On Improving Data Accessibility in Storage Based Sensor Networks

Tan Apaydin
Department of CSE
The Ohio State University
apaydin@cse.ohio-state.edu

Serdar Vural
Department of ECE
The Ohio State University
vurals@ece.osu.edu

Prasun Sinha
Department of CSE
The Ohio State University
prasun@cse.ohio-state.edu

Abstract

In-network data storage techniques can provide robustness against failure of dedicated sink nodes in sensor networks. For storing data in the network it is critical to ensure that the data can be accessed from other nodes for a long period of time. As replicating the data will increase consumption of resources, we propose a technique for data relocation that optimizes the accessibility of the data in the network. The two challenging constraints for solving this problem are "limited sensor lifetime" and "cost of data relocation". We study both centralized and distributed approaches for computing accessibility of each node. Using extensive simulations, we evaluate the performance of our techniques.

1 Introduction

There are two prevalent types of wireless sensor network architectures: *sink-based* and *storage-based*. Sink-based architectures have one or more dedicated sink nodes, that are often resource-rich. However, failure of the sink nodes can render the network useless even when the other nodes in the network are functional. To design for robustness against node failures, storage-based architectures [1] have emerged that use in-network storage and often do not have any dedicated sink nodes. Queries for data are assumed to originate from any node in the network. However, limited amount and fast depletion of energy resources restrict information storage capability of individual sensors. Furthermore, sensor nodes run the risk of being unavailable to the rest of the network when they exhaust the energy resources. Similarly, a sensor node whose neighbors run out of energy is disconnected from the network which again causes the stored data to be unavailable. Hence, data must be moved to other locations to avoid loss of critical information before node failures occur due to insufficient energy. Therefore, *it is desirable to have the data available to a large number of sensors for long periods of time*.

In literature, the data persistency problem in sensor networks is studied from different perspectives. In [2] and [3], replication of data to multiple locations is considered for enhancing data persistency. However, storing the data at multiple nodes consumes significant resources that is not feasible for some applications. Furthermore, the storage sensors should be frequently updated by the sensors collecting data, as new data arrives and existing ones may become obsolete, which leads to high network traffic. In contrast, this paper focuses on smart relocation of a single copy of the data for increased accessibility. The problem of information retrieval from a sink-based network, where node failures occur, has been recently studied in [7]. Data is encoded using a simple XOR based coding technique. Data encoding for later recovery during network failures is also studied in [6]. On the other hand, in this paper we consider a network without a dedicated sink and instead focus on the problem of determining more suitable locations to improve data accessibility. Another related problem is the maximization of network lifetime. In [12], the objective is to maximize the time until the energy of the first battery in the network is depleted. Similarly in [10], a distributed algorithm is proposed for the problem of maximization of network lifetime based on multi-commodity flow formulation. However, in a real deployment, the network still remains capable of collecting and storing information even after one or multiple nodes have depleted battery energy.

Maximum data extraction (gathering) problem is similar to the maximum lifetime problem in continuous data-gathering sensor networks with the addition of "data awareness" along with "energy awareness". In [9], the aim is to maximize the data collected by a single stationary sink from an energy-limited sensor network. In [5], the study focuses on finding an efficient way for data collection while maximizing the system lifetime. The data gathering problem is also dealt with in [4] for which the aim is to evenly distribute energy load in sensor networks with the use of base stations in local clusters. A later study which addresses the data gathering problem is presented in [8] with the objectives of minimization of energy consumption and latency of data gathering. However, this paper focuses on a different problem of in-network data relocation to optimize its accessibility.

In this study, we introduce the problem of moving the data of a node that is expected to get disconnected from the rest of the network. A node's disconnection from the network is considered to be caused by the depletion of its energy, or its neighbors' energies, or both. Such a situation is regarded as an *emergency incidence*. In this paper, we assume that node failures occur only due to battery drainage. Failures due to physical damage,

1-4244-1455-5/07/$25.00

though critical in certain scenarios, are out of the scope of this paper.

Our primary objective is to determine the most suitable node to relocate the data in an emergency incidence. Such a node is expected to be highly *accessible* to the network and can provide its data for long periods of time. However, "limited sensor lifetime" and "cost of data relocation" are two challenging constraints and we focus on the former constraint in this paper. To the best of our knowledge, there are no prior solutions to this problem in the literature. We seek to design efficient and scalable solutions to the problem.

The contribution of this study is threefold. First, we introduce a novel metric called *accessibility*, which corresponds to the capability of a node to make its data more available to the rest of the network. This metric captures the usefulness of storage-based networks even after node failures occur in the network. Second, we provide both distributed and centralized approaches to compute this metric. Finally, we evaluate the performance of our techniques for large-scale networks and demonstrate that our methods are scalable.

The remaining part of the paper is organized as follows. In Section 2, we first introduce the accessibility metric. In Section 3, we focus on a challenging scalability problem that restricts the calculation of accessibility in a large network. We elaborate on the results of our methods in Section 4. Finally, Section 5 concludes the paper with directions for future work.

2 Preliminaries

2.1 Emergency Incidence

Emergency incidence corresponds to the time instant when data loss is expected due to possible node failure. Therefore, it is defined as the state when a node's energy is below a certain amount. Equivalently, critically low energy levels of the neighbors of a particular node indicate that the node will be disconnected from the network. The definition can be further extended to include the energy levels of both the node and its neighbors in the same formulation as well. We consider the definition of emergency incidence as a design choice and deal with the more important problem of determining a more suitable location where data lifetime can be extended.

The lifetime of a sensor is directly related with its residual energy level, i.e., the more energy a node has, the longer period of time its data survives. It should be noted that we are not considering depletion of energy due to consumption in packet transmissions. Taking that into account is a much harder problem and we assume that node survival times are linearly proportional to the remaining energy levels. Furthermore, we assume that all nodes deplete their energy at the same constant rate.

2.2 A Measure of Node Ranking: Accessibility

Relocation of data to sensors that provide the best persistency requires achieving two goals. First, data should be available to a large number of nodes. Second, nodes that need the data should have long periods of connection (a path with alive nodes) to the node where the data is stored. In order to meet these requirements, a metric to measure how desirable a particular node is to relay and store data needs to be determined. In this paper, the concept of *accessibility* is proposed as such a measure, which takes into account the following issues.

- There is at least one path between two connected nodes. Obviously, if the nodes are not neighbors, this path consists of multiple hops. Furthermore, there can be more than one such multihop path between a pair of nodes.
- The node with the least energy on a multihop path constitutes the *bottleneck* node of that path. The energy of the bottleneck node is called the *bottleneck energy* of the path.
- The lifetime of the connection between two nodes is proportional to the bottleneck energy between them. If there are multiple paths, then the lifetime is proportional to the maximum bottleneck energy, i.e., *widest-path*.

Before discussing the *accessibility* formally, utilizing the above considerations we first define a term called *availability* of a node to another node:

Definition 1: *Availability ($\beta_{m,n}$) of a node m to another node n is the maximum bottleneck energy over all paths between m and n. In other words, it is the bottleneck energy of the widest-path (the path with the largest bottleneck energy) connecting m and n. If there is no path from m to n, $\beta_{m,n}$ = 0. By symmetry, $\beta_{m,n} = \beta_{n,m}$.*

An illustration of availability is given in Figure 1(a). $\beta_{B,A}$ is determined by considering the paths between A and B and the individual energy levels of these two nodes. Nodes A and B have 30 and 60 units of energy, respectively. Node B can be available to node A at most the period of time required to drain a battery that has 60 units of energy. Furthermore, by considering only the path between the two nodes through C, whose energy is 15, node B's availability to node A is limited by 15 units, hence $\beta_{B,A} = 15$. In addition, taking into account the alternative path through D, which is the widest-path, B's availability to A is limited by 20 instead of 15, which makes $\beta_{B,A} = 20$.

Utilizing the availability (β) discussion above, we now define the total *accessibility* of a node, which is a metric associated with a particular node.

Definition 2: *Accessibility ($\Psi(n)$) of a node n is the sum of availabilities of all nodes in the network to n.*

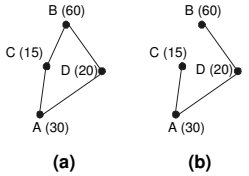


Figure 1. a) *Sample topology.* **b)** *Accessibility tree of node A.*

The general formula is given as follows:

$$\Psi(n) = \sum_{i=1}^{N} \beta_{i,n} \quad (1)$$

where $\beta_{i,n}$ is the availability of node i to node n, and N is the total number of nodes.

As an example, in Figure 1(a), the accessibility of node A is the sum of the availabilities of all other nodes to A. Hence, $\Psi(A) = \beta_{A,A} + \beta_{B,A} + \beta_{C,A} + \beta_{D,A} = 30 + 20 + 15 + 20 = 85$.

Note that the value of Ψ depends on two factors: the number of nodes that have connection to a node and the lifetime of individual connections. Apparently, increasing these quantities improves the accessibility of that node. Furthermore, since query generation events typically have exponential inter-arrival times, the expected number of queries that a node receives is proportional to the lifetime of individual connections. Hence, accessibility of a node is directly related to the total query traffic that it can receive while it is connected to the network. However, this paper does not focus on query generation events and instead we assume constant rate of query generation in the network.

In order to ease the presentation of the availability calculation, we present the following definition.

Definition 3: *Accessibility Tree of a node n is a tree such that the path on the tree from a node to n is a widest-path in the given topology. (Note that widest-paths are not necessarily unique).*

As an example, Figure 1(b) illustrates the accessibility tree of node A for the topology of Figure 1(a). A greedy algorithm similar to the Dijkstra's algorithm can be used to construct the Accessibility Tree for a given node n. Initially only node n is in the tree with $\beta_{n,n} = E(n)$, where $E(n)$ is the energy level of n, and all other nodes are assigned an availability value of 0. The links incident on n are explored to possibly increase the availability values of the neighboring nodes. At each step, the node that is not in the tree and has the maximum widest path estimate to n, is added to the computed tree, and the incident links are explored to adjust the availability values of neighboring nodes.

Theorem 1: *The greedy algorithm outlined above computes an Accessability Tree for a given node.*

Proof: Based on proof of Dijkstra's algorithm. ■

3 Methodology

Each node needs to advertise its bottleneck information to the connected nodes to enable topology discovery and computation of accessibility. Although the accessibility of a node represents the scope of data to the network, calculation of a particular node's accessibility to the entire network is a challenging problem. Note that, a brute-force technique would have a complexity of $O(n^2)$, which is obviously not an efficient approach for a large network; hence a scalable technique is needed.

One can think of applying a variety of approaches based on packet delays in order to reduce the complexity of accessibility calculation in a large network. For instance, a node can delay sending the received bottleneck information of a node to its neighbors so that the node does not have to resend an updated information. This technique can decrease the number of packet transmissions to some extent, but still requires network-wide packet transmissions.

As a remedy to the scalability problem in accessibility calculation, we propose an approach based on partitioning the network into a grid structure. Segmentation of the network can be achieved by distributing segment identification numbers (SID) to sensor nodes in pre-deployment time. Alternatively, sensors can estimate their coordinate information using a localization scheme [11], or obtain it from a location service. Since node localization is not the focus of this study, we elaborate on a network structure where nodes are assumed to know their locations. With this information, sensor nodes discover their individual SIDs. In our study, we consider rectangular topologies, however our proposed methods can be extended to other structures as well.

Figure 2 illustrates segmentation of a network into rectangular areas. With a segmented network, the determination of accessibility is performed within each segment and the gathered accessibility information is summarized over each segment separately in a distributed way. Each segment has a *segment-head* that coordinates the exchange of accessibility values within the segment. For that reason, nodes periodically report their local information, e.g., energy level, to the segment-head. Following the accessibility calculation in each segment, segment-heads communicate with each other to determine the relocation strategies.

Initially, one of the nodes with the highest energy is pre-selected as a temporary segment-head for each segment. When the network is deployed, the node with the largest accessibility within a segment is determined by the temporary segment-head. Then, this node is selected to be the permanent segment-head until its energy level becomes critically low. This choice reduces

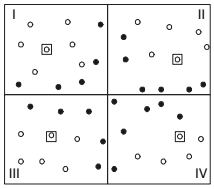


Figure 2. *Grid segmentation of a network. Darker nodes represent boundary nodes. Square-embraced nodes are the segment-heads.*

the number of segment-head changes when the energy of the segment-head itself becomes low. The change of the segment-head is performed centrally. When the segment-head decides that its energy level is critically low, it selects the node with the highest accessibility within the segment as the new segment-head. Then, a unicast "change of segment-head" (CSH) message is sent to the new head which then starts running the centralized algorithm. The CSH message includes the neighborhood information and most recent energy levels of the nodes in the segment. The new segment-head broadcasts a short "segment-head advertisement" packet (SHA) within the segment to inform segment nodes of its identity.

3.1 Accessibility calculation within a segment

In this section, we present a centralized approach to be computed at the segment-head, followed by a distributed algorithm for computing the accessibility of nodes within a segment.

1) Centralized accessibility calculation within a segment

In order to reduce the packet exchange traffic for frequent accessibility calculations by the segment nodes, we propose a centralized approach which utilizes existing spanning tree algorithms in the current literature.

Let G be a graph representing the current topology of a network. Each edge (u, v) in G is assigned a cost of $c(u, v) = \frac{1}{min\{E(u),E(v)\}}$, where $E(u)$ is the energy value of u. In addition, let T be a spanning tree that minimizes the total cost of the edges on the tree. Using Theorem 1, accessibility trees can be computed for each individual node. The following theorem shows that by carefully constructing a tree, all accessibility calculations can be confined to that tree, thus reducing the computation complexity.

Theorem 2: *If T is the least cost spanning tree of G, where $c(u, v) = \frac{1}{min\{E(u),E(v)\}}$, $\beta^T_{x,y} = \beta^G_{x,y} \forall x, \forall y$, where $\beta^T_{x,y}$ is the availability of x to y in T.*

Proof: For the sake of contradiction let us assume that for a specific pair of nodes (i, j), $\beta^T_{i,j} \neq \beta^G_{i,j}$. Let $<u_1, u_2, ...u_k>$ be a widest-path in G from i to j, where $u_1 = i$ and $u_k = j$. Some links on this path must be absent in the tree, otherwise $\beta^T_{x,y}$ will be equal to $\beta^G_{x,y}$. Consider such a link (u_l, u_{l+1}). There is exactly one path connecting u_l to u_{l+1} in T. The cost of all links on that path in T must be at most $c(u_l, u_{l+1})$, otherwise the edge (u_l, u_{l+1}) can be added to T and any other link on the resulting cycle can be removed causing a reduction in cost of T, thus contradicting its optimality. Using the definition of the cost of a link, the width (cost of bottleneck node) of the path in T connecting u_l and u_{l+1} must be at least $min\{E(u_l), E(u_{l+1})\}$. Note that by definition $\beta^G_{i,j} \leq min\{E(u_l), E(u_{l+1})\}$ $\forall l \in (1, k-1)$. Therefore, by concatenating the paths from u_l to u_{l+1}, $\forall l \in (1, k-1)$ on the tree, we get a trail on the tree whose width is at least $\beta^G_{i,j}$. Therefore the path from i to j on the tree, (obtained by removing the loops in the trail), must have a width at least $\beta^G_{i,j}$, i.e,. $\beta^T_{i,j} \geq \beta^G_{i,j}$. As T is a subgraph of G, $\beta^T_{i,j} \leq \beta^G_{i,j}$. Therefore, $\beta^T_{i,j} = \beta^G_{i,j}$, contradicting our assumption. ■

For computing the accessibility of a node v, perform a tree-traversal (breadth-first-search or depth-first-search) on T starting at v. During the traversal keep track of the widest-path from v to the visited node and the cumulative value of the availability to the visited nodes. After completing the traversal, the final value will represent the accessability of v. The above steps are repeated for each node to compute its accessability.

2) Distributed accessibility calculation within a segment

The distributed algorithm allows the nodes to compute their accessibility information periodically by broadcasting a special packet named as "Energy Advertisement Packet" (EAP) in a segment. An EAP packet is signed by the sender node's local ID within the segment, which is named as the "Packet Originator" (PO). As an EAP packet travels through the segment over a path, the current bottleneck of the path is stored in the packet's B_e (bottleneck energy) field.

The pseudo code is shown in Figures 3 and 4. The first time a node (i) receives an EAP from a PO, the node compares its own energy with B_e of EAP. If B_e is higher, the node updates B_e with its own energy, which means this node is the bottleneck of the path from PO to itself. Now, node i needs to forward this information to its neighbors. However, in order to reduce the number of EAPs created, node i delays the transmission of the packet. The first time i receives an EAP, a timer is started. If the node receives different EAPs with the same PO from its neighbors, i keeps updating B_e of this EAP before transmission. If the timer expires, i transmits the EAP to the neighbors. A new (*late*) EAP received after expiry of the timer is transmitted separately.

Note that the reception of EAP allows the node to

$E(i)$: energy of node i, B_e: bottleneck field of EAP,
SID_{own}: Segment ID of the segment that i is in,
SID_{PO}: Segment ID of the segment that PO is in.

At node i:
01 LOOP
02 wait until an EAP received
03 **if** (i is PO) OR (SID_{own} != SID_{PO})
04 discard packet
05 **continue**
06 **if** (first EAP from this PO)
07 $\beta_{PO,i} \leftarrow min(B_e, E(i))$
08 start *timer*$_{PO}$ // a unique timer for PO
09 **else**
10 **if** ($\beta_{PO,i} < min(B_e, E(i))$)
11 $\beta_{PO,i} \leftarrow min(B_e, E(i))$
12 **if** (*timer*$_{PO}$ expired) // late EAP
13 *sendEAP (PO)* // call sendEAP function
14 **else**
15 discard packet

Figure 3. *Distributed Algorithm*

sendEAP (PO)
01 create an EAP
02 B_e of EAP $\leftarrow \beta_{PO,i}$
03 send EAP to all neighbors

Figure 4. *sendEAP Function*

learn $\beta_{PO,i}$. In this way, i learns all the bottleneck information from the connected nodes and therefore can calculate its own accessibility. The algorithm of distributed accessibility calculation of a node n, given in Figure 3, utilizes the *sendEAP* function in Figure 4. Note that if a timer expires for a PO, we assume that *sendEAP* is automatically called. In addition, note that the given algorithm is loop-free which can be verified based on the packet discarding actions at the lines 4 and 15.

3.2 Diffusion of accessibility through segments

The previous section discussed the accessibility calculation within segments. From a global point of view, the part of the accessibility tree of a node that spreads through the other segments has not been considered in the calculation yet. Now, we present a technique for diffusing the accessibility values among the segments.

We define the nodes within a segment that have neighbors belonging to other segments as *Boundary Nodes* (BN), e.g., darker nodes in Figure 2. Each BN has the duty of summarizing the accessibility information of its segment to the nodes in the neighboring segments. Figure 5 presents the Neighbor Segment Summarization (*NSS*) technique. In Figure 5(a), for the sake of illustration the whole topology is partitioned into two segments. Nodes A_1, A_2, B_1, B_2 are the BNs.

Note that BNs B_1 and B_2 have already calculated their accessibility within segment II as explained in the previous section. It is important to note that there can be more than one BN (which are the neighbors of, say, B_1) in segment I. Therefore, B_1 should choose a neighbor BN with the largest energy, say A_1, to pass the summarized information. However, if B_1 and B_2 would directly pass this information to A_1 and A_2, respectively, the availability values of nodes that are common in accessibility trees of B_1 and B_2 (e.g., node C in Figure 5(a)) would be passed twice. Hence, the nodes in segment I (such as n) would compute an accessibility value which is much higher than its global value. In order to avoid the problem, B_1 and B_2 should construct *disjoint accessibility trees* within segment II. Then, each BN should calculate a partial accessibility value by only considering the nodes in its own disjoint tree and pass that to segment I. In Figure 5(a), the number of nodes in the tree of B_1 (B_2) is n_1 (n_2) and the partial accessibility value is a_1 (a_2).

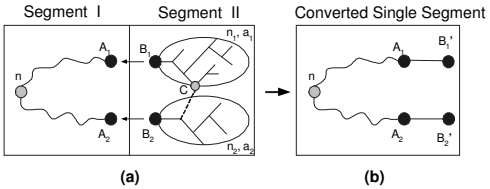


Figure 5. *Neighbor Segment Summarization (NSS) Technique*

While constructing these disjoint trees, to avoid additional packet overhead, a special field in EAPs is set by BNs. This enables intermediate nodes to distinguish the EAPs of BNs and take actions in addition to the regular EAP handling. Upon receiving EAPs from different BNs, a non-boundary node adds itself to the tree of the BN for which it has the largest availability so that the trees will be disjoint. For example, node C in Figure 5(a) receives EAPs from both B_1 and B_2. If a wider-path exists from B_1 to C (compared to the paths from B_2 to C), node C is added to the disjoint tree of B_1.

The process of accessibility summarization (from segment II to I) is analogous to having two more nodes (B_1', B_2') as shown in Figure 5(b), whose energies are $\frac{a_1}{n_1}$ and $\frac{a_2}{n_2}$. These energy values are passed to A_1 and A_2, respectively, who transmit them within segment I. As these values travel towards node n, the availability $\beta_{B_1',n}$ will be updated. Then, node n adds $n_1 \times \beta_{B_1',n}$ and $n_2 \times \beta_{B_2',n}$ to its accessibility since B_1' and B_2' represent n_1 and n_2 number of nodes, respectively.

3.3 Incorporating all segments into the calculation

The *NSS* technique outlined in the previous section is the basis for accessibility calculation in networks with more than two segments. In this section, *NSS* is generalized to calculate the network-wide accessibility.

The structure of a network with a high number of segments can be simplified by a grid network model. That is, each segment is considered as a single node and boundaries between neighbor segments are represented by directed links as given in Figure 6.

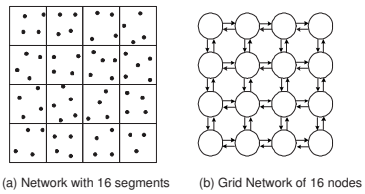


Figure 6. *Grid Network Model*

Recall that in the *NSS* technique there are more than one pair of neighbor BNs that are interacting. However, the model in this section further simplifies the approach by considering only a single representative *summarization value* between the segments. Assume that there are m boundary nodes $B_1, \ldots, B_m$ in Segment II in Figure 5. Furthermore, recall that the new nodes in Segment I formed by summarizing nodes of Segment II using *NSS* as shown in Figure 5(b) are denoted by B'_i. Denoting the energy of a boundary node A_i by $E(A_i)$, the availability of B'_i to A_i is $\beta_{B'_i,A_i} = \min(E(A_i), \frac{a_i}{n_i})$, where n_i denotes the number of nodes that B'_i represents in Segment II. Using the values $\beta_{B'_i,A_i}$ for $i = 1, \ldots, m$, we discuss three main schemes to calculate a single summarization value from Segment II to I as follows:

- **Scheme 1** Average of partial availabilities:

$$\overline{E_{II,I}} = \frac{1}{m} \sum_{i=1}^{m} \beta_{B'_i,A_i} \quad (2)$$

- **Scheme 2** Maximum of partial availabilities:

$$\overline{E_{II,I}} = \max(\beta_{B'_i,A_i}),\ i = 1, \ldots, m \quad (3)$$

- **Scheme 3** Weighted sum of partial availabilities:

$$\overline{E_{II,I}} = \frac{\sum_{i=1}^{m}(n_i \times \beta_{B'_i,A_i})}{\sum_{i=1}^{m} n_i} \quad (4)$$

Scheme 1 represents the summarization value from segment II to I (as shown in Figure 5), $\overline{E_{II,I}}$, by the average partial availabilities, while Scheme 2 uses the maximum value. Scheme 3, on the other hand, considers the number of nodes in individual disjoint trees in segment II and computes a weighted average.

The segment-heads keep the total number of nodes in the disjoint trees of the neighbor segments, e.g., from segment II to I: $n_{II,I} = \sum_{i=1}^{m} n_i$. After segment I determines the summarization value from neighbor segment II, i.e., $\overline{E_{II,I}}$, and the total number of nodes $n_{II,I}$, these values are distributed to all other segment-heads. Each segment-head stores the summarization values, $\overline{E_{i,j}}$, and also $n_{i,j}$ values for all neighboring pairs of segments i and j. These values are used to compute the availabilities of individual segments to other segments, which is explained next. Note that for two non-neighbor segments i and j; $\overline{E_{i,j}}$, $\overline{E_{j,i}}$, $n_{i,j}$, $n_{j,i}$ are all equal to zero.

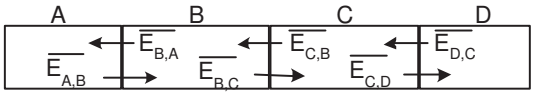


Figure 7. *Calculation of the segment availability on a path of segments.*

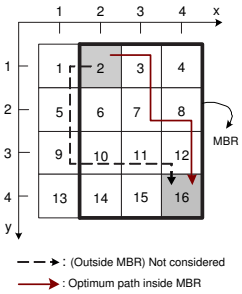


Figure 8. *Widest-path of segments within an MBR of two non-neighbor segments.*

1) Calculation of inter-segment availabilities

Utilizing the above $\overline{E_{i,j}}$ and $n_{i,j}$ values, each segment-head computes the widest-path from all other segments to its own segment (on the grid structure), which is similar to the availability calculation method within a segment. As an example, consider Figure 7, where there are four segments (A, B, C, and D). The availability of segment D to A is, $\beta_{D,A}$ = $min(\overline{E_{B,A}}, \overline{E_{C,B}}, \overline{E_{D,C}})$.

In order to reduce the complexity of the widest-path computation, only the paths within the minimum bounding rectangle (MBR) of two non-neighbor segments are considered. For instance, in Figure 8, the MBR of segments 2 and 16 is highlighted. Note that a path which is partially outside the MBR may exist. However, the probability of such a path being the widest-path is small. Note that a more complex algorithm can consider all possible paths. However, for large networks with a large number of segments, such algorithms can be computationally costly. Hence, MBR is proposed as a simple and effective solution to determine the widest-path of segments.

Considering the MBRs, each segment-head computes the availability of other segments to its own segment (S) and keeps these values in a matrix called M. For a grid-network with $4 \times 4 = 16$ segments, M is a 4×4 matrix. The entry $m_{i,j}$ of matrix M is the availability of segment $SID(i, j)$ to S. By definition, the entry corresponding to S in matrix M is zero (segment availability to itself). This is due to the fact that the accessibility values within the segment are already calculated.

Furthermore, the number of nodes that contribute to the availability of segment S in each segment $SID(i, j)$ is stored in the entry $t_{i,j}$ of another matrix T. For instance, in Figure 8, from segment $SID(1, 2) = 2$ to

segment 16, we have $t_{1,2} = n_{2,3}$. Note that segment 3 is neighbor of segment 2 on the path towards segment 16.

2) Finalizing accessibility calculation

Now, we discuss how the inter-segment availabilities are contributed to the accessibility values of individual nodes within a segment.

Method 1: *Summation of Segment Availabilities*

One simple approach is to add the inter-segment availabilities that are found in matrix M, scaled by the number of nodes in matrix T, to the locally computed node accessibility values. This gives an estimation of the global accessibility values in a segment as follows:

$$\Psi(n) = \sum_{i=1}^{N_x} \sum_{j=1}^{N_y} t_{i,j} m_{i,j} + \Psi_{local}(n), \quad (5)$$

where $\Psi_{local}(n)$ is the local accessibility of a node n in the segment and $\Psi(n)$ is the estimated global accessibility of node n. Although this scheme provides a fairly simple estimation, the segment availabilities in matrix M should be compared with the energy levels of nodes within S (which is taken care of by Method 2 below). If the intermediate nodes within S have lower energies than a $m_{i,j}$ value in M, the availability of the corresponding segment, i.e., SID(i,j), should be updated with these lower energy values. Hence, the summation approach outlined by Equation 5 provides an upper bound on the estimated node accessibility values and this is validated in the experimental results.

Method 2: *Segment Grouping and Average Availabilities*

In Method 2, all segments are included in different groups according to the neighbor segment of S that is included in the widest-path towards S. Since S has at most four neighbor segments, at most four groups are formed. We can denote these groups as S_k where $k \in \{1, 2, 3, 4\}$. Figure 9(a) shows such a grouping for segment $S = 6$, and Figure 9(b) is a representation of these groups. For instance, the widest-paths of segments 1, 2, 3 to segment 6 pass through the neighbor segment 2 so that they form a group which is shown by number 2 in Figure 9(b) and group S_1 in Figure 9(c).

The sum of the availabilities of segments in each group S_k is computed using the matrices M and T. This sum is denoted by λ_k. Furthermore, the total number of nodes that contribute to the availability of the segments in each group S_k is denoted by $\mathfrak{n}_k$. Then, similar to *NSS* technique, each group S_k is treated as a single node (with energy $E_k = \frac{\lambda_k}{\mathfrak{n}_k}$) and included as a new node in segment S. This is shown in Figure 9(c). This provides updating the segment availabilities to individual nodes

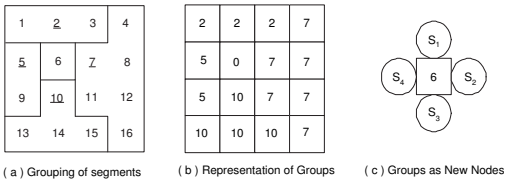


Figure 9. *Grouping of segments according to SIDs of neighbor segments. Segment 6 treats other segments as single nodes after grouping them (c). Neighbor segments are underlined in (a).*

in S according to the local node energy levels, and more accurate global accessibility estimation is achieved compared to Method 1.

As the availability information of a new node S_k propagates towards intermediate nodes, the availability should be scaled by $\mathfrak{n}_k$ since S_k represents $\mathfrak{n}_k$ number of nodes in its group and updated by the intermediate nodes similar to the *NSS* approach presented in Section 3.2.

4 Experimental Results

4.1 Experimental Setup

The network topologies used in the experimental evaluations are formed by uniform distribution of N sensors with density σ in a 400×400 unit^2 field. Each topology is divided into 16 segments each with dimensions LxL, where $L = 100$ units ($\sigma = \frac{N}{L^2}$). Sensors have disk-shaped communication ranges with radius R. Each node is assumed to receive all packets within its communication range. Specific MAC layer protocols are not considered. The focus of evaluation is on network layer communication. When initializing the network, random energy levels are assigned to nodes with a maximum value of E_{max}. We choose $E_{max} = 100$ units in our simulations.

4.2 Global vs. Estimated Accessibility

This section presents our observations on how globally obtained values compare with the results of our methods. First, the comparison is provided for the diffusion of accessibility between two segments. Then, results for a network with 16 segments are demonstrated.

1) Two Segments

In this section, a network of size Lx$2L$, which has two segments, is used for evaluation. Figure 10 presents the comparison of global accessibility to the estimated accessibility found by the *NSS* technique. As shown in Figure 5 in Section 3.2, the accessibility values in segment I are calculated using the local summary of segment II. Then, the accessibility value for each node in segment I is calculated over the entire network globally (without segmentation). Figure 10 validates that the *NSS* technique estimates global results accurately.

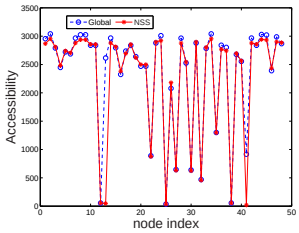


Figure 10. *Comparison of accessibility: Global vs.* NSS *technique.*

In Figure 11, the effects of node density and communication range on accessibility are demonstrated. Various densities are obtained by changing the number of nodes in the topology ($N = 10, 20, \ldots, 100$). Note that a 10-node network has a node density $\sigma = 0.0005$ $nodes/unit^2$ and a 50-node network has a node density $\sigma = 0.0025$ $nodes/unit^2$. Furthermore, since neighborhood relationship and connectivity patterns are affected by the communication range R, we use various values of R to observe its effects on accessibility values ($R = 30, 40, \ldots 80$). Due to the random sensor deployment in each scenario, average results are obtained for 1000 topologies of size Lx$2L$ with two segments.

The effects of R and σ on the accessibility calculation lead to various levels of accuracy, which is illustrated in Figure 11. The increase in node density σ suggests an initial increase in error, however, the amount of this error then stabilizes. *The stabilization shows that our scheme is no more affected by the increase in the number of nodes, hence it is scalable with the change in node density.* In general, the stabilization in percent error occurs at larger node densities for scenarios where a smaller R is used. Moreover, the stabilized percent error value is smaller for larger R. *These results show that the scheme is more scalable for networks that have larger communication ranges.*

2) Higher Number of Segments

In this section, we investigated the accuracy of the three accessibility diffusion schemes as explained in Section 3.3, namely the "Mean", "Weighted", and "Max" schemes. Furthermore, the results of the two methods presented in Section 3.3 for using inter-segment availabilities in node accessibility calculations are provided.

Figure 12(a) presents the results of Mean Scheme for node accessibility values. The node indices in a randomly chosen segment is shown in the x-axis. As can be observed, using the mean of partial availabilities to represent inter-segment availabilities produces smaller accessibility values for most of the nodes in the segment. This suggests that such nodes in fact have wider paths

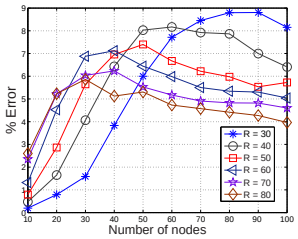


Figure 11. *The effect of R and σ for 2 segments.*

towards other segments. Hence, the Mean Scheme underestimates the true global accessibility values.

In Figure 12(b), accessibility results for the Max Scheme are presented. In contrast to the Mean Scheme, the computed values are noticeably larger than the true global node accessibility for most of the nodes. In fact, nodes may not have paths to the boundaries or the paths towards individual nodes may have smaller bottlenecks. Hence, choosing the boundary node with the largest inter-segment availability over-estimates inter-segment availabilities.

The results for the Weighted Scheme are presented in Figure 12(c). This figure illustrates higher accuracy in computed accessibility values compared to the Mean and Max Schemes. This improvement can be attributed to the fact that the Weighted Scheme considers the disjoint partial trees of BNs in a boundary and assigns weights to their availabilities proportional to the number of nodes they represent.

Figures 12(a), 12(b), and 12(c) also show the performance of Method 1, as presented in Section 3.3, which adds the segment availabilities to local node accessibility values to compute global accessibility of nodes. As can be observed in these figures, this method provides upper bound values for the Mean, Max, and Weighted schemes, which is mentioned in Section 3.3-2. Since segment availabilities are updated and can only be reduced by the path bottlenecks within segments, a plain summation of segment availabilities ignores such effects and provides an upper bound value.

Figure 13 illustrates the average percent error in accessibility calculations for all nodes over 16 segments. Average results of 20 random topologies are provided. The number of nodes in the simulations is varied from

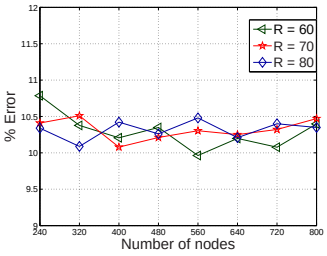
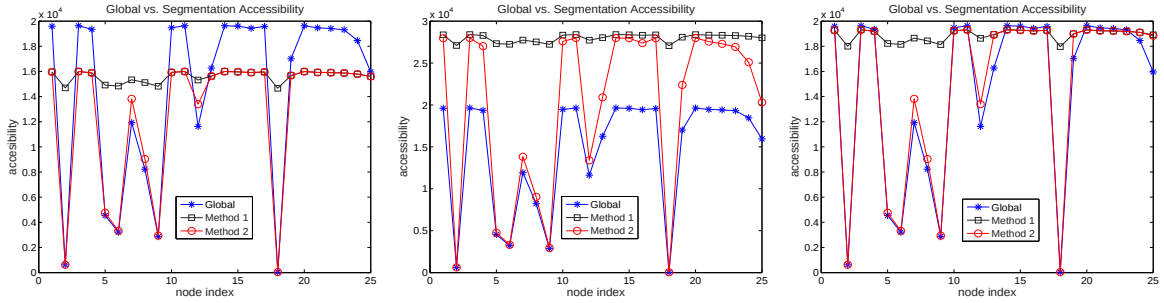


Figure 13. *The effect of R and σ for 16 segments.*



(a) *Scheme 1: Mean* (b) *Scheme 2: Max* (c) *Scheme 3: Weighted*

Figure 12. *Global vs. estimated accessibilities in a segment for a network with* 16 *segments.*

$N = 240$ to $N = 800$ with increments of 80. These values correspond to the ranges $N = 30, 40, \ldots, 100$ for a network with two segments. Note that this choice of the number of nodes for 16 segments keeps the node density and the network connectivity patterns similar to the two-segment-topologies. Furthermore, communication range values are chosen as $R = 60, 70, 80$. Figure 13 demonstrates that error around 10 % occurs regardless of the increase in the number of nodes for all communication ranges. Despite the calculation of accessibility over 16 segments instead of 2, the percent error is only slightly increased from 8% to 10%. This shows that the proposed technique of summarizing segment availabilities is a scalable method despite the increase in the number of segments and node density.

5 Conclusion

Due to fast depletion of energy resources in a sensor network, stored data run the risk of being inaccessible to the rest of the network. In this study, we introduce a new metric called "accessibility", which is a measure of the total amount of time that nodes have access to the data. Instead of data replication, we propose to relocate the data of sensors with critically low energy resources to other nodes in the network with higher accessibility values. Since accessibility calculation requires global information about the network, scalability is a challenging issue when large networks are considered. We address this problem by partitioning the network into segments and provide both centralized and distributed approaches to calculate accessibility within segments. Furthermore, we propose an effective method to summarize the information gathered from network segments to compute network-wide node accessibility values.

As part of future work, we are investigating techniques for data relocation while considering additional practical constraints such as limited storage space, and failure rate of nodes due to reasons other than battery drainage. We also plan to extend our work to consider data moving costs and a more detailed modeling of data load due to network's data traffic. Furthermore, a more general case in which nodes consume their energy in unequal rates is a topic of future research.

References

[1] Y. Diao, D. Ganesan, G. Mathur, and P. J. Shenoy. Rethinking data management for storage-centric sensor networks. In *CIDR, Asilomar, CA, Jan, 2007.*

[2] T. Hara. Effective replica allocation in ad hoc networks for improving data accessibility. In *IEEE INFOCOM, Anchorage, Alaska, USA, Vol.3, pp.1568-1576, Apr, 2001.*

[3] T. Hara, N. Murakami, and S. Nishio. Replica allocation for correlated data items in ad-hoc sensor networks. In *ACM SIGMOD, Vol.33, No.1, pp.38-43, Mar, 2004.*

[4] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan. Energy-efficient communication protocol for wireless microsensor networks. In *HICSS, Maui, Hawaii, Jan, 2000.*

[5] K. Kalpakis, K. Dasgupta, and P. Namjoshi. Efficient algorithms for maximum lifetime data gathering and aggregation in wireless sensor networks. In *Computer Networks, pp.497-716, 2003.*

[6] A. Kamra, J. Feldman, V. Misra, and D. Rubenstein. Data persistence in sensor networks: Towards optimal encoding for data recovery in partial network failures. In *Workshop on Mathematical Performance Modeling and Analysis, June, 2005.*

[7] A. Kamra, J. Feldman, V. Misra, and D. Rubenstein. Growth codes: Maximizing sensor network data persistence. In *ACM SIGCOMM, Pisa, Italy, Sep, 2006.*

[8] S. Lindsey, C. Raghavendra, and K. M. Sivalingham. Data gathering algorithms in sensor networks using energy metrics. In *IEEE Transactions on Parallel and Distributed Systems, Vol.13, No.9, pp.924-935, Sep, 2002.*

[9] N. Sadagopan and B. Krishnamachari. Maximizing data extraction in energy-limited sensor networks. In *IEEE INFOCOM, Hong Kong, Mar, 2004.*

[10] A. Sankar and Z. Liu. Maximum lifetime routing in wireless ad hoc networks. In *IEEE INFOCOM, Hong Kong, Mar, 2004.*

[11] C. Savarese, J. Rabaey, and K. Langendoen. Robust positioning algorithms for distributed ad-hoc wireless sensor networks. In *USENIX Annual Technical Conference, pp.317-327, 2002.*

[12] G. Zussman and A. Segall. Energy efficient routing in ad hoc disaster recovery networks. In *IEEE INFOCOM, San Francisco, Apr, 2003.*