HEED: A Hybrid, Energy-Efficient, Distributed Clustering Approach for Ad Hoc Sensor Networks

Ossama Younis, *Student Member*, *IEEE*, and Sonia Fahmy, *Member*, *IEEE*

Abstract—Topology control in a sensor network balances load on sensor nodes and increases network scalability and lifetime. Clustering sensor nodes is an effective topology control approach. In this paper, we propose a novel distributed clustering approach for long-lived ad hoc sensor networks. Our proposed approach does not make any assumptions about the presence of infrastructure or about node capabilities, other than the availability of multiple power levels in sensor nodes. We present a protocol, *HEED* (Hybrid Energy-Efficient Distributed clustering), that periodically selects cluster heads according to a hybrid of the node residual energy and a secondary parameter, such as node proximity to its neighbors or node degree. HEED terminates in $O(1)$ iterations, incurs low message overhead, and achieves fairly uniform cluster head distribution across the network. We prove that, with appropriate bounds on node density and intracluster and intercluster transmission ranges, HEED can asymptotically almost surely guarantee connectivity of clustered networks. Simulation results demonstrate that our proposed approach is effective in prolonging the network lifetime and supporting scalable data aggregation.

Index Terms—Sensor networks, clustering, network lifetime, energy efficiency, fault tolerance.

✦

1 INTRODUCTION

SENSOR networks have recently emerged as a platform for several important surveillance and control applications [1], [2]. Sensor nodes are typically less mobile, more limited in capabilities, and more densely deployed than mobile ad hoc networks (MANETs). This necessitates devising novel energy-efficient solutions to some of the conventional wireless networking problems, such as medium access control, routing, self-organization, bandwidth allocation, and security. Exploiting the tradeoffs among energy, accuracy, and latency, and using hierarchical (tiered) architectures are important techniques for prolonging the network lifetime.

Network lifetime can be defined as the time elapsed until the first node (or the last node) in the network depletes its energy (dies). For example, in a military field where sensors are monitoring chemical activity, the lifetime of a sensor is critical for maximum field coverage. Energy consumption in a sensor node can be attributed to either "useful" or "wasteful" sources. Useful energy consumption can be due to

1. transmitting/receiving data,
2. processing query requests, and
3. forwarding queries/data to neighboring nodes.

Wasteful energy consumption can be due to

1. idle listening to the media,
2. retransmitting due to packet collisions,

3. overhearing, and
4. generating/handling control packets.

A number of protocols have been proposed to reduce useful energy consumption. These protocols can be classified into three classes. Protocols in the first class control the transmission power level at each node by increasing network capacity while keeping the network connected [3], [4]. Protocols in the second class make routing decisions based on power optimization goals, e.g., [5], [6], [7], [8]. Protocols in the third class control the network topology by determining which nodes should participate in the network operation (be awake) and which should not (remain asleep) [9], [10], [11]. Nodes in this case, however, require knowledge of their locations via GPS-capable antennae or via message exchange.

Hierarchical (clustering) techniques can aid in reducing useful energy consumption [8]. Clustering is particularly useful for applications that require scalability to hundreds or thousands of nodes. Scalability in this context implies the need for load balancing, efficient resource utilization, and data aggregation. Routing protocols can also employ clustering [12], [13]. Clustering can be extremely effective in one-to-many, many-to-one, one-to-any, or one-to-all (broadcast) communication.

Although many protocols proposed in the literature minimize energy consumption on forwarding paths to increase energy efficiency, such protocols do not necessarily prolong network lifetime when certain nodes are "popular," i.e., present on most forwarding paths in the network. Even if dynamic routing (in which data is forwarded to nodes with the highest residual energy) is used, it may cause problems such as unbounded delay and routing loops. With clustering, a popular node is guaranteed to "lose its popularity" as new clusters (and forwarding paths) are

● *The authors are with the Department of Computer Sciences, Purdue University, 250 N. University Street, West Lafayette, IN 47907-2066. E-mail: {oyounis, fahmy}@cs.purdue.edu.*
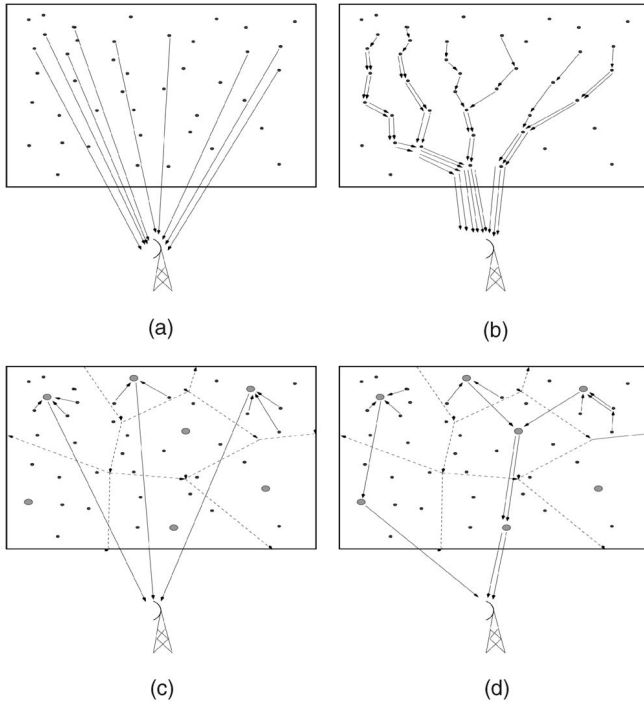
www.manaraa.com

Fig. 1. Sensor information forwarding with and without clustering and aggregation. (a) Single hop without clustering. (b) Multihop without clustering. (c) Single hop with clustering. (d) Multihop with clustering.

constructed. Of course, node popularity due to interest in the data it provides can only be reduced by deploying several redundant nodes and rotating among them (e.g., [9]).

The essential operation in sensor node clustering is to select a set of cluster heads from the set of nodes in the network, and then cluster the remaining nodes with these heads. Cluster heads are responsible for coordination among the nodes within their clusters and aggregation of their data (intracluster coordination), and communication with each other and/or with external observers on behalf of their clusters (intercluster communication). Fig. 1 depicts an application where sensors periodically transmit information to a remote observer (e.g., a base station). The figure illustrates that clustering can reduce the communication overhead for both single-hop and multihop networks. Periodic reclustering can select nodes with higher residual energy to act as cluster heads. Network lifetime is prolonged through

1. reducing the number of nodes contending for channel access,
2. summarizing information and updates at the cluster heads, and
3. routing through an overlay among cluster heads, which has a relatively small network diameter.

Clustering protocols have been investigated in the context of routing protocols [3], [14], [12], [15], [8], or independent of routing [16], [17], [13], [18], [19], [20]. In this work, we present a general distributed clustering approach that considers a hybrid of energy and communication cost. Based on this approach, we present the HEED (Hybrid, Energy-Efficient, Distributed) clustering protocol. HEED has four primary objectives [21]:

1. prolonging network lifetime by distributing energy consumption,
2. terminating the clustering process within a constant number of iterations,
3. minimizing control overhead (to be linear in the number of nodes), and
4. producing well-distributed cluster heads.

Our clustering approach does not make assumptions about the distribution of nodes or about node capabilities, e.g., location-awareness. The approach only assumes that sensor nodes can control their transmission power level.

The problem that we address in this work has unique requirements that distinguish it from the classical load-balancing problem in distributed systems. In classical distributed systems, a node can either be a server or a source, but not both. A fixed number of servers is known to every source in the system, and a server is always available for processing (see [22] for more details). In our model, every node can act as both a source and a server (cluster head), which motivates the need for efficient algorithms to select servers according to the system goals outlined below. A node only knows about the servers that are within its reachable range, which implies that achieving global goals cannot always be guaranteed but can be approximated through intelligent local decisions. Finally, a node may fail if its energy resource is depleted, which motivates the need for rotating the server role among all nodes for load balancing.

The remainder of this paper is organized as follows: Section 2 describes the network model and states the problem that we address in this work. Section 3 presents the HEED protocol and argues that it satisfies its objectives. Section 4 shows HEED effectiveness via simulations and compares it to other clustering techniques. Section 5 discusses applications that can use HEED, and compares HEED with a generalized energy-efficient version of LEACH [8]. Section 6 discusses some of the HEED design issues and possible extensions. Section 7 briefly surveys related work. Finally, Section 8 gives concluding remarks and directions for future work.

2 PROBLEM STATEMENT

We first describe the network model and then give our objectives.

2.1 Network Model

Consider a set of sensors dispersed in a field. We assume the following properties about the sensor network:

- The sensor nodes are quasi-stationary. This is typical for sensor network applications.
- Links are symmetric, i.e., two nodes v_1 and v_2 can communicate using the same transmission power level.
- The network serves multiple mobile/stationary observers, which implies that energy consumption is not uniform for all nodes.
- Nodes are location-unaware, i.e., not equipped with GPS-capable antennae. This justifies why some techniques, such as [10], [23] are inapplicable.

- All nodes have similar capabilities (processing/ communication), and equal significance. This motivates the need for extending the lifetime of every sensor.
- Nodes are left unattended after deployment. Therefore, battery recharge is not possible. Efficient, energy-aware sensor network protocols are thus required for energy conservation.
- Each node has a fixed number of transmission power levels. An example of such sensor nodes are Berkeley Motes [24]. It is typically straightforward to set the transmission power level via the standard *ioctl()* system call.

Let the clustering process duration, T_{CP}, be the time interval taken by the clustering protocol to cluster the network. Let the *network operation interval*, T_{NO}, be the time between the end of a T_{CP} interval and the start of the subsequent T_{CP} interval. We must ensure that $T_{NO} \gg T_{CP}$ to reduce overhead. (Section 5 further discusses how to set T_{NO}.) Although we assume that nodes are not mobile, clustering can still be performed if nodes that announce their willingness to be cluster heads are quasi-stationary during the T_{CP} interval in which they are selected, and the ensuing T_{NO} interval. Nodes that travel rapidly in the network may degrade the cluster quality because they alter the node distribution in their cluster.

We currently assume that node failures are primarily caused by energy depletion. In Section 6.3, we discuss measures to withstand unexpected node failures in hostile environments, such as volcanic areas or military fields.

It is important to note that in our model, *no* assumptions are made about

1. homogeneity of node dispersion in the field,
2. network density or diameter,
3. distribution of energy consumption among sensor nodes,
4. proximity of querying observers, or
5. node synchronization.

In Section 3.3 and Section 4.4, we show that unsynchronized nodes can still execute HEED independently, but cluster quality may be affected. For time sensitive applications, the network can be synchronized using techniques, such as RBS [25].

2.2 The Clustering Problem

Assume that n nodes are dispersed in a field and the above assumptions hold. Our goal is to identify a set of cluster heads which cover the entire field. Each node v_i, where $1 \leq i \leq n$, must be mapped to exactly one cluster c_j, where $1 \leq j \leq n_c$, and n_c is the number of clusters ($n_c \leq n$). A node must be able to directly communicate with its cluster head (via a single hop). Cluster heads can use a routing protocol to compute intercluster paths for multihop communication to the observer(s), as discussed in Section 4. The following requirements must be met:

1. Clustering is completely distributed. Each node independently makes its decisions based only on local information.

2. Clustering terminates within a fixed number of iterations (regardless of network diameter).
3. At the end of each T_{CP}, each node is either a cluster head, or not a cluster head (which we refer to as a regular node) that belongs to exactly one cluster.
4. Clustering should be efficient in terms of processing complexity and message exchange.
5. Cluster heads are well-distributed over the sensor field and have relatively high average residual energy compared to regular nodes.

3 THE HEED PROTOCOL

In this section, we describe the HEED protocol. First, we define the parameters used in the clustering process. Second, we present the protocol design and pseudocode. Finally, we prove that the protocol meets its requirements.

3.1 Clustering Parameters

The overarching goal of our approach is to prolong network lifetime. For this reason, cluster head selection is primarily based on the residual energy of each node. Measuring this residual energy is not necessary since the energy consumed per bit for sensing, processing, and communication is typically known and, hence, residual energy can be estimated. To increase energy efficiency and further prolong network lifetime, we also consider intracluster "communication cost" as a secondary clustering parameter. For example, cost can be a function of neighbor proximity or cluster density.

We use the primary clustering parameter to probabilistically select an initial set of cluster heads, and the secondary parameter to "break ties" among them. A tie in this context means that a node falls within the "range" of more than one cluster head. To understand what "range" denotes in this case, observe that a node typically has a number (e.g., six) of discrete transmission power levels. Thus, the *cluster range* or *radius* is determined by the transmission power level used for intracluster announcements and during clustering. We refer to this level as the *cluster power level*. The cluster power level should be set to one of the lower power levels of a node, to increase spatial reuse, and reserve higher power levels for intercluster communication. These higher power levels should cover at least two or more cluster diameters to guarantee that the resulting intercluster overlay will be connected. If this condition cannot be satisfied, then our approach for clustering in conjunction with power level selection is inapplicable. We analyze intercluster connectivity conditions in Section 3.4. The cluster power level dictates the number of clusters in our network. It is nontrivial to determine an optimal cluster power level because network topology changes due to node failures and energy depletion.

The secondary clustering parameter, intracluster communication cost, is a function of 1) cluster properties, such as cluster size, and 2) whether or not variable power levels are permissible for intracluster communication. If the power level used for intracluster communication is fixed for all nodes, then the cost can be proportional to 1) node degree, if the requirement is to distribute load among cluster heads, or 2) $\frac{1}{node\ degree}$, if the requirement is to create

TABLE 1
Definitions of Communication Cost According to Goals and
Intracluster Communication Power

Goal \ Power	Same	Minimum
Load distribution	node degree	AMRP
		node degree
Dense clusters	$\frac{1}{node\ degree}$	AMRP
		closest node

dense clusters. This means that a node joins the cluster head with minimum degree to distribute cluster head load (possibly at the expense of increased interference and reduced spatial reuse), or joins the one with maximum degree to create dense clusters. We use the terms *minimum degree cost* and *maximum degree cost* to denote these cost types. Observe that intercluster communication is not incorporated in the cost function since local information is insufficient in this case.

Now, consider the case when variable power levels are allowed for intracluster communication. Let $MinPwr_i$ denote the minimum power level required by a node v_i, $1 \leq i \leq M$, to communicate with a cluster head u, where M is the number of nodes within the cluster range. We define the *average minimum reachability power (AMRP)* as the mean of the minimum power levels required by all M nodes within the cluster range to reach u, i.e.,

$$AMRP = \frac{\sum_{i=1}^{M} MinPwr_i}{M}.$$

If each node is allowed to select the appropriate power level to reach its cluster head, then AMRP provides a good estimate of the communication cost. The AMRP of a node is a measure of the expected intracluster communication energy consumption if this node becomes a cluster head. Using AMRP as cost in selecting cluster heads is superior to just selecting the closest cluster head, since it provides a unified mechanism for all nodes, including cluster heads, to break ties among tentative cluster heads. If a node has to select its cluster head among nodes *not* including itself, the closest neighbor within its cluster range (the neighbor reached using the smallest power level) can be selected as its cluster head. Table 1 summarizes the different options for computing the communication cost.

3.2 Protocol Operation

As discussed in Section 2, clustering is triggered every $T_{CP} + T_{NO}$ seconds to select new cluster heads. At each node, the clustering process requires a number of iterations, which we refer to as N_{iter}. Every step takes time t_c, which should be long enough to receive messages from any neighbor within the cluster range. We set an initial percentage of cluster heads among all n nodes, C_{prob} (say 5 percent), assuming that an optimal percentage cannot be computed a priori. C_{prob} is only used to limit the initial cluster head announcements, and has no direct impact on the final clusters. Before a node starts executing HEED, it

sets its probability of becoming a cluster head, CH_{prob}, as follows:

$$CH_{prob} = C_{prob} \times \frac{E_{residual}}{E_{max}}, \qquad (1)$$

where $E_{residual}$ is the estimated current residual energy in the node and E_{max} is a reference maximum energy (corresponding to a fully charged battery), which is typically identical for all nodes. The CH_{prob} value of a node, however, is not allowed to fall below a certain threshold p_{min} (e.g., 10^{-4}), that is selected to be inversely proportional to E_{max}. This restriction is essential for terminating the algorithm in $N_{iter} = O(1)$ iterations, as we will show later. Observe that our clustering approach is capable of handling heterogeneous node batteries. In this case, every node will have its own E_{max} value.

During any iteration i, $i \leq N_{iter}$, every "uncovered" node (as defined below) elects to become a cluster head with probability CH_{prob}. After step i, the set of tentative cluster heads, S_{CH}, is set to {cluster heads after step $i-1 \cup$ new heads selected in step i}. A node v_i selects its cluster head (*my_cluster_head*) to be the node with the lowest cost in S_{CH} (S_{CH} may include v_i itself if it is selected as a tentative cluster head). Every node then doubles its CH_{prob} and goes to the next step. The pseudocode for each node is given in Fig. 2. Note that if different power levels can be used for intracluster communication, then line 1 in phase I must be modified as follows: Discover neighbors within every power level $Pwr_i \leq Pwr_c$, where Pwr_c is the cluster power level. In this case only, we assume that if cluster head u can reach a node v with power level l, then v can reach u with level l as well. Neighbor discovery is not necessary every time clustering is triggered. This is because, in a stationary network, where nodes do not die unexpectedly, the neighbor set of every node does not change very frequently. In addition, HEED distribution of energy consumption extends the lifetime of all the nodes in the network, which adds to the stability of the neighbor set. Nodes also automatically update their neighbor sets in multihop networks by periodically sending and receiving heartbeat messages.

Note also that if a node elects to become a cluster head, it sends an announcement message *cluster_head_msg(Node_ID, selection status, cost)*, where the selection status is set to *tentative_CH*, if its CH_{prob} is less than 1, or *final_CH*, if its CH_{prob} has reached 1. A node considers itself "covered" if it has heard from either a *tentative_CH* or a *final_CH*. If a node completes HEED execution without selecting a cluster head that is *final_CH*, it considers itself uncovered, and announces itself to be a cluster head with state *final_CH*. A *tentative_CH* node can become a regular node at a later iteration if it finds a lower cost cluster head. Note that a node can elect to become a cluster head at consecutive clustering intervals if it has high residual energy and low cost.

3.3 Correctness and Complexity

The protocol described in Fig. 2 meets the requirements listed in Section 2.2, as discussed next.

Observation 1. *HEED is completely distributed (requirement 1).*
A node can either elect to become a cluster head according to its

I. Initialize

1. $S_{nbr} \leftarrow \{v: v$ lies within my cluster range$\}$

2.   Compute and broadcast cost to $\in S_{nbr}$

3. $CH_{prob} \leftarrow max(C_{prob} \times \frac{E_{residual}}{E_{max}}, p_{min})$

4.   is_final_CH $\leftarrow$ FALSE

III. Finalize

1.   If (is_final_CH = FALSE)

2.   If (($S_{CH} \leftarrow \{v: v$ is a final cluster head$\}) \neq \phi$)

3.   my_cluster_head $\leftarrow$ least_cost(S_{CH})

4.   join_cluster(cluster_head_ID, NodeID)

5.   Else Cluster_head_msg(NodeID, final_CH, cost)

6.   Else Cluster_head_msg(NodeID, final_CH, cost)

II. Main Processing

Repeat

1.   If (($S_{CH} \leftarrow \{v: v$ is a cluster head$\}) \neq \phi$)

2.   my_cluster_head $\leftarrow$ least_cost(S_{CH})

3.   If (my_cluster_head = NodeID)

4.   If (CH_{prob} = 1)

5.   Cluster_head_msg(NodeID, final_CH, cost)

6.   is_final_CH $\leftarrow$ TRUE

7.   Else

8.   Cluster_head_msg(NodeID, tentative_CH, cost)

9.   ElseIf (CH_{prob} = 1)

10.   Cluster_head_msg(NodeID, final_CH, cost)

11.   is_final_CH $\leftarrow$ TRUE

12.   ElseIf Random(0,1) $\leq CH_{prob}$

13.   Cluster_head_msg(NodeID, tentative_CH, cost)

14. $CH_{previous} \leftarrow CH_{prob}$

15. $CH_{prob} \leftarrow min(CH_{prob} \times 2, 1)$
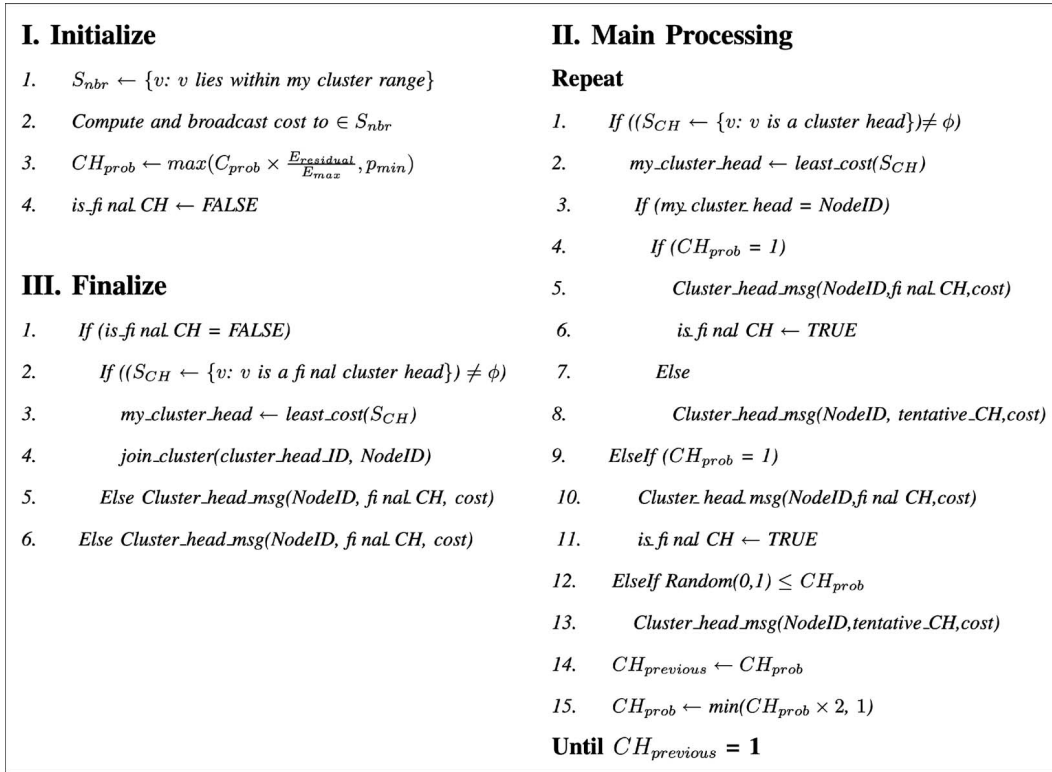
Until $CH_{previous}$ **= 1**

Fig. 2. HEED protocol pseudocode.

CH_{prob} or join a cluster according to overheard cluster head messages within its cluster range.

Lemma 1. *HEED terminates in $N_{iter} = O(1)$ iterations (requirement 2).*

Proof. The worst case occurs when a node has a very low $E_{residual}$. This node will start executing HEED with CH_{prob} set to p_{min}. However, CH_{prob} doubles in every step, and phase II of the protocol terminates one step (iteration) after CH_{prob} reaches 1. Therefore, $2^{N_{iter}-1} \times p_{min} \geq 1$ and, hence,

$$N_{iter} \leq \left\lceil log_2 \frac{1}{p_{min}} \right\rceil + 1. \qquad (2)$$

Therefore, $N_{iter} \approx O(1)$.   □

With the appropriate choice of the minimum probability of becoming a cluster head, the number of iterations can be bounded by a reasonable constant (requirement 2). For example, for $p_{min} = 10^{-4}$, a low-energy node will need 15 iterations in phase II. When $E_{residual}$ is close to E_{max}, the number of iterations is much lower, and depends on the value of C_{prob}. For example, for $C_{prob} = 5\%$, high-energy nodes will exit HEED in only six iterations. Thus, nodes with high residual energy will terminate HEED earlier than nodes with lower residual energy. This allows low energy nodes to join their clusters.

Lemma 2. *At the end of phase III of the HEED protocol, a node is either a cluster head or a regular node that belongs to a cluster (requirement 3).*

Proof. Assume that a node terminates its execution of HEED without electing to become a cluster head or

joining a cluster. This implies that the condition in line 1 of phase III is satisfied, while the condition in line 2 is not satisfied (hence, line 4 is not executed). In this case, line 5 will be executed and the node will become a cluster head, which is a contradiction.   □

To prolong the sensor network lifetime, cluster head selection is primarily based on the residual energy of each node (an estimated value will be sufficient). To increase energy efficiency and further prolong network lifetime, we also consider intracluster "communication cost" as a secondary clustering parameter. For example, cost can be a function of neighbor proximity or cluster density.

Lemma 3. *HEED has a worst-case processing time complexity of $O(n)$ per node, where n is the number of nodes in the network (requirement 4).*

Proof. Phase I in the HEED protocol takes a processing time of at most n to compute the cost, if the cost definition is the AMRP. Similarly, Phase III also takes a processing time of at most n to arbitrate among the nodes which declared their willingness to be cluster heads with state *final_CH*. For Phase II, the time taken to arbitrate among cluster heads (for all passes) is at most $N_{iter} \times n$ cluster heads. From Lemma 1, N_{iter} is a constant. Therefore, the total time is still $O(n)$. All other iterations have an $O(1)$ time complexity. Therefore, the total processing complexity is $O(n)$.   □

Lemma 4. *HEED has a worst-case message exchange complexity of $O(1)$ per node, i.e., $O(n)$ in the network (requirement 4).*

Proof. During the execution of HEED, a tentative cluster head generates at most N_{iter} cluster head messages

($O(1)$). A regular node is silent until it sends one join message to a cluster head. The number of these join messages in the network is strictly less than n, since at least one node will decide to be a cluster head with state *final_CH* during the clustering process. Hence, the number of messages exchanged in the network is upper-bound by $N_{iter} \times n$, i.e., $O(n)$. □

Lemma 5. *The probability that two nodes within each other's cluster range are both cluster heads is small, i.e., cluster heads are well-distributed (requirement 5).*

Proof. Consider the following worst-case scenario. Assume that v_1 and v_2 are two isolated neighboring nodes (i.e., each one does not have any other neighbor in close proximity). We compute the probability, p_{nbr}, that at the end of Phase III, both of them are cluster heads (we assume that they are fully synchronized). In the worst case, neither of the two nodes decides to be a cluster head before its CH_{prob} reaches 1. Otherwise, one of them will concede to the other. Two cases may occur in this scenario:

Case 1. The CH_{prob} values of v_1 and v_2 are different enough such that they do not execute the same number of iterations in Phase II. Without loss of generality, assume that $CH_{prob1} > CH_{prob2}$. In this case, v_1 will elect to become a cluster head with state *final_CH* before v_2. Hence, v_2 will receive a cluster head message and register with v_1. The same argument applies for unsynchronized nodes because they will likely terminate their computations at different times. That is why we state in Section 2.1 that synchronization is not critical for HEED operation.

Case 2. v_1 and v_2 will execute the same number of iterations in Phase II. In this case, at any step $i < N_{iter}$, neither v_1 nor v_2 decides to be a cluster head with probability $p_i = (1 - CH_{prob1})(1 - CH_{prob2})$. Let $prob_1$ denote the initial CH_{prob1} and $prob_2$ denote the initial CH_{prob2}. During step i, $0 \le i \le N_{iter} - 2$, the current $CH_{prob1} = prob_1 \times 2^i$ and $CH_{prob2} = prob_2 \times 2^i$. Let p_{nbr} be the probability that neither v_1 nor v_2 elects to become a cluster head at any step i:

$$p_{nbr} = \prod_{i=0}^{N_{iter}-2} (1 - prob_1 \times 2^i)(1 - prob_2 \times 2^i).$$

When $prob_1 = prob_2 = p$, we get

$$p_{nbr} = \prod_{i=0}^{(\lceil log_p^{\frac{1}{p}} \rceil - 1)} (1 - p \times 2^i)^2. \tag{3}$$

With typical values of the initial CH_{prob} for all nodes, the probability p_{nbr} is very small. For example, for $p = 3\%$, the resulting $p_{nbr} = 0.00016$, while for $p = 5\%$, the resulting $p_{nbr} = 0.006$. A loose upper bound for (3) is $p_{nbr} < e^{-2p(1+2+4+...+2^{(\lceil log_p^{\frac{1}{p}} \rceil - 1)})}$ or $p_{nbr} < e^{-2p(2^{\lceil log_p^{\frac{1}{p}} \rceil}-1)}$. This probability, however, is expected to be much smaller in practical situations, in which a node is likely to have more than one neighbor and similar starting CH_{prob} values will not be the common case. □

In all our experiments in Section 4, no two neighboring nodes were selected by HEED to act as cluster heads. This property remained valid with different transmission ranges, variable node density, and different cost types. Intuitively, the probabilistic choice of cluster heads according to their residual energy results in cluster heads that have higher average residual energy than regular nodes. We demonstrate this behavior is Section 4.

3.4 Intercluster Communication

After the network is clustered, intercluster organization depends on the network application. For example, cluster heads can communicate with each other to aggregate their information via multiple hops. For multihop communication among cluster heads, the selected transmission range among cluster heads may vary to ensure a certain degree of connectivity and to control interference. For example, in [26], the authors assume that the nodes are uniformly distributed in the network field and that each cell of size $c \times c$ in the network contains at least one node. In this case, the network is guaranteed to be connected if the intercluster transmission range $R_t = (1 + \sqrt{5})c$. A cell in this context is defined as an area in the 2-dimensional space in which every node can communicate with every other node residing in every neighboring cell. In a clustered network, a cell can be defined as an area where every node can reach every other node residing in the same cell. The cell side length is therefore $\le R_c/\sqrt{2}$, where R_c is the cluster range. Thus, we can conduct a similar analysis to [26], [27] to select R_t.[1] In [3], the authors suggest using the minimum possible power level to reach a destination, in order to reduce interference. In [4], the authors propose a technique to select the minimum power level to use across the entire network in order to keep it connected, assuming uniform node dispersion. Any of these techniques can be adopted to guarantee a connected intercluster overlay graph.

For intercluster communication, the definition of connectivity depends on its multihop organization and the relationship between the intercluster transmission range, R_t, and the intracluster transmission range, R_c. The following lemmas and theorem define the required density model and provide the necessary conditions for asymptotically almost surely (a.a.s.) multihop network connectivity.

Lemma 6. *Assume that n nodes are uniformly and independently dispersed at random in an area $R = [0, L]^2$. Also, assume that the area is divided into N square cells of size $\frac{R_c}{\sqrt{2}} \times \frac{R_c}{\sqrt{2}}$. If $R_c^2 n = aL^2 ln\, L$, for some $a > 0$, then $lim_{n,N \to \infty} E[\eta(n, N)] = 1$, where $\eta(n, N)$ random variable that denotes the minimum number of nodes in a cell (i.e., each cell contains at least one node a.a.s., or the expected number of empty cells is zero a.a.s.).*

A similar theorem was proven in [27]. Therefore, the proof is omitted.

---

1. Our definition of a cell is different from that in [27] which assumes that a node residing in a cell can communicate with all the nodes in its complete neighborhood (i.e., its eight surrounding cells). They use this definition to analyze the performance of cell-based approaches (e.g., GAF [10]). We regard a cell as an approximation of a cluster and, thus, R_c is used to define the required density and R_t is used to define connectivity. In the analysis in [27], only one transmission range is used to define both density and connectivity.
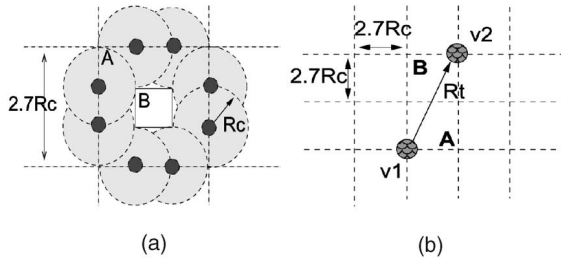
Fig. 3. Conditions on transmission range for network connectivity. (a) External cluster heads covering parts of area A. (b) Minimum transmission range for intercluster communication.

Lemma 7. *There exists at least one cluster head in any $(2 + \frac{1}{\sqrt{2}})R_c \times (2 + \frac{1}{\sqrt{2}})R_c$ area a.a.s.*

Proof. We prove this lemma by contradiction. Assume that Lemma 6 holds, and that there there does not exist any cluster heads in an $(2 + \frac{1}{\sqrt{2}})R_c \times (2 + \frac{1}{\sqrt{2}})R_c$ area A. This implies that every node v within this area A is connected to a cluster head that lies outside A. Even if cluster heads outside A are on the borders of A, then there is at least an area $B = \frac{R_c}{\sqrt{2}} \times \frac{R_c}{\sqrt{2}}$ inside A which cannot be covered by cluster heads outside A (as depicted in Fig. 3a). But, area B contains at least one node a.a.s. according to Lemma 6 and this node is connected to a cluster head within A. This contradicts the initial assumption and, therefore, there exists at least one cluster head within A a.a.s. □

Lemma 8. *For any two cluster heads v_1 and v_2 in two neighboring areas A and B of size $(2 + \frac{1}{\sqrt{2}})R_c \times (2 + \frac{1}{\sqrt{2}})R_c$, v_1, and v_2 can communicate if $R_t \geq 6R_c$.*

Proof. Fig. 3b shows an organization where a $(2 + \frac{1}{\sqrt{2}})R_c \times (2 + \frac{1}{\sqrt{2}})R_c$ area A contains one cluster head v_1 in the bottom left corner. A cluster head v_2 is the farthest from v_1 when it resides in the top right corner of the closest $(2 + \frac{1}{\sqrt{2}})R_c \times (2 + \frac{1}{\sqrt{2}})R_c$ area B. Using Euclidean geometry, the distance between v_1 and $v_2 \approx 6R_c$, which is the minimum transmission range R_t for v_1 to reach v_2. □

Theorem 1. *HEED produces a connected multihop cluster head graph (overlay) a.a.s.*

Proof. Assume that the conditions in the previous three lemmas hold. We prove this theorem by contradiction. Assume that HEED produces two connected components (graphs) of cluster heads $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$, such that any $v_1 \in V_1$ cannot communicate with any $v_2 \in V_2$. Without loss of generality, assume that V_2 lies on the right of V_1, and that a cluster head $v_1 \in V_1$ lies on the rightmost border of V_1. v_1 is able to communicate with a cluster head v_2 on its right side, since the condition in Lemma 8 holds. v_2 must reside inside V_2, which contradicts with the assumption that a cluster head in one component cannot communicate with one in the other component. Thus, V_1 and V_2 are connected a.a.s. □

We surmise that clustering and data aggregation in a dense network conserve energy. To evaluate this conjecture, we conduct a very simple worst-case analysis on an operational scenario. The goal of this analysis is to quantify the required node density to achieve such energy conservation for that scenario.

Assume that transmission proceeds from all nodes in the top left cell to an observer in the bottom right cell. Define the energy gain, E_g, as the difference between the energy consumed for transmitting one bit of data by all the nodes in the top left cell without clustering E_o, and the energy consumed for transmitting one bit of data by all the nodes in the top left cell using clustering and data aggregation E_c. Therefore, $E_g = E_o - E_c$.

Now, assume that:

1. Nodes are dispersed uniformly at random in a field.
2. One unit of energy is consumed for transmitting 1 bit of data per one unit of distance.
3. Every cell (as defined above) has one cluster head.

We now show that $E_g > 0$, if $n > 2\sqrt{2}(L/R_c)^2$.

Since $L >> R_c$, the optimal path length from the source nodes to the observer $= \sqrt{2}L$. To compute the suboptimal path length (in the clustered network), consider each 2×2 cell. The clustered network path at most deviates by a factor of $\sqrt{2}$ from the optimal $2 \times R_c$ path. Therefore, the suboptimal path length $= 2 \times \sqrt{2} \times R_c \times \sqrt{2}L/(2 \times R_c) = 2L$. The average number of nodes per cell (assuming uniform distribution) $= n \times R_c^2/(2 \times L^2)$.

E_o = energy consumed by all the nodes in the cell to send one bit along the $\sqrt{2}L$ path to the observer $= (n \times R_c^2/(2 \times L^2)) \times \sqrt{2}L = n \times R_c^2/\sqrt{2}L$.

E_c = energy consumed by all the nodes in the cell to send one bit to their cluster head at range R_c + the energy consumed by the cluster head to transmit on the suboptimal path to the destination $= [n \times R_c^2/(2 \times L^2) - 1] \times R_c + 2L$. Therefore, $E_c \approx n \times R_c^3/2L^2 + 2L$.

$$E_g = E_o - E_c = n \times R_c^2/\sqrt{2}L - n \times R_c^3/2L^2 - 2L > 0$$

$$\Longrightarrow n[(R_c^2/\sqrt{2}L) - (R_c^3/2L^2)] > 2L \Longrightarrow n > \frac{4L^3}{R_c^2(\sqrt{2}L - R_c)}.$$

Since $L >> R_c$, therefore, $\sqrt{2}L >> R_c$ and

$$n > 2\sqrt{2}(L/R_c)^2.$$

4 PERFORMANCE EVALUATION

In this section, we evaluate the performance of the HEED protocol via simulations. Unless otherwise specified, we assume that 1,000 nodes are uniformly dispersed into a field with dimensions $2,000 \text{ m} \times 2,000 \text{ m}$. We set the minimum probability for becoming a cluster head (p_{min}) to 0.0005 (which is reasonable for nodes with batteries of energy < 10 Joule). In this case, the maximum number of iterations that HEED may take at any node is 12 (according to Lemma 1). Initially, $CH_{prob} = C_{prob} = 5\%$ for all nodes. Wireless transmission laws dictate that power attenuation be proportional to the square of the covered distance (assuming fixed transmission power). If the distances are small (up to hundreds of meters), then the power attenuation can be assumed to be linear with the transmission radius [28]. Practically, other factors may also affect the received power, such as noise or physical

obstacles. For simplicity, we assume the absence of these factors in our experiments and, therefore, use the distance between nodes to account for the required transmission power level among them. We vary the cluster radius (range) from 25 m to 400 m to study how the protocol works with low to high coverage ranges. Every result shown is the average of 100 experiments. Each experiment uses a different randomly-generated topology, where each node is assigned a different randomly-generated residual energy level between 0 and 1 Joule (J). Residual energy is discretized into 20 levels to increase ties.

We compare HEED to a generic weight-based clustering protocol that is suitable for quasi-stationary ad hoc networks. DCA [16] and WCA [17] are examples of such weight-based clustering. In our experiments, the real-valued weight used for generic clustering is simply the node residual energy. During any step of the clustering process, a node does not make a decision about which cluster to join (or if it should become a cluster head itself) until all neighboring nodes with higher weights have already decided (similar to DCA [16]). This generic clustering (GC) protocol is a good baseline for comparison because it has the following features:

1. Clustering is distributed and only based on local information.
2. Selected cluster heads are guaranteed to be the nodes with the highest weights (residual energy) within their clusters.
3. A node is associated with only one cluster head.
4. No underlying assumptions about node dispersion in the field are made.
5. The number of iterations of the protocol is a function of network diameter, similar to most currently proposed clustering approaches in mobile ad hoc networks.
6. The time and message complexities are $O(n)$ and $O(1)$ per node, respectively.
7. It is guaranteed that no two cluster heads are neighbors, i.e. cluster heads are well-distributed in the network field.

In this section, we compare HEED to the GC protocol in terms of:

1. number of iterations required for the clustering process,
2. ratio of the number of clusters to the number of nodes in the network,
3. ratio of clusters with more than one node to the number of clusters,
4. standard deviation of the number of nodes in a cluster and maximum number of nodes in a cluster, and
5. average residual energy of the selected cluster heads.

We also study the case where nodes are not fully synchronized.

4.1 Clustering Iterations

We compare the number of iterations required for HEED and GC protocols to terminate. As previously discussed, the number of iterations in HEED can be deterministically
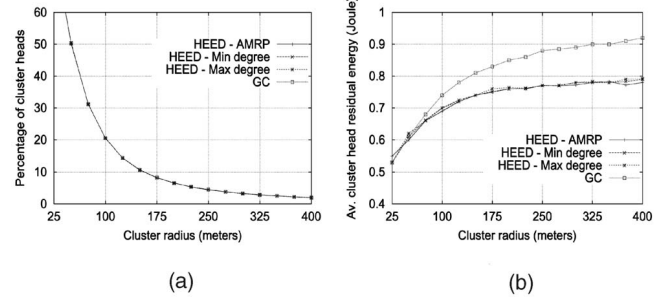

Fig. 4. Characteristics of selected cluster heads. (a) Percentage of cluster heads. (b) Average residual energy per cluster head.

computed using Lemma 1, which is independent of the cluster radius. For GC, the number of iterations grows quickly as the cluster radius increases because more neighbors are available for every node as the the cluster radius increases. Thus, a node will have to wait longer for higher weight nodes to decide which clusters to join. Our experiments show that GC takes only three iterations to terminate for a cluster radius of 25. The number of iterations, however, grows to 85 for a cluster radius of 400. HEED takes six iterations to terminate for all cluster ranges.

4.2 Cluster Head Characteristics

The number of selected cluster heads varies according to the specified cluster radius. The smaller the radius, the larger the required number of cluster heads to fully cover the entire network. Fig. 4a shows that the average number of cluster heads selected by both GC and HEED (with different cost types) are almost identical. This is not surprising since both GC and HEED tend to select cluster heads that are not neighbors within a cluster radius. The percentage of cluster heads is very high (80 percent) for very small cluster ranges, and becomes smaller as the range increases.

In HEED, tentative cluster heads are randomly selected based on their residual energy. Therefore, HEED cannot guarantee optimal head selection in terms of energy since it uses the secondary parameter to resolve conflicts. GC, a weight-based approach, does guarantee that the highest energy node will be the cluster head within its cluster range. Fig. 4b compares the two protocols in terms of residual energy. The results show that the cluster heads selected by HEED have high residual energy, and their average residual energy is not far lower than that with GC (at most 12 percent difference).

4.3 Cluster Characteristics

Application requirements dictate which cluster characteristics are favored in particular contexts. If it is required to balance load on cluster heads, then it is important to have clusters with small variance in the number of nodes they cover. Fig. 5a illustrates the standard deviation of the number of nodes per cluster for each cost type (cost types were defined in Section 3.1). The maximum degree cost type and GC show similar results. For minimum degree cost, the standard deviation is the lowest, because ties are broken by joining the smaller degree node, thus balancing the cluster
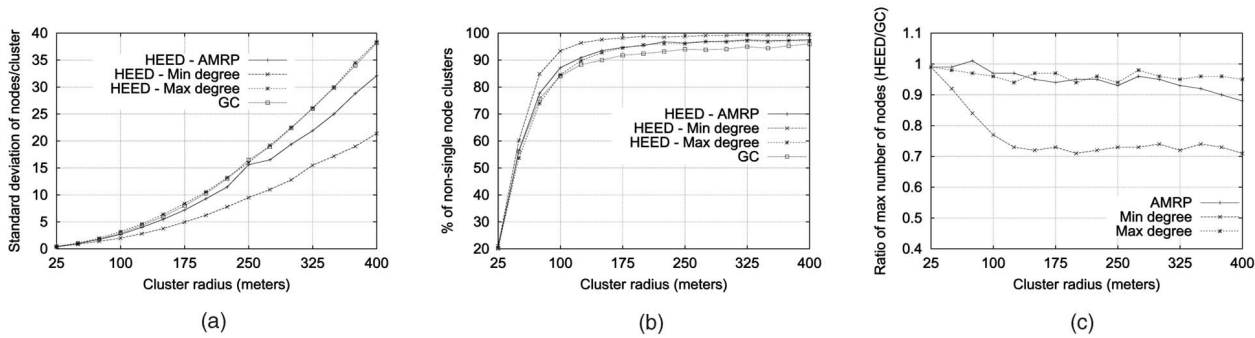
Fig. 5. Characteristics of clusters. (a) Standard deviation of the number of nodes/cluster. (b) Percentage of non-single-node clusters. (c) Ratio of maximum number of nodes in a HEED cluster to a GC cluster.

sizes. AMRP results lie between the two extremes. Therefore, AMRP provides a compromise between load balancing and cluster density.

Another appealing cluster property is minimizing clusters with only a single node (the cluster head), and minimizing the maximum number of nodes in a cluster. Single-node clusters arise when a node is forced to represent itself (because of not receiving any cluster head messages). A cluster may also contain a single node if this node decides to act as a cluster head and, due to cost definition, all its neighbors register themselves with other cluster heads. Fig. 5b illustrates the percentage of clusters with more than one node. The figure shows that HEED produces a higher percentage of non-single-node clusters than GC for all cost types. Fig. 5c shows that the maximum number of nodes in a cluster in HEED is on the average smaller than that of GC for all cost types, but especially for the minimum degree cost. Together, with the results about variance in the number of nodes in a cluster, presented in Fig. 5a, we can conclude that HEED produces balanced clusters.

4.4 Node Synchronization

In Section 2.1, we claimed that node synchronization is not critical for the operation of HEED. We argued why this claim holds in the proof of Lemma 5 (Case 1). We have conducted a number of experiments to study the effect of synchronization on the average cluster head energy. To compare the strictly synchronized case with a pseudosynchronized case, we assume that every node starts the clustering process randomly within a $3 \times t_c$ interval, i.e., within three iterations of the start of clustering process. This is a reasonable choice since using $C_{prob} = 0.05$ implies that Phase II of the HEED protocol terminates in six iterations in the case of a fully-charged battery. Fig. 6 illustrates the average cluster head energy for networks with synchronized versus pseudosynchronized nodes (labeled "unsynch"). Results indicate that the selected cluster heads in both cases have comparable residual energy. Results for other cluster and cluster head characteristics were also found to be similar to those presented above.

Several approaches can be applied to trigger the HEED protocol in an unsynchronized network. One possible approach is for nodes with faster clocks to trigger their slower neighbors to start the execution of HEED. A node is considered to have a "faster" clock, compared to its neighbors, if it has not received any HEED messages. This

approach will work in networks where nodes with faster clocks are evenly distributed in the network. If the nodes with faster clocks are clustered in certain regions, clustering will be triggered in these regions. In this case, "new" cluster heads are selected in these regions. These heads can rapidly discover their neighboring "old" cluster heads in regions where clocks are slower.

4.5 Nonuniform Node Distribution

We have considered uniform distribution of nodes in all of the experiments presented above. In this section, we consider nonuniform node distribution in the network field. HEED primarily elects cluster heads according to their residual energy, which is independent of node distribution. If nodes with high residual energy are all clustered in one region in the network, the design of HEED which relies on using an intracluster power level during clustering reduces the likelihood that cluster heads are neighbors within the cluster range. Based upon this, we conjecture that node distribution does not impact the quality of clustering, in terms of the residual energy of cluster heads, and their distribution in the field. Of course, nonuniform node distribution may result in an increase in the variance of the number of nodes per cluster, which is inevitable since we only use one cluster range.

To verify this conjecture, we conduct an experiment in which the network is divided into four areas (regions) of equal sizes $(A_1, \ldots, A_4)$, and the probability of a node residing in each of the areas is 3 percent for A_1, 5 percent for A_2, 10 percent for A_3, and 82 percent for A_4. We use the same simulation settings as in Section 4 and compare to a generic clustering protocol. Fig. 7a shows that the average percentage of cluster heads is much lower in the
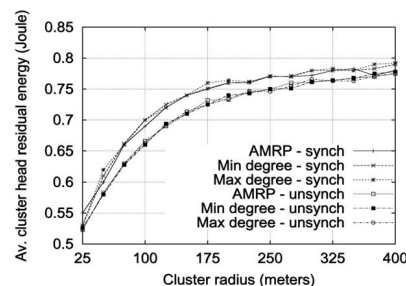

Fig. 6. HEED average cluster head energy for synchronized and pseudosynchronized nodes.
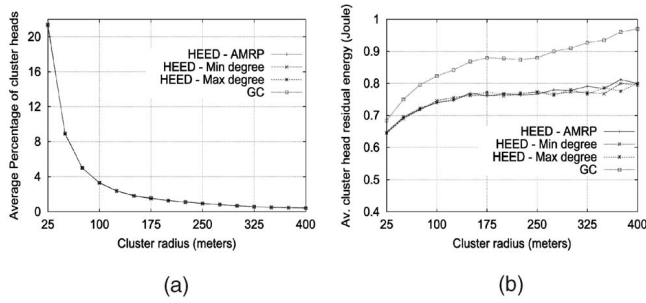
Fig. 7. Clustering quality with nonuniform node distribution. (a) Average number of cluster heads. (b) Average residual energy per cluster head.

nonuniform case than in the uniform case. This is at the expense of a much higher variance in the number of nodes per cluster. Fig. 7b shows that the average residual energy per cluster head is still as high as that in the uniform node distribution case.

5 CLUSTERING APPLICATIONS

Our approach can be used for constructing energy-efficient hierarchies for routing protocols, in which higher tier nodes should have more residual energy. Our approach can also be effective for sensor applications requiring efficient data aggregation and prolonged network lifetime, such as environmental monitoring applications. We consider one such application (similar to the one described in [8]) in this section. Cluster heads in our application do not consume similar amount of energy during every T_{NO} interval, as assumed in [8].

In [8], a distributed clustering protocol for microsensor networks (LEACH) was introduced for prolonging the network lifetime. LEACH was proposed for an application in which sensor nodes are randomly distributed on a grid-like area and are continuously sensing the environment to send reports to a remote sink (e.g., observer/base station). The application assumes that nodes are equally significant and data aggregation is possible. LEACH clustering proved to be $4 \times$ to $8 \times$ more effective in prolonging the network lifetime than direct communication or minimum energy transfer (shortest path multihop routing).

In LEACH, a node elects to become a cluster head randomly according to a target number of cluster heads in the network and its own residual energy. Clustering starts by computing the optimal number of clusters in the network. When clustering is triggered, certain nodes broadcast their willingness to become cluster heads, and regular nodes join clusters according to cluster head proximity. Each cluster head then creates a TDMA schedule for its nodes and broadcasts it. Every node sends its data to its cluster head according to the specified TDMA schedule. Direct Sequence Spread Spectrum (DSSS) codes are used to minimize intercluster interference (therefore, we ignore collisions in our simulation). Each cluster head fuses the data it receives from its nodes into one frame and sends it to the sink. Clustering is triggered every T_{NO} TDM frames.

It is easy to see that, under optimal conditions (no interference or data losses), the maximum network lifetime occurs at the minimum possible choice of T_{NO} (i.e., for $T_{NO} = 1$) if the clustering overhead is incomparable to the application load. However, such small values of T_{NO} cause

TABLE 2
Simulation Parameters

Type	Parameter	Value
Network	Network grid	From (0,0) to (100,100)
	Sink	At (50,175)
	Initial energy	2 J/battery
Application	Cluster radius	25 m
	Data packet size	100 bytes
	Broadcast packet size	25 bytes
	Packet header size	25 bytes
	Round (T_{NO})	5 TDM frames
Radio model	E_{elec}	50 nJ/bit
	ϵ_{fs}	10 $pJ/bit/m^2$
	ϵ_{mp}	0.0013 $pJ/bit/m^4$
	E_{fusion}	5 $nJ/bit/signal$
	Threshold distance (d_0)	75 m

the system to be always in an unstable state, which might lead to undesirable effects, such as excessive interference, data losses, and delayed response. Thus, T_{NO} can be in the range of seconds for applications where all nodes are continuously sending reports, and a cluster head consumes a significant portion of its energy in serving its cluster members. On the other hand, for data-driven applications (where reports are sent upon request), and the aggregation and forwarding processes are not very expensive, T_{NO} can be in the range of minutes or even hours.

We compare our HEED clustering to a generalized LEACH (gen-LEACH) approach in which two features are added to the application-specific LEACH protocol, described in [8]. First, the routing protocol is assumed to propagate node residual energy throughout the network. Although this approach requires extensive message exchange (for residual energy information), it selects better cluster heads than the original LEACH and, thus, prolongs the network lifetime (this approach was proposed in the code released by the authors of [8]). A node executing gen-LEACH elects itself to become a cluster head at time t with probability $CH_{prob}(t)$, where $CH_{prob}(t) = \min(\frac{E_i(t)}{E_{total}} \times k, 1)$. Here, E_i is the residual energy of node i, and $E_{total} = \sum_{i=1}^{N} E_i(t)$. Second, a node selects a cluster head in its cluster range proximity, which is not assumed to span the entire network area. This generalizes LEACH for multihop networks.

Most of our simulation parameters are similar to those in [8]. The parameters are listed in Table 2. In the simple radio model that we use, energy is expended to serve: 1) digital electronics, E_{elec}, (actuation, sensing, signal emission/reception), and 2) communication, E_{amp}. E_{amp} varies according to the distance d between a sender and a receiver: $E_{amp} = \epsilon_{fs}$ assuming a free space model when $d < d_0$, while $E_{amp} = \epsilon_{mp}$ assuming a multipath model when $d \geq d_0$, where d_0 is a constant distance that depends on the environment. To transmit n_b bits for a distance d, the radio expends $n_b(E_{elec} + E_{amp} \times d^n)$ J, where $n = 2$ for $d < d_0$, and $n = 4$ for $d \geq d_0$. To receive n_b bits at the receiver, the radio expends $n_b \times E_{elec}$ J. This energy model assumes a continuous function for energy consumption.
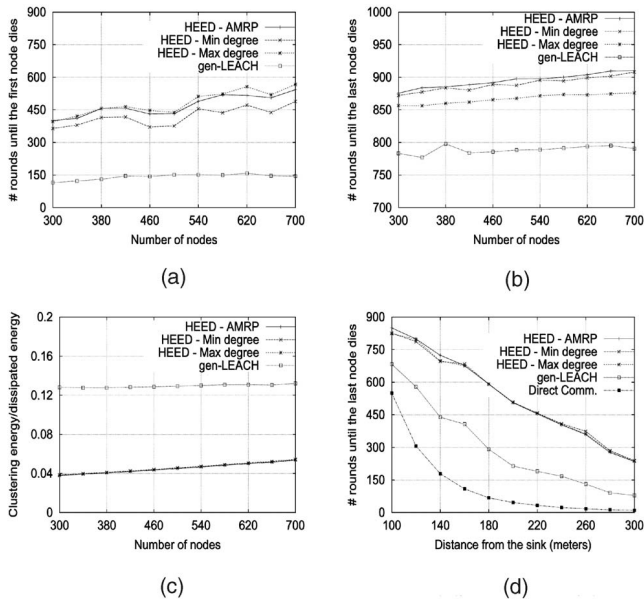
Fig. 8. Performance of HEED on network applications. (a) Network lifetime (first node death). (b) Network lifetime (last node death). (c) Ratio of energy used in clustering to total dissipated energy. (d) Network lifetime as the sink travels farther.

A node is considered "dead" if it has lost 99.9 percent of its initial energy. For HEED, 5 percent is used as an initial tentative percentage of cluster heads (C_{prob}). For gen-LEACH, k_{opt} was selected to be 11 for 300-700 node networks, which falls in the range of k_{opt} computed according to [8]. Fig. 8a compares network lifetime with HEED to gen-LEACH, where network lifetime is the time until the *first* node dies. HEED clustering clearly improves network lifetime over gen-LEACH clustering for all cost types. This is because gen-LEACH randomly selects cluster heads (and, hence, cluster sizes), which may result in faster death of some nodes. This is avoided in HEED because final cluster heads are selected such that they are well-distributed across the network and communication cost is minimized. Similar results are obtained for the number of rounds until *last* node death as shown in Fig. 8b.

We also measure the energy consumed in clustering as a fraction of the total dissipated energy in the network. For gen-LEACH, we assume that, at the end of each round, each node sends its residual energy information to its cluster head, which aggregates this information and broadcasts it across the network using only one message. Fig. 8c illustrates the energy ratio for different numbers of nodes (the results of the three HEED cost types are almost superimposed). HEED expends less energy in clustering than gen-LEACH because gen-LEACH propagates residual energy information. It is also worth mentioning that we found that the original LEACH protocol expends less energy in clustering and produces longer lifetime than both HEED and gen-LEACH when used specifically for the application described in [8], and under the assumptions made there. This is intuitive since HEED will produce only one cluster head for the entire network if we assume that every node can reach all other nodes in the network in one hop (very large R_c).

Finally, we study the effect of the distance between the sink and the network on the network lifetime (using the "last node death" definition of network lifetime). In this experiment, we compute the number of rounds in which the network was alive using different HEED cost types, gen-LEACH, and direct communication. We fix the *x*-coordinate of the sink and varied its height (*y*-coordinate). The distance is computed from the sink to the closest point to it on the network. The number of nodes was fixed at 500. Fig. 8d shows that HEED prolongs network lifetime, compared to gen-LEACH and to direct communication. Network lifetime severely deteriorates when using direct communication as the distance increases, which emphasizes the advantages of network clustering. Direct communication to long distances also results in severe interference problems, especially in dense networks. Using direct communication may be tolerable only in when the sink is very close to the data source in the network (which is not the case in most applications), to avoid clustering overhead.

6 DISCUSSION

In this section, we discuss a number of possible extensions for practical deployment of HEED in different environments.

6.1 Intracluster and Intercluster Routing

In the description of HEED operation, we assumed single-hop communication among cluster heads and their registered cluster members. This is desirable in source-driven networks, where reports are periodically transmitted by the sensor nodes. In this case, a TDM frame may be constructed at each cluster head to eliminate interference within a cluster. Clearly, constructing TDM frames requires node synchronization, and in lightly-loaded networks, using TDM frames may waste resources. A better approach in this case is to allow channel contention. Multihop routing to the cluster head can increase network capacity in this case. The reader should refer to [29], [19], [30] for detailed studies addressing the issue of single-hop versus multihop routing in clustered networks.

Cluster head overlay (i.e., intercluster) routes are used to communicate among clusters, or between clusters and the observer(s). In this case, an ad hoc routing protocol, such as Directed Diffusion [5] or Dynamic Source Routing (DSR) [31], can be employed for data forwarding among cluster heads. TinyOS beaconing is the approach currently specified for sensors running TinyOS. This constructs a breadth-first spanning tree rooted at the base station. In a clustered network, the beaconing approach can be applied to only the cluster head overlay, instead of the entire network.

If two regular nodes from different clusters attempt to communicate, communication through their cluster heads is suboptimal if the two regular nodes can directly communicate via a shorter path. This, however, is not the typical communication pattern for sensor network applications, where data is transmitted to an observer which is not close to the target source of data, and data may be aggregated by cluster heads. In addition, since the cluster range is typically limited (compared to the network size), the network can be approximately viewed as a grid-like area, where optimal routes along the grid are computed using routing tables or through reactive routing techniques.

6.2 Selecting Transmission Ranges

Careful selection of the intercluster transmission range (R_t) and the intracluster transmission range (R_c) is crucial for maintaining network connectivity (as discussed in Section 3.4). Reducing interference, maximizing network capacity (concurrent transmissions), and reducing energy consumption are also important objectives to consider when selecting these ranges. Since requirements and transmission patterns (query-based data-driven versus source-driven) widely vary for different applications, determining transmission ranges must be performed on a per-application basis. The network density, radio model, and available number of power levels are constraints that affect the selection process.

A key concern that is common to all applications is that the cluster head overlay and, consequently, the entire network, remain connected. This can be achieved if the relationship between the number of nodes in the cluster head overlay n_0, and the intercluster transmission range R_t satisfy the connectivity condition specified in [32] for unit square region:

$$R_t^2 \sim \frac{log\ n_0}{n_0}. \tag{4}$$

More generally, assuming that a node is active with probability p, the necessary condition for connectivity and coverage is $R_t^2 \geq \frac{c\ log\ n_0}{p\ n_0}$, where $c = \frac{1}{\pi\beta^2}$, and $\beta \leq 0.5$ [33].

Therefore, a simple process for selecting transmission ranges in a clustered network may proceed as follows: The cluster range R_c is selected, say as the median range in the set of ranges $\{R_{min}, \ldots, R_{max}\}$ that are available at any node. The selected R_c must have a corresponding R_t that satisfies the connectivity requirements specified in Section 3.4. R_c can also be selected to limit the number of nodes in a cluster, assuming that the network area, A, is known and nodes are uniformly distributed in the field. Using these two assumptions, the number of nodes in the cluster head overlay (i.e., the number of clusters), n_0, can be computed as $n_0 = \frac{A}{\pi \times R_c^2}$. If the pair (R_t, n_0) satisfies (4), then the pair (R_c, R_t) is a viable transmission range pair for the clustered network. If the process fails, it must be repeated for a smaller R_c until a viable pair is found.

A method to compute the optimal number of clusters in a sensor network was presented in [8]. The goal of that study was to minimize energy dissipation and, consequently, prolong the network lifetime. However, their analysis is specific to the scenario they study in [8], which assumes single-hop transmission is always possible. Selecting the transmission ranges for optimizing a system objective, such as maximizing the network lifetime, is left for future work. This paper only focuses on designing mechanisms for clustering the network for a given (R_c, R_t) pair.

6.3 Fault Tolerance

HEED clustering is periodically triggered in order to distribute energy consumption among sensor nodes. Reclustering also provides fault tolerance against unexpected failures, especially failures of cluster heads. In hostile environments (such as military fields), however, unexpected failures may be frequent. This may cause parts of the network to be unreachable. Reclustering frequency has to be carefully selected in this case to withstand expected failure rates. This is practically difficult for two reasons: First, the failure rates in hostile environments are usually unpredictable and highly variable. This means that frequent reclustering may result in significant resource waste if the failure rate is low most of the time. Second, frequent reclustering is not always feasible since it limits the time a sensor is "available" to conduct its primary operations (sensing and data communication), and increases the need for node synchronization.

An alternative to frequent clustering is to maintain backup cluster heads. This mitigates the single point of failure problem at each cluster head since a node can find an alternative path to the observer(s) if its cluster head fails. Finding backup cluster heads that are able to cover the entire cluster (i.e., act as cluster heads for all nodes in the original cluster whose head failed) may not always be feasible, however. A solution to this problem is to construct multiple (say k) node-disjoint overlays of cluster heads, assuming node density allows this. In this case, k-connected graphs can be constructed, where k is an environment-dependent constant specified by the application. If k-connectivity must be guaranteed, we need a density model different from the one presented in Section 3.4, since at least k nodes per cell are required in this case. We plan to investigate the design of fault tolerant clustering mechanisms for ad hoc sensor networks in our future research.

7 RELATED WORK

Many protocols have been proposed for ad hoc and sensor networks in the last few years. Reducing energy consumption due to wasteful sources has been primarily addressed in the context of adaptive MAC protocols, such as PAMAS [34], DBTMA [35], EAR [36], and S-MAC [37]. For example, S-MAC [37] periodically puts nodes to sleep to avoid idle listening and overhearing. TinyOS [38] introduces random delays to break synchronization. Blue Noise Sampling [39] selects well-distributed nodes to awaken in order to achieve optimal field coverage.

Data dissemination protocols proposed for sensor networks consider energy efficiency a primary goal [6], [5], [40], [7]. SPIN [6] attempts to reduce the cost of flooding data, assuming that the network is source-centric (i.e., sensors announce any observed event to interested observers). Directed diffusion [5], on the other hand, selects the most efficient paths to forward requests and replies on, assuming that the network is data-centric (i.e., queries and data are forwarded according to interested observers). Rumor routing [40] provides a compromise between the two approaches (source-centric versus data-centric). In [7], the dissemination problem is formulated as a linear programming problem with energy constraints. This approach assumes *global* knowledge of node residual energy, and requires sensors with specific processing capabilities. In [41], a disjoint path routing scheme is proposed in which energy efficiency is the main parameter.

Clustering can be a side effect of other protocol operations. For example, in topology management protocols, such as GAF [10], SPAN [11], and ASCENT [9], nodes are classified according to their geographic location into equivalence classes. A fraction of nodes in each class (representatives) participate in the routing process, while other nodes are turned off to save energy. In GAF, geographic information is assumed to be available based on a positioning system such as GPS. SPAN infers

geographic proximity through broadcast messages and routing updates. GAF, SPAN, and ASCENT share the same objective of using redundancy in sensor networks to turn radios on and off and prolong network lifetime. In CLUSTERPOW [3], nodes are assumed to be nonhomogeneously dispersed in the network. A node uses the minimum possible power level to forward data packets, in order to maintain connectivity while increasing the network capacity and saving energy. The Zone Routing Protocol (ZRP) [42] for MANETs divides the network into overlapping, variable-sized zones.

Several distributed clustering approaches have been proposed for mobile ad hoc networks and sensor networks. The Distributed Clustering Algorithm (DCA) [16] assumes quasi-stationary nodes with real-valued weights. The Weighted Clustering Algorithm (WCA [17]) combines several properties in one parameter (weight) that is used for clustering. In [13], the authors propose using a spanning tree (or BFS tree) to produce clusters with some desirable properties. Energy efficiency, however, is not the primary focus of this work. In [15], the authors propose passive clustering for use with on-demand routing in ad hoc networks. Earlier work also proposed clustering based on degree (connectivity) or lowest identifier heuristics [12]. Clustering time complexity in all of the above approaches is dependent on the network diameter, unlike HEED which terminates in a constant number of iterations.

LEACH clustering [8] terminates in a constant number of iterations (like HEED), but it does not guarantee good cluster head distribution and assumes uniform energy consumption for cluster heads. In [19], the authors use LEACH-like randomized clustering, and provide methods to compute the optimal values of the algorithm parameters a priori and use multihop forwarding for intracluster and intercluster communications. In [43], a multilevel hierarchical structure is proposed, where cluster heads are selected according to their residual energy and degree. ACE [44] clusters the sensor network in a constant number of iterations using the node degree as the main parameter. The approach in [20] selects a d-hop dominating set in $O(d)$ time to cluster the network based on node ID, while the approach in [45] selects a dominating set in constant time using linear programming relaxation techniques. In [29], the authors study the effect of different communication paradigms (single hop versus multihop) on the performance of clustering protocols. Finally, a number of approaches construct a clustered network in order to optimize routing while supporting mobility, e.g., [14].

8   CONCLUSIONS AND FUTURE WORK

In this paper, we have presented a distributed, energy-efficient clustering approach for ad hoc sensor networks. Our approach is hybrid: Cluster heads are probabilistically selected based on their residual energy, and nodes join clusters such that communication cost is minimized. We assume quasi-stationary networks where nodes are location-unaware and have equal significance. A key feature of our approach is that it exploits the availability of multiple transmission power levels at sensor nodes.

Based on this approach, we have introduced the HEED protocol, which terminates in a constant number of iterations, independent of network diameter. Simulation results demonstrate that HEED prolongs network lifetime, and the clusters it produces exhibit several appealing

characteristics. HEED parameters, such as the minimum selection probability and network operation interval, can be easily tuned to optimize resource usage according to the network density and application requirements. HEED achieves a connected multihop intercluster network when a specified density model and a specified relation between cluster range and transmission range hold.

Our approach can be applied to the design of several types of sensor network protocols that require scalability, prolonged network lifetime, fault tolerance, and load balancing. Although we have only provided algorithms for building a two-level hierarchy, we can extend the protocols to multilevel hierarchies. This can be achieved by recursive application at upper tiers using bottom-up cluster formation [19].

REFERENCES

[1] D. Estrin, L. Girod, G. Pottie, and M. Srivastava, "Instrumenting the World with Wireless Sensor Networks," *Proc. Int'l Conf. Acoustics, Speech, and Signal Processing (ICASSP 2001),* May 2001, http://citeseer.nj.nec.com/estrin01instrumenting.html.

[2] G.J. Pottie and W.J. Kaiser, "Wireless Integrated Newtork Sensors," *Comm. ACM,* vol. 43, no. 5, pp. 51-58, May 2000.

[3] V. Kawadia and P.R. Kumar, "Power Control and Clustering in Ad Hoc Networks," *Proc. IEEE INFOCOM,* Apr. 2003.

[4] S. Narayanaswamy, V. Kawadia, R.S. Sreenivas, and P.R. Kumar, "Power Control in Ad-Hoc Networks: Theory, Architecture, Algorithm and Implementation of the COMPOW protocol," *Proc. European Wireless 2002. Next Generation Wireless Networks: Technologies, Protocols, Services and Applications,* pp. 156-162, Feb. 2002.

[5] C. Intanagonwiwat, R. Govindan, and D. Estrin, "Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks," *Proc. ACM/IEEE Int'l Conf. Mobile Computing and Networking (MOBICOM),* 2000.

[6] J. Kulik, W. Heinzelman, and H. Balakrishnan, "Negotiation-Based Protocols for Disseminating Information in Wireless Sensor Networks}," *ACM Wireless Networks,* vol. 8, nos. 2-3, pp. 169-185, 2002, citeseer.nj.nec.com/kulik99negotiationbased. html.

[7] J.-H. Chang and L. Tassiulas, "Energy Conserving Routing in Wireless Ad-Hoc Networks," *Proc. IEEE INFOCOM,* Mar. 2000, http://www.ieee-infocom.org/2000/papers/417.ps.

[8] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "An Application-Specific Protocol Architecture for Wireless Microsensor Networks," *IEEE Trans. Wireless Comm.,* vol. 1, no. 4, pp. 660-670, Oct. 2002.

[9] A. Cerpa and D. Estrin, "ASCENT: Adaptive Self-Configuring Sensor Networks Topologies," *Proc. IEEE INFOCOM,* June 2002.

[10] Y. Xu, J. Heidemann, and D. Estrin, "Geography-Informed Energy Conservation for Ad Hoc Routing," *Proc. ACM/IEEE Int'l Conf. Mobile Computing and Networking (MOBICOM),* pp. 70-84, July 2001.

[11] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris, "Span: An Energy-Efficient Coordination Algorithm for Topology Maintenance in Ad Hoc Wireless Networks," *ACM Wireless Networks,* vol. 8, no. 5, Sept. 2002.

[12] C.R. Lin and M. Gerla, "Adaptive Clustering for Mobile Wireless Networks," *IEEE J. Selected Areas Comm.,* Sept. 1997.

[13] S. Banerjee and S. Khuller, "A Clustering Scheme for Hierarchical Control in Multi-Hop Wireless Networks," *Proc. IEEE INFOCOM,* Apr. 2001.

[14] B. McDonald and T. Znati, "Design and Performance of a Distributed Dynamic Clustering Algorithm for Ad-Hoc Networks," *Proc. Ann. Simulation Symp.,* 2001.

[15] M. Gerla, T.J. Kwon, and G. Pei, "On Demand Routing in Large Ad Hoc Wireless Networks with Passive Clustering," *Proc. IEEE Wireless Comm. & Networking Conf. (WCNC),* 2000.

[16] S. Basagni, "Distributed Clustering Algorithm for Ad-Hoc Networks," *Proc. Int'l Symp. Parallel Architectures, Algorithms, and Networks (I-SPAN),* 1999.

[17] M. Chatterjee, S.K. Das, and D. Turgut, "WCA: A Weighted Clustering Algorithm for Mobile Ad Hoc Networks," *Cluster Computing,* pp. 193-204, 2002.

[18] T.J. Kwon and M. Gerla, "Clustering with Power Control," *Proc. MilCOM '99,* 1999.

[19] S. Bandyopadhyay and E. Coyle, "An Energy-Efficient Hierarchical Clustering Algorithm for Wireless Sensor Networks," *Proc. IEEE INFOCOM,* Apr. 2003.

[20] A.D. Amis, R. Prakash, T.H.P. Vuong, and D.T. Huynh, "Max-Min D-Cluster Formation in Wireless Ad Hoc Networks," *Proc. IEEE INFOCOM,* Mar. 2000.

[21] O. Younis and S. Fahmy, "Distributed Clustering in Ad-Hoc Sensor Networks: A Hybrid, Energy-Efficient Approach," *Proc. IEEE INFOCOM,* Mar. 2004.

[22] B. Shirazi, A.R. Hurson, and K.M. Kavi, *Scheduling and Load Balancing in Parallel and Distributed Systems.* IEEE CS Press, 1995.

[23] F. Ye, H. Luo, J. Chung, S. Lu, and L. Zhang, "A Two-Tier Data Dissemination Protocol for Large-Scale Wireless Sensor Networks," *Proc. ACM/IEEE Int'l Conf. Mobile Computing and Networking (MOBICOM),* Sept. 2002.

[24] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D.E. Culler, and K.S.J. Pister, "System Architecture Directions for Networked Sensors," *Architectural Support for Programming Languages and Operating Systems,* pp. 93-104, 2000.

[25] J. Elson, L. Girod, and D. Estrin, "Fine-Grained Network Time Synchronization Using Reference Broadcasts," *Proc. Symp. Operating Systems Design and Implementation (OSDI),* 2002.

[26] F. Ye, G. Zhong, S. Lu, and L. Zhang, "PEAS: A Robust Energy Conserving Protocol for Long-Lived Sensor Networks," *Proc. Int'l Conf. Distributed Computing Systems (ICDCS),* 2003.

[27] D.M. Blough and P. Santi, "Investigating Upper Bounds on Network Lifetime Extension for Cell-Based Energy Conservation Techniques in Stationary Ad Hoc Networks," *Proc. ACM/IEEE Int'l Conf. Mobile Computing and Networking (MOBICOM),* 2002.

[28] W.C.Y. Lee, *Mobile Cellular Telecommunications.* McGraw-Hill, 1995.

[29] V. Mhatre and C. Rosenberg, "Design Guidelines for Wireless Sensor Networks Communication: Clustering and Aggregation," *Ad-Hoc Networks J.,* to appear.

[30] S. Lindsey and C.S. Raghavendra, "PEGASIS: Power-Efficient Gathering in Sensor Information Systems," *Proc. Int'l Conf. Comm. (ICC '01),* 2001.

[31] D.B. Johnson and D.A. Maltz, "Dynamic Source Routing in Ad Hoc Wireless Networks," *Mobile Computing,* vol. 353, 1996, citeseer.ist.psu.edu/johnson96dynamic.html.

[32] P. Gupta and P.R. Kumar, "Critical Power for Asymptotic Connectivity in Wireless Networks," *Stochastic Analysis, Control, Optimizations, and Applications: A Volume in Honor of W.H. Fleming, W.M. McEneaney, G. Yin, and Q. Zhang (Eds.),* 1998.

[33] S. Shakkottai, R. Srikant, and N. Shroff, "Unreliable Sensor Grids: Coverage, Connectivity and Diameter," *Proc. IEEE INFOCOM,* Mar. 2003, citeseer.ist.psu.edu/shakkottai03unreliable.html.

[34] S. Singh and C. Raghavendra, "PAMAS: Power Aware Multi-Access protocol with Signalling for Ad Hoc Networks," *ACM Computer Comm. Rev.,* vol. 28, no. 3, pp. 5-26, July 1998, citeseer.nj.nec.com/460902.html,

[35] Z. Haas and S. Tabrizi, "On Some Challenges and Design Choices in Ad-Hoc Communications," *Proc. IEEE MILCOM '98,* Oct. 1998.

[36] K. Sohrabi, J. Gao, V. Ailawadhi, and G.J. Pottie, "Protocols for Self-Organization of a Wireless Sensor Network," *IEEE Personal Comm. Magazine,* vol. 7, no. 5, Oct. 2000, http://www.comsoc.org/pci/private/2000/oct/pottie.html.

[37] W. Ye, J. Heidenmann, and D. Estrin, "An Energy-Efficient MAC Protocol for Wireless Sensor Networks," *Proc. IEEE INFOCOM,* June 2002.

[38] A. Woo and D. Culler, "A Transmission Control Scheme for Media Access in Sensor Networks," *Proc. ACM/IEEE Int'l Conf. Mobile Computing and Networking (MOBICOM),* 2001, http://today.cs.berkeley.edu/tos/media.html.

[39] M. Perillo, Z. Ignjatovic, and W. Heinzelman, "An Energy Conservation Method for Wireless Sensor Networks Employing a Blue Noise Spatial Sampling Technique," *Proc. Third Int'l Symp. Information Processing in Sensor Networks (IPSN '04),* Apr. 2004.

[40] D. Braginsky and D. Estrin, "Rumor Routing Algorithm for Sensor Networks," *Proc. First Workshop on Sensor Networks and Application (WSNA '02),* Sept. 2002.

[41] A. Srinivas and E. Modiano, "Minimum Energy Disjoint Path Routing in Wireless Ad-Hoc Networks," *Proc. ACM/IEEE Int'l Conf. Mobile Computing and Networking (MOBICOM),* Sept. 2003.

[42] Z.J. Haas, M.R. Pearlman, and P. Samar, "The Zone Routing Protocol (ZRP) for Ad-Hoc Networks," Internet Draft, draft-ietf-manet-zone-zrp-04.txt, July 2002.

[43] D. Estrin, R. Govindan, J. Heidemann, and S. Kumar, "Next Century Challenges: Scalable Coordination in Sensor Networks," *Proc. ACM/IEEE Int'l Conf. Mobile Computing and Networking (MOBICOM),* Aug. 1999.

[44] H. Chan and A. Perrig, "ACE: An Emergent Algorithm for Highly Uniform Cluster Formation," *Proc. First European Workshop Sensor Networks (EWSN),* Jan. 2004.

[45] F. Kuhn and R. Wattenhofer, "Constant-Time Distributed Dominating Set Approximation," *Proc. ACM Symp. Principles of Distributed Computing (PODC),* July 2003.

Ossama Younis (S'02) received the BS and MS degrees from the Computer Science Department, Alexandria University, Egypt, in 1995 and 1999, respectively. Since 2000, he has been pursuing his PhD degree in the Department of Computer Sciences, Purdue University. His current research interests include sensor networks, Internet routing and tomography, and network security. He is a student member of the IEEE.

Sonia Fahmy (S'96-A'00-M'03) received the PhD degree from The Ohio State University in August 1999. Since then, she has been an assistant professor in the Computer Sciences Department at Purdue University. Her research interests are in the design and evaluation of network architectures and protocols. She is currently investigating Internet tomography, overlay networks, network security, and wireless sensor networks. Please see http://www.cs.purdue.edu/homes/fahmy/ for more information. She is a member of the IEEE and the IEEE Computer Society.

▷ **For more information on this or any other computing topic, please visit our Digital Library at** www.computer.org/publications/dlib.