Robust Communications for Sensor Networks in Hostile Environments

Ossama Younis and Sonia Fahmy
Department of Computer Sciences, Purdue University
250 N. University Street, West Lafayette, IN 47907–2066, USA
E-mail: {oyounis,fahmy}@cs.purdue.edu

Paolo Santi
Instituto di Informatica e Telematica, CNR
Via G. Moruzzi 1, 56124 Pisa, Italy
E-mail: paolo.santi@iit.cnr.it

***Abstract*— Clustering sensor nodes increases the scalability and energy efficiency of communications among them. In hostile environments, unexpected failures or attacks on cluster heads (through which communication takes place) may partition the network or degrade application performance. In this work, we propose a new approach, REED (Robust Energy Efficient Distributed clustering), for clustering sensors deployed in hostile environments. Our primary objective is to construct a k-fault-tolerant (i.e., k-connected) network, where k is a constant determined by the application. Fault tolerance can be achieved by selecting k independent sets of cluster heads (i.e., cluster head overlays) on top of the physical network, so that each node can quickly switch to other cluster heads in case of failures or attacks on its current cluster head. The independent cluster head overlays also provide multiple vertex-disjoint routing paths for load balancing and security. Network lifetime is prolonged by selecting cluster heads with high residual energy and low communication cost, and periodically re-clustering the network in order to distribute energy consumption among sensor nodes. We prove that REED can asymptotically achieve k-fault tolerance if certain constraints on node density are satisfied. We also investigate via simulations the clustering properties of REED, and show that building multiple cluster head overlays does not consume significant energy.**

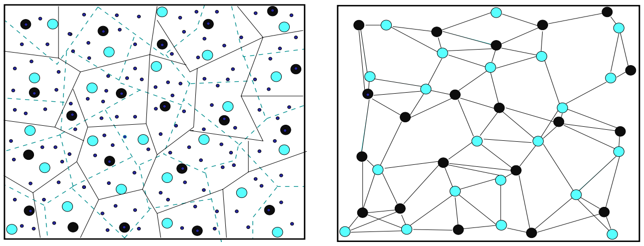
I. Introduction

Sensor networks are emerging as versatile computing platforms for several monitoring and control applications. Sensor networks are typically deployed and left unattended under possibly harsh conditions, e.g., in volcanic areas or battle fields. Operating in hostile environments necessitates devising efficient network self-organization techniques for providing the applications with a robust and fault-tolerant computing and communication environment. Energy efficiency is also critical. For example, consider a sensor network for seismic monitoring, battle field surveillance, or radiation level control in a nuclear plant. In such applications, the lifetime of each sensor significantly impacts the quality of surveillance. Since re-charging batteries is infeasible in this case, energy consumption must be minimized to prolong the lifetime of individual sensors, and consequently the sensor network.

Clustering is an effective self-organization technique that can prolong the network lifetime. In a clustered network, a set of cluster heads is selected from sensor nodes. The remaining nodes, which we refer to as *regular nodes*, register themselves with one or more cluster heads. A cluster head is responsible for: (i) communicating with its registered cluster nodes (e.g., using a TDMA schedule if the network is source-driven) typically via a single hop or a few hops, and (ii) communicating with other cluster heads or with the observer(s) on behalf of its cluster. This communication can be single-hop or multi-hop. Clustering prolongs the network lifetime by reducing contention on transmission channels and supporting data aggregation at cluster heads, e.g., computing the maximum temperature in a field. In addition, cluster heads rotate their functionality to distribute energy consumption. Clustering is thus vital for efficient resource utilization and load balancing in large scale networks [1], [2], [3], [4], [5].

Failures of cluster heads, however, may isolate their cluster nodes, partitioning the network until the next clustering process is triggered. In addition, in environments with malicious attackers, cluster heads can be compromised. The primary goal of this work is to achieve fault tolerance by providing alternate routing paths from sources to observers. Towards this end, we propose REED (Robust, Energy Efficient, Distributed clustering)– a protocol to build multiple independent cluster head overlays on top of the physical network. These cluster head overlays (i.e., sets of cluster heads and their associated nodes) provide means for every node to automatically and independently switch cluster heads on detecting a cluster head failure. REED has low asymptotic computation and communication overhead, as will be shown in Section IV. We use HEED clustering [1] as the underlying clustering approach for REED because of its generality and energy efficiency. However, any energy efficient sensor clustering approach, such as LEACH [2], can employ our multiple overlay idea, as discussed in Section VII. Multiple overlays can also be combined with any clustering protocol, such as [5], [6], [7], for fault-tolerant communication.

We prove that REED constructs asymptotically k-fault-tolerant clustered networks if certain constraints on node density are satisfied. A graph $G = (V, E)$ is k-fault-tolerant (i.e., k-connected) if it remains connected under failures of up to $k - 1$ nodes, where k is a constant determined by the application. Fig. 1(a) illustrates a possible organization of a 2-fault-tolerant clustered network. In this organization, two cluster head overlays are built on top of the physical network. Each node belongs to 2 clusters in 2 separate cluster head overlays, but only uses one at a time. To the best of our knowledge, no practical protocols have been proposed



(a) Two cluster head overlays. Larger circles denote cluster heads.

(b) A 2-connected inter-cluster network graph. Only cluster heads are shown.

Fig. 1. A network with 2 cluster head overlays.

in the literature to construct k-fault tolerant clustered sensor networks.

An additional advantage of forming multiple cluster head overlays is the possibility of multi-path communication. Security protocols, such as Multi-path Key Reinforcement [8], can exploit multiple paths for securing key re-distribution, in conjunction with protocols that use threshold cryptography [9]. The availability of multiple vertex-disjoint inter-cluster routing paths depends on the node density in the network, and the available transmission power levels. A sensor node typically has a number of discrete transmission power levels that correspond to a number of transmission ranges. In REED, nodes use a transmission range R_c as the cluster range (for cluster formation and intra-cluster communication), and reserve a higher transmission range R_t for inter-cluster communication. Fig. 1(b) illustrates how two inter-cluster vertex-disjoint paths (among dark-colored nodes versus among light-colored nodes, in addition to paths that include both) can be available in a clustered network.

The remainder of this paper is organized as follows. Section II defines the system model and states the protocol requirements. Section III briefly surveys related work. Section IV presents our proposed protocol, REED, and argues that it satisfies its goals. Section V demonstrates the effectiveness of REED via simulations. Section VI discusses how REED prolongs the network lifetime and allows multi-path routing. Section VII shows the application of REED on LEACH as the underlying clustering protocol. Finally, Section VIII gives concluding remarks and directions for future work.

II. System Model and Objectives

Consider a set of sensor nodes dispersed uniformly and independently at random on a field. The network exhibits the following properties: (1) nodes are quasi-stationary, which is typical for most sensor network applications, (2) links are symmetric, i.e., two nodes can communicate using the same transmission power, (3) all nodes have similar processing and communication capabilities and equal significance, (4) the network may serve multiple mobile or stationary observers. Therefore, energy consumption is not uniform for all nodes, (5) nodes are deployed in a hostile environment which results in unexpected node failures. The definition of failure rate depends on the network application. For example, failure may depend on time, network density, or human intervention, (6) nodes are not location-aware and are left unattended after deployment, and (7) each node has a number (at least 2 for R_c and R_t) of transmission power levels. An example of such sensor nodes are Berkeley Motes [10]. It is typically straightforward to set the transmission power level via the standard *ioctl()* system call.

Our primary objective is to design a distributed algorithm for constructing a fault-tolerant clustered network. Ideally, each node should be able to directly communicate with at least one cluster head using the intra-cluster range R_c, even when a number of cluster heads fail. The following requirements should be met:

1) The clustered network should be k-fault-tolerant if density constraints are satisfied (as defined in Section IV-D). In a k-fault-tolerant network, the network remains connected even under the failure of up to $k-1$ nodes. If density constraints cannot be met, the protocol should degrade gracefully.
2) Additional basic requirements are that: (i) clustering is fully distributed, (ii) clustering terminates within a fixed number of iterations (regardless of network diameter), (iii) clustering is efficient in terms of processing complexity and message exchange, and (iv) cluster heads have relatively high residual energy.

III. Related Work

Clustering protocols have been previously investigated as either independent protocols for organizing ad-hoc networks, e.g., [5], [7], [6], [11], or in the context of routing and power management, e.g., [12], [13], [2], [3], [14]. Clustering protocols that use a generic parameter(s) or weight associated with each node to make clustering decisions are known as weight-based clustering protocols. Examples of weight-based clustering protocols include [7], [1], [2]. In [6], the authors propose using a spanning tree (or BFS tree) to produce clusters. Earlier work also proposed clustering based on degree or lowest identifier heuristics [13]. CLUSTERPOW [12] proposes that a node use the minimum possible power level to forward data packets to the next hop, in order to maintain connectivity while increasing the network capacity and saving energy.

Most of the previously proposed clustering protocols have a time complexity dependent on the network diameter. For the protocols that are independent of network diameter, such as [2], [3], important requirements, such as good cluster head distribution across the network, cannot be met. Therefore, in our previous work, we have developed HEED (Hybrid Energy Efficient Distributed clustering) [1]– a low complexity clustering protocol that aims at prolonging the network lifetime and selecting well-distributed cluster heads. The impact of sudden failures due to harsh environmental conditions, however, was not considered in all the above studies, including our own previous work.

Our model assumes that sensor nodes are all equally significant, which distinguishes it from protocols that are based on redundant node deployment, such as [15], [16], [17], [18], [19]. In these protocols, nodes are classified according to their geographic location into equivalence classes (cells). A fraction of nodes in each class (representatives) participate in the routing process, while other nodes are turned off to save energy. For example, in ASCENT [19], a node decides whether to wake up or go to sleep based on a function of the number of currently active nodes and measured link loss rate. In the Probing Environment and Adaptive Sleeping (PEAS) protocol [15], a sleeping node wakes up after a random period of time to check if there is a working neighbor. If so, it goes back to sleep. As the network becomes denser, cell-based techniques have been shown to significantly prolong the network lifetime [20].

Several other approaches were proposed to find the minimum cost k-connected undirected subgraph of a graph G [21]. The link cost can be set to the required power for transmission. In [22], an approximation of this problem is formulated as a linear programming (LP) problem. Solving the LP problem may be difficult for sensor nodes with limited processing capabilities. This approach also requires centralized control. In [23], a number of heuristics are proposed for efficient energy consumption using centralized control. Distributed algorithms are also presented in for the 2-connectivity and 3-connectivity problems. In [24], a distributed heuristic solution, the Cone-Based Topology Control (CBTC), is proposed. The CBTC approach, however, assumes that nodes can determine the direction of their neighbors.

IV. The REED Protocol

In this section, we discuss the design and operation of REED, and prove that it satisfies its requirements.

A. Design Rationale

To guarantee k-connectivity in a clustered network, every node must be able to connect to k distinct cluster heads. There are two options for selecting these k cluster heads: (1) maintaining $k-1$ backup cluster heads in every cluster (self-healing), or (2) maintaining k independent cluster head overlays, such that each node can connect to one cluster from each overlay. A cluster head overlay is a set of clusters (and their associated cluster heads) that covers the entire network. The first option has one major limitation: it is difficult to guarantee that there exist k nodes in a cluster that would cover, using the same cluster power level R_c (i.e., same transmission range), all cluster members. Therefore, we investigate the second option further in this paper. Fig. 2 illustrates a network with k cluster head overlays. Observe that each sensor node can register with one cluster head in each overlay. The protocol must not select the same node as cluster head in multiple cluster head overlays (a uniqueness requirement). This can only be satisfied if the node distribution and density in the network allow it (as specified in Section IV-D). We can now re-state the first requirement in Section II as follows: Each node v should be assigned a cluster head CH_j from each independent cluster head overlay CS_i, where $1 \leq j \leq nch_i$, nch_i is the number of cluster heads in a certain cluster head overlay CS_i, $CH_j \in CS_i$, and $1 \leq i \leq k$. Single-hop communication is used for intra-cluster routing (except where mentioned below), while multi-hop communication is likely among cluster heads for inter-cluster routing and to communicate with the observer(s).

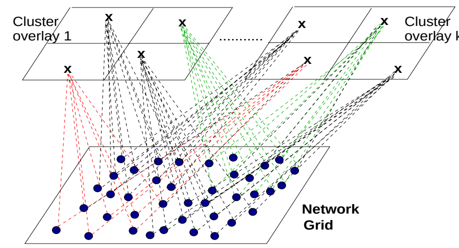


Fig. 2. A network with k cluster head overlays. Every node has one cluster head in each overlay.

When a cluster head fails during network operation, nodes communicating with that cluster head simply switch to their respective next cluster heads in their respective next cluster head overlays. The primary advantage of our approach is that the node quickly finds an alternate head and alternate communication paths, without waiting until re-clustering is triggered. Observe, however, that in this case, media access control protocols must take into consideration that the same node can be acting both as a cluster head to other nodes and as a regular node with respect to another cluster head at the same time.

To construct multiple cluster head overlays, we have two design alternatives in terms of power levels. The first alternative is to use the same cluster power level R_c in all cluster head overlays. This approach allows the application to select the best cluster range a priori and use it, even under failures. Selection of the best cluster range is beyond the scope of this work: it is typically selected to increase spatial reuse, maintain connectivity, and reduce energy consumption. The second alternative is to construct different cluster head overlays using different power levels. A node that detects failure of its cluster head at a certain power level can quickly switch to another cluster head using a different level. The second alternative poses three practical problems. First, it assumes that each node has sufficient usable power levels to support $\geq k$ different ranges. Second, it couples the clustering and power control decisions. Third, using different transmission ranges for communication at the same time may lead to collisions with certain MAC protocols due to the hidden terminal problem.[1] Therefore, we adopt the first alternative in our design.

[1]The hidden terminal problem occurs when a node A transmits to another node B because it senses a free channel. Node B, however, may be receiving another transmission from a third node C, whose range does not include A. Thus, node C is *hidden* from node A.

B. Protocol Operation

The REED clustering process constructs k-cluster head overlays. Within a single cluster head overlay, clusters are disjoint, i.e., a node belongs to exactly one cluster. In this section, we apply the REED approach to HEED clustering [1] (we also show REED application to LEACH [2] in Section VII). Therefore, REED cluster head selection is based upon two parameters: (1) a primary parameter, which is the node residual energy, and (2) a configurable secondary parameter, communication cost, which is used to break ties among selected cluster heads within the same cluster range.

Each node executing REED proceeds through three phases: (i) Initialization, (ii) Main Processing, and (iii) Finalization. The Initialization phase is similar to the initialization in the HEED protocol [1], while the Main processing and Finalization phases significantly differ from those in HEED. In the "Initialization" phase, each node sets its probability of becoming a cluster head to: $CH_{prob} = C_{prob} \times \frac{E_r}{E_{max}}$, where E_r is the estimated residual energy in the node, E_{max} is a reference maximum energy (corresponding to a fully charged battery) which is typically identical for all nodes, and C_{prob} is a fixed small probability (e.g., 0.05) used to limit the initial number of nodes competing to become cluster heads (we have found that varying C_{prob} does not have a significant effect on protocol performance).

Each node then discovers the neighbors within its cluster range R_c, and computes its *communication cost* at this cluster range. The communication cost for node v (which should be minimized) can be [1]: (1) *degree(v)*, if the application requirement is to balance load among cluster heads, or (2) $\frac{1}{degree(v)}$, if the requirement is to create dense clusters, or (3) the average minimum reachability power (AMRP), which is the mean of the minimum power levels required by all M nodes within a cluster range to reach the cluster head v, i.e., $AMRP = \frac{\sum_{i=1}^{M} MinPwr_i}{M}$, where $MinPwr_i$ is the minimum power level required by node v_i to reach v. The AMRP is an indicator of intra-cluster communication power requirements. A node can discover the minimum power level to communicate with each of its neighbors by maintaining a neighbor list for each of its power levels.

In the "Main processing" phase, REED interleaves the construction of cluster head overlays, in order to terminate the clustering process in O(1) iterations. Each node iterates $N_{initial}$ times. Each iteration must last long enough for message transfer within the cluster range. During each iteration, a node arbitrates among the cluster head announcements that it receives to select its cluster head for each cluster head overlay according to the lowest cost. The node also probabilistically elects itself to become a cluster head for the first of the remaining "uncovered" overlays. A node is uncovered in an overlay if it has not received cluster head announcements for that particular overlay. Therefore, if the uncovered cluster head overlays are $CS_{k_1}, CS_{k_2}, \ldots, CS_k$, $k_1 < k_2 < \ldots < k$, then the node competes for CS_{k_1} first, then CS_{k_2}, and so on. If successfully elected, the node announces itself as a *tentative* cluster head for that overlay. A node does not compete to become a cluster head for more than one cluster head overlay. At the end of each iteration, the node doubles its CH_{prob}. This operation is repeated until CH_{prob} reaches 1. Thus, $N_{initial}$ can be computed as follows:

$$N_{initial} = \lceil log_2 \frac{E_{max}}{C_{prob} \times E_r} \rceil + 1 \tag{1}$$

When CH_{prob} reaches 1, the node announces itself as a *final* cluster head for any cluster head overlay within which it has previously claimed to be a *tentative* cluster head (if any), provided that it has not received any announcements of other tentative cluster heads with lower cost. After doing so, the node continues to listen to cluster head announcements for cluster head overlays in which it is not covered. This process continues for a total of $N_{complete}$ iterations (including the $N_{initial}$ iterations). This ensures that all nodes, even those with high residual energy (i.e., CH_{prob} reaches 1 quickly), are likely to have *distinct* cluster heads for different cluster head overlays. If only the $N_{initial}$ original iterations are performed, then such high energy nodes may have to represent themselves in the remaining cluster head overlays in which it is not covered. Note that the initial value of CH_{prob} is not allowed to fall below a threshold p_{min}. Thus, the value of $N_{complete}$ is a constant, computed as:

$$N_{complete} \leq \lceil log_2 \frac{1}{p_{min}} \rceil + 1 \tag{2}$$

Since $N_{complete}$ is a constant, the protocol terminates in O(1) iterations. For example, for $p_{min} = 10^{-4}$, $N_{complete} = 15$. Assuming that the value of k is small (e.g., below 10), that p_{min} is appropriately selected, and that the sensor network is sufficiently dense (as described in Section IV-D), the probability of nodes serving as cluster heads in more than one cluster overlay is very small. We show via simulations in Section V that this does not occur for reasonably dense networks.

In the "Finalization" phase, REED attempts to ensure that a node is a cluster head in no more than one cluster head overlay. A node u that is not covered in $k_u < k$ overlays can send a list $S_{uncovered}$ of these overlays to all its neighbors in its cluster range. Each neighbor nbr_i which is covered in one or more of the overlays in $S_{uncovered}$ waits for a random period of time inversely proportional to its residual energy before responding. Each nbr_i then replies back with its covered overlays among $S_{uncovered}$ to u if it does not hear any other replies to u. Node u arbitrates among the replies and selects one neighbor for each uncovered overlay to act as its *proxy*. Thus, for these uncovered overlays, node u can reach a cluster head in *two hops* via the proxies. For the covered overlays, node u reaches its cluster head in a *single hop*. A node does not act as a proxy for more than one neighbor, in order to maintain the uniqueness of paths provided by each cluster head overlay. If a node cannot find a cluster head or a proxy to *cover* an overlay, it simply represents itself in that overlay. Fig. 3, 4, and 5 give pseudo-code for the REED protocol executed at a node u. In the pseudo-code, S_{nbr} is set of neighbors of u

Fig. 3. Pseudo-code of the REED protocol at node u: Initialization

1. $S_{nbr} \leftarrow \{v$: distance(u, v) $\leq R_c\}$
2. Compute cost(u) and broadcast to S_{nbr}
3. $CH_{prob} \leftarrow max(C_{prob} \times \frac{E_r}{E_{max}}, p_{min})$
4. Compute $N_{initial}$ (Eq. 1) and $N_{complete}$ (Eq. 2)
5. **For** $i \leftarrow 1$ **To** k
6. my_CH[i] $\leftarrow$ NULL
7. is_CH $\leftarrow$ FALSE

Fig. 4. Pseudo-code of the REED protocol at node u: Main processing

Repeat $N_{initial}$ **times**
1. **For** $i \leftarrow 1$ **To** k
2. $S_{CH}[i] \leftarrow \{v$: v is a cluster head for overlay $i\}$
3. **If** $S_{CH}[i] \neq \phi$
4. my_CH[i] $\leftarrow$ least_cost($S_{CH}[i]$)
5. **If** my_CH[i] = NodeID
6. is_CH $\leftarrow$ TRUE
7. **If** (CH_{prob} = 1)
8. Cluster_head_msg(NodeID,final_CH,i,cost)
9. **Else**
10. Cluster_head_msg(NodeID,tentative_CH,i,cost)
11. **ElseIf** (!is_CH) and (Random(0,1) $\leq CH_{prob}$)
12. Cluster_head_msg(NodeID,tentative_CH, i,cost)
13. my_CH[i] $\leftarrow$ NodeID
14. is_CH $\leftarrow$ TRUE
15. $CH_{prob} \leftarrow$ min($CH_{prob} \times 2$, 1)

EndRepeat

Repeat ($N_{complete} - N_{initial}$) **times**
16. **For** $i \leftarrow 1$ **To** k
17. **If** (my_CH[i] = NULL) **OR** (state(my_CH[i]) $\neq$ final_CH)
18. $S_{CH}[i] \leftarrow \{v$: v is a cluster head for overlay $i\}$
19. **If** $S_{CH}[i] \neq \phi$
20. my_CH[i] $\leftarrow$ least_cost($S_{CH}[i]$)

EndRepeat

(i.e., at a distance $\leq R_c$), and CS_i is the i^{th} cluster head overlay, $i \leq k$. The variable $my\_CH[i]$ is the cluster head of u for CS_i, $S_{CH}[i]$ is the set of cluster heads in CS_i, and $proxy\_CH$ is the set of messages from neighbors indicating the cluster head overlays in which they can serve as proxies. We do not show message reception in the pseudo-code.

C. Routing and Synchronization

During network operation, a regular node communicates only with its cluster head(s) (via one or two hops). In the cluster head overlays, an ad-hoc routing protocol, such as Directed Diffusion or Dynamic Source Routing (DSR), can be employed for determining routes among cluster heads for selected subsets of the overlays, or for all cluster heads in all overlays. Such routes will be used to communicate among clusters, or between clusters and the observer(s). If two regular nodes from different clusters attempt to communicate, communication through their cluster heads is sub-optimal 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.

Fig. 5. Pseudo-code of the REED protocol at node u: Finalization

1. $S_{uncovered} \leftarrow \{k_i$: k_i is uncovered cluster head overlay$\}$
2. **For** $i \leftarrow 1$ **To** k
3. **If** (my_CH[i] $\neq$ NULL) and (state(my_CH[i])=final_CH)
4. join_cluster(i, my_CH[i], NodeID)
5. **Else** $S_{uncovered} \leftarrow S_{uncovered} \cup \{i\}$
6. If $|S_{uncovered}| > 0$
7. $proxy\_CH \leftarrow$ proxy reply messages from neighbors
8. i = 0
9. While ($S_{uncovered} \neq \phi$ and $i < |proxy\_CH|$)
10. join_cluster(k_i, proxy_CH[i++], NodeID)
11. $S_{uncovered} \leftarrow S_{uncovered} - \{k_i\}$
12. While ($S_{uncovered} \neq \phi$)
13. Cluster_head_msg(NodeID, final_CH, i,cost)
14. $S_{uncovered} \leftarrow S_{uncovered} - \{k_i\}$

Another important consideration is whether synchronization is required for REED operation. We have shown that node synchronization is not critical for the operation of HEED clustering in [1]. We believe that node synchronization is less significant for REED. This is because lack of synchronization may result in less optimal choices of cluster heads (in terms of residual energy) in the first cluster head overlay. However, when nodes with slower clocks start receiving cluster head messages for the first cluster head overlay from nodes with faster clocks, REED is triggered at these slower nodes which may become cluster heads in other overlays. Other approaches can be applied in an unsynchronized network to trigger the REED protocol. For example, nodes with fast clocks can flood a message with a limited time-to-live field to trigger REED at their neighbors. The REED design which interleaves the selection of cluster heads for different overlays has an important advantage (in addition to low complexity): it eliminates any need for synchronizing the start of each cluster head overlay construction.

D. Analysis

We now show that the REED protocol meets the requirements outlined in Section II.

Observation 1: The number of iterations in the Main Processing phase of REED is a function of p_{min}. Thus, p_{min} must be selected such that at least k iterations are performed. REED terminates in $\Omega(k)$ and in $O(1)$ iterations.

Observation 2: In sufficiently dense networks, cluster heads are well-distributed in each cluster head overlay. This means that there is a high probability that no two cluster heads in the same cluster head overlay are neighbors, i.e., fall within each other's cluster range (refer to [1] for a formal proof). REED preserves this property for each of the independent cluster head overlays.

Lemma 1: REED has a worst case processing time and message exchange complexity of $O(kn)$ and $O(k)$ per node, respectively, where n is the number of nodes in the network.

Proof. For each node, the Initialization phase takes a processing time of at most n to compute the cost for each cluster range (i.e., $O(n)$). For the Main Processing phase, the time taken to arbitrate among cluster heads in all iterations is at most $N_{complete} \times k \times n$. Similarly, the Finalization phase takes a processing time of at most kn to arbitrate among the nodes which declared themselves as tentative cluster heads. Therefore, the total time complexity is $O(kn)$. For message exchange, a node is not permitted to generate more than one cluster head message at any iteration in the Main Processing phase. Therefore, a node can generate at most $N_{complete}$ cluster head messages. A regular node is silent until it sends a join message to a cluster head. The number of these join messages per node is at most k (i.e., $O(k)$). □

We have previously demonstrated that using appropriate bounds on the cluster range and inter-cluster transmission range, and under the density model defined in [20], a cluster head overlay is connected asymptotically almost surely (a.a.s.) [1]. In this section, we show that with REED, the network graph is k-connected a.a.s. under a different density model.

Assume that n nodes are uniformly and independently dispersed at random in an area $R = [0, L]^2$. Also assume that R is divided into N square cells of size $\frac{R_c}{\sqrt{2}} \times \frac{R_c}{\sqrt{2}}$ (thus $N = \frac{2L^2}{R_c^2}$). This means that every node in each cell can reach every other node residing in the same cell using a transmission range R_c. A cluster head overlay is connected if its cluster heads can use an inter-cluster transmission range R_t, where $R_t \geq 6R_c$ and $\exists$ at least one node per cell a.a.s. [1]. Note that $6R_c$ is a conservative lower bound to guarantee connectivity of a cluster head overlay. In typical cases, however, a cluster head in a cell will not cover all the nodes in its neighboring cells using R_c. Thus, every cell may contain a cluster head, and consequently $R_t \geq 2R_c$ will suffice. We demonstrate this via simulations in Section V.[2]

We now show that a minimum cell occupancy of at least k a.a.s., together with our protocol operation (which attempts to assign unique cluster heads for all k cluster head overlays), provide the necessary conditions for REED to satisfy the k-connectivity and uniqueness requirements. Let $\eta(n, N)$ be a random variable that denotes the minimum number of nodes in a cell, α be n/N, and $p_h(\alpha)$ be $\frac{\alpha^h}{h!} \times e^{-\alpha}$. The following theorem has been proven in [25]:

Theorem 1: If $\frac{\alpha}{ln\ N} \to 1$ as $n, N \to \infty$, and $h = h(\alpha, N)$ is chosen such that $h < \alpha$, and $Np_h(\alpha) \to \lambda$, for some positive constant λ, then $P(\eta(n, N) = h) \to 1 - e^{-\lambda}$, and $P(\eta(n, N) = h + 1) \to e^{-\lambda}$.

[2]Our definition of a cell is different from that in [20] 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). This definition is used to analyze the performance of cell-based approaches (e.g., GAF [17]). We view 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 [20], only one transmission range is used to define both density and connectivity.

Based upon this, we can prove the following:

Theorem 2: For any fixed arbitrary $k > 0$, assume that n nodes are uniformly and independently distributed at random in an area $R = [0, L]^2$. Assume R is divided into N square cells, each of side $R_c/\sqrt{2}$. If $R_c^2 n \geq aL^2 ln\ N$ for some constant $a \geq 2$, $R_c \ll L$, and $n \gg 1$, then $lim_{n,N\to\infty} E[\eta(n, N)] = k$ iff $k \approx ln\ N$ (i.e., each cell contains at least k nodes a.a.s. iff the number of cells is approximately e^k).

Proof. According to Theorem 1, a minimally occupied cell has either h or $h+1$ nodes a.a.s. if the condition stated in the theorem holds. Assume that the relation $R_c^2 n \geq aL^2 ln\ N$ holds (i.e., $n = \frac{aL^2 ln\ N}{R_c^2}$), and without loss of generality assume that $a = 2$. To prove the theorem, we must compute the condition on h which satisfies $lim_{n,N\to\infty} Np_h(\alpha) = \lambda$. Using our cell definition, $N = \frac{2L^2}{R_c^2}$, and therefore $\alpha = n/N = ln\ N$. Therefore,

$$lim_{n,N\to\infty} Np_h(\alpha) = lim_{n,N\to\infty} \frac{N(ln\ N)^h}{h!} e^{-ln\ N} = lim_{n,N\to\infty} \frac{(ln\ N)^h}{h!} = \lambda$$

Taking the logarithm of both sides, and using the Sterling formula approximation:

$$ln(ln\ N)^h - ln\ h! \approx h\ ln(ln\ N) - h\ ln\ h$$

which will not converge to a constant result unless $h \approx ln\ N$ ($h \equiv k$ in Theorem 2). □

Theorem 3: REED can produce a k-connected network among all k cluster head overlays a.a.s. for networks satisfying the conditions in Theorem 1 and Theorem 2.

Proof. We have shown in [1] that a cluster head in cluster head overlay CS_i is connected (at the cluster head level) to at least four other cluster heads a.a.s., assuming a density model in which each $\frac{R_c}{\sqrt{2}} \times \frac{R_c}{\sqrt{2}}$ cell is non-empty, and $R_t \geq 6R_c$. Now assume that our density model satisfies Theorem 1 and Theorem 2 (i.e., at least k nodes per cell a.a.s.). Since REED builds k cluster head overlays, and does not permit a node to act as head except in one overlay in this case, a cluster head can a.a.s. reach a number $n_c \geq k$ of other cluster heads (from all overlays) via the inter-cluster communication range. For non-border cluster heads, a cluster head will typically be able to reach $n_c \geq 4k$ other cluster heads. □

Note that the protocol degrades gracefully when the density constraints cannot be met.

V. Performance Evaluation

In this section, we evaluate the performance of the REED protocol via simulations. We study the connectivity and fault tolerance of REED clusters, as well as energy efficiency and routing properties.

A. Cluster and Connectivity Properties

We assume that 1000 nodes are uniformly dispersed in $R = [0, 100\ m]^2$ area (i.e., $L = 100$ m), unless otherwise specified. We set C_{prob} to 0.05, and p_{min} to 0.0005. Therefore, $N_{complete}$ = 12, whereas $N_{initial}$ varies according to the node current residual energy. Each node initially has a residual energy E_r = Uniform(0,1) Joule. We use AMRP as the secondary clustering parameter (to break ties). Each result shown is the average of 1000 simulation runs.

We have verified that each independent cluster head overlay that REED constructs exhibits the following properties (we omit these results for brevity): (1) cluster heads are well-distributed, i.e., few cluster heads are neighbors within the same cluster range R_c, (2) cluster heads have high residual energy, (3) the variance in the number of nodes in a cluster is small, and (4) the percentage of single-node clusters is very small.

We now compare the cluster properties of the independent cluster head overlays, since a cluster head can be selected for CS_i before cluster heads are selected for CS_{i+1}. Fig. 6 depicts the average number of cluster heads for 5 independent cluster head overlays ($k = 5$) at different cluster ranges (from 4 m to 20 m). The figure shows that the number of cluster heads tends to be uniform for all cluster head overlays. This is because the cluster range R_c is the same for all overlays and the basic clustering approach generates cluster heads that are not neighbors at R_c.

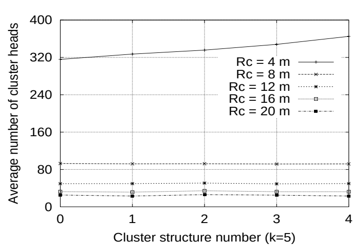


Fig. 6. Average number of cluster heads in a cluster head overlay

We also investigate the number of neighboring cluster heads for each cluster head in any cluster head overlay, for different values of k and transmission ranges R_t. The cluster range R_c is set to 20 m and R_t is varied from 20 m to 64 m. Fig. 7(a) illustrates that each cluster head can find neighboring cluster heads, even for small values of R_t (close to R_c). As R_t increases, every cluster head has an increasing number of neighboring cluster heads, thus increasing connectivity and fault tolerance. This agrees with our discussion in Section IV-D that $R_t \geq 2R_c$ is typically sufficient for inter-cluster connectivity. Since R_c = 20 m satisfies the condition in Theorem 2 (for $a = 10$), as R_t grows with respect to R_c, the network is k-connected as illustrated in the figure.

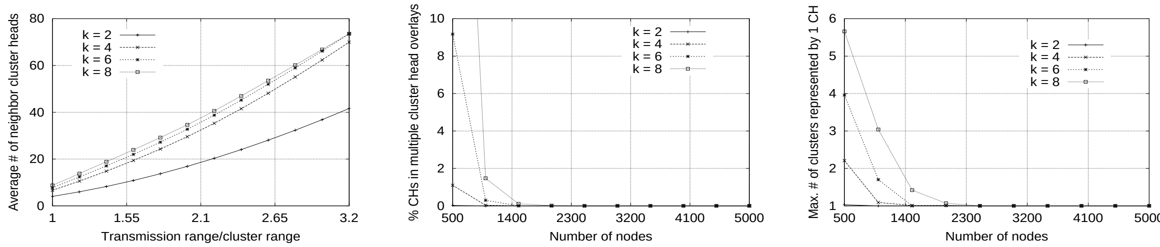
Fig. 7(b) shows the percentage of nodes that act as cluster heads in multiple cluster head overlays, as the number of nodes in the network increases. Here, R_c is set to 10 m. From the figure, the percentage is small and depends on the cluster density $\alpha = n/N$, which depends on the number of nodes, their distribution, and the cluster range. This percentage becomes zero as the mean cluster density increases from $\alpha = 2.5$ nodes/cell (for $n = 500$ nodes, and $N = 200$ cells) to $\alpha = 25$ nodes/cell (for $n = 5000$ nodes, and $N = 200$ cells). Note that $\frac{\alpha}{ln\ N}$ (Theorem 1) $\approx$ 0.5 for $n = 500$, but exceeds 1 for $n > 1100$. This explains why the curves rapidly drop at $n = 1100$ for $k = 2$ and $k = 4$. However, for $k = 6$ and $k = 8$, the curve drops at $n > 1600$, since at $n = 1100$, the condition that $h < \alpha$ in Theorem 1 is not yet satisfied ($h \equiv k$). Also note that for the selected $R_c = 10$ m, $\nexists$ a value of $a \geq 2$ that satisfies the condition in Theorem 2 for $n < 1000$. Our results are also credited to the proxy approach (discussed in Section IV-B) which reduces the possibility of a node serving as a cluster head in more than one cluster head overlay. Fig. 7(c) shows that the maximum number of cluster head overlays in which the same node can act as a cluster head is relatively high for small values of α, but becomes zero for larger α values.

B. Fault Tolerance Properties

In this section, we study the fault tolerance of a simple sensor data aggregation application that utilizes REED clustering. In this application, regular nodes report their data to their cluster heads. Cluster heads periodically send their aggregated reports to a single distant observer (e.g., a base station) via a single hop (Section VI studies multi-hop communication). Each cluster head is assumed to create a TDMA schedule for its nodes. All regular nodes send their data to their respective cluster heads according to this specified TDMA schedule. We assume Direct Sequence Spread Spectrum (DSSS) codes can be used to minimize inter-cluster interference. Therefore, we ignore collisions in our simulation. We assume that the data propagated by the sensor nodes is aggregated (e.g., using operations such as minimum, average, sum). Each cluster head aggregates the data it receives from its nodes into one frame and sends it to the observer. Clustering is triggered every N_{TDMA} TDM frames. Let the *clustering process interval*, T_{CP}, be the time 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. The network operation interval $T_{NO} = T_f \times N_{TDMA}$, where T_f is the time required to collect messages and send one TDM frame. We send 30 TDM frames every T_{NO} in our simulations.

We use a simple radio model, as in [2]. The following parameters are defined in that model: (i) E_{elec}: energy expended in digital electronics, and (ii) E_{amp}: energy expended in communication. Two models are considered: (i) free space model, in which $E_{amp} = \epsilon_{fs}$ when $d < d_0$, and (ii) the multi-path model, in which $E_{amp} = \epsilon_{mp}$ when $d \geq d_0$, where d_0 is a constant distance that depends on the environment. We use the AMRP cost as the secondary clustering parameter in our simulations. Our simulation parameters are listed in Table I.

We study the performance of REED during the first 10 clustering rounds, where a clustering round = $T_{CP} + T_{NO}$.



(a) Average number of neighboring cluster heads per cluster head for different $\frac{R_t}{R_c}$ ratios

(b) Percentage of nodes acting as cluster heads in multiple cluster overlays

(c) Maximum number of cluster head overlays in which the same node acts as cluster head

Fig. 7. Inter-cluster connectivity after employing REED

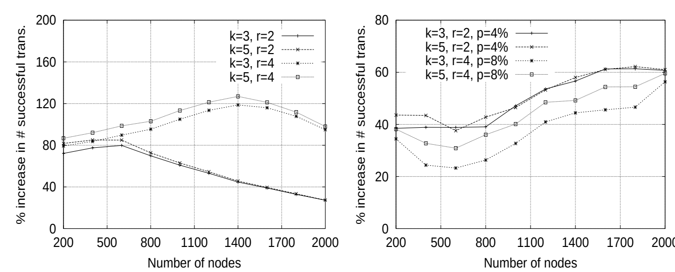
TABLE I
SIMULATION PARAMETERS

	Parameter	Value
Network	Network coordinates	From (0,0) to (100,100)
	Observer	At (200,200)
	Cluster range R_c	15 m
	Initial energy (E_{max})	10 J/battery
Application	Data packet size	100 bytes
	Broadcast packet size	25 bytes
	Packet header size	25 bytes
	Cluster round (N_{TDMA})	30 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

This allows for the isolation of the effect of unexpected failures from that of energy depletion. To study a worst case scenario, we consider failure of cluster heads only, not regular nodes. This can occur if, e.g., an eavesdropper receives cluster head announcement messages and maliciously destroys a number of these cluster heads. We study the gains of applying REED clustering to the network under two different fault scenarios.

Fault Scenario 1: We assume that at the beginning of every TDM frame interval, $r = 2$ or $r = 4$ cluster heads fail. For the case of $r = 2$, one (two for $r = 4$) randomly selected cluster head from the currently used cluster head overlay fails, and another randomly selected cluster head (two for $r = 4$) from any other randomly selected cluster head overlay fails (and thus does not send its cluster information to the observer). Without REED, nodes assigned to a cluster with a failed cluster head have to wait until clustering is re-triggered to transmit their sensed information. We use two values of k. We measure the percentage of increase in the number of successful transmissions from a regular node to a cluster head with $r = 2$ and $r = 4$. The increase in successful transmissions is compared to the case where $k = 1$. Fig. 8(a) illustrates that the increase is significant. The increase is more pronounced for the case of an increased number of failures ($r = 4$). A value of $k = 3$ is sufficient for this failure rate, and $k = 5$ only slightly increases successful transmissions.

Fault Scenario 2: Here, at the beginning of each TDM frame interval, two (or four) randomly selected cluster heads fail from any of the k cluster head overlays. In addition, any node (cluster head or regular node) may fail at any time with probability p. We plot the increase in number of successful transmissions for $k = 3$ and $k = 5$, using $p = 4\%$ (with $r = 2$) and $p = 8\%$ (with $r = 4$). Fig. 8(b) illustrates that significant increase is observed with both values of k (a slightly higher increase is observed for both fault rates with larger k). The increase is not as large as that in Fig 8(a). This is attributed to the difference in initial node failures. In the first scenario, failures directly target the top cluster head overlay, thus resulting in a lower number of successful transmissions for $k = 1$.



(a) Fault scenario 1

(b) Fault scenario 2

Fig. 8. Effect of using REED on the number of successful transmissions

We have also conducted another experiment to estimate the percentage of energy dissipated by the clustering process as a fraction of the total dissipated energy. The energy dissipated by the clustering process includes energy for: (i) the message exchange to announce a cluster head or to join a cluster, and (ii) the processing and data aggregation performed at the cluster head, in order to send one message on behalf of its cluster members to the external observer. We assume that other processing overhead, such as computing the cost, is negligible. The total dissipated energy includes: (i) the message exchange between regular nodes and their cluster heads, (ii) the message exchange between cluster heads and the observer, and (iii) the clustering overhead. The REED clustering energy consumption is expected to be low, since the process of building multiple cluster head overlays is embedded within a constant number of iterations, and the message and computational overhead are increased over the case of a single overlay by only a small factor of k. Our results (omitted here for brevity) indicate only a slight increase in the REED energy consumption as the network density increases, which is expected. The results also agree with our conjecture that the increase is not significantly affected by using larger values of k.

VI. Energy Efficiency and Multi-path Route Availability

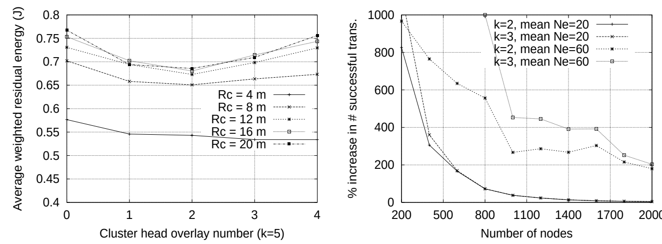
In this section, we briefly investigate how REED prolongs the network lifetime and allows multi-path routing. REED over HEED elects cluster heads which are rich in residual energy, and rotates the cluster head functionality among nodes. We compute the average residual energy per cluster head, weighted by the number of nodes in the cluster using the settings in Section V-A. Let m be the number of clusters in CS_i, $1 \leq i \leq k$. Let e_j be the residual energy of cluster head ch for cluster $C_j \in CS_i$, $1 \leq j \leq m$. Let n_j be the number of nodes in C_j. The average weighted residual energy for CS_i, e_{wi}, is computed as follows:

$$e_{wi} = \frac{\sum_{j=1}^{m} e_j \times n_j}{\sum_{j=1}^{m} n_j}$$

Fig. 9(a) shows that the average weighted residual energy of the selected cluster heads is high (above 65% in most cases). In addition, the cluster range (R_c) has a minor impact on the average cluster head residual energy, except for very small clusters (i.e., large number of cluster heads). This is not surprising since cluster head selection is based upon the node residual energy, and not on cluster range. The secondary clustering parameter, however, may be affected by R_c since more neighboring nodes are reachable with higher ranges.

REED also allows vertex-disjoint multi-path routing by selecting node-disjoint cluster head overlays when the node density and ranges R_c and R_t allow it. Node-disjoint-paths between a source/destination pair can be established by constructing a path through the cluster heads on each overlay. This can be demonstrated on the network in Fig. 1(b) where node-disjoint paths exist through dark-colored cluster heads and through light-colored cluster heads.

The use of multiple overlays in multi-hop inter-cluster routing is illustrated in the following experiment. We use the same simulation parameters as in Section V-B, but we assume that packets are being transmitted from the bottom left corner of the network (the closest cluster head to (0,0)) to the top right corner of the network (the closest cluster head to (100,100)). We use $R_t = 2 \times R_c$. The application operates for 20 clustering rounds, with 30 TDM frames per T_{NO} interval. During every T_{NO} interval, the network loses N_e cluster heads, where N_e = Uniform(10, 30), or N_e = Uniform(10, 110). Whenever a regular node detects a cluster head failure, it automatically switches forwarding to another cluster head on the next available cluster head overlay. Inter-cluster routing can utilize functional cluster heads in any specified subset of overlays, or all functional cluster heads in all overlays. We exploit all functional cluster heads in all overlays in this experiment. Fig. 9(b) shows that the successful transmission gains (computed here for both intra and inter-cluster transmissions) for larger k are significant, especially for low network density. This is because low density implies low inter-cluster network connectivity. Larger N_e values (higher failure rates) also generally yield a larger improvement.



(a) Average weighted residual energy per cluster head

(b) Multi-hop communications in an operational scenario

Fig. 9. Energy efficiency and multi-hop connectivity in REED

The figure also shows that as the network becomes denser, the gain with larger values of k diminishes. This can be attributed to the fact that although the number of nodes in a cluster increases with increased density (which implies a more severe cluster head failure effect), the percentage of failed nodes decreases, since our choice of number of failed nodes is independent of network density. This leaves more operational nodes during each round, resulting in less room for improvement by increasing k. We expect the improvements to persist in configurations where the failure percentage is dependent on network density, as demonstrated in the second scenario in Section V-B.

VII. REED over LEACH

LEACH [2] is an application-specific distributed clustering protocol for prolonging the network lifetime. With LEACH, nodes are assumed to have large transmission ranges that can span the entire network area. A node elects to become a cluster head randomly according to its residual energy and a target number of cluster heads 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. Cluster heads fuse the data they receive from their cluster members and send reports to distant observers.

The REED approach can be employed over LEACH. Building one cluster head overlay in LEACH requires a single iteration, and hence, building k overlays in an interleaved manner will also require a single iteration. Selecting disjoint sets of cluster heads for each cluster head overlay can be carried out as follows. Let $P_i(t)$ denote the probability that a node i elects to become a cluster head at time t. Let n_c denote the expected number of cluster heads in one cluster head overlay. Let r denote the clustering round, and $C_i(t)$ denote whether node i has been a cluster head in the most recent $(r \ mod \ \frac{n}{kn_c})$ rounds. Therefore, $P_i(t) = \frac{kn_c}{n - kn_c(r \ mod \ \frac{n}{kn_c})}$, if $C_i(t) = 1$ and zero otherwise. A node that has elected to become a cluster head decides which cluster head overlay to serve in with uniform probability $1/k$.

Using REED over LEACH has a number of attractive properties, such as fast termination and guaranteed cluster head set uniqueness in each cluster head overlay if $n \geq kn_c$. Observe, however, that the analysis in Section IV-D does not apply here because LEACH assumes that single-hop communication is possible between any pair of nodes. Applying this approach in a multi-hop network, where nodes have limited transmission ranges, may suffer from serious limitations, because of the uncertainty in the number and distribution of cluster heads in the network.

VIII. CONCLUSIONS

In this paper, we have presented REED, a distributed clustering protocol for robust ad-hoc sensor networks. REED provides fault tolerance and avoids the detrimental effect of cluster head failures, by constructing multiple independent cluster head overlays on top of the physical network. A node joins one cluster in each of the independent overlays. The REED clustering process terminates in a constant number of iterations. The message overhead is $O(k)$ per node, and the processing overhead is linear in the number of nodes. By carefully selecting the cluster power level, transmission power level, and satisfying the density model presented in Section IV-D, k-connectivity is achieved a.a.s. Multiple vertex-disjoint routing paths are also available in this case. This can be useful to security protocols, such as those using threshold cryptography to withstand node compromises. The REED clustering process does not consume a significant amount of energy.

Our basic approach can be applied to the design of sensor network protocols that require scalability, fault tolerance, prolonged network lifetime, security, and load balancing. We plan to study how to adapt REED to changing node density by varying cluster and transmission power levels, and re-computing k. We will also investigate different node distribution models, other than the uniform distribution. We are currently setting up a testbed for conducting small-scale experiments.

ACKNOWLEDGMENTS

This research has been sponsored in part by NSF grant ANI-0238294 (CAREER) and the Schlumberger Foundation.

REFERENCES

[1] O. Younis and S. Fahmy, “Distributed Clustering in Ad-hoc Sensor Networks: A Hybrid, Energy-Efficient Approach,” in *Proceedings of IEEE INFOCOM*, March 2004.

[2] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, “An Application-Specific Protocol Architecture for Wireless Microsensor Networks,” *IEEE Transactions on Wireless Communications*, vol. 1, no. 4, pp. 660–670, October 2002.

[3] S. Bandyopadhyay and E. Coyle, “An Energy-Efficient Hierarchical Clustering Algorithm for Wireless Sensor Networks,” in *Proceedings of IEEE INFOCOM*, April 2003.

[4] D. Estrin, R. Govindan, J. Heidemann, and S. Kumar, “Next Century Challenges: Scalable Coordination in Sensor Networks,” in *Proceedings of the ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM)*, August 1999.

[5] H. Chan and A. Perrig, “ACE: An Emergent Algorithm for Highly Uniform Cluster Formation,” in *Proceedings of the First European Workshop on Sensor Networks (EWSN)*, January 2004.

[6] S. Banerjee and S. Khuller, “A Clustering Scheme for Hierarchical Control in Multi-hop Wireless Networks,” in *Proceedings of IEEE INFOCOM*, April 2001.

[7] S. Basagni, “Distributed Clustering Algorithm for Ad-hoc Networks,” in *International Symposium on Parallel Architectures, Algorithms, and Networks (I-SPAN)*, 1999.

[8] H. Chan, A. Perrig, and D. Song, “Random Key Predistribution Schemes for Sensor Networks,” in *IEEE Symposium on Security and Privacy*, 2003.

[9] L. Zhou and Z. J. Haas, “Securing Ad Hoc Networks,” *IEEE Network*, vol. 13, no. 6, pp. 24–30, 1999.

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

[11] A. D. Amis, R. Prakash, T. H. P. Vuong, and D. T. Huynh, “Max-Min D-Cluster Formation in Wireless Ad Hoc Networks,” in *Proceedings of IEEE INFOCOM*, March 2000.

[12] V. Kawadia and P. R. Kumar, “Power Control and Clustering in Ad Hoc Networks,” in *Proceedings of IEEE INFOCOM*, April 2003.

[13] C. R. Lin and M. Gerla, “Adaptive Clustering for Mobile Wireless Networks,” in *IEEE J. Select. Areas Commun.*, September 1997.

[14] B. McDonald and T. Znati, “Design and Performance of a Distributed Dynamic Clustering Algorithm for Ad-Hoc Networks,” in *Annual Simulation Symposium*, 2001.

[15] F. Ye, G. Zhong, S. Lu, and L. Zhang, “PEAS: A Robust Energy Conserving Protocol for Long-lived Sensor Networks,” in *International Conferenece on Distributed Computing Systems (ICDCS)*, 2003.

[16] F. Koushanfar, M. Potkonjak, and A. S. Vincentelli, “Fault Tolerance Techniques for Wireless Ad Hoc Sensor Networks,” in *IEEE Sensors*, 2002.

[17] Y. Xu, J. Heidemann, and D. Estrin, “Geography-Informed Energy Conservation for Ad Hoc Routing,” in *Proceedings of the ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM)*, Rome, Italy, July 2001, pp. 70–84.

[18] 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, September 2002.

[19] A. Cerpa and D. Estrin, “ASCENT: Adaptive Self-Configuring Sensor Networks Topologies,” in *Proceedings of IEEE INFOCOM*, New York, NY, June 2002.

[20] D. M. Blough and P. Santi, “Investigating Upper Bounds on Network Lifetime Extension for Cell-Based Energy Conservation Techniques in Stationary Ad Hoc Networks,” in *Proceedings of the ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM)*, 2002.

[21] P. Crescenzi and V. K. (Eds.), *A Compendium of NP-Optimization Problems*. http://www.nada.kth.se/viggo/problemlist/compendium.html, 1998.

[22] J. Cheriyan, V. Vempala, and A. Vetta, “Approximation Algorithms for Minimum-Cost k-vertex connected subgraphs,” in *Proceedings of the 34th ACM Symposium on the Theory of Computing (STOC)*, 2002.

[23] M. T. Hajiaghayi, N. Immorlica, and V. S. Mirrokni, “Power Optimization in Fault-tolerant Topology Control Algorithms for Wireless Multi-hop Networks,” in *Proceedings of the ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM)*, September 2003.

[24] R. Wattenhofer, L. Li, V. Bahl, and Y. M. Wang, “Distributed Topology Control for Power Efficient Operations in Mutil-hop Wireless Ad hoc Networks,” in *Proceedings of IEEE INFOCOM*, 2001.

[25] V. F. Kolchin, B. A. Sevastyanov, and V. P. Chistyakov, *Random Allocations*. V. H. Winston and Sons, Washington D.C., 1978.