

The Overlapped K-hop (OK) Clustering Algorithm

Adel Youssef, Moustafa Youssef, Mohamed Younis, and A. Agrawala

24 July 2005

Contents

1	Introduction	4
1.1	Applications of the OK Clustering Algorithm	8
2	Problem Formulation	9
2.1	System Model	9
2.2	Definitions	10
2.3	The Overlapped K-hop (OK) Clustering Problem	13
3	The OK Protocol Architecture	16
3.1	Data Structures	18
3.2	Messages	19
3.3	Timers	20
4	Performance Evaluation	21
4.1	Coverage, Cluster Overlapping and Connectivity Ratio	27
4.2	Cluster Size	32
4.3	Scalability	35
5	Analysis of the Results	40
5.1	Assumptions	40
5.2	Average Cluster Size	44
5.3	Average Overlapping Degree	45
5.4	Overall Communication Overhead	48
6	Correctness and Complexity	50
7	Related Work	54
8	Conclusions and Future Work	58
A	Appendix	60
A.1	Area of Intersection Between Two Identical Circles	60

Abstract

Clustering is a standard approach for achieving efficient and scalable performance in wireless sensor networks. Clustering algorithms are mostly heuristic in nature and aim at generating the minimum number of disjoint clusters. In this report, we formulate the overlapping multi-hop clustering problem as an extension to the k -dominating set problem. Then we propose a fast, randomized, distributed multi-hop clustering algorithm (OK) for organizing the sensors in a wireless sensor network into *overlapping* clusters with the goal of minimizing the overall communication overhead, and processing complexity. OK assumes a quasi-stationary network where nodes are location-unaware and have equal significance. No synchronization is needed between nodes. OK is scalable; the clustering formation terminates in a constant time regardless of the network topology or size. The protocol incurs low overhead in terms of processing cycles and messages exchanged. We analyze the effect of different parameters (e.g. node density, network connectivity) on the performance of the clustering algorithm in terms of communication overhead, node coverage, and average cluster size. The results show that although we have overlapped clusters, the OK clustering algorithm still produces approximately equal-sized clusters.

Keywords: multi-hop clustering, k -dominating set, weakly connected dominating set, scalability, ad-hoc networks, wireless sensor networks, algorithms.

1 Introduction

In recent years, wireless ad-hoc sensor networks have attracted much interest in the wireless research community as a fundamentally new tool for a wide range of monitoring and data-gathering applications. In general sensor networks classify as ad-hoc networks, however, sensor networks have their own unique features. Sensor networks typically consist of hundreds to thousands of unattended sensor nodes randomly spread over the area that is to be probed. In a typical architecture of a sensor network, the data collected by each sensor is communicated through the network, using a radio, to a single processing command center that uses all reported data to determine characteristics of the environment or detect

an event. Sensor nodes are significantly constrained in the amount of available resources such as energy, storage and computational capacity. Due to energy constraints, a sensor can communicate directly only with other sensors that are within a small distance. To enable communication between sensors not within each other's communication range, the sensors form a multi-hop communication network. Sensor nodes are usually assumed to be static. These constraints make the design and operation of sensor networks considerably different from contemporary ad-hoc networks.

Clustering is a standard approach for achieving efficient and scalable performance in wireless sensor networks. Clustering facilitates the distribution of control over the network and, hence, enables locality of communication. Clustering nodes into groups saves energy and reduces network contention because nodes communicate their data over shorter distances to their respective clusterheads. The clusterheads forward the aggregated information to the base station. Only the clusterheads need to communicate far distances to the base station; this burden can be alleviated further by hierarchical clustering, i.e., by applying clustering recursively over the clusterheads of a lower level.

Many clustering protocols have been investigated as either standalone protocols [6, 33, 8, 10, 31, 49, 9, 21, 4, 26, 47, 7, 5, 60, 39, 32] or as a side effect of other protocol operations, e.g., in the context of routing protocols [44, 48, 39], or in topology management protocols [58, 22, 18]. The majority of those protocols construct clusters where every node in the network is no more than 1 hop away from a cluster head [6, 33, 10, 8, 31, 60, 32, 39]. We call these *single hop* (1-hop) clusters. In large networks this approach may generate a large number of cluster heads and eventually lead to the same problem as if there is no clustering. Few papers have addressed the problem of *multi-hop* (k -hop) clustering [5, 7]. These algorithms are mostly heuristic in nature and aim at generating the minimum number of disjoint clusters such that any node in any cluster is at most k hops away from only one cluster head. The proposed OK clustering algorithm belongs to the multi-hop category.

In the last few years, there have been few clustering algorithms designed for sensor networks [60, 7, 39, 32, 37, 35]. Most of those algorithms aim at generating the minimum number of disjoint clusters that maximize the network lifetime. The algorithms discussed in [39, 60, 7] are randomized where the sensors elect themselves as cluster heads with some

probability p and broadcast their decisions to neighbor nodes. The remaining sensors join the cluster of the cluster head that requires minimum communication energy. The proposed OK clustering protocol belongs to the class of randomized algorithms. Both the HEED algorithm [60] and LEACH algorithm [39] form single-hop non-overlapping clusters with the objective of prolonging network lifetime. In [7], the authors proposed a LEACH-like randomized multi-hop clustering algorithm for organizing the sensors in a hierarchy of clusters with an objective of minimizing the energy spent in communicating the information to the processing center. None of the above algorithms construct *overlapping* clusters.

In this report, we propose a fast, randomized, distributed multi-hop clustering algorithm (OK) for organizing the sensors in a wireless sensor network in *overlapping* clusters. After the termination of the clustering process, each node is either a cluster head or within k hops from *at least one* cluster head, where k (*cluster radius*) is a parameter in the algorithm. To the best of our knowledge, this is the first paper to discuss the problem of overlapping multi-hop clustering. OK operates in quasi-stationary networks where nodes are location-unaware and have equal significance. The protocol incurs low overhead in terms of processing cycles and messages exchanged. OK was designed with the following goals:

1. Is completely distributed (i.e. each node independently makes its decisions based on local information and without any centralized control).
2. Is scalable in terms of processing time (i.e. the clustering process terminates in a constant time independent of network size) and in terms of communication overhead (the number of control messages transmitted by node is independent of network size).
3. Does not make any assumptions about the location of the nodes.
4. Is asynchronous (Due to the large number of nodes involved, it is desirable to let the nodes operate asynchronously. OK does not assume any kind of clock synchronization between nodes, hence, The clock synchronization overhead is avoided, providing additional processing savings).

5. Is energy efficient in terms of processing complexity and message exchange (control overhead is linear in the number of nodes).
6. Is efficient in terms of memory used by the data structures required to implement the algorithm.
7. Chooses cluster heads that are well distributed over the sensor field.
8. Allows multi-hop clusters to be formed.
9. Ensures overlapped clusters with some average overlapping degree.

To the best of our knowledge, the proposed algorithm is the first algorithm to address the above goals in an integrated manner. We formulate the overlapping k -hop clustering problem as an extension to the k -dominating set problem [38]. Then we propose OK, a randomized multi-hop distributed algorithm to solve the problem. The nodes randomly elect themselves as cluster heads with some probability p . The cluster head probability (p) is another parameter in the algorithm that can be tuned to control the number of overlapping clusters in the network. The clustering process terminates in $O(1)$ iterations, independent of the network diameter. It does not depend on the network topology or size. We also analyze the effect of different parameters (e.g. node density, network connectivity) on the performance of the clustering algorithm in terms of communication overhead, node coverage, and average cluster size. The results show that although we have overlapped clusters, the OK clustering algorithm still produces approximately equal-sized clusters, which is a desirable property because it enables an even distribution of control between cluster head nodes.

The paper is organized as follows. In the remainder of this section we go through some of the applications of the proposed algorithm. Section II describes the network model and states the problem that we address in this work. Section III presents the OK protocol architecture and proves that it satisfies its design goals. Section IV shows the performance of OK via simulations and in section V, we provide analytical models for the results. We study the complexity and correctness of the proposed protocol in section VI. Then, sec-

tion VII briefly surveys related work. Finally, Section VIII gives concluding remarks and suggestions for future work.

1.1 Applications of the OK Clustering Algorithm

Most of clustering algorithms have a primary goal of producing approximately equal-sized non-overlapping clusters. Equal-sized clusters is a desirable property because it enables an even distribution of control (e.g., data processing, aggregation, storage load) over cluster heads; no cluster head is overburdened or under-utilized. However, having overlapping clusters with some degree is desirable and beneficial in some applications (e.g. node localization [62, 61, 53, 43], routing [45], TDMA-based MAC [57]). The nodes that belong to two or more clusters can serve as gateways for inter-cluster head communication when the cluster heads do not have long range communication capabilities.

Overlapped clusters can boost the network robustness against node failure or compromise. Given the resource constraints, sensor nodes become dysfunctional rather fast when they deplete all their on-board energy supply. In addition, sensors are often deployed in harsh environments and thus exposed to damage. Moreover, sensor networks usually operate unattended making the nodes an easy target for capture by an adversary. While the loss of a sensor node can be tolerated given the large node population, recovering from the failure or the compromise of a cluster-head is a challenge. Pursuing contemporary clustering schemes often requires provisioning for recovery through the deployment/designation of a spare cluster-head, which causes resource underutilization, or through performing network re-initialization to form new clusters, which is a slow and disruptive process. Establishing overlapped clusters would facilitate and expedite the recovery process since nodes can join others alternate clusters.

Another application for overlapping clusters is *anchor-free* localization. Recently, there has been a great interest in *anchor-free (GPS-free)* node localization especially in the context of sensor networks [62, 61, 17, 54, 53, 43]. Anchor-free localization algorithms try to compute nodes' positions without the use of anchor nodes (i.e. nodes that know their positions usually using GPS). In this case, instead of computing absolute node positions,

the algorithm estimates relative positioning, in which the coordinate system is established by a reference group of nodes [62, 17, 54, 53]. The network is divided into small clusters of nodes where each cluster has its own coordinate system. Then the cluster heads communicate with each other in order to calculate the global network topology by transforming from one coordinate system to another. In order for two cluster heads to perform this transformation, there must be at least three *boundary nodes*¹ (i.e. the two clusters are overlapping with degree at least 3). The proposed OK algorithm can be used in this case to guarantee with high probability that the clusters have an overlapping degree of at least three.

Overlapping clusters can also be used in case of cluster-based routing protocols [20, 45, 30]. For example, the authors in [45] developed a routing protocol that uses a single boundary node in order to route between overlapping clusters (in this case the overlapping degree is 1). Although they used only one boundary node to simplify the clustering algorithm, they recommended using multiple boundary nodes. Using multiple boundary nodes will be more robust and also distribute packet-forwarding load between clusters. Their algorithms can be extended easily to benefit from overlapping clusters as generated by OK.

2 Problem Formulation

An ad-hoc network can be modeled as a graph $G = (V, E)$, where two nodes are connected by an edge if they can communicate with each other. If all nodes are located in the plane and have the same *transmission range* (T_r), then G is called a *unit disk graph*. We will start by describing the considered system model. Then, we will review a number of definitions from graph theory that will be used in the problem formulation. Finally, we will formulate the overlapping k -hop clustering problem as an extension to the k -dominating set problem.

2.1 System Model

We consider a wireless sensor network where all nodes are alike and each node has a unique id. The nodes are location-unaware, i.e. not equipped with GPS. There are neither base stations nor infrastructure support to coordinate the activities of subsets of nodes.

¹A *boundary node* is a node that belongs to more than one cluster.

Therefore, all the nodes have to collectively make decisions. We assume that the nodes are static. This assumption about node mobility is typical for sensor networks. All sensors transmit at the same power level and hence have the same transmission range (T_r). Each sensor uses 1 unit of energy to transmit or receive 1 unit of data. We also assume that nodes have timers, but we do not require time synchronization across the nodes. Timers are used for tasks such as timing out of a node when waiting on a condition.

All communication is over a single shared wireless channel. A wireless link can be established between a pair of nodes only if they are within wireless range of each other. The OK algorithm only considers bidirectional links. It is assumed the MAC layer will mask unidirectional links and pass bidirectional links to OK. Two nodes that have a wireless link will, henceforth, be said to be 1-hop away from each other. They are also said to be immediate neighbors. Nodes can identify neighbors using beacons.

2.2 Definitions

Let n denote the number of vertices (nodes) and e denote the number of edges. That is, $n = |V|$ and $e = |E|$. Let S be a subset of nodes. We shall use $\langle S \rangle$ to denote the subgraph induced by the set S .

- *Open Neighbor Set*, $N(u) = \{v | (u, v) \in E\}$, is the set of vertices that are neighbors of u . For a set of nodes S , $N(S) = \bigcup_{u \in S} N(u)$.
- *Closed Neighbor Set*, $N[u] = N(u) \cup \{u\}$, is the set of neighbors of u and u itself. For a set of nodes S , $N[S] = \bigcup_{u \in S} N[u] = N(S) \cup S$.
- *Node Degree*, $deg(u) = |N(u)|$.
- *Graph Distance*, $d_G(u, v)$, the distance between two vertices u and v is the minimum number of edges in a $u - v$ path.
- *Graph Power*, the k th power of a graph G (G^k) is a graph with the same set of vertices as G and an edge between two vertices iff there is a path of length at most k between them [55]. Given $G = (V, E)$ then $G^k = (V, E^k)$ where $E^k = \{(u, v) | u, v \in V \text{ and } d_G(u, v) \leq k\}$.

- *Independent Set*, is a subset of V such that no two vertices within the set are adjacent in V .
- *Maximal Independent Set (MIS)*, is an independent set such that adding any vertex not in the set breaks the independence property of the set. Thus, any vertex outside of the maximal independent set must be adjacent to some vertex in the set. Finding the MIS is NP-Hard [34].
- *Dominating Set*, S , is defined as a subset of V such that each vertex in $V - S$ is adjacent to at least one vertex in S . Thus, every MIS is a dominating set. However, since vertices in a dominating set may be adjacent to each other, not every dominating set is an MIS. Finding a minimum-sized dominating set or MDS is NP-Hard [34].
- *Minimum Dominating Set (MDS)* is the dominating set with minimum cardinality. Each MIS is also an MDS. Finding the MDS is also NP-Hard [34].
- *Connected Dominating Set (CDS)*, S , is a dominating set of G that induces a connected subgraph of G (i.e. $\langle S \rangle$ is connected). One approach to constructing a CDS is to find an MIS, and then add additional vertices as needed to connect the vertices in the MIS.
- *Minimum Connected Dominating Set (MCDS)* is a CDS with minimum cardinality. Finding the MCDS is also NP-Hard [34].
- *Weakly Connected Dominated Set (WCDS)*, S , is a dominating set such that $N[S]$ induces a connected subgraph of G (i.e. $\langle N[S] \rangle$ is connected). Given a connected graph G , all of the dominating sets of G are weakly connected. Computing a minimum WCDS is NP-Hard [34].
- *Total Dominating Set*, S , is a *total dominating set* if every vertex $u \in V$ is adjacent to a vertex in S . In this context a vertex does not dominate itself. Equivalently, $\bigcup_{s \in S} N(s) = V$.

The above definitions can be generalized for the multi-hop (k -hop) case as follows:

- *k-Connected Set*, S , a set S is said to be k connected if each vertex in S is within distance k from at least one other vertex in S , where $k \geq 1$ is an integer.
- *Open k-Neighbor Set*, $N_k(u) = \{v | d_G(u, v) \leq k\}$, is the set of vertices, different from u , that are at distance at most k from u . For a set of nodes S , $N_k(S) = \bigcup_{u \in S} N_k(u)$.
- *Closed k-Neighbor Set*, $N_k[u] = N_k(u) \cup \{u\}$, is the set of k -neighbors of u and u itself. If u is a cluster head, then $N_k[u]$ is the set of all vertices in the cluster and $|N_k[u]|$ is the cluster size. For a set of nodes S , $N_k[S] = \bigcup_{u \in S} N_k[u] = N_k(S) \cup S$.
- *Node k-Degree*, $deg_k(u) = |N_k(u)|$.
- *k-Independent Set (KIS)*, S , is a subset of V such that for any two vertices $u, v \in S$, $d_G(u, v) > k$.
- *k-Dominating Set (KDS) OR Distance Domination*, S , is defined as a subset of V such that each vertex in $V - S$ is within distance k from at least one vertex in S , where $k \geq 1$ is an integer. That is $N_k[S] = V$.
- *k-Connected Dominating Set (KCDS)*, S , is a k -dominating set of G that induces a connected subgraph of G (i.e. $\langle S \rangle$ is connected).
- *k-Weakly Connected Dominating Set (KWCDs)*, S , is a k -dominating set of G and S is a $2k$ connected set (i.e. each vertex in S is within distance $2k$ from at least one other vertex in S , where $k \geq 1$ is an integer).
- *Total k-Dominating Set OR Total Distance Domination*, Let $k \geq 1$ be an integer, a set S is a *total k-dominating set* if every vertex $u \in V$ is within distance k from at least one vertex in S other than itself (i.e. a vertex does not k -dominate itself).
- *k-Independent Dominating Set (KIDS)*, is a subset of V that is both k -independent and k -dominating.

To clarify the above definitions, we will use the graph in Fig. 1 as an illustrative example. For the graph shown, $S_1 = \{A, G\}$ is a 2-dominating set (2DS) and it is also

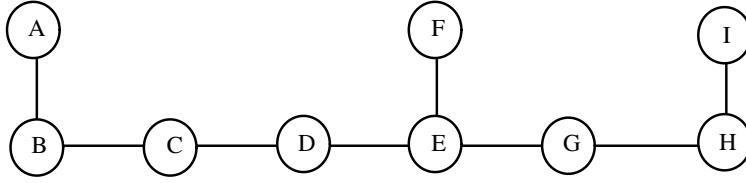


Figure 1: Illustrative Example

a 2-independent set (2IS); hence; S_1 is a 2-independent-dominating set (2IDS). The set $S_2 = \{C, D, E, G\}$ a 2-connected-dominating set (2CDS). The set is $S_3 = \{C, G\}$ is a 2-weakly-connected-dominating set (2WCDS). The set $S_4 = \{C, E, H\}$ a total 2-dominating set.

2.3 The Overlapped K-hop (OK) Clustering Problem

Given an ad-hoc network that is modelled as a unit disk graph $G = (V, E)$, the OK clustering problem can be formulated as finding the set of nodes S such that:

1. *Coverage Condition.* S is a KDS. This means that each node is either a cluster head or within k hops from a cluster head (i.e. $N_k[S] = V$).
2. *Overlapping Condition.* For each node $u \in S \exists$ at least one node $v \in S$ such that $N_k[u] \cap N_k[v] \geq \omega$, where ω is a certain threshold. In other words, for each cluster, there exists at least one other cluster that overlaps with it with overlapping degree $\geq \omega$.
3. *Connectivity Condition.* S is k -Weakly Connected Dominating Set (KWCDs). This means that S is a $2k$ connected set (i.e. each node in S is within distance $2k$ from at least one other node in S , where $k \geq 1$ is an integer). This condition guarantees that the graph induced by the set of cluster heads is $2k$ connected which implies that for each cluster head node u , there is at least one other cluster head node v such that $d_G(u, v) \leq 2k$; hence; the two clusters are overlapped with degree ≥ 1 .

Finding the minimum KDS (MKDS) is a nice design goal to achieve. Minimizing the cardinality of the computed KDS can help to decrease the control overhead since broadcasting

for route discovery and topology update is restricted to a small subset of nodes. Therefore the broadcast storm problem [59] inherent to global flooding can be greatly decreased. However, from a computational point of view, the problem of finding the minimum KDS (*MKDS*) is very difficult. In fact there is no known efficient centralized algorithm for solving this problem and a corresponding decision problem is NP-hard [38]. Even if the graph G belongs to certain special classes of graphs (for example if G is bipartite or chordal graph), the problem remains NP-hard [13]. The *MKDS* remains also NP-hard for unit-disk graphs as the case in wireless ad-hoc networks. Further aspects of the commutability of *MKDS* are discussed in [38, 19].

In [41], the authors described a centralized algorithm that finds a KDS of cardinality at most $n/(k+1)$. The algorithm firsts creates a rooted spanning tree from the original network topology. Then, an iterative labeling strategy is used to classify the nodes in the tree to be either dominator (cluster head) or dominated (non-cluster head). In [40], the authors described another centralized algorithm for finding the total KDS such that the cardinality is bounded by $2n/(2k+1)$. Since both algorithms are centralized, the communication overhead is high in case of large-scale networks like sensor networks. There is no known efficient distributed algorithm for finding the *MKDS* with some performance bound. For example, the MaxMin heuristic [5] finds a KDS, however, there is no reported performance bound on the cardinality of the resulting KDS. Similarly, in [7] the objective is to find a KDS that minimize energy consumption and maximize network lifetime.

A related problem that has been widely investigated in the context of wireless networks is the problem of finding the minimum connected dominating set (*MCDS*). The *MCDS* problem can be viewed as a special case of *MKDS* problem when $k = 1$. The *MCDS* is NP-hard for general graphs and for unit-disk graphs in particular [28]. Although there are many applications for CDS in wireless networks [12], the primary application of CDS is the construction of virtual backbone (spine) in wireless ad hoc networks. In the last decade, many CDS construction algorithms have been proposed in the context of MANETs and sensor networks. These algorithms are either centralized [14, 15, 24, 36] or distributed [3, 1, 2, 23, 25, 16, 56]. The centralized approaches seek a minimum connected dominating set (*MCDS*) as their major design goal. Thus performance bounds are

their primary design parameter. However, centralized algorithms have high communication overhead and time complexity. On the other hand, distributed algorithms seek a connected dominating set (not necessarily the minimum) that provides a good resource conservation property. Thus performance bound is not their primary consideration. Instead, time complexity (specially when nodes are mobile) and message complexity is taken into consideration. Distributed algorithms have a time complexity of $O(n)$ and a message complexity of $O(n \log n)$ [1, 25] or $O(n)$ [2, 16]. This quicker execution time comes at a cost of a larger CDS. A more detailed analysis of the performance of those algorithms is discussed at [12].

Any of the distributed heuristics for finding a CDS can be modified to find a KDS. In this case, we need to construct a k -closure (a graph power of order k) on the original connectivity network graph before running any of the heuristics. Recall from section 3.2 that the k th power of the graph yields a modified graph in which nodes A and B are 1-hop neighbors if they were at most k -hops away in the actual topology graph. When any of the distributed CDS heuristics are run on this modified graph, they form clusters where each node is at most k wireless hops away from its cluster head. Constructing the k th power of a graph is $O(kn^3)$, where n is the number of vertices in the graph [55]. Even if we used Strassen's algorithm for matrix multiplication [29], the best performance in terms of floating point operations is $O(kn^{2.807})$. For sensor networks, this is considered very expensive not in terms of communication overhead only but also the Strassen's algorithm is difficult to implement efficiently because of the data structures required to maintain the array partitions [29, 42]. Moreover, we are still generating non-overlapping clusters! Modifying an existing distributed CDS algorithm, to generate a KDS in a distributed randomized fashion, is a challenging problem in itself. We leave this as a future work.

The problem of overlapping clusters is totally new. There was no formulation of the problem in the literature. So there is no known algorithm that satisfies the two conditions described at the beginning of this section. The proposed OK clustering algorithm is a distributed simple randomized algorithm that meets the above two conditions with high probability. The main design goal behind the proposed algorithm is not to find the minimum KDS. Thus performance bound is not the primary consideration. Instead, we are more concerned about time complexity, processing complexity, and message complexity. We

will show that by tuning some of the protocol parameters (k , p , node density), we can generate with high probability overlapping clusters with some average overlapping degree. OK is scalable; the clustering formation terminates in a constant time $O(k)$ regardless of the network topology or size. The protocol incurs low overhead in terms of processing cycles and messages exchanged. OK assumes a quasi-stationary networks where nodes are location-unaware and have equal significance. No synchronization is needed between nodes. In general, OK will produce a an overlapping KDS with the goal of minimizing the overall communication overhead, and processing complexity. We will discuss in the results section how we can tune the parameters of the algorithm to guarantee a weakly connected KDS (*KWCDS*) with high probability.

3 The OK Protocol Architecture

In this section we describe the operations of the OK protocol in more detail. The essential operation in any clustering protocol is to select a set of cluster heads among the nodes in the network, and cluster the rest of the nodes with these heads. OK does this in a distributed fashion, where nodes make autonomous decisions without any centralized control. The algorithm initially assumes that each sensor in the network becomes a cluster head (*CH*) with probability p . Each cluster head then advertises itself as a cluster head to the sensors within its radio range. This advertisement is forwarded to all sensors that are no more than k hops away from the *CH*. Any sensor that receives such advertisements joins the cluster even if it already belongs to another cluster. Any sensor that is neither a *CH* nor has joined any cluster itself becomes a *CH*. Since the advertisement forwarding is limited to k hops, if a sensor does not receive a *CH* advertisement within time duration t_1 (where t_1 units is a value greater than the time required for data to reach the cluster head from any sensor k hops away), it can infer that it is not within k hops of any cluster head and hence become a *CH*. We assume that each cluster has a unique identifier, which is the node identifier of the cluster head. The flowchart of the OK algorithm is shown in Fig. 2. Each node maintains a table that stores information about the clusters known to this node. If the table contains more than one entry, this means that the node is a *boundary node*.

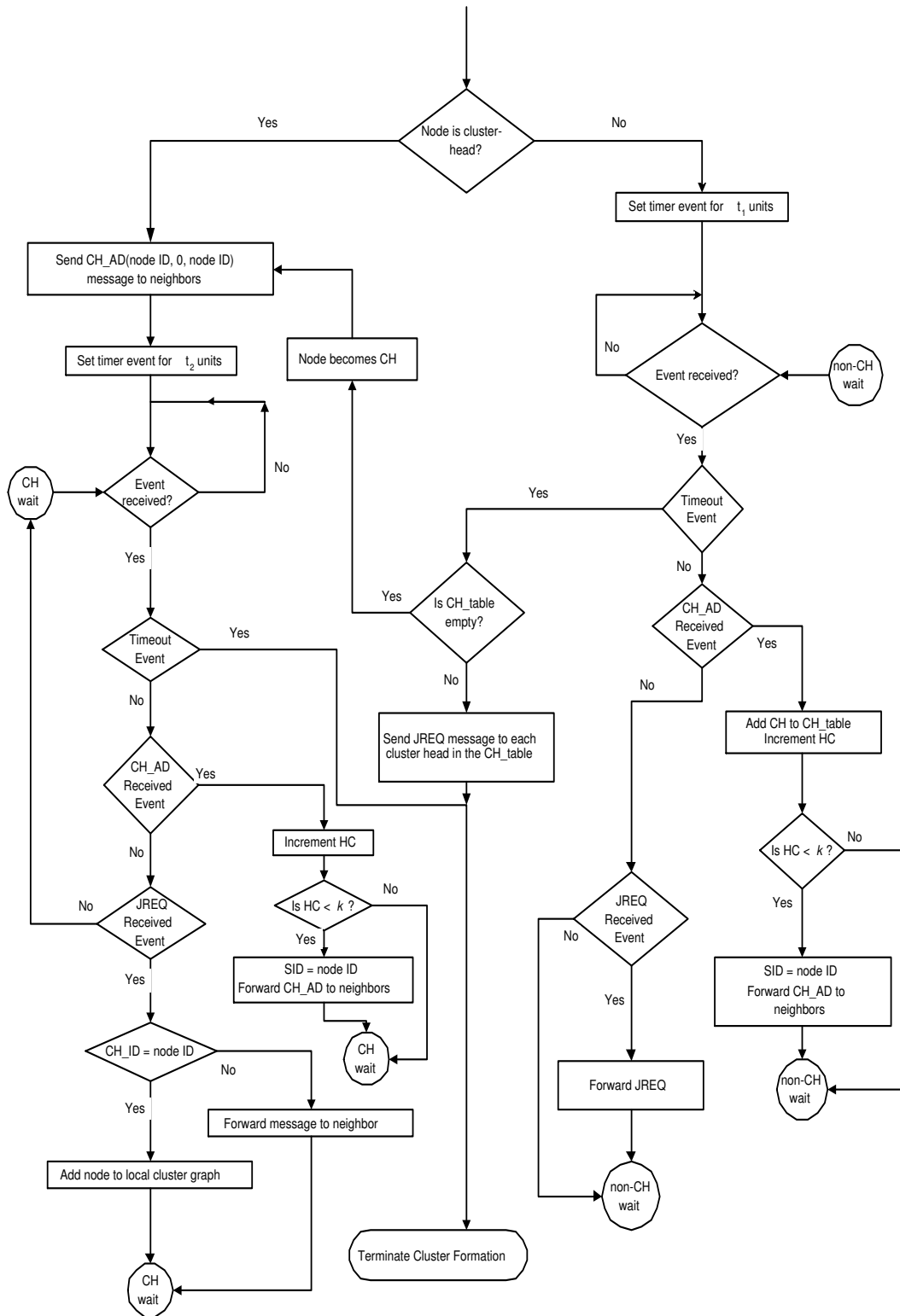


Figure 2: Flowchart of the OK cluster formation algorithm

Each cluster head maintains a list of all cluster members, a list of adjacent clusters, and a list of boundary nodes to reach those clusters. There can be multiple boundary nodes between overlapping clusters. Moreover, a node can be a boundary node for more than two overlapping clusters. In the remainder of this section, we first discuss the necessary data structures to be maintained at each node for the clustering protocol. We also discuss the message formats and the timers maintained by each node. We then explain the cluster formation protocol and give pseudo-code. Finally, we prove that the protocol meets its design goals.

3.1 Data Structures

Each node maintains the following variables:

- Node ID (*NID*): A unique ID assigned to each node before deploying the network.
- *Status*: {CH, NCH}. The status of the node. A node can be either a cluster head (CH) or a non-CH (NCH). Initially all nodes are set to NCH.
- Node Degree (*d*): The number of 1-hop neighbors. Calculated after discovering the number of neighbors.
- Reliable Ranges (*RR*): The number of reliable ranges known to the node. $RR \leq d$. Initialized during range estimation phase.
- Local Cluster Graph (*LCG*): $LCG = (V; E)$, a weighted undirected graph maintained by CH nodes corresponding to the local cluster that belongs to this CH node. The edge weight (w_{ij}) represents the range measurement between nodes i and j . Initially *LCG* consists of the CH node and all one-hop neighbors that it hears from during range estimation phase, i.e. $|V| = d+1$ and $|E| = d$, where d is the CH node degree.
- Adjacent Clusters Table (*AC_table*): A table maintained by CH nodes to store information about adjacent clusters. The table consists of tuples of the form (*CHID*, *BN*), where *CHID* is the CH node ID, and *BN* is a list of *boundary node* Ids. Initially the table is empty.

- Cluster Heads Table (CH_table): A table maintained by each node to store information about the clusters known to this node. If the table contains more than one entry, this means that the node is a *boundary node*. The table consists of tuples of the form $(CHID, HC, prev)$, where $CHID$ is the CH node ID, HC is the number of hops leading to this cluster head, and $prev$ is the node ID of a 1-hop neighbor node that can lead to the CH node of this cluster using minimum number of hops. The table acts as a routing table where the $CHID$ field uniquely identifies a route to a CH node. Initially the table is empty.

3.2 Messages

There are two types of messages:

- CH advertisement (CH_AD) message: This is the message broadcast by a CH node to advertise its existence. It has the form $(CH_AD, SID, CHID, HC)$, where SID is the sender node ID, $CHID$ is cluster head ID, and HC is the number of hops leading to the CH node. The SID field is used to update the $CH_table.prev$ field such that each node knows a unique path to the cluster head. The HC field is used to limit the flooding of the CH_AD message to k hops. The CH_AD message has a fixed size.
- Join request (JREQ) message: This is a message sent by a node when it knows about the existence of a CH node and decides to join this cluster. To limit the flooding, the message is unicasted using the field $CH_table.prev$. The message contains information about the range measurements to neighbors along with the clusters that this node can hear from. The message has the form $(JREQ, RID, SID, CHID, d, (NID, R_{SID,NID})_{1..nd}, nc, (CHID)_{0..nc})$, where RID is the receiver ID², SID is the ID of the node that will join the cluster, $CHID$ is the ID of the CH node responsible for this cluster, d is the node degree, $(NID, R_{SID,NID})_{1..nd}$ are one or more couples containing information about the range measurements between this node and its 1-hop neighbors, nc is the number of clusters that this node can hear from them ($=|AC_table|$), and $(CHID)_{0..nc}$ are 0 or more clusters that this node can hear from.

²This equals to $CH_table.prev$ corresponding to the $CH_table.CHID$.

Notice that the size of the JREQ message is variable and depends on the number of clusters (nc) and the node degree (d).

3.3 Timers

Each node maintains the following timers³:

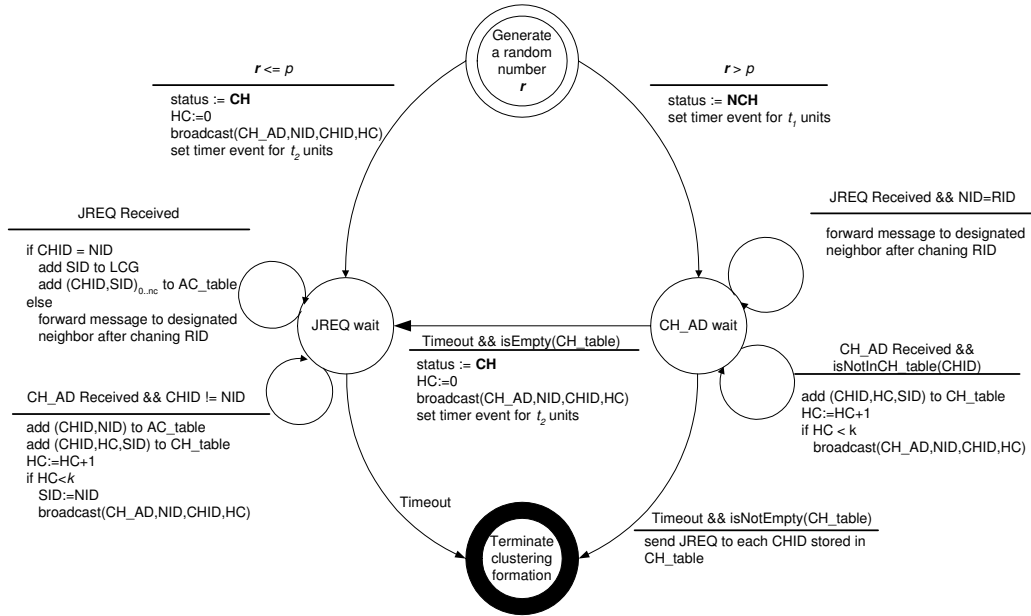


Figure 3: Finite state machine of the OK protocol

- **CH_AD_WAIT** timer. This timer is set by a non-CH node. It represents the maximum time that a node should wait for CH advertisement messages. It is equal to $t(k) + \delta$, where $t(k)$ is the time needed for a message to travel k hops and δ is the maximum time needed for any node to finish bootstrapping and start executing the OK protocol. In our simulator, we assume that all the CH nodes will finish bootstrapping and start transmitting CH_AD messages within $t(k)/2$ time units. Hence, we set δ to be $t(k)/2$.
- **JREQ_WAIT** timer. This timer is set by a CH node. It represents the maximum time that a CH node should wait for JREQ messages. It is approximately equal to $2 * t(k) + \delta$.

³We assume a timer that is set to a certain number of units and fires once.

The events of the OK clustering algorithm are listed in table 1. A finite state machine for the protocol is given in Fig. 3. The activities of the OK clustering algorithm are shown in Fig. 4, using an event-based notation.

Event Name	Description
Initialization()	An event executed once to initialize the status of the node.
CH_AD_Received (<i>SID</i> , <i>CHID</i> , <i>HC</i>)	An event triggered when CH_AD message is received.
JREQ_Received (<i>RID</i> , <i>SID</i> , <i>CHID</i> , <i>nd</i> , (<i>NID</i> , <i>RSID</i> , <i>NID</i>) _{1..nd} , <i>nc</i> , (<i>CHID</i>) _{0..nc})	An event triggered when JREQ message is received.
ChangeStatus	An event triggered when the CH_AD_WAIT timer fires indicating that an NCH node should either change its status to CH node or join a cluster if any.
EndClusterFormationPhase	An event triggered when the JREQ_WAIT timer fires indicating that a CH node should terminate the clustering phase and start the Local Location Discovery (LLD) phase.

Table 1: Events summary of the OK clustering algorithm

4 Performance Evaluation

We have validated the OK clustering algorithm using simulation. The OK clustering algorithm was implemented using MATLAB 6.1 release 12.1. Initially, each node is assigned a unique node id. There are four parameters used in our simulation:

```

Initialization() // executed once
1. ac:
2. r = generate random number from 0..1;
3. if r < p then
4.   status := CH;
5.   broadcast (CH_AD, NID, NID, 1);
6.   set JREQ_WAIT timer;
7. else
8.   status := NCH;
9.   set CH_AD_WAIT timer;

CH_AD_Received (SID, CHID, HC)
10. ac: if status = NCH
11.   if CHID is not in the CH_table
12.     Add (CHID, HC, SID) to CH_table;
13.     if HC < k
14.       HC := HC + 1;
15.       broadcast (CH_AD, NID, CHID, HC);
16.     // else HC ≥ k, do not forward the message more than k hops
17.     // else you have already heard of this cluster, do nothing
18. else
19.   // node is a CH node
20.   if CHID = NID
21.     discard the message; // This is an echo message
22.   if CHID is not in the AC_table
23.     Add (CHID, NID) to AC_table;
24.     Add (CHID, HC, SID) to CH_table;
25.     if HC < k
26.       HC := HC + 1;
27.       broadcast (CH_AD, NID, CHID, HC);
28.     // else HC ≥ k, do not forward the message more than k hops
29.     // else you have already heard of this cluster, do nothing

JREQ_Received (RID, SID, CHID, nd, (NID, RSID, NID)1..nd, nc, (CHID)0..nc)
30. ac: if status = NCH
31.   if RID = NID
32.     RID := CH_table[CHID].prev;
33.     broadcast (JREQ, RID, SID, CHID, nd, (NID, RSID, NID)1..nd, nc, (CHID, cost)0..nc);
34.   // else do nothing to limit the flooding of JREQ message
35. else
36.   // node is a CH node
37.   if CHID = NID
38.     Add SID to the set of vertices in LCG;
39.     Add (NID, RSID, NID)1..nd to the set of edges in LCG;
40.     Add (CHID, cost, SID)0..nc to the AC_table;
41.   else
42.     RID := CH_table[CHID].prev;
43.     broadcast (JREQ, RID, SID, CHID, nd, (NID, RSID, NID)1..nd, nc, (CHID, cost)0..nc);

EndClusterFormationPhase
44. ec: JREQ_WAIT timer fires. // for CH node
45. ac: Start the Local Location Discovery (LLD) phase using information stored in LCG and AC_table.

ChangeStatus
46. ec: CH_AD_WAIT timer fires. // for NCH node
47. ac: if CH_table empty
48.   status := CH;
49.   broadcast (CH_AD, NID, NID, 1);
50.   set JREQ_WAIT timer;
51. else
52.   for all CHID in CH_table
53.     RID := CH_table[CHID].prev;
54.     broadcast (JREQ, RID, NID, CHID, (NID, RSID, NID)1..d, (CHID)0..m);

```

Figure 4: The OK Algorithm

1. *Network size (n)*: the number of sensor nodes in the network. Since all the simulation experiments assume a square area of side length l , changing the network size will implicitly change the node density in the network (μ). Node density (μ) is defined to be the number of nodes in unit area:

$$\mu = n/l^2 \quad (1)$$

2. *Cluster radius (k)*: the maximum graph distance between any node in the cluster and the cluster head. Recall from section 3.2 that the graph distance between two vertices u and v , $d_G(u, v)$, is the minimum number of edges in a $u - v$ path. Hence, if u is a CH node, then $k = \max_{v \in N_k(u)} (dist_G(u, v))$.

3. *Average Node Degree (d)*: the average node degree in the network. Recall from section 3.2, the node degree of a node u , is the number of nodes that are neighbors of u . Node degree is a function of the node transmission range (T_r). Assuming that n sensor nodes are uniformly distributed over a square field of side l , the probability $P(d)$ of a node u having degree d is given by binomial distribution [52]:

$$P(d) = P_r^d (1 - P_r)^{n-d-1} \binom{n-1}{d} \quad (2)$$

where P_r is the probability of being within the transmission range T_r from node u

$$P_r = \frac{\pi \cdot T_r^2}{l^2} \quad (3)$$

For large values of n tending to infinity, the above binomial distribution converges to a Poisson distribution:

$$P(d) = \frac{\lambda^d}{d!} e^{-\lambda} \quad (4)$$

where $\lambda = nP_r$ is the average node degree. Hence, the relation between the average node degree (d) and the transmission range (T_r) of a node is given by:

$$d = nP_r = \frac{n \cdot \pi \cdot T_r^2}{l^2} = \mu \cdot \pi \cdot T_r^2 \quad (5)$$

We will use the above equation frequently to map between average node degree and transmission range.

4. The cluster head probability (p). Since each node decides randomly to be a cluster head with probability p , then the average number of clusters is pn . Hence, increasing p will increase the number of clusters in the network.

All experiments were performed over 150 different topologies representing different network sizes (n) ranging from 50 to 800 sensor nodes. The nodes were randomly placed according to a uniform distribution on a 100x100 area. For each topology, the transmission range of each node (T_r) was varied in order to achieve different average *node degree* (d) ranging from 7 to 21. In a wireless ad-hoc network with a uniform distribution of nodes, in order to guarantee global network connectivity, the average node degree should be at least 6 [46]. Hence, we chose the minimum average node degree to be 7. The cluster radius (k) ranges from 1 to 5. The cluster head probability (p) was varied from 0.05 to 0.5. For each topology, since cluster heads are chosen randomly, we repeat the experiment 30 times, each time with a different random set of cluster heads. To evaluate the performance of the OK clustering algorithm, we use the following performance metrics:

1. *Percentage of Covered Nodes (CN)*: this metric tests if the generated clusters satisfy the *coverage condition* as defined in section 2.3. CN is defined as the percentage of nodes that are either cluster heads or within k -hops from a cluster head after the first wave of CH advertisement is propagated through the network (i.e. after $t(k)$ time units where $t(k)$ is the time needed for a message to be forwarded for k hops). We will prove in section 6 (*lemma 6.2*) that after $3t(k) + \delta$, the OK clustering terminates and each node is either CH or NCH.
2. *The Average Overlapping Degree (AOD)*: this metric tests if the generated clusters satisfy the *overlapping condition* as defined in section 2.3. AOD is defined as the average overlapping degree between any two overlapping clusters in the network. Assume that u, v are any two cluster head (CH) nodes. Then the overlapping degree between the two corresponding clusters (\mathbf{O}) is a discrete random variable where $\mathbf{O} =$

$|N_k[u] \cap N_k[v]|$ and $N_k[u] \cap N_k[v] \neq \emptyset$. Notice that the overlapping degree is defined only for overlapping clusters (i.e. the random variable \mathbf{O} can not take the value 0). We define AOD as the mean of this random variable \mathbf{O} (i.e. $AOD = E(\mathbf{O})$).

3. *The Connectivity Ratio (CR)*: this metric tests if the generated clusters satisfy the *connectivity condition* as defined in section 2.3. Let S be the set of CH nodes. Let G_S be the undirected graph induced by S such that an edge exists between two nodes $u, v \in S$ if $dist_G(u, v) < 2k$ (i.e the two corresponding clusters overlap). Notice that G_S is not necessary a connected graph. Then the connectivity ratio (CR) is defined as ratio between the number of nodes in the largest spanning tree of G_S to the number of CH nodes ($|S|$). If $CR = 1$, this means that G_S is a connected graph.
4. *The Average Cluster Size (N_c)*: the average number of nodes per cluster taken overall clusters. If u is a CH node, then $N_c = |N_k[u]|$. We use this metric to show that OK generates equal-sized clusters, which is a desirable property to balance the load of control overhead between cluster head nodes.
5. *The Average Number of Edges per Cluster (E_c)*: the average number of edges per cluster taken over all clusters. This metric is important for localization applications [62, 61] since the number of edges in the graph affect the accuracy of the estimated node positions.
6. *The Average CLIQUE Factor per Cluster (CF)*: the CLIQUE factor of a cluster measures how close the subgraph induced by cluster to a complete graph. The CF is calculated as follows:

$$CF = \frac{2 * E_c}{N_c * (N_c - 1)} \quad (6)$$

7. *Communication Overhead*: this metric measures the total energy spent in communication. Without loss of generality, it is assumed that the cost of transmitting 1 unit of data (byte) is 1 unit of energy. This is a valid assumption since we assume that all the nodes have a fixed transmission range.

The first three performance metrics measure how close is OK to meet the conditions listed in section 2.3. N_c , E_c , and CF give more insight into the size of each cluster. Finally,

measuring the communication overhead shows how scalable the proposed algorithm is in terms of messages exchanged between nodes. For simplicity, we assume that the communication environment is contention-free and error-free; hence, sensors do not have to retransmit any data. The Multiple Access with Collision Avoidance (MACA) protocol [51] may be used to allow asynchronous communication while avoiding collisions. MACA utilizes a Request To Send/Clear To Send (RTS/CTS) handshaking to avoid collision between nodes. Other MAC protocols such as TDMA [27] may be used to provide collision-free MAC layer communication.

Our main goals behind the simulation experiments are: (1) to show that with the careful selection of input parameters (p, k, d), the proposed clustering algorithm meets the conditions listed in section 2.3 with high probability; (2) to show that although we have overlapped clusters, the OK clustering still produces approximately equal-sized clusters; (3) to show that OK is scalable in terms of communication overhead. Since each of the above protocol parameters has a different effect on one of the performance metrics, we wanted to give a sensor network engineer a set of parameters to tune to achieve different design goals (minimize power consumption by playing with node transmission range, increase overlapping degree, reduce cluster size, increase inter-cluster connectivity, reduce number of clusters, reduce cluster formation time). In order to qualify the impact of the various parameters, we will try answering the following questions:

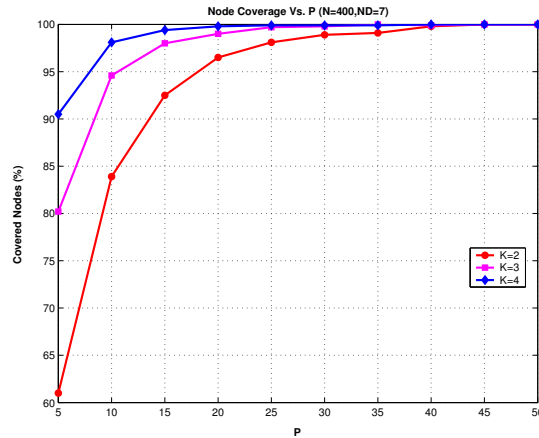
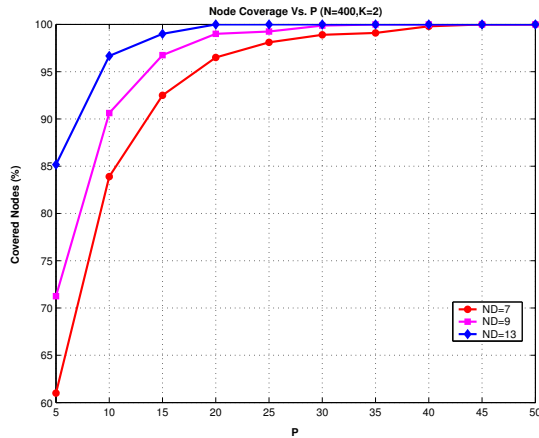
- $Q1$: What is the effect of different simulation parameters (k, d, p) on the percentage of covered nodes (CN) (section 4.1)?
- $Q2$: What is the effect of different simulation parameters (k, d, p) on the average overlapping degree (section 4.1)?
- $Q3$: What is the effect of different simulation parameters (k, d, p) on the connectivity ratio (section 4.1)?
- $Q4$: What is the effect of different simulation parameters (k, d, p) on N_c, E_c and CF (section 4.2)?

- Q5: What is the total energy spent in communication until the clustering protocol terminates (section 4.3)?
- Q6: Is the OK protocol scalable (section 4.3)?
- Q7: Given (n, k, d) , what are the best protocol parameters that guarantee that all the conditions discussed in section 2.3 are satisfied with high probability?

4.1 Coverage, Cluster Overlapping and Connectivity Ratio

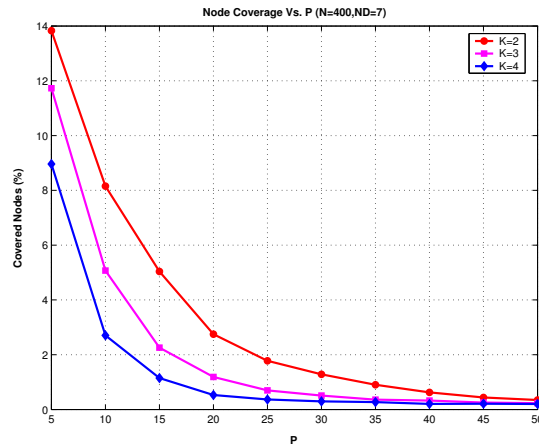
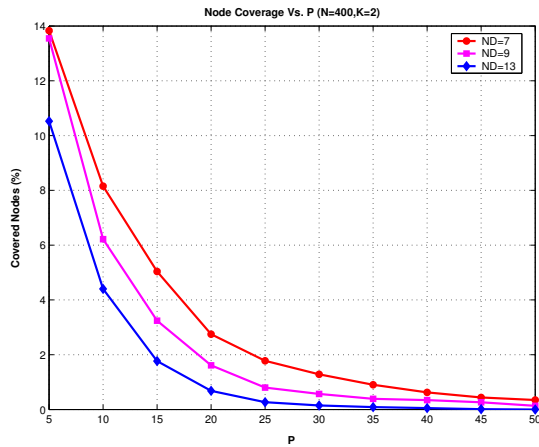
We start by studying the effect of cluster head probability (p) on the percentage of covered nodes (CN). From Fig. 5, we can see that increasing p increases the coverage almost exponentially specially for lower values of d (i.e. low transmission range). The standard deviation curves (Fig. 5(c) and 5(d)) show that the coverage is guaranteed within 2% for $p \geq 0.25$. It is also clear that for each combination of (k, d) , there is a minimum value for p that guarantees 100% coverage with high probability. We will discuss this in more details in section 5.4.

The impact of average node degree (d) on the percentage of covered nodes is shown in Fig. 6(a). Increasing d increases the coverage almost exponentially for lower values of k . For $k > 1$, increasing d above a certain threshold has almost no effect on the coverage. The standard deviation curve (Fig. 6(c)) shows that this is guaranteed within 1% with high probability for $d \geq 16$ and $k > 1$. In Fig. 6(b), the relation between cluster radius (k) and percentage of covered nodes is shown. Increasing k seems to increase the coverage exponentially. Again we can see from the standard deviation curve in Fig. 6(d) that the results are within 1% if $k > 2$ and $d \geq 9$. These values for k and d are very common and realistic in sensor networks applications. As a summary, the effect of increasing d , k is the same. However, d is directly proportional to transmission range; hence it affects node energy dramatically. On the other hand, k is application dependent. For example, in routing protocols, increasing k will increase cluster size, and latency; in localization applications, increasing k will reduce the accuracy of the estimated node position. We will see later in section 4.3, that both k and d increase communication overhead. However, the communication overhead is proportional to k^3 as we will discuss in section 4.3. Finally,



(a) The impact of cluster head prob. (p) on the percentage of covered nodes (different d)

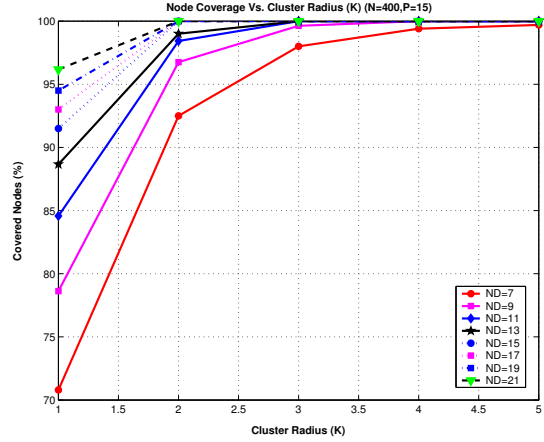
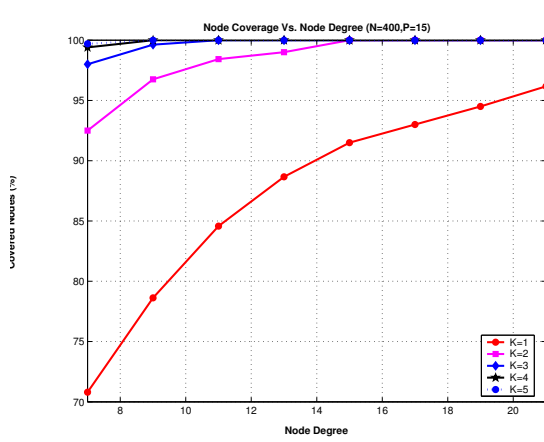
(b) The impact of cluster head prob. (p) on the percentage of covered nodes (different k)



(c) The standard deviation of percentage of covered nodes as (p) increases (different d)

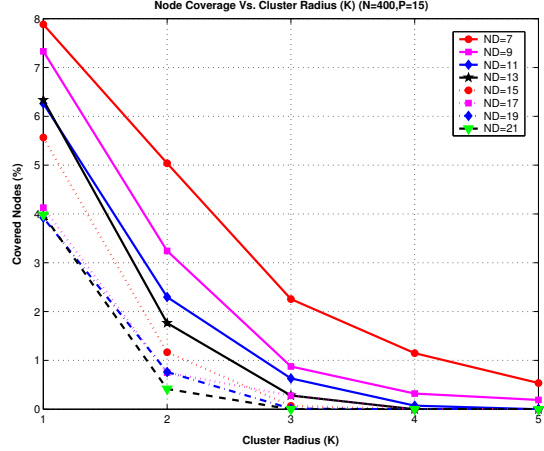
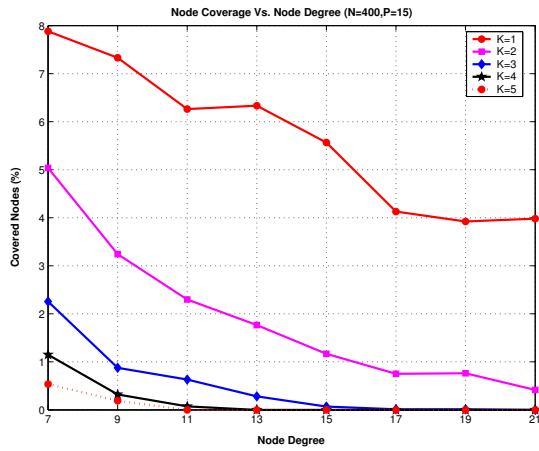
(d) The standard deviation of percentage of covered nodes as (p) increases (different k)

Figure 5: The relation between cluster head prob. (p) and percentage of covered nodes



(a) The impact of average node degree (d) on the percentage of covered nodes

(b) The effect of cluster radius (k) on the percentage of covered nodes



(c) The standard deviation of percentage of covered nodes as d increases

(d) The standard deviation of percentage of covered nodes as k increases

Figure 6: The impact of average node degree (d) and cluster radius (k) on percentage of covered nodes

from the figures we can see that by careful selection of the parameters (p, d, k) we can guarantee 100% coverage with high probability. This means that each node is either a cluster head or belongs to at least one cluster (i.e. the *coverage condition* discussed in section 2.3 is satisfied with high probability).

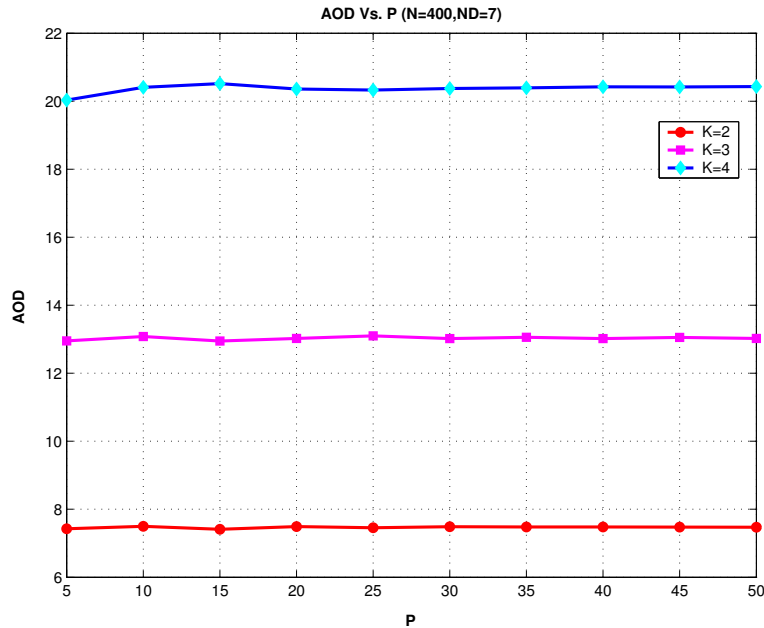
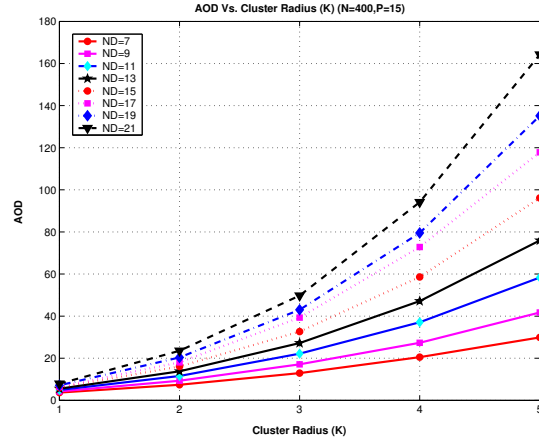
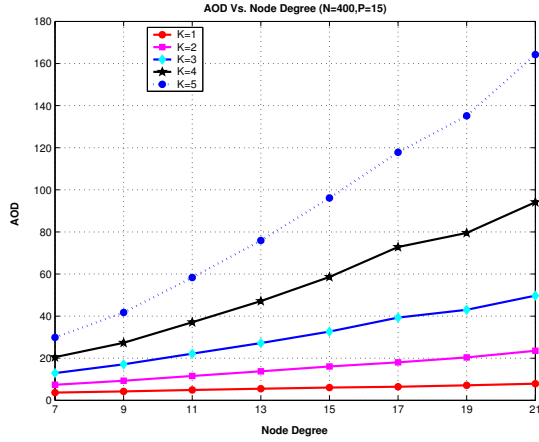


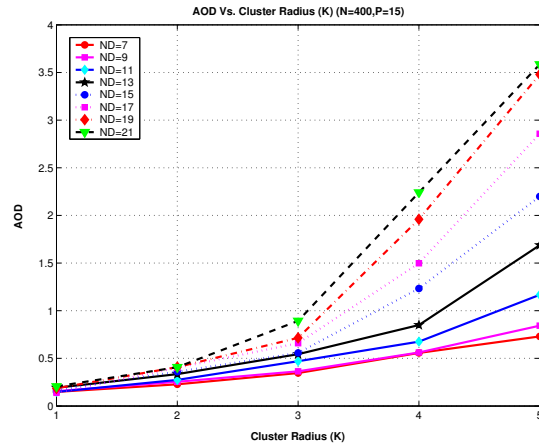
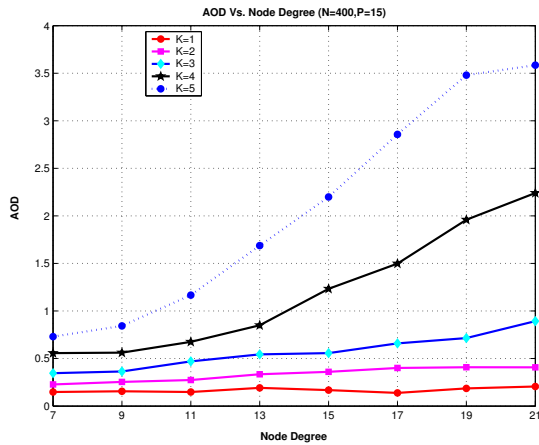
Figure 7: The cluster head prob. (p) has no effect on the average overlapping degree (AOD)

We will now turn our discussion to the study of the average overlapping degree between clusters. Fig. 7 shows an interesting anomaly. Although one may think that increasing p (i.e. increasing number of cluster heads and hence clusters) should increase the average overlapping degree (AOD), the results showed that p has no effect on AOD regardless of the values of other parameters (d, k) and network size (n). We will prove analytically in section 5.3 that the AOD does not depend on p . This will leave us with only two parameters to play with to control the overlapping between clusters d , and k . As shown in Fig. 8(a), the AOD is linearly proportional with d . Notice that AOD can never exceed the network size n so the curve saturates at n . On the other hand, increasing the cluster radius (k) will increase the AOD quadratically as shown in Fig. 8(b). We will discuss analytically in more details the relation between AOD and d and k in section 5.3. Notice that for many applications, the required AOD between clusters should be below 10. For example in local-



(a) The impact of average node degree (d) on the average overlapping degree (AOD)

(b) The effect of cluster radius (k) on the average overlapping degree (AOD)



(c) The standard deviation of average overlapping degree (AOD) as d increases

(d) The standard deviation of average overlapping degree (AOD) as k increases

Figure 8: The impact of average node degree (d) and cluster radius (k) on average overlapping degree

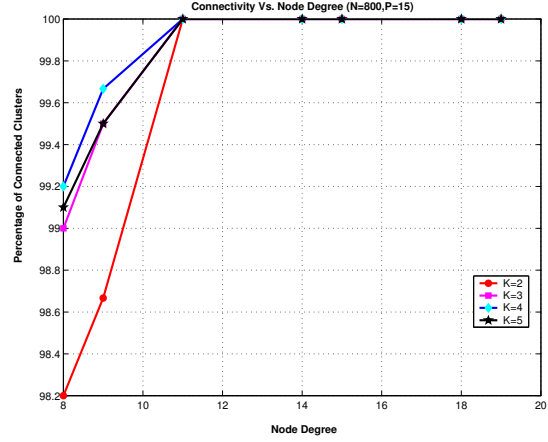
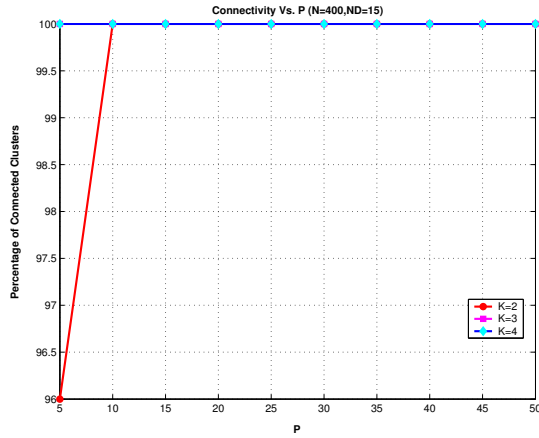
ization, an AOD of 3 is enough [61, 62] and in routing protocols having 10 gateway nodes between clusters is more than enough. It is clear that we can guarantee an AOD of more than 10 with high probability using small d (i.e. low transmission range) and small cluster radius ($k = 2$). This is confirmed also by the standard deviation curves, Fig. 8(c) and 8(d). We can clearly see from the curves that an AOD of at least 10 can be guaranteed with high probability if $k \geq 2$ and any $d > 6$.

Finally, to show that the OK protocol satisfies the *connectivity condition*, as defined in section 2.3, we study the connectivity between clusters. Fig. 9(a) shows the relation between connectivity ratio and p for different values of k . The figures show that with 15% of the nodes are cluster heads; we can have 100% connectivity with high probability. This means that for any cluster head, there is a path of less than $2k$ hops to at least another cluster head (i.e. there is at least one border node between the two clusters). We can see that this still holds for any value of k and $d > 10$ as shown in Fig. 9(b). the standard deviation curves (Fig. 9(c) and 9(d)) confirm the above results with high probability.

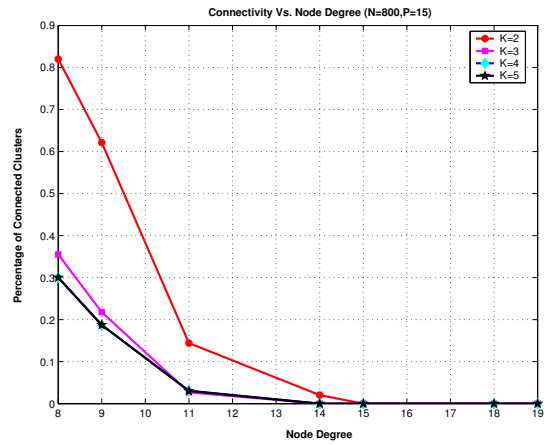
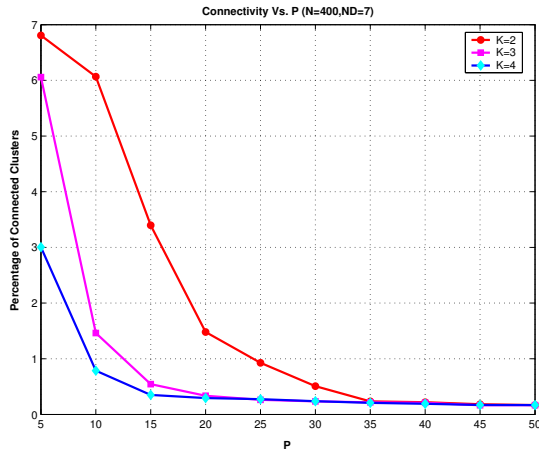
As a general conclusion, it is clear that the OK protocol satisfies with high probability the three conditions, defined in section 2.3. The cluster head probability (p) plays an important role in terms of coverage and connectivity between cluster. The average node degree (d) and the cluster radius (k) can be tuned to achieve a reasonable average overlapping degree between clusters regardless of p .

4.2 Cluster Size

In this section we will study the properties of the generated clusters in terms of average cluster size (N_c), average number of edges per cluster (E_c) and average CLIQUE factor (CF). Since the clusters are overlapping, increasing the number of clusters will not affect the cluster size. Hence, p has no effect on N_c , E_c and CF as shown in Fig. 10. On the other hand, increasing d increases N_c linearly, as shown in Fig. 11(a), and increases E_c quadratically (Fig. 11(b)). Substituting in Eq. 6, we can see why the CF is almost constant as d increases (Fig. 11(c)). A detailed analytical model for the average cluster size is discussed in section 5.2.

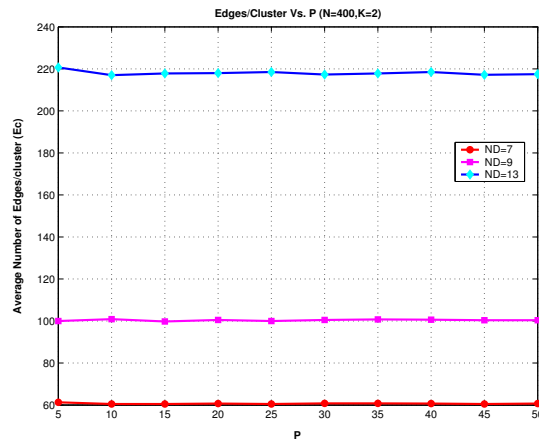
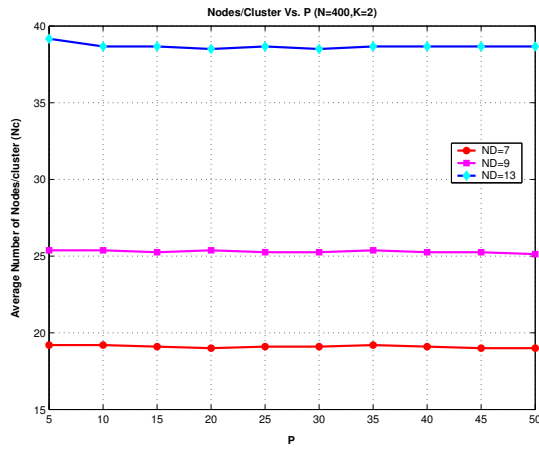


(a) The relation between connectivity ratio (CR) and the cluster head prob. (p) (b) The relation between connectivity ratio (CR) and average node degree (d)



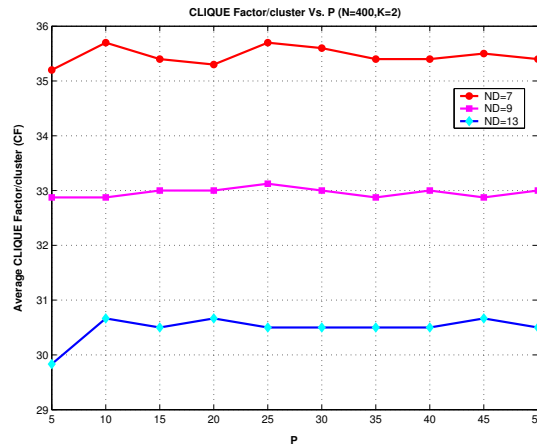
(c) The standard deviation of percentage of connected clusters as p increases (d) The standard deviation of percentage of connected clusters as d increases

Figure 9: The effect of the cluster head prob. (p) and average node degree (d) on percentage of connected clusters



(a) The cluster head prob. (p) has no effect on the average number of nodes/cluster

(b) The cluster head prob. (p) has no effect on the average number of edges/cluster



(c) The cluster head prob. (p) has no effect on the average CLIQUE factor (CF)

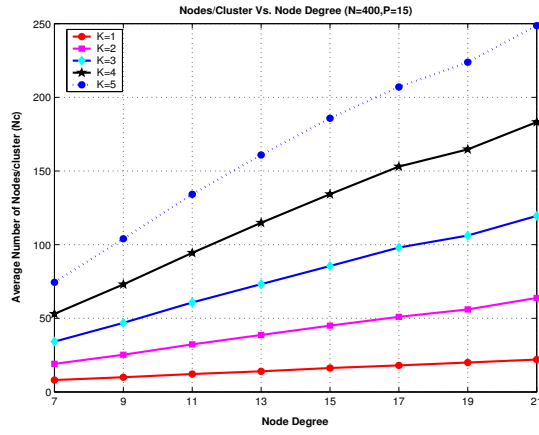
Figure 10: The cluster head prob. (p) has no effect on cluster size properties

As a measure of load balancing, the standard deviation of average number of nodes per cluster is shown in Fig. 11(d). The figure shows very low standard deviation regardless of the values of d and k . This means that the OK protocol produces equal-sized clusters. The same facts can be concluded from the standard deviation curves of number of edges per cluster (Fig. 11(e)) and the average CLIQUE factor (Fig. 11(f)). From Fig. 12(a) and 12(b), we can see that both N_c and E_c are proportional with the square of the cluster radius (k^2). Hence, from Eq. 6, we can see why the average CLIQUE factor (CF) decreases quadratically as k increases (Fig. 12(c)). Again the standard deviation curves, Fig. 8 12(d), 12(e), 12(f), confirm that OK produces equal-sized clusters regardless of the values of d and k .

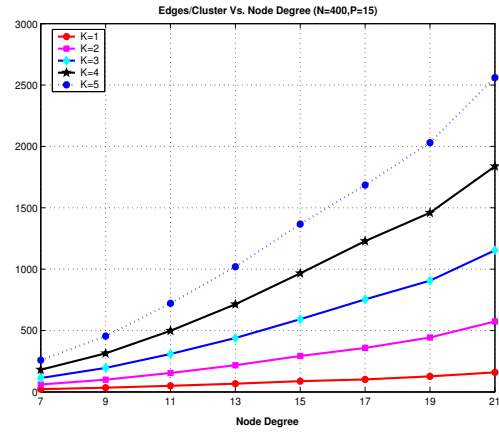
As a final conclusion, although the OK protocol generates overlapping clusters, the simulation results show that those clusters are equal-sized. Equal-sized clusters is a desirable property because it enables an even distribution of control (e.g., data processing, aggregation, storage load) over cluster heads; no cluster head is overburdened or underutilized. Moreover, the results show that the average cluster size can be controlled by tuning the average node degree (d) or the cluster radius k . A closed form for the upper bound of the average cluster size (N_c) as a function of d and k is given in section 5.2. Finally, the average number of edges and the intra-cluster connectivity, measured by the CF metric, can also be controlled by changing d and k . This is a desirable feature in anchor-free localization applications [62, 61, 53, 43] where the number of edges in the cluster determines the accuracy of the estimated node positions.

4.3 Scalability

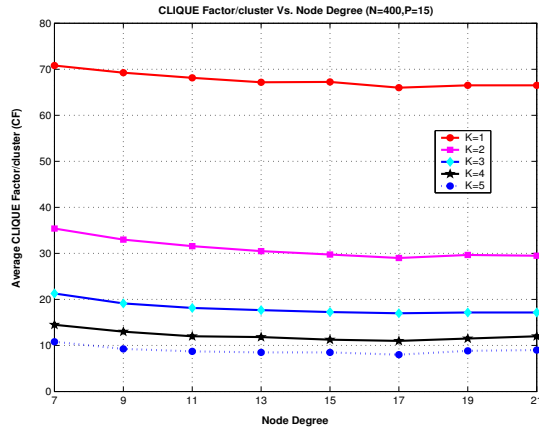
In this section we analyze the communication overhead of the OK clustering protocol and show that OK is scalable and energy efficient in terms of communication overhead. The total energy spent in communication is measured in terms of the number of bytes transmitted per node. Without loss of generality, it is assumed that the cost of transmitting 1 unit of data (byte) is 1 unit of energy. This is a valid assumption since we assume that all the nodes have a fixed transmission range. We will start by describing the model used for estimating



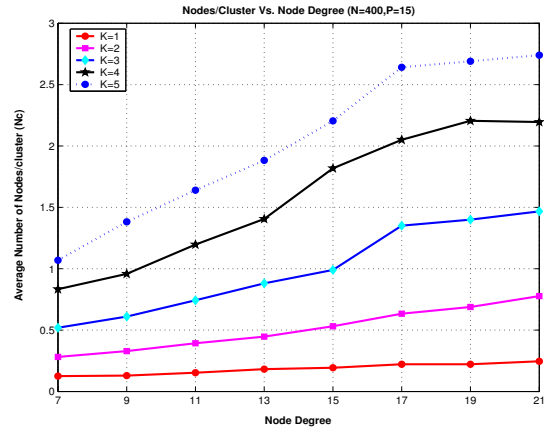
(a) The effect of average node degree (d) on number of nodes/cluster (N_c)



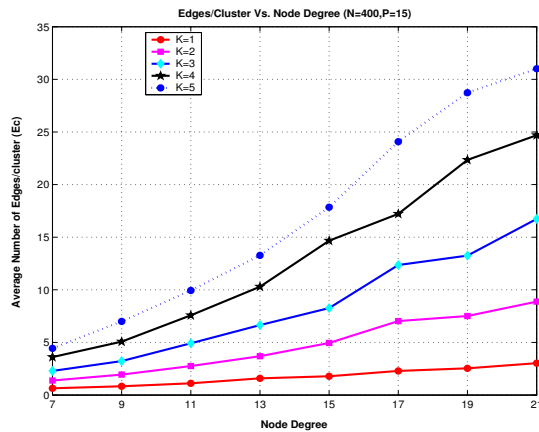
(b) The effect of average node degree (d) on number of edges/cluster (E_c)



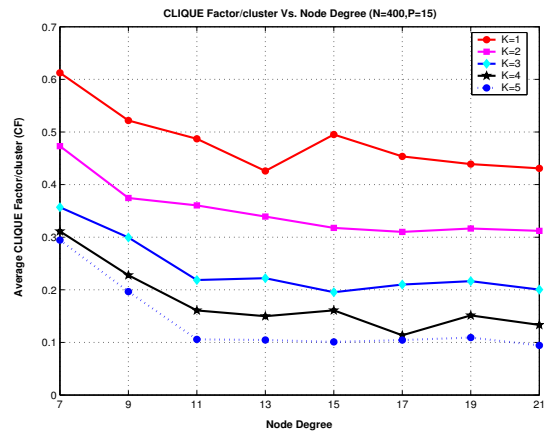
(c) The effect of average node degree (d) on the CLIQUE factor (CF)



(d) The standard deviation of the average number of nodes/cluster (N_c) as d increases

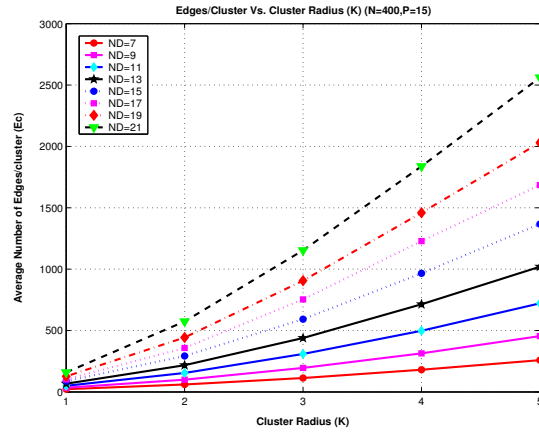
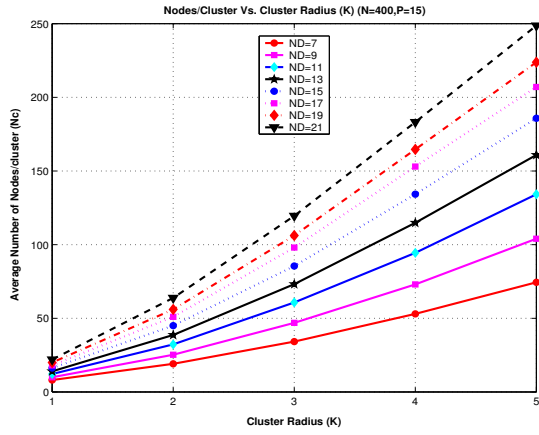


(e) The standard deviation of the average number of edges/cluster (E_c) as d increases



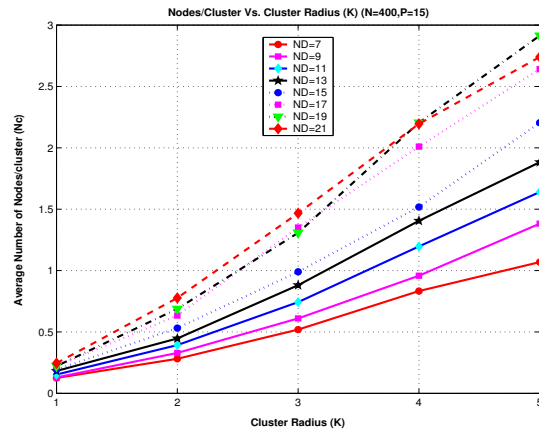
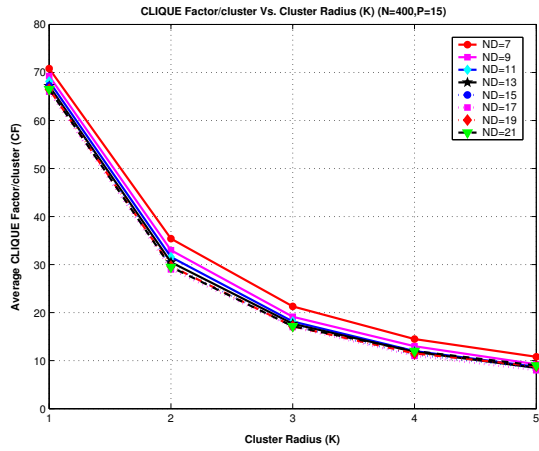
(f) The standard deviation of the average CLIQUE factor (CF) as d increases

Figure 11: The effect of average node degree on cluster size properties



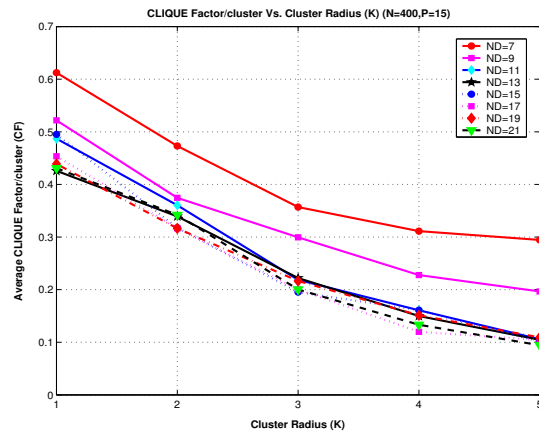
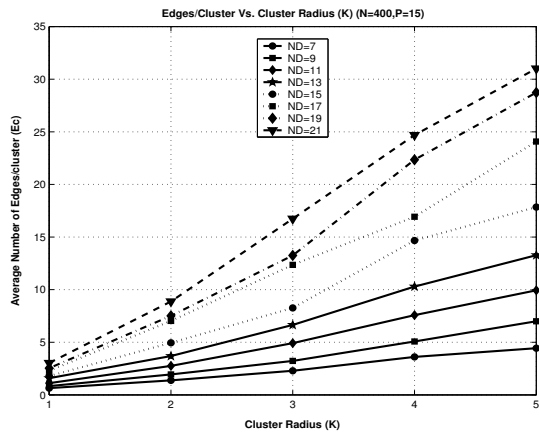
(a) The effect of cluster radius (k) on number of nodes/cluster (N_c)

(b) The effect of cluster radius (k) on number of edges/cluster (E_c)



(c) The effect of cluster radius (k) on the CLIQUE factor (CF)

(d) The standard deviation of the average number of nodes/cluster (N_c) as k increases



(e) The standard deviation of the average number of edges/cluster (E_c) as k increases

(f) The standard deviation of the average CLIQUE factor (CF) as k increases

Figure 12: The effect of cluster radius on cluster size properties

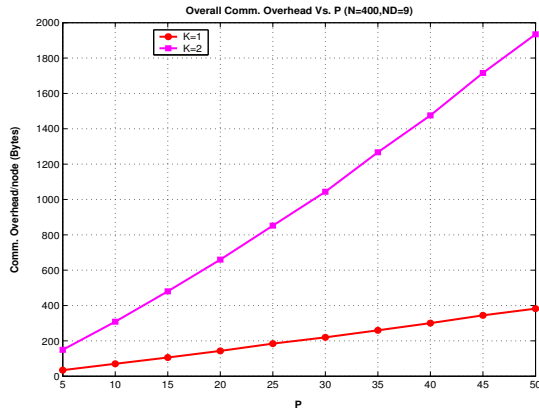
the communication overhead. Then we show the impact of different simulation parameters on the overall communication overhead and study the scalability of the OK protocol. An analytical model for the communication overhead is discussed in the next section (5.4).

There are two phases in the OK protocol: the cluster head advertisement phase (CHAD phase) and the join request phase (JREQ phase). For each network topology i with network size n , we calculate the total number of bytes sent by all the nodes during the two phases ($TotalMsgSize(i)$). We then repeat the experiment over 900 different topologies, with the same network size n . Hence, the average number of bytes sent by all nodes ($avgTotalMsgSize$) is the mean of the vector $TotalMsgSize(i)$ for $i= 1..900$. Finally, we divide $avgTotalMsgSize$ by the network size (n) in order to get the average number of bytes sent by one node $avgCommOverhead$. We use the last metric to measure the average energy spent by a node in communication.

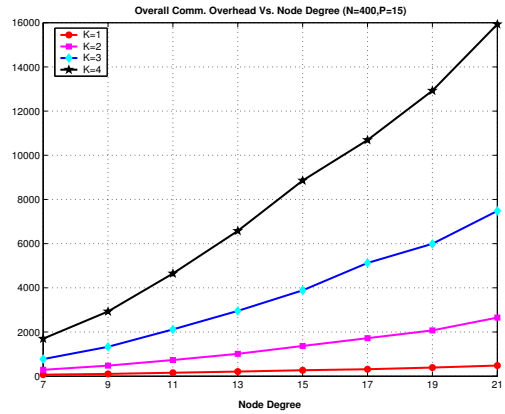
Fig. 13 shows the impact of different simulation parameters on communication overhead. The effect of increasing cluster head probability (p) is shown in Fig. 13(a). We observe that the communication energy increase linearly as p increases. We can also notice that the rate increases significantly as the cluster radius (k) increases. This is can be clearly seen in Fig. 13(c) where it can be shown that the communication overhead is cubically proportional to the cluster radius (k). Mainly this cost is incurred during the JREQ phase as we will show analytically in section 5.4.

The effect of average node degree (d) on communication overhead is shown in Fig. 13(b). We can notice that the communication overhead increases linearly as d increases. Although, we will discuss the relation between communication overhead and average node degree (d) analytically in section 5.4, we can intuitively explain that by analyzing the relation between average number of nodes per cluster N_c and average node degree (d) (Fig. 11(a)). As the average node degree increases, the average number of nodes per cluster increases linearly and hence the average number of JREQ messages increases linearly leading to a linear increase in the overall communication overhead.

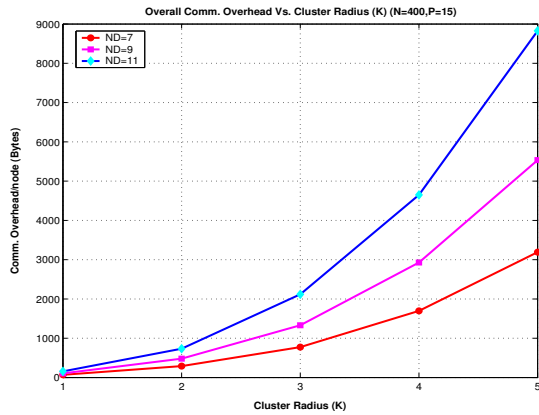
Finally, we will show that OK is scalable in terms of processing time in section 6, (*lemma* 6.2). However, in this section, we study the scalability in terms of communication overhead. We tested the OK protocol for different network size ranging from 50 to 800



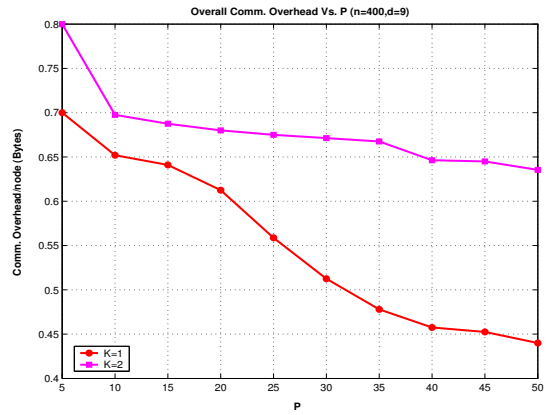
(a) The relation between communication overhead and the cluster head prob. (p)



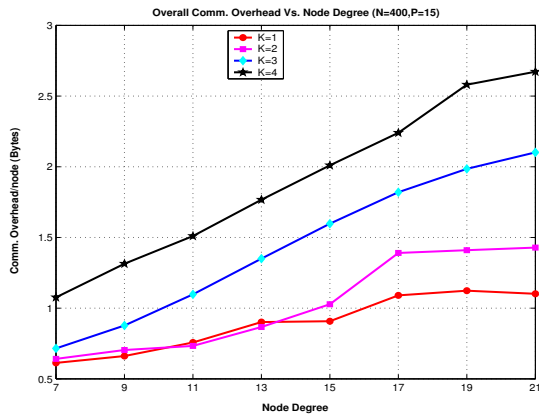
(b) The relation between communication overhead and the average node degree (d)



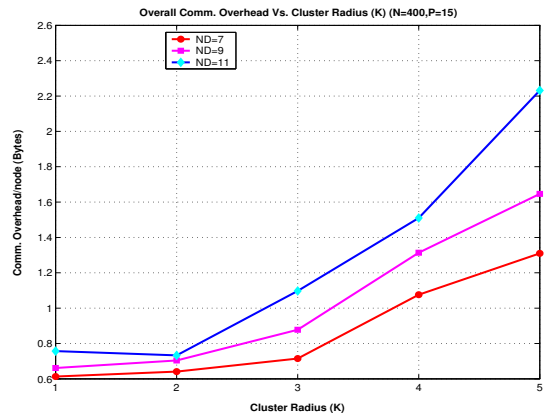
(c) The relation between communication overhead and the cluster radius (k)



(d) The standard deviation of the communication overhead as the cluster head prob. (p) increases



(e) The standard deviation of the communication overhead as the average node degree (d) increases



(f) The standard deviation of the communication overhead as the cluster radius (k) increases

Figure 13: The effect of different simulation parameters on communication overhead per node

nodes. Fig. 14 shows the overall communication overhead per node as network size increases. We can clearly see that the number of bytes transmitted by a node slowly increases as the network size increases from 100 to 400. Then it remains almost constant afterwards. The standard deviation curves (Fig. 14(b)) show that this happens with high probability (± 2 bytes).

5 Analysis of the Results

In this section, using unit disk graph properties, and simple geometry, we will analytically show the following:

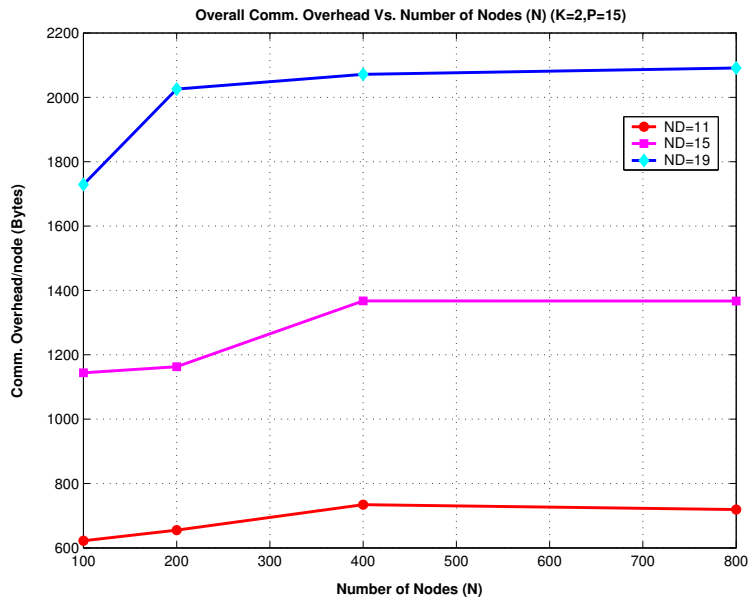
- The average number of nodes per cluster (N_c) is linear in d and quadratic in k (section 5.2).
- The average number of edges per cluster (E_c) is quadratic in both d and k (section 5.2).
- The cluster head probability (p) does not affect the average overlapping degree (AOD) between clusters (section 5.3).
- The average overlapping degree (AOD) is linearly proportional to the average node degree (d) and quadratically proportional to the cluster radius (k) (section 5.3).
- The overall communication overhead is linearly proportional with d and cubically proportional with k (section 5.4).

We will start by describing the assumptions behind the proposed analytical model.

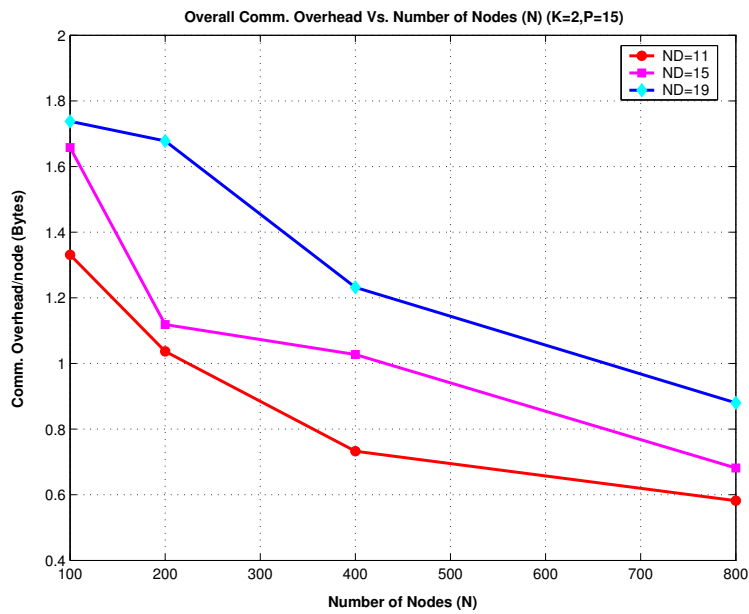
5.1 Assumptions

In order to simplify the proofs, we make the following assumptions:

- Each cluster can be approximated ideally by a circle of radius R .



(a) The impact of network size (n) on the communication overhead incurred per node



(b) The standard deviation of the communication overhead as the network size n increases

Figure 14: Increasing the network size n does not effect the communication overhead

- Since the transmission range of each node is fixed (T_r), and since only nodes that are within k hops from the cluster head can belong to this cluster, then we can approximate R as follows:

$$R = kT_r \quad (7)$$

In this case, R is considered the maximum euclidian distance that a node can be away from cluster head. Hence, the circle representing the cluster is considered the largest area that can be covered by a cluster.

- The cluster head is located at the center of this circle.

The above geometric representation of a k -hop cluster is considered the largest possible area for the cluster. This will lead to considering some areas as belonging to the cluster when they are not. We will refer to such an area as a *false area*. For example, if the cluster head node is located within distance R from the boundary of the sensor field, the circle representing the cluster will be clipped by the rectangle representing the field as shown in Fig. 15. Hence, the area which lies outside the sensor field is a *false area* since it is considered within the cluster but it does not really belong to it. Since the proposed analytical model represents an upper bound, *false area* will just make the upper bound not tight enough when compared with the simulation results. We will also show in the following that as the number of cluster heads increases, either because the network size (n) increases or the cluster head probability (p) increases, the probability of having cluster heads within distance R decreases; hence the effect of *false area* decreases. So by carefully selecting the simulation parameters, we can safely ignore the effect of *false area*.

Recalling that the average number of cluster heads is pn , and assuming a square field with side length l , then the probability (P_{IN}) that a cluster head node is *at least* at distance R away from the boundary of the field (i.e. inside the dotted rectangle as shown in Fig. 15), is given by the following:

$$P_{IN} = \frac{(l - 2R)^2}{l^2} \quad (8)$$

Then the probability that a cluster head node is within distance R from the boundary will be (P_{OUT}):

$$P_{OUT} = 1 - P_{IN}$$

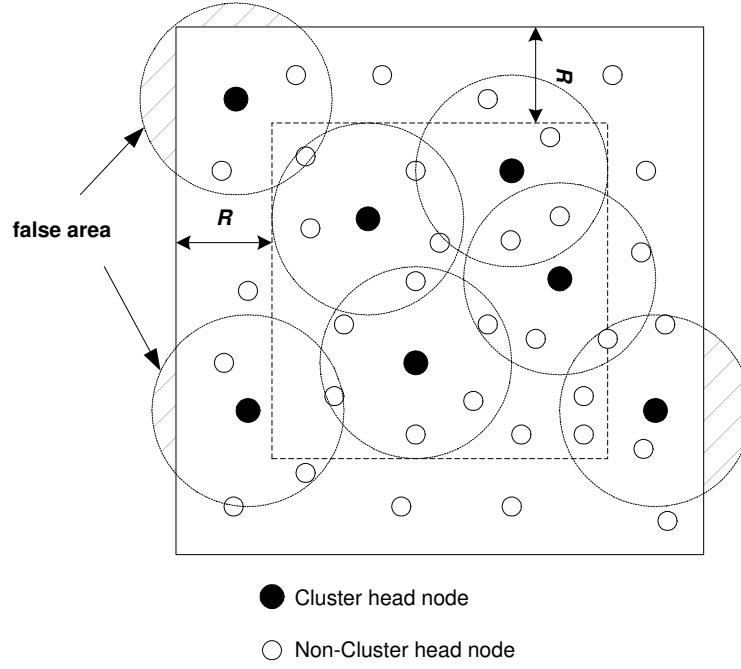


Figure 15: Circle representation of clusters

Let I be a discrete random variable representing the number of cluster heads that are within distance R from the boundary of the field. Then I can be expressed as a binomial distribution:

$$P(I = m|pn) = P_{OUT}^m P_{IN}^{pn-m} \binom{pn}{m}$$

and the expected number of cluster head nodes that are within distance R from the field boundary is:

$$E(I) = pnP_{OUT}$$

In order to ignore the effect of cluster heads that are near the boundary; hence decreasing the size of *false area*:

$$\begin{aligned} P_{OUT} &\ll P_{IN} \\ \frac{l^2}{(l-2R)^2} &\ll 2 \\ 4R^2 - 8lR + l^2 &\gg 0 \end{aligned}$$

Solving the quadratic equations in R , and substituting $R = kT_r$, one of the following conditions must hold:

$$T_r \ll \frac{(1 - \sqrt{3}/2)l}{k} \quad (9)$$

OR

$$T_r \gg \frac{(1 + \sqrt{3}/2)l}{k} \quad (10)$$

The first condition was used in the simulations since it implies reducing the node transmission range; hence reducing energy consumption. However, we must be careful in reducing the node transmission range (T_r) since there is a minimum critical value for T_r in order for the graph to be connected [46]. Moreover, as k increases, it becomes difficult to satisfy the first condition while guaranteeing connectivity. Since our main goal is to have a connected graph, we have to violate the first condition as k increases. In a similar way, we notice that in order to increase the average node degree (d), we have to increase T_r ; hence, we may violate the first condition to achieve a certain average node degree. That's why we will notice that the analytical model diverges a little bit from the simulation results as k or d increases since the effect of cluster heads near the boundary starts increasing.

5.2 Average Cluster Size

We shall start by estimating an upper bound of the average cluster size (average number of nodes per cluster). The cluster will be represented by a circle with radius $R = kT_r$ as discussed in section 5.1. Assume that N_c is a discrete random variable representing the cluster size. Then using the same analysis as we did in the previous section, we can show that N_c can be expressed by the following binomial distribution:

$$P(N_c = m) = P_c^m (1 - P_c)^{n-m} \binom{n}{m} \quad (11)$$

where n is the network size and P_c is the probability that a node is inside the circle representing the cluster

$$P_c = \frac{\pi R^2}{l^2} = \frac{\pi k^2 T_r^2}{l^2} \quad (12)$$

where l is the side length of the square field. Now substituting from equation 5, in order to get P_c in terms of average node degree (d), we get

$$P_c = \frac{dk^2}{n} \quad (13)$$

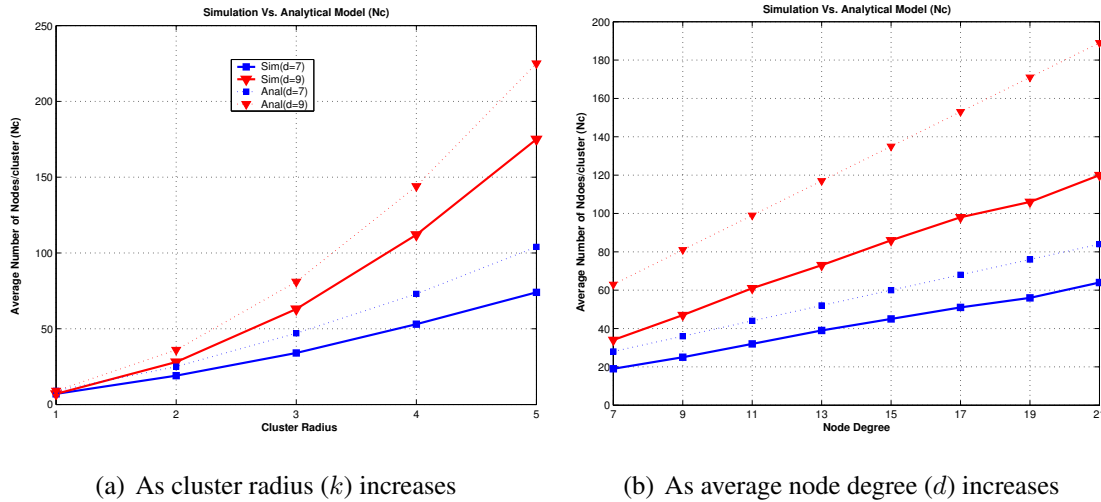


Figure 16: The relation between the analytical model for average cluster size (N_c) and simulation results

Hence the average cluster size ($E(N_c)$) is:

$$E(N_c) = nP_c = dk^2 \quad (14)$$

The above equation shows that the average cluster size is linearly proportional with average node degree (d) and quadratically proportional to the cluster radius (k). This conforms with the simulation results shown in section 5.2. Moreover, we can see that N_c is not a function of the cluster head probability (p). Fig. 16 shows the relation between the simulation results and analytical model given by equation 14.

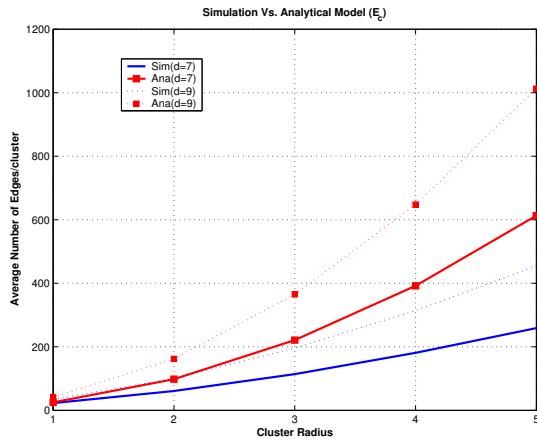
Using the above model, we can estimate the average number of edges per cluster (E_c) as follows. Since each node has an average node degree (d), and since the average number of nodes per cluster is N_c , then

$$E_c = \frac{dN_c}{2} = \frac{d^2k^2}{2} = O(d^2k^2) \quad (15)$$

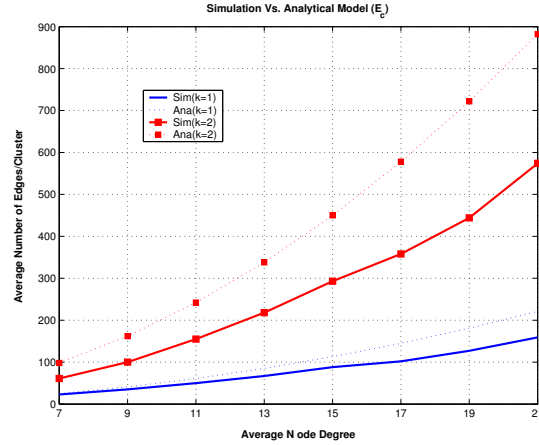
Fig. 17 shows the relation between the simulation results and analytical model given by equation 15.

5.3 Average Overlapping Degree

Using the assumptions in section 5.1, we shall calculate an upper bound for the average overlapping degree (AOD). Assume that A, B are any two cluster head (CH) nodes. Then



(a) As cluster radius (k) increases



(b) As average node degree (d) increases

Figure 17: The relation between the analytical model for average number of edges per cluster (E_c) and simulation results

we recall from section 4 that the overlapping degree between the two corresponding clusters (\mathbf{O}) is a random variable where $\mathbf{O} = |N_k[A] \cap N_k[B]|$ and $N_k[A] \cap N_k[B] \neq \emptyset$. Notice that the overlapping degree is defined only for overlapping clusters (i.e. the random variable \mathbf{O} does not take the value 0). We define AOD as the mean of this random variable \mathbf{O} (i.e. $AOD = E(\mathbf{O})$). As shown in Fig. 18, the two clusters A and B are represented by two

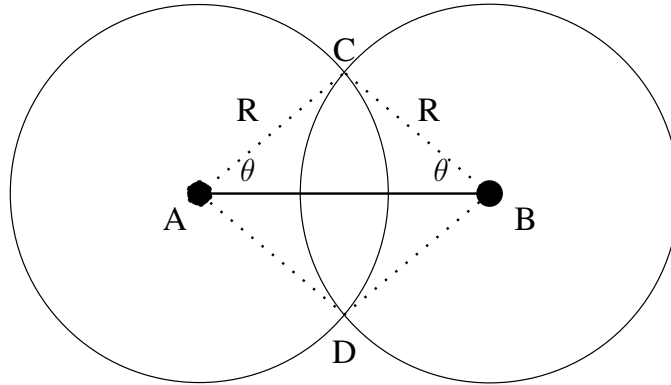


Figure 18: Overlapping Degree (\mathbf{O}) between two overlapping clusters

symmetric circles of radius $R = kT_r$. Instead of calculating the exact intersection of the two sets ($N_k[A] \cap N_k[B]$), we shall estimate the intersection of the two sets by the area of intersection between the two corresponding circles. Let W be the euclidian distance between the two CH nodes. Then, W is a continuous random variable that can take values

ranging from 0 to $2R$. The two clusters are completely overlapped if $W = 0$ and there is no overlapping if the distance between the two cluster heads is greater than or equal $2R$. Let $F(w)$ and $f(w)$ be the CDF and PDF of the random variable W consequently. Then

$$F(w) = P(W < w) = \frac{\pi w^2}{\pi(2R)^2} = \frac{w^2}{4R^2} \quad (16)$$

$$\therefore f(w) = \frac{dF(w)}{dw} = \frac{w}{2R^2} \quad (17)$$

We will express \mathbf{O} as a function of w as follows. The area of intersection between two symmetric circles A and B (I_{AB}) is ⁴:

$$I_{AB} = (2\theta - \sin 2\theta)R^2 = E(\mathbf{O} | w) \quad (18)$$

where $w = 2R \cos \theta$ (using cosine rule). Hence, \mathbf{O} is a continuous random variable that is represented as a function of θ or w alternatively.

$$\therefore E(\mathbf{O}) = \int_0^{2R} E(\mathbf{O} | w) f(w) dw = \int_0^{2R} (2\theta - \sin 2\theta) R^2 f(w) dw \quad (19)$$

Substituting from Eq. 17, 18, and noticing that $dw = 2R \sin \theta$, we have the following:

$$E(\mathbf{O}) = \int_0^{\pi/2} (2\theta - \sin 2\theta) R^2 2 \sin \theta \cos \theta d\theta = R^2 \int_0^{\pi/2} (2\theta - \sin 2\theta) \sin 2\theta d\theta \quad (20)$$

It can be shown that $\int_0^{\pi/2} (2\theta - \sin 2\theta) \sin 2\theta d\theta = \frac{\pi}{4}$. Hence,

$$E(\mathbf{O}) = \frac{\pi R^2}{4} \quad (21)$$

Substituting from Eq. 7, we have:

$$E(\mathbf{O}) = \frac{\pi k^2 T_r^2}{4} \quad (22)$$

Substituting from Eq. 5 to get the relation in terms of average node degree, we reach the following:

$$E(\mathbf{O}) = \frac{dk^2 l^2}{4n} = \frac{dk^2}{4\mu} = \mathbf{AOD} \quad (23)$$

⁴For more details of the proof, please refer to appendix A.1

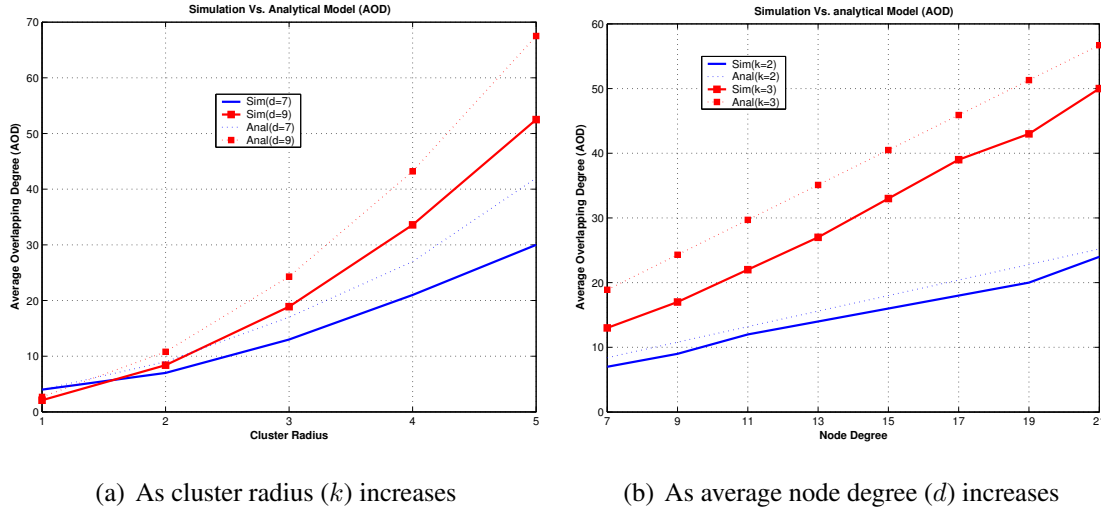


Figure 19: The relation between the analytical model for average overlapping degree (AOD) and simulation results

The above equation shows that the average overlapping degree is linearly proportional with average node degree d and quadratically proportional to the cluster radius k . This conforms with the simulation results shown in section 5.3. We can also notice that AOD is not a function of cluster head probability p . Fig. 19 shows the relation between the simulation results and analytical model given by equation 23.

5.4 Overall Communication Overhead

In this section, we will calculate an upper bound of the average number of control messages transmitted by a node. As we did in the previous sections, the cluster will be approximated by a circle with radius $R = kT_r$. Recall that there are two phases in the OK protocol: the cluster head advertisement phase (CHAD phase) and the join request phase (JREQ phase). We will estimate the number of messages sent during each phase per node. Then the overall communication overhead per node will be the total number of messages. We will start by estimating the average number of nodes that are exactly k hops away from the cluster head (n_k). From equation 14, the average number of nodes in k -hop cluster is:

$$E(N_c) = nP_c = dk^2 = E_k(N_c) \quad (24)$$

the graph rooted at the CH node.

$$M_{CHAD} = 1 + \sum_{i=1}^{k-1} n_i \quad (27)$$

where n_i is the expected number of nodes that are exactly i hops way from the CH node (Eq. 26). Substituting from Eq. 26 and simplifying the expression, we reach the following:

$$M_{CHAD} = 1 + \frac{2d(k-1)^2}{2} = O(dk^2) \quad (28)$$

Using a similar approach, we can calculate the average number of JREQ messages (M_{JREQ}) *unicasted* from non-CH nodes to the CH node. We assume that we do not do any aggregation of the messages⁵; hence; a JREQ message, unicasted from a leaf node in the spanning tree, will be forwarded k times till it reach the CH node. Therefore, M_{JREQ} can be calculated as follows:

$$M_{JREQ} = kn_k + (k-1)n_{k-1} + \dots + 2n_2 + n_1 = \sum_{i=1}^k in_i \quad (29)$$

Substituting from Eq. 26 and simplifying the expression, we reach the following expression:

$$M_{JREQ} = \frac{dk(4k-1)(k+1)}{6} = O(dk^3) \quad (30)$$

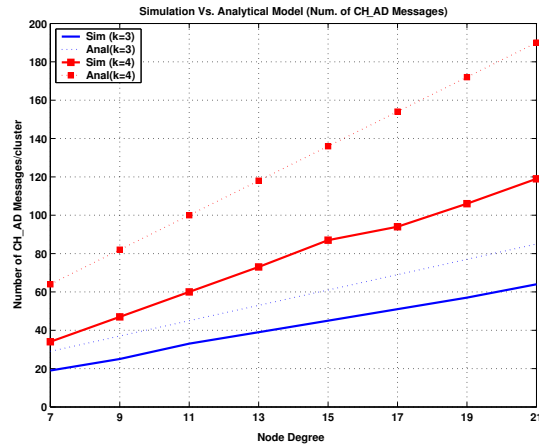
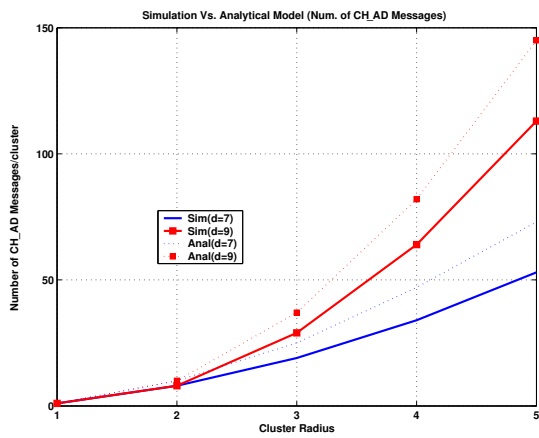
Fig. 21 shows the relation between the simulation results and analytical model of the communication overhead.

6 Correctness and Complexity

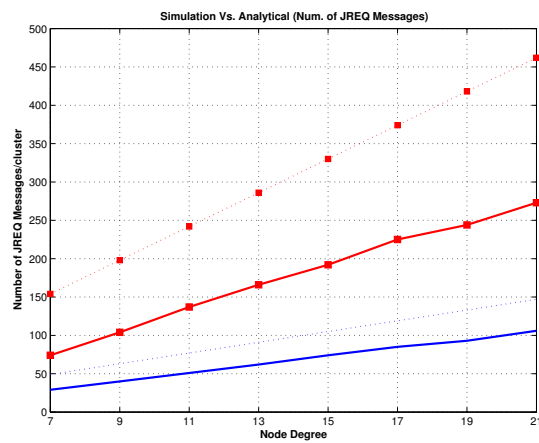
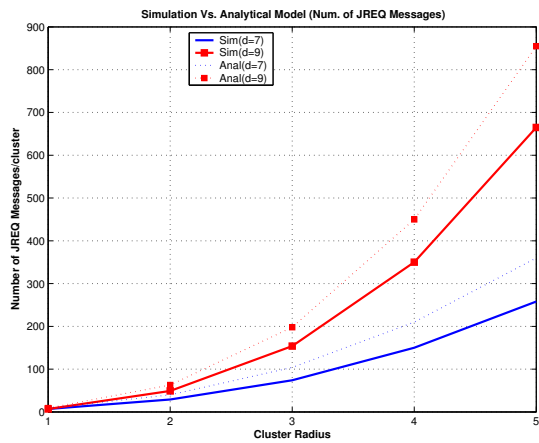
In this section we shall discuss that the OK protocol provided in Fig. 4 meets the following design goals (requirements):

1. Completely distributed.
2. Terminates within $O(k)$ iterations, regardless of network diameter, where k is the cluster radius.

⁵Of course, if message aggregation is used, the overall communication overhead will improve. So the above analysis is considered a worst case analysis.



(a) Average number of CH_AD messages per cluster as cluster radius (k) increases (b) Average number of CH_AD messages per cluster as average node degree (d) increases



(c) Average number of JREQ messages per cluster as cluster radius (k) increases (d) Average number of JREQ messages per cluster as average node degree (d) increases

Figure 21: The relation between the analytical model for overall communication overhead per node and simulation results

3. At the end of the algorithm, each node is either a cluster head, or non-cluster head node that belongs to one or more clusters.
4. Efficient in terms of memory usage.

Observation 1. OK is completely distributed (requirement 1). A node can either elect to become a cluster head, or join a cluster if it receives CH_AD messages within its cluster radius. Thus, node decisions are based solely on local information.

Lemma 6.1. *The time complexity of OK is $O(k)$ (requirement 2).*

Proof. The worst case scenario is: a non-CH (NCH) node does not receive any CH_AD messages and changes its status to CH. Then broadcasts a CH_AD message and waits for JREQ messages. Recall from section 3.3 that the maximum time that an NCH node waits for a CH_AD message is equal to $t(k) + \delta$, where $t(k)$ is the time needed for a message to travel k hops and δ is a constant value independent from k . Hence, the total time of this worst case scenario is $t(k) + \delta + 2t(k)$. Therefore the maximum time that a node should wait before terminating OK is $t(k) + \delta + 2t(k) = 3t(k) + \delta = O(k)$. \square

Lemma 6.2. *At the end of the OK algorithm, a node is either a cluster head, or non-cluster head node that belongs to one or more clusters (requirement 3).*

Proof. Initially each node is either CH or NCH node. If the node is a CH node, it will terminate the OK algorithm after $2t(k) + \delta$ time units when the JREQ_WAIT timer fires. In case of NCH node, after $t(k) + \delta$ time units, either it joins one or more clusters that it heard from or changes status to CH and terminates the OK algorithm after $2t(k)$ time units. \square

Lemma 6.3. *The expected number of adjacent overlapping clusters is $O(pdk^2)$, where p is the cluster head probability, d is the average node degree, and k is the cluster radius.*

Proof. Recall that the expected number of clusters is np where n is the network size. Let u and v be two cluster head nodes. Then the two corresponding clusters of u and v are overlapping iff $dist_G(u, v) < 2k$. Using the circle approximation of the cluster as discussed

in section 5.1, then the probability (P_{Adj}) that a CH node is within distance $2R$, $R = kT_r$, from m other CH nodes is given by the following binomial distribution:

$$P_{Adj}(m) = P_{2R}^m (1 - P_{2R})^{np-m-1} \binom{np-1}{m}, \text{ where } P_{2R} = \frac{\pi(2R)^2}{l^2} \quad (31)$$

Hence, the expected number (f adjacent clusters is ($E(P_{Adj})$):

$$E(P_{Adj}) = P_{2R}(np-1) \simeq npP_{2R} = \frac{4\pi npR^2}{l^2} \quad (32)$$

Since $R = kT_r$, substituting from equation 5 and simplifying the expression, we get the following:

$$E(P_{Adj}) = 4\pi pdk^2 = O(pdk^2) \quad (33)$$

□

Lemma 6.4. *The OK algorithm has an average memory usage of $O(1)$ per node (requirement 4).*

Proof. The two major data structures used by the OK protocol are: *CH_table* and *AC_table*. Any other data structures will take $O(1)$ memory to store. Recall from section 3.1, *CH_table* is used by each node, whether CH or NCH, to store information about the known CH nodes. Hence, the average size of the *CH_table* is equal to the expected number of clusters that cover a certain node; which is equal to the expected number of adjacent clusters ($E(P_{Adj})$). Therefore, using lemma 6.3, the average size of the *CH_table* is $O(dk^2)$. Since both d , and k are constants and independent of the network size, the average size of *CH_table* is $O(1)$ ⁶.

Recall from section 3.1, *AC_table* is used by only CH nodes to keep track of adjacent clusters. Hence, we can calculate the average size of *AC_table* as follows:

$$\text{size}(AC_table) = E(P_{Adj}) \times \text{the expected number of boundary nodes}$$

However, the expected number of boundary nodes is equal to the average overlapping degree (AOD). Substituting from Eq.23, we get the following:

$$\text{size}(AC_table) = E(P_{Adj}) \times \frac{dk^2}{4\mu} = O\left(\frac{d^2k^4}{\mu}\right)$$

⁶Notice that the maximum size of *CH_table* can not exceed the average number of clusters (pn)

Since both d , and k are constants and independent of the network size, the average size of AC_table is $O(1)$. Hence, on the average, the total memory usage per node is $O(1)$. \square

7 Related Work

In the last few years, many algorithms have been proposed for clustering in wireless ad-hoc networks [60, 39, 32, 7, 5, 37, 35, 6, 33, 8, 10, 31, 49, 9, 21, 4, 26, 47]. Clustering algorithms can be classified as either deterministic or randomized. Deterministic algorithms [9, 10, 21, 6, 33, 5, 48, 50] use weights associated with nodes to elect cluster heads. These weights can be calculated based on number of neighbors (*node degree*) [9, 10], node id [6, 33, 5], residual energy, and mobility rate [21]. Each node broadcasts the calculated weight. Then a node is elected as a cluster head if it is the highest weight among its neighboring nodes. In randomized clustering algorithms, the nodes elect themselves as cluster heads with some probability p and broadcast their decisions to neighbor nodes [39, 60, 8, 7, 11]. The remaining nodes join the cluster of the cluster head that requires minimum communication energy. The probability p is an important parameter in a randomized algorithm. It can be a function of node residual energy [39] or hybrid of residual energy and a secondary parameter [60]. In [7], the authors obtain analytically the optimal value for p that minimizes the energy spent in communication. In OK, the probability p is tuned to control the number of overlapping clusters in the network.

The Distributed Clustering Algorithm (DCA) [10] elects the node that has the highest node degree among its 1-hop neighbors as the cluster head. The DCA algorithm is suitable for networks in which nodes are static or moving at a very low speed. The Distributed and Mobility-adaptive Clustering Algorithm (DMAC) [9] modifies the DCA algorithm to allow node mobility during or after the cluster set-up phase. The Weighted Clustering Algorithm (WCA) [21] calculates the weight based on the number of neighbors, transmission power, battery-life and mobility rate of the node. The algorithm also restricts the number of nodes in a cluster so that the performance of the MAC protocol is not degraded. In the Linked Cluster Algorithm (LCA) [6], a node becomes the cluster head if it has the highest identity among all nodes within one hop of itself or among all nodes within one hop of one of its

neighbors. The LCA algorithm was revised [33] to decrease the number of cluster heads produced in the original LCA. In this revised version of LCA (LCA2), the algorithm elects as a cluster head the node with the lowest id among all nodes that are neither a cluster head nor are within 1-hop of the already chosen cluster heads. Both LCA and LCA2 heuristics were developed to be used with small networks of less than 100 nodes. As the number of nodes in the network grows larger, LCA/LCA2 will impose greater delays between node transmissions in the TDMA communication scheme and may be unacceptable.

Many of these clustering algorithms [6, 33, 21, 9, 48] are specifically designed with an objective of generating stable clusters in environments with mobile nodes. But in a typical wireless sensor network, the sensors' locations are fixed and the instability of clusters due to mobility of sensors is not an issue. However, the network is still dynamic because of node failure or adding new nodes. Moreover, the clustering time complexity in some protocols [10, 21, 8, 48, 50] is dependent on the network diameter. Most of these algorithms have a time complexity of $O(n)$, where n is the total number of nodes in the network. This makes them less suitable for sensor networks that have a large number of sensors. Unlike those protocols, OK terminates in a constant number of iterations.

Some clustering algorithms make assumptions about node capabilities, e.g., location-awareness or clock synchronization. In [58, 59, 18, 22], the geographic location of each node is assumed to be available based on a positioning system such as GPS or through broadcast messages and routing updates [SPAN]. This is again not a reasonable assumption in case of low-cost low-power sensor networks. The clustering algorithm proposed in [Chiasserini02] assumes that each node is aware of the whole network topology, which is usually impossible for wireless sensor networks with a large number of nodes. Some algorithms [6, 33, 39, 21, 10, 9] require time synchronization among the nodes, which makes them suitable only for networks with a small number of sensors.

The majority of clustering algorithms construct clusters where every node in the network is no more than 1 hop away from a cluster head [6, 33, 10, 21, 9, 8, 31, 60, 32, 39]. We call these single hop (1-hop) clusters. For example, the HEED [60] algorithm forms single-hop non-overlapping clusters with the objective of prolonging network lifetime. Cluster heads are randomly selected according to a hybrid of their residual energy and a secondary

parameter, such as node proximity to its neighbors or node degree. A careful selection of the secondary clustering parameter can balance load among cluster heads. HEED performance was analyzed assuming synchronized nodes. However, the authors showed that unsynchronized nodes can still execute HEED independently, but cluster quality will be affected. In [32], the authors present a clustering algorithm (FLOC) that produces non-overlapping and approximately equal-sized clusters. The clustering is such that all nodes within one hop from a cluster head belongs to its cluster, and no node m hops away from the cluster head may belong to its cluster. In [37, 35] the clustering algorithm assumes gateway (*master*) nodes are already known and the objective is to perform load balancing between different clusters by changing cluster radius. In large networks single-hop clustering may generate a large number of cluster heads and eventually lead to the same problem as if there is no clustering.

In [39], Heinzelman et al. have proposed a distributed algorithm for wireless sensor networks (LEACH) in which the sensors randomly elect themselves as cluster heads with some probability and broadcast their decisions. The remaining sensors join the cluster of the cluster head that requires minimum communication energy. This algorithm allows only 1-hop clusters to be formed. LEACH assumes that all nodes are within communication range of each other and the base station (i.e. complete graph). LEACH clustering terminates in a constant number of iterations (like OK), but it does not guarantee good cluster head distribution and assumes uniform energy consumption for cluster heads [60].

Few papers have addressed the problem of multi-hop (k -hop) clustering [5, 7, 11]. These algorithms are mostly heuristic in nature and aim at generating the minimum number of disjoint clusters such that any node in any cluster is at most k hops away from the cluster head. For example, the algorithm described in [11] constructs clusters such that all the nodes within $R/2$ hops of a cluster head belong to that cluster head and the farthest distance of any node from its cluster head is $3.5R$ hops where R is an input parameter to the algorithm. With high probability, a network cover is constructed in $O(R)$ rounds; the communication cost is $O(R^3)$. The OK clustering algorithm has a much lower communication overhead. In [5], the authors presented the Max-Min heuristic to form non-overlapping k -clusters in a wireless ad hoc network. Nodes are assumed to have non-deterministic

mobility pattern. Clusters are formed by broadcasting node identities along the wireless links. When the heuristic terminates, a node either becomes a cluster head, or is at most k wireless hops away from its cluster head. The value of k is a parameter of the heuristic. Although the Max-Min algorithm generates k -hop clusters with a run-time of $O(k)$ rounds, it does not ensure that the energy used in communicating information to the information center is minimized. Both OK and MaxMin have $O(k)$ iterations. However, OK needs exactly $2k$ iterations to terminate but MaxMin needs at least $2k$ iterations. This means that the communication overhead is reduced in OK compared with MaxMin. In case of sensor networks, this directly affects the energy level of the node. In [7], the authors proposed a LEACH-like randomized clustering algorithm for organizing the sensors, in a wireless sensor network, in a hierarchy of clusters with an objective of minimizing the energy spent in communicating the information to the processing center (*base station*). They used results from stochastic geometry to obtain analytically the optimal number of cluster heads at each level of clustering.

None of the above algorithms construct overlapping clusters. Most of these algorithms are heuristic in nature and their aim is either to generate the minimum number of multi-hop clusters [5] or to minimize the energy spent in the network [7]. To the best of our knowledge, this is the first paper to discuss the problem of overlapping multi-hop clustering. We show that constructing the minimum overlapping k -hop dominating set in an ad hoc network is NP-complete. Then we propose OK, a randomized multi-hop distributed algorithm to solve the problem. The nodes randomly elect themselves as cluster heads with some probability p . The clustering process terminates in $O(1)$ iterations, independent of the network diameter, and does not depend on the network topology or size. OK operates in quasi-stationary networks where nodes are location-unaware and have equal significance. The protocol incurs low overhead in terms of processing cycles and messages exchanged. We also analyze the effect of different parameters (e.g. cluster radius, network connectivity, cluster head probability) on the performance of the clustering algorithm in terms of communication overhead, node coverage, and average cluster size.

OK is similar to the clustering algorithm described in [7] since both algorithms belong to the class of randomized multi-hop clustering. In [7], the main focus of the work

was to find the optimal number of cluster heads at each level of clustering analytically, and apply this recursively to generate one or more levels of clustering. However, our main focus is to generate overlapping clusters with certain overlapping degree. Our main contributions are (i) to formalize the problem of overlapping multi-hop clustering; (ii) extend the work in [7] to meet the design goals; (iii) show how to tune the parameters (p and k) given to the algorithm in order to achieve the design goals; (iv) give analytical models to formulate the problem. In [7], the cluster radius (k) was calculated analytically to minimize the energy. In OK, the cluster radius is a parameter that can be tuned to increase overlapping degree between clusters, or to decrease the cluster size (load balancing), or to decrease communication overhead.

8 Conclusions and Future Work

In this report, we have presented a scalable randomized multi-hop clustering protocol for ad-hoc sensor networks. OK organizes the sensors into overlapping clusters in a distributed manner. We have formulated the overlapping multi-hop clustering problem as an extension to the k -dominating set problem. OK is scalable in terms of communication overhead and terminates in a constant number of iterations independent of the network size. Although OK generates overlapped clusters, the simulation results show that the clusters are approximately equal in size. OK parameters, such as cluster radius, average node degree, and cluster head probability can be easily tuned to achieve the design goals with high probability. We have developed a detailed analytical model and have shown that it is valid by comparison with the simulation results. The proposed clustering scheme can be used by several types of sensor network protocols that require scalability, load balancing, and some degree of overlapping between the clusters. While OK appears to be a promising protocol, there are some areas for improvement to make the protocol more widely applicable. In the current implementation of OK, we assume stationary nodes, which is a valid assumption, for sensor networks. In the future, we plan to analyze the case when the nodes are mobile. Another issue that we are currently looking at is sharing the cluster head role among nodes. In general, cluster head nodes spend relatively more energy than other sensors because they

have to receive information from all the sensors within their cluster. Hence, they may run out of their energy faster than other sensors. One possible solution is to run the clustering algorithm periodically for load balancing. Another possibility is that cluster heads trigger the clustering algorithm when their energy levels fall below a certain threshold. We are currently investigating the behavior of the proposed clustering algorithm in the event of sensor failures. In the analytical model, we assume circle representation for the cluster. We are looking to different improvements to derive a tighter bound by studying the actual cluster shape using more complex stochastic geometry techniques.

A Appendix

A.1 Area of Intersection Between Two Identical Circles

Assume that we have two identical circles A and B that intersect in some area I_{AB} . Let r be the radius length and w be the distance between the two centers A and B as shown in Fig. 22. then the intersection (I_{AB}) can be calculated as follows:

$$I_{AB} = 2 (\text{area of sector } CBD - \text{area of triangle } CBD)$$

$$\text{Area of sector } CBD = \frac{1}{2} \cdot 2\theta \cdot R^2 = \theta \cdot R^2$$

$$\therefore I_{AB} = 2(\theta R^2 - \frac{1}{2} \cdot R^2 \sin 2\theta) = (2\theta - \sin 2\theta)R^2$$

$$\text{where } w = 2R \cos \theta \text{ (using cosine rule)}$$

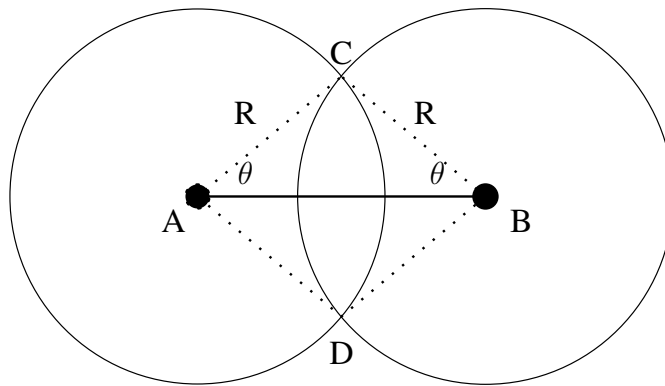


Figure 22: Area of intersection of two circles

References

- [1] K.M. Alzoubi, P.-J. Wan, and O. Frieder. Distributed Heuristics for Connected Dominating Sets in Wireless Ad Hoc Networks. *Journal of Communications and Networks*, 4(1), March 2002.
- [2] K.M. Alzoubi, P.-J. Wan, and O. Frieder. Message-Optimal Connected Dominating Sets in Mobile Ad Hoc Networks. In *MOBIHOC*, EPFL Lausanne, Switzerland, June 2002.
- [3] K.M. Alzoubi, P.-J. Wan, and O. Frieder. New Distributed Algorithm for Connected Dominating Set in Wireless Ad Hoc Networks. In *Proceedings of the 35th Hawaii International Conference on System Sciences*, Big Island, Hawaii, 2002.
- [4] A.D. Amis and R. Prakash. Load-Balancing Clusters in Wireless Ad Hoc Networks. In *Proceedings of ASSET*, Richardson, Texas, March 2000.
- [5] Alan D. Amis, Ravi Prakash, Thai H. P. Vuong, and Dung T. Huynh. Max-Min D-Cluster Formation in Wireless Ad Hoc Networks. In *IEEE INFOCOM*, March 2000.
- [6] D. J. Baker and A. Ephremides. The Architectural Organization of a Mobile Radio Network via a Distributed Algorithm. *IEEE Transactions on Communications*, 29(11):1694–1701, November 1981.
- [7] Seema Bandyopadhyay and Edward Coyle. An Energy-Efficient Hierarchical Clustering Algorithm for Wireless Sensor Networks. In *IEEE INFOCOM*, San Francisco, CA, March 2003.
- [8] S. Banerjee and S. Khuller. A clustering scheme for hierarchical control in multi-hop wireless networks. In *IEEE INFOCOM*, 2001.
- [9] S. Basagni. Distributed and Mobility-Adaptive Clustering for Multimedia Support in Multi-Hop Wireless Networks. In *Proceedings of Vehicular Technology Conference*, volume 2, 1999.

- [10] S. Basagni. Distributed Clustering for Ad Hoc Networks. In *Proceedings of International Symposium on Parallel Architectures, Algorithms and Networks*, pages 310–315, June 1999.
- [11] J. Beal. A robust amorphous hierarchy from persistent nodes. *AI Memo*, (11), 2003.
- [12] Jeremy Blum, Min Ding, Andrew Thaeler, and Xiuzhen Cheng. *Handbook of Combinatorial Optimization*. Kluwer Academic Publishers, 2004.
- [13] K. S. Booth and J. H. Johnson. Dominating sets in chordal graphs. *SIAM J. Comput.*, 11, 1982.
- [14] S. Butenko, X. Cheng, C. Oliveira, and P.M. Pardalos. *Recent Developments in Cooperative Control and Optimization*. Kluwer Academic Publishers, 2004.
- [15] S. Butenko, C. Oliveira, and P.M. Pardalos. A new algorithm for the minimum connected dominating set problem on ad hoc wireless networks. In *Proceedings of CCCT'03*, pages 39–44, 2003.
- [16] M. Cadei, X. Cheng, and D.-Z. Du. Connected Domination in Ad Hoc Wireless Networks. In *Proc. 6th International Conference on Computer Science and Informatics*, 2002.
- [17] Srdjan Capkun, M. Hamdi, and J. P. Hubaux. GPS-free Positioning in Mobile Ad-Hoc Networks. *Cluster Computing Journal*, April 2002.
- [18] A. Cerpa and D. Estrin. ASCENT: Adaptive Self-Configuring Sensor Networks Topologies. In *Proceedings of IEEE INFOCOM*, New York, NY, June 2002.
- [19] G. J. Chang and G. L. Nemhauser. The k-domination and k-stability problem on graphs. Technical Report 540, School of Operations Research and Industrial Engineering, Cornell University, 1982.
- [20] Y.-L. Chang and C.-C. Hsu. Routing in wireless/mobile ad-hoc networks via dynamic group construction. *Mobile Networks and Application*, May 2000.

- [21] M. Chatterjee, S. K. Das, and D. Turgut. WCA: A Weighted Clustering Algorithm for Mobile Ad hoc Networks. *Journal of Cluster Computing, Special issue on Mobile Ad hoc Networking*, (5):193–204, 2002.
- [22] 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*, 8(5), September 2002.
- [23] Y. Chen and A. Liestman. Approximating minimum size weakly-connected dominating sets for clustering mobile ad hoc networks. In *MOBIHOC*, EPFL Lausanne, Switzerland, June 2002.
- [24] X. Cheng, M. Ding, and D. Chen. An approximation algorithm for connected dominating set in ad hoc networks. In *Proc. of International Workshop on Theoretical Aspects of Wireless Ad Hoc, Sensor, and Peer-to-Peer Networks (TAWN)*, 2004.
- [25] X. Cheng, M. Ding, D.H. Du, and X. Jia. On The Construction of Connected Dominating Set in Ad Hoc Wireless Networks. *Special Issue on Ad Hoc Networks of Wireless Communications and Mobile Computing*, 2004.
- [26] C.F. Chiasserini, I. Chlamtac, P. Monti, and A. Nucci. Energy Efficient design of Wireless Ad Hoc Networks. In *Proceedings of European Wireless*, February 2002.
- [27] L. Clare, G. Pottie, and J. Agre. Self-organizing distributed sensor networks. In *SPIE Conf. Unattended Ground Sensor Technologies and Applications*, pages 229–237, Orlando, FL, April 1999.
- [28] B. N. Clark, C. J. Colbourn, and D. S. Johnson. Unit disk graphs. *Discrete Mathematics*, 86, 1990.
- [29] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms, Second Edition*. MIT Press and McGraw-Hill, 2001.
- [30] M. Scott Corson and A. Ephremides. A Distributed Routing Algorithm for Mobile Wireless Networks. *ACM Journal on Wireless Networks*, 1(1):61–81, 1995.

- [31] B. Das and V. Bharghavan. Routing in Ad-Hoc Networks Using Minimum Connected Dominating Sets. In *ICC*, 1997.
- [32] Murat Demirbas, Anish Arora, and Vineet Mittal. FLOC: A Fast Local Clustering Service for Wireless Sensor Networks. In *1st Workshop on Dependability Issues in Wireless Ad Hoc Networks and Sensor Networks*, Florence, Italy, June 2004.
- [33] A. Ephremides, J.E. Wieselthier, and D. J. Baker. A Design concept for Reliable Mobile Radio Networks with Frequency Hopping Signaling. *Proceeding of IEEE*, 75(1):56–73, 1987.
- [34] M. R. Garey and D. S. Johnson. *Computers and Intractability: A guide to the theory of NP-completeness*. Freeman, San Francisco, 1978.
- [35] S. Ghiasi, A. Srivastava, X. Yang, and M. Sarrafzadeh. Optimal Energy Aware Clustering in Sensor Networks. *Sensors Magazine*, (1):258–269, January 2002.
- [36] S. Guha and S. Khuller. Approximation algorithms for connected dominating sets. *Algorithmica*, 20(4):374–387, April 1998.
- [37] G. Gupta and M. Younis. Load-Balanced Clustering in Wireless Sensor Networks. In *the International Conference on Communication (ICC 2003)*, Anchorage, Alaska, May 2003.
- [38] T. W. Haynes, S. T. Hedetniemi, and P. J. Slater. *Domination in Graphs: Advanced Topics*. Marcel Dekker, Inc. New York, 1998.
- [39] W. B. Heinzelman, A. P. Chandrakasan, , and H. Balakrishnan. An Application Specific Protocol Architecture for Wireless Microsensor Networks. *IEEE Transactions on Wireless Networking*, 1(4), October 2002.
- [40] M. A. Henning, O. R. Oellermann, and H. C. Swart. The diversity of domination. *Discrete Mathematics*, 161(3):161–173, December 1996.
- [41] M. A. Henning, O.R. Oellermann, and H.C. Swart. Bounds on distance domination parameters. *J. Combin. Inform. System Sci.*, 16, 1991.

- [42] Steven Huss-Lederman, Elaine M. Jacobson, Anna Tsao, Thomas Turnbull, and Jeremy R. Johnson. Implementation of strassen's algorithm for matrix multiplication. In *Supercomputing '96: Proceedings of the 1996 ACM/IEEE conference on Supercomputing (CDROM)*, Washington, DC, USA, 1996.
- [43] Xiang Ji. Sensor Positioning in Wireless Ad-hoc Sensor Networks with Multidimensional Scaling. In *In Proc. of IEEE InfoCom*, Hong Kong, March 2004.
- [44] V. Kawadia and P. R. Kumar. Power Control and Clustering in Ad Hoc Networks. In *IEEE INFOCOM*, San Francisco, CA, March 2003.
- [45] P. Krishna, N. H. Vaidya, M. Chatterjee, and D. K. Pradhan. A Distributed Routing Algorithm for Mobile Wireless Networks. *ACM SIGCOMM Computer Communications Review*, 1997.
- [46] B. Krishnamachari, S. Wicker, and R. Bejar. Phase Transition Phenomena in Wireless Ad-hoc Networks. In *in GLOBECOM*, San Antonio, TX, 2001.
- [47] T. J. Kwon and M. Gerla. Clustering with Power Control. In *Proceeding of MilCOM99*, 1999.
- [48] C. R. Lin and M. Gerla. Adaptive Clustering for Mobile Wireless Networks. *Journal on Selected Areas in Communication*, 15:1265–1275, September 1997.
- [49] A. B. McDonald and T. Znati. A Mobility Based Framework for Adaptive Clustering in Wireless Ad-Hoc Networks. *IEEE Journal on Selected Areas in Communications*, 17(8):1466–1487, August 1999.
- [50] A. K. Parekh. Selecting Routers in Ad-Hoc Wireless Networks. In *Proceedings of ITS*, 1994.
- [51] T. Rappaport. *Wireless Communications: Principles & Practice*. Englewood Cliffs, NJ: Prentice-Hall, 1996.

- [52] Andreas Savvides, C. C. Han, and M. Srivastava. Dynamic Fine-Grained Localization in Ad-hoc Networks of Sensors. In *in Proceedings of ACM Mobicom*, pages 166–179, Rome, Italy, July 2001.
- [53] Yi Shang and Wheeler Ruml. Improved MDS-Based Localization. In *In Proc. of IEEE InfoCom*, Hong Kong, March 2004.
- [54] Yi Shang, Wheeler Ruml, Ying Zhang, and Markus P. J. Fromherz. Localization From Mere Connectivity. In *In Proc. of ACM MOBIHOC 2003*, pages 201–212, Annapolis, MD, June 2003.
- [55] S. Skiena. *Handbook of Combinatorial Optimization*. Reading, MA: Addison-Wesley, 1990.
- [56] P.-J. Wan, K.M. Alzoubi, and O. Frieder. Distributed Construction of Connected Dominating Sets in Wireless Ad Hoc Networks. In *IEEE INFOCOM*, 2002.
- [57] Tao Wu and Subir K. Biswas. A self-reorganizing slot allocation protocol for multi-cluster sensor networks. In *Fourth International Conference on Information Processing in Sensor Networks (IPSN '05)*, pages 309–316, April 2005.
- [58] 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)*, pages 70–84, Rome, Italy, July 2001.
- [59] F. Ye, H. Luo, J. Chung, S. Lu, and L. Zhang. A Two-Tier Data Dissemination Protocol for Large-Scale Wireless Sensor Networks. In *Proceedings of the ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM)*, Atlanta, Georgia, September 2002.
- [60] Ossama Younis and Sonia Fahmy. Distributed Clustering in Ad-hoc Sensor Networks: A Hybrid, Energy-Efficient Approach. In *IEEE INFOCOM*, Hong Kong, March 2004.
- [61] A. Youssef, A. Agrawala, and M. Younis. Accurate Anchor-Free Localization in Wireless Sensor Networks. In *in the Proceedings of the 1st IEEE Workshop on In-*

formation Assurance in Wireless Sensor Networks (WSNIA 2005), Phoenix, Arizona, April 2005.

- [62] Adel Youssef. *SALAM: A Scalable Anchor-Free Localization Algorithm for Wireless Sensor Networks*. PhD thesis, Computer Science Department, University of Maryland, College Park, 2006.