Complexity of Data Collection, Aggregation, and Selection for Wireless Sensor Networks

Xiang-Yang Li[‡] Yajun Wang[†] Yu Wang[*]

***Abstract*—Processing the gathered information efficiently is a key functionality for wireless sensor networks. In this article, we study the time complexity, message complexity (number of messages used by all nodes), and energy cost complexity (total energy used by all nodes for transmitting messages) of some tasks, such as data collection (collecting raw data of all nodes to a sink), data aggregation (computing the aggregated value of data of all nodes), and queries for a multi-hop wireless sensor network of n nodes. We first present a lower-bound on the complexity for the optimal methods, and then, for most of the tasks studied in this article, we provide an (asymptotically matching) upper-bound on the complexity by presenting efficient distributed algorithms to solve these problems. Let ϱ_T, ϱ_M, and ϱ_E be the approximation ratio of an algorithm in terms of time complexity, message complexity, and energy complexity respectively for a certain task. We show that, for data collection, there are networks of maximum degree Δ, such that $\varrho_M \varrho_E = \Omega(\Delta)$ for *any* algorithm. We then present an efficient algorithm for data collection with $\varrho_T = O(1)$, $\varrho_M = O(1)$, and $\varrho_E = O(\Delta)$. For data aggregation, we show that there are networks of maximum degree Δ, such that $\varrho_T \varrho_E = \Omega(\Delta)$ for any algorithm. We then present an efficient algorithm for data aggregation with $\varrho_T = O(1)$, $\varrho_M = O(1)$, and $\varrho_E = O(\Delta)$. For data selection, we show that any deterministic distributed algorithm needs $\Omega(\Delta + D \log_D N)$ time to find the median of all data items, where N is the number of total elements collected by sensors. We then present a randomized algorithm that achieves this lower-bound with high probability (*w.h.p.*). In terms of the message complexity, there is a network G, such that $\Omega(n \log h)$ messages are required to compute the k^{th} smallest element in G in expectation and with probability at least $1/n^\delta$ for every constant $\delta < 1/2$, where $h = \min(k, N-k)$. We also present a randomized algorithm that achieves this bound *w.h.p.*.**

I. Introduction

Wireless sensor networks (WSNs) have drawn considerable amount of research interests recently because they can monitor the environment in a more efficient and convenient way. To design and deploy successful WSNs, a number of issues must be resolved such as deployment strategies, energy conservation, routing in dynamic environment, and localization. For WSNs, often the ultimate goal is to collect the data (either the raw data or in-network-processed data) from a set of targeted wireless sensors to some sink nodes and then perform some further analysis at sink nodes, or support various queries from the sink node(s), such as those formed in an SQL-like language. It is envisioned that the sink node issues queries regarding the data collected by some target senors, and the senors collaboratively generate an accurate or approximate response. Convergecast is a common many-to-one communication pattern used for these sensor network applications.

In this article, we study three different data processing operations, namely, *data collection*, *data aggregation*, and *data selection*. For each problem, we will study its complexity and present efficient algorithms to solve it. The complexity of a problem is defined as the worst case [1] cost (time, message or energy) by an optimal algorithm. Here, message complexity is defined as the number of messages used by all nodes while energy cost complexity is defined as the total energy used by all nodes for transmitting these messages. Studying the complexity of a problem is often challenging. We will design efficient algorithms whose complexity is asymptotically same as (or within a certain factor of) the complexity of that problem. *Data collection* is to collect the set of data items A_i stored in each individual node v_i to the sink node v_0. In *data aggregation*, the sink node wants to know the value $f(A)$ for a certain function f of all data items A, such as minimum, maximum, average, variance and so on. *Data selection* is to find the kth smallest (or largest) value of the set A where k could be any arbitrary value, *i.e.*, it solves aggregation queries about order statistics and percentiles. One typical example of data selection is to find the median.

Data collection and aggregation have been extensively studied in the community of networking and database for wired networks. Surprisingly, little is known about distributed (network) selection, despite it is a significant part in understanding the data aggregation, especially for wireless networks. For data collection, it is a folklore result that the total number of packet relays will be the smallest if we collect data using the breadth-first-search tree (BFS). It also has the smallest delay for wired networks. In [22], five distributive aggregations *max*, *min*, *count*, *sum* and *average* are carried out efficiently on a spanning tree. Subsequent work did not quite settle the time complexity, the message complexity and the energy complexity of data collection and aggregation, nor the tradeoffs

[‡] Xiang-Yang Li is with Department of Computer Science, Illinois Institute of Technology, Chicago, IL, USA and Institute of Computer Application Technology, Hangzhou Dianzi University, Hangzhou, PRC. The research of Xiang-Yang Li is partially supported by US National Science Foundation (NSF) CNS-0832120, National Natural Science Foundation of China under Grant No. 60828003, No.60773042 and No.60803126, the Natural Science Foundation of Zhejiang Province under Grant No.Z1080979, National Basic Research Program of China (973 Program) under grant No. 2010CB328100, the National High Technology Research and Development Program of China (863 Program) under grant No. 2007AA01Z180.

[†] Yajun Wang is with Microsoft Research Asia, Beijing, China.

[*] Yu Wang is with Department of Computer Science, University of North Carolina at Charlotte, Charlotte, NC, USA. The research of Yu Wang is supported in part by the US National Science Foundation (NSF) under Grant No. CNS-0721666 and CNS-0915331

[1]The worst case is to consider all networks of n nodes (and possibly diameter D, and maximum nodal degree Δ) and all possible distributions of all data items A over all nodes V.

among these three possibly conflicting objectives. The closest results to our article are [17], [18], [20]. All assumed a wireless network that can be modeled by a complete graph, which is usually not true in practice.

To the best of our knowledge, we are the first to study tradeoffs among the message complexity, time complexity, and energy complexity for data collection, data aggregation and data selection; we are the first to present lower bounds (and matching upper-bounds for some cases) on the message complexity, time complexity, and energy complexity for these three operations in WSNs. The main contributions of this article are as follows.

Data Collection: We design algorithms whose time complexity and message complexity are within constant factors of the optimum. We show that no data collection algorithm can achieve approximation ratio ϱ_M for message complexity and ϱ_E for energy complexity with $\varrho_M \cdot \varrho_E = o(\Delta)$, where Δ is the maximum degree of the communication network. We then prove that our data collection algorithm has energy cost within a factor $O(\Delta)$ of the optimum while its time and message complexity are within $O(1)$ of the corresponding optimum. Thus, our method achieves the best tradeoffs among the time complexity, message complexity and energy complexity.

Data Aggregation: We design algorithms for data aggregation whose time complexity and message complexity are within constant factors of the optimum. The minimum energy data aggregation can be done using minimum cost spanning tree (MST). We show that no data aggregation algorithm can achieve approximation ratio ϱ_T for time complexity and ϱ_E for energy complexity with $\varrho_T \cdot \varrho_E = o(\Delta)$. We then show that our data aggregation algorithm has energy cost within a factor $O(\Delta)$ of the optimum. In other words, our method achieves the best tradeoffs among the time complexity, message complexity and energy complexity with $\varrho_T = O(1)$, $\varrho_M = 1$, $\varrho_E = O(\Delta)$.

Data Selection: We first show that any deterministic distributed algorithm needs at least $\Omega(\Delta + D \log_D N)$ time to find the median of all data items when each node has at least one data item. We then present a randomized algorithm to find the median in time $O(\Delta + D \log_D N)$ when each node has $O(1)$ data item. Here D is the diameter of the communication network and N is the total number of data items. In terms of the message complexity, we show that, $\Omega(n \log h)$ messages are required to compute the k^{th} smallest element in expectation, and with probability at least $1/n^{\delta}$ for every constant $\delta < 1/2$, where $h = \min\{k, N - k\}$. We also present a randomized algorithm that can find the median with $O(N + n_{\mathbf{C}} \log N)$ messages with high probability (*w.h.p.*), where $n_{\mathbf{C}}$ is the size of the minimum connected dominating set (MCDS). In terms of energy complexity, we present a randomized efficient method that finds the median with energy cost at most $O(\omega(MST) \cdot \log N)$ *w.h.p.*, which is at most $O(\log N)$ times of the minimum. Value sensitive methods (whose complexity depending on the found value f_k) are also presented for finding the kth smallest element.

The rest of the article is organized as follows. In Section II, we first present our wireless sensor network model, define the problems to be studied, and then briefly review the connected dominating set (CDS). We study the complexity of distributed data collection, data aggregation, and data selection in WSNs in Section III, Section IV, and Section V respectively. We review the related work in Section VI and conclude the article in Section VII.

II. Preliminaries and System Models

A. Network Model

In this article, we mainly focus on studying the complexities of various data operations in wireless sensor networks. For simplicity, we assume a simple and yet general enough model that is widely used in the community. We assume that $n+1$ wireless sensor nodes $V = \{v_0, v_1, v_2, \cdots, v_n\}$ are deployed in a certain geographic region, where v_0 is the sink node. Each wireless sensor node corresponds to a vertex in a graph G and two vertices are connected in G iff their corresponding sensor nodes can communicate directly. The graph G is called the communication graph of this sensor network. We assume that links are "reliable": when a node v_i sends some data to a neighboring node v_j, the total message cost is only 1. In some of the results, we further assume that all sensor nodes have a communication range r and a fixed interference range $R = \Theta(r)$. Let $h_G(v_i, v_j)$ be the hop number of the minimum hop path connecting v_i and v_j in graph G, and $D(G)$ be the diameter of the graph, *i.e.*, $D(G) = \max_{v_i, v_j} h_G(v_i, v_j)$. Here, we assume that $D(G) \geq 2$. If $D(G) = 1$, then the graph G is simply a complete graph and all questions studied in this article can either be trivial or have been solved [17], [18], [20]. For a graph G, we denote the maximum node degree as $\Delta(G)$. When each node v_i has n_i data items, we define the weighted degree, denoted as $\tilde{d}_{v_i}(G)$, of a node v_i in graph G as $n_i + \sum_{v_j : v_i v_j \in G} n_j$. The maximum weighted degree of a graph G, denoted as $\tilde{\Delta}(G)$, is defined as $\max_i \tilde{d}_{v_i}(G)$.

Each wireless node has the ability to monitor the environment, and collect some data (such as temperature). Assume that $A = \{a_1, a_2, \cdots, a_N\}$ is a totally ordered multi-set of N elements collected by all n nodes. Here, N is the cardinality of set A. Each node v_i has a subset A_i of the raw data, n_i is the cardinality of A_i, and $A = \bigcup_{i=1}^{n} A_i$. Since A is a multi-set, it is possible that $A_i \cap A_j \neq \emptyset$. Then $\langle A_1, A_2, \cdots, A_n \rangle$ is called a distribution of A at sites of V. We assume that one packet (*i.e.*, message) can contain one data item a_i, the node ID, and a constant number of additional bits, *i.e.*, the packet size is at the order of $\Theta(\log n + \log U)$, where U is the upper-bound on values of a_i. Such a restriction on the message size is realistic and needed, otherwise a single convergecast would suffice to accumulate all data items to the sink which will subsequently solve the problems easily. We consider a TDMA MAC schedule and assume that one time-slot duration allows transmission of exactly one packet.

If energy consumption is to be optimized, we assume that the *minimum* energy consumption by a node u to send data correctly to a node v, denoted as $E(u, v)$, is $c_1 \cdot \|u - v\|^{\alpha} + c_2$, where c_1 (normalized to 1 hereafter) and $\alpha \geq 2$ are constants depending on the environment, and c_2 is the constant overhead cost by nodes u and v. In some of our results, we assume that $c_2 = 0$. We assume that each sensor node can dynamically adjust its transmission power to the minimum needed. We also assume that when the sensor node is in idle state (not transmitting, not receiving), its energy consumption is negligible. Since we assume that a TDMA MAC is used, the activity cycles for sensor nodes are assumed to be synchronized, and for any time slot, no sensor nodes will be listening for transmissions if it is not scheduled to receive data packets.

For data queries in WSNs, we often need build a spanning tree T of the communication graph G first for pushing down queries and propagating back the intermediate results. Given a tree T, let $H(T)$ denote the height of the tree, *i.e.*, the number of links of the longest path from root to all leave nodes. The depth of a node v_i in T, denoted as $\ell_T(v_i)$, is the length of the path from the root to v_i. The subtree of T rooted at a node v_i is denoted as $T(v_i)$, the parent node of v_i is denoted as $p_T(v_i)$, and the set of children nodes of a node v_i is denoted as $\mathrm{Child}(v_i)$.

B. Problem Definitions and Complexity Measures

We will study the time complexity, message complexity, and energy complexity of three different data operations, namely, data collection, data aggregation, and data selection.

The complexity measures we use to evaluate the performance of a given protocol are worst-case measures. The *message complexity* (and the *energy complexity*, respectively), of a protocol is defined as the maximum number of total messages (the total energy used, respectively) by all nodes, over all inputs, *i.e.*, over all possible wireless networks $\mathcal{G}$ of n nodes (and possibly with additional requirement of having diameter D and/or maximum nodal degree Δ) and all possible data distributions of A over V. The *time complexity* is defined as the elapsed time from the time when the first message was sent to the time when the last message was received. The *lower bound* on a complexity measure is the minimum complexity required by *all* protocols that answer the queries correctly. The approximation ratio ϱ_T (resp. ϱ_M and ϱ_E) for an algorithm denotes the worse ratio of the time complexity (resp. message complexity and energy consumption) used by this algorithm compared to an optimal solution over all possible problem instances. Here we assume that a TDMA MAC is used for channel usage. Obviously, the complexity depends on the TDMA schedule policy $\mathcal{S}$. Let $X(v_i, t)$ denote whether node v_i will transmit at time slot t or not. Then a TDMA schedule policy $\mathcal{S}$ is to assign 0 or 1 to each variable $X(v_i, t)$. A TDMA schedule should be *interference free*: no receiving node is within the interference range of the other transmitting node. When a schedule $\mathcal{S}$ is defined for tree T, it is interference free if and only if for any time slot t, if $X(v_i, t) = 1$, then $X(v_j, t) \neq 1$ for any node v_j such that $p_T(v_i)$ is within the interference range of v_j.

We now formally define the three data operation problems.

Data collection is to collect the set of *raw* data items A from all sensor nodes to the sink node. It can be done by building a spanning tree T rooted at the sink v_0, and sending the data from every node v_i to the root node along the unique path in the tree. The **message complexity** of data collection along T is $\sum_{i=1}^{n} n_i \cdot \ell_T(v_i)$. The **energy complexity**, defined as the total energy needed by all nodes for completing an operation, of data collection using T is $\sum_{i=1}^{n}[E(v_i, p_T(v_i)) \cdot \sum_{v_j \in T(v_i)} n_j]$.

The TDMA schedule should also be *valid* in the sense that every datum in the network will be relayed to the root. In other words, in tree T, when node v_i sends a datum to its parent $p_T(v_i)$ at a time slot t, node $p_T(v_i)$ should relay this datum at some time-slot $t' > t$. The largest time $\mathcal{D}$ such that there exists a node v_i with $X(v_i, \mathcal{D}) = 1$ is called the **time complexity** of this valid schedule. Time $\mathcal{D}$ is also called the *makespan* of the schedule $\mathcal{S}$. Then the time-complexity of the data collection problem is to find a spanning tree T and a *valid*, *interference-free* schedule $\mathcal{S}$ such that the makespan is minimized.

Data Aggregation: The database community classifies aggregation functions into three categories, see [16]: be distributive (*e.g.*, max, min, sum, and $count$), algebraic (*e.g.*, $plus$, $minus$, $average$, $variance$) and holistic (*e.g.*, $median$, k^{th} smallest or largest). Here we call the distributive or algebraic aggregation as *data aggregation* and the holistic aggregation as *data selection*. A function f is said to *distributive* if for every disjoint pair of data sets X_1, X_2, $f(X_1 \cup X_2) = h(f(X_1), f(X_2))$ for some function h. Typically we have $h = f$. For example, when f is sum, then h can be set as sum. For wired networks, it has been well-known that the distributive and algebraic functions can easily be computed using *convergecast* operations, which is straightforward applications of flooding-echo on a spanning tree.

Given an algebraic function f and a wireless network G, it is easy to show that each node only needs to send out information once. Hence, the connectivity of the communication graph of the data aggregation implies that it should be a tree to be optimal. Our task is to construct a data aggregation tree T and nodes' transmission schedule to optimize the time-complexity, or the message complexity, or the energy-cost complexity. Generally, we assume that the algebraic aggregation function f can be expressed as a combination of a constant number of (say k) distributive functions as $f(X) = h(g_1(X), g_2(X), \cdots, g_k(X))$. For example, when f is $average$, then $k = 2$ and g_1 can be set as sum and g_2 can be set as $count$ (obviously both g_1 and g_2 are distributive) and h can be set as $h(y_1, y_2) = y_1/y_2$. Hereafter, we assume that an algebraic function f is given in formula $h(g_1(X), g_2(X), \cdots, g_k(X))$. Thus, instead of computing f, we will just compute $y_i = g_i(X)$ distributively for $i \in [1, k]$ and $h(y_1, y_2, \cdots, y_k)$ at the sink node.

Given a distributive function g_i and a data aggregation tree T for it, the **message complexity** is the number of edges in T, which is fixed as n (recall that the root is node v_0). The **energy-cost complexity** is the total energy-cost used by all n links, *i.e.*, $\sum_{i=1}^{n} E(v_i, p_T(v_i))$. This can be found using minimum spanning tree algorithm where the link cost of uv is the energy-cost for supporting the communication of a link uv. The **time complexity** of data aggregation depends on the schedule $\mathcal{S}$. A schedule $\mathcal{S}$ is *valid* for data aggregation of A using tree T, if for every node v_i it is scheduled to transmit at a time slot t only if it has received data from *all* of its children nodes. Consequently, the time-complexity of *any* data aggregation scheme for a wireless network G is at least the height of the BFS tree, $H(\text{BFS}(G))$, rooted at sink v_0.

Data Selection is to find the kth ranked number from a given N numbers (possibly stored in a network). It is well-known that data selection can be done in linear time in a centralized manner [11]. Data selection is a holistic operation. Aggregate function f is *holistic* if there is no constant bound on the size of the storage needed to describe a sub-aggregate. All proposed algorithms for data selection are *iterative*, in the sense that they continuously reduce the set of possible solutions. The search space is iteratively reduced until the correct answer is located.

In this article, we will mainly study the complexity and efficient algorithm for these operations in wireless sensor networks. To address each of these problems, we usually first build a spanning tree T and then decide an interference-free and valid schedule of nodes activities such that certain complexity measure is optimized. However, our lower bound and approximation argument do not depend on the communication graph used, which may not be a tree.

C. Connected Dominating Set

A number of our methods will be based on a "good" connected dominating set (CDS) that has a bounded degree $\mathbf{d}$ and a bounded hop spanning ratio. Here a subgraph H of G is a CDS if (1) graph H is connected, and (2) the set of vertices of H is a *dominating set*, *i.e.*, for every node $v \in G \setminus H$, there is a neighboring node $u \in H$, *i.e.*, $uv \in G$. A node not in H is called a *dominatee node*. A subgraph H of G has a bounded spanning ratio (also known as stretch factor) if for every pair of nodes u and v in H, the distance (hop or weighted distance) of the shortest path connecting u and v in H is at most a constant times of the distance of the shortest path connecting them in G.

A number of methods have been proposed in the literature to construct such a good CDS. See [1], [2] for more details. A simple method is to partition the deployment region into grid of size $r/\sqrt{2}$, select a node (called *dominator*) from each cell if there is any, and then find nodes (called *connectors*) to connect every pair of dominators that are at most 3-hops apart. Then the diameter of the constructed CDS is at most a constant times of the diameter of graph G. Hereafter, we assume the availability of a good CDS $\mathbf{C} = (V_{\mathbf{C}}, E_{\mathbf{C}})$, with the maximum node degree $\Delta(\mathbf{C}) \leq \mathbf{d}$ for a constant $\mathbf{d}$. Notice that a good CDS also guarantees a constant approximation of the diameter of the graph.

Given a graph $G = (V, E)$, let $\mathbf{C} = (V_{\mathbf{C}}, E_{\mathbf{C}})$ be a connected dominating set of G. See Figure 1 for illustration. For an arbitrary node $u \in V_{\mathbf{C}}$ (*e.g.* the center node in Figure 1), let $T_{\mathbf{C}}$ be a BFS tree of $\mathbf{C}$ rooted at u. For a node $v \in V \setminus V_{\mathbf{C}}$, we define a unique dominator $\phi(v)$ which is the one having the shortest hop distance to the sink v_0. If there are ties, node IDs can be used to break them. The edge connecting v to its dominator $\phi(v)$ is denoted by $\overline{v\phi(v)}$. Our data communication tree is basically the union of $T_{\mathbf{C}}$ and all edges connecting dominatee to its unique dominator.

Definition 1 (Data Communication Tree (DCT)): Given a graph G and a CDS $\mathbf{C}$ of G, the data communication tree T is defined as $T = (V, T_{\mathbf{C}} \cup \{\overline{v\phi(v)} \mid v \in V \setminus V_{\mathbf{C}}\}$.

Fig. 1. Illustrations of a graph G, a CDS $\mathbf{C}$ of G, the BFS tree $T_{\mathbf{C}}$, and the data communication tree (DCT) T.

When the network (denoted by a graph G) is sparse, *i.e.*, the maximum node degree is bounded from above by a small constant, then the CDS will be more or less the same as the original network. In such a case, we do not need go through the process of building a CDS. We can build a spanning tree (such as DFS tree) on the original network directly.

Given data communication tree, an aggregate operation consists of (possibly repeated) two phases: a *propagation* phase where the query demands are pushed down into the sensor network along the tree; and an *aggregation* phase where the aggregated values are propagated up from the children to their parents. We now discuss some properties of the data communication tree T (or $T_{\mathbf{C}}$).

Theorem 1: Let G and $\mathbf{C}$ be a graph and a good CDS of G respectively. The data communication tree T built on top of $\mathbf{C}$ has following properties:

1) $\Delta(T_{\mathbf{C}}) \leq \mathbf{d}$, *i.e.*, the core part of T (not counting the leaves) has bounded degree.
2) For any edge $e \in E_T$, let $I(e)$ be the set of edges in $T_{\mathbf{C}}$ that have interferences with e, then $|I(e)| \leq c \cdot \mathbf{d} \cdot \Delta(G)$ for some constant c depending on R/r.

Proof: The first property directly comes from the property of the CDS $\mathbf{C}$. For the second property, for any edge $e = \overline{uv} \in E_T$, either u or v will be in $\mathbf{C}$ based on our construction. Assume $u \in V(\mathbf{C})$. For all edges having interferences with e, both end nodes should be within distance $2r + R$ from u by triangle inequality. Since $R = \Theta(r)$ and the CDS has a constant-bounded degree, there are at most a constant number $(\leq (\frac{R+2r}{r})^2 \cdot \mathbf{d})$ of nodes of $\mathbf{C}$ that are within distance $2r + R$ from u. On the other hand, all edges of T have at least one node in $\mathbf{C}$. Since each node is adjacent to at most $\Delta(G)$ edges, $|I(e)| \leq (\frac{R+2r}{r})^2 \cdot \mathbf{d} \cdot \Delta(G)$. This finishes the proof. ■

All our methods will be based on a good CDS and using **data clustering**: given a good CDS, for a node $v \in V \setminus V_{\mathbf{C}}$, it sends the data items to its dominator $\phi(v)$ in a TDMA manner.

Lemma 2: Given a *good* CDS of the graph G, data clustering can be done in time $O(\tilde{\Delta}(G))$.

Proof: We use the DCT tree T to do data clustering. For a node $v \in V \setminus V_{\mathbf{C}}$, assume that the edge $\overline{v\phi(v)}$ interferes with an edge $\overline{u\phi(u)}$. Then dominator nodes $\phi(u)$ and $\phi(v)$ are within distance at most $R + 2r$. Thus, there are at most $\frac{(R+2r)^2}{r^2}\mathbf{d}$ such dominator nodes. Consequently, the total number of data items of all nodes u such that $\overline{u\phi(u)}$ interferes with $\overline{v\phi(v)}$ is at most $\frac{(R+2r)^2}{r^2}\mathbf{d} \cdot \tilde{\Delta}(G) = \Theta(\tilde{\Delta}(G))$. Hence, every such edge $\overline{v_i\phi(v_i)}$ can be scheduled to transmit n_i times in $\Theta(\tilde{\Delta}(G))$ time-slots using a simple greedy coloring method that colors the nodes sequentially using the smallest available color. ■

After data clustering, all data elements are clustered in $T_{\mathbf{C}}$. In other words, each node v_i in the CDS will have data from all nodes dominated by v_i. Notice that the total number of messages for data clustering is $\sum_{v_i \notin V_{\mathbf{C}}} n_i$. Observe that $\Omega(\tilde{\Delta})$

is a lower bound on the time complexity for data collection.

III. Data Collection

A. Message, Energy, and Time Complexity

Obviously, the data collection can be done with minimum number of messages $\sum_{i=1}^{n} n_i \cdot h(v_i, v_0)$ using a BFS tree with root v_0. We now study the data collection with the minimum energy cost. Apparently, for any element, it should follow the minimum energy cost path from its origin to the sink node v_0 in order to minimize the energy consumption, where the weight of each link is the energy needed to support a successful transmission using this link. So minimizing the energy is equivalent to the problem of finding the shortest paths from the sink to all nodes, which can be done distributively in time $O(m + n \log n)$ for a communication graph of n nodes and m links [13]. We call the tree formed by minimum energy paths from the root to all nodes as the *minimum energy path tree (MEPT)*. Then we study the time complexity of data collection.

Algorithm 1 Efficient Data Collection Using CDS

Input: A CDS $\mathbf{C}$ with a bounded degree $\mathbf{d}$, tree $T_{\mathbf{C}}$.

1: Every node v_i sends its data to its dominator node $\phi(v_i)$.
2: **for** $t = 1$ to N **do**
3:     **for** each node $v_i \in V_{\mathbf{C}}$ **do**
4:         If node v_i has data not forwarded to its parent, v_i sends a new data to its parent in $T_{\mathbf{C}}$ in round t.

Algorithm 1 presents our efficient data collection method based on a good CDS $\mathbf{C}$. The constructed CDS has the maximum nodal degree at most a constant $\mathbf{d}$, and similar to Theorem 1, all nodes in CDS can be scheduled to transmit once in constant $\beta = \Theta(\mathbf{d})$ time-slots without causing interferences to other nodes in CDS. We take β time-slots as one *round*.

First, the data elements from each dominatee node (a node not in $\mathbf{C}$) are collected to the corresponding dominator node in the CDS $\mathbf{C}$. Here the dominatee nodes that are one-hop away from the sink node v_0 will directly send the data to v_0. Notice that this can be done in time-slots $O(\tilde{\Delta}(G))$.

Now we only consider the dominator nodes and the breadth-first-search spanning tree $T_{\mathbf{C}}$ of nodes in CDS rooted at the sink v_0. Every edge in the tree $T_{\mathbf{C}}$ will be scheduled exactly once in each round. For simplicity, we do not schedule sending an element more than once in the same round. At every round, nodes in CDS push one data item to its parent node until all data are received by v_0.

Theorem 3: Given a network G, data collection can be done in time $\Theta(N)$, with $\Theta(\sum_{i=1}^{n} n_i h(v_i, v_0))$ messages.

Proof: From Lemma 2, in $O(\tilde{\Delta}(G))$ time-slots, the data elements from each dominatee node are collected to the corresponding dominator node in the CDS. We show that after $N + H(T_{\mathbf{C}})$ rounds, all elements can be scheduled to arrive in the root, where $H(T_{\mathbf{C}})$ is the height of the BFS tree $T_{\mathbf{C}}$. Algorithm 1 illustrates our method to achieve this.

A CDS node v is in level i if the path from v to v_0 in BFS tree $T_{\mathbf{C}}$ has i hops. A level i is said to be *occupied* at a time instance if there exists one CDS node from level i that has at least one data. Assume that originally all levels $i \in [1, H(T_{\mathbf{C}})]$ are occupied, after collecting data from all dominatee nodes. We will show that each round the root will get at least one data item if there are data items in the network. We essentially will show that the occupied levels are *continuous*, *i.e.*, before each round t, there exists L_t such that all levels in $[1, L_t]$ are occupied and levels in $[L_t + 1, H(T_{\mathbf{C}})]$ are not. We prove this by induction. This is clearly true for round 1. Assume that it is true for round t. Then in round t, for each level $i \in [1, L_t - 1]$, every node in level $i + 1$ will send its data to its parent in level i. Then every level $i \in [1, L_t - 1]$ will have data for sure before round $t + 1$. Then $L_{t+1} = L_t$ if some nodes in level L_t still have some data; otherwise we set $L_{t+1} = L_t - 1$. Consequently, root will get at least one data item for each round whenever there are data items in the network. Since there are at most N data items, Algorithm 1 will take at most N rounds, *i.e.*, $O(N)$ time-slots because each round is composed of constant β time-slots.

When not all levels are occupied initially (*i.e.*, not all nodes have data items), at each round, each node in CDS will forward one data item (if there is any) to its parent node in T. Then we can show that after at most $\Theta(H(T_{\mathbf{C}}))$ rounds, the occupied levels will be *continuous*. Hence, the collection can be done in at most $N + H(T_{\mathbf{C}})$ rounds. Notice that $H(T_{\mathbf{C}}) = \Theta(D(G))$. Consequently, the total time-slots are at most $O(\tilde{\Delta}(G)) + O(N + D) = O(N)$ since $\tilde{\Delta}(G) \leq N$.

On the other hand, for any data collection algorithm, it needs at least N time slots since the sink can only receive one data item in one time slot and there are N data items.

The total number of messages used by the algorithm is of course at most $4 \sum_{i=1}^{n} n_i h(v_i, v_0)$ as the element at node v_i are relayed by at most $4 \times h(v_i, v_0)$ nodes in CDS (since $h(v_i, v_0) \geq 2$). Obviously any algorithm needs at least $\sum_{i=1}^{n} n_i h(v_i, v_0)$ messages. This finishes the proof. ■

B. Complexity Tradeoffs

One may want to design a universal data collection algorithm whose time-complexity, message-complexity and energy-complexity are all within constant factors of the optimum. Observe that Algorithm 1 is a constant approximation for both time-complexity and message-complexity. However, it is not a constant approximation for energy-complexity. Consider the following line network example: $n + 1$ nodes are uniformly distributed in a line segment $[0, 1]$; Sink v_0 is the leftmost node and node v_i is at position i/n and has one data item. Here we assume $r = 1$. See Figure 2 for illustration. Assume the energy cost for a link uv is $\|uv\|^2$. Then the minimum energy cost data collection is to let node v_i send all its data to node v_{i-1}. The total energy cost is $\sum_{i=1}^{n} i \cdot \frac{1}{n^2} \simeq 1/2$. While the energy cost of collecting data via CDS is $\sum_{i=1}^{n} (\frac{i}{n})^2 \simeq n/3$. On the other hand, the total number of messages of the minimum-energy data collection scheme is $n(n-1)/2$ and the number of time slots used by this scheme is $\Theta(n^2)$; both of which are $\Theta(n)$ times of the corresponding minimum.

Consider any data collecting algorithm $\mathcal{A}$. Let ϱ_M and ϱ_E be the approximation ratio for the message-complexity and

Fig. 2. Example: (a) a line network with $n+1$ nodes; (b) the minimum energy data collection tree; (c) the data collection tree via CDS, where v_0 is the only dominator.

energy-complexity of algorithm $\mathcal{A}$. We show that there are graphs of n nodes such that $\varrho_M \cdot \varrho_E = \Omega(n)$.

Theorem 4: Assume the energy cost for supporting a link uv is $\|uv\|^2$. For any data collection algorithm $\mathcal{A}$, there are graphs of n nodes, such that $\varrho_M \cdot \varrho_E = \Omega(n)$.

Proof: Consider the line graph example defined previously. For a node v_i, assume that the data collection path is composed of k_i hops and the length of the k_i links are $x_{i,1}, x_{i,2}, \cdots, x_{i,k_i}$. Then $\sum_{j=1}^{k_i} x_{i,j} = \frac{i}{n}$. The total energy cost, denoted as e_i, of such data collection path is $e_i = \sum_{j=1}^{k_i} x_{i,j}^2 \geq \frac{(\sum_{j=1}^{k_i} x_{i,j})^2}{k_i}$. Thus, $e_i \cdot k_i \geq (\frac{i}{n})^2$.

Obviously, the total number of messages are $\sum_{i=1}^{n} k_i$ and the total energy cost is $\sum_{i=1}^{n} e_i$. We will use the Holder's inequality: for positive a_i and b_i, $p > 0$, $q > 0$ with $\frac{1}{p} + \frac{1}{q} = 1$, we have

$$(\sum_{i=1}^{n} a_i^p)^{\frac{1}{p}} (\sum_{i=1}^{n} b_i^q)^{\frac{1}{q}} \geq \sum_{i=1}^{n} a_i \cdot b_i.$$

Equivalently, $(\sum_{i=1}^{n} a_i)^{\frac{1}{p}} (\sum_{i=1}^{n} b_i)^{\frac{1}{q}} \geq \sum_{i=1}^{n} a_i^{\frac{1}{p}} \cdot b_i^{\frac{1}{q}}$. Then

$$(\sum_{i=1}^{n} k_i)(\sum_{i=1}^{n} e_i) \geq (\sum_{i=1}^{n} \sqrt{e_i} \cdot \sqrt{k_i})^2 \geq (\sum_{i=1}^{n} \frac{i}{n})^2 = \frac{(n-1)^2}{4}.$$

Clearly, for data collection in this network example, the minimum number of messages is n for any scheme and the minimum energy cost is $1/2$ for any scheme. Thus, $\varrho_M \cdot \varrho_E \geq (n-1)^2/(2n) = \Theta(n)$. This finishes the proof. ■

Notice that we generally assumed that the energy cost for supporting a link uv is $\|uv\|^\alpha$. Then we can show that

$$(\varrho_M)^{\alpha-1} \varrho_E \geq \frac{n^{\alpha-1}}{2^{\alpha-1}}.$$

Notice that since $\varrho_E \geq 1$ and $\alpha \geq 2$, we have $(\varrho_M)^{\alpha-1}(\varrho_E)^{\alpha-1} \geq (\varrho_M)^{\alpha-1}\varrho_E \geq \frac{n^{\alpha-1}}{2^{\alpha-1}}$. Consequently, $\varrho_M \cdot \varrho_E \geq n/2$ still holds.

When we also take the maximum degree Δ into account, the preceding theorem implies the following corollary (the proof is essentially same by considering a network in which Δ nodes are evenly placed on a segment of length 1 and other $n - \Delta$ nodes are placed evenly with distance 1).

Corollary 5: For any data collection algorithm $\mathcal{A}$, there are graphs with maximum degree Δ, such that $\varrho_M \cdot \varrho_E = \Omega(\Delta)$.

The preceding theorem also implies that for any data collection algorithm $\mathcal{A}$, $\varrho_M \cdot \varrho_E \cdot \varrho_T = \Omega(\Delta)$, where ϱ_T is the approximation on the time-complexity by algorithm $\mathcal{A}$. We then show that for Algorithm 1, $\varrho_E = O(\Delta(G))$.

Theorem 6: Algorithm 1 is $\varrho_E = \Theta(\Delta(G))$-approximation for energy cost when the energy to support a link uv is $\|uv\|^2 + c_2$, where $c_2 = \Theta(r^2)$ is the energy cost of a node to receive a packet correctly.

Proof: Consider any node v_i and its minimum energy path $P_{v_i v_0}(G) = u_1 u_2 \cdots u_k$ to the sink node v_0 in the original communication graph G, where $u_1 = v_i$ and $u_k = v_0$. Assume that the total Euclidean length of this path is L. Obviously, $k \leq L \cdot \Delta / r$ since any node can have at most Δ neighbors within distance r and any path with length r contains at most Δ nodes. Let $x_i = \|u_i u_{i+1}\|$. Then the total energy cost is $\sum_{i=1}^{k-1}(x_i^2 + c_2) = (k-1)c_2 + \sum_{i=1}^{k-1} x_i^2$. Obviously, $\sum_{i=1}^{k-1} x_i^2 \geq \frac{(\sum_{i=1}^{k-1} x_i)^2}{k-1} \geq \frac{L^2 \cdot r}{L\Delta} = rL/\Delta$. On the other hand, since the Euclidean distance of the shortest path in G between v_i and v_0 is at most h, the shortest hop path connecting them is at most $2\lceil L/r \rceil$ hops. Thus, we can find a path using CDS to connect v_i and v_0 using at most $2 + 3 \cdot \lceil 2L/r \rceil \leq 4\lceil \frac{2L}{r} \rceil$ hops. The inequality is due to $\lceil L/r \rceil \geq 2$. Consequently, the total energy of the path connecting v_i and v_0 based on CDS is at most $4\lceil \frac{2L}{r} \rceil \cdot (r^2 + c_2)$. Observe that our data collection algorithm based on CDS will use the shortest hop path to route the data from v_i to the sink v_0. Thus, the energy cost of data collection using CDS is at most $\frac{4\lceil 2L/r \rceil \cdot (r^2 + c_2)}{rL/\Delta + (k-1)c_2} \leq \Theta(\Delta)$ times of the minimum. This finishes the proof. ■

Thus Algorithm 1 is asymptotically optimum if we want to optimize the time-complexity, message-complexity and energy-cost-complexity simultaneously. On the other hand, the minimum energy data-collection based on minimum energy path tree (MEPT) has delay that is at most $O(\Delta^3)$ times of the optimum.

Theorem 7: Data collection using MEPT is $\varrho_T = O(\Delta(G)^3)$-approximation for time complexity when the energy cost for supporting a link uv is $\|uv\|^2$.

Proof: Consider the node v such that its minimum energy path P to the root has maximum number of hops, which contains data. Assume P has h hops with Euclidean length $y_1, y_2, \cdots, y_h$. Then $\sum_{i=1}^{h} y_i \geq \frac{h \cdot r}{\Delta}$ since every node can have at most Δ nodes within r distance. The total energy of this path is $\sum_{i=1}^{h} y_i^2 \geq \frac{(\sum_{i=1}^{h} y_i)^2}{h} \geq \frac{hr^2}{\Delta^2}$. On the other hand, consider the path from v to root with minimum number of hops h_2. For this path, its energy cost is at most $h_2 r^2$, which should be at least $\sum_{i=1}^{h} y_i^2$ due to optimality of P. Thus, $h_2 r^2 \geq \frac{hr^2}{\Delta^2}$ implies that $h \leq h_2 \Delta^2$.

Now consider an edge in the MEPT, scheduling this edge will interfere $O(\Delta)$ nodes when the interfere range $R = O(r)$. In other words, if we take one round to be $O(\Delta)$ time slots, each edge in the MEPT can be scheduled once. The height of the MEPT is h. Scheduling the MEPT in a fashion similar to Algorithm 1 can finish the data collection operation in $O(N + h)$ rounds, hence $O(\Delta(N+h))$ time slots. On the other hand, any data collection algorithm will take $\Omega(N + h_2)$ time slot. As $h \leq h_2\Delta^2$, data collection using MEPT has time complexity that is at most $\varrho_T = O(\Delta(G)^3)$ times of the minimum. ■

Theorem 8: There is a network example that the delay of data collection by using MEPT is at least $\Delta(G)^2/8$ times of the optimum.

Proof: We construct a network example of a network with $n = p(\Delta^2/8 + 1)$ nodes, in which the MEPT has delay that is $\Omega(\Delta^2)$ times of the optimum. Consider a rectangle $uvwz$ with four segments uv, vw, wz, zu and side-length $\|uv\| = p \cdot r$ and $\|uz\| = p\frac{\Delta-2}{8} r(1-\epsilon)$. See Figure 3 for illustrations. There are $p+1$ nodes $u = u_1, u_2, \cdots, u_{p+1} = v$ uniformly distributed over the segment uv and $q = p\Delta^2/8 - 1$ nodes $v_1, v_2, \cdots, v_q$ uniformly distributed over the rest of the 3 segments. Then the MEPT path connecting u and the sink v is $uv_1v_2\cdots v_qv$, with $q = p\Delta^2/8 - 1$ hops. Obviously, the path $u_1u_2\cdots u_p$ connecting u and v has the least delay p. Thus, under this network example, the delay of data collection by using MEPT is at least $\Delta(G)^2/8$ times of the optimum. ■

Fig. 3. Example in which MEPT has delay $\Omega(\Delta^2)$ times of the optimum.

IV. Data Aggregation

We consider the case when, given any node v and its set of children nodes u_1, u_2, $\cdots$, u_d in a data aggregation tree, the aggregation data produced by node v has size same as the maximum size of data from all children nodes. Typical examples of such aggregation are *min*, *max*, *average*, or *variance*. In data aggregation, if one node sends information twice, it can always save the first transmission. Hence, the data aggregation should be done using a tree.

A. Message, Energy, and Time Complexity

The total message complexity for data aggregation using any tree T is n, where n is the number of nodes of the network. This is because every node v needs send at least once. We obviously can do data aggregation using any spanning tree and every node only needs to send once.

In addition, since every node need and only need send an aggregated data to its parent node in the data aggregation tree once, the minimum cost spanning tree is the energy efficient data aggregation tree, where the cost of any link uv is the energy cost of sending a unit amount of data over this link.

Time Complexity: We will show that the time complexity for any data aggregation is of the order $\Theta(D + \Delta(G))$. Algorithm 2 illustrates our method.

Theorem 9: Data aggregation can be done in $\Theta(D + \Delta)$ time with n messages.

Proof: For the node v that has the largest hop distance from the root, it needs at least D time slots to reach the root. Additionally, we need at least $\Delta(G)$ time-slots to schedule all nodes' transmissions due to interference constraints. Thus, $\max(D, \Delta)$ is a lower bound on time complexity.

We then show that Algorithm 2 takes time $\Theta(D + \Delta(G))$. The first step that let each dominatee node send its data to its dominator node will take time-slots at most $\Theta(\Delta(G))$. Then we perform aggregation round by round, where each round is composed of β time slots. (Constant β is the number of colors needed to color the interference graph induced by all CDS nodes.) Let H be the height of the BFS spanning tree $T_{\mathbf{C}}$ constructed in Algorithm 2. In round 1, all nodes in level H (all leaves) send a message to their parents. In round t, all nodes in level $H - t + 1$ should have received all the messages from their children, compute the aggregation of all data received so far, and then send the aggregated values to their parents. In all, the total number of rounds to finish data aggregation is H. Recall that each round is composed of β time-slots and $H = O(D)$. ■

If there are more than one aggregation functions, we can deliver the messages one by one. We call this as sequential aggregation (or pipelined aggregation).

Algorithm 2 Efficient Data Aggregation Using CDS

Input: A CDS $\mathbf{C}$ with bounded degree $\mathbf{d}$, a distributive function f and corresponding function h.

for each dominator node v_i **do**

  For the set of dominatee nodes of the node v_i, we build a minimum spanning tree (MST) rooted at v_i, where the link weight is the energy cost for supporting the link communication. The data elements from all these dominatee nodes are then *aggregated* to the corresponding dominator node v_i along the minimum spanning tree of these dominate nodes. In other words, any node v_k will compute $h(f(A_i), x_{k,1}, x_{k,2}, \cdots, x_{k,d_k})$ where $x_{k,j}$, for $j \in [1, d_k]$, is the aggregated value node v_k received from its jth child in the minimum spanning tree and d_k is the number of children of node v_k in the MST of all dominatee nodes of v_i. Notice that this aggregation can be done in time-slots $\Theta(\Delta(G))$.

Now we only consider the dominator nodes and the breadth-first-search spanning tree $T_{\mathbf{C}}$ of nodes in CDS rooted at the sink v_0. Let H be the height of $T_{\mathbf{C}}$.

for $t = 1$ to H **do**

  for each node $v_i \in V_{\mathbf{C}}$ **do**

    If node v_i has received aggregated data from all its children nodes in $T_{\mathbf{C}}$, it sends the aggregated data (using its own data and all aggregated data from its children) to its parent node in round t.

Corollary 10: k sequential data aggregations can be done in $O(D + \Delta + k)$ time with kn messages.

B. Complexity Tradeoffs

Again, we may want to design a data aggregation method that has constant approximation ratios for message complexity, time complexity, and energy complexity. However, we first show that aggregation based on MST (that is energy optimum for aggregation) is not efficient for time complexity.

Theorem 11: The minimum energy data aggregation based on MST is $\varrho_T = \Omega(\min(\frac{n}{\Delta}, \sqrt{n\Delta}))$-approximation for time complexity. On the other hand, $\varrho_T = O(\frac{n}{\Delta})$.

Proof: Consider a set of wireless nodes in a grid, with size length r'. Then $\Delta = \Theta((1/r')^2)$ and $D = \Theta(\sqrt{n}r') = \sqrt{n/\Delta}$, assuming the communication range to be 1. There exists a MST T which consists of n sequential line segments. See Figure 4 for an example. In fact, we can perturb the grid slightly, so that this bad MST is the only MST on the grid.

Fig. 4. Example of a bad MST.

Clearly, the data aggregation on T takes $\Theta(n)$ time slots. On the other hand, Algorithm 2 takes $O(\Delta + D)$ time slots. Notices that the diameter of the CDS is a constant factor of the original graph. Hence, $\varrho_T \geq \Omega(\frac{n}{\Delta + D}) = \Omega(\frac{n}{\Delta + \sqrt{n/\Delta}})$. The lower bound follows by considering the cases that $n \geq \Delta^3$ and $n \leq \Delta^3$.

Now consider the upper bound, the data aggregation on a MST takes at most $n-1$ time slots. However, any optimal solution should take $\Omega(\Delta + D)$. Hence $\varrho_T = O(n/\Delta)$. ■

Observe that our method (Algorithm 2) has constant ratio for both message complexity and time complexity. However, it is not always energy efficient due to the following theorem.

Theorem 12: Algorithm 2 is $\varrho_E = (\mathbf{d}+6)(\Delta(G)+1)$-approximation for energy cost, where **d** is the maximum nodal degree of CDS.

Proof: We assume the CDS is constructed by extending a maximal independent set. First, consider any dominator u and let $v_1, v_2, \cdots, v_k$ be the k dominatee nodes associated with u, where $k \le \Delta$, and $\|v_i u\| \le r$. Let x_1, x_2, $\cdots$, x_k be the k edges of the minimum spanning tree connecting u and its associated dominatees. It was proved in [3], [4] that $\sum_{i=1}^{d} x_i^2 \le 6r^2$. Assume that there are m dominator nodes in CDS (size of the maximal independent set). By the definition of CDS, $m \cdot (\Delta+1) \ge n$. Then the total energy cost of aggregating data from all dominatee nodes to dominators is at most $6mr^2$. Recall that in our CDS, all nodes in V_C will have at most **d** neighbors. It is easy to show that the total energy cost of aggregating data over CDS is at most $\mathbf{d} \cdot m \cdot r^2$. Thus, the total energy cost of aggregating data using Algorithm 2 is at most $(\mathbf{d}+6) \cdot m \cdot r^2 = \Theta(m \cdot r^2)$.

We now study the minimum cost of data aggregation, which is to use the minimum spanning tree of original communication graph G. Let y_1, y_2, $\cdots$, y_n be the length of the n edges of the MST connecting $n+1$ nodes v_0, v_1, $\cdots$, v_n. Since there are m independent nodes, the minimum spanning tree of these nodes has total length at least $m \cdot r$. As the MST is also a *Steiner tree* of the m independent nodes, the total length of the MST is at least half the length of the minimum spanning tree of the independent nodes, *i.e.*, $\sum_{i=1}^{n} y_i \ge m \cdot r/2$. The total energy cost of data aggregation based on the MST is $\sum_{i=1}^{n} y_i^2$. Notice, $\sum_{i=1}^{n} y_i^2 \ge (\sum_{i=1}^{n} y_i)^2/n \ge m^2r^2/(4n)$.

Then our algorithm has approximation ratio on energy cost at most $\frac{(\mathbf{d}+6)\cdot m \cdot r^2}{m^2r^2/(4n)} \le 4(\mathbf{d}+6)(\Delta+1) = O(\Delta)$, because of $m \cdot (\Delta+1) \ge n$.

We then show that there are examples of networks (with maximum degree Δ) such that Algorithm 2 is $\varrho_E = \Theta(\Delta)$-approximation for energy cost. Consider a line graph composed of $n+1$ nodes evenly distributed in a segment $[0, \frac{2r}{\Delta}n]$, *i.e.*, node v_i is at position $\frac{2r}{\Delta} \cdot i$, for $i \in [0, n]$. See Figure 5 for illustration. It is easy to show that the minimum energy data aggregation tree is simply the path $v_0 v_1 \cdots v_{n-1} v_n$, whose total energy cost is $n(\frac{2r}{\Delta})^2$. On the other hand, the energy cost of using tree T_C is $\Theta(\frac{2n}{\Delta}r^2)$ since the CDS will have $\frac{2n}{\Delta}$ nodes. The energy cost using tree T_C is $\Theta(\Delta)$ times of the minimum. This finishes the proof. ■

Fig. 5. Example: (a) the line network with $n+1$ nodes; (b) the minimum energy data aggregation tree; (c) the tree T_C.

Although our method is not energy efficient in the worst case (with approximation ratio up to $\Theta(\Delta)$ in the worst case), we show that it is the best we can do if we want to achieve $\Theta(1)$ ratio in delay. Again, given a data aggregation algorithm $\mathcal{A}$, let ϱ_E, ϱ_T and ϱ_M be the approximation ratios of $\mathcal{A}$ over all networks with n nodes and maximum degree Δ. We prove the following theorem.

Theorem 13: For any data aggregation algorithm $\mathcal{A}$, there are graphs of n nodes with maximum degree Δ, such that $\varrho_T \cdot \varrho_E = \Omega(\Delta)$.

Proof: Again consider the line network example used in the proof of Theorem 12. Assume that we choose a tree T for data aggregation. Consider the unique path P from v_0 to v_n in T. Assume P has k edges. Then the data aggregation using T takes at least k time slots. Let $\{x_i\}$ be the Euclidean lengths of the k edges in P. Clearly, $\sum_1^k x_i \ge 2rn/\Delta$. Then, the energy cost of this path is $\sum_1^k x_i^2 \ge (\sum_1^k x_i)^2/k \ge 4r^2n^2/(k\Delta^2)$. Notice that for any algorithm, the minimum delay is $2n/\Delta$ and the minimum energy cost is nr^2/Δ^2 (using MST). Thus, the approximation ratio ϱ_T and ϱ_E satisfy that: $\varrho_T \cdot \varrho_E \ge \frac{k}{2n/\Delta} \cdot \frac{4n^2r^2/(k\Delta^2)}{nr^2/\Delta^2} = 2\Delta$. ■

Consequently, our method for data aggregation is asymptotically optimum in terms of the tradeoffs between time-complexity, and energy-complexity when the energy needed to support a link uv is proportional to $\|uv\|^\alpha$. It remains a challenging question to design algorithms with best tradeoffs, when the energy needed to support a link uv is $\|uv\|^\alpha + c_2$ or, more generally, an arbitrary function.

V. Data Selection

In this section, we consider the scenario when we want to find the kth smallest data (or median when $k = N/2$) among all N data items stored in n wireless sensor nodes. Here we assume that each wireless sensor node will store at least one data item, and may store multiple data items. All data items are assumed to have a complete order. In most results here, we use the selection of median as an example to study the complexity.

A. Time Complexity

First we give a lower bound on the time complexity of any *deterministic* distributed algorithm.

Theorem 14: Any deterministic distributed method needs $\Omega(\Delta + D \log_D N)$ time to find the median of all data items.

Proof: For any deterministic algorithm, each node in the wireless network need send at least one message. In fact, if a node does not announce at least once, the adversary could place the median (or the kth largest item) in it. Hence, the time complexity is $\Omega(\Delta)$ due to wireless interferences. On the other hand, the time complexity of finding the median for a wireless network is at least as expensive as that of finding the median at a corresponding wired network (by assuming that no interferences exist among all transmission links). It has been proved in [21] that any deterministic algorithm for finding median in a wired network G of n nodes with diameter D, and total N data items has time complexity at least $\Omega(D \log_D N)$. Finding kth smallest element need time at least $\Omega(D \log_D k)$ when $k \le N/2$. Consequently, for wireless network G of n nodes with diameter D, any deterministic distributed algorithm needs at least $\Omega(\Delta + D\log_D N)$ time to find the median of all data items. ■

We then present our method (Algorithm 3) for distributed data selection in WSNs. In our method, we first collect data from dominatee nodes to corresponding dominator nodes, then we will run the distributed selection method for wired networks (from [21] and is summarized in Algorithm 4 for completeness of presentation) over the CDS. Algorithm 4 will be run by the sink node and the basic idea is as follows:

1) Initially, let $L = -\infty$ and $U = \infty$. The sink node will first broadcast control message **getRndElementsInRange**$(t, (L, U))$ to all nodes, asking for t *independent random* elements from *all* elements in the interval (L, U).
2) All nodes with data in this range together will return t random elements using t sequential findings of *one* random element. This can be done in time $O(D + t)$. Let $x_1, x_2, \cdots, x_t$ be the t random elements in the increasing order.
3) The sink node then broadcasts control message **countElementsInRange** to count the total number of items in the range of $(x_{i-1}, x_i]$ for $i \in [2, t]$. This can be done using simple counting aggregation in time $O(D)$ with the number of messages $n_{\mathbf{C}}$.
4) The sink node can then find the interval $(x_{j-1}, x_j]$ where the globally kth smallest element locates. We find the kth smallest element if x_j is. Otherwise, repeat the preceding steps using the new interval $(L, U) \Leftarrow (x_{j-1}, x_j)$.

Algorithm 3 Data Selection With Low Delay

Input: A CDS with bounded degree **d**.

1: Each dominatee node sends its data to its dominator node. This can take place in time $\Theta(\tilde{\Delta})$.

2: Then the median is found using only the connected dominating set, *i.e.*, only nodes in CDS will participate. We run the randomized Algorithm 4 with $t = 8\lambda D$ with a constant $\lambda > 1$ (see [21] for details). This method has time complexity $O(D\log_D N)$ in wired communication model. Notice that for wired networks, a node v_i can send a message to *each* of its neighboring nodes in one time-slot. This cannot be done in wireless networks. We will mimic the wired communication of CDS nodes using wireless links: one round of wireless communications corresponds to one time-slot in the corresponding wired network.

Theorem 15: There is a randomized distributed algorithm that can find the median of all data items in expected time $O(\tilde{\Delta} + D\log_D n)$ and also in time $O(\tilde{\Delta} + D\log_D n)$, *w.h.p.*.

Proof: The time-costs of our algorithm are as follows (1) the first step has time-complexity $\Theta(\tilde{\Delta})$; (2) the second step will cost $O(D\log_D N)$ rounds of communications with high probability [21], *i.e.*, $O(D\log_D N)$ time-slots *w.h.p.*, since each round is composed of β time-slots. ■

Notice that, if each node has single data item, then the time complexity of Algorithm 3 is $O(\Delta + D\log_D n)$ with high probability. Similarly, if we run the best deterministic algorithm for data selection for wired networks [21], we have the following theorem.

Theorem 16: There is a deterministic distributed algorithm that can find the median of all data items in time $O(\tilde{\Delta} + D\log_D^2 N)$ for wireless ad hoc networks of n nodes with diameter D and maximum weighted degree $\tilde{\Delta}$.

Observe that the lower bound $D\log_D N$ on the time complexity for wired networks is not tight for wireless networks. Consider a network formed by a sink node with coordinate $(0, 0)$, and other n nodes evenly distributed in the circle centered at $(0, 0)$ with radius r. Then $D = 2$ and $D\log_D N$ is only $2\log n$. On the other hand, $\Delta = n$, thus, data selection needs time at least n due to wireless interferences.

Algorithm 4 Random Data Selection $RDS(t, k)$

```
L ← −∞; U ← ∞; phase← 0;
repeat
  x_0 ← L; x_{t+1} ← U; phase←phase+1;
  {x_1, ..., x_t} ← getRndElementsInRange(t, (L, U))
  for i = 1, ..., t in parallel do
    r_i = countElementsInRange((x_{i−1}, x_i])
  if x_0 ≠ −∞ then
    r_1 ← r_1 + 1
  j ← min_{l∈{1,...,t+1}} Σ_{i=1}^{l} r_i > k
  k ← k − Σ_{i=1}^{j−1} r_i
  if k ≠ 0 and j ≠ 1 then
    k ← k + 1
until r_j ≤ t or k = 0
if k = 0 then
  return x_j
else
  {x_1, ..., x_s} =getElementsInRange([x_{j−1}, x_j]);
  return x_k
```

B. Message Complexity

We now study the message complexity of finding median of all numbers stored in the network.

1) Lower Bounds: Our lower bound on message complexity is based on the result on a two-party model. For two nodes connected by a link, each with $N/2$ data items, finding the median need $\Theta(\log N)$ messages [8], [29]; or generally, the kth smallest element ($k < \frac{N}{2}$) can be found using $\Theta(\log k)$ messages. In [21], Kuhn *et al.* studied the lower bound of the time complexity for the selection problem. Especially, they proved the following result on the two-party problem where both nodes have n elements. This result concludes the number of rounds (thus, an obvious lower bound on the number of messages) needed to compute the k^{th} smallest element.

Theorem 17 ([21]): Let $h = \min\{k, 2N - k\}$. Every, possibly randomized, generic two-party protocol needs at least $\Omega(\log h)$ rounds(messages) to find the element with rank k in expectation and with probability at least $1/h^{\delta}$ for any constant $\delta \leq 1/2$.

Fig. 6. A network example in which $\Omega(n\log h)$ messages are required to compute the k^{th} smallest element in G.

Fig. 7. Assume $a_i + b_i \leq a_{i+1} + b_{i+1}$. The intermediate vertex w_i can just forward messages between w_{i-1} and w_{i+1} without increase the total message complexity.

Based on the result, we show that there exist graphs that require $\Omega(n \log h)$ messages to compute the median. Our construction is similar to the lower bound of time complexity obtained in [21].

We first construct a line graph G with n nodes $u = w_0, w_1, \ldots, w_{n-2}, w_{n-1} = v$ as follows. See Figure 6 for illustration. The left and right vertices are u and v, each having $N/2$ elements. The other $n-2$ intermediate vertices $w_1, w_2, \cdots, w_{n-2}$ do not contain any element. Each node v_i is connected to v_{i+1} for $i \in [0, n-2]$. The distance between two consecutive nodes is r. We can construct a wireless communication graph which can be contracted to this example.

For simplicity, we first assume all intermediate vertices can only duplicate messages without any computation. This is exactly the case for the general two-party protocol. The following theorem is directly implied by Theorem 17.

Theorem 18: Assume that all intermediate vertices are only allowed to relay messages in G. $\Omega(n \log h)$ messages are required to compute the k^{th} smallest element in G in expectation and with probability at least $1/n^{\delta}$ for every constant $\delta < 1/2$.

Of course, in practice, we may allow intermediate vertices to perform certain computation on the messages it received before it sends out messages. However, we show that this additional freedom does not reduce the message complexity required. In particular, we argue that it is not necessary for any intermediate vertex to preform computation. Consider an intermediate vertex w_i, and its left vertex w_{i-1} and right vertex w_{i+1}. For each $i \in [1, n-2]$, assume that during the computation the vertex w_i receives a_i messages from w_{i-1} and sends b_i messages to w_{i-1}. Now consider the vertex w_i, without loss of generality, we assume $a_i + b_i \leq a_{i+1} + b_{i+1}$. Instead of performing computation, we let w_i forward all a_i messages from w_{i-1} to w_{i+1}. Because all b_i messages from w_i to w_{i-1} are computed from the a_i and b_{i+1} messages which w_{i+1} already poses after the forwarding. Hence, we can just send the b_i messages from w_{i+1} to w_{i-1} by passing w_i. See Figure 7 for illustration. In all, the number of messages does not increase. On the other hand, all the information w_i original obtained now available on w_{i+1}. Hence, this change does not affect the computation process.

We can pick i so that $a_i + b_i$ is (one of) the smallest. The preceding procedure can propagate from w_i to both leaves u and v, so that each intermediate vertex will forward $a_i + b_i$ messages. As we argued, the total number of messages does not increase.

Theorem 19: Let $h = \min\{k, N-k\}$. There is a wireless network with n nodes, such that $\Omega(n \log h)$ messages are needed to compute the k^{th} smallest element in expectation and with probability at least $1/h^{\delta}$ for every constant $\delta < 1/2$.

In Figure 7, its diameter $D = n$. Therefore, the lower bound stated in previous theorem can directly imply next result. The $\Omega(n)$ lower bound comes from the fact that each node need send at least one message.

In the preceding study of the lower bound on the message complexity of distributed selection, we only use the graph size n and number of data items N as parameters. We then extend this idea to get a more precise lower bound for finding median for all graphs with size n and diameter D. We construct a graph G as follows. Let $p = \frac{n-D+1}{2}$. On the left side, there are p vertices $u_1, u_2, \cdots, u_p$. On the right side, there are p vertices $v_1, v_2, \cdots, v_p$. Each of the vertices u_i and v_i has $\frac{N}{2p}$ elements, where N is total number of elements. The other $D-1$ intermediate vertices $w_1, w_2, \cdots, w_{D-2}, w_{D-1}$ do not contain any element. Graph G only has following edges $u_i w_1$, $w_{D-1} v_i$, for $1 \leq i \leq p$, and $w_j w_{j+1}$ for $j \in [1, D-2]$. See Figure 8 for illustration. This line graph G has diameter D. We show that finding the kth smallest element in G requires $\Omega(D \log h)$ messages.

Fig. 8. A network example in which any algorithm finding median needs at least $\Omega(n + D \log N)$ messages.

Theorem 20: Let $h = \min\{k, N-k\}$. There is a graph (wired or wireless communication model) with n nodes and diameter D, any algorithm finding kth smallest (or median) needs at least $\Omega(n + D \log h)$ (or $\Omega(n + D \log N)$) messages.

2) Upper Bound: We then present a randomized algorithm that will find the median of all N data items using expected number $O(n \log N)$ of messages, and also $O(n \log N)$ messages with high probability. The algorithm essentially is to find a random element x and then count the number of elements that are less than x. It is likely that a considerable fraction of all nodes no longer need be considered. By iterating this procedure on the remaining candidate nodes, the kth smallest element can be found quickly for all k. Algorithm 5 illustrates our basic method.

Algorithm 5 Data Selection With Less Messages

Input: A CDS with bounded degree **d**.

1: The dominatee node will send its data to its dominator node. This can take place using $O(N)$ total messages. Then only nodes in CDS will participate the second step.
2: We run the randomized data selection Algorithm 4 with $t = \lambda$ for some constant integer $\lambda \geq 1$.

Theorem 21: Let $h = \min\{k, N-k\}$. Given a wireless network with n nodes (each with one data item and having the same transmission range) and diameter D, Algorithm 5 can find the median with $O(N + n_{\mathbf{C}} \log N)$ messages with high probability, where $n_{\mathbf{C}}$ is the number of nodes in the CDS..

Proof: The first step costs at most N messages. Then we will prove that variable *phase* (defined in Algorithm 4) is at most $2 \log_{1/c} N$ (for a constant $c < 1$) with high probability when the median is found. Obviously, in each "phase" of Algorithm 4, the total number of messages is $2\lambda n_{\mathbf{C}}$: for each randomly selected data, each node in CDS will forward at most one control message from the sink and at most one data message back to the sink. Thus, the total number of messages used, with high probability, is at most $N + 4\lambda n_{\mathbf{C}} \log_{1/c} N$.

We then prove that variable *phase* is at most $2\log_{1/c} N$ (say for a constant $c = 1/2$) with high probability when the median is found. First, we compute an upper bound on the probability that after any phase i the wanted element is in a fraction of size at least c times the size of the fraction after phase $i-1$ for a suitable constant c, *i.e.*, $n^{(i)} \geq c \cdot n^{(i-1)}$. Here $n^{(i)}$ is the size of the all data items we have to check to find the kth smallest data before the phase i starts. Notice $n^{(0)} = N$. Let $\{a_1, a_2, \cdots, a_{n^{(i)}}\}$ be the sorted list of the $n^{(i)}$ data items that we will check for the kth smallest element in phase $i+1$. The probability that none of the λ randomly selected elements is in $\{a_k, a_{k+1}, \cdots, a_{k+cn^{(i)}/2}\}$ is at most $(1-c/2)^\lambda$. Same argument holds for data items $\{a_{k-cn^{(i)}/2}, \cdots, a_{k-1}, a_k\}$. Thus,

$$\Pr(n^{(i)} \geq c \cdot n^{(i-1)}) \leq 2(1-c/2)^\lambda \leq 2e^{-c\lambda/2}.$$

If $n^{(i)} \leq c \cdot n^{(i-1)}$ the phase i is called *successful*; otherwise it is called *failed*. Clearly, we need at most $S = \log_{1/c} N$ successful phases to find the kth smallest element. A phase i will fail with probability at most $p = 2e^{-c\lambda/2}$. Then among $2S$ phases, the probability that we have less than S successful phases (*i.e.*, at least $S+1$ failed phases) is at most

$$\begin{aligned} & \sum_{i=S+1}^{2S} \binom{2S}{i} p^i (1-p)^{2S-i} \leq \binom{2S}{S} p^S \leq (\frac{2eS}{S})^S \cdot p^S \\ = \;& (4e^{1-\frac{c\lambda}{2}})^{\log_{1/c} n} = 1/n^{(\frac{c\lambda}{2}-1-\ln 4)/\ln\frac{1}{c}} \end{aligned}$$

When $(\frac{c\lambda}{2}-1-\ln 4)/\ln\frac{1}{c} > 1$ (equivalently, $\lambda \geq \frac{2\ln\frac{4e}{c}}{c}$), this probability is at most $1/n$. For example, we can set $c = 1/2$, then $\lambda = \lceil 4\ln(8e)\rceil = 7$. Then, with probability at least $1-\frac{1}{n}$, Algorithm 5 will terminate in $2\log_2 N$ phases. Each phase will cost at most $2\lambda n_{\mathbf{C}}$ messages. This finishes the proof. ∎

Instead of collecting data from dominatee nodes to the dominator nodes, we can directly run Algorithm 4 on the wireless network G. By an argument similar to the proof of Theorem 21, the algorithm will find the median with $\Theta(n\log N)$ messages with high probability. Notice that this could be better than Algorithm 5 when N is very large, *e.g.*, $N = \Omega(n\log N)$.

We then discuss the message complexity when n sensor nodes are randomly and uniformly deployed in a square of $[0,a]\times[0,a]$ and each sensor node has one data item. It has been proved in [15] that, to guarantee the random wireless sensor network is connected with high probability, the transmission range r should satisfy that $n\pi r^2 = \Theta(a^2\cdot\log n)$. Thus, the number of dominators, using a maximal independent set, is of order $\frac{a^2}{r^2} = \Theta(\frac{n}{\log n})$. Thus, size of CDS $n_{\mathbf{C}} == \Theta(\frac{n}{\log n})$. Consequently, the message complexity of Algorithm 5 for random networks, with high probability, is $\Theta(n + n_{\mathbf{C}}\cdot\log n) = \Theta(n)$ when total data items is $N = O(n)$. This is asymptotically minimum.

C. Other Models

In previous discussions, we only consider the *comparison model*, *i.e.*, we assume the only operation between data items is to *compare their values*. A number of additional information can be used to improve the message and/or time complexities. For example, we may know that the values of all data items are positive integers or integers in range $[L, U]$.

Value Sensitive Query: We first consider the case that all data items are *positive integers*. We show that the message complexity of finding the median is no more than $\min\{N + 4n_{\mathbf{C}}\log f_k, 4n\log f_k\}$ based on methods in [7], [26] for wired networks. Here $n_{\mathbf{C}}$ is the size of the connected dominating set and f_k is the value of the kth smallest data. We assume synchronized communications are used by all wireless nodes. The method is essentially to solve the unbounded search: we first iterate to find i (starting from $i = 0$) such that the kth smallest element is in the range $(2^i, 2^{i+1}]$; we then use binary search to locate the kth smallest element in this range. It is easy to show that we need at most $2\log f_k$ such rounds and each round will cost us at most $2n_{\mathbf{C}}$ messages.

Known Intervals: When we know the interval $[L, U]$, then the message complexity is no more than $\min\{N + 2n_{\mathbf{C}}\log\frac{U}{L}, 2n\log\frac{U}{L}\}$, where U is the largest possible value and L is the lowest possible value among all data, by using a simple distributed binary search method. Observe that both U and L can be found using a simple distributive function $\max$ and $\min$ with n messages.

Notice that we can combine the preceding two techniques as follows. We first call $\min$ function to find L. Then we iterate to find i (starting from $i = \lfloor\log L\rfloor$) such that the kth smallest element is in the range $(2^i, 2^{i+1}]$; we then use binary search to locate the kth smallest element in this range. It is easy to show that we need at most $2\log\frac{f_k}{L}$ such rounds and each round will cost us at most $2n_{\mathbf{C}}$ messages when CDS is used or $2n$ messages if original network G is used. Thus, the total messages complexity is at most

$$\min\{N + n + 4n_{\mathbf{C}}\log\frac{f_k}{L}, 2n\log\frac{f_k}{L}\}.$$

D. Energy Complexity

At last, we study the energy cost of finding the median in any wireless networks by presenting some lower bound and upper bound.

Theorem 22: Any algorithm that can correctly find the median needs energy cost at least $\omega(MST) = \sum_{uv\in MST} E(u,v)$, where MST is the minimum spanning tree of G with weight of a link uv defined as the energy cost $E(u,v)$ for supporting link uv.

Proof: First of all, using adversary argument, we can show that every node needs to send at least one message to reveal some information about the data item it has. If it did not, adversary can put the median at this node to prevent the algorithm from finding the correct median. Let H be the graph over V and its set of edges are edges used by an optimum algorithm for communications. Graph H must be a connected graph; otherwise, the adversary can put the median at a connected component that does not contain the sink node. Consequently, the total link weight of minimum spanning tree is the lower bound for the energy consumption of any data selection algorithm. ∎

Assume that we are given the minimum spanning tree a prior. To minimize the energy consumption, we will directly run Algorithm 4, or value sensitive query methods discussed in previous subsection, on top of MST. Then we have the following theorem.

Theorem 23: There are algorithms that can correctly find the median with energy cost at most $O(\omega(MST) \cdot \log N)$ or $O(\omega(MST) \cdot \log \frac{f_k}{L})$ where L is the smallest value and f_k is the kth smallest value of all data items.

Proof: We showed that Algorithm 4 will terminate after at most $2 \log N$ phases with high probability. At each phase, the sink node need broadcast a control message and then all related nodes will reply with a certain answer. Obviously, both the broadcast from the sink along the MST and convergecast of the answer back to the sink cost energy $\omega(MST)$. Thus, Algorithm 4, run on top of MST has energy cost $O(\omega(MST) \cdot \log N)$.

For value sensitive query method, we first find L (which has energy cost at most $\omega(MST)$) and then then query will terminate after at most $2 \log \frac{f_k}{L}$ phases, where each phase cost energy at most $2\omega(MST)$. This finishes the proof. ■

If we interleave the preceding two methods (a phase is an atomic step), then we have algorithm whose energy cost is at most $O(\min\{\log N, \log \frac{f_k}{L}\})$ times of the minimum for data selection. Observe that from the network example illustrated by Figure 8 we can show that

Theorem 24: There are networks G of n nodes and diameter D, and placement of data items such that, the minimum energy required by *any* data selection algorithm is $\Omega(\omega(MST(G)) \log N)$.

However, this does not mean that, for *any* graph, it is always the case. In particular, it does not give the bound on ϱ_E for our algorithm on the MST. We make the following conjecture:

Conjecture 1: For any algorithm that can correctly find the median and *any* network, there exists a *placement of data items* such that the algorithm will cost energy at least

$$O(\omega(MST) \cdot \min\{\log N, \log \frac{f_k}{L}\})$$

where L is the smallest value of all data items, f_k is the value of the kth smallest element.

VI. Related Work

As the fundamental many-to-one communication pattern in sensor network applications, convergecast has been studied in both networking and database communities in recent years.

Most existing convergecast methods [5], [14], [32] are based on a tree structure and with minimum either energy or data latency as the objective. For example, [32] first constructs a tree using greedy approach and then allocates DSSS or FHSS codes for its nodes to achieve collision-free, while [5], [14] uses TDMA to avoid collisions. In [5], the authors did *not* give any theoretical tradeoffs between energy cost and latency. Gandham [14] mainly studied the minimum time convergecast for linear networks and tree networks. They presented a lower bound $3n - 2$ for time-complexity for convergecast in linear networks and proposed a distributed convergecast scheduling algorithm that requires at most $3n$ time slots for tree networks. They perform convergecast based on BFS, whose internal nodes implicitly forms a CDS structure, which is used here. However, BFS structure cannot guarantee the best theoretical performance in terms energy consumption. Furthermore, they did not provide theoretical results for general network topologies. Zhang and Huang [34] proposed a hop-distance based temporal coordination heuristic for adding transmission delays to avoid collisions. They studied the effectiveness of packet aggregation and duplication mechanisms with such convergecast framework. Kesselman and Kowalski [20] proposed a randomized distributed algorithm for convergecast that has the expected running time $O(\log n)$ and uses $O(n \log n)$ times of minimum energy in the worst case, where n is the number of nodes. They also showed the lower bound of running time of any algorithm in an arbitrary network is $\Omega(\log n)$. However, they assume that all nodes can dynamically adjust its transmission power from 0 to any arbitrary value and a data message by a node can contain *all* data it has collected from other nodes. In [9], Chu *et al.* studied how to provide approximate and bounded-loss data collection in sensor networks instead of accurate data. Their method used replicated dynamic probabilistic models to minimize communication from sensor nodes to the base station.

To significantly reduce communication cost in sensor networks, in-network aggregation has been studied and implemented. In TAG (Tiny AGgregation service) [22], besides the basic aggregation types (such as *count*, *min*, *max*, *sum*, *average*) provided by SQL, five groups of possible sensor aggregates are summarized: distributive aggregates (*e.g.*, *count*, *min*, *max*, *sum*), algebraic aggregates (*e.g.*, *average*), holistic aggregates (*e.g.*, *median*), unique aggregates (*e.g.*, *count distinct*), and content-sensitive aggregates (*e.g.*, *fixed-width histograms* and *wavelets*). Notice that the first two groups aggregates are very easy to achieve by a tree-based method. To overcome the severe robustness problems of the tree approaches [22], [23], [33], multi-path routing for in-network aggregation has been proposed [10], [25]. Then recently Manjhi *et al.* [24] combined the advantages of the tree and multi-path approaches by running them simultaneously in different regions of the network. In [17], Kashyap *et al.* studied a randomized (gossip-based) scheme using which all the nodes in a complete overlay network can compute the common aggregates of min, max, sum, $average$, and $rank$ of their values using $O(n \log \log n)$ messages within $O(\log n \log \log n)$ rounds of communication. Kempe *et al.* [18] earlier presented a gossip-based method which can get the average in $O(\log n)$ rounds with $O(n \log n)$ messages.

Data selection (*e.g.*, *median* or k-th *smallest element*) is much harder than general distributive and algebraic aggregates. Distributed selection has been studied in general networks [26]. Recently, Kuhn *et al.* [21] studied the distributed selection for general networks with n nodes and diameter D. They proved that distributed selection is strictly harder than convergecast by giving a lower bound of $\Omega(D \log_D n)$ on the time complexity. They then present a novel randomized algorithm which matches this lower bound with high probability and derandomized it to a deterministic distributed selection algorithm with a time complexity of $O(D \log_D^2 n)$ which constitutes a substantial improvement over prior art. However, there are no many results on distributed selection in wireless networks. In [27], Patt-Shamir presented a deterministic algorithm that computes the median value such that each node transmits only $O((\log n)^2)$ bits and a randomized algorithm that computes an approximate median in which each node transmits $O((\log\log n)^3)$ bits. He also proved that computing the exact number of distinct elements in the data set indeed requires linear communication in the worst case. His method implies total $O(n \log n)$ messages for finding median when each node has one data item, while our method can find the median in $O(n_{\mathbf{C}} \log n)$ messages. However, no lower bound on message complexity or time complexity is given in [27]. In [28], Patt-Shamir and Shafrir considered a distributed system where each node has a local count for each item. A top-k query in such a system asks which are the k items whose sum of counts, across all nodes in the system, is the largest. They presented a Monte-Carlo algorithm that outputs, with high probability, a set of k candidates which approximates the top-k items.

In [31], Shrivastava *et al.* proposed a novel data structure, *quantile digest*, to provide provable guarantees on approximation error and maximum resource consumption for approximate quantile queries. If the values returned by the sensors are integers in the range of $[1, \delta]$, then using quantile digest, they can answer quantile queries using message size m within an error of $O(\log(\delta)/m)$. Their method can also support range queries, most frequent items and histograms. Recently, there are several papers studied the various aggregations on streaming data, such as thresholded counts [19], threshold function query [30], approximate quantile queries [12], top-k monitoring [6].

VII. Conclusion

In this article, we study the time complexity, message complexity, and energy complexity of data collection, algebraic data aggregation, and data selection in WSNs. We first study lower bounds of the complexities for these problems and then present efficient algorithms that achieve asymptotically optimal time complexity, and message complexity. A number of interesting questions remain unsolved. One is to design efficient algorithms when each node will produce a data stream. The second challenge is what is the best algorithm when we do not require that the found data item to be precise, *i.e.*, we allow certain relative errors, or additive errors on the found answer. We also need to derive better lower bounds on energy cost and design efficient algorithms for holistic data operations. Another question is to study the time complexity and message complexity for other holistic queries such as *most frequent items*, *number of distinctive items*. The last, but not the least important is to study lower bounds on complexities, and to design efficient algorithms to address these questions when the communication links are not reliable.

References

[1] Alzoubi, K., Li, X.Y., Wang, Y., Wan, P.J., and Frieder, O. Geometric spanners for wireless ad hoc networks. *IEEE Trans. on Parallel and Distributed Processing 14*, 4 (2003), 408–421.

[2] Alzoubi, K., Wan, P.J., Frieder, O. Message-optimal connected dominating sets in mobile ad hoc networks. In *MobiHoc* (2002).

[3] Ambuhl, C., An optimal bound for the MST algorithm to compute energy efficient broadcast trees in wireless networks. In *Proc. of Int'l Colloquium on Automata, Languages and Programming*, (2005)

[4] Ambuhl, C. An optimal bound for the MST algorithm to compute energy efficient broadcast trees in wireless networks. *networks 1*, 2, 7.

[5] Arumugam, M., and Kulkarni, S. S. Tradeoff between energy and latency for convergecast. In *Proc. of INSS* (2005).

[6] Babcock, B., and Olston, C. Distributed top-k monitoring. In *Proc. of ACM SIGMOD*, 2003.

[7] Bentley, J., and Yao, A. An Almost Optimal Algorithm for Unbounded Searching. *Info. Processing Letters 5*, 3 (1976), 82–87.

[8] Chin, F., and Ting, H. F. An improved algorithm for finding the median distributively. *Algorithmica 2*, 1 (1987), 235–249.

[9] Chu, D., Deshpande, A., Hellerstein, J. M., and Hong, W. Approximate data collection in sensor networks using probabilistic models. In *Proc. of IEEE ICDE* (2006).

[10] Considine, J., Li, F., Kollios, G., and Byers, J. Approximate aggregation techniques for sensor databases. In *IEEE ICDE* (2004).

[11] Cormen, T. J., Leiserson, C. E., and Rivest, R. L. *Introduction to Algorithms*. MIT Press and McGraw-Hill, 1990.

[12] Cormode, G., Garofalakis, M., Muthukrishnan, S., and Rastogi, R. Holistic aggregates in a networked world: distributed tracking of approximate quantiles. In *Proc. of ACM SIGMOD* (2005).

[13] Faloutsos, M., and Molle, M. Optimal distributed algorithm for minimum spanning trees revisited. In *Symposium on Principles of Distributed Computing* (1995), pp. 231–237.

[14] Gandham, S., Zhang, Y., and Huang, Q. Distributed minimal time convergecast scheduling in wireless sensor networks. In *Proc. of IEEE ICDCS* (2006).

[15] Gupta, P., and Kumar, P. R. Critical power for asymptotic connectivity in wireless networks. *Stochastic Analysis, Control, Optimization and Applications, W.M.McEneaney, et al. (Eds.)* (1998).

[16] Han, J., and Kamber, M. *Data mining: concepts and techniques*. Morgan Kaufmann, 2006.

[17] Kashyap, S., Deb, S., Naidu, K. V. M., Rastogi, R., and Srinivasan, A. Efficient gossip-based aggregate computation. In *Proc. of ACM PODS* (2006).

[18] Kempe, D., Dobra, A., and Gehrke, J. Gossip-based computation of aggregate information. In *IEEE FOCS* (2003).

[19] Keralapura, R., Cormode, G., and Ramamirtham, J. Communication-efficient distributed monitoring of thresholded counts. In *Proc. of ACM SIGMOD* (2006).

[20] Kesselman, A., and Kowalski, D. R. Fast distributed algorithm for convergecast in ad hoc geometric radio networks. *J. Parallel Distrib. Comput. 66*, 4 (2006), 578–585.

[21] Kuhn, F., Locher, T., and Wattenhofer, R. Tight bounds for distributed selection. In *Proc. of ACM SPAA* (2007).

[22] Madden, S., Franklin, M. J., Hellerstein, J. M., and Hong, W. TAG: A Tiny AGgregation service for ad-hoc sensor networks. In *Proc. 5th USENIX OSDI* (2002).

[23] Madden, S., Franklin, M. J., Hellerstein, J. M., and Hong, W. The design of an acquisitional query processor for sensor networks. In *Proc. of ACM SIGMOD* (2003).

[24] Manjhi, A., Nath, S., and Gibbons, P. B. Tributaries and deltas: efficient and robust aggregation in sensor network streams. In *Proc. of ACM SIGMOD* (2005).

[25] Nath, S., Gibbons, P. B., et al. Synopsis diffusion for robust aggregation in sensor networks. In *Proc. of ACM Sensys* (2004).

[26] Negro, A., Santoro, N., and Urrutia, J. Efficient distributed selection with bounded messages. *IEEE Trans. on Parallel and Distributed Systems 8*, 4 (1997), 397–401.

[27] Patt-Shamir, B. A note on efficient aggregate queries in sensor networks. In *Proc. of ACM PODC* (2004).

[28] Patt-Shamir, B., and Shafrir, A. Approximate top-k queries in sensor networks. In *SIROCCO* (2006), pp. 319–333.

[29] Rodeh, M. Finding the Median Distributively. *J. of Computer and System Sciences 24*, 2 (1982), 162–166.

[30] Sharfman, I., Schuster, A., and Keren, D. A geometric approach to monitoring threshold functions over distributed data streams. In *Proc. of ACM SIGMOD* (2006).

[31] Shrivastava, et al. Medians and beyond: new aggregation techniques for sensor networks. In *Proc. of ACM Sensys* (2004).

[32] Upadhyayula, S., Annamalai, V., and Gupta, S. A low-latency and energy-efficient algorithm for convergecast in wireless sensor networks. In *Proc. of IEEE GLOBECOM* (2003).

[33] YAO, Y., AND GEHRKE, J. Query processing in sensor networks. In *Proc. of Conf. on Innovative Data System (CIDR)* (2003).

[34] ZHANG, Y., AND HUANG, Q. Coordinated Convergecast in Wireless Sensor Networks. *IEEE Military Communications Conf.* (2005).