Improving the Lifetime of Sensor Networks via Intelligent Selection of Data Aggregation Trees[1,2]

Koustuv Dasgupta, Konstantinos Kalpakis, and Parag Namjoshi

Abstract

We consider a network of energy–constrained sensors that are deployed over a region. Each sensor periodically produces information as it monitors its vicinity. The basic operation in such a network is the systematic gathering and transmission of sensed data to a base station for further processing. During data gathering, sensors have the ability to perform in–network aggregation or fusion of data packets enroute to the base station. The lifetime of such a sensor system is the time during which we can gather information from all the sensors to the base station. A key challenge in data gathering is to maximize the system lifetime, given the energy constraints.

The Maximum Lifetime Data Aggregation (MLDA) problem is, given the location of sensors and their available energies together with the location of the base station, find an efficient manner in which data should be collected and aggregated from all sensors and transmitted to the base station, such that the system lifetime is maximized.

In this paper, we present an approximation scheme for solving the maximum lifetime data aggregation problem in sensor networks. Our scheme is based on intelligent selection of data aggregation trees from a candidate set of trees. Further, we describe two simple approaches for generating candidate sets of data aggregation trees, leading to two efficient approximation algorithms for the MLDA problem. Finally, we provide experimental results which demonstrate the effectiveness of the proposed algorithms.

Keywords: sensor networks, data gathering, data aggregation, energy–efficient protocols, lifetime.

1 Introduction

Recent advances in micro–sensor (MEMS) technology and low–power analog/digital electronics, have led to the development of distributed, wireless networks of sensor devices [9, 15, 16]. Sensor networks of the future are envisioned to consist of hundreds of inexpensive nodes, that can be readily deployed in physical environments to collect useful information (e.g. seismic, acoustic, medical and surveillance data) in a robust and autonomous manner. However, there are several obstacles that need to be overcome before this vision becomes a reality [7]. Such obstacles arise from the limited energy, computing capabilities and communication resources available to the sensors.

We consider a system of sensor nodes that are homogeneous and highly energy–constrained. Further, replenishing energy via replacing batteries on hundreds of nodes (in possibly harsh terrains) is infeasible. The basic operation in such a system is the systematic gathering of sensed data to be eventually transmitted to a base station for processing. The key challenge in such data gathering is conserving the sensor energies, so as to maximize their lifetime. To this end, there are several power–aware routing protocols for wireless ad hoc networks discussed in the literature [3, 4, 18]. In the context of sensor networks, LEACH [6] proposes a clustering–based protocol for transmitting data to the base station. The main features include local co–ordination for cluster formation among sensors, randomized rotation of cluster heads for improved energy utilization, and local data compression to reduce global communication. In related work, Bhardwaj et al [2]

[1]Supported in part by NASA under Cooperative Agreement NCC5–315 and Contract NAS5-32337, and by NSF under grant IRI–9729495.

[2]Computer Science and Electrical Engineering Department, University of Maryland Baltimore County, 1000 Hilltop Circle, Baltimore, MD 21250. E-mail: {kalpakis,dasgupta,nam1}@csee.umbc.edu. Phone: 410-455-3143. Fax: 410-455-3969. **Please send all correspondence to Prof. Kalpakis.**

derive upper bounds on the lifetime of a sensor network that collects data from a specified region using some energy–constrained nodes.

Data fusion or aggregation has emerged as a useful paradigm in sensor networks. The key idea is to combine data from different sensors to eliminate redundant transmissions, and provide a rich, multi–dimensional view of the environment being monitored. Krishnamachari et al [11] argue that this paradigm shifts the focus from address–centric approaches (finding routes between pairs of end nodes) to a more data–centric approach (finding routes from multiple sources to a destination that allows in–network consolidation of data). Madden et al [14] describe the TinyOS operating system that can be used by an ad-hoc network of sensors to locate each other and route data. The authors discuss the implementation of five basic database aggregates, i.e. COUNT, MIN, MAX, SUM, and AVERAGE, based on the TinyOS platform and demonstrate that such a generic approach for aggregation leads to significant power (energy) savings. The focus of the work in [14] is on a class of aggregation predicates that is particularly well suited to the in-network regime. Such aggregates can be expressed as an aggregate function f over the sets a and b, such that $f(a \cup b) = g(f(a), f(b))$. Directed diffusion [8] is based on a network of nodes that can co–ordinate to perform distributed sensing of an environmental phenomenon. Such an approach achieves significant energy savings when intermediate nodes aggregate responses to queries. The SPIN protocol [7] uses meta–data negotiations between sensors to eliminate the transmission of redundant data . In PEGASIS [12], sensors form chains so that each node transmits and receives from a nearby neighbor. Gathered data moves from node to node, gets aggregated and is eventually transmitted to the base station. Nodes take turns to transmit so that the average energy spent by each node gets reduced. In [13], the authors propose a hierarchical scheme based on PEGASIS that reduces the average energy and delay incurred in gathering the sensed data.

Kalpakis, Dasgupta and Namjoshi [10] described the Maximum Lifetime Data Aggregation (MLDA) algorithm for data gathering and aggregation in sensor networks. They introduce the notion of aggregation trees, which are trees that indicate how values from the various sensors are gathered, aggregated, and transmitted to the base station. The proposed method significantly outperforms existing protocols in terms of system lifetime, and delivers system lifetime that is very close to the optimal system lifetime. However, the complexity of the MLDA algorithm renders it impractical for large sensor networks.

In this paper, we present an approximate scheme for the maximum lifetime data aggregation problem in sensor networks, when the sensors are allowed to perform in–network aggregation of data packets. Our approximation scheme utilizes intelligent selection of aggregation trees from a candidate set of aggregation trees. Then, we describe two simple approaches for generating suitable candidate sets of aggregation trees, leading to the $\mathcal{A}$–LRS and $\mathcal{A}$–R–LRS algorithms, which are computationally efficient. The $\mathcal{A}$–LRS algorithm uses, as candidate set, the trees in [13], while the $\mathcal{A}$–R–LRS algorithm augments the candidate set with randomized permutations of the sensors. We present experimental results demonstrating that the two proposed algorithms attain significant improvements in system lifetime when compared to existing protocols. For example, the $\mathcal{A}$–R–LRS algorithm provides an improvement of over 50% in the lifetime for network sizes ≥ 100, with respect to existing protocols; moreover, for network sizes ≤ 100 it provides system lifetime within 2.5% of the optimal system lifetime. Furthermore, we introduce a simple "limited aggregation" model for the MLDA problem, and shed some light on the effect of limited aggregation on the system lifetime. Experimental results show that the $\mathcal{A}$–R–LRS algorithm delivers superior performance even in the limited aggregation case.

The rest of the paper is organized as follows. In section 2, we formulate the data gathering problem in sensor networks. We describe our approximation scheme and the $\mathcal{A}$–LRS and $\mathcal{A}$–R–LRS algorithms for solving the problem in section 3. In section 4, we present detailed experimental results and comparison of the proposed scheme/algorithms with existing algorithms, and finally in section 5 we conclude the paper.

2 The Data Gathering Problem

Consider a network of n sensor nodes $1, 2, \ldots, n$ and a base station node t distributed over a region. The locations of the sensors and the base station are fixed and known a priori. Each sensor produces some information as it monitors its vicinity. We assume that each sensor generates one data packet per time unit to be transmitted to the base station. For simplicity, we refer to each time unit as a *round*. We assume that all data packets have size k bits. The information from all the sensors needs to be gathered at each round and sent to the base station for processing. We assume that each sensor has the ability to transmit its packet to any other sensor in the network or directly to the base station. Further, each sensor i has a battery with finite, non-replenishable energy $\mathcal{E}_i$. Let $\mathcal{E}$ be an $n \times 1$ vector with its ith entry equal to $\mathcal{E}_i$, $i = 1, 2, \ldots, n$. Whenever a sensor transmits or receives a data packet it consumes some energy from its battery. The base station has an unlimited amount of energy available to it. Our energy model for the sensors is based on the first order radio model described in [6]. A sensor consumes $\epsilon_{elec} = 50nJ/bit$ to run the transmitter or receiver circuitry and $\epsilon_{amp} = 100pJ/bit/m^2$ for the transmitter amplifier. Thus, the energy consumed by a sensor i in receiving a k-bit data packet is given by,

$$Rx_i = \epsilon_{elec} \times k \tag{1}$$

while the energy consumed in transmitting a data packet to sensor (or base station) j is given by,

$$Tx_{i,j} = \epsilon_{elec} \times k + \epsilon_{amp} \times d_{i,j}^2 \times k \tag{2}$$

where $d_{i,j}$ is the distance between nodes i and j.

During the process of data gathering, sensors have the ability to perform fusion (or aggregation) of data packets coming from different sources enroute to the base station. Such aggregation can be performed when data from different sensors are highly correlated and leads to significant energy savings due to reduction in the number and size of data transmissions. Note that, most of the existing data gathering schemes ([6, 10, 12]) make the assumption that an intermediate sensor can aggregate multiple incoming packets into a single outgoing packet.

We define the *lifetime* T of the system to be the time until the first sensor is drained of its energy. A *data gathering schedule* specifies, for each round, how the data packets from all the sensors are collected and transmitted to the base station. For brevity, we refer to a data gathering schedule simply as a schedule. The lifetime of a schedule equals the lifetime of the system under that schedule. Our objective is to find a schedule with maximum lifetime T.

The Maximum Lifetime Data Aggregation (MLDA) problem: Given a collection of sensors and a base station, together with their locations and the initial energy of each sensor, find a data gathering schedule with maximum lifetime, where sensors are permitted to aggregate incoming data packets.

Observe that, a schedule can be thought of as a collection of T (not necessarily distinct) directed trees, each rooted at the base station and spanning all the sensors, i.e. a schedule has one tree for each round. Each such tree specifies how data packets are gathered and transmitted to the base station. We call these trees *data aggregation trees* (or simply aggregation trees). An aggregation tree may be used for one or more rounds; we define the lifetime of an aggregation tree to be the number of rounds that the aggregation tree is used in a schedule. The MLDA problem can then be restated as finding a set of aggregation trees along with their lifetimes, such that the sum of their lifetimes (i.e. the system lifetime) is maximized. [3]

3 Finding a Maximum Lifetime Data Gathering Schedule

3.1 The MLDA algorithm

Kalpakis, Dasgupta, and Namjoshi [10] proposed a polynomial–time approximation algorithm for the maximum lifetime data aggregation problem and reported on its performance. For the sake of completeness, we next provide a brief description of the MLDA algorithm.

Consider a schedule $\mathcal{S}$ with lifetime T rounds. Let $f_{i,j}$ be the total number of packets that node i (a sensor) transmits to node j (a sensor or base station) in $\mathcal{S}$. Since any valid schedule must respect the energy constraints of the sensors, it follows that for each sensor $i = 1, 2, \ldots, n$,

$$\sum_{j=1}^{n+1} f_{i,j} \cdot Tx_{i,j} + \sum_{j=1}^{n} f_{j,i} \cdot Rx_i \leq \mathcal{E}_i. \tag{3}$$

[3] It can be shown that the integral MLDA problem is NP–complete, by reduction from the Hamiltonian Path problem.

Table 1: Integer program for finding an optimal admissible flow network for the MLDA problem.

Objective :

maximize T (4)

Constraints :

$$\sum_{j=1}^{n+1} f_{i,j} \cdot T\mathrm{x}_{i,j} + \sum_{j=1}^{n} f_{j,i} \cdot R\mathrm{x}_i \leq \mathcal{E}_i \tag{5}$$

$$\sum_{j=1}^{n} \pi_{j,i}^{(k)} = \sum_{j=1}^{n+1} \pi_{i,j}^{(k)}, \quad \forall i = 1,2,\ldots,n \text{ and } i \neq k \tag{6}$$

$$T + \sum_{j=1}^{n} \pi_{j,k}^{(k)} = \sum_{j=1}^{n+1} \pi_{k,j}^{(k)} \tag{7}$$

$$0 \leq \pi_{i,j}^{(k)} \leq f_{i,j}, \quad \forall i = 1,2,\ldots,n \text{ and } \forall j = 1,2,\ldots,n+1 \tag{8}$$

$$\sum_{i=1}^{n} \pi_{i,n+1}^{(k)} = T \tag{9}$$

where $k = 1,2,\ldots,n$ and all variables T, $f_{i,j}$, and $\pi_{i,j}^{(k)}$ are required to be non-negative integers.

Recall that each sensor, for each one of the T rounds, generates one data packet that needs to be collected, possibly aggregated, and eventually transmitted to the base station. Furthermore, in MLDA, any sensor can aggregate multiple incoming packets into a single outgoing packet. A schedule $\mathcal{S}$ induces a flow network $G = (V, E)$, i.e. a directed graph having as nodes all the sensors and the base station, and having edges (i, j) with capacity $f_{i,j}$ whenever $f_{i,j} > 0$. The following theorem can then be proved.

Theorem 1 *Let $\mathcal{S}$ be a schedule and let G be the flow network induced by $\mathcal{S}$. Then, $\mathcal{S}$ has lifetime T if and only if for each sensor s, the maximum flow from s to the base station t in G is $\geq T$.*

Proof. Refer to [10] for details. ■

Thus, a necessary condition for a schedule to have lifetime T is that each node in the induced flow network can push flow T to the base station t. Stated otherwise, each sensor s must have a minimum $s - t$ cut of capacity (size) $\geq T$ to the base station.

Next, we consider the problem of finding a flow network G with maximum T, that allows each sensor to push flow T to the base station, while respecting the energy constraints in (5) at all the sensors. We call such a flow network G an *admissible* flow network with lifetime T. An admissible flow network with maximum lifetime is called an *optimal admissible* flow network.

An optimal admissible flow network can be found using an integer program with linear constraints, given in Table 1. The integer program, in addition to the variables for the lifetime T and the edge capacities $f_{i,j}$, uses the following variables: for each sensor $k = 1, 2, \ldots, n$, $\pi_{i,j}^{(k)}$ is a flow variable indicating the flow that k sends to the base station t over the edge (i, j). The integer program computes the maximum system lifetime T subject to the energy constraint (5) and the additional linear constraints (6)–(9) for each sensor. For each sensor $k = 1, 2, \ldots, n$, constraints (6) and (7) enforce the flow conservation principle at the sensor; constraint (9) ensures that T flow from sensor k reaches the base station; and constraint (8) ensures that the capacity constraints on the edges of the flow network are respected. Moreover, constraint (5) is used to guarantee that the edge capacities of the flow network respect the sensor's available energy. Finally, for the integer program, all variables are required to take non–negative integer values. The linear relaxation of the above integer program, i.e. when all the variables are allowed to take fractional values,

Table 2: Integer program for the $\mathcal{A}$–restricted MLDA problem.

Objective :

$$\text{maximize} \quad \sum_{A_i \in \mathcal{A}} \lambda_i \tag{10}$$

Energy constraint for the sensors:

$$\sum_{A_i \in \mathcal{A}} \lambda_i \cdot E(A_i) \leq \mathcal{E}, \tag{11}$$

where all λ_i's are required to be non-negative integers.

can be computed in polynomial–time. Then, we can obtain a very good approximation for the optimal admissible flow network by first fixing the edge capacities to the floor of their values obtained from the linear relaxation so that the energy constrains are all satisfied; and then solving the linear program (4) subject to constraints (6)–(9) without requiring anymore that the flows are integers. Finally, we use the admissible flow network to construct a data gathering schedule, i.e. a collection of aggregation trees using a polynomial–time algorithm (see [10] for details).

Experimental results provided in [10], show that the system lifetime obtained by the MLDA algorithm is near–optimal. [4] However, it involves solving a linear program (in Table 1) with $\Theta(n^3)$ variables and constraints. For large sensor networks, i.e. for large values of n, this is computationally expensive. [5] We next present an alternate formulation for the maximum lifetime data gathering and aggregation problem and we provide an efficient and effective approximation scheme for the MLDA problem.

3.2 $\mathcal{A}$–restricted MLDA: An Approximation Scheme for the MLDA Problem

Recall that a data gathering schedule $\mathcal{S}$ is a collection of aggregation trees. Further, any *feasible* schedule must satisfy the energy constraints at the individual sensors.

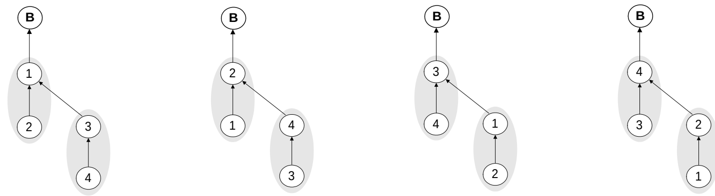
Now, consider the following variation of the MLDA problem. Let $\mathcal{A} = \{A_1, A_2, \ldots, A_k\}$ be a set of k aggregation trees. Each aggregation tree specifies how one data packet is collected and aggregated from each sensor and transmitted to the base station. For each aggregation tree A_i, we define the its *energy vector* $E(A_i)$ to be an $n \times 1$ vector indicating the energy consumed by each sensor $j \in A_i$. Intuitively, the energy vector for an aggregation tree A_i denotes the energy expended by each sensor during one round of data gathering using A_i. Note that, given the data packet size and the locations of the sensors and the base station, we can compute the energy vector for any aggregation tree by simply using the first order radio model [6].

The $\mathcal{A}$–restricted MLDA problem can then be stated as follows: Given a set of n sensors and of the base station, together with their locations and the initial energy of each sensor, and a set $\mathcal{A} = \{A_1, A_2, \ldots, A_k\}$ of k aggregation trees, find a maximum lifetime data gathering schedule that uses *only* the aggregation trees in $\mathcal{A}$.

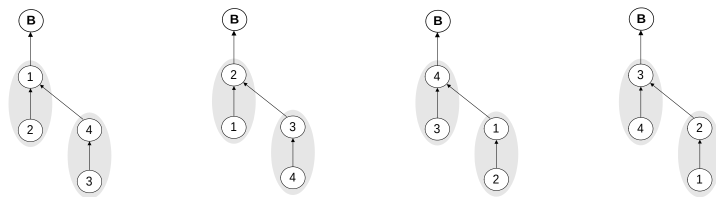
We refer to $\mathcal{A}$ as the *candidate tree set* (or simply candidate set). Let $\mathcal{S}$ be a schedule that uses only aggregation trees in $\mathcal{A}$. Let λ_i be the number of rounds an aggregation tree $A_i \in \mathcal{A}$ is used in $\mathcal{S}$, i.e. λ_i is the lifetime of A_i. In order to solve the $\mathcal{A}$–restricted MLDA problem, clearly one needs to find the λ_i's for all trees in $\mathcal{A}$, so that the energy constraint of each sensor is satisfied and the lifetime of the resulting schedule $\mathcal{S}$, i.e. $\sum_{A_i \in \mathcal{A}} \lambda_i$ is maximized. The $\mathcal{A}$–restricted MLDA problem can be solved using the integer program with linear constraints, shown in Table 2. This integer program computes a maximum lifetime schedule $\mathcal{S}$

[4] e.g. for 20 random networks of 100 sensors in a 50m $\times$ 50m region, the average integral lifetime obtained by MLDA was 8290 rounds, while the the (fractional) optimal solution was equal to 8292.6 [10].

[5] The worst–case running time of MLDA for n sensors is $O(n^{15} \log n)$ [10]. The average running time (measured CPU time) ranges from 20 seconds for a system of 20 sensors to about 330 minutes for 100 sensors [10].



(a) LRS Trees



(b) Randomized-LRS Trees

Figure 1: (a) LRS trees for a network of 4 sensors and a base station B, using a 2–level hierarchy. Each shaded region indicates a cluster (chain) at the lowest–level of the hierarchy. (b) RANDOMIZED–LRS trees for the same network obtained by permuting the sensors in the chain $(3, 4)$.

comprising of trees from a candidate set $\mathcal{A}$, subject to the energy constraint (11). Constraint (11) ensures that the energy each sensor expends in $\mathcal{S}$ is no more than its initial energy. Note that this integer programs has n constraints (one for each sensor) and k variables (one for each aggregation tree in $\mathcal{A}$). The linear relaxation of the integer program in Table 2, i.e. when the λ_i's are allowed to take fractional values, can be efficiently computed in polynomial–time; for $k = \Omega(n)$ the running time is $O(k^5 \log k)$ [1, pp. 418]. Then, we can obtain an approximate solution by fixing the lifetime of each aggregation tree to the floor of its value obtained from that linear relaxation (thereby ensuring that the energy constraints are all satisfied).

Clearly, if the candidate set $\mathcal{A}$ includes all possible aggregation trees, then a solution to the $\mathcal{A}$–restricted MLDA problem is also an optimal solution for the MLDA problem. Unfortunately, for a network with n sensors, the number of such aggregation trees is $(n+1)^{n-1} = \Omega(2^n)$ (follows from Cayley's theorem [5]). Solving a linear program with $\Omega(2^n)$ variables is computationally infeasible. In essence, the linear relaxation of the integer program in Table 2 provides us with an approximation scheme for the MLDA problem: the larger the candidate set $\mathcal{A}$ is, the better is the lifetime of the resulting schedule. [6] Thus, we are interested in finding easily computable candidate sets with few (e.g. $O(n^c)$ for some small constant c) aggregation trees, that provide us with data gathering schedules with superior lifetimes. In the rest of the section, we describe two approaches to generate such candidate tree sets. We discuss the performance of our approximation scheme along with the candidate set generation approaches in section 4.

3.3 Two Simple Approaches for Generating Candidate Sets

3.3.1 The LRS Approach

Our first approach is directly based upon the data gathering protocol proposed by Lindsey, Raghavendra and Sivalingam [13]. For brevity, we refer to this protocol as the LRS protocol. We choose this protocol since it significantly outperforms other competitive protocols (e.g. LEACH [6]) in terms of system lifetime. The LRS protocol induces a candidate set with n aggregation trees as explained.

In the LRS protocol, sensor nodes are initially grouped into clusters based on their distances from the base station. A chain is formed among the sensor nodes in a cluster at the lowest level of the hierarchy.

[6] To be precise, if $\mathcal{A}_1 \subseteq \mathcal{A}_2$ then the lifetime of $\mathcal{A}_1$ is potentially $\leq$ to the lifetime of $\mathcal{A}_2$.

Gathered data, moves from node to node, gets aggregated, and reaches a designated leader in the chain, i.e. its cluster head. At the next level of the hierarchy, the leaders from the previous level are clustered into one or more chains, and the data is collected and aggregated in each chain in a similar manner. Thus, for gathering data in each round, each sensor transmits to a close neighbor in a given level of the hierarchy. This occurs at every level, the only difference being that nodes that are leaders at each level are the only nodes that rise to the next level in the hierarchy. Finally at the top level, there is a single leader node transmitting to the base station.

To increase the lifetime of the system, in LRS the leader in each chain is chosen in a round-robin manner in each round. Specifically, LRS imposes an (implicit) ordering among the sensors in each chain at every level of the hierarchy. This ordering is determined during initialization (possibly at random), but remains fixed throughout the lifetime of the system. Having done so, LRS simply alternates between the members of each chain in a round–robin fashion to determine the leader(s) for a particular round. This is done at every level of the hierarchy, until there is a single leader transmitting to the base station.

Observe that, during the first n rounds of data gathering, LRS induces a set of n distinct aggregation trees; each tree being determined by the manner in which sensors transmit their data in these first n rounds. After the first n rounds, these n trees are used by the LRS protocol in a round–robin manner. We refer to these trees as the LRS aggregation trees. The LRS trees form a suitable candidate set for the $\mathcal{A}$–restricted MLDA approximation scheme. We refer to this approach as the $\mathcal{A}$–LRS algorithm and discuss its performance in the next section.

3.3.2 The Randomized–LRS Approach

As noted earlier, a larger candidate set of aggregation trees (in the subset–sense) can potentially lead to a data gathering schedule with better lifetime. Thus, we propose to augment the LRS set of aggregation trees by permuting the sensors in each cluster (chain) at the lowest level of the LRS hierarchy.

Consider the example network with 4 sensors and a base station shown in Figure 1(a). Figure 1(a) shows the candidate set of LRS aggregation trees. Sensors 1 and 2 form one chain, while sensors 3 and 4 form another chain in the lowest level of the hierarchy. In the first round of data gathering, sensors 1 and 3 are chosen as leaders in their respective chains – the two leaders rise to form a chain in the second level, whereby sensor 1 becomes the leader that transmits the aggregated data packet to the base station. In the second round, sensors 2 and 4 become the respective chain leaders, among which sensor 2 transmits to the base station. In the third round, sensors 1 and 3 again become leaders; however, due to the round-robin policy, sensor 3 is chosen to transmit to the base station. Finally, in the fourth round, sensors 2 and 4 again become leaders and sensor 4 transmits to the base station. The LRS protocol then repeats with the first aggregation tree.

Observe that, LRS imposes an (implicit) ordering among the sensors within each cluster (chain) at the lowest level of the hierarchy. This ordering limits the number of distinct aggregation trees used. By using a different ordering of the sensors in one or more chains, one can easily construct a different set of aggregation trees. For example, Figure 1(b) shows the four LRS aggregation trees obtained by reversing the order in which leaders are chosen in the chain $(3, 4)$.

We propose to augment the LRS candidate set of aggregation trees with the aggregation trees that result after randomly permuting the sensors in one or more chains. Specifically, our approach works as follows: we start with a greedy clustering of the sensors into chains, so that every sensor transmits to a close neighbor. Using P random permutations of the sensors, where each permutation randomly permutes the sensors within each chain at the lowest level of the hierarchy, we obtain an additional $P \cdot n$ LRS aggregation trees. By choosing P to be a small constant, we can still solve the $\mathcal{A}$–restricted MLDA problem efficiently, and can potentially obtain superior system lifetimes with respect to the $\mathcal{A}$–LRS algorithm. We refer to this approach as the $\mathcal{A}$–RANDOMIZED–LRS algorithm ($\mathcal{A}$–R–LRS , for brevity) and discuss its performance in the next section.

Table 3: Performance of different data gathering algorithms for a sensor network of size n in a 50m×50m field, with chains of size c. For each network size n, we show the system lifetime T given by each algorithm and the percentage gain G in lifetime with respect to lifetime of the LRS data gathering protocol, where P is the number of random permutations for the $\mathcal{A}$–R–LRS algorithm. OPT is the optimal (fractional) system lifetime.

		$n = 40$, $c = 5$		$n = 50$, $c = 5$		$n = 60$, $c = 5$		$n = 80$, $c = 10$		$n = 100$, $c = 10$	
Algorithm		T	G	T	G	T	G	T	G	T	G
LRS		5592	0	5872	0	5466	0	6002	0	5526	0
$\mathcal{A}$–LRS		6118	9.40	6490	10.52	6213	13.66	6545	9.04	6265	13.37
$\mathcal{A}$–R–LRS	$P = 10$	6212	11.08	6493	10.57	6465	18.27	6622	10.32	6541	18.36
	$P = 25$	6548	17.09	6510	11.00	6795	24.31	6677	11.24	6541	18.36
	$P = 50$	6602	18.06	6620	12.73	6811	24.60	6748	12.42	7005	26.76
	$P = 75$	6602	18.06	6692	13.96	6813	24.64	7223	20.34	7398	33.87
	$P = 100$	6602	18.06	6755	15.03	6813	24.64	7416	23.55	7440	34.63
	$P = 250$	6603	18.07	6782	15.49	6946	27.07	7645	27.37	7440	34.63
	$P = 500$	6603	18.07	6807	15.92	6946	27.07	7705	28.37	7593	37.40
	$P = 750$	6605	18.11	6808	15.94	7021	28.44	7827	30.40	7833	41.74
	$P = 1000$	6606	18.13	6808	15.94	7021	28.44	7827	30.40	8082	46.25
MLDA		6610	18.20	6808	15.94	7174	31.24	7945	32.37	8290	50.01
OPT		6611.8	18.23	6809.0	15.95	7176.2	31.28	7946.9	32.40	8292.6	50.06

4 Performance Evaluation

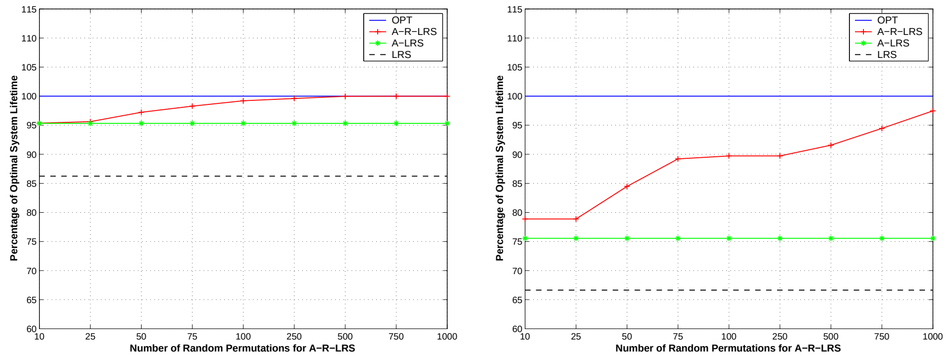
In this section, we compare the lifetime of a data gathering schedule obtained from the $\mathcal{A}$–LRS and $\mathcal{A}$–R–LRS algorithms, with the lifetime given by the LRS data gathering protocol [13], the MLDA algorithm [10] and the optimal (fractional) lifetime obtained from the linear relaxation of the integer program in Table 1. [7]

We conduct and report the results from two sets of experiments. For the first set of experiments, we consider a network of sensors randomly distributed in a 50m × 50m field, with the base station located at (25m, 150m). The number of sensors n in the network varies between 40, 50, 60, 80 and 100. Each sensor has an initial energy of 1J and generates packets of size 1000 bits. The energy model for the sensors is the first order radio model [6]. For this set of experiments, sensors are permitted to aggregate multiple incoming packets into one single outgoing packet.

For the sake of comparison, we implemented the hierarchical LRS protocol to perform data gathering with aggregation. We fix the size c of each chain at the lowest level of the hierarchy to 5 for networks with 40, 50, and 60 sensors, and to 10 for networks with 80 and 100 sensors. Given the location of the sensors and the base station, we employ the chain–forming algorithm used by the LRS protocol [13] – pick a sensor i farthest from the base station and form a chain that includes i and its $c - 1$ nearest neighbors; repeat the process with the remaining sensors until all sensors have been included in some chain. The number of levels in LRS was adjusted based on the network size (with a maximum of 3 levels as proposed in [13]).

Each experiment corresponds to a random placement of the sensors, for a particular network size. In each experiment, we measure the lifetime T for the data gathering schedule given by the LRS protocol. For the same placement of sensors, we measure the optimal (fractional) lifetime OPT and the (integral) lifetime obtained from the MLDA algorithm. We also compute the system lifetimes attained by the $\mathcal{A}$–LRS algorithm and the $\mathcal{A}$–R–LRS algorithm (for different number P of random permutations of the lowest–level chains). Specifically, we measure the lifetimes given by the $\mathcal{A}$–R–LRS algorithm for $P = 10, 25, 50, 75, 100, 250, 500, 750$ and 1000 permutations. For each data gathering algorithm, we denote $G = 100(T_A - T_{\text{LRS}})/T_{\text{LRS}}$ to be the percentage gain (in terms of the lifetime T_A) obtained from the corresponding algorithm with respect to the

[7] Note that the optimal fractional system lifetime is an upper bound on the optimal integer lifetime.



(a) Results for a 50 sensor network (b) Results for a 100 sensor network

Figure 2: Comparison of the system lifetimes given by the LRS, $\mathcal{A}$–LRS, and $\mathcal{A}$–R–LRS data gathering algorithms with the optimal (fractional) lifetime.

the lifetime T_{LRS} obtained by the LRS protocol.

Observe that, the $\mathcal{A}$–LRS and $\mathcal{A}$–R–LRS algorithms are centralized in nature. This implies that the clustering of the sensors is pre–computed at the base station. Similarly, a data gathering schedule is pre–computed at the base station (which is less likely to be resource–constrained) and transmitted to the individual sensors. [8] We take advantage of the fact that the base–station is aware of the locations of the sensors and has sufficient processing capabilities to compute efficient data–gathering schedule(s) for the sensors. [9]

Table 3 summarizes our main results for the first set of experiments. Note that the presented values for lifetimes are averaged across 20 different experiments for each network size. From this table, we make the following observations:

- The MLDA algorithm achieves gains in lifetime ranging from 15.94% to 50.01% over the LRS protocol. For the given set of results, MLDA is always within 3 rounds of the optimal (fractional) lifetime.
- The $\mathcal{A}$–LRS algorithm gives improvements in the lifetime ranging from 9.04% to 13.66% over the LRS protocol. Hence, by intelligently selecting the number of rounds that each LRS aggregation tree is used, instead of a simple round–robin selection, we can realize considerable improvement in system lifetime.
- The $\mathcal{A}$–R–LRS algorithm achieves gains ranging from 10.32% to 46.25% over the LRS protocol. Further, for all network sizes, the gain in lifetime obtained from $\mathcal{A}$–R–LRS algorithm increases as the number of random permutations (i.e. the size of the candidate set) is increased. Hence, randomization together with intelligent tree selection enables us to obtain significant improvement in system lifetime with respect to LRS.
- the $\mathcal{A}$–R–LRS algorithm gives lifetimes that are within 2.5% of the optimal fractional lifetimes, i.e. lifetimes that are quite close to the optimal system lifetimes.

Figure 2 provides a graphical representation of our results for networks of size 50 and 100. It is interesting to observe that the lifetime obtained from the $\mathcal{A}$–R–LRS algorithm with 1000 permutations is only one round less than the optimal fractional lifetime for the 50 sensor networks. For the 100 sensor networks, we observe that the lifetime given by the $\mathcal{A}$–LRS algorithm is about 75% of the optimal fractional lifetime, while the $\mathcal{A}$–R–LRS algorithm significantly improves the system lifetime from 78.8% (with 10 random permutations) to as much as 97.5% (with 1000 random permutations) of the optimal fractional lifetime. Hence, as we

[8] For each aggregation tree A_i in a schedule, the base station informs each sensor of its successor and number of predecessors in the A_i along with the lifetime of A_i. This information is sufficient for each sensor to collect and transmit data packets.

[9] Since a schedule is pre–computed at the base–station, it is possible to use that schedule, i.e. the set of data aggregation trees, to deploy a suitable energy–efficient address assignment scheme for the sensors (e.g. by using fewer bits for sensor addresses [17]).

expected, the $\mathcal{A}$–restricted MLDA approximation scheme does provide us with superior system lifetimes for larger candidate sets of aggregation trees.

For the second set of experiments, we consider larger networks of sensors randomly distributed in a 100m × 100m field, with the base station located at (50m, 300m). The number of sensors n in the network varies between 100, 200, 300, 400, and 500. The initial energy, energy model, and packet size for each sensor are as in the first set of experiments.

In the first set of experiments, in each round, we allowed each sensor to aggregate any number of incoming packets into one single outgoing packet. There are practical situations where this may not be the case. In this second set of experiments, we shed some light on the role of limited aggregation to system lifetime by also considering a simple "limited" aggregation model. Observe that in a single round of data gathering, each packet that a sensor sends reflects the measurements (values) from some number of sensors that influenced the information in that packet (through data aggregation). We define the *weight* of a data packet to be the number of sensors whose measurements are reflected in the information the data packet carries. For example, consider the leftmost aggregation tree in Figure 1(a). The packet sensor 2 sends to sensor 1 will have weight 1, while the packet sensor 3 sends to sensor 2 will have weight 2. In the unlimited data aggregation case, i.e. when sensors can to aggregate any number of incoming packets into one single outgoing packet, the weight that each data packet can have is not bounded by any constant.

We introduce the **limited aggregation model** which assumes that the weight of each data packet can have is bounded by K, which we call the *packet capacity.* Moreover, in any single round, whenever a sensor receives a number of incoming packets of total weight W it will send as many packets as required to hold weight $W+1$, i.e. it will send $\lceil (W+1)/K \rceil$ outgoing packets, with each outgoing packet having weight no more than K, and the total weight of all those outgoing packets equal to $W+1$. We say that the *aggregation ratio* of the limited aggregation model is $K : 1$. Note that the limited aggregation model allows unlimited aggregation by simply choosing the aggregation ratio to be $\infty : 1$. Let us also note that the MLDA algorithm cannot be used in this limited aggregation model. However, our $\mathcal{A}$–restricted MLDA approximation scheme, and therefore the $\mathcal{A}$–LRS and the $\mathcal{A}$–R–LRS algorithms can be used to solve the data aggregation problem even in the limited aggregation case.

Table 4 summarizes the results of the second set of experiments. We present results for network size n of 100, 200, 300, 400, and 500 with the chain size c fixed to 10, 10, 15, 20, and 25 respectively. We consider aggregation ratios of $\infty : 1$, $100 : 1$, $10 : 1$, $4 : 1$, and $2 : 1$. Once again, the presented values for lifetime are averaged across 20 different experiments for each network size. For each sensor placement (experiment) and aggregation ratio we measure the lifetime T obtained from the LRS algorithm, the $\mathcal{A}$–LRS algorithm, and the $\mathcal{A}$–R–LRS algorithm with 100 random permutations of the lowest–level chains. We make the following observations:

- the $\mathcal{A}$–LRS algorithm gives lifetime gain G between 21.55% to 37.42% with respect to the LRS protocol, for all network sizes and aggregation ratios considered.
- the $\mathcal{A}$–R–LRS algorithm gives lifetime gain G between 55.88% to 71.04% with respect to the LRS protocol, for all network sizes and aggregation ratios considered. Hence, randomization together with intelligent tree selection can deliver substantial improvements in the lifetime of large sensor networks as compared to the LRS protocol.
- the system lifetime decreases as the aggregation ratio decreases, i.e. there is an almost 50% reduction in the system lifetime when going from an aggregation ratio of $\infty : 1$ (unlimited aggregation) to an aggregation ratio of $2 : 1$.

5 Conclusions

In this paper, we present an approximate scheme for the maximum lifetime data aggregation problem in sensor networks, when the sensors are allowed to perform in–network aggregation of data packets. Our

Table 4: Performance of the data gathering schemes for a 100m×100m sensor network, using different values for the aggregation ratio (AR). For each network size and each aggregation ratio, we show the system lifetime (T) given by LRS ,$\mathcal{A}$–LRS , and $\mathcal{A}$–R–LRS as well as the percentage gain (G) in lifetime, when compared to the LRS protocol.

				$\mathcal{A}$–LRS		$\mathcal{A}$–R–LRS	
n	c	AR	LRS	T	G	T	G
100	10	$\infty : 1$	2454	2983	21.55	3902	59.00
100	10	100 : 1	2454	2983	21.55	3902	59.00
100	10	10 : 1	2315	2883	24.53	3770	62.85
100	10	4 : 1	2083	2624	25.97	3549	70.37
100	10	2 : 1	1516	1966	29.68	2593	71.04
200	10	$\infty : 1$	2854	3748	31.35	4449	55.88
200	10	100 : 1	2840	3725	31.16	4442	56.40
200	10	10 : 1	2489	3333	33.90	3886	60.15
200	10	4 : 1	2041	2709	32.73	3430	68.05
200	10	2 : 1	1372	1826	33.09	2281	66.25
300	15	$\infty : 1$	3119	3969	27.25	5037	61.49
300	15	100 : 1	3011	3819	26.83	4738	57.35
300	15	10 : 1	2714	3576	31.76	4255	56.80
300	15	4 : 1	2230	2949	32.22	3583	60.67
300	15	2 : 1	1607	2110	31.26	2532	57.55
400	20	$\infty : 1$	3090	4098	32.62	5263	70.30
400	20	100 : 1	2893	3866	33.63	4867	68.25
400	20	10 : 1	2605	3437	31.95	4327	66.12
400	20	4 : 1	2154	2851	32.49	3588	66.59
400	20	2 : 1	1470	1929	31.26	2426	65.04
500	25	$\infty : 1$	2996	4061	35.55	4939	64.85
500	25	100 : 1	2852	3864	35.48	4601	61.33
500	25	10 : 1	2563	3484	35.92	4102	60.05
500	25	4 : 1	1981	2711	36.85	3312	67.22
500	25	2 : 1	1117	1534	37.42	1899	69.96

approximation scheme utilizes intelligent selection of data aggregation trees from a *candidate set* of trees. We describe two approaches for generating suitable candidate sets of aggregation trees, leading to the $\mathcal{A}$–LRS and $\mathcal{A}$–R–LRS algorithms, which are simple and computationally efficient. We present experimental results demonstrating that the two proposed algorithms attain significant improvements in system lifetime, when compared to existing protocols; while the $\mathcal{A}$–R–LRS algorithm, for relatively small candidate sets, gives lifetimes that are quite close to the optimal system lifetimes. Furthermore, we introduce a simple limited aggregation model for the MLDA problem, and shed some light on the effect of limited aggregation on the system lifetime. As demonstrated by the experimental results, the $\mathcal{A}$–R–LRS algorithm delivers superior performance even in the limited aggregation case.

There are a number of important issues related to the maximum lifetime data gathering problem that need further investigation. As part of our current research, we are exploring a more complex scenario where a sensor is permitted to aggregate its own packets with only certain sensors, while acting as a router for other incoming packets. In the future, we plan to investigate modifications to our algorithms that would allow sensors to be added to (removed from) the network, without having to re–compute the entire schedule. Further, we plan to study the effect of sensor deployment strategies on the system lifetime, in order to attain desired trade-offs between the coverage provided by the sensors and the lifetime achieved by the system.

References

[1] D. Bertsimas and J. N. Tsitsiklis. *Introduction to Linear Optimization.* Athena Scientific, Massachusetts, 1997.

[2] M. Bhardwaj, T. Garnett, and A.P. Chandrakasan. Upper Bounds on the Lifetime of Sensor Networks. In *Proceedings of International Conference on Communications*, 2001.

[3] J.H. Chang and L. Tassiulas. Energy Conserving Routing in Wireless Ad-hoc Networks. In *Proceedings of IEEE INFOCOM*, 2000.

[4] J.H. Chang and L. Tassiulas. Maximum Lifetime Routing in Wireless Sensor Networks. In *Proceedings of Advanced Telecommunications and Information Distribution Research Program, College Park, MD*, 2000.

[5] Shimon Even. *Graph Algorithms.* Computer Science Press, 1979.

[6] W. Heinzelman, A.P. Chandrakasan, and H. Balakrishnan. Energy-Efficient Communication Protocols for Wireless Microsensor Networks. In *Proceedings of Hawaiian International Conference on Systems Science*, 2000.

[7] W. Heinzelman, J. Kulik, and H. Balakrishnan. Adaptive Protocols for Information Dissemination in Wireless Sensor Networks. In *Proceedings of 5th ACM/IEEE Mobicom Conference*, 1999.

[8] C. Intanagonwiwat, R. Govindan, and D. Estrin. Directed diffusion: A scalable and robust communication paradigm for sensor networks. In *Proceedings of 6th ACM/IEEE Mobicom Conference*, 2000.

[9] J. M. Kahn, R. H. Katz, and K. S. J. Pister. Mobile Networking for Smart Dust. In *Proceedings of 5th ACM/IEEE Mobicom Conference*, 1999.

[10] K. Kalpakis, K. Dasgupta, and P. Namjoshi. Maximum Lifetime Data Gathering and Aggregation in Wireless Sensor Networks. In *Proceedings of IEEE International Conference on Networking*, 2002.

[11] B. Krishnamachari, D. Estrin, and S. Wicker. The Impact of Data Aggregation in Wireless Sensor Networks. In *Proceedings of International Workshop on Distributed Event-Based Systems*, 2002.

[12] S. Lindsey and C. S. Raghavendra. PEGASIS: Power Efficient GAthering in Sensor Information Systems. In *Proceedings of IEEE Aerospace Conference*, 2002.

[13] S. Lindsey, C. S. Raghavendra, and K. Sivalingam. Data Gathering in Sensor Networks using the Energy*Delay Metric. In *Proceedings of the IPDPS Workshop on Issues in Wireless Networks and Mobile Computing*, 2001.

[14] S. Madden, R. Szewczyk, M. J. Franklin, and D. Culler. Supporting Aggregate Queries Over Ad-Hoc Wireless Sensor Networks. In *Proceedings of 4th IEEE Workshop on Mobile Computing and Systems Applications*, 2002.

[15] R. Min, M. Bhardwaj, S.H. Cho, A. Sinha, E. Shih, A. Wang, and A.P. Chandrakasan. Low-Power Wireless Sensor Networks. In *VLSI Design*, 2001.

[16] J. Rabaey, J. Ammer, J.L. da Silva Jr, and D. Patel. PicoRadio: Ad-hoc Wireless Networking of Ubiquitous Low-Energy Sensor/Monitor Nodes. In *Proceedings of the IEEE Computer Society Annual Workshop on VLSI*, 2000.

[17] C. Schurgers, G. Kulkarni, and M. Srivastava. Distributed On-Demand Address Assignment in Wireless Sensor Networks. In *IEEE Transactions on Parallel and Distributed Systems, Vol. 13*, 2002.

[18] S. Singh, M. Woo, and C. Raghavendra. Power–aware Routing in Mobile Ad Hoc Networks. In *Proceedings of 4th ACM/IEEE Mobicom Conference*, 1998.