On the Construction of Maximum-Quality Aggregation Trees in Deadline-Constrained WSNs

Bahram Alinia*, Mohammad H. Hajiesmaili*†, and Ahmad Khonsari*‡
* School of ECE, College of Engineering, University of Tehran, Iran
† Institute of Network Coding, The Chinese University of Hong Kong, Hong Kong
‡ School of Computer Science, Institute for Research in Fundamental Sciences (IPM), Iran
Emails: b.alinia@ece.ut.ac.ir, mohammad@inc.cuhk.edu.hk, ak@ipm.ir

***Abstract*—In deadline-constrained data aggregation in wireless sensor networks (WSNs), the imposed sink deadline in an interference-limited network hinders participation of all sensor nodes in data aggregation. Thus, a subset of nodes can contribute in aggregation and quality of aggregation (QoA) increases with the growth of the number of participating nodes. Scheduling the nodes' transmissions is a central problem, which aims to maximize the QoA, while satisfying the sink deadline, i.e., on-time delivery of the sensed data to the sink node. Although the previous studies have proposed optimal scheduling algorithms to this problem given a particular aggregation tree, there is no work on constructing optimal tree in this context. The underlying aggregation tree can make a big difference on QoA since we demonstrate that the ratio between the maximum achievable QoAs of different trees could be as large as $O(2^D)$, where D is the sink deadline. In this paper, we cast an optimization problem to address optimal tree construction for deadline-constrained data aggregation in WSNs. The problem is combinatorial in nature and difficult to solve as we prove its NP-hardness. We employ Markov approximation framework and devise two distributed algorithms with different computation overheads to find bounded close-to-optimal solutions. Simulation experiments in a set of representative randomly-generated scenarios show that the proposed algorithms significantly improve QoA by 101% and 93% on average compared to the best, to our knowledge, existing alternative methods.**

I. Introduction

Nowadays, a plethora of Wireless Sensor Networks (WSNs) have emerged for monitoring and tracking applications. Data gathering is witnessed as a primary operation in such applications. However, packet transmission as the major source of energy depletion turns data gathering into an acute task [1]. To mitigate this problem, *data aggregation* [2], [3] has been proposed as a promising energy conservation mechanism to eliminate the necessity of redundant transmission. Herein, some intermediate nodes may combine the gathered data of different sensors by in-network computation and transmit a single packet to the next hop.

Many recent applications are sensitive to the amount of the latency imposed by data aggregation. For example, in target tracking application the detected location of a moving object may have perceptible error with the actual location if data aggregation process takes too long [4]. Thus, the delay contribution of an efficient data aggregation algorithm may result in a reduced latency of the whole process so as to satisfy the delay constraint of the application.

Some previous researches have addressed delay-efficient aggregation problem by contemplating delay as the optimization variable to be minimized [5], [6] in which they consider participation of all sensor nodes in data aggregation. In these studies, other performance metrics such as energy efficiency have been incorporated as the constraints of the problem. However, participation of all sensor nodes introduces severe interference and may lead to terminating data aggregation in a time that is beyond the application's tolerable delay even attempting in delay minimization. One solution is to get the maximum application-specific tolerable delay, namely *deadline*, and try to participate the sensor nodes as much as possible before the aggregation time exceeds the deadline. Consequently, the problem turns into maximizing *Quality of Aggregation* (QoA) constrained by the application's deadline [7], [8].

In deadline-constrained WSNs, the number of simultaneous transmissions is restricted due to interference among the nodes. To cope with the interference problem, each sensor node employs an algorithm that schedule its transmission to occur only when the channel is free. However, if *waiting time* of a node exceeds a specific value, its data cannot be delivered to the sink before the deadline. Consequently, deadline-constrained data aggregation comes at the expense of sacrificing the QoA by decreasing the number of participating nodes in order to deliver the sensed data to the sink within the deadline. Devising an efficient algorithm that schedules the nodes' activity while preventing degradation of the QoA appreciably and meeting the delay constraint of the application is a challenging problem.

To maximize the number of participant nodes the following two issues should be addressed appropriately: 1) the scheduling policy, and 2) the structure of aggregation tree. While the primary task in the scheduling algorithm is to provide interference-free transmissions, a proper policy aims to exploit simultaneous transmissions to increase the number of participant nodes within the deadline [7]. The structure of data aggregation tree is another important factor. For data aggregation, a tree rooted at the sink node is the common structure since it simplifies design of routing and aggregation protocols and also helps to avoid problems such as double counting [9]. Without constructing an appropriate aggregation tree, we may not be able to achieve a desired level of QoA

978-1-4799-8381-0/15/$31.00 ©2015 IEEE

even by designing the scheduling algorithm optimally.

Existing approaches have tackled QoA maximization problem by devising optimal scheduling algorithms. In [7], authors have addressed this problem by proposing a polynomial time optimal algorithm. However, the authors do not consider the effect of data aggregation tree and so the algorithm is optimal only under a given aggregation tree. In Section III, we show that the ratio between QoAs of two data aggregation trees is $O(2^D)$ in the worst case where D is the aggregation deadline. This observation makes the problem of constructing maximum QoA tree attractive.

In this paper, we take into account both optimal scheduling and tree construction to improve QoA. Namely, we aim to construct an optimal aggregation tree and run an optimal tree-specific scheduling algorithm on the tree to maximize QoA. However, constructing the optimal aggregation tree given the network topology is nontrivial even in centralized manner. This is more problematic when we seek an appropriate solution amenable to distributed realization so as the sensor nodes choose their parents just using local information. We address this problem in single-sink WSNs setting through the following contributions:

- We show the impact of data aggregation tree structure on QoA by analytical discussion and explanatory example. Besides, we prove that the problem of optimal tree construction belongs to the class of NP-hard problems.
- We apply the recently proposed Markov approximation framework [10] to devise two close-to-optimal algorithms in which the sensor nodes contribute to migrate toward the optimal tree. The highlights are distributed implementation, bounded approximation gap, and robustness against the error of global estimation of sensor nodes by local information.
- Through simulations experiments, we show the superiority of our algorithms and compare them to the previous work [7]. Results demonstrate that two versions of our algorithm (the high and the low overhead versions) increase the QoA of the proposed algorithm in [7] by 103% and 93%, respectively.

The remaining of this paper is organized as follows. We briefly review related work in Section II. In Section III, the system model, problem formulation, and the proof of NP-hardness are explained. In Section IV, we devise two distributed algorithms for the problem. Simulation results are described in Section V, and the paper is concluded in Section VI.

II. Related Work

A. Minimum delay and deadline-constrained aggregation

The problem of minimum delay data aggregation tackled intensively in the literature. In [11], it is proved that the Minimum Latency Aggregation Scheduling (MLAS) problem is NP-hard and a $(\Delta - 1)$-approximation algorithm has been presented where Δ is the maximum node degree in the network. The current best approximation algorithms in [6], [12] achieve an upper bound of $O(\Delta+R)$ on data aggregation delay where R is the network radius. While most studies consider a protocol interference model, the solutions in [5], [13] assume a physical interference model that is more practical than the former. In [13], a scheduling algorithm for tree-based data aggregation is designed that achieves a constant approximation ratio by bounding the delay at $O(\Delta+R)$. The work is extended in [5] to consider any arbitrary network topology.

Within the context of deadline-constrained data aggregation models, the goal is not to minimize the delay as an objective of the problem. Rather, the objective is to maximize the number of sensor nodes participating in aggregation while respecting the application-specific deadline. This type of real-time data aggregation has recently gained attention in some works [4], [7], [8], [14]. In this regard, [7] presented a polynomial time optimal algorithm for the problem under the deadline and one-hop interference constraints. The problem is extended in [4] for a network with unreliable links under an additional constraint on nodes' energy level. In [4], the authors proved that in a network with V nodes, the problem is NP-hard when the maximum node degree of the aggregation tree is Δ. They propose a polynomial-time exact algorithm when $\Delta = O(\log V)$. In [8], the authors considered the same problem of [7] by taking into account the effect of data redundancy and spatial dispersion of the participants in the quality of final aggregation result and proposed an approximated solution for proved NP-hard problem. A main drawback of the aforementioned studies is that they all have tried to maximize the quality of data aggregation on a given tree and neglect the impact of changing the data aggregation tree structure.

B. Optimum Aggregation Tree Construction

Several studies have tackled the problem of constructing optimal data aggregation tree [15]–[18], [20] where all have been shown to be NP-hard. The study in [17] considers a sensor network composed of source and non-source nodes. Then, the problem is to construct an aggregation tree such that the minimum number of non-source nodes included. In [18], the problem of maximum lifetime aggregation tree is studied for single sink WSNs. This work is extended for multi-sink WSNs in [16]. The problem of constructing an aggregation tree in order to minimize the total energy cost is addressed in [20]. As solution, a constant factor approximation algorithm is proposed. In [15], the problem of constructing a minimum cost aggregation tree under Information Quality (IQ) constraint has been tackled. The authors considered event-detection WSNs and defined IQ as detection accuracy. In this paper, however, we aim to construct maximum quality aggregation tree under deadline constraint. This problem has not been addressed yet by the research community. Moreover, our solution method in solving the problem is completely different from the previous research and is based on a recently-proposed theoretical foundation, namely Markov approximation that may be considered as a potential solution for the same category of problems.

III. System Model

A. WSN Model

We consider a WSN whose topology is a graph $\mathcal{G} = (\mathcal{V} \cup \{S\}, \xi)$ where S is the sink node, $\mathcal{V}$ is the set of sensor nodes with $|\mathcal{V}| \triangleq V$, and ξ is the set of links between sensor nodes. All nodes have a fixed communication range and $(i, j) \in \xi$ if nodes i and j are adjacent, i.e., they are in the communication range of each other. Without loss of generality, we assume that each link has a unit capacity. Moreover, we suppose that the system is time-slotted and synchronized and a transmission takes exactly one time slot. The interference model is one-hop [7], [8], [14] such that simultaneous transmissions over links having a node in common cause an interference. The data aggregation topology is a spanning tree $\psi \in \mathcal{T}(\mathcal{G})$ rooted at the sink node where $\mathcal{T}(\mathcal{G})$ is the set of all spanning trees in the graph $\mathcal{G}$. In our deadline-constrained scenario, the data has to be received by the sink by the end of at most D time slots, where the value of D is specified by the deadline requirement of the applications. Moreover, let $H^{\psi}(i) \subseteq \mathcal{V}$ be the set that consists node i and all its predecessors (except the sink) in aggregation tree ψ.

We use binary decision variable F_i, where $F_i = 1$ if node i is a source and $F_i = 0$ otherwise. We note that a node is source if it is ready to send its sensed data. Moreover, binary variable n_i^{ψ} with $n_i^{\psi} = 1$ indicates that node i in tree ψ is allowed to send data to its parent and $\vec{n}^{\psi} = [n_i^{\psi}, i \in \mathcal{V}]$. Indeed, $n_i^{\psi} = 1$ indicates that node i participates in data aggregation. In this case, if $F_i = 1$ then node i is a source participant, otherwise node i participates in data aggregation as a relay node, i.e., it just aggregates the received data from its successors and forwards to its parent.

Let $\mathcal{V}_{\text{leaf}}^{\psi} \subseteq \mathcal{V}$ be the set of all leaf nodes and $\mathcal{V}_{\text{sel-src}}^{\psi} \subseteq \mathcal{V}$ the set of source nodes *selected* for data aggregation in tree ψ. Indeed, $i \in \mathcal{V}_{\text{sel-src}}^{\psi}$ if i is a source and all of its predecessors are selected for aggregation or more formally, $\mathcal{V}_{\text{sel-src}}^{\psi} = \left\{ i \in \mathcal{V} : F_i = 1 \text{ and } \prod_{j \in H^{\psi}(i)} n_j^{\psi} = 1 \right\}$.

To devise a feasible aggregation scheme, we assign a deadline of $W_i^{\psi}, 0 \leq W_i^{\psi} \leq D$ time slots to each *participant* node i in aggregation tree ψ and $\vec{W}^{\psi} = [W_i^{\psi}, i \in \mathcal{V}]$. The notion $QoA_{\psi}^{D}(\vec{W})$ denotes the quality of aggregation in tree ψ under the imposed deadline D and assigned waiting times determined by $\vec{W}$ and is defined as the number of source nodes that participate in data aggregation [7]:

$$QoA_{\psi}^{D}(\vec{W}) = |\mathcal{V}_{\text{sel-src}}^{\psi}| = \sum_{i \in \mathcal{V}} F_i \prod_{j \in H^{\psi}(i)} n_j^{\psi}. \qquad (1)$$

Moreover, in tree ψ, we define $QoA_{\psi}^{D}(\vec{W}_i)$ as QoA of the sub-tree rooted at node i with assigned waiting time of W_i. Hereafter, we use QoA and W_i instead of $QoA_{\psi}^{D}(\vec{W}_i)$ and W_i^{ψ} when the corresponding tree and scheduling are obvious or, a specific tree or scheduling is not the matter of concern. The following example shows how to find the maximum QoA in a given tree in simple and tractable cases. It also clarifies the data aggregation model for the reader.

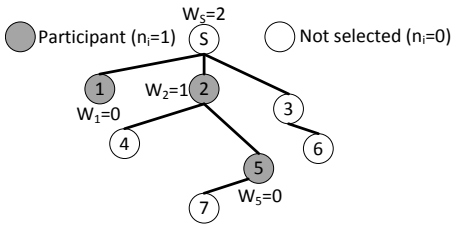


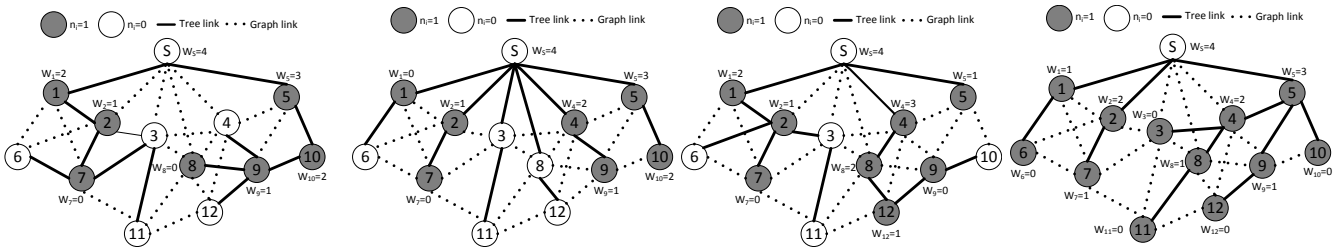
Fig. 1: Example 1 (Sink deadline $D = 2$).

Example 1. Consider the data aggregation tree illustrated in Fig. 1 where the sink deadline is set to $D = 2$ and all nodes are source. With the given deadline, the sink can choose at most $D = 2$ children (deadline constraint) and assign their waiting times as *distinguished* values (interference constraint) between 0 and $D - 1 = 1$. To maximize the number of source participant nodes (i.e., maximize QoA), one of the optimal choices for the sink is the assignment of $W_1 = 0$ and $W_2 = 1$. With this assignment, node 2 can assign a waiting time of 0 to one of its children (in this example node 5 with $W_5 = 0$). Eventually, the maximum QoA is 3 and participant nodes are 1, 2, 5. During the aggregation process, in the first time slot, node 1 and node 5 send their packets to their parents in parallel. In the second time slot, node 2 aggregates its own packet with the received data from node 5 and sends the aggregated data to the sink.

In [7], an algorithm is proposed to achieve the maximum $QoA_{\psi}^{D}(\vec{W})$ in given tree ψ. However, as we stated before, this algorithm is an optimal scheduling algorithm given a fixed tree as input and it does not change the structure of the tree for further improvement of QoA.

We argue that the aggregation tree structure may significantly impact QoA. When sink imposes a deadline D, all nodes with height "$\geq D$" cannot participate in data aggregation due to the delay constraint. Consequently, it seems that the structure of the optimal tree should not follow *chain-like* long trees. Instead, one might prefer a tree so as the height of the majority of nodes is "$\leq D$". But, the waiting time of a node with height h is upper bounded by $D - h$ and hence it can choose at most $D - h$ children of itself as the participants. The others together with their successors are ignored. Thus, the same as the long tree, a *star-like* fat tree may yield a non-optimal QoA. *In general, an aggregation tree which is neither so long nor so fat is suitable.* It is important to stress that the above conditions cannot bring significant insights to devise an algorithm to construct the optimal tree. Example 2 demonstrates the impact of data aggregation tree in more details.

Example 2. Fig. 2 illustrates a WSN topology with four data aggregation trees where bold lines indicate the tree links. Sink deadline is $D = 4$ and all nodes are the sources. The gray nodes indicates the aggregation participants and their waiting time is computed using the optimal scheduling algorithm [7]. Fig. 2a is an example of long tree where the height of nodes 4, 8, 12, 3, and 6 is 4. With sink deadline D, at most one node in height D of aggregation tree can participate in data aggregation. Moreover, node 11 is in distance $5 > D$ and there is no way to participate this node. Putting together these



(a) Long tree: Max QoA = 7 (b) Fat tree: Max QoA = 7 (c) Random tree: Max QoA = 8 (d) Optimal tree: Max QoA = 12

Fig. 2: A sample WSN scenario.

considerations, the maximum QoA of tree in Fig. 2a is 7. In Fig. 2b that demonstrates a star-like fat tree, 6 nodes are direct neighbors of the sink, but at most 4 out of 6 have the chance of being selected and other nodes along with their successors should be ignored. Consequently, 5 nodes lose the opportunity of being a participant. Fig. 2c shows another random tree with the maximum QoA of 8. Finally, the optimal data aggregation tree is shown in Fig. 2d where all nodes are participants. The optimal tree is obtained by trial and error. We remark that finding the optimal aggregation tree is not straightforward even in our tractable topology with only 12 nodes, while in practice the scale of the network is much larger than that of this example.

Theorem 1 below characterizes that the gap between the maximum achievable QoAs of two aggregation trees in a same network is extremely large in the worst case.

Theorem 1. *For an imposed deadline $D \leq \log V$ where all nodes are source, the maximum values of QoA in the optimal tree and worst-case tree are bounded to $2^D - 1$ and D, respectively.*

Proof. It is proved in [8] that QoA is bounded to $2^D - 1$ regardless of the aggregation tree structure. The bound is touchable when the network graph is dense enough (an obvious case is a complete graph). Therefore, we proceed to calculate the upper bound in the worst case. Indeed, the worst case occurs when we construct a chain-like tree with sink as the head of the chain. Observe that for a node i, $|H^{\psi}(i)|$ is equal to the distance of i to the sink in aggregation tree ψ. In a chain tree, there is only one possible way of scheduling where each node i having the property $|H^{\psi}(i)| \leq D$ assigned a waiting time of $D - |H^{\psi}(i)|$ and is a participant. There are D such nodes and therefore the maximum QoA of the tree is D. □

B. Problem Formulation

We formulate the following optimization problem to find the optimal tree. Namely, given network $\mathcal{G} = (\mathcal{V}, \xi)$ and deadline D, our objective is to construct an aggregation tree which maximizes the maximum QoA defined in Equation (1):

$$
\begin{aligned}
Z: \quad & \max \quad QoA_{\psi}^{D}(\vec{W}) && (2)\\
& \text{s.t.} \quad \forall i \in \{S\} \cup \mathcal{V} \backslash \mathcal{V}_{\text{leaf}}^{\psi} : \forall C \subseteq \{(j,i) : (j,i) \in \xi^{\psi}\}, \\
& \qquad \sum_{j:(j,i)\in C} n_j^{\psi} \leq W_i^{\psi} - \min_{j:(j,i)\in C} W_j^{\psi}, && (3)\\
& \qquad \vec{n}^{\psi} \in \{0,1\}^{V}, && (4)\\
& \qquad W_i^{\psi} \in \{0,1,\ldots,D-1\}, \quad \forall i \in \mathcal{V}, && (5)\\
& \qquad W_S^{\psi} = D, && (6)\\
& \qquad \psi \in \mathcal{T}(\mathcal{G}). && (7)
\end{aligned}
$$

Constraints (4)-(7) are straightforward based on the definitions. The most important constraint is given by Equation (3) to ensure that the deadline and interference constraints are not violated. Constraint (3) states that in a feasible scheduling, the sum of participant children of a parent node i and the minimum waiting time of its children should be less than or equal to node i's waiting time, W_i. We explain this constraint in detail. Observe that a selected children of i can only be assigned a waiting time of $W_i - 1, \ldots, 0$ due to deadline constraint in the parent node. Moreover, no two children of i can have a same waiting time otherwise, their simultaneous transmissions will be interfered in the parent node. Therefore, parent i can choose at most W_i children with distinct assigned waiting times chosen from the set $\{W_i - 1, \ldots, 0\}$. Now, suppose that minimum waiting time of children is M_i. This minimum is not always zero because in some cases number of selected children of i is less than W_i. Therefore, transmissions can only occur in time slots $M_i, M_i + 1, \ldots, W_i - 1$ which gives us a total of $W_i - M_i$ time slots. Since in each time slot we have at most one transmission (interference constraint), the total number of selected children cannot exceeds from $W_i - M_i$.

C. NP-hardness

The problem of finding the optimal tree is hard to solve as the number of trees in the network is extremely large in reality since in a complete network graph with V nodes and a sink, the number of feasible trees is V^{V-2}. We prove that problem Z is at least as hard as a variant of classical Maximum Coverage Problem (MCP) called Maximum Coverage Problem with Group Budget Constraint (MCPG) which is known to be NP-hard [21].

Maximum Coverage Problem (MCP). Given a collection of n sets $U = \{S_1, S_2, \ldots, S_n\}$ and a number l, the goal of MCP is to form set U' by choosing at most l sets from U such that the union of selected sets has the maximum cardinality:

$$MCP: \quad \max_{U'} \left| \bigcup_{S_i \in U'} S_i \right|, \qquad \text{s.t.} \quad U' \subseteq U, \quad |U'| \leq l.$$

Maximum Coverage Problem with Group Budget Constraint (MCPG). In [21], MCPG is introduced as a general case of MCP. In MCPG, n sets $S_1, \ldots, S_n$ at MCP are partitioned to L groups $G_1, \ldots, G_L$. MCPG has two versions namely the cost and the cardinality versions where the latter is our interest. In the cardinality version of MCPG, given number l, we should select at most l sets from U such that the cardinality of union of the selected sets is maximized where $U = \{S_1, S_2, \ldots, S_n\}$. Moreover, we are permitted to choose at most one set of each group. MCPG is clearly NP-hard because MCP which is known to be NP-hard [21] is a special case of MCPG where each set in U is considered as a group.

$$\begin{aligned} MCPG: \max_{U'} & \left| \bigcup_{S_i \in U'} S_i \right| \\ \text{s.t.} \quad & U' \subseteq U, \\ & |U'| \leq l, \\ & |U' \cap G_i| \leq 1, \forall i \in \{1, \ldots, L\}. \end{aligned} \tag{8}$$

The similarity between our tree construction problem and MCPG is that in both cases the objective is to maximize the cardinality. In MCPG we can choose at most one set from each group. Similarly, in problem Z, each node can subscribe (cover) different set of sensor nodes based on its deadline and we are allowed to choose at most one set according to the assigned deadline.

Theorem 2. *Problem Z is NP-hard.*

Proof. To prove, we reduce MCPG to problem Z with a polynomial time algorithm. To this end, we construct network graph $\mathcal{G}$ such that the sink is directly connected to L non-source sensor nodes $C_1, \ldots, C_L$ where L is the number of groups in MCPG. There are V other sensor nodes all considered as source nodes connected to $C_1, \ldots, C_L$ either directly or indirectly where V is equal to the total number of distinct elements in all groups. That is, $V = \sum_{i=1}^{L} \sum_{j=1}^{|G_i|} |g_{i,j}|$ where $|g_{ij}|$ is the cardinality of j^{th} set in group i and $|G_i|$ is the number of sets in group i. Then, we set the sink deadline to $D \geq N$ where N is the total number of sets in L groups, i.e., $N = \sum_{i=1}^{L} |G_i|$. We connect V sensor nodes to $C_1, \ldots, C_L$ and to each other such that if we assign a deadline of $D - ((\sum_{k=1}^{i-1} |G_k|) + j - 1)$ to the sink's neighbor C_i, j^{th} set of $G_i, 1 \leq j \leq |G_i|$ denotes the maximum cardinality set of the sensor nodes who will participate in data aggregation as the successors of C_i in a sub-tree rooted at this node in aggregation tree. An optimal assignment of deadlines to $C_1, \ldots, C_L$ is equal to select at most one set from each group of MCPG where this optimal assignment results in maximizing both the number of participants in data aggregation tree as well as the number of covered elements in MCPG. Therefore, a polynomial time optimal algorithm of problem Z leads to a polynomial solution of MCPG which completes the proof. □

IV. Approximation

Since problem Z is NP-hard, it is not possible to devise a computationally-efficient algorithm for the optimal solution even in a centralized manner. As such, we pursue approximated solutions. Among different approximation methods, we leverage Markov approximation framework [10] to propose an efficient near-optimal solution for the problem. Generally, in this framework the goal is to tackle combinatorial optimization problems in distributed manner so as 1) to construct a class of problem-specific Markov chains with a target steady-state distribution and 2) to investigate a particular structure of Markov chain that is amenable to distributed implementation. We first begin with a brief primer of the theoretical approximation framework [10] in the next subsection.

A. Markov Approximation

Recall that $\mathcal{T}$ denotes the set of all possible trees (configurations) of the network. For notational convenience, let us define $\Phi_\psi^D = \max(QoA_\psi^D(\vec{W}))$, i.e., when the network relies on tree $\psi \in \mathcal{T}$ for data aggregation and sink deadline is D, the maximum data aggregation quality is Φ_ψ^D. In addition, p_ψ denotes the percentage of time that configuration ψ is employed to accomplish data aggregation. Using these notations we can rewrite problem Z as follows:

$$Z^{\mathbf{eq}}: \max_{\{p_\psi \geq 0, \psi \in \mathcal{T}\}} \quad \sum_{\psi \in \mathcal{T}} p_\psi \Phi_\psi^D, \quad \text{s.t.} \quad \sum_{\psi \in \mathcal{T}} p_\psi = 1.$$

To derive a closed-form of the optimal solution of problem $Z^{\mathbf{eq}}$ and to open new design space for exploring a distributed algorithm, we formulate problem Z^β as an approximated version of $Z^{\mathbf{eq}}$ using *log-sum-exp* approximation [10]

$$Z^\beta: \max_{\{p_\psi \geq 0, \psi \in \mathcal{T}\}} \sum_{\psi \in \mathcal{T}} p_\psi \Phi_\psi^D - \frac{1}{\beta} \sum_{\psi \in \mathcal{T}} p_\psi \log p_\psi \tag{9}$$

$$\text{s.t.} \qquad \sum_{\psi \in \mathcal{T}} p_\psi = 1, \tag{10}$$

where β is a large enough positive constant that controls the accuracy of the approximation. Problem Z^β is an approximated version of problem Z off by an entropy term $-\frac{1}{\beta} \sum_{\psi \in \mathcal{T}} p_\psi \log p_\psi$ and it is a convex optimization problem so by solving KKT conditions, its optimal solution is obtained by

$$p_\psi^* = \frac{\exp\left(\beta \Phi_\psi^D\right)}{\sum_{\psi' \in \mathcal{T}} \exp\left(\beta \Phi_{\psi'}^D\right)}, \quad \psi \in \mathcal{T}. \tag{11}$$

Moreover, the optimal value is

$$\widehat{\Phi}_\psi^D = -\frac{1}{\beta} \log \left(\sum_{\psi \in \mathcal{T}} \exp\left(\beta \Phi_\psi^D\right) \right). \tag{12}$$

Finally, the approximation gap is characterized as:

$$|\max_{\psi \in \mathcal{T}} \Phi_\psi^D - \widehat{\Phi}_\psi^D| \leq \frac{1}{\beta} \log |\mathcal{T}|, \tag{13}$$

where the approximation gap approaches to zero as β approaches to infinity. This means that with larger values of β the approximation model is more accurate.

In the next step, our endeavor is to obtain the solution of problem Z^{β} by time-sharing among different tree configurations according to p^*_{ψ} in Eq. (11). According to the basic framework, the key is to investigate a well-structured and distributed-friendly Markov chain whose stationary distribution is p^*_{ψ}.

B. Markov Chain Design

We design a time-reversible Markov chain with states space being $\mathcal{T}$ and the stationary distribution being p^*_{ψ}. Then, we resort this Markov chain structure to hop (migrate) among different states (trees) such that a tree with high QoA has more chances to be visited by Markov random walks. The problem is solved when the Markov chain converges to the ideal steady-state distribution.

Given the Markov chain state space, the next step is to construct the transition rate between two states. Let $\psi, \psi' \in \mathcal{T}$ be two states of Markov chain and $q_{\psi,\psi'}$ be the transition rate from ψ to ψ'. Herein, the theoretical framework enriches us by two degrees of freedom. It turns out that the key in designing distributed algorithms is to design a Markov chain such that (i) any two states are reachable from each other (i.e., Markov chain is irreducible) and (ii) the detailed balance equation is satisfied (i.e., $p^*_{\psi} q_{\psi,\psi'} = p^*_{\psi'} q_{\psi',\psi}, \forall \psi, \psi' \in \mathcal{T}$). Moreover, we are allowed to set the transition rates between any two states to be zero if they are still reachable from any other states. We skip the details and refer the readers to [10] for further explanation.

In practice, however, direct transition between two states means migration between two tree structures. To derive a distributed algorithm, we only allow direct transitions between two states when there is exactly one difference between the edges of the current and the target trees. Namely, two states ψ and ψ' are directly reachable from each other if we can construct tree ψ' by deleting an edge $(i,j) \in \xi$ from ψ and adding edge $(i,k) \in \xi$ to ψ. Using this transition structure the next step is to set the transition rate as follows:

$$q_{\psi,\psi'} = \frac{1}{\exp(\alpha)} \frac{\exp(\beta \Phi^D_{\psi'})}{\exp(\beta \Phi^D_{\psi}) + \exp(\beta \Phi^D_{\psi'})} \qquad (14)$$

where $\alpha \geq 0$ is a constant and $q_{\psi',\psi}$ is defined symmetrically.

C. Algorithm Design

Our goal is to realize a distributed implementation of the Markov chain proposed in the previous section. In this part, we detail our implementation.

To compute transition rate between the states, the maximum QoA of both the current (Φ^D_{ψ}) and the target ($\Phi^D_{\psi'}$) states are required. To calculate these values we employ the scheduling algorithm proposed in [10], namely "Waiting-Assignment" algorithm. "Waiting-Assignment" is a distributed polynomial time algorithm to find the optimal waiting time of the nodes and hence Φ^D_{ψ} in tree ψ.

Our algorithm runs as follows. Given initial aggregation tree ψ and deadline D, we first run "Waiting-Assignment" algorithm to obtain Φ^D_{ψ}. Then, based on the underlying Markov chain design and in an iterative manner, we proceed to migrate to a target aggregation tree ψ' with (probably) better $\Phi^D_{\psi'}$ than Φ^D_{ψ}. To realize this end, each sensor node individually runs "Parent-Changing" algorithm which is summarized as Algorithm 1.

Algorithm 1: "Parent-Changing" algorithm for node $i \in \mathcal{V}$

Input: α, β
Output: New parent of node i

1 $P_i \leftarrow$ parent of node i
2 $\mathcal{N}_{\geq i} \leftarrow \{j : (i,j) \in \xi, W_j \geq W_i\}$
3 Node i generates a timer $\tau_i \sim \exp(\lambda_i)$ with mean $\lambda_i = \frac{1}{|\mathcal{N}_{\geq i}|}$ and starts to count down
4 When τ_i expires, node i randomly selects one of its neighbors $P'_i \in \mathcal{N}_{\geq i}$.
5 $\Phi_{\text{prev}} \leftarrow$ node i's estimation of Φ^D_{ψ} in Equation (14), i.e., the maximum QoA of the current tree
6 Node i changes its parent to P'_i
7 $\Phi_{\text{next}} \leftarrow$ node i's estimation of $\Phi^D_{\psi'}$ in Equation (14), i.e., the maximum QoA of the new tree
8 With the probability of $q_{\psi,\psi'}$, node i keeps the new tree configuration and with probability $1 - q_{\psi,\psi'}$ switches back and connects to the previous parent P_i
9 **if** *i changed its parent in Step 8* **then**
10 $\quad P'_i$ invokes "Waiting-Assignment" algorithm on its sub-tree
11 $\quad P_i$ invokes "Waiting-Assignment" algorithm on its sub-tree
12 Node i refreshes the timer and begins counting down

The detailed description of Algorithm 1 is as follows. In Line 3, an exponentially distributed random number with mean $\lambda_i = \frac{1}{|\mathcal{N}_{\geq i}|}$ is generated as the timer value in which this setting is required to ensure the convergence of the corresponding Markov chain. In Line 4, node i selects a new parent P'_i such that $W_{P'_i} \geq W_i$. This ensures that after the parent changing, the data structure still remains a tree. The point is that the new structure is not a tree only if node i chooses its new parent from its successors where all have a less waiting time than node i's waiting time. Meanwhile, this strategy is also rational because finding a new parent with a short waiting time declines node i's new waiting time which probably reduces QoA. In Lines 5-7, node i temporarily changes its parent and estimates the impact of this change on the maximum QoA of data aggregation. Based on the estimation and transition rate given by Equation (14), in Line 8, node i decides whether to keep its new parent or not. If the new state is established, then nodes P_i and P'_i should run "Waiting-Assignment" algorithm to update waiting time of their successors because of their sub-tree changes. It is worthy to note that the parameter β not only affects the accuracy of the approximation, but also with large values of β, the algorithm migrates towards better configurations more greedily, whereas it may lead to premature convergence and trap into local optimum trees.

Proposition 1. *"Parent-Changing" algorithm in fact implements a time reversible Markov chain with stationary distribution in Equation (11).*

Proof. The designed Markov chain is finite space state ergodic Markov chain where each tree configuration in state space is reachable from any other state by one or more parent changing process. We proceed to prove that the stationary state of designed Markov chain is Equation (11). Let $\psi \to \psi'$ denote transition from state ψ to ψ' at a timer expiration and $A = \frac{1}{\exp(\alpha)} \frac{\exp(\beta\Phi^D_{\psi'})}{\exp(\beta\Phi^D_{\psi}) + \exp(\beta\Phi^D_{\psi'})}$. Moreover, $\Pr(\psi \to \psi')$ is the probability of this transition.

This probability can be calculated as follows:

$$\begin{aligned} &\Pr(\psi \to \psi') \qquad (15)\\ &= \Pr(i \text{ chooses } P' | i\text{'s timer expires}).\Pr(i\text{'s timer expires})\\ &= \frac{1}{|\mathcal{N}_{\geq i}|}.A.\frac{|\mathcal{N}_{\geq i}|}{\sum_{j\in\mathcal{V}} |\mathcal{N}_{\geq j}|}\\ &= \frac{1}{\sum_{j\in\mathcal{V}} |\mathcal{N}_{\geq j}|}.A \end{aligned}$$

In the algorithm, node i counts down with rate $|\mathcal{N}_{\geq i}|$. Therefore, the rate of leaving state ψ is $\sum_{j\in\mathcal{V}} |\mathcal{N}_{\geq j}|$. We can calculate transition rate $q_{\psi,\psi'}$ as follows:

$$q_{\psi,\psi'} = \sum_{j\in\mathcal{V}} |\mathcal{N}_{\geq j}|.\frac{1}{\sum_{j\in\mathcal{V}} |\mathcal{N}_{\geq j}|}.A = A \qquad (16)$$

We can see that $p^*_{\psi}.q_{\psi,\psi'} = p^*_{\psi'}.q_{\psi',\psi}$. Therefore, the detailed balance equation holds and the stationary distribution of constructed Markov chain is Equation (11) [23]. □

"Parent-Changing" algorithm is distributed if we can estimate Φ_{next} and Φ_{prev} in the algorithm in a distributed manner. By exact calculation of these values, the designed Markov chain will converges to stationary distribution in Equation (11). Hence, "Parent-Changing" algorithm can give us a near-optimal solution of problem Z. However, exact calculation of Φ_{next} and Φ_{prev} is not possible in nodes locally since they can only be calculated in the sink by running "Waiting-Assignment" algorithm. Therefore, we need to estimate their values. We estimate the values by two different methods.

Approx-1: First method of estimating Φ_{next} and Φ_{prev}. When node i wants to modify its parent from P_i to P'_i (and subsequently tree ψ to ψ'), one possible way of estimation is running "Waiting-Assignment" algorithm by nodes P_i and P'_i on their sub-trees. Let $\Phi_{\text{prev}}[s]$ and $\Phi_{\text{next}}[s]$ denote the maximum achievable QoAs in a sub-tree rotted at node s respectively before and after the sub-tree change. Then, we have the following estimation:

$$\Phi_{\text{next}} \approx (\Phi_{\text{next}}[P_i] + \Phi_{\text{next}}[P'_i]), \qquad (17)$$

$$\Phi_{\text{prev}} \approx (\Phi_{\text{prev}}[P_i] + \Phi_{\text{prev}}[P'_i]). \qquad (18)$$

When node i changes its parent from P_i to P'_i, only sub-trees rooted at the P_i and P'_i change and all other parts of the tree remain intact and so the estimation accuracy is expected to be high. This estimation comes with the overhead of running "Waiting-Assignment" algorithm at nodes P_i and P'_i to calculate Φ_{next} and Φ_{prev}.

Approx-2: Second method of estimating Φ_{next} and Φ_{prev}. Another way of estimation is just using waiting times of nodes P'_i and i:

$$\Phi_{\text{next}} \approx W_{P'_i}, \qquad (19)$$

$$\Phi_{\text{prev}} \approx W_i. \qquad (20)$$

A larger value of $W_{P'_i}$ indicates that node i *probably* will be assigned a greater waiting time if it joins to sub-tree of P'_i and vice versa. In Section V, we evaluate the efficiency of both mentioned methods by simulation.

D. Perturbation Analysis

In "Parent-Changing" algorithm, if we obtain the accurate value of Φ^{ψ}_D to calculate transition rates, the designed Markov chain converges to the stationary distribution given by Equation (11). Thus, we have a near-optimal solution of problem Z with optimality gap determined in Equation (13). In distributed fashion, however, we estimate the optimal tree-specific QoAs by Equations (17)-(20). Consequently, the designed Markov chain may not converge to the stationary distribution in Equation (11). Fortunately, our employed theoretical approach can provide a bound on the optimality gap due to the perturbation errors of the inaccurate estimation using a quantization error model.

We assume that in a tree configuration ψ, the corresponding perturbation error is bounded to $[-\Delta_{\psi}, \Delta_{\psi}]$. In order to simplify the approach, we further assume that Φ^{ψ}_D takes only one of the following $2n_{\psi}+1$ values:

$$\begin{aligned} &[\Phi^{\psi}_D - \Delta_{\psi}, \ldots, \Phi^{\psi}_D - \frac{1}{n_{\psi}}\Delta_{\psi}, \Phi^{\psi}_D,\\ &\qquad \Phi^{\psi}_D + \frac{1}{n_{\psi}}\Delta_{\psi}, \ldots, \Phi^{\psi}_D + \Delta_{\psi}], \end{aligned} \qquad (21)$$

where n_{ψ} is a positive constant. Moreover, with probability $\eta_{j,\psi}$, the maximum quality of aggregation is equal to $\Phi^{\psi}_D + \frac{j}{n_{\psi}}\Delta_{\psi}, \forall j \in \{-n_{\psi}, \ldots, n_{\psi}\}$ and $\sum_{j=-n_{\psi}}^{n_{\psi}} \eta_{j,\psi} = 1$.

Let $\tilde{p}$ denote the stationary distribution of the *states* in the *perturbed* Markov chain [23]. We also denote stationary distribution of the *configurations* in the original and perturbed Markov chains by $p^* : \{p^*_{\psi}, \psi \in \mathcal{T}\}$ and $\bar{p} : \{\bar{p}_{\psi}, \psi \in \mathcal{T}\}$, respectively. Then, we have [23]

$$\tilde{p} \triangleq [\tilde{p}_{\psi,\Phi^{\psi}_D + \frac{j}{n_{\psi}}\Delta_{\psi}}, j \in \{-n_{\psi}, \ldots, n_{\psi}\}, \psi \in \mathcal{T}], \qquad (22)$$

$$\bar{p}_{\psi}(\Phi) = \sum_{j\in\{-n_{\psi},\ldots,n_{\psi}\}} \tilde{p}_{\psi,\Phi^{\psi}_D + \frac{j}{n_{\psi}}\Delta_{\psi}}, \forall \psi \in \mathcal{T}. \qquad (23)$$

Using total variance distance [22] we can measure the distance of p^*_{ψ} and $\bar{p}_{\psi}$ as

$$d_{TV}(p^*, \bar{p}) \triangleq \frac{1}{2}\sum_{\psi\in\mathcal{T}} |p^*_{\psi} - \bar{p}_{\psi}|. \qquad (24)$$

Theorem 3. a) *The total variance distance between p^*_{ψ} and $\bar{p}_{\psi}$ is bounded by $[0, 1 - \exp(-2\beta\Delta_{max})]$ where $\Delta_{max} = \max_{\psi\in\mathcal{T}} \Delta_{\psi}$.*

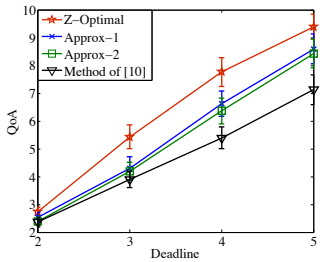


Fig. 3: Quality of aggregation vs. deadline ($V = 10$).

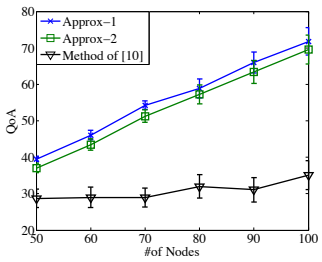


Fig. 4: Quality of aggregation vs. network size.

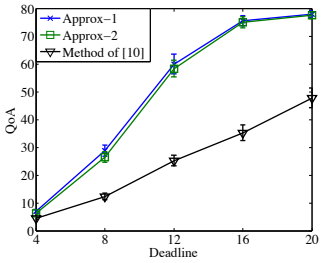


Fig. 5: Quality of aggregation vs. deadline.

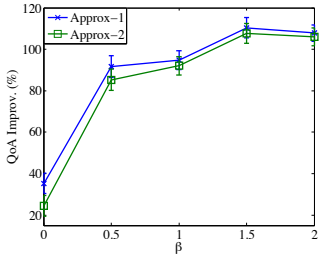


Fig. 6: Improvement of quality of aggregation vs. β.

TABLE I: Comparisons

Notation	Description
Z-Optimal	The optimal solution of problem Z implemented using exhaustive search.
Approx-1	Approximation algorithm that estimates the current QoA using Equation (17) and (18) in each node (has some overheads).
Approx-2	Approximation algorithm that estimates the current QoA using Equation (19) and (20) in each node (has no overheads).
Method of [7]	Optimal algorithm presented in [7] to maximize QoA in a given tree.

b) *By defining* $\Phi_{\max} = \max_{\psi \in \mathcal{T}} \Phi_{\psi}^{D}$, *the optimality gap in* $|p^* - \bar{p}|$ *is*

$$|p^* - \bar{p}| \leq 2\Phi_{\max}(1 - \exp(-2\beta\Delta_{max})). \tag{25}$$

For proof and remarks, we refer to [23].

V. Simulation Results

In this section, we evaluate our proposed algorithms through extensive simulations. Unless otherwise specified, the settings are as follows: 100 sensor nodes uniformly dispersed in a square field with side length of 300m. Sink node is located at the center of the top side of the square field i.e., its position is (150,300). Communication range of nodes is 75m, i.e., two nodes are connected in the network if their distance is "$\leq$ 75m". After deployment, sensor nodes construct an initial data aggregation tree based on Greedy Incremental Tree (GIT) algorithm [24]. We let $\alpha = 0.2$ and $\beta = 2$, and choose 80% of nodes randomly as sources. Each data point of the figures belongs to the average value of 50 runs with the 95% confidence interval where each run is a different random topology. Moreover, for each topology, sink imposes a deadline in terms of time slots uniformly and randomly selected from interval [10,20]. We report the results of approximation algorithms after 200 iterations where an iteration is defined as a timer expiration of a sensor node.

A. Performance Comparison with the Optimal Solution

In this part, we compare the performance of our algorithm to the optimal one. Due to the computationally intractable feasible solution space of large networks, we set up a small comparison experiment where 10 sensor nodes with communication range of 7.5m dispersed in a field with side length of 30m and sink ordination is (15,30). Moreover, due to small network size, we consider all sensors as the source nodes.

Fig. 3 portrays QoA of different methods against sink deadline. The main purpose is to compare our scheme with the optimal. The results show that "Approx-1" and "Approx-2" methods approximate the optimal solution with an accuracy of 87% and 84% on average, respectively. We believe that in real-world scenarios with the higher number of sensor nodes, the improvement is higher than that of the small scenario. To scrutinize this claim in more detail, we set up another set of experiments to investigate the improvements against various network sizes in the next subsection.

B. The Effect of Network Size

Fig. 4 depicts obtained QoA values for network sizes of 50 to 100 with step 10. The interesting result is that the improvement of approximation algorithms against "Method of [7]" significantly increases as network size grows. The reason lies behind the number of transitions in large networks. In large networks the corresponding Markov chain comes with the higher number of states and so the probability of more useful transitions is higher. Thus, higher improvement is expected in larger networks. This observation corroborates our claim that the improvements of our proposed algorithms are significantly better in large networks rather than small ones. The improvement by approximation methods to "Method of [7]" are at least 37% and 29% in all network sizes for "Approx-1" and "Approx-2", respectively. Moreover, the average improvements are 80% and 72%, respectively. "Method of [7]" shows little or no variation in obtained QoA value while network size grows.

C. The Effect of Deadline

We now study the effect of sink deadline on QoA. Based on Fig. 5, the trend is that QoA improves as deadline increases. This is in line with the fact that by increasing the deadline, more sensor nodes have the opportunity to participate in data aggregation. A notable observation is the small difference between "Approx-1" and "Approx-2". On average, "Approx-2" is 96% close to "Approx-1". This makes "Approx-1" as a good choice with respect to its small overhead and simplicity. Compared to "Method of [7]", "Approx-1" and "Approx-2" enhance QoA by 101% and 93% on average, respectively. The poor quality of "Method of [7]" is a result of neglecting the impact of data aggregation tree structure.

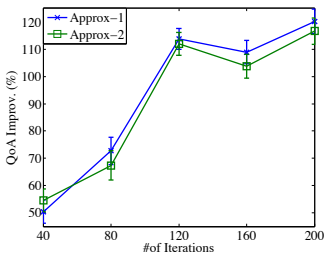


Fig. 7: Improvement of QoA vs. iteration numbers

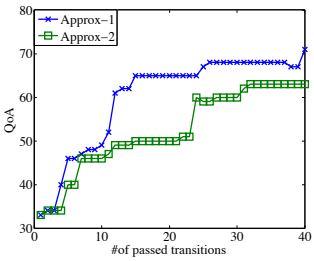


Fig. 8: Improvement of QoA for a random topology

D. The Effect of Parameter β

As it is stated in Section IV.A, the approximation gap theoretically decreases while β increases. We depict the effect of β by simulation in Fig. 6. Since "Method of [7]" is independent of the value of β, Fig. 6 only portrays the amount of improvement to "Method of [7]". By increasing β, in addition to achieving higher improvements, we observe that the improvement momentum of our schemes to "Method of [7]" degrades while β grows. This is a consequence of fast convergence of approximation schemes to the optimal where in the proximity of optimal solution improvements are smaller. The experimental results of Fig. 6 confirm the theory.

E. The Effect of Transition Numbers

Since the transition rates are set wisely to improve maximum QoA of the aggregation tree, we expect to obtain a better QoA as the number of transitions increases. Each transition can only occur after a node's timer expiration. However, all timer expiration does not lead to a transition. In Fig. 7, the effect of number of iterations (i.e., timer expiration) is shown for different iteration numbers. As it is expected, by increasing the number of iterations the efficiency of our schemes enhances. However, in one case when iteration numbers increased from 120 to 160, the improvements decreased. This can occur due to local optimum points in solution space. After more iterations, approximation methods leave this local optimum and converge to global optimal point. Finally, in a *microscopic* view in Fig. 8, we demonstrate the evolution of the maximum achieved QoA after each transition, i.e., migrating to a new aggregation tree, for a randomly selected sample topology.

VI. Conclusion

In this paper, we addressed the problem of constructing the maximum quality data aggregation tree in deadline-constrained WSNs. The objective is to maximize the number of nodes that the sink receives their data within a given application-specific aggregation deadline. We proved that the optimization problem is NP-hard and devised two distributed approximation algorithms. Observations on experiments corroborated our theoretical claim on the importance of constructing the optimal aggregation tree. Moreover, experimental results demonstrated that our methods not only achieve a close-to-optimal solution, but also significantly outperform the existing methods that rely on random underlying data aggregation trees. As an ongoing study, we plan to extend this work to multi-sink WSNs with unreliable links.

References

[1] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "An application specific protocol architecture for wireless microsensor networks," *IEEE Trans. on Wireless Communications*, vol.1, no. 4, pp. 660-670, 2002.

[2] B. Krishnamachari, D. Estrin, and S. B. Wicker, "The impact of data aggregation in wireless sensor networks," in *Proc. IEEE ICDCSW*, 2002.

[3] R. Rajagopalan and K.P. Varshney, "Data aggregation techniques in sensor networks: a survey," *IEEE Commun. Surveys Tutorials*, vol. 8, no. 4, pp. 48-63, 2006.

[4] S. Hariharan, Z. Zheng, and N. B. Shroff, "Maximizing information in unreliable sensor networks under deadline and energy constraints," *IEEE Trans. on Automatic Control,* pp. 1416-1429, 2013.

[5] H. Li, C. Wu, Q.S. Hua, and F. Lau, "Latency-minimizing data aggregation in wireless sensor networks under physical interference model," *Ad Hoc Networks*, vol. 12, pp. 52-68, 2014.

[6] X. Xu, X.-Y. Li, X. Mao, S. Tang, and S. Wang, "A delay-efficient algorithm for data aggregation in multihop wireless sensor networks," *IEEE Trans. on Parallel and Distributed Systems*, vol. 22, pp. 163-175, 2011.

[7] S. Hariharan, and N. B. Shroff, "Maximizing aggregated information in sensor networks under deadline constraints," *IEEE Trans. on Automatic Control*, vol. 56, no. 10, pp. 2369-2380, 2011.

[8] B. Alinia, H. Yousefi, M. S. Talebi, and A. Khonsari, "Maximizing quality of aggregation in delay-constrained wireless sensor networks", *IEEE Communications Letters* , vol. 17, no. 11, pp. 2084-2087, 2013.

[9] S. Nath, P. Gibbons, B. Phillip, S. Seshan, and R.Z. Anderson, "Synopsis diffusion for robust aggregation in sensor networks," in *Proc. ACM SenSys*, 2004.

[10] M. Chen, S. C. Liew, Z. Shao, and C. Kai, "Markov approximation for combinatorial network optimization," *IEEE Trans. on Information Theory*, vol. 59, no. 10, pp. 6301-6327, 2013.

[11] X. Chen, X. Hu, and J. Zhu, "Minimum data aggregation time problem in wireless sensor networks," in *Proc. IEEE MSN*, 2005.

[12] P.J. Wan, S.C.H. Huang, L. Wang, Z. Wan, and X. Jia, "Minimum-latency aggregation scheduling in multihop wireless networks," in *Proc. ACM MOBIHOC*, 2009.

[13] X.Y. Li, X. Xu, S. Wang, S. Tang, G. Dai, J. Zhao, and Y. Qi, "Efficient data aggregation in multi-hop wireless sensor networks under physical interference model," in *Proc. IEEE MASS*, 2009.

[14] S. Hariharan, and N. B. Shroff, "Deadline constrained scheduling for data aggregation in unreliable sensor networks," in *Proc. IEEE WiOpt*, 2011.

[15] H.X. Tan, M.C. Chan, W. Xiao, P.Y. Kong, and C.K. Tham, "Information quality aware routing in event-driven sensor networks," in *Proc. IEEE INFOCOM*, 2010.

[16] Y. Wu, Z. Mao, S. Fahmy, and N. B. Shroff, "Constructing maximum lifetime data-gathering forests in sensor networks" *IEEE/ACM Trans. on Networking*, vol. 18, no. 5, pp. 1571-1584, 2010.

[17] D. Li, J. Cao, M. Liu, and Y. Zheng, "Construction of optimal data aggregation trees for wireless sensor networks," in *Proc. IEEE ICCCN*, 2006.

[18] Y. Wu, S. Fahmy, and N.B. Shroff, "On the construction of a maximum-lifetime data gathering tree in sensor networks: NP-completeness and approximation algorithm," in *Proc. IEEE INFOCOM*, 2008.

[19] T. W. Kuo, and M.J. Tsai, "On the construction of data aggregation tree with minimum energy cost in wireless sensor networks: NP-completeness and approximation algorithms," in *Proc. IEEE INFOCOM*, 2012.

[20] H. Trent, "A note on the enumeration and listing of all possible trees in a connected linear graph," *Nat. Acad. Sci. U.S.A.*, vol. 40, no. 10, pp. 1004-1007, 1954.

[21] C. Chekuri, and A. Kumar, "Maximum coverage problem with group budget constraints and applications," *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, vol. 3122, pp. 72-83, 2004.

[22] P. Diaconis and D. Stroock, "Geometric bounds for eigenvalues of Markov chains," *The Annals of Applied Probability*, pp. 3661, 1991.

[23] S. Zhang, Z. Shao, M. Chen, and L. Jiang, "Optimal distributed P2P streaming under node degree bound," *IEEE/ACM Trans. on Networking*, vol. 22, no. 3, June 2014.

[24] B. Krishnamachari, D. Estrin, and S. Wicker, "Modelling data-centric routing in wireless sensor networks," in *Proc. IEEE INFOCOM*, 2002.