-node

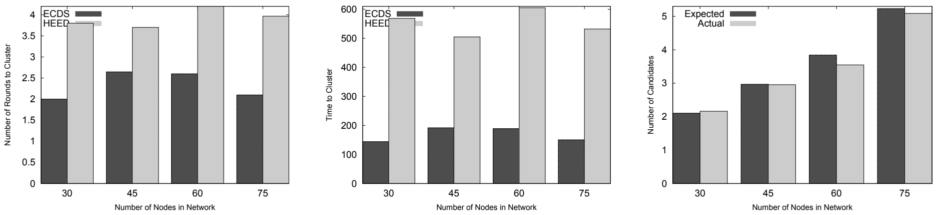

Fig. 3. 3(a) Rounds to Cluster 3(b) Time to Cluster 3(c) Expected vs. Actual Number of Cluster Heads

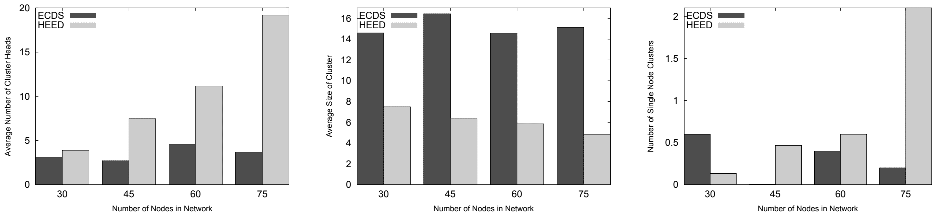

Fig. 4. 4(a) Dominating Set Size 4(b) Cluster Size 4(c) Average Number of Single Node Clusters

clusters (clusters in which the cluster head is the only node). A single node cluster does not improve performance, but they are unavoidable. A good algorithm will minimize the number of such clusters. Figure 4(c) shows the average number of single node clusters for ECDS and HEED. For ECDS the number of single node clusters decreases as the size of the network grows. ECDS chooses only neighbors which are "near" as cluster heads, some nodes will not be near a cluster head and thus create single node clusters. As the network grows, each node has more opportunities at finding a near cluster head, hence the decrease. On the other hand, HEED's single node clusters increase in number as the network grows.

C. *Lifetime of Sensor Nodes*

In a wireless sensor network, the early death of some nodes can disconnect other nodes from the base station. This situation can lead to a reduced usefulness of the network because some data cannot reach the base station. We measure lifetime in two ways: (1) the time at which the first node dies and (2) the time at which the last node dies. The time at which the first node dies is important because it can lead to a disconnection of part of the network. The time at which the last node dies shows how long nodes are able to run the protocol. Figure 5(a) shows the time at which the first node died for the ECDS and the HEED algorithm. The time of the first death asymptotically decreases in ECDS and is constant for HEED. Figure 5(b) shows the time of death for the last node in the network. It is equally important that all nodes die around the same time. A single node that outlives others by a large margin is of little use. It can be estimated that the lifetimes will be similar for ECDS and HEED in large networks.

For both ECDS and HEED the first and last deaths are within 200 seconds of each other, indicating an even energy consumption across the network.

D. *Energy Consumption*

The amount of energy used during the execution of a protocol is very important in sensor networks. Figure 5(c) shows the average energy consumption for the two protocols. We use PowerTossim to determine energy used. From that we can calculate the average energy used per node and the average per simulation. The energy consumption of the HEED algorithm is linear, while the energy consumption of the ECDS algorithm is asymptotically decreasing. As the networks grow larger, the energy consumption for ECDS and HEED will be similar. In sensor networks, the energy consumption for each message sent should be considered in addition to the overall energy consumption. A sensor network that uses very little energy is not useful if it does not produce an adequate amount of data. Figure 5(d) shows the average energy consumption for each message sent. Since ECDS clusters faster, it generates more messages.

VII. Conclusion and Open Problems

In this paper, two different algorithms are presented to address the problem of energy constrained clustering for wireless sensor networks. For the greedy algorithm we provide an $\mathcal{O}(\log n)$ approximation guarantee. The second algorithm presented is a distributed algorithm for the energy constrained dominating set. We proved that this algorithm runs in $\mathcal{O}(\log n \log \Delta)$ rounds whp. This algorithm performs well on the random graphs in our simulations. Our simulations showed that our algorithm performs very well in terms of time to cluster, cluster size, and energy consumption. We compared our algorithm with the HEED algorithm. It outperformed HEED in terms of cluster size, time to cluster, and energy consumption per message sent. Future work will include extending the algorithm to consider node proximity when selecting cluster heads and deciding which nodes to add to the cluster. Considering node proximity will produce tighter

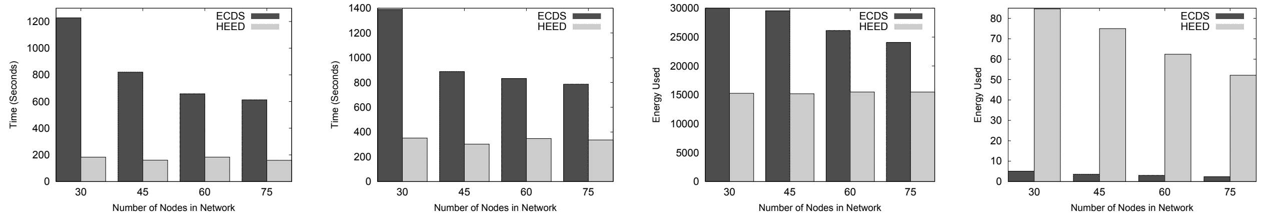

Fig. 5. 5(a) First Death 5(b) Last Death 5(c) Average Energy 5(d) Average Energy Per Message

clusters and minimize the overall energy consumption within each cluster. Secondly, we plan on extending the algorithm to allow for multi-hop clusters. Currently every node is one hop from its cluster head. We will extend the algorithm to allow nodes to be k-hops from their cluster heads. Additionally, we plan on extending the algorithm to allow each node to have multiple cluster heads which will ensure that each node has access to at least one cluster head at all times. Ensuring multiple coverings for each node will allow for the use of multi-path routing in clustered networks.

References

[1] W. R. Heinzelman, J. Kulik, and H. Balakrishnan, "Adaptive protocols for information dissemination in wireless sensor networks," in *MobiCom '99: Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*. ACM Press, 1999, pp. 174–185.

[2] S. Lee, C. Kim, and S. Kim, "Constructing energy efficient wireless sensor networks by variable transmission energy level control," in *Computer and Information Technology, 2006. CIT '06. The Sixth IEEE International Conference on*, 2006, pp. 225–225.

[3] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," in *HICSS '00: Proceedings of the 33rd Hawaii International Conference on System Sciences-Volume 8*. Washington, DC, USA: IEEE Computer Society, 2000, p. 8020.

[4] D. Wei and A. H. Chan, "Clustering algorithm to balance and to reduce power consumptions for homogeneous sensor networks," in *Wireless Communications, Networking and Mobile Computing, 2007. WiCom 2007. International Conference on*, 2007, pp. 2723–2726.

[5] R. Krishnan and D. Starobinski, "Efficient clustering algorithms for self-organizing wireless sensor networks," *Ad Hoc Networks*, vol. 4, no. 1, pp. 36–59, January 2006.

[6] C. Duan and H. Fan, "A distributed energy balance clustering protocol for heterogeneous wireless sensor networks," in *Wireless Communications, Networking and Mobile Computing, 2007. WiCom 2007. International Conference on*, 2007, pp. 2469–2473.

[7] O. Younis and S. Fahmy, "HEED: a hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks," *Mobile Computing, IEEE Transactions on*, vol. 3, no. 4, pp. 366–379, 2004.

[8] F. Kuhn and R. Wattenhofer, "Constant-time distributed dominating set approximation," in *In Proc. of the 22nd ACM Symposium on the Principles of Distributed Computing*, 2003.

[9] R. Karp, "Reducibility among combinatorial problems," in *Complexity of Computer Computations*, R. E. Miller and J. W. Thatcher, Eds. Plenum Press, 1972, pp. 85–103.

[10] F. Kuhn, T. Moscibroda, and R. Wattenhofer, "Unit disk graph approximation," in *DIALM-POMC '04: Proceedings of the 2004 joint workshop on Foundations of mobile computing*. ACM Press, 2004, pp. 17–23.

[11] J. Wu and H. Li, "Domination and its applications in ad hoc wireless networks with unidirectional links," in *ICPP '00: Proceedings of the Proceedings of the 2000 International Conference on Parallel Processing*. Washington, DC, USA: IEEE Computer Society, 2000, p. 189.

[12] D. Dubhashi, A. Mei, A. Panconesi, J. Radhakrishnan, and A. Srinivasan, "Fast distributed algorithms for (weakly) connected dominating sets and linear-size skeletons," in *SODA '03: Proceedings of the fourteenth annual ACM-SIAM symposium on Discrete algorithms*. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 2003, pp. 717–724.

[13] V. Mhatre and C. Rosenberg, "Design guidelines for wireless sensor networks: communication, clustering and aggregation," *Ad Hoc Networks*, vol. 2, no. 1, pp. 45–63, 2004.

[14] M. Younis, M. Youssef, and K. Arisha, "Energy-aware routing in cluster-based sensor networks," in *MASCOTS '02: Proceedings of the 10th IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunications Systems (MASCOTS'02)*. Washington, DC, USA: IEEE Computer Society, 2002, p. 129.

[15] S. Banerjee and S. Khuller, "A clustering scheme for hierarchical control in multi-hop wireless networks," in *IEEE INFOCOM*, April 2001, pp. 1028–1037.

[16] M. Chen and J. Wu, "EECS: an energy efficient clustering scheme in wireless sensor networks," *Performance, Computing, and Communications Conference, 2005. IPCCC 2005. 24th IEEE International*, pp. 535–540, 2005.

[17] F. Grandoni, J. Knemann, A. Panconesi, and M. Sozio, "Primal-dual based distributed algorithms for vertex cover with semi-hard capacities," in *PODC '05: Proceedings of the twenty-fourth annual ACM SIGACT-SIGOPS symposium on Principles of distributed computing*. New York, NY, USA: ACM Press, 2005, pp. 118–125.

[18] L. Jia, R. Rajaraman, and T. Suel, "An efficient distributed algorithm for constructing small dominating sets," *Distrib. Comput.*, vol. 15, no. 4, pp. 193–205, 2002.

[19] S. Kutten and D. Peleg, "Fast distributed construction of k-dominating sets and applications," in *PODC '95: Proceedings of the fourteenth annual ACM symposium on Principles of distributed computing*. ACM Press, 1995, pp. 238–251.

[20] Y. Chen and A. Liestman, "Approximating minimum size weakly-connected dominating sets for clustering mobile ad hoc networks," in *MobiHoc '02: Proceedings of the 3rd ACM international symposium on Mobile ad hoc networking & computing*. New York, NY, USA: ACM Press, 2002, pp. 165–172.

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

[22] B. Liang and Z. Haas, "Virtual backbone generation and maintenance in ad hoc network mobility management," in *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 3, 2000, pp. 1293–1302 vol.3.

[23] P. Levis, N. Lee, M. Welsh, and D. Culler, "Tossim: accurate and scalable simulation of entire tinyos applications," in *SenSys '03: Proceedings of the 1st international conference on Embedded networked sensor systems*. New York, NY, USA: ACM Press, 2003, pp. 126–137.

[24] O. Younis and S. Fahmy, "An experimental study of routing and data aggregation in sensor networks," *Mobile Adhoc and Sensor Systems Conference, 2005. IEEE International Conference on*, pp. 8 pp.–, Nov. 2005.

[25] A. Boukerche, X. Fei, and R. B. Araujo, "An optimal coverage-preserving scheme for wireless sensor networks based on local information exchange," *Computer Communications*, vol. 30, no. 14-15, pp. 2708–2720, October 2007.