Hierarchical Clustering and Routing for Data Centric Sensor Networks

Srikanth Kandula, Jennifer C. Hou, and Lui Sha
Department of Computer Science
University of Illinois at Urbana Champaign
Urbana, IL 61801
{*kandula,jhou,lrs*}@*cs.uiuc.edu*

***Abstract*—**

In this paper, we develop, and rigorously prove the optimality (in order of magnitude) of, a decentralized, hierarchical clustering algorithm that considers the several unique characteristics of data-centric sensor networks. These characteristics include non-uniform distribution of devices, device heterogeneity, limited battery lives of sensors, and spatially localized information redundancy in sensor data collection.

Specifically, we envision a two-level, hierarchical cluster structure in which a level-0 cluster is statically composed of sensors in a hexagon in a two-dimensional hexagonal grid, and a level-1 cluster is "centered" at an energy-rich sensor device (called an *accumulator*). Level 1 clusters are dynamically formed by all the level-0 hexagons (and the sensors therein) using notions of geographical proximity. Sensors send the location/tracking information that they have gathered, either periodically or on-demand, to their respective level-0 cluster heads which then relay this information to the nearest accumulator. The accumulator is then responsible for generating a concise digest of all the sensor data received within an interval and sending this digest to a data sink. The network lifetime is then improved by taking advantage of route redundancy and turning off some of the sensors in a level-0 cluster that will not impair network connectivity. Moreover, we show by exploiting certain geometrical properties of hexagons, the control overhead incurred in the dynamic, level-1 cluster formation process can be minimized, the network will not be partitioned after clustering except under rare conditions, and routing is greatly facilitated with the information obtained during the cluster formation process (thereby incurring little extra overhead). In particular, we have shown that the total number of control messages exchanged is the least possible for any cluster formation mechanism, and that the information throughput (rather than raw data throughput) is optimized.

Index Terms — **Data-centric sensor networks, hierarchical cluster formation, cluster-based routing, network connectivity and longevity.**

I. Introduction

Recent technological advances have led to the emergence of small, low-power devices that integrate sensors and actuators with limited on-board processing and wireless communication capabilities. Pervasive networks of such sensors and actuators open new vistas for constructing complex control systems. Unlike traditional wired or wireless networks, sensor networks possess certain characteristics which warrant their treatment as a special class of ad hoc networks:

(a) **Data-centric** Sensor networks are largely data-centric, with the objective of delivering data collected to the required destination rather than performing intensive computation on the data.
(b) **Application-oriented** While traditional wired and wireless networks are expected to cater to a variety of user applications, a sensor network is usually deployed to perform specific tasks. This property makes it possible to enable nodes to respond in an application-aware fashion.

An example of this type of sensor networks is homeland security at airports, bridges, and public buildings. The overall effectiveness of surveillance would be improved when a man-in-the-loop approach is coupled with a large number of low cost, autonomous motion detectors and acoustic monitors that are equipped with wireless interfaces and peform the audio-visual locating and tracking of the region. Another example is a military surveillance network that consists of a large number of wireless devices that can be deployed from air.

A major requirement of this type of sensor networks is for the sensors to reliably disseminate information within a time frame that allows the controller to take necessary action, even in the case of poor spatial distribution of sensor devices, wireless/acoustic interference, and malicious destruction/sabotage. This presents a key technical challenge in cooperative engagement — how to effectively coordinate and control sensors over an unreliable wireless ad hoc network. The first step to tackle this challenge is to devise, with consideration of power management and data forwarding/routing, an effective algorithm for sensors to self-organize themselves and dynamically form an ad hoc network.

There are several unique characteristics of data-centric sensor networks that one should take into account in devising such a self-organizing algorithm for sensor networks. Firstly, because of physical, geographical, and communication constraints, sensor devices may not be evenly distributed over a region. Air-deployment is greatly affected by extraneous factors such as wind direction and terrain. Secondly, all sensors need not be created equal. Most of the sensor devices have limited battery life and transmission/computational capability, but a few of them may be equipped with better processing capability, stronger transmission power, and longer battery life, and can perform processing and communication operations. Thirdly, as more than one sensor may observe the same phenomenon, the information collected by sensors may be correlated, redundant, and/or of different qualities. Finally, as most of the sensor devices have limited battery life, it is essential to exercise power management in order to improve the network lifetime (so as to maximize the surveillance period). These characteristics suggest that an ad hoc network formed in the structure of hierarchical clusters offer the greatest performance benefits, as hierarchies easily take into account non-uniform spatial distribution and allow us to exploit device heterogeneity.

In this paper, we develop, and rigorously prove the optimality[1] of, a decentralized, hierarchical clustering algorithm that exploits heterogeneity among nodes to obtain greater information throughput, piggybacks routing over cluster management and improves the network lifetime.

The rest of the paper is organized as follows. In Section II, we outline the design objective and give the network and system models. Sections III discusses the hierarchical clustering algorithm. Following that, we address in Section IV routing in and between different levels of clusters in the hierarchical cluster architecture. We formally prove several properties of the hierarchical clustering algorithm in Section V and present related work in Section VII. We conclude the paper in Section VIII.

II. Design Objectives and System Model

Design objectives: As can be extracted from the application scenarios discussed in Section I, the most important requirements for devising a decentralized, hierarchical clustering algorithm for data-centric sensor networks are:

(1) **Energy conservation**: Stringent constraints on battery life make energy conservation imperative. Energy consumption is directly proportional to the time spent in transmitting/receiving data and to the time spent in the "eager to relay" state. Paradoxically, nodes that serve as natural bridges by virtue of their geographical location are essential to maintain network connectivity but they are also usually the earliest to deplete their batteries due to high relaying load.

(2) **Maintenance of network connectivity**: For a sensor network to perform its tasks, a majority, if not all, of the sensors should remain connected. Energy conservation techniques, such as pushing nodes into "will not relay" states, may result in network partitions and hence should be adopted with care.

(3) **Robustness**: The sensors should have the capability to self-organize themselves into a functional network and to recover in the case of link/node failure and/or malicious destruction.

(4) **Scalability**: The algorithms devised to realize the required network functions should not introduce significant computation and control message overheads, and should scale well with the number of nodes in the network.

(5) **Adapted to limited mobility**: It is expected that node mobility is limited in sensor networks. After the initial deployment, topological changes may occur due to factors like wind, rain, hostile activity, or battery outages.

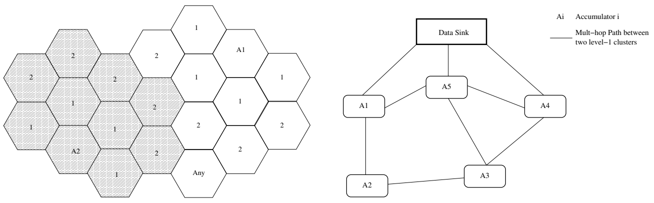
System Model: We classify sensor devices into three classes: sensors, accumulators, and data sinks. Sensors are limited in power and computational capacity, and are only responsible for gathering partial location information and transmitting it to the nearest accumulator. For, ease of specification, it is assumed that two sensors would be able to communicate whenever they are within a geographical distance of W, called the *communicable distance*, of each other. The value of W is conservatively chosen to be smaller than the actual wireless transmission range to account for shadowing, interference and other adverse effects and is clearly dependent on the topology of the scene of deployment. Sensors are also assumed to be equipped with GPS devices that allow them to obtain accurate estimates of their location. An accumulator, by design, is equipped with better computational and transmission power and may also have privileged access to the shared wireless channel. It is the responsibility of an accumulator to store the sensorial information captured by the sensors belonging to its cluster. The accumulator, then compacts the data received using application-specific algorithm(s) into a digest, timestamps the digest and transmits it to one or more data sinks. Data sinks are special accumulators that may be equipped with dedicated links, such as encrypted satellite uplinks, that allow them to transmit the information to interested users who may be physically located at great distances from the deployment area.

III. Decentralized, Hierarchical Clustering Algorithm

In compliance with the device classification in Section II, we envision a two level cluster structure. A level-0 cluster is composed of sensors in a hexagon of side $\frac{W}{\sqrt{13}}$ in a two-dimensional hexagonal grid. Among the sensors in a level-0 cluster, a sensor device (or an accumulator if one exists) that wins the distributed cluster head election phase acts as the level-0 cluster head. By assigning the job of relaying cross traffic to level-0 cluster heads alone, power at all the other sensors is conserved. A level-1 cluster is "centered" at an accumulator and comprises all the level-0 hexagons (and the sensors therein) for whom the accumulator is the closest in

[1] in order of magnitude

terms of hops over level-0 clusters. For example, as shown in Fig. 1(a), the white and shaded areas are, respectively, two level-1 clusters with accumulators A_1 and A_2 as their level-1 cluster heads. The hop counts of each hexagonal grid are as shown in the picture.



(a) (b)

Fig. 1. A two-level hierarchical clustering structure: (a) Two neighboring level-1 clusters and their associated level-0 clusters. The label of a hexagonal grid is its distance to the closest accumulator. (b) A network graph that is composed of level-1 clusters. Each multi-hop path between two level-1 clusters traverses several level-0 clusters.

Sensors send the location information gathered either periodically or on-demand to their respective level-0 cluster heads which then relay this data to the nearest level-1 cluster (accumulator). The accumulator is then responsible for generating a single concise digest from these correlated, redundant data samples that are possibly of different quality, and send the digest to a data sink. The route between accumulators and/or from an accumulator to a data sink may pass through several level-1 clusters, and the (multi-hop) path between two level-1 clusters traverses several level-0 clusters (Fig. 1(b)). Under the assumption of a non-sparse (but not necessarily uniform) distribution of sensors, multiple routes of the same hop count exist between any two accumulators. This flexibility facilitates maintenance of network connectivity and robustness (Section IV). Sensor failures can be seamlessly handled by re-routing through alternate routes without need for an explicit route discovery phase.

A. Formation of Level-0 Clusters

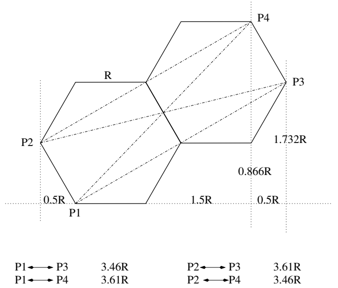


Fig. 2. Determination of the size of the level-0 hexagonal grid.

Similar to GAF [7] in which rectangular grids are formed, we establish a two dimensional hexagonal grid of side size $\frac{W}{\sqrt{13}}$. All the sensors in each hexagonal grid form a level 0 cluster. As shown in Fig. 2, the maximum distance between any pair of devices in adjacent hexagonal grids of side R is $\sqrt{13} \times R$. Hence, by setting $R = \frac{W}{\sqrt{13}}$, we ensure devices in adjacent grids are within "communicable distance" of each other. A device can map its GPS obtained location to the containing grid in constant time [4].

A cluster head is elected based on a distributed election algorithm in which a sensor device responds to an election with a delay that has two components. The deterministic part of the delay is inversely proportional to the residual power that the node has, while the randomized part of the delay is an order of magnitude smaller than the deterministic part and attempts to avoid collisions.

$$Delay = \frac{x}{Power} + R + S, where\ R \in [0, \frac{x}{2 * MaxPower}] \quad (1)$$

$$S_{accumulator} = 0, \quad S_{sensor} \in [0, \frac{x}{2 * MaxPower}] \quad (2)$$

Hence sensor devices with a larger power, would respond with a smaller delay, and are more likely to be elected as cluster heads. To ensure an accumulator (if present) is always elected as a level-0 cluster head, we enforce sensors to delay their responses to a cluster head election by a constant duration during which only accumulators can respond. Responses to the election request contain duration values indicating the duration of time for which the responder would want to be the leader. Accumulators advertise infinite duration values while sensors advertise smaller values proportional to the remaining power.

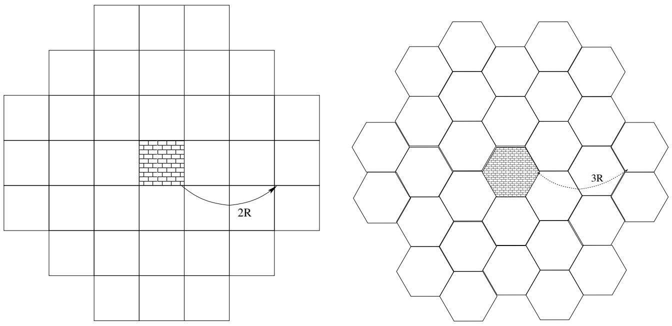
$$D_{accum} = \infty, \ \ D_{sensor} = y * Power_{remaining} \quad (3)$$

In the case that no accumulator exists in a hexagonal grid, the responsibility of serving as a cluster head is rotated among the existing sensors to improve the overall lifetime of the network. At any time, only the cluster head has to remain awake in each hexagonal grid in order to maintain network connectivity. Moreover, during the period in which sensors wake up, perform surveillance, and forward location/tracking information to their level-0 cluster head, they need to only use a transmission power of $2R$, instead of $\sqrt{13}R$, during the periods of waking up and performing surveillance.

The reason for constructing level-0 clusters in the form of hexagons is three fold:

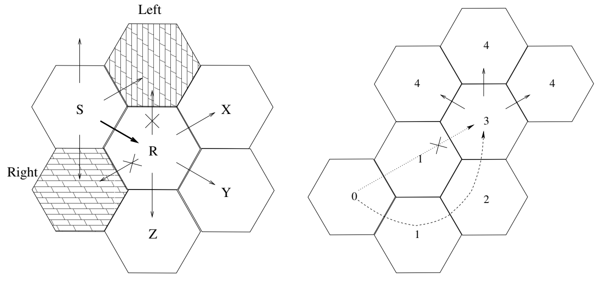
(i) **Grid Homogeneity:** In hexagonal grids, all the neighboring grids are homogeneous in the sense that there exists no neighboring grid that shares a corner (as in the case of rectangular grids), rather than a side.

(ii) **Less chance of network partition after clustering:** A network that is originally connected in the flat topology would become disconnected, only when a node A has at least one neighbor, say node B, within its wireless transmission range, but *not* within any of its own/neighboring hexagonal grids. We formally show in Section V that the probability of loss of connectivity in hexagonal grids is 25% of that in rectangular grids, or a 300% improvement.



(a) Rectangular grid (b) Hexagonal grid

Fig. 3. The number of grids in the vicinity that may conflict with a given grid. The grid in consideration is filled and all other grids in the vicinity that could conflict with this grid have been shown.



(a) (b)

Fig. 4. How the number of join messages can be reduced in the course of cluster formation. (a) The shaded level-0 clusters are neighbors of the originating level-0 cluster and have already received a join message. (b) MAC layer scheduling order might lead to explosion of join messages

(iii) **Lower area of interference** We assume two nodes to be contending if they are within the communicable distance W of each other. We note that a node can only lower the number of other nodes it contends with by reducing its transmission power. Given the choice of hexagonal side length (Section III-A), the number of clusters whose transmission may contend with a given cluster is 31 in the worst case (Figure 3 (b)). In comparison, if rectangular grids are used and the grid size is set to R to maintain network connectivity as in GAF[7], where $R = \frac{W}{\sqrt{5}}$ then the number of potentially contending clusters is 37 in the worst case (Figure 3 (a)). This is a 16% improvement in terms of both the number of grids and the geographical area of coverage.

B. Formation of Level-1 Clusters

Level 0 clusters (hexagonal grids) are statically formed using geographical information in order to increase the network lifetime. Level 1 clusters, on the other hand, are dynamically formed in a decentralized manner by accumulators in order to exploit node heterogeneity, i.e., nodes with significantly larger resources (called accumulators) gather correlated, redundant data of different quality and compress them into more compact digests.

Forwarding of join *messages:* Once an accumulator A_1 is elected as the level-0 cluster head of its local hexagon, it broadcasts a join message to its adjacent level-0 clusters. The join message contains, among other things, A_1's identifier and the hop count from A_1 (which is initialized to 1). Upon receipt of a join message, a level-0 cluster head checks if it has received a join message from any accumulator with a smaller hop count. If not, it updates its (accumulator id, hop count) record and forwards the join message to its neighbors.

To reduce the number of join messages exchanged, we note that

1) A join message should not be forwarded back to either the originator grid or to neighbors of the originating grid. To achieve this, the join message can be extended to contain the grid identifier of the originating level-0 cluster. Upon receipt of a join message, a level-0 cluster, identifies the originator and forwards the message to the *other* three clusters that are neighbors in the opposite direction as shown in Fig. 4 (a).
2) A join message advertising a longer path should *never* be forwarded. As shown in Fig. 4 (b), if a join message advertising a longer path ($0 \rightarrow 1 \rightarrow 2 \rightarrow 3$) arrives earlier at a level-0 cluster than the one advertising the best path ($0 \rightarrow 1 \rightarrow 3$), the level-0 cluster head may forward the first message to all subsequent level-0 clusters in the opposite direction. A best-effort MAC layer might lead to such a situation due to scheduling conflicts. Although this does not impair the correctness of the algorithm as the join message advertising the best path will eventually reach all level-0 clusters, significant power may have been consumed in sending unnecessary join messages. This also increases the cluster formation time as more messages are now generated and need to be received before clusters can be formed.

To deal with this problem, the join message is further extended to include the hop counts of the forwarding level-0 cluster's two neighbors that are also adjacent to the cluster to which the message is being forwarded. For example, in Fig. 4(a), when grid S forwards join to grid R, it also includes the hop counts of the grids marked *left* and *right*. That is, a join message has the following fields:

(i) A : the Accumulator Identifier;
(ii) H : the advertised hop count;
(iii) G : the identifier of the forwarding level-0 cluster (grid);
(iv) H_r, H_ℓ : the hop counts of the common neighbors of the sending and receiving grids, where the right and left neighbors are defined with respect to the sender;
(v) S: the sequence number.

With the above information, a level-0 cluster head can judge whether or not it should forward a join message, with the hop count H_{advt} to a neighboring cluster using the following rules.

$$R \rightarrow X : \textbf{if } H_\ell = H \textbf{ then } H_{advt} = H + 1, \text{forward } \textbf{else } \textbf{!}\text{forward} \quad (4)$$
$$R \rightarrow Y : H_{advt} = H + 1, \text{forward} \quad (5)$$
$$R \rightarrow Z : \textbf{if } H_r = H \textbf{ then } H_{advt} = H + 1, \text{forward } \textbf{else } \textbf{!}\text{forward} \quad (6)$$

Figure 5 depicts how join messages are being forwarded in the same example given in Figure 4 (b). As shown in the Figure 5 when the leader of the grid marked 'X' receives message 'a', it realizes that its northern neighbor 'B' also has a hop-count of '1' and so forward its join message onward to the neighboring grid on the eastern side 'Y'. Now when this grid 'Y' receives the message 'b', it realizes that the neighbor to the north-west 'B' has a lower hop count than itself and so desists from forwarding a message to its northern neighbor 'Z'. Thus, by exploiting geometrical properties, we avoid the forwarding of join messages that advertise incorrect hop counts.

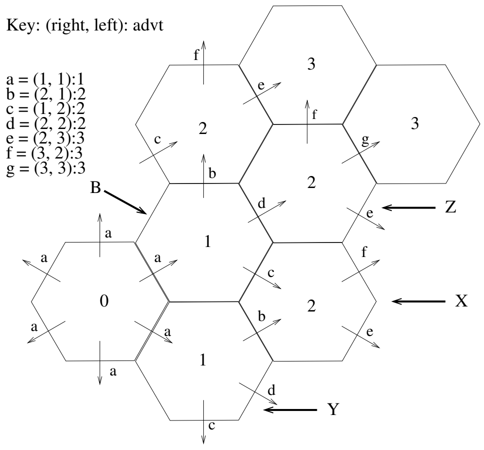


Fig. 5. An illustration that demonstrates how join messages are being forwarded.

Note that the state kept at each level-0 cluster head is soft, and each level-0 cluster head periodically re-transmits a join message so that message losses will not adversely affect the correctness of the protocol. If the underlying link layer protocol (MAC) does not have a mechanism that allows each level-0 cluster head to know which neighboring hexagonal grid is alive, i.e., at least one sensor exists in that grid, then neighbors would have to periodically exchange *HELLO* messages indicating "liveness". This liveness information is used to forward join messages in the following ways: (i) join messages are forwarded only to "live" neighbors; and (ii) the hop count of a "dead" grid is taken to be infinite when join messages are exchanged between grids that are both neighbors of a "dead" grid.

As a further (*broadcast join*) optimization, instead of dispatching individual join messages, all the forwarding activity of a grid can be performed by a single broadcast from the grid leader, exploiting the idea that the values of H_r, H_ℓ that need to be forwarded are either $H, H-1$ or $H+1$, when H is the hop count of the current grid and using bit-masks. Interested readers are referred to [4].

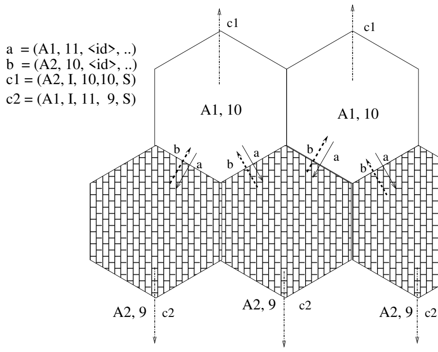


Fig. 6. Termination of the join message forwarding process.

Creation of level-1 clusters: The process of forwarding join messages (originated by an accumulator A_1) terminates when the message reaches a level-0 cluster with a smaller hop count to a different accumulator A_2 (Fig. 6). The recipient of the join message discards the message, and returns a **N**eighbor **A**c**K**nolwedgement message toward the sender cluster. The NAK message is essentially the join message of the other accumulator. Upon receipt of a NAK message, a level-0 cluster head knows it is a border grid of the originating accumulator A_1, and forwards a **B**order **R**outer **M**essage back to A_1 via its upstream neighbor grids. The BRM message has the following fields:

(i) A_2: the identifier of the adjacent accumulator (level-1 cluster head);
(ii) B: the identifier of the level-0 border grid;
(iii) H_1: the hop count from B to the original accumulator A_1;
(iv) H_2: the hop count from B to the adjacent accumulator A_2;
(v) S: the sequence number.

All the level-0 clusters that forward the BRM message to the accumulator maintain a list of accessible border grids and, for each border grid, the list of neighboring grids that can be used to reach this border grid. As will be briefly discussed in Section IV, this facilitates routing between accumulators. For details, refer [4]. Upon receipt of a BRM message, an accumulator keeps track of the list of its adjacent level-1 clusters and for each adjacent level-1 cluster, the set of associated border grids. Again all the states kept by level-0 cluster heads are soft, and hence in case of accumulator failure, all the level-0 cluster heads will rejoin another accumulator.

C. Properties of the Proposed Cluster Formation Algorithm

The proposed cluster formation algorithm, without the *broadcast join* optimization, possesses several desirable properties, which we state below and will formally prove in Section V:

(P1) Each level-0 cluster head, except the originating accumulator, forwards a join message to at least two distinct grids and at most three distinct grids; and each level-0 cluster head receives a join message from at least one grid and at most two distinct grids.

(P2) The number of join messages that are sent during the formation of a level-1 cluster is bounded and is of $O(n)$, where n is the number of grids in that cluster. In comparison, a simple protocol of always broadcasting join messages results in $O(n^2)$ messages.

(P3) The total number of join messages exchanged during the *entire* cluster formation phase is bounded by $O(N)$, where N is the total number of live hexagonal grids. This is the best possible for any cluster formation mechanism.

IV. Routing in the Two-Level Cluster Architecture

Routing in the two-level cluster topology can be categorized into the following types:

Within level-0 clusters: Sensors within a level-0 cluster merely forward location/tracking information to their level-0 cluster head. There is very little communication from a level-0 cluster head to individual sensors (except for the cluster head election within a level-0 cluster). This also means that level-0 clusters have to buffer little information for forwarding to other sensors within their level-0 clusters.

Within level-1 clusters: Each level-0 cluster head forwards sensor information to its nearest level-1 cluster head (an accumulator). This can be easily realized by having level-0 cluster heads keeping appropriate soft state upon receipt of join, BRM, and NAK messages. Specifically,

(i) Upon receipt of a join message, a level-0 cluster head updates the identifier of the nearest accumulator and the associated upstream neighbor level-0 cluster heads.

(ii) Upon receipt of a BRM message, if the sequence number of this message is smaller than the one currently recorded for the same border grid, a level-0 cluster head discard the message. Otherwise, the level-0 cluster head updates the grid identifier of the reachable border cluster and the associated downstream neighbor level-0 cluster head(s).

(iii) Upon receipt of a NAK message, a level-0 cluster head identifies itself as a border cluster and keeps track of a list of adjacent level-1 clusters and the neighbor grid to reach each adjacent level-1 cluster.

Sensor information that needs to be sent to an accumulator is simply forwarded to an upstream neighbor level-0 cluster head, i.e. the grids that sent out the "best" join messages to the current grid. If there are more than one upstream neighbors, a level-0 cluster head may select an upstream neighbor based on a choice of load balancing or randomization decision.

Across level-1 clusters: Digests generated by accumulators and control messages exchanged among accumulators constitute inter-cluster traffic. Each accumulator keeps track of the following state:

(iv) a list of adjacent level-1 clusters;

(v) for each adjacent level-1 cluster, a list of level-0 border clusters (grids) to reach the adjacent accumulator and the associated hop counts;

(vi) for each border cluster, the list of immediate, downstream neighbor grids that can be used to reach that border cluster.

All of the information can be collected/updated from the BRM messages received. For details, refer [4]. In addition, accumulators communicate with one another (via link state or distance vector protocol) to obtain the level-1 network topology in which each node is a level-1 cluster head.

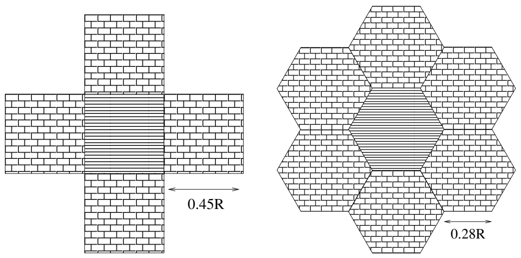
To send a message to an adjacent accumulator (or a data sink), an accumulator picks up a level-0 border cluster and forwards the message to it. Upon receipt of such a message, an interior grid checks whether the identifier of this border cluster is already present in its list of border clusters that are downstream to it. If yes, it forwards the message to one of the downstream grids that can be used to reach that border cluster (item (iii)); otherwise, it simply ignores this message. The latter (erroneous) case may occur during transient conditions. Upon receipt of a message, a level-0 border cluster forwards it to one of its neighboring level-0 clusters that belong to the destination level-1 cluster. Upon receipt of a cross-border message, a level-0 border cluster forwards it to its accumulator as if it were sensor information generated by this grid, albeit with a different priority in order to account for the fact that this information has already been "processed".

Note that this hierarchical cluster structure handles the existence of "empty" grids as long as there exists some alternate path that by-passes the empty grids. Otherwise, the sensors and/or accumulators are in a network partition that does not have a data sink and are thus rendered useless. This is because, each accumulator keeps track of, for each neighboring accumulator, a set of routes that can reach that neighboring accumulator. The network is partitioned only if sensors are extremely sparsely distributed and multiple, empty grids exist to "block" all possible routes to certain direction. As mentioned before, "empty" grids are handled by assigning to them, a hop-count of "infinity", during join message forwarding.

V. Properties of Two-Level Clustering and Their Formal Proofs

In this section, we prove several properties stated in Sections III-A and III-C.

We first prove in Theorem 1 that the probability of network partition as a result of clustering is smaller under hexagonal grids than under rectangular grids. A similar argument proves that the contention is 16% lesser in hexagonal grids but has been omitted due to space constraints.



(a) Area covered by a rectangular grid (b) Area covered by a hexagonal grid

Fig. 7. The fraction of neighborhood preserved after clustering.

Theorem 1: Let p_r and p_h denote, respectively, the probability that a pair of sensors within the wireless transmission range of each other remains in neighboring clusters under rectangular grids and hexagonal grids. Then, assuming uniform density, $p_h = 4.2 \times p_r$.

Proof: The probability that a pair of sensors within the wireless transmission range of each other remains in neighboring clusters is

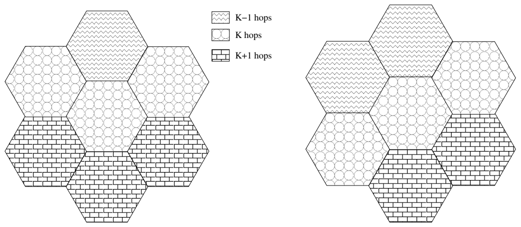
$$p = \frac{\text{Communicable area in a grid scheme}}{\text{Communicable area in the flat network}}.$$

As shown in Figure 7, a node in a rectangular grid can communicate with nodes in its four neighboring grids, while a node in a hexagonal grid can communicate with nodes in its six neighboring grids. That is, Communicable area in a

$$\begin{aligned} \text{flat network} &= \pi \times W^2, \text{ for both schemes,} \\ \text{rectangular grid} &= 5 \times (\frac{W}{\sqrt{5}})^2 = W^2, \\ \text{hexagonal grid} &= 7 \times 6 \times \frac{3\sqrt{3}}{4} \times (\frac{W}{\sqrt{13}})^2 = 4.2 \times W^2. \end{aligned}$$

Hence, $p_h = 4.2 \times p_r$. □

We now establish the theoretical base of (**P1**) with Theorems 2–5. For notational convenience, we define the *label* of a level-0 grid as the distance (in terms of hop count) from itself to the closest accumulator.



(a) Type 1 (b) Type 2

Fig. 8. Types of the labels associated with a non-accumulator grid and its six neighbors.

Theorem 2: Under the assumption that there exists at least one sensor in each hexagonal grid, the labels associated with a non-accumulator grid and its six neighbors are of one of the types shown in Figure 8.

Sketch of proof: The intuition behind this result is based on the following observation. Since the hop count of a grid is directly proportional to the distance from the geometric center of the accumulator grid to its geometric center, the geometric centers of grids that have a label k for any $k \geq 1$ form a hexagon. It can be observed that only the grids at the corners of this hexagon are of type 1, while every other grid is of type 2. A formal algebraic proof is given in Appendix □

Theorem 3: Every join message that originates in a grid with a label of n is forwarded to a grid with a label of $n + 1$.

Proof: We prove this by Induction on the hop-count advertised in a join message.
Base Case: Every join message that advertises a hop-count of 1 originates from the accumulator (hop-count $= 0$) and is forwarded only to the 6 neighboring grids (those grids with a label of 1).
Inductive Hypothesis: Every join message advertising a hop-count of $k + 1$ originates in a grid with a (hop-count) label k and reaches a grid with a (hop-count) label $k + 1$.
Inductive Step: A join message advertising a hop-count of $k + 1$ is sent by a grid if and only if the grid receives a join message that advertises a hop-count of k. By the inductive hypothesis, every originator of a $k + 1$-hop join message has a (hop-count) label of k. Furthermore, the k-hop join message received uniquely classifies this grid as either a type 1 or a type 2 grid. By construction of the join message forwarding algorithm, a $k + 1$-hop join message is forwarded only to those grids with a (hop-count) label of $k + 2$. This proves the correctness of the algorithm. □

Theorem 4: Each level-0 cluster head receives a join message from at least one grid and at most two distinct grids.

Proof: This follows directly from Theorems 2 and 3. Each hexagonal grid is either of type 1 or type 2. Let the (hop-count) label of the grid under consideration be k. Theorem 3 concludes that a join message advertising a hop-count of k can only originate from a grid with a hop-count label of $k - 1$. Furthermore, only neighboring grids can forward messages to this grid. Thus, if this grid were of type 1, it would receive a join message from exactly 1 neighbor and if it were of type 2, it would receive a join message from exactly 2 neighbors. □

Theorem 5: Each level-0 cluster head, except the originating accumulator, forwards a join message to at least two distinct grids and at most three distinct grids.

Proof: This follows directly from Theorems 2 and 3 along the lines of Theorem 4. Cluster-heads forward join messages only to thier nieghbors and are either of Type 1 or Type 2. □

Properties (**P2**) and (**P3**) can be deduced directly from the above theorems. For completeness of the paper, we list the theorems below and leave their proofs in [4].

Theorem 6: The number of join messages that are sent within a level-1 cluster during cluster formation is bounded and is of $O(n)$, where n is the number of grids in a level-1 cluster. Moreover, the total number of join messages exchanged is bounded by $O(N)$, where N is the total number of live hexagonal grids. □

VI. Simulation Results

We performed ns-2 simulations to verify the correctness of the proposed clustering algorithm, to evaluate the performance of the clustering algorithm at various levels of sensor and accumulator densities both in isolation and in comparison with the naive approach, to analyze the improvements in data throughput realized by on-the-fly compression at the accumulators en-route and to quantify the network lifetime improvements.

A. Methodology

A pre-defined hexagonal grid map of side $\frac{W}{\sqrt{13}}$ was superimposed on a rectangular flat terrain, varying in area between .36 km^2 to 4 km^2. The wireless range of each sensor was assumed to be 250m. Between 100 to 500 nodes were distributed uniformly-at-random over the rectangular region. One of the nodes was randomly chosen to be the sink and between 0 to 100 accumulators were used in the experiments. Protocol agents that participate in cluster formation and route establishment/maintenance were deployed at each of these nodes. The cluster formation phase is repeated periodically. Data sources were chosen randomly from the non-accumulator nodes and each was configured with a constant bit rate traffic generator working above an UDP agent. For all simulations wherein necessary, the accumulators are assumed to perform a 50% data compression, i.e. the digest functionality of the accumulator generates 1 digest packet for every 2 raw packets.

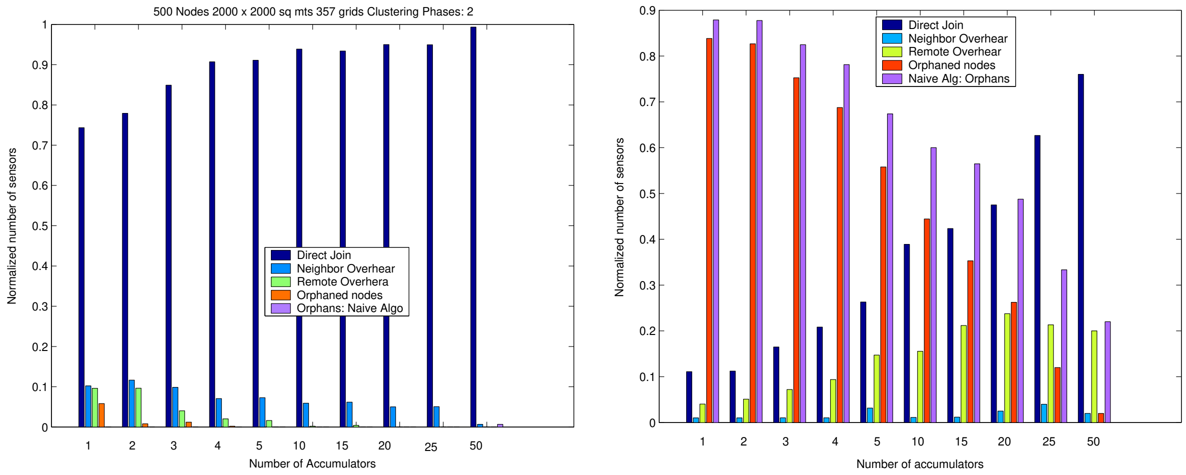
B. Clustering Message Overhead and Success rate

The proposed cluster formation algorithm reduces overhead by eliminating redundant (and some potentially incorrect) advertisement-forwarding instances. Empty grids, if not detected prior to clustering, could lead to a number of isolated nodes in cluster formation as the proposed algorithm optimistically prunes advertisements. We introduced two new heuristics into the cluster formation algorithm in order to persevere in the prescence of empty grids. The key idea behind both these heuristics is to exploit the broadcast nature of advertisement forwarding by allowing nodes to use overheard advertisements. We ensure the correctness of the protocol by disabling forwarding of overheard advertisements and by conservative updation of the current state relative to the advertised hop count. The heuristics differ in terms of the source of the overheard message.

- Neighbor Overhear: Due to the conservative cluster formation algorithm, advertisement broadcasts contain a bitmask that enumerates the neighboring grids that need to process this advertisement. When the leader of a grid G_1 overhears an advertisement with a shorter hop-count from a neighboring grid G_2 that does not explicitly contain G_1 in its destination list, G_1 has to, in the ideal case, receive an advertisement with the same or shorter hop-count from a different neighbor. Hence, G_1 can safely update its current state to that advertised by G_2 while refraining from forwarding this advertisement further as the advertised hop-count could be sub-optimal for G_1. Figure 9(a) shows that this heuristic could account for almost 10% of the nodes during cluster formation.
- Remote Overhear: Proceeding on similar lines to the above heuristic, let a grid G_1 overhear an advertisement from a grid G_2 that is not a neighbor then, the longest possible distance in-terms of hop-count between G_1 and G_2 is 4 due to the nature of hexagonal grid formation. If G_2 advertises a hop-count of h, then G_1, in the ideal case, cannot have a hop-count worse than $h+4$. Hence, G_1 updates its local state if the previously known hop-count is worse than $h + 4$, and refrains from forwarding this advertised hop-count onwards as it could be sub-optimal. Figure 9(b) shows that this heuristic could account for almost 20% of the nodes during cluster formation.

For a moderately dense network with 500 nodes in $4km^2$ area, Figure 9(a) shows that a concentration of about 1.25% (4 out of 500) is sufficient to have a completely successful clustering, with some contribution from the nieghbor overhear heuristic. Figure 9(b) shows a much sparser topology with 100 nodes distributed among 357 grids. The large number of empty grids present necessitate a high proportion of accumulators. A 25% accumulator concentration leads to coverage of about 90% of the nodes. However, the number of failures of the proposed algorithm are significantly smaller than that of the naive algorithm partly due to reduced overheads and interference and partly due to the heuristics. The high fraction of holes in the sparse topology lead to the remote overhear heuristic contributing significantly more than the neighbor overhear heuristic. On both denser and smaller topologies as shown in Figure 10, the proposed algorithm achieves a 0% failure rate as long as an accumulator is present. The smaller the topology, the less dense it needs to be as a significant fraction of empty grids are bridged by the remote overhear metric.

Figure 11 compares the number of cluster-formation messages created during cluster formation. Recall that due to scheduling conflicts effects the naive algorithm could create as many as $O(n^2)$ messages as opposed to the $O(n)$ bound in messages created by the proposed algorithm. In practice, Figure 11(a) shows that the overhead of the naive algorithm is between 210% to 120% of the overhead of the proposed algorithm. Whenever scheduling conflicts cause an anomalous advertisement to reach a grid earlier than the appropriate one,



(a) Moderately Dense Network (b) Sparse Network

Fig. 9. Normalized number of nodes that succeeded after two runs of the cluster formation algorithm compared to the number of failures in the naive algorithm, at various concentrations of accumulators

#Nodes	Area	#Accumulators	%Failure
500	.36 km^2	1, 2, 3, 4, 5, 10, 15	0
100	.36 km^2	1, 2, 3, 4, 5, 10, 15	0

Fig. 10. Cluster formation in Dense Topologies

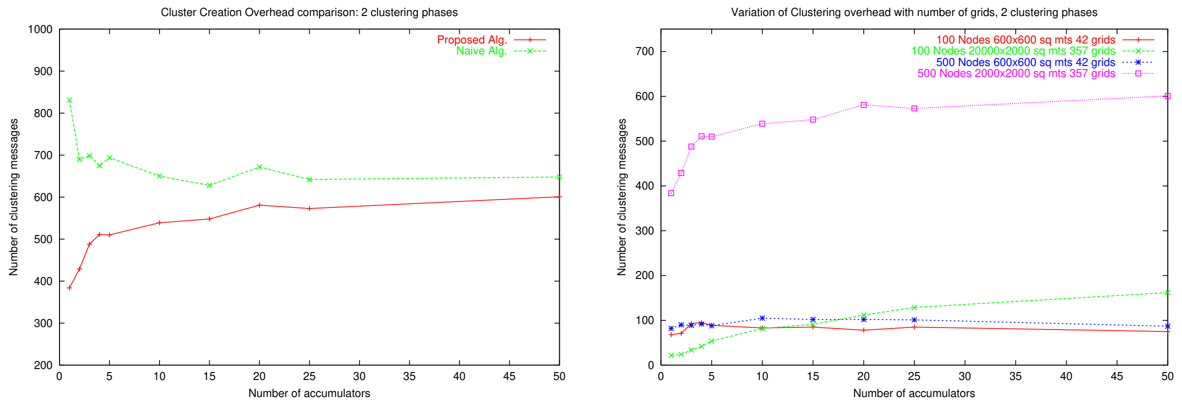
a spurious chain of advertisements would start from this grid. The longer this chain of spurious advertisements is, the more will be the number of redundant messages that are created. The length of an advertisement chain is inversely proportional to the number of accumulators, as fewer accumulators would necessitate larger Level 1 clusters and hence the difference between the naive algorithm and the proposed algorithm is magnified at lower numbers of accumulators.

Figure 11(b) compares the clustering overhead of the proposed algorithm at different sensor and accumulator density. The overhead of the proposed algorithm is $O(n)$ where n is the number of grids and is therefore independent of the number of accumulators. This is evident from the fact that the cluster formation overhead converges as the number of accumulators increase. Further the larger the area of the map, the more the number of grids n, hence the overhead is proportional to the area of the simulation topology. While bounded above by the number of grids in the region, the overhead is more tightly bound by the number of non-empty grids as only these grids are responsible for the creation and receipt of messages. Hence the sparse topology of 100 nodes in 357 grids has a much smaller overhead than the much denser 500 nodes in 357 grids scenario.

C. Data throughput improvements

At each fixed CBR source rate, we observe the improvement in the percentage of packets that are successfully delivered at each level in the heirarchy. Figures 12(a, b) show that the level 0 packets, packets travelling from each sensor to its closest accumulator have a significantly higher success ratio (greater than 200%) when compared to the base case of 0 accumulators. This is because of reduction in the amount of cross-traffic across Level 1 clusters and also the reduced distance each message has to traverse. In general, the longer a packet that is guaranteed un-prioritized best-effort delivery travels in the network, the greater would be its loss probability. Since the size of each Level 1 cluster grows smaller as the number of accumulators increase, the Level 0 success probability consistently increases with increasing accumulators. Figures 12 (c, d) demonstrate the fact that the probability of success of a Level 1 message, sent from an accumulator to the sink, consistently increases with increasing number of accumulators. The reason for this is that the sheer volume of L1 messages decrease with increase in the number of accumulators, thereby reducing interference.

Figures 12 (e, f) depict variation in the end-to-end success ratio, which is defined as the fraction of data packets that have been successfully assimilated by the sink. While both the L0 success ratio and the L1 success ratio play a key role in determining the end-to-end success probability, L1 success ratios have been found to be strongly correlated with the number of data sources, probably because the longer distance each L1 packet has to traverse increases the probability of its being dropped at higher load. However, for both the lightly loaded 5 CBR source case and the heavily loaded 15 CBR source case, it is seen that the prescence of accumulators has a considerable positive impact on the end-to-end success ratio. Figures



(a) Overhead of the naive algorithm

(b) Overhead variation with node and accumulator density

Fig. 11. Cluster formation overhead comparison

12 (g, h) depict variation in the end-to-end ratio, under the assumption of idealized L1 data delivery. If a sensor network deployment can ensure guaranteed data transmission between the accumulators and the sink, then it is seen that even under heavy load a high end-to-end success ratio is guaranteed in the prescence of accumulators, considerably better than the base case.

VII. Related Work

Several clustering algorithms have been proposed in the literature, among which the k-cluster-based routing scheme [6], the zone routing protocol (ZRP) [3], the spine routing framework [2], and the min ID/max degree scheme [5] may have received the most attention. Although none of them is specifically targeted for data-centric sensor networks or builds upon the notion of hierarchy, we summarize those that are most pertinent to our work and compare them against our work.

K-cluster-based routing and ZRP: In k-cluster-based routing [6], the network is dynamically organized into k-clusters, where all the nodes in a cluster can be reached from any other node within the cluster in k hops. Then the Dijkstra's shortest path algorithm is used to build the routing table. Similarly, ZRP [3] enables nodes to maintain their own *routing zones* — clusters of nodes that can be reached along paths that are at most n hops and routes using a hybrid strategy (proactive intra-zone routing + demand-based inter-zone routing) to balance the tradeoff between proactive and reactive routing.

Spine: The spine routing framework [2] was built upon the notion of *spines* (virtual backbones) — a set of relatively stable and connected nodes such that every node is either part of the spine or one hop away from a node in the spine. While the framework does not specify completely its routing algorithm, it presents two approaches: clustered spine routing and partial spine routing. The framework presents a dedicated backbone for information dissemination, but spine maintenance and routing are costly and introduce significant control traffic when updates are made. Our architecture exploits the geometric property of level-0 clusters to devise a simpler, much lighter weight routing mechanism. Distributed maintenance of state in each level 0 cluster head ensures route recovery and also allows for balancing the load on the routes.

Min-ID/max degree scheme: In the min ID/max degree scheme [5], min ID and max degree are used to group nodes into clusters (in a greedy fashion) within which two nodes are at most two hops away. In the min ID algorithm, each node has a globally unique ID and clustering is done in multiple stages. In each stage, every node that has atleast one neighbor not already belonging to a cluster elects itself if it has the least ID among all such neighbors. Similarly, in the max degree algorithm, each node broadcasts the list of nodes it can hear. A node is elected as a cluster head if it has the maximal node degree among all the "uncovered" neighbor nodes, where a uncovered node is one which does not yet have an elected cluster head.

There are several notable differences between the Min-id/max-degree clusters and our level-0 clusters.

(i) in the min-ID/max degree scheme, cluster heads are elected in a deterministic fashion and leadership cannot be rotated among nodes to achieve power optimality. In contrast, in the proposed scheme leadership is rotated and the burden of forwarding packets is spread across all sensors within a cluster.

(ii) The transmission power can not be changed and remains to be maximum in the min-ID/max degree scheme. In contrast, in the proposed scheme non-leader nodes can reduce their transmission power to $\frac{2}{\sqrt{13}}$ of their maximum wireless transmission power.

(iii) The overhead incurred in forming level-0 clusters in the min-ID/max-degree scheme is significantly higher ($O(n)$ time) than the proposed solution ($O(1)$ time). The price the proposed scheme pays in achieving the little overhead, is, however, the need for GPS support.

(iv) A simulation study[4] has shown that in the min-ID/max degree scheme the number of neighboring clusters that could conflict with a cluster (within the

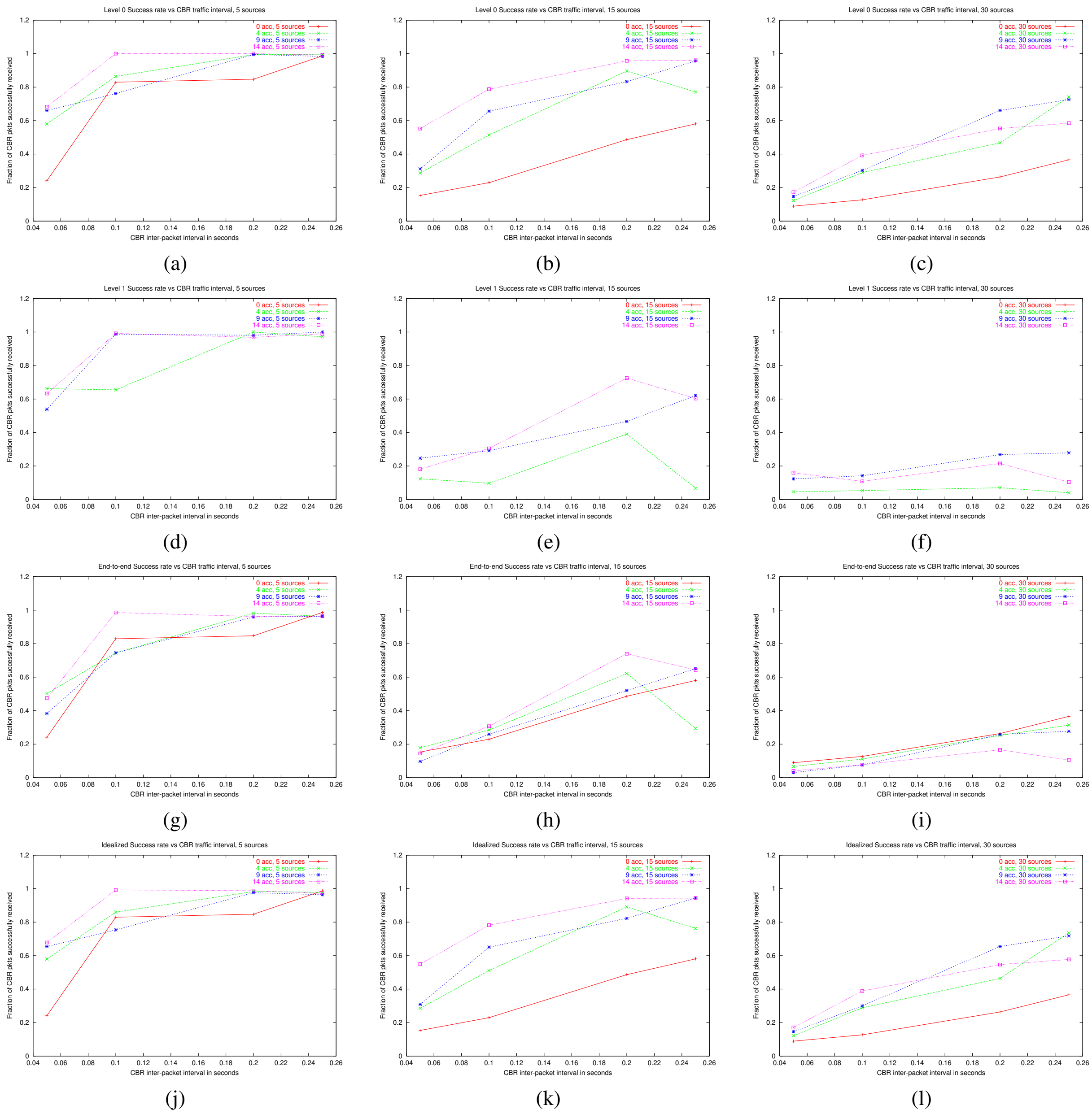


Fig. 12. Throughput Improvements

wireless transmission range) is 9 to 11 on average and 19 in the worst case. The proposed scheme, on the other hand, has 31 neighboring clusters that may conflict with a given cluster on average [4]

Span: Span [1] is a distributed and randomized algorithm, and does not require the support of GPS. In Span, nodes can make local decisions on whether they should sleep or join a forwarding backbone as a coordinator. The nodes that choose to stay awake and maintain network connectivity/capacity are called coordinators. The rule for electing coordinators is that if two neighbor nodes of a non-coordinator node can neither directly communicate with each other nor through one or two coordinators, then this node volunteers to be a coordinator. The results show that Span saves energy by a factor of 3.5 or more, and increases the lifetime of the network by a factor of 2.

However, level-0 clusters in Span, may be formed in $O(n)$ time (in the worst case) as compared to the $O(1)$ time in the proposed scheme. Also, Span's clusters are greedily formed causing intra-cluster distances as large as W, thereby eliminating power savings that could otherwise be achieved by reducing transmission range of non-leaders.

GAF: Our level-0 cluster formation algorithm bears the closest resemblance to GAF [7]. GAF assumes the availability of GPS and conserves energy by dividing a region into rectangular grids, ensuring that the maximum distance between any pair of nodes in adjacent grids is within the transmission range of each other, and electing a leader in each grid to stay awake and relay packets (while putting all the other nodes into sleep).

As discussed in Section III-A, the number of clusters that may conflict with a given cluster is 37 in the worst case in GAF (as compared to 31 in the worst case in the proposed scheme[4]. Furthermore, as proved in [4], GAF requires a higher density of node distribution to prevent network partition from occurrence after geographical clustering. Heterogenous nodes and hierarchical clusters are absent in GAF.

VIII. Conclusion

In this paper, we develop, and rigorously prove the optimality of, a decentralized, hierarchical clustering algorithm that considers the several unique characteristics of data-centric sensor networks, such as non-uniform distribution of devices, device heterogeneity, limited battery lives of sensors, and information redundancy in sensor data collected in the vicinity. The proposed algorithm possesses several desirable properties. Specifically,

(a) **Network connectivity**: If the network is sufficiently dense, network connectivity is maintained by our clustering algorithm. The only case in which a cluster formation mechanism would disconnect a network that is originally connected in the flat topology is when a node A has at least one neighbor, say node B, within its wireless transmission range, but *not* within any of its own/neighboring hexagonal grids. We have formally proved in [4] that such a probability is much lower under hexagonal grids than under rectangular grids.

(b) **Power consumption**: As mentioned in Section III-A, only the cluster head in a level 0 cluster needs to be active at any point of time. Furthermore, each sensor that is not a cluster head, needs only to communicate with its level-0 cluster leader, and hence can decrease its transmission power to $2R$, rather than $\sqrt{13}R$, where R is the side size of the hexagonal grid. The fact that the control overhead is controlled to be the least possible for any cluster formation mechanism also contributes to network longevity.

(c) **Scalability**: With (**P1**)–(**P3**) in Section III-C, the number of control messages (join, NAK, and BRM) is significantly reduced in the course of dynamic, level-1 cluster formation. As a matter of fact, we prove that the total number of control messages exchanged is the least possible for any cluster formation mechanism.

(d) **Information throughput**: As each accumulator generates a single concise digest from correlated, redundant data before relaying the information across level-1 clusters to a data sink, the amount of growth in the quantity of information to be relayed to a data sink is clearly far less than in the case of a homogeneous, flat network consisting merely of sensors and data sinks. This allows us to have fewer data sinks to handle the same number of sensors and to transmit more information.

(e) **Robustness**: As has already been indicated in Section IV, these exist several valid known routes between any pair of accumulators. The allows for tolerance of faults that arise due to power outages, malicious destruction, and/or limited mobility due to environmental factors (such as wind). The fact that all the states kept at level-0/1 cluster heads are soft and periodic refresh/retransmission of join messages ensure message losses and/or accumulator failure will not adversely affect the correctness.

(f) **Mobility Adaptation**: Mobility of nodes affects the proposed architecture in the following manner. Level 0 cluster leaders might move out of the geographical hexagons or grids on the path from a node to the accumulator might become empty. Both these cases are efficiently handled, by periodic re-election and periodic re-clustering.

We have identified several avenues for future research. We will study, in the context of hierarchical cluster structure, how level-0 cluster heads may achieve load balance with consideration of power management, when they forward messages to accumulators/data sinks. Although cluster-based routing is robust as long as there exists an alternate path that by-passes these empty grids, the network may still be partitioned in rare cases. In the case of network partition and/or existence of communication bottlenecks, we will study how to place, and coordinate movement of, mobile sensor devices that act as "routers" to fill the "holes."

References

[1] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris, "Span: An Energy-Efficient Coordination Algorithm for Topology Maintenance, in Ad Hoc Wireless Networks," *Proc. of the 7th ACM International Conference on Mobile Computing and Networking*, July 2001.

[2] B. Das, R. Sivakumar, and V. Bharghavan, "The Clade Vertebrate: Spines and Routing in Ad-Hoc Networks," *In Proc. of IEEE Symposium on Computers and Communication,* 1998.

[3] Z. Haas, "A New Routing Protocol for The Reconfigurable Wireless Networks," *In Proc. of the IEEE International Conference on Universal Personal Communications,* October 1997.

[4] S. Kandula, J. Hou and L. Sha, "Hierarchical Clustering for Data Centric Sensor Networks," Technical Report, Department of Computer Science, University of Illinois at Urbana Champaign, May 2002. Also available at http://www.cs.uiuc.edu/ kandula.

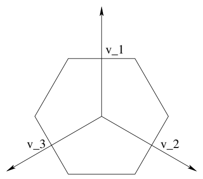
[5] C. R. Lin and M. Gerla, "Adaptive Clustering for Mobile Wireless Networks," *IEEE Journal on Selected Areas in Communications*, Vol. 15, No. 7, September 1997.

[6] N. H. Vaidya *et al.*, "A Cluster-Based Approach for Routing in Dynamic Networks," *ACM SIGCOMM Computer Communication Review,* pp. 49-65, April 1997.

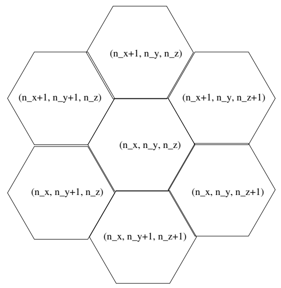
[7] Y. Xu, J. Heidemann and D. Estrin, "Geography-informed Energy Conservation for Ad Hoc Routing, *Proceedings of the 7th ACM International Conference on Mobile Computing and Networking*, July 2001.

APPENDIX

We prove the following stronger result in order to prove Theorem 2. To facilitate derivation, we define the unit vectors on a two dimensional plane, $\vec{v_1}$, $\vec{v_2}$ and $\vec{v_3}$ as depicted in Figure (a). Also, the label of a hexagonal grid is the distance from itself to the closest accumulator.



(a) Definition of unit vectors



(b) Neighbor information

Theorem 7: A grid can be reached from the accumulator grid in k hops if and only if the distance from the accumulator to this grid can be uniquely represented as a vector of the type $n_1\vec{v_1} + n_2\vec{v_2} + n_3\vec{v_3}$, where for some permutation (x, y, z) of $(1, 2, 3)$, the co-ordinates satisfy the condition $0 = n_x \leq n_y \leq n_z = k$.

Proof: We prove by induction on k, the number of hops.
Base Case: $k = 1$, as shown in Figure (a), all the six grids that have a hop count of 1 are represented as one of $\vec{v_1}$, $\vec{v_2}$, $\vec{v_3}$, $\vec{v_1} + \vec{v_2}$, $\vec{v_2} + \vec{v_3}$, and $\vec{v_1} + \vec{v_3}$.
Inductive Hypothesis: Assume that all grids with a hop-count label of $h \leq k$ have a *valid* representation and that every representation $n_1\vec{v_1} + n_2\vec{v_2} + n_3\vec{v_3}$ uniquely refers to a grid with a hop-count label of h whenever $0 = n_x \leq n_y \leq n_z = h$ for some permutation (x, y, z) of $(1, 2, 3)$.
Inductive Step: Every grid with a hop-count label $k + 1$ has at least one neighbor with a hop-count label of k, by definition of hop counts. Also, every grid with a hop-count label of k has neighbors with hop-count labels k, $k + 1$ or $k - 1$. Let us consider a grid with a hop-count label of k. Figure (b) depicts all possible neighbors of a grid represented by a permutation of (n_x, n_y, n_z). We have $0 = n_x \leq n_y \leq n_z = k$ from our inductive hypothesis. Let us evaluate each of the six possible neighbors.

Case 1: Neighbor $(n_x + 1, n_y, n_z)$. Clearly $n_z = k > 0$. If $n_y = 0$, then we have

$$0 = n_y < n_x = 1 < n_z = k.$$

By our inductive hypothesis, this grid has a hop-count label of k. If $n_y \neq 0$, then since $n_x + n_y + n_z = 0$, this neighbor can also be represented as $(n_x, n_y - 1, n_z - 1)$. Clearly we have,

$$0 = n_x \leq n_y - 1 \leq n_z - 1 = k - 1.$$

Again by our inductive hypothesis, this grid has a hop count label of $k - 1$.

Case 2: Neighbor $(n_x, n_y + 1, n_z$. If $n_y = n_z = k$, then we have

$$0 = n_x \leq n_z \leq n_y + 1 = k + 1.$$

This grid neither has a hop-count label of k nor has a hop-count label of $k-1$ by our inductive hypothesis. Thus this grid clearly has a hop-count label of $k + 1$ and can be represented as $(n_a = n_x, n_b = n_z, n_c = n_y + 1)$, where $0 = n_a \leq n_b \leq n_c = k + 1$. Alternatively, if $n_y < n_z$ then we have

$$0 = n_x \leq n_y + 1 \leq n_z = k.$$

Again by our inductive hypothesis, this grid has a hop-count label of k.

Case 3: Neighbor $(n_x, n_y, n_z + 1)$. Clearly we have

$$0 = n_x \leq n_y \leq n_z + 1 = k + 1.$$

By our inductive hypothesis, this grid neither has a hop-count label of k nor has a hop-count label of $k - 1$. Thus by virtue of its being a neighbor of a grid with a hop-count label of k, it has to have a hop-count label of $k + 1$.

Case 4: Neighbor $(n_x, n_y + 1, n_z + 1)$. We have

$$0 = n_x \leq n_y + 1 \leq n_z + 1 \leq k + 1.$$

Following the above argument this grid has a hop-count label of $k + 1$ and has a valid representation.

Case 5: Neighbor $(n_x + 1, n_y, n_z + 1)$. If $n_y = 0$, then we have

$$0 = n_y < n_x + 1 = 1 \leq n_z + 1 = k + 1.$$

This grid again has a hop-count label of $k + 1$ and has the correct representation. Otherwise, $(n_x + 1, n_y, n_z + 1) = (n_x, n_y - 1, n_z)$ and we have

$$0 = n_x \leq n_y - 1 \leq n_z = k.$$

This grid has a hop-count label of k.

Case 6: Neighbor $(n_x + 1, n_y + 1, n_z)$. We have $(n_x + 1, n_y + 1, n_z) = (n_x, n_y, n_z - 1)$. Thus if $n_y = n_z$, we have

$$0 = n_x \leq n_z - 1 \leq n_y = k.$$

This grid has a hop-count label of k. If $n_y < n_z$, we have

$$0 = n_x \leq n_y \leq n_z - 1 = k - 1.$$

This grid has a hop-count label of $k - 1$.

The above case analysis proves that every grid with a hop-count label of $k + 1$ can be represented in the valid format as it has to be the neighbor of a grid with a hop-count label of k. To prove that every representation (n_a, n_b, n_c) where $0 = n_a \leq n_b \leq n_c = k + 1$ represents a unique grid with a hop-count label of $k+1$, we argue as follows. Observe that all grids with a hop-count label of $k + 1$ have their centers on a hexagon, whose end-points have unique representations $(k +$

$1)\vec{v_1}$, $(k+1)\vec{v_2}$, $(k+1)\vec{v_3}$, $(k+1)(\vec{v_1}+\vec{v_2})$, $(k+1)(\vec{v_1}+\vec{v_2})$, $(k+1)(\vec{v_1}+\vec{v_2})$. Also, there are exactly k grids whose centers lie directly on the paths between any two adjacent end-points. All these grids have a hop-count label of $k+1$ by observation. Thus there are exactly $6+6k$ grids with a hop-count label of $k+1$. Each of these grids has a valid representation. If we count the total number of valid representations, we get exactly $6+6k$ of these as follows. The position of hop count $k+1$ can be chosen in three ways, that of 0 can be chosen in two ways and the remaining field could have any value from 1 to k. These give us $6k$ valid representations. Now, we have exactly three representations each of which has 2 fields equal to $k+1$ (and the third field = 0) or has 2 fields equal to 0 (and the other = $k+1$). Thus every $k+1$ valid representation has to uniquely represent a grid with a hop-count label of $k+1$. □

From Theorem 7, Theorem 2 follows from the above case analysis. Cases 3 and 4 always yield neighbors with a hop-count label of $k+1$. Furthermore, if $n_y = 0$, cases 1, 2, 5 and 6 yield neighbors with a hop count label of k, k, $k+1$ and $k-1$ respectively. This grid is clearly of type (a) as depicted in Figure 8. If $n_y = k$, then cases 1, 2, 5 and 6 yield neighbors with a hop count label of $k-1$, $k+1$, k and k respectively. This grid again is of type (a). Finally if $0 < n_y < k$, then cases 1, 2, 5 and 6 yield neighbors with a hop count of $k-1$, k, k and $k-1$ respectively. This grid is clearly of type (b) as depicted in Figure 8. □