International Journal of Distributed Sensor Networks, 1: 305–327, 2005
Copyright © Taylor & Francis Inc.
ISSN: 1550-1329 print/1550-1477 online
DOI: 10.1080/15501320500330786

An Architecture for Robust Sensor Network Communications

OSSAMA YOUNIS

Department of Electrical & Computer Engineering, University of Arizona, USA

SONIA FAHMY

Department of Computer Science, Purdue University, USA

PAOLO SANTI

Instituto di Informatica e Telematica, CNR, Pisa, Italy

Node clustering in sensor networks increases scalability, robustness, and energy-efficiency. In hostile environments, unexpected failures or attacks on cluster heads (through which communication takes place) may partition the network or degrade application performance. We propose REED (Robust Energy-Efficient Distributed clustering), for clustering sensors deployed in hostile environments in an interleaved manner with low complexity. Our primary objective is to construct a k*-fault-tolerant (i.e.,* k*-connected) clustered network, where* k *is a constant determined by the application. Fault tolerance is 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. The independent cluster head overlays also give 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. We prove that REED asymptotically achieves* k*-connectivity if certain conditions on node density are met. We also discuss inter-cluster routing and MAC layer considerations, and investigate REED clustering properties via extensive simulations.*

Keywords Sensor Networks; Clustering; Robustness; *k*-Connectivity; Disjoint Paths

1. Introduction

Sensor networks are emerging as versatile computing platforms for several monitoring and control applications. These 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 self-organization techniques to give the applications a robust and fault-tolerant computing and communication platform. Energy efficiency is also critical. For example, in a sensor network for seismic monitoring or radiation level control in a nuclear plant, the lifetime of each sensor significantly impacts the quality of surveillance.

The authors would like to thank Andrew Campbell (Columbia University) for his helpful comments on this work.

Address correspondence to Ossama Younis, Department of Electrical and Computer Engineering, University of Arizona. E-mail: Oyounis@cs.purdue.edu

Clustering is a self-organization technique to 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:

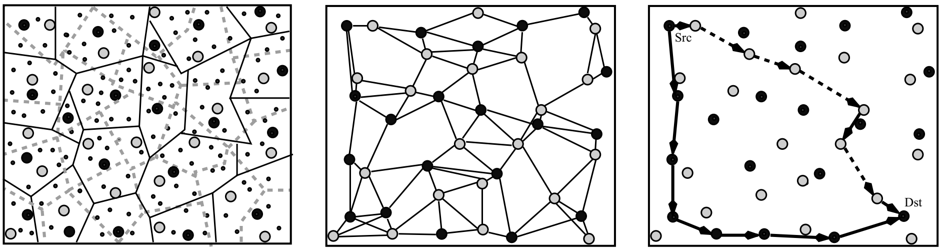
1. communicating with its registered cluster nodes, typically via a single hop or a few hops, and
2. communicating with other cluster heads or with the observer(s) on behalf of its cluster (via a single-hop or multi-hops).

Clustering prolongs the network lifetime by reducing contention on transmission channels, supporting data aggregation at cluster heads, and giving nodes the opportunity to sleep while not sensing. In addition, cluster heads can rotate their functionality to distribute energy consumption [1], [2], [3], [4], [5], [6], [7].

In this work, we consider applications that operate on large-scale networks deployed in harsh environments. These applications may be **source-driven** (i.e., nodes periodically report their readings) with the possibility of data aggregation, or **query-driven** where nodes respond to queries issued by the observer(s) if they possess the information of interest. Although clustering allows data aggregation and load balancing, two challenges remain to be addressed. First, failures of cluster heads may isolate their cluster nodes, partitioning the network until the next clustering process is triggered. Second, cluster heads can be compromised in environments with malicious attackers, which necessitates the construction of alternate (node-disjoint) routing paths from sources to observers. 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].

Toward 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 regular nodes) provide means for every node to automatically and independently switch cluster heads on detecting a cluster head failure. 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. REED is completely distributed and has low asymptotic computation and communication overhead, as will be shown in Section III. REED *interleaves* the construction of the k cluster head overlays, and operates correctly with unsynchronized nodes. We use HEED clustering [3] as the underlying clustering approach for REED because of its generality and energy efficiency. Our ideas, however, can be easily incorporated into any clustering protocol (e.g., REED was combined with LEACH clustering [4] in [10].

We prove that REED constructs k-fault-tolerant clustered networks if certain conditions 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. Figure 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 distributed protocols have been proposed in the literature to construct k-fault tolerant clustered sensor networks. Our approach also allows vertex-disjoint multi-path routing by selecting node-disjoint cluster head overlays when the node density allows it. Node-disjoint-paths between a (source, destination) pair can be established by constructing a path through the cluster heads on each overlay. This is demonstrated in Fig. 1(b) Fig. 1(c), where node-disjoint paths exist through dark-colored and light-colored cluster heads.



(a) Two cluster head overlays.

(b) A 2-connected inter-cluster network graph.

(c) Availability of disjoint intercluster paths.

FIGURE 1 A network with 2 cluster head overlays. Larger circles denote cluster heads.

The remainder of this paper is organized as follows. Section 2 defines the system model and states the protocol requirements. Section 3 presents our proposed protocol, REED, and argues that it achieves its goals. Section 4 demonstrates the effectiveness of REED via simulations. Section 5 briefly surveys related work. Finally, Section 6 gives concluding remarks and directions for future work.

2. System Model and Objectives

Consider a set of sensor nodes dispersed uniformly and independently at random on a field. We study **applications** that:

1. utilize a large-scale network;
2. may impose different load in different parts of the network (i.e., energy consumption not necessarily uniform for all nodes);
3. can exploit data aggregation. The applications may be source-driven or query-driven; and
4. operate in hostile, unattended environments, where nodes may unexpectedly fail or are compromised.

Failure may depend on time, network density, or human intervention. The above assumptions describe a class of application scenarios that requires scalable self-organization mechanisms. A hierarchical network organization will not only facilitate data aggregation, but will also facilitate the formation of uniform, robust backbones, and give an opportunity for nodes to sleep while not sensing or transmitting.

We consider a **network** with the following properties:

1. nodes are quasi-stationary, which is typical for most sensor network applications;
2. links are symmetric, and communication is multi-hop;
3. all nodes have similar processing and communication capabilities and equal significance;
4. nodes are not location-aware, have limited battery power, and are not necessarily synchronized (we discuss node asynchrony in Section 3-D);
5. all nodes use a single radio frequency, e.g., 900 MHz for Mica2 sensor motes, or 2.4 GHz for MicaZ motes (the selection of a MAC layer protocol depends on the type of the application as discussed in Section 3-E);
6. failed nodes can not be revived since they are left unattended. If it is possible for a failed node to recover, it starts the clustering process as a newly deployed node (see Section 3-F); and

7. each node has a number (at least 2) of transmission power levels. An example of such sensor nodes is Berkeley motes [11].

The only limiting assumption about the network is the availability of at least two usable transmission power levels for intra-cluster and inter-cluster communication. We only assume link symmetry for nodes that are using the same transmission power. To avoid link asymmetry in practical settings, a node *u* does not consider node *v* to be its neighbor unless *v* declares that it has heard the heartbeat from *u*. The fifth assumption does not limit our solution, as we will show in Section 3-E. Our primary objective is to design a distributed algorithm for constructing a fault-tolerant clustered network. Note that our assumption about node stationarity is essential at least for the elected cluster heads to maintain k-connectivity during network operation. The following requirements of the algorithm must be met:

1. The clustered network should be *k*-fault-tolerant if certain conditions on node density are satisfied (defined in Section 3-C). In a *k*-fault-tolerant network, the network remains connected even under the failure of up to $k - 1$ nodes. If the conditions on node density cannot be met, the protocol performance should degrade gracefully.
2. The network should provide *k* node-disjoint paths between every source-destination pair in the network.
3. Clustering must be fully distributed; clustering must terminate within a fixed number of iterations, regardless of network diameter; clustering must be efficient in terms of processing complexity and message exchange; and cluster heads must have relatively high residual energy, compared to regular nodes.

3. 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 achieve *k*-connectivity in a clustered network, every node must a least 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, 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 can cover – using the same cluster power level R_c (i.e., same transmission range) – all cluster members. The second option overcomes this limitation and allows low-overhead construction of *k* disjoint paths between every source-destination pair in the network. It also handles the worst-case scenario of *k*-1 cluster head failures. Therefore, we investigate the second option further in this paper. If the node-disjoint paths requirement is unessential, and node failures are random (i.e., cluster heads are not targeted), the number of constructed overlays can be reduced to $< k$.

Figure 2 illustrates a network with *k* cluster head overlays. Observe that each sensor node can register with one cluster head in each overlay. In order to achieve *k* fault-tolerance, the protocol must not select the same node as cluster head in multiple cluster head overlays. This can only be satisfied if the node distribution and density in the network allow

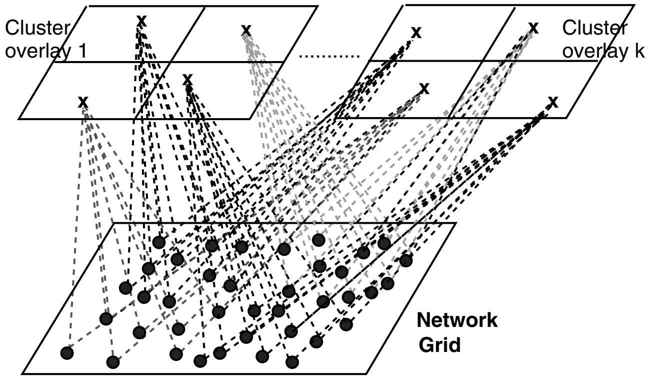


FIGURE 2 A network with *k* cluster head overlays. Every node has one cluster head in each overlay.

it (as specified in Section 3-C). We can now re-state the first requirement in Section 2 as follows: Each node *v* should be assigned a cluster head CH_j from each independent cluster head overlay CS_i, where $1 \le j \le nch_i$, nch_i is the number of cluster heads in a certain cluster head overlay CS_i, $CH_j \in CS_i$, and $1 \le i \le k$. 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).

When a cluster head fails during network operation, nodes communicating with that cluster head simply switch to another cluster head in another cluster head overlay (see Section 3-G). 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 simultaneously acting as a cluster head to other nodes and as a regular node with respect to another cluster head. Media access considerations in this case are discussed in Section 3-E. We 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.

B. Protocol Operation

The REED clustering process constructs *k*-cluster head overlays. Within a single cluster head overlay, clusters are disjointed, i.e., a node belongs to *exactly one* cluster. In this section, we apply the REED approach to HEED clustering [3]. We have also applied REED to LEACH [4] clustering in [10]. 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 each other cluster range.

Each node $u \in V$ executing REED proceeds through three phases:

1. Initialization,
2. Main Processing, and
3. Finalization.

In the "Initialization" phase, *u* sets its probability of becoming a cluster head to:

$$CH_{prob} = C_{prob} \times \frac{E_r}{E_{max}} \tag{1}$$

where E_r is the estimated residual energy in *u*, E_{max} is a reference maximum energy (e.g., corresponding to a fully charged battery), and C_{prob} is a small fixed probability (e.g., 0.05) used to limit the initial number of nodes competing to become cluster heads.

Node *u* then discovers the neighbors within its cluster range R_c, and computes its *communication cost* at this cluster range. The communication cost for node *u* (which should be minimized) can be [3]:

1. *degree(u)*, if the application requirement is to balance load among cluster heads, or
2. $\frac{1}{degree(u)}$, 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 *u*, i.e., $AMRP = \frac{\sum_{i=1}^{M} Min\,Pwr_i}{M}$, where $MinPwr_i$ is the minimum power level required by node u_i to reach *u*. 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 as in [12].

In the "Main processing" phase, REED interleaves the construction of cluster head overlays. Node *u* iterates N_{initia} times. Each iteration has a fixed duration that accommodates a small number of messages (cluster head announcements)[1]. At the end of each iteration, *u* arbitrates among the cluster head announcements that it receives to select its cluster head for each cluster head overlay according to the lowest cost. *u* 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 for a node are CS_{k1}, CS_{k2},…, CS_k, $k_1 < k_2 < \ldots < k$, then *u* competes for CS_{k1} first, then CS_{k2}, and so on. If successfully elected, *u* announces itself as a *tentative* cluster head (*T_CH*) for that overlay. Node *u* does not compete to become a cluster head for more than one cluster head overlay. At the end of each iteration, *u* doubles its CH_{prob}. This operation is repeated until CH_{prob} reaches 1. Thus, $N_{initial}$ is computed as follows:

$$N_{initial} = \left\lceil log_2 \frac{E_{max}}{C_{prob} \times E_r} \right\rceil + 1 \tag{2}$$

When CH_{prob} reaches 1, *u* announces itself as a *final* cluster head (*F_CH*) for any cluster head overlay within which it had previously announced itself 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, *u* continues to listen to cluster head announce-

[1] Due to our probabilistic clustering approach, the number of cluster head announcements per iteration is small. Observe also that even if node *u* misses an announcement in one iteration, it is likely to receive one in a subsequent iteration(s).

ments 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. 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:

$$N_{complete} \leq \left\lceil log_2 \frac{1}{p_{min}} \right\rceil + 1 \tag{3}$$

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 (below 10), that p_{min} is correctly selected, and that the sensor network is sufficiently dense (as described in Section 3-C), the probability of nodes serving as cluster heads in more than one cluster overlay is extremely small. We verify this via simulations in Section 4.

In the "Finalization" phase, REED ensures that, if possible, a node is a cluster head in no more than one cluster head overlay. A node u that is uncovered in $k_u < k$ overlays sends a list S_{un} 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_{un}, 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_{un} 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*. 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 in 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. Figure 9 gives a pseudo-code for the REED protocol executed at a node u. In the pseudo-code, S_{nbr} is set of neighbors of u (i.e., at a distance $\leq R_c$), and CS_i is the i^{th} cluster head overlay, $i \leq k$. The variable $my\_CH[i]$ stores the cluster head of u for CS_i, $S_{CH}[i]$ stores 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. Analysis

We now show that the REED protocol meets the requirements outlined in Section 2.

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 is $O(1)$ iterations.

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. 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 send more than one cluster head message at any iteration in the Main Processing phase. Therefore, a node sends 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 [13], a cluster head overlay is connected asymptotically almost surely (a.a.s) [3]. In this section, we show that with REED, the network graph is k-connected a.a.s. under the density model described below.

Assume that n nodes are uniformly and independently dispersed at random in an area $R = [0,L]^2$. 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. [3]. Note that $6R_c$ is a conservative lower bound to guarantee connectivity of a cluster head overlay. In typical cases, 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 4. Note that our definition of a cell is different from that in [13] 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 [14]). 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 [13], only one transmission range is used to define both density and connectivity.

We now show that a minimum cell occupancy of at least k a.a.s., together with our protocol operation (which assigns unique cluster heads for all k cluster head overlays), allow satisfying the k-connectivity and cluster head uniqueness requirements of REED. 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 [15]:

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

Based upon this theorem, we extend the analysis in [13] to give sufficient conditions for k-connectivity. We prove the following two theorems:

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 $a \geq 2$, $R_c << L$ and $n >> 1$, then $\lim_n, N \rightarrow \infty E[\eta(n, N)] = k$ iff $k \sim \ln N$.

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., $\frac{\alpha L^2 ln N}{R_c^2}$, and without loss of generality assume that $a = 2$. To prove the theorem, we must compute the condition on h which satisfies *limn, N→∞Nph(α)* = *λ*. Using our cell definition, $N = \dfrac{2L^2}{R_c^2}$, and therefore α = *n/N* = *ln N*. Therefore,

$$lim_{n,N\to\infty} N_{ph}\,(\alpha) = lim_{n,N\to\infty} \frac{N(ln\,N)^h}{h!} e^{-ln\,N} = \\ lim_{n,N\to\infty} \frac{(ln\,N)^h}{h!} = \lambda \tag{4}$$

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

$$ln(ln\,N)^h - ln\,h! \sim ln\,(ln\,N) - h\,ln\,h$$

which will not converge to a constant result unless *h ln N* (*h k* in Theorem 2).

Theorem 3. REED produces 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 [3] that a cluster head in cluster head overlay $\mathrm{CS_i}$ is connected (at the cluster head level) to at least four other cluster heads a.a.s., assuming a density model in which each $\dfrac{R_c}{\sqrt{2}} \times \dfrac{R_c}{\sqrt{2}}$ cell is non-empty, and $Rt \geq 6Rc$. Using this result, we have proved that a cluster head overlay, $\mathrm{CS_i}$, and consequently the entire network, is connected a.a.s. 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 $nc \geq k$ 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 $nc \geq 4k$ other cluster heads.

Observe that the coverage problem is orthogonal to the connectivity problem addressed in this work. If the coverage/sensing and transmission ranges of nodes are similar, simple sleep/wake up protocols, such as PEAS [16], can be employed to maintain coverage. Conditions on the number of nodes and its relation to the coverage range and failure probability are given in [17]. Our density model remains valid for connectivity and coverage as long as the sensing range R_s exceeds or is equal to R_c.

D. Node Synchronization

Node synchronization is not required for REED operation. Lack of synchronization may only result in less optimal choices of cluster heads (in terms of residual energy) in the first cluster head overlay. 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. 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. The particular sensor application may, however, need to synchronize the network, e.g., for data aggregation, caching, TDMA scheduling, or if nodes use a duty cycle to switch to sleep mode during idle periods. Mechanisms such as [18], [19],

[20] can easily be employed to synchronize the nodes, thus paving the way for more efficient clustering. The synchronization requirement in these cases is pertinent to the multi-hop nature of the network, and is *not* required by the clustering protocol.

To accommodate nodes with highly variable clocks, as well as newly deployed nodes in the network, we trigger REED as follows. When the clustering timer is triggered in a cluster head with a fast clock, it broadcasts a message to its members and all its neighboring heads to start REED clustering. A cluster head that receives such a trigger message immediately broadcasts a message to its members and neighboring heads and starts REED clustering. Therefore, when REED is triggered in certain regions in the network, these in turn trigger other regions. In addition, nodes that do not receive the trigger messages are automatically triggered when they start receiving cluster head announcements as described above.

A newly deployed node *v* must start REED clustering immediately after deployment. If cluster head announcements (through routing updates or heartbeats) reach *v* before its clustering process is complete, *v* can join existing clusters. Otherwise, *v* elects to become a cluster head during the rest of this round for uncovered overlays, and it is triggered by a neighboring cluster head for the start of the next round. We have implemented this approach on Berkeley sensor motes for single overlay clustering [21].

E. Medium Access Control and Duty Cycle

The network organization resulting from REED clustering is ideal if each node possesses k+1 different radios or non-interfering channels. In this case, different frequencies can be assigned for intra-cluster and inter-cluster communication for each of the k overlays. For cheap sensors, however, only a single radio frequency is typically available (as stated in Section 2). In this case, we recommend parallelizing the intra-cluster and inter-cluster communications by assigning two different CDMA codes: one for intra-cluster communication and the other for inter-cluster communication, to enable them to proceed simultaneously. At the inter-cluster level, contention-based MAC schemes, such as the 802.11 DCF, can be used. At the intra-cluster level, it is typically easy to synchronize a regular node with its cluster head(s) using techniques such as [18], [19]. Intra-regional synchronization facilitates constructing TDM schedules for each cluster. Therefore, after the k overlays are constructed, every node contacts each of its k cluster heads to be assigned a time slot in the TDM frame of the cluster head. A node keeps a list of its schedules with each cluster head to be able to switch to another head when one fails. Since it is difficult to assign different frequencies or CDMA codes to the cluster heads, cluster heads should compete for the medium using contention-based MAC layer protocols, e.g., 802.11 DCF.

Energy-efficiency is significantly improved if a duty cycle is applied where nodes wake up for short periods of time to sense the medium and report their information, and then go back to sleep. Clustering gives an opportunity for regular nodes to exploit this duty cycle because they are not part of the routing infrastructure. In a clustered, multi-hop network, the cluster head will be responsible for synchronizing its members' sleep and wakeup cycles to ensure their participation in the clustering process, guarantee an effective data rate, and coordinate the TDMA schedule (if applicable). Three considerations are important for duty cycle support:

1. the protocol used for scheduling the node duty cycles should effectively yield an active set of nodes that uniformly covers the entire network area. This is possible if sleeping nodes are selected intelligently or a sampling technique is used, such as [22];
2. the set of active nodes should satisfy the density model specified in Section 3-C; and

3. a node that acts as a cluster head in any of the *k* overlays should not be put to sleep until it gives up its role when clustering is re-triggered, or its duty cycle is synchronized with its cluster members and its neighboring cluster heads. Integrating node duty cycle with topology management is planned in our future work.

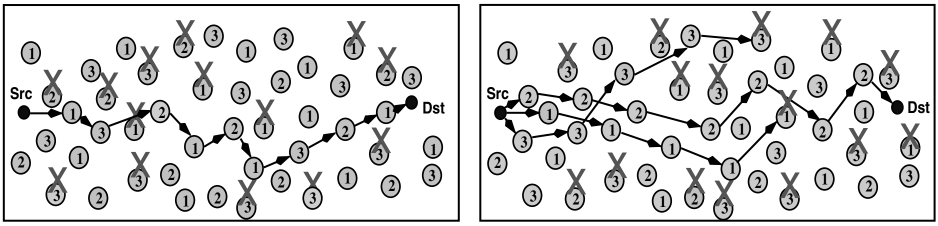
F. Detecting Cluster Head Failures

In hostile environments, a cluster head must periodically broadcast a “heartbeat” message to inform its members that it is still functional. A regular node determines that its cluster head has failed if it does not receive any heartbeats for a specified interval of time. This approach entails that cluster members are synchronized with their cluster heads to be able to wake up to capture heartbeats. Alternatively, a regular node can solicit a heartbeat from its cluster head after sending a certain number of messages. This is useful in networks where regular nodes go through a duty cycle and cannot hear periodic cluster head heartbeats. The cluster head would, however, need to respond to each individual member instead of periodically broadcasting a heartbeat to the entire cluster in this case. Cluster heads detect neighboring cluster head failures using routing updates if pro-active routing is used at the inter-cluster level. Failures are implicitly detected without extra overhead if reactive routing is used. If failed nodes can be revived, a node starts the clustering process as any newly deployed node would, and listens to cluster head announcements in order to cover its *k* overlays. Regular nodes are not part of the routing infrastructure and their failure does not impact network connectivity. When a node detects the failure of its primary cluster head, it switches its transmissions to the next cluster head on its list.

G. Inter-cluster Routing

In the cluster head overlays, an ad-hoc routing protocol, such as Directed Diffusion or Dynamic Source Routing (DSR), is employed for determining routes among cluster heads for selected subsets of the overlays. These routes are used to communicate among clusters, or among clusters and observer(s). As discussed in Section 3-E, CDMA is used to allow inter-cluster and intra-cluster communications to proceed simultaneously without interference. Two alternatives are possible to exploit the multiple overlay organization according to the environment:

- If unexpected node failures are common, all the cluster heads in the *k* overlays participate in the routing infrastructure to increase the degree of connectivity. This alternative is demonstrated in Fig. 3(a). We quantify the increase in the number of successful transmissions with this approach in Section 4-B and Section 4-C.



(a) Forwarding to any cluster head in the 3 overlays. “X” denotes a failed node.

(b) Forwarding on each overlay independently. “X” denotes compromised nodes that deliberately drop packets.

FIGURE 3 Inter-cluster routing alternatives on a network with 3 overlays. The number on each node denotes which overlay it belongs to.

- In the presence of compromised nodes that may deliberately drop packets, a source node transmits duplicate packets on each of the k overlays independently. In other words, node-disjoint paths are exploited to avoid black holes (nodes that drop packets instead of forwarding them) in the network. This is demonstrated in Fig. 3(b) where only the packet on overlay 2 succeeds to reach the destination. We demonstrate the effectiveness of this approach in Section 4-C. Note that we are merely trying to find more than one connected path to get around compromised nodes and not to identify the compromised nodes.

H. Varying the Number of Overlays (k)

Thus far, we have assumed that k is a constant determined by the application. Since our model assumes that nodes fail unexpectedly or due to energy depletion, the node density decreases during network operation. Thus, it may become impossible to construct k independent overlays. A reasonable action to take in this case is to reduce the value of k. It is difficult, however, to have the entire network agree on a new value of k, since we do not assume the presence of any centralized control entity. Fortunately, this problem is mitigated by the clustering approach itself. Since cluster head overlays are "gradually" constructed, nodes in some areas of the network will be unable to find cluster heads for all k overlays. Therefore, these nodes will have to represent themselves for some of the k overlays. If this holds for nodes in different areas of the network, this implies that nodes are actually using a smaller k. In other words, only $k' < k$ overlays are independent. Therefore, the protocol degrades gracefully when the density constraints cannot be met. We show in Section 4-A that cluster heads serve in more than one overlay when the density decreases (Figs. 5(b) and 5(c)). This corresponds to a smaller k.

4. Performance Evaluation

In this section, we study the connectivity and fault tolerance of REED clusters, as well its energy-efficiency and routing properties. We have implemented our own discrete-event simulator that is much simpler than existing simulators, such as ns-2 or GloMoSim. This simplicity allows the simulator to scale to thousands of nodes. The simulator does not model all the details of a realistic MAC layer protocol, e.g., interference. This is a reasonable approach for three reasons. First, all the results presented below are comparative and use the same simplifications for all the scenarios. Second, we assume that the MAC layer uses CDMA (or orthogonal channels) to allow simultaneous intra-cluster and inter-cluster transmissions. The size of the DSSS codes is accounted for in the simulations. Third, the typical packet sizes are small (the default is 29 bytes for TinyOS [23]), which reduces the probability of collisions, especially when aggregation is used.

A. Cluster and Connectivity Properties

We assume that 1000 nodes are uniformly dispersed in $R = [0, L]^2$ area where $L = 100$ m, unless otherwise specified. We set C_{prob} to 0.05, and p_{min} to 0.0005. Therefore, $N_{complete} = 12$. 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 several properties shown for HEED [3] (we omit these results for brevity). The most important property is that cluster heads have high residual energy, compared to regular nodes.

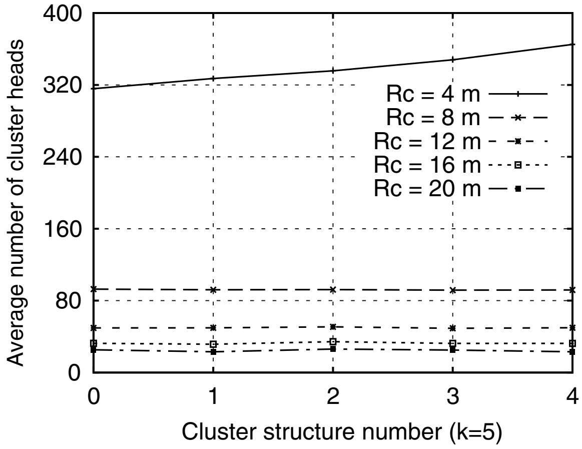
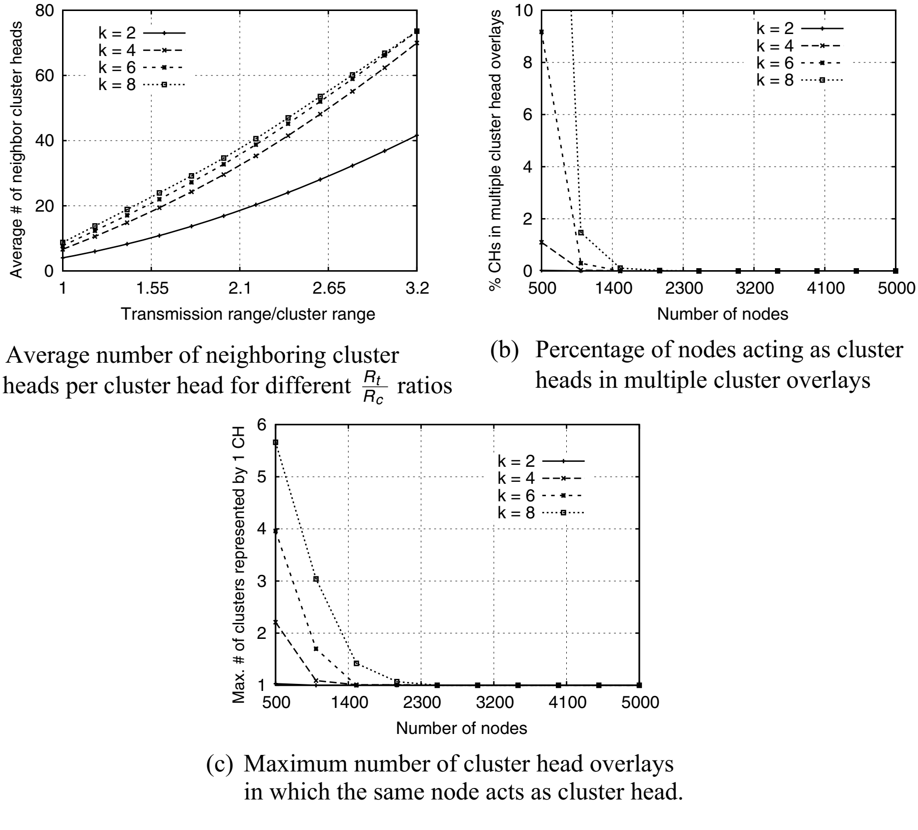


FIGURE 4 Average number of cluster heads in a cluster head overlay.



(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.

FIGURE 5 Cluster properties and connectivity after employing REED.

We compare the cluster properties of the independent cluster head overlays. Results (shown in Fig. 4) illustrate that the average number of cluster heads for 5 independent cluster head overlays (k = 5) at different cluster ranges (from 4 m to 20 m) 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 selects cluster heads that are not neighbors at R_c.

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 range Rt. The cluster range Rc is set to 20 m and Rt is varied from 20 m to 64 m. Figure 5(a) illustrates that each cluster head can find neighboring cluster heads, even for small values of Rt (close to Rc). As Rt 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 3-C which concluded that $Rt \geq 2Rc$ is typically sufficient for inter-cluster connectivity. Since $Rc = 20$ m satisfies the condition in Theorem 2 (for $a = 10$), as Rt grows with respect to Rc, the network is k-connected as illustrated in the figure. The node degree of a cluster head is therefore dependent on the Rt / Rc ratio. We can see from the figure that each cluster head about 4–8 neighboring cluster heads per overlay for ratios of 2.5–3.

Figure 5(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, Rc 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) ≈ 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 $k = 4$. 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 k). Note that for the selected $Rc = 10$ m, a value of $a \geq 2$ that satisfies the condition in Theorem 2 for $n < 1000$. Our results are credited to the proxy approach (discussed in Section 3-B) which reduces the possibility of a node serving as a cluster head in more than one cluster head overlay. Figure 5(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 4-C studies mutli-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 are 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 (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 [4]. The following parameters are defined in that model:

1. E_{elec}: energy expended in digital electronics, and
2. E_{amp}: energy expended in communication.

Two models are considered:

1. free space model, in which $E_{amp} = \in_{f_s}$ when $d < d_0$, and
2. multi-path model, in which $E_{amp} = \in_{mp}$ when $d \geq d_0$, where d_0 is a constant distance that depends on the environment. Our simulation parameters are listed in Table 1.

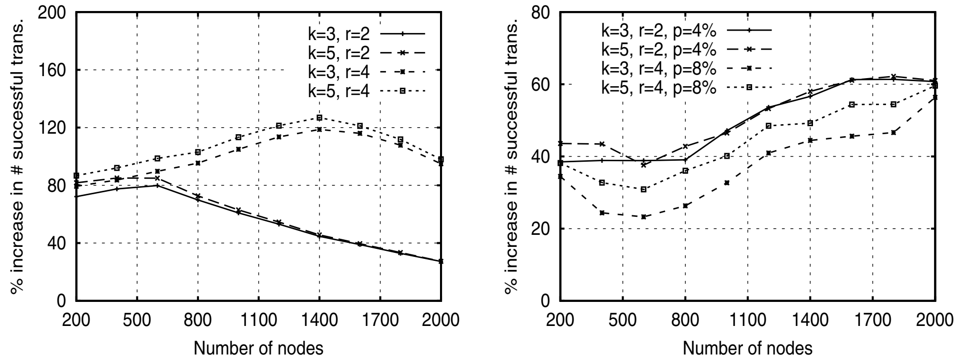
We study the performance of REED during the first 10 clustering rounds, where a clustering round = $T_{CP} + T_{NO}$. This allows for the isolation of the effect of unexpected failures from that of energy depletion. To study a worst case scenario, we consider failure of cluster heads only, not regular nodes. This can occur if, for instance., an eavesdropper receives cluster head announcement messages and destroys a number of these cluster heads. We quantify 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 $k = 1$ (the HEED protocol [3]). Figure 6(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. At the beginning of each TDM frame, two (or four) randomly selected cluster heads fail from *any* of the k cluster head overlays. In addition, any

TABLE 1 Simulation Parameters

	Parameter	Value
Network	Network coordinates	From (0,0) to (100,100)
	Observer	At (200,200)
	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	E_{elec}, E_{fusion}	50 *nJ/bit*, 5 *nJ/bit/signal*
Model	$E_{fs}, \in_{mp}$	10 *pJ/bit/m*^{2}, 0.0013 *pJ/bit/m*^{4}
	Threshold distance (d_0)	75 m



(a) Fault scenario 1

(b) Fault scenario 2

FIGURE 6 Effect of using REED on the number of successful transmissions (single hop communication).

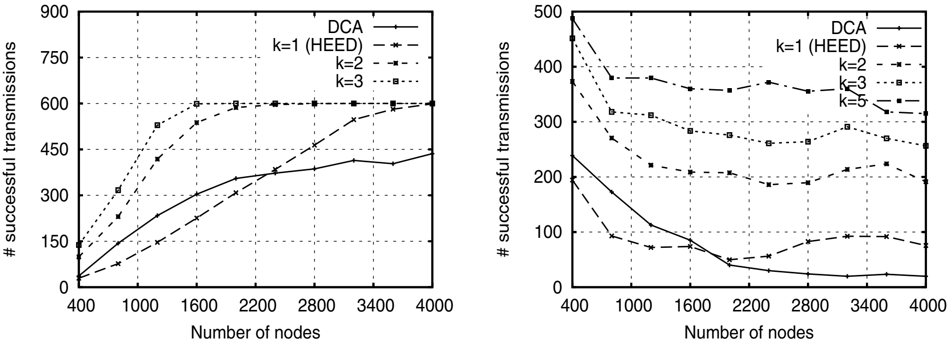
node (cluster head or regular node) may fail at any time with probability *p*. We plot the increase in the number of successful transmissions for *k* = 3 and *k* = 5, using *p* = 4% (with *r* = 2) and *p* = 8% (with *r* = 4). Figure 6(b) illustrates that a significant increase is observed with both values 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 6(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.

C. Multi-path Route Availability and Energy-efficiency

In this section, we investigate REED multi-path routing and energy-efficiency properties to assess the energy-reliability trade-off.

Inter-cluster Routing. We use the same simulation parameters as in Section 4-B, except 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 $R_t = 2.5 \times R_c$. The application operates for 20 clustering rounds, with 30 TDM frames per T_{NO} interval (a maximum of 600 packets can be transmitted). After every T_{NO} interval, N_e cluster heads fail, where N_e = Uniform(10, 110). When a regular node detects a cluster head failure for *k* = 2 or *k* = 3, it automatically switches forwarding to another cluster head on the next available cluster head overlay. We exploit all functional cluster heads in all overlays in this experiment, which is the first routing alternative in Section 3-G.

We compare REED performance for *k* = 2 and *k* = 3 to the case of *k* = 1 (which corresponds to the HEED protocol [3]) and the case of the Distributed Clustering Algorithm (DCA) [2]. We chose DCA as a representative weight-based clustering approach that selects well-distributed cluster heads in the network (other examples of weight-based approaches are given in Section 5). Figure 7(a) 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 intercluster network connectivity. 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



(a) Cluster head availability

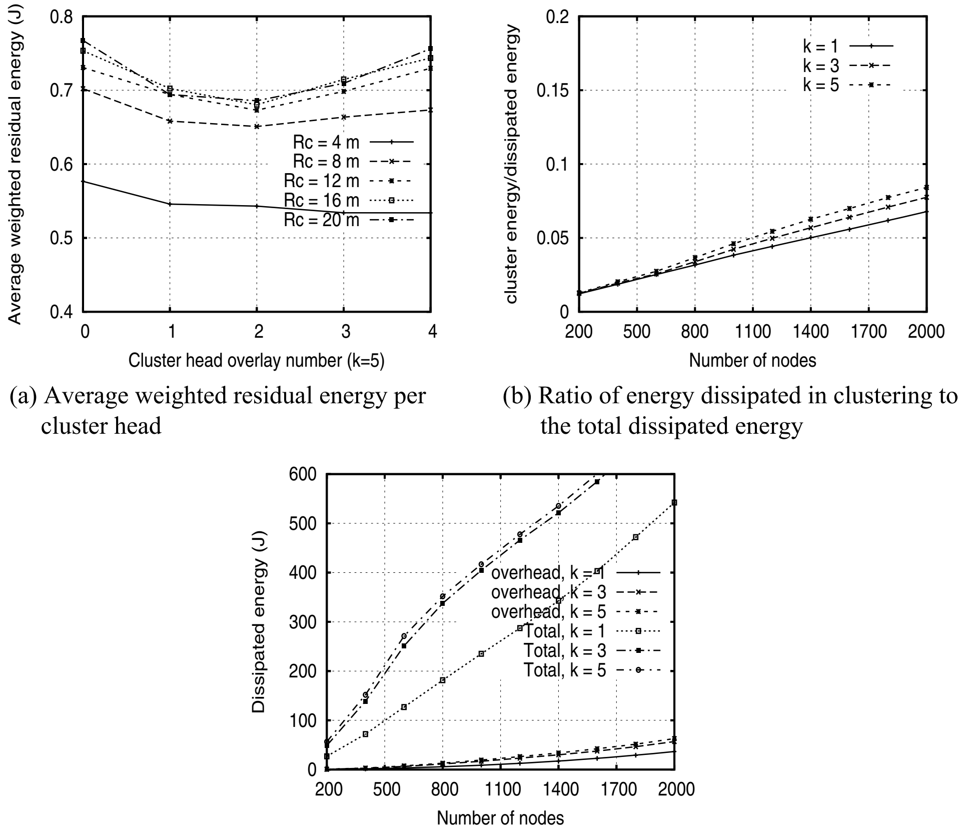
(b) Transmission on independent paths

FIGURE 7 Effect of REED on the number of successful transmissions (multi-hop communication).

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*. The figure also shows that higher reliability is achieved at smaller node densities for larger values of *k* due to increased path availability.

We now study the second inter-cluster routing alternative described in Section 3-G, and compute the number of successful transmissions from a source at the bottom left corner of the network to a destination at the top right corner. In this scenario, the source sends every packet on *each* of the *k* overlays independently to improve its chance of reaching its destination. At every T_{NO} interval, 2.5% of the nodes fail, which means that at the end of 20 rounds, 50% of the nodes in the network have failed. Figure 7(b) shows that as *k* increases, the number of successful transmissions increases significantly, compared to DCA and $k = 1$ (HEED). The number of successful transmissions is higher for lower node densities. This is because fewer nodes (and consequently cluster heads) fail at lower densities.

Energy-efficiency. REED 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 4-A. Let m be the number of clusters in CS_i, $1 \le i \le k$. Let e_j be the residual energy of cluster head ch for cluster $C_j \in CS_i$, $1 \le \mathrm{j} \le m$. Let n_j be the number of nodes in Cj. The average weighted residual energy for CS_i, e_{wi}, is computed as follows: $e_{wi} = \dfrac{\sum_{j=1}^{m} e_j \times n_j}{\sum_{j=1}^{m} n_j}$. Figure 8(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 (Rc) has a minor impact on the average cluster head residual energy, except for extremely 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 Rc since more neighboring nodes are reachable with higher ranges. Quantifying the energy savings due to clustering is beyond the scope of this work, as it was the primary focus of our previous work [3].



(a) Average weighted residual energy per cluster head

(b) Ratio of energy dissipated in clustering to the total dissipated energy

(c) Energy dissipation due to network operation and overhead

FIGURE 8 Energy-efficiency and clustering overhead in REED.

Using the first scenario described in Section 4-B, we 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:

1. message exchange to announce a cluster head or to join a cluster, and
2. 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 additional processing overhead, such as computing costs, is negligible. The *total* dissipated energy includes:

1. message exchange between regular nodes and their cluster heads,
2. message exchange between cluster heads and the observer, and
3. clustering overhead.

The results demonstrated in Fig. 8(b) indicate that

1. the ratio of energy dissipated by REED is ≤ 8% of the total energy dissipated,
2. the REED energy consumption slightly increases as the network density increases, and
3. the energy dissipation is not significantly affected by using larger values of *k*.

Phase I. Initialization

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

Phase III. Finalization

```
$S_{un} \leftarrow \{k_i$: $k_i$ is an uncovered CH overlay$\}$
For $i \leftarrow 1$ To $k$
    If (my_CH[i]) and (my_CH[i].state = F_CH)
        join_cluster(i, my_CH[i], NodeID)
    Else $S_{un} \leftarrow S_{un} \cup \{i\}$
If $|S_{un}| > 0$
    $proxy\_CH \leftarrow$ neighbor proxy replies
$i$ = 0
While ($S_{un} \neq \phi$ and $i < |proxy\_CH|$)
    join_cluster($k_i$, proxy_CH[i++], NodeID)
    $S_{un} \leftarrow S_{un} - \{k_i\}$
While ($S_{un} \neq \phi$)
    Cluster_head_msg(NodeID, F_CH, i)
    $S_{un} \leftarrow S_{un} - \{k_i\}$
```

Phase II. Main processing

```
Repeat $N_{initial}$ times
For $i \leftarrow 1$ To $k$
    $S_{CH}[i] \leftarrow \{v$: $v \in CS[i]\}$
    If $S_{CH}[i] \neq \phi$
        my_CH[i] $\leftarrow$ least_cost($S_{CH}[i]$)
        If my_CH[i] = NodeID
            is_CH $\leftarrow$ TRUE
            If ($CH_{prob}$ = 1)
                Cluster_head_msg(NodeID,F_CH,i)
            Else
                Cluster_head_msg(NodeID,T_CH,i)
    ElseIf (!is_CH) and (U(0,1) $\leq CH_{prob}$)
        Cluster_head_msg(NodeID,T_CH, i)
        my_CH[i] $\leftarrow$ NodeID
        is_CH $\leftarrow$ TRUE
$CH_{prob} \leftarrow$ min($CH_{prob} \times 2$, 1)
EndRepeat
Repeat ($N_{complete} - N_{initial}$) times
For $i \leftarrow 1$ To $k$
    If (my_CH[i] = NULL) OR
        (my_CH[i].state $\neq$ F_CH)
        $S_{CH}[i] \leftarrow \{v$: $v$ is CH in overlay $i\}$
        If $S_{CH}[i] \neq \phi$
            my_CH[i] $\leftarrow$ least_cost($S_{CH}[i]$)
EndRepeat
```

FIGURE 9 Pseudo-code of the REED protocal at node u.

Fig. 8(c) shows the energy dissipation for *control overhead* and total energy dissipation during the operation period. The control overhead corresponds to the packets used for clustering. The three curves at the bottom of Fig. 8(c) indicate that the control overhead for *k* = 3 is at most twice that for *k* = 1 (which corresponds to HEED clustering [3]). In addition, the control overhead is reduced as *k* increases (e.g., *k* = 5). This is because the competition for high-index overlays results in less cluster head announcements than low-index overlays, since the cluster head probability is closer to 1 and fewer iterations than $N_{iter\ 2}$ are remaining. Note that the routing update overhead (heartbeat messages) is not accounted for in both the overhead and total dissipation results in Fig. 8(c). The results only account for clustering-related overhead.

The results in Figs. 8(b) and (c), in conjunction with the results in Fig. 7, clearly indicate that the energy cost for achieving significant reliability improvements using REED is extremely small. We therefore conclude that REED is an effective topology management protocol for sensor networks.

5. Related Work

Clustering protocols have been investigated as either independent protocols for organizing ad-hoc networks, e.g., [1], [7], [2], [24], [5], or in the context of routing and power management, e.g., [12], [25], [4], [26]. Weight-based clustering protocols use a generic parameter(s) or weight associated with each node to make clustering decisions [2], [1], [3], [4]. In [24], the authors propose using a spanning tree (or BFS tree). Earlier work also proposed clustering based on degree or lowest identifier heuristics [25], [27], [28], [29]. LEACH [4] is an application-specific data dissemination protocol that uses clustering to prolong the network lifetime. In [6], a multi-level hierarchical structure is proposed, where cluster heads are elected according to their residual energy and degree. 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. Another approach proposed for mobile radio networks [30] builds multiple virtual subnets by carefully assigning node addresses. We proposed HEED [3]– a low complexity clustering protocol that converges in constant time and aims at prolonging the network lifetime and selecting well-distributed cluster heads. The impact of sudden failures due to harsh environmental condition, however, was not considered in all the above studies, including our own previous work.

Our model assumes that sensor nodes are all equally significant and location unaware, unlike protocols such as [14], [31], where 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 remaining nodes are turned off to save energy. In the Probing Environment and Adaptive Sleeping (PEAS) protocol [16], 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 [13].

Several approaches have been proposed to modify the network topology by changing the transmission power levels in such a way that the resulting communication graph is k-connected. Finding the minimum cost k-connected graph is an NP-hard problem [32]. In [33], an approximation of this problem is formulated as a linear programming (LP) problem. Solving the LP problem is difficult for sensor nodes with limited processing capabilities. This approach also requires centralized control. In [34], 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 [35], 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.

6. Conclusions

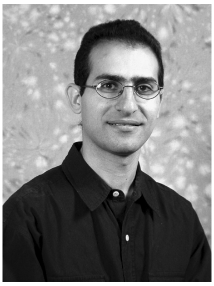
In this paper, we have presented REED, a distributed clustering protocol for robust ad-hoc sensor networks. REED achieves fault tolerance by constructing multiple independent cluster head overlays on top of the physical network. A node joins one cluster in each of the independent overlays. The REED clustering process terminates in a constant number of iterations and has low message and computation overhead. By carefully selecting the cluster and transmission power levels, *k*-connectivity is achieved a.a.s. when node density allows it. Multiple vertex-disjoint routing paths are also available in this case. This is useful for security protocols, such as those using threshold cryptography to withstand node compromises. Simulation results show that REED can significantly increase the number of successful transmissions for both single-hop and multi-hop networks. Results also indicate that the energy cost for achieving significant reliability improvements using REED is extremely small.

Our 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 and duty cycles, and how it performs under different node distribution models.

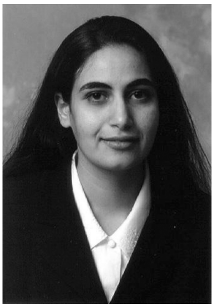
References

1. M. Chatterjeo, S. K. Das, and D. Turgut, "WCA: A weighted clustering algorithm for mobile ad hoc networks," *Cluster Computing*, 193–204, 2002.
2. S. Basagni, "Distributed clustering algorithm for ad-hoc networks," *International Symposium on Parallel Architectures, Algorithms, and Networks (I-SPAN)*, 1999.
3. O. Younis, and S. Fahmy, "Distributed clustering in ad-hoc sensor networks: A hybrid, energy-efficient approach," *Proceedings of IEEE INFOCOM*, (Hong Kong, March, 2004), an extended version appeared in IEEE Transactions on Mobile Computing, **3,** 4, Oct-Dec, 2004.
4. W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "An application-specific protocol architecture for wireless microsensor networks," *IEEE Transactions on Wireless Communications*, **1**, 4, 660–670, October, 2002.
5. A. D. Amis, R. Prakash, T. H. P. Vuong, and D. T. Huynh, "Max-min D-cluster formation in wireless ad hoc networks," *Proceedings of IEEE INFOCOM*, March, 2000.
6. D. Estrin, R. Govindan, J. Heidemann, and S. Kumar, "Next century challenges: Scalable coordination in sensor networks," *Proceedings of ACM MOBICOM*, August, 1999.
7. H. Chan, and A. Perrig, "ACE: An emergent algorithm for highly uniform cluster formation," *Proceedings of the First European Workshop on Sensor Networks (EWSN)*, January, 2004.
8. H. Chan, A. Perrig, and D. Song, "Random key predistribution schemes for sensor networks," *IEEE Symposium on Security and Privacy*, 2003.
9. L. Zhou, and Z. J. Haas, "Securing ad hoc networks," *IEEE Network*, **13**, 6, 24–30, 1999.
10. O. Younis, S. Fahmy, and P. Santi, "Robust communications for sensor networks in hostile environments," *The Twelfth International Workshop on Quality of Service (IWQoS'04)*, Montreal, Canada, June, 2004.
11. J. Hill, R. Szewezyk, 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,* 93–104, 2000.
12. V. Kawadia, and P. R. Kumar, "Power control and clustering in ad hoc networks," *Proceedings of IEEE INFOCOM*, April, 2003.
13. D. M. Blough, and P. Santi, "Investigating upper bounds on network lifetime extension for cell-based energy conservation techniques in stationary ad hoc networks," *Proceedings of ACM MOBICOM*, 2002.
14. Y. Xu, J. Heidemann, and D. Estrin, "Geography-informed energy conservation for ad hoc routing," *Proceedings of ACM MOBICOM,* Rome, Italy 70–84, July, 2001.
15. V. F. Kolchin, B. A. Sevastyanov, and V. P. Chistyakov, in *Random Allocations.* Washington D.C: V. H. Winston and Sons, 1978.
16. F. Ye, G. Zhong, S. Lu, and L. Zhang, "PEAS: A robust energy conserving protocol for long-lived sensor networks," *Proceedings of IEEE International Conference on Distributed Computing Systems (ICDCS)*, 2003.

17. S. Shakkottai, R. Srikant, and N. Shroff, "Unreliable sensor grids: Coverage, connectivity, and diameter," *Proceedings of IEEE INFOCOM*, March, 2003.
18. J. Elson, L. Girod, and D. Estrin, "Fine-grained network time synchronization using reference broadcasts," *Proceedings of OSDI*, 2002.
19. O. Younis, and S. Fahmy, "On time synchronization in multi-hop sensor networks purdue university," *Tech. Rep.* CSD-TR-04–020, June, 2004.
20. S. Ganeriwal, R. Kumar, and M. Srisvastava, "Timing-sync protocol for sensor networks," *Proceedings of ACM Conference on Embedded Networked Sensor Systems (ACM SenSys)*, November, 2003.
21. O. Younis, and S. Fahmy, "An experimental study of routing and data aggregation in sensor networks," *Proceedings of the International Workshop on Localized Communication and Topology Protocols for Ad Hoc Networks (LOCAN'05)* (Washington, DC). November, 2005.
22. M. Perillo, Z. Ignjatovic, and W. Heinzelman, "An energy conservation method for wireless sensor networks employing a blue noise spatial sampling technique," *Proceedings of the Third International Symposium on Information Processing in Sensor Networks (IPSN'04)*, April, 2004.
23. "TinyOS," http://www.tinyos.net, 2005.
24. S. Banerjee, and S. Khuller, "A clustering scheme for hierarchical control in multi-hop wireless networks," *Proceedings of IEEE INFOCOM*, April, 2001.
25. C. R. Lin, and M. Gerla, "Adaptive clustering for mobile wireless networks," *IEEE Journal on Selected Areas in Communications*, September, 1997.
26. B. McDonald, and T. Znati, "Design and performance of a distributed dynamic clustering algorithm for ad-hoc networks," *Annual Simulation Symposium*, 2001.
27. D. J. Baker, and A. Ephremides, "The architectural organization of a mobile radio network via a distributed algorithm," *IEEE Transactions on Communications*, **29**, 11, 1694–1901, 1981.
28. M. Gerla, and J. T. C. Tsai, "Multicluster, mobile, multimedia radio network," *IEEE Transactions on Communications*, **1**, 3, 255–265, 1995.
29. A. K. Parekh, "Selecting routers in ad-hoc wireless networks," *Proceedings of SBT/IEEE International Telecommunications Symposium*, August, 1994.
30. J. Sharony, "An architecture for mobile radio networks with dynamically changing topology using virtual subnets," *ACM Mobile Networks and Applications (MONET)*, **1**, 1, 75–86, August, 1996.
31. A. Cerpa, and D. Estrin, "ASCENT adaptive self-configuring sensor networks topologies," *Proceedings of IEEE INFOCOM*, New York, June, 2002.
32. P. Crescenzi and V. K. (Eds.), *A Compendium of NP-Optimization Problems*. http://www.nadaukth.se/viggo/problemlist/compendium.html, 1998.
33. J. Cheriyan, V. Vempala, and A. Vetta, "Approximation algorithms for minimum-cost k-vettex connected subgraphs," *Proceedings of the 34th ACM Symposium on the Theory of Computing (STOC)*, 2002.
34. M. T. Hajiaghayi, N. Iromorlica, and V. S. Mirrokni, "Power optimization in fault-tolerant topology control algorithms for wireless multi-hop Networks," *Proceedings of ACM MOBICOM*, September, 2003.
35. R. Wattenhofer, L. Li, V. Bahl, and Y. M. Wang, "Distributed topology control for power efficient operations in multi-hop wireless ad hoc networks," *Proceedings of IEEE INFOCOM*, 2001.



Ossama Younis is a post-doctoral research associate in Computer Engineering at the University of Arizona. He received his PhD in Computer Science from Purdue University, USA in August 2005. He received his B.S. and M.S. degrees with honors in Computer Science from Alexandria University, Egypt in 1995 and 1999, respectively. His research interests include wireless sensor network protocols and applications, Internet tomography, and video streaming. He is a member of the IEEE and ACM.



Sonia Fahmy is an associate professor of Computer Science at Purdue University. She received her PhD from the Ohio State University in 1999. 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. She received the National Science Foundation CAREER award in 2003, the Schlumberger foundation technical merit award in 2000 and 2001, and the OSU presidential fellowship for dissertation research in 1998. She is a member of the ACM, Phi Kappa Phi, Sigma Xi, and Upsilon Pi Epsilon, and a senior member of the IEEE.

Paolo Santi received his M.S. PhD degrees in computer science from the University of Pisa, Italy, in 1994 and 2000, respectively. During 2000, he was a research assistant at the Instituto di Elaborazione dell'Informazione, Pisa. From January–June 2001, he visited the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA. On September 2001, he joined the Italian National Research Council, Istituto di Informatica e Telematica, in Pisa. In the period May–December 2003 he visited the Department of Computer Science, Carnegie Mellon University. His research interests include fault diagnosis in multiprocessors and safety-critical systems, the study of structural properties of wireless ad hoc networks, and combinatorial auctions. He is a member of ACM and SIGMOBILE.