Efficient Aggregation of Delay-Constrained Data in Wireless Sensor Networks

Kemal Akkaya and Mohamed Younis
Dept. of Computer Science and Electrical Engineering
University of Maryland, Baltimore County
Baltimore, MD 21250
kemal1 | younis@cs.umbc.edu

Moustafa Youssef
Department of Computer Science
University of Maryland College Park
College Park, MD 20742
moustafa@cs.umd.edu

Abstract

Recent years have witnessed a growing interest in the application of wireless sensor networks in unattended environments. Nodes in such applications are equipped with limited energy supply and need careful management in order to extend their lifetime. In order to conserve energy, many of the routing protocols proposed for wireless sensor networks reduce the number of transmitted packets by pursuing in-network data aggregation. Almost all of the aggregation schemes presented in the literature strive to save sensor's energy while considering unconstrained data traffic. However, aggregation extends the queuing delay at the relay nodes and can thus complicate the handling of latency-constrained data. In this paper, we analyze the conditions for effective aggregation of data traffic that is subject to end-to-end delay constraints. We present an algorithm for achieving maximal possible energy saving through data aggregation while meeting the desired level of timeliness. A Weighted Fair Queuing based mechanism for packet scheduling is employed at each node in order to perform service differentiation and ensure bounded delay for constrained traffic. The performance of the proposed approach is qualified via simulation. The simulation results have demonstrated that the proposed approach provides a balance between energy consumption and timeliness level.

Keywords: Sensor networks, In-network data aggregation, Rea-time communication, Energy-aware design, QoS routing.

1 Introduction

Recent advances in microelectronics and low-power design have led to active research in large-scale, highly distributed systems of miniaturized wireless sensors [1]. Each sensor is capable of

measuring ambient conditions in its vicinity and reporting the sensed data on a radio channel. Over the last few years, the design of sensor networks has gained increasing attention due to their potential in a number of civil and military applications. For example, a network of sensors can be used to gather meteorological variables such as temperature and pressure. These measurements can be used in preparing forecasts or detecting harsh natural phenomena. In disaster management situations such as earthquakes, sensor networks can be used to selectively map the affected regions directing emergency response units to survivors. In combat zones, sensor networks can assist in surveillance missions and detect moving targets, chemical gases, or presence of micro-agents. Sensors in such applications are typically unattended and their batteries cannot be recharged. Therefore, energy-aware network operation becomes essential in order to extend the lifetime of the deployed sensors.

An important energy saving mechanism for sensor nodes is to exploit in-network data aggregation. In wireless sensor networks the raw sensed data is typically forwarded to a sink (gateway) node for processing. The main idea of in-network data aggregation is to eliminate unnecessary packet transmission by filtering out redundant sensor data and/or by performing an incremental assessment of the semantic of the data, e.g. picking the maximum temperature reading. Recent research on data aggregation in sensor networks focused on generating optimal aggregation trees for reduced energy consumption. The proposed mechanisms promote path sharing as much as possible and therefore trade increased per node queuing delay, and consequently boost the overall delivery latency, for further energy savings [2][3][4]. Moreover, significant attention has also been dedicated by the database community to the development of lightweight query languages and tool suite that enables task level analysis of the potential aggregation [5][6][7].

The increasing interest in the use of sensor networks in real-time applications introduces new challenges for the in-network data aggregation. For instance, in monitoring applications some queries that are subject to aggregation may require a bounded response time in order to ensure timely reaction to important findings. A typical query can be specified as "Report the average (maximum, minimum, etc.) measurement (temperature, pressure, radiation, etc.) in a certain region within D time units every T time units", where $D < T$. When processing such query, in-network data aggregation should not only be performed in an energy-efficient manner but should also achieve timeliness for some designated paths from the source nodes to the gateway. Other

applications of energy-efficient delay-constrained data aggregation include real-time target tracking in battle environments and critical relaying of after-shock events in disaster management environments.

In this paper, we consider sensor network setups that involve the collection of both real-time and non-real-time data. Under normal conditions data are routed in a best-effort manner with flexible latency bounds. Contemporary in-network aggregation techniques are employed in order to save communication-related energy consumption. Real-time data are generated and relayed to the gateway in response to delay-sensitive queries. While in-network aggregation of real-time data will still be highly desirable, the timeliness of the delivery of real-time packets has to be guaranteed. Meeting the timing constraints would require clever management of the packet queues at the relaying nodes in order to provide differentiated services based on the type of traffic. In-network data aggregation may extend the buffering time and thus may negatively impact the latency for delivering real-time packets. This paper investigates the problem of efficient in-network data aggregation of delay-constrained traffic in wireless sensor networks. We employ the Weighted Fair Queuing (WFQ) methodology [8][9][10] to schedule packets at relay nodes according to their type. WFQ enables the estimation of cumulative queuing delay on a path for non-burst traffic. We formally derive conditions for on-time packet delivery over an aggregation tree. We further present an algorithm that sets routes for maximal possible in-network data aggregation while meeting end-to-end delay constraints.

In the balance of this section we describe the sensor network architecture that we consider and summarize related work on in-network data aggregation. In section 2, we analyze the issues related to in-network aggregation of delay-constrained traffic and propose an algorithm for efficient aggregation of real-time packets. Section 3 discusses the algorithm's validation in a simulated environment and evaluates its performance. Finally we conclude the paper in section 4 with a summary and an outline of future research directions.

1.1 Sensor Network Architecture

A set of sensors is spread throughout an area of interest to monitor possible events in this area. The sensors are battery-operated with diverse capabilities and types and are empowered with limited data processing engines. The mission for these sensors is dynamically changing to serve the need of one or multiple command nodes. Command nodes can be stationary or mobile. A gateway node is a less energy-constrained node deployed in the physical proximity of sensors.

The gateway is responsible for organizing the activities at sensor nodes to achieve a mission, fusing data collected by sensor nodes, coordinating communication among sensor nodes and interacting with command nodes. The gateway node sends to the command node reports generated through fusion of sensor readings, e.g. tracks of detected targets. The command node presents these reports to the user and performs system-level fusion of the collected reports for overall situation awareness. In this paper we only consider stationary gateway and sensor nodes. All the sensors are assumed to be within the communication range of the gateway node. The architecture is depicted in Fig. 1.

The sensor is assumed to be capable of operating in an active mode or a low-power stand-by mode. The sensing and processing circuits can be powered on and off. In addition the radio transmitter and receiver can be turned on and off and the transmission power can be programmed for a required range. It is also assumed that the sensor can act as a relay to forward data from another sensor. Moreover, a sensor is assumed to switch between generating real-time and non-real-time data. It is worth noting that most of these capabilities are available on some of the advanced sensors, e.g. the Acoustic Ballistic Module from SenTech Inc. [11]. The gateway node is assumed to know its location, e.g. via the use of GPS. While the gateway will take charge of sensor organization based on the mission, we assume knowledge of which sensors need to be active in signal processing.

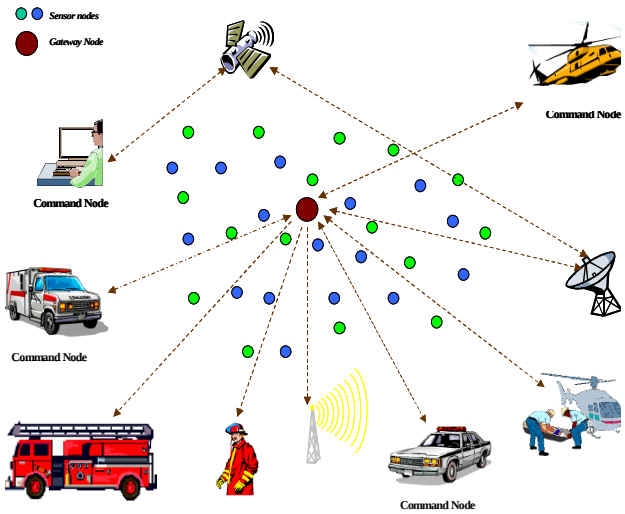

Fig. 1: Three-tier sensor network architecture

1.2 Related Work

Although significant efforts have been dedicated to performing data aggregation for sensor networks [2]-[7][12][13], to the best of our knowledge, no prior work has investigated the handling of end-to-end delay requirements for queries that utilize aggregate functions. Some researchers have looked at latency issues from different perspectives. For instance, the approach of Intanagonwiwat et al. [3] exploits latency and credibility trade-off in order to propose a

solution to the problem of "How long a node should wait before aggregating and sending its data to its parent?", where a parent denotes the next hop. The more it waits the better the credibility i.e. the significance of the result based on the number of contributing sensors. On the other hand, waiting too much can increase the end-to-end delay. However, they neither propose an algorithm to construct the aggregation tree nor consider a latency bound for the data. Our approach on the other hand constructs an aggregation tree for real-time packets and strives to meet the required bound on the end-to-end delay when two types of traffic coexist in the network.

Another in-network data aggregation scheme that aims at minimizing the end-to-end delay is proposed in [12]. This scheme does not consider any latency bound but tries to minimize the average end-to-end delay by concatenating multiple packets into one at the MAC layer. The idea is to limit the medium access contention so that the packet queuing delay will be reduced. Moreover, they use a feedback mechanism at each sensor node to adjust the number of concatenated packets based on the current traffic conditions. However, the proposed feedback mechanism is too complex for a resource constrained sensor node. Moreover, they just consider MAC layer and there is no optimization at network layer. When the concatenated packet is dropped then the recovery process will be very expensive, diminishing the energy and latency gains.

In [2], finding the optimal aggregation tree is modeled as a minimum Steiner tree problem. Since forming the minimal Steiner Tree is an NP-hard problem, three sub-optimal solutions were proposed. The suggested schemes include the Center at the Nearest Source (CNS), in which data is aggregated at the source nearest to the destination; Shortest Path Trees (SPT), where data is sent along the shortest path from source to gateway and aggregated at common intermediate hops along the way; and Greedy Incremental Trees (GIT), which builds an aggregation tree sequentially to merge paths. This work however does not deal with end-to-end delay constraints. Nonetheless, we use the SPT heuristic in our approach to build an initial aggregation tree, as we explain in the next section.

An interesting study that considers data aggregation subject to latency constraints is reported in [13]. The paper proposes packet scheduling algorithms for data gathering in real-time monitoring applications using a technique called modulation scaling. The idea is to model the transmission energy using the Quadrature Ampitude Modulation (QAM) scheme, which represents the energy consumption as a function of the modulation frequency. By adjusting the

bit rate at each node, significant energy saving can be achieved while keeping the latency in check. While this work considers a similar problem, our approach does not employ any modulation scaling techniques. Moreover, we consider mixed types of traffic which is not the case in their work.

2 Efficient Aggregation of Delay-Constrained Data

In this section we analyze the issues of performing in-network aggregation of delay-constrained traffic and describe our approach for addressing them. The following subsection serves as an extended problem statement. Section 2.2 summarizes how we support the routing of real-time data packets. Section 2.3 is dedicated to discussing our approach for building an aggregation tree that maximizes energy savings while ensuring timeliness.

2.1 Issues of Aggregating Constrained Traffic

Achieving the least energy consumption through data aggregation has been modeled as a minimum Steiner tree problem [2]. Given a graph $G = (V, E)$, where V is the set of vertices and E is the set of weighted edges, and a subset $S \subset V$ of required vertices, a Steiner tree is a sub-graph of G that includes all the vertices in S and has the minimum sum of weights. The Steiner tree of all the vertices when $S=V$, simply defines the minimum spanning tree of G. The links on the minimum Steiner tree, assuming a perfect aggregation of all packets, corresponds to the minimal number of packet transmissions and consequently the least communication energy.

When we consider mixed-type traffic where real-time and non-real-time traffic coexist, data aggregation becomes more challenging. In that case, we need to consider delay requirements for real-time data along with energy consumption of both real-time and non-real-time traffic. The problem can be stated as follows: Given a gateway and a set of sensors that can continually generate non-real-time data along with intermittent real-time data and an initial data aggregation tree for non-real-time data, propose a mechanism that strives to meet the end-to-end delay constraints of real-time data while providing maximal possible in-network data aggregation for both traffic. This problem can be modeled as finding a constrained Steiner tree, which achieves both minimum cumulative weights (energy consumption) and meets a certain bound (end-to-end delay) for each source-gateway pair. Delay-constrained minimum Steiner tree is shown to be NP-hard [15].

Since the aggregation tree will be built initially based on the flow of non-real-time data, the tree may need to be modified/re-established to handle the intermittently involved real-time traffic. There is a trade-off between performing in-network data aggregation and achieving timeliness. While using the initial data aggregation tree allows maximum path sharing and helps in reducing the number of packet transmissions, it boosts the queuing delay at relay nodes due to increased inbound traffic and waiting time for the arrival of the data packets to be aggregated. The increased queuing time can risk the timeliness of constrained traffic and thus can overshadow the energy savings of the in-network aggregation. Our approach, as we later explain, utilizes a service differentiation mechanism to control the queuing time and modifies the routes as needed in order to ensure timeliness. In the next subsections, we first describe the underlying service differentiation mechanism and then explain our approach for conducting efficient delay-constrained in-network data aggregation.

2.2 Supporting Real-time Packets

When the real-time traffic emerges, we need a service differentiation mechanism to handle both types of traffic and provide end-to-end guarantees for real-time data. In order to provide such service differentiation, we employ the Weighted Fair Queuing (WFQ) packet scheduling methodology at each node. WFQ has been shown to provide, in statistical term, an upper bound on path delay for a leaky bucket constrained flow [10]. In [16], we have used the WFQ packet scheduling technique for ensuring end-to-end delay bounds for time-constrained sensor data. Such solution however, does not consider any data aggregation at intermediate nodes enroute and all the processing is done at the gateway. In this paper, WFQ will be used along with data aggregation and activated at the point where real-time data is involved in the network. Before explaining our data aggregation mechanism, in this subsection we give a brief background on WFQ and how we employ it at the constrained sensor nodes.

To support real-time traffic, each sensor node applies a packet scheduling discipline that approximates Generalized Processor Sharing (GPS) [8]. GPS achieves exact weighted max-min fairness by dedicating a separate FIFO queue for each session (flow) and serving an infinitely small amount of data from each queue in a weighted round robin fashion. The packetized version of GPS is called Weighted Fair Queuing (WFQ). One interesting property of WFQ is that when combined with leaky bucket constrained sources, it can provide upper end-to-end delay bounds for each flow [9][10]. Assuming flow i is constrained by a leaky bucket with parameters (σ_i, ρ_i),

the maximum end-to-end delay (transmission + queuing delay) for a packet of flow i under WFQ, given in [10], is:

$$D(i) \le \frac{\sigma_i}{g(i)} + \sum_{m=1}^{M-1} \frac{P_{\max}(i)}{g_i^m} + \sum_{m=1}^{M} \frac{P_{\max}}{C} \qquad [1]$$

$$g_i^m = \frac{\Phi_i^m}{\sum_{j=1}^{n} \Phi_j^m} C \qquad [2]$$

- C is the link bandwidth
- σ_i is the maximum burst size for leaky bucket on flow i
- ρ_i is the average data rate of the flow i
- $P_{max}(i)$ *is* maximum packet size for flow i
- P_{max} *is* maximum packet size in the network
- g_i^m is the service rate on node m for flow i
- $g(i)$ is the minimum of all service rates for flow i
- M is the number of nodes on path of flow i
- Φ_i^m is the link share on node m for flow i

While WFQ is flow based, we use an approximation of WFQ by considering each real-time sensor node as a source of different real-time flow but with only one real-time queue at each relay node for the incoming packets of these multiple flows (Fig. 2) [16]. This model is used due to two reasons. First, having a different queue for each real-time flow will be inefficient in terms of the storage capacity of a sensor node. Second, the real-time flows are generated dynamically depending on the number of active real-time source sensors. Since the number of such flows can change during the sensing activity, having one queue will reduce the maintenance overhead.

The service ratio "r" is the bandwidth ratio and is used in allocating the amount of bandwidth to be dedicated to the real-time and non-real-time traffic on a particular outgoing link. This value is also used to calculate the service rate for each type of traffic on that particular node, with $r_m\mu$ and (1- r_m)μ being respectively the service rate for real-time and non-real-time data on sensor node m. In this case, r_m for the real-time queue on a node is the summation of link shares ($\Phi_1^m, \Phi_2^m, ..., \Phi_N^m$) of all real-time flows passing through that node as shown in Fig. 2. The link shares can be calculated directly from (2), $\Phi_i^m = \left(g_i^m * \sum_{j=1}^{N} \Phi_j^m \right) \Big/ C$. This r-value is calculated for each node at the gateway and then sent to the corresponding node by the gateway. It should be noted that r_m should be in [0, 1] for all links on a particular path so that the end-to-end

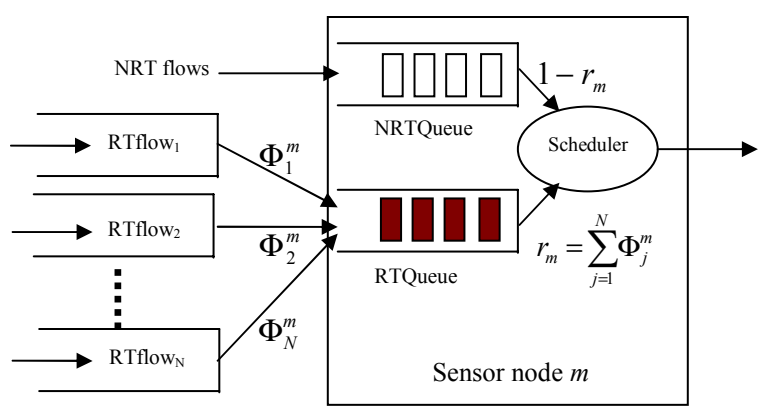

Fig. 2: Queuing model on sensors

delay constraint is met.

We argue that calculating the r-values and setting the routes is not a major burden in terms of energy and timeliness since it is handled by the gateway itself before the real-time packet generation has started.

2.3 Aggregation Tree for Real-time Traffic

As mentioned earlier, we consider a scenario where a sensor network application sets initial routes for data to be collected and aggregated periodically. However, at times the gateway queries certain regions for further critical information whose delivery would be subject to latency constraints. In our approach we first establish the initial routes to maximize the potential of in-network data aggregation of non-real-time traffic. We use the Shortest Path Trees heuristic, discussed in [2]. This heuristic tries to build a minimum weight Steiner tree by finding the shortest path between each source to the gateway and then combining the overlapping paths. In the absence of real-time traffic, in-network aggregation is performed without attention to packet delivery delay. It should be noted that we do not aggregate real-time flows with non-real-time flows.

Before the sensor nodes start generating real-time data in response to a request from the gateway, for each source our algorithm checks whether current routes can meet the delivery deadline or not. This is done by trying to find a valid *r-value* for each relay node on the path from a source to the gateway when both real-time and non-real-time traffic coexist in the network. Note that a valid *r-value* should be in [0, 1]. If the timeliness is violated, i.e. an *r-value* between 0 and 1 cannot be found, a new route should be searched. When searching for that route, the best alternative should be the one providing data aggregation the most and meeting the deadline. Therefore a node disjoint path will be the last option in the search process.

Let us call the longest path in terms of hops to the gateway in the region to be the main stream. The aim is to find a valid *r-value* for that route in order to meet the end-to-end delay bounds. Once such a valid *r-value* can be found, paths for other real-time sources should be checked. We call the route, if any, from each real-time source to this main stream as sub-branches and for each of these sub-branches we compute the *r-value* for a feasible delay-bounded path. If all the relay nodes on this sub-branch, from the source till the connection node to the main stream, have been found to have a valid *r-value*, we argue that all the source nodes generating real-time traffic and connecting the main stream will achieve the desired end-to-end

delay bounds. Before explaining the reasoning behind this argument, we introduce the two lemmas. The following notation is used in the balance of this subsection:

$P(s, G)$	:	Path from the source node s to the gateway G
r_s	:	r-value for node s
RT	:	Set of sources that generate real-time data
$\Psi(s)$	:	The number of hops from node s to the gateway
$D(s,G)$	:	End-to-end delay from node s to gateway
Γ	:	End-to-end delay bound for the application
Θ_j	:	The shared node for the path from source j to gateway
$\oplus$	:	Associative aggregation operator

Lemma 1: *If there are multiple real-time flows $f_1, f_2, \ldots, f_n$ sharing the same relay node i, they can be combined into a single flow f_i.*

Proof: Since the shared relay node has the ability to aggregate packets from the flows passing thought it, $Packet(i) \leftarrow Packet(f_1) \oplus Packet(f_2) \oplus \ldots \oplus Packet(f_n)$. However, all involved packets should be available at the time of aggregation. Therefore, each arriving packet from a flow is aggregated incrementally until processing all of them. Finally, the relay node will have only one packet, yielding one outgoing flow. □

Note that the incremental aggregation at the node will diminish the potential for buffer overflow since multiple packets would otherwise need to be stored in the buffer waiting for aggregation. Moreover, the nodes close to the gateway that are expected to be heavily loaded with real-time data will be relieved. By taking advantage of the decreased number of flows, we can guarantee end-to-end delay bounds for the sources sharing the same path on the aggregation tree. This also helps in reducing the number of transmissions in the network and hence provides significant energy savings for sensor nodes. We now prove how such end-to-end delay guarantee can be achieved by introducing the following lemma:

Lemma 2: *If there is a path $P(s_i, G)$ with* $0 \le r_s < 1 \;\; \forall s \in P(s_i, G)$ *and* $\Psi(s_i) = Max_{\forall j \in RT}(\Psi(s_j))$, *then*

$D(s_j, G) \le \Gamma \;\; \forall s_j \in RT - \{s_i\}$ *if* $\exists r_s \mid 0 \le r_s < 1 \;\; \forall s \in P(s_j, \Theta_j)$.

Proof: If each node on $P(s_i, G)$ has a valid r-value and $P(s_i, G)$ is the longest path from real-time sensors to the gateway, the other sources of real-time data will have sub-branches merging at some relay nodes on $P(s_i, G)$.The data packets generated at those sources will reach the relay (aggregator) nodes before the packets of s_i reach them since $\exists r_s \mid 0 \le r_s < 1 \;\; \forall s \in P(s_j, \Theta_j)$ and $P(s_i,G)$ is the longest path. The aggregator node will generate a single outgoing flow as proved with Lemma 1. Hence, the existing r-value will be sufficient to serve this single flow (see Fig. 3). Therefore, once a valid r-value can be found for every real-time source then the end-to-end delay requirements will be satisfied without any further actions. □

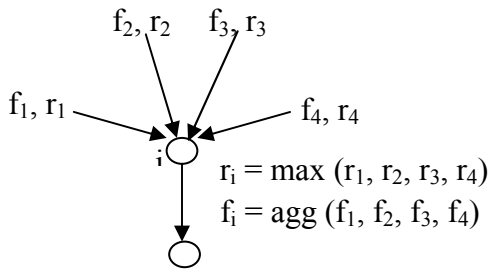

Fig. 3: Combining multiple flows into one with data aggregation.

In some circumstances we may not be able to find a feasible *r-value* for all the nodes on the longest path to the gateway. In that case, our algorithm designates the second longest path to the gateway as a potential main stream and checks for an appropriate *r-value*. This process continues until the main stream is identified. Then, the data aggregation process will be similar, based on Lemma 1 and 2. However, we need to look for alternative routes for packets pursuing disqualified paths, when a valid *r-value* cannot be found. In order to achieve timeliness, we consider less energy efficient paths from the affected sources to the gateway. It should be noted that path sharing is still desirable for in-network data aggregation when searching for the alternate routes. Therefore, we first try to connect to the main stream at a later point, i.e. at a relay node on the path that is closer to the gateway, by finding routes from the corresponding real-time sources to that relay. The process continues until a route with a valid *r-value* from each affected source to the gateway can be found. It should be noted that we may end up with a node disjoint path to the gateway when all the potential aggregation routes are explored.

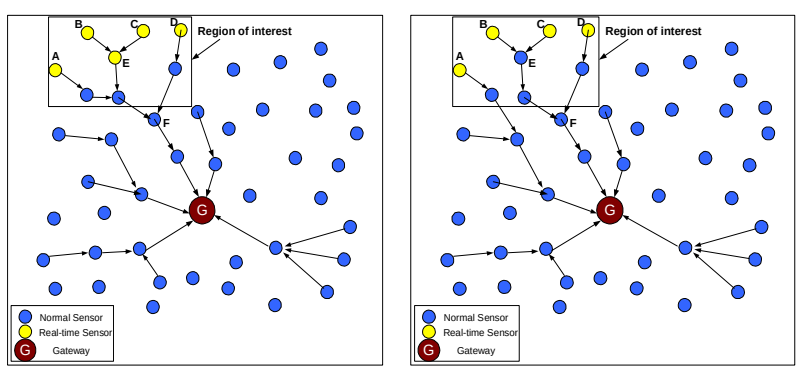

Fig. 4: a) Initial routes b) Adjusting the routes when real-time sources involved for timeliness of real-time packets.

A pictorial illustration of a sample situation is depicted in Fig. 4. We consider the route

from A to G the main stream initially. However, since an *r-value* cannot be found for the nodes on that path, we changed the main stream to B-G path. Node A had to find a node disjoint path to G in order to find feasible *r-value* for its nodes. Node C and D joins the main stream at the aggregation nodes E and F respectively.

Our proposed algorithm for handling aggregate queries such as min, max, average etc., in sensor networks in the presence of mixed-type traffic is outlined in Fig. 5. In lines 1-2, we form the initial aggregation tree. In line 3, we find the path with the maximum number of hops to the gateway and designate it as the main stream. When real-time data is involved, we need to find a feasible *r-value* for the main stream as shown in line 4. If it is not possible, then we pick the next longest route as the main stream until a feasible *r-value* can be found for the selected path to the gateway as shown in lines 5-8. After picking the main stream, we compute the *r-values* for the sub-branches connecting to this stream in lines 9-11. In lines 12-13, the *r-value* of an aggregator node is adjusted due to the reduction of multiple inbound flows to just one outgoing flow (Lemma 1). We then find alternative less energy efficient paths yet meeting the end-to-end delay bounds for the sources which could not find a valid *r-value* earlier, as shown in lines 15-19. This alternative path either tries to connect to the main stream at later points if possible or left as a node disjoint path to the gateway.

1 *Find the shortest path for each source to G*
2 *Combine the overlapping links to form the aggregation tree*
/* Determine the main stream to the gateway in the region
When the real-time data is involved */
3 $main\_stream \leftarrow \underset{\forall s_i \in RT}{Max}(\Psi(s_i))$
4 ***if*** $(\exists s_i \in main\_stream \mid r_{s_i} \notin [0,1])$
5 ***repeat***
6 $main\_stream \leftarrow$ next longest path
7 ***until*** $\forall s_i \in main\_stream \mid r(s_i) \in [0, 1]$
8 ***endif***
/* Find necessary r-values*/
9 ***for*** $(\forall s_i \in RT$ *sharing with main_stream)* ***do***
10 $\forall j \in P(s_i, G)$ *compute* r_j // To meet the latency
11 ***endfor***
/*Adjust r-values due to aggregation */
12 ***for*** *(each* Θ_k *on main_stream)* ***do***
13 $r_{\Theta_k} \leftarrow \underset{l \in children\ (\Theta_k)}{Max}(r_l)$ // max r-value of its children
14 ***endfor***
/*modify the routes that do not have feasible r-value initially */
15 ***for*** $(\forall s_i \in RT$ *such that* $\exists j \in (P(s_i, G) \mid r_j \notin [0, 1])$ ***do***
16 ***repeat***
17 *Find another less energy-efficient* $P_{new}(s_i, G)$
18 *Try connecting to the main_stream*
19 ***until*** $\forall j \in (P(s_i, G)) \mid r_j \in [0, 1]$
20 ***endfor***

Fig. 5: Pseudo code for data aggregation and re-computing *r-values*

3 Experimental Validation

The effectiveness of our approach is validated through simulation. This section describes the underlying network operation, simulation environment, performance metrics and experimental results.

3.1 Validation Setup

We have adapted the system architecture of [17] for validating our approach. The gateway electively engages some sensors in probing the surroundings based on missions that are assigned to the network. Unselected sensors can switch to a low-energy sleep mode. The gateway broadcasts the routing table prior to starting or resuming data transmission. The link cost is a function of the sender's energy reserve and the distance between the transmitter and receiver. Rerouting is triggered by either an application-related event that requires the involvement of different set of sensors, a need for a more efficient network topology or the depletion of the battery of an active node. During this process, the gateway runs the routing algorithm and sends new routes to each node and informs each sensor about its new state.

In the validation experiments, the network consists of varying number of sensor nodes (50 to 250) randomly placed sensors deployed in a 500×500 m^2 area. The gateway initial position is determined randomly within the area boundaries. A free space propagation channel model is assumed [18] with the capacity set to 2 Mbps. The size of a data packet is 10 Kbit [11]. Each node is assumed to have an initial energy of 5 joules. A node is considered nonfunctional if its energy level reaches 0. The transmission range for a sensor node is assumed to be 100m [19]. We assumed a link error rate of 0.01. For a node that is actively sensing the environment, packets are generated at a constant rate. Each packet has an energy field that is updated during the packet transmission to calculate the total energy consumption in the network. In the experiment we assume that the network is tasked with a habitat monitoring application, where all sensors are actively probing the environment reporting delay-unconstrained data. Intermittently the gateway queries a random region getting some nodes (about 10% of the deployed sensors) to generate real-time data.

While each node is generating non-real time packets periodically to send to the gateway, the set of sensing nodes for real-time data exchange is selected based on a delay-constrained query specification in a certain region of the deployment area. These nodes are called real-time sensors and they generate data for a certain amount of time during the simulation. Packets, generated by

such sensors, are labeled as of real-time type and treated differently at the relaying nodes. The *r*-value is initially assumed to be 0 and is recalculated as real-time sensing begins. The default end-to-end delay requirement for real-time data is taken to be 0.08 sec [20]. We assume perfect aggregation, i.e. multiple packets can be combined into one after aggregation.

3.2 Performance Metrics and Results

The goal of the performance experiments is to qualify the impact of aggregation of real-time data on both energy and timeliness metrics and to capture the effect of traffic density and network size on how our approach performs. As a base for comparison, we have used the initial SPT aggregation algorithm without considering any service differentiation and packet scheduling at the nodes. In addition, we compared our WFQ based aggregation approach to the one without considering any aggregation at the nodes. In the graphs SPT indicates the baseline approach, WFQ indicates the approach employing packet scheduling but no aggregation and WFQ-AGG refers to our algorithm. We have used the following performance metrics:

- *Deadline Miss Ratio*: This is one of the most important metrics in real-time applications, which indicates the number of packets that could not meet the specified delivery deadline.
- *Total energy*: This indicates the total energy consumption in transmission and reception of all packets in the network. This metric along shows how efficient the algorithm is with respect to energy consumption.

We have applied 5 distinct seeds in order to generate random network topologies. Separate simulation experiments were performed for each topology. Each simulation lasted 2000 sec. We observed that with 90% confidence level, the simulation results stay within 6%-10% of the sample mean.

<u>Timeliness:</u> We have measured the effect of aggregation on timeliness. Recall that data aggregation can increase the queuing delay at relay nodes since it strives to combine paths and waits for the arrival of all packets to be aggregated. Figures 6 to 8 compare the rate of missing packet delivery deadline for our algorithm to that achieved by both SPT and WFQ-AGG. In Fig. 6, we have fixed the network size and varied the packet generation rate measured in terms of the inter-packet time (round). At low rate the queuing delay introduced by aggregation becomes negligible and our algorithm performs similar to the baseline case of no aggregation of real-time data. At high rate the queuing delay becomes excessive and hurts timeliness. We observe that at

both low and high rates, SPT causes most of the real-time packets to miss their deadlines since WFQ packet scheduling is not employed.

Figures 7 and 8 capture the effect of the size of the network on timeliness. For both high and low data generation rates, WFQ and WFQ-AGG perform significantly better than SPT in terms of hit ratio for real-time data because of the similar reason stated above. When comparing WFQ and WFQ-AGG, for high rates, the increase in miss rate magnifies for larger network sizes as shown in Fig. 7. This is very much expected as hinted in Fig. 6. However, it is worth noting that for small networks the impact of the generation rate on real-time packet is insignificant. On the other hand, at low rates the effect of aggregation on timeliness is almost unfelt where Fig. 8 shows the performance of the WFQ and WFQ-AGG is very similar. Finally, the miss rate grows for larger network sizes.

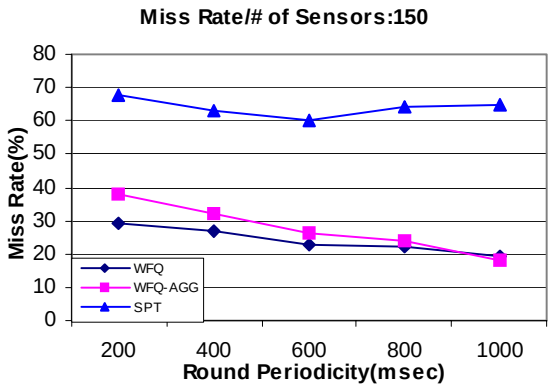

Fig. 6: Miss rate for real-time packets for varying traffic density.

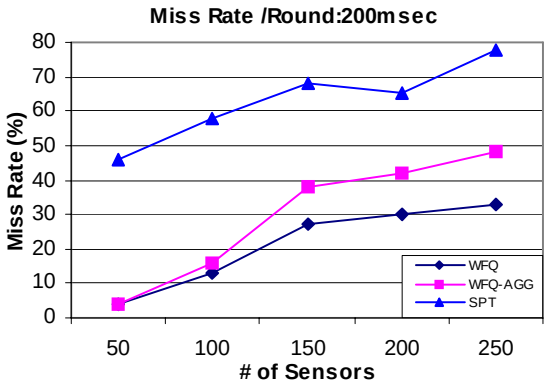

Fig. 7: Miss rate for real-time packets under different number of sensors (high rate)

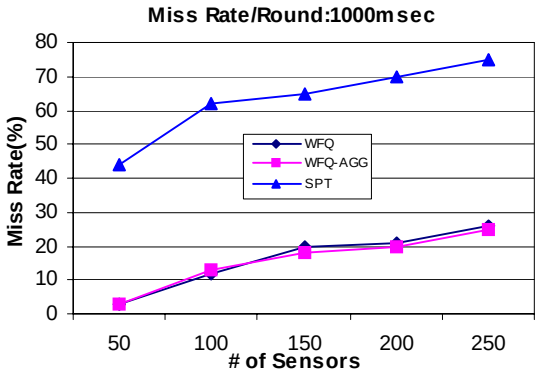

Fig. 8: Miss rate for real-time packets under different number of sensors (low rate)

<u>Energy:</u> Looking at the total energy consumption in the network, it can be observed that our approach, as expected, conserves significant energy compared to WFQ. Moreover, WFQ-AGG's total energy consumption is very close to the baseline (SPT) where maximum aggregation, and consequently maximum energy, saving is possible. Fig. 9 shows that our aggregation mechanism helped in achieving about 30% additional energy saving when compared to WFQ. Such positive impact is more obvious when the number of sensors is high. With the increasing number of

sensors, more path sharing and hence more in-network aggregation is possible. Since processing-related energy consumption is much less than wireless communication, in-network data aggregation provides a significant reduction in the number of transmissions in the network. Therefore, more energy savings are possible as the network scales. Similar observation can be made regarding the packet generation rate as seen in Fig. 10. For high packet density the energy savings grows in significance compared to the WFQ.

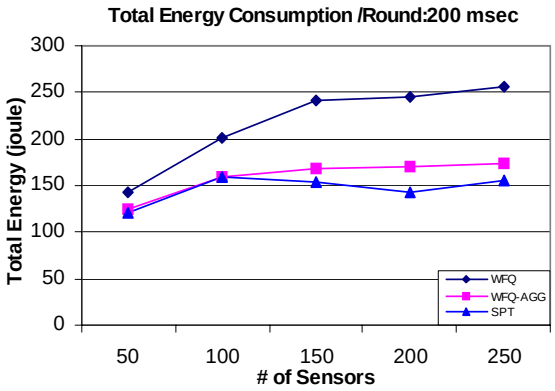

Fig. 9: Total energy consumption in the network under varying number of sensors.

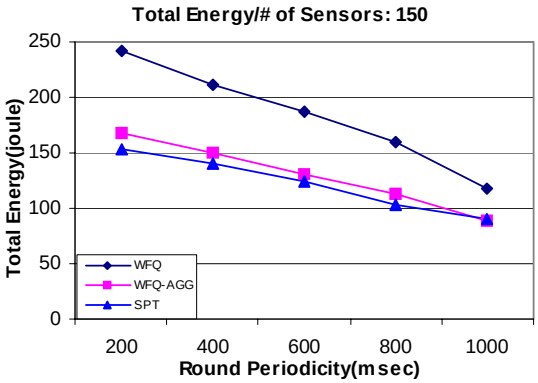

Fig. 10: Effect of increased packet generation rate on total energy consumption.

Collective analysis of all performance graphs indicates that there is a trade-off between the level of energy conservation and timeliness. Noting that we try to stick to a tree that maximizes the potential for aggregating non-real-time data, timeliness of constrained traffic is always going to be hard to achieve under heavy traffic. We thus envision that an application level analysis would be necessary to qualify the importance of timeliness in comparison to energy conservation. For large networks or under high rate of real-time packets, it may be necessary to sacrifice the level of aggregation of contemporary traffic.

4 Conclusion

In this paper, we have presented an efficient approach for providing timeliness in sensor networks when in-network data aggregation is utilized. The proposed approach initially forms an aggregation tree that suits contemporary best-effort traffic and utilizes WFQ in order to support on-time delivery of delay-constrained (real-time) data. The idea is to identify the longest path in terms of hop counts on the aggregation tree for which the end-to-end delay is acceptable. A work around mechanism is presented to ensure timeliness of packets on unfeasible paths by adjusting

the tree so that the packets are aggregated at another relay node that is closer to the gateway (sink) node.

We analytically proved that when a feasible path is found for the longest path in terms of hop counts among the real-time sources, the other sources connecting to this longest path can meet the end-to-end delay bounds. The effectiveness of the proposed approach is verified through simulation. Simulation results show that our approach provides significant increase in timeliness at the price of a slight increase in energy consumption when compared to non-QoS-aware aggregation. The approach maintains the same level of timeliness for low traffic rates and slightly increases deadline misses for reasonably higher rate. Our future plan includes extending the presented framework to support mobile nodes and exploring the effect of complex aggregation operators on timeliness and energy.

References

[1] I. F. Akyildiz, et al., "Wireless sensor networks: a survey", *Computer Networks*, Vol. 38, pp. 393-422, March 2002.

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

[3] C. Intanagonwiwat, D. Estrin, R. Govindan, and J Heidemann, "*Impact of Network Density on Data Aggregation in Wireless Sensor Networks*", in Proceedings of the 22nd International Conference on Distributed Computing Systems, Vienna, Austria, July 2002.

[4] Yuan W., Krishnamurthy S.V., and Tripathi, S.K., "Synchronization of Multiple Levels of Data Fusion in Wireless Sensor Networks," *IEEE GLOBECOM* 2003.

[5] S. Madden, M. Franklin, J. Hellerstein, W. Hong, " TAG: A Tiny Aggregation Service for ad hoc Sensor Networks," OSDI Conf., Boston, December 2002.

[6] Mohamed A. Sharaf, Jonathan Beaver, Alexandros Labrinidis and Panos K. Chrysanthis, TiNA: A Scheme for Temporal Coherency-Aware in-Network Aggregation, *ACM MobiDE'03*, San Diego, CA,2003.

[7] A. Demers et al, "The Cougar Project: A Work-in-Progress Report," *ACM SIGMOD Record*, vol. 34, no. 4, Dec. 2003.

[8] A. K. Parekh and G. Gallager, "A generalized processor sharing approach to flow control in integrated services networks: The single-node case", *IEEE Trans. on Networking*, vol. 1, no. 3, pp. 344-357, June 1993.

[9] A. Demers et al., “Analysis and simulation of a fair queuing algorithm” in *Journal of Internetworking Research and Experience*, pp 3-26, October 1990.

[10] A. K. Parekh and G. Gallager, “A generalized processor sharing approach to flow control in integrated services networks Services Networks: The Multiple Node Case”, in the *Proceedings of IEEE INFOCOM 1993*.

[11] Data sheet for the Acoustic Ballistic Module", SenTech Inc., http://www.sentech acoustic.com/

[12] T. He, B.M.Blum, J. A. Stankovic, T. F. Abdelzaher, "AIDA: Adaptive Application Independent Aggregation in Sensor Networks", in *Special issue on dynamically adaptable embedded systems, ACM Transaction on Embedded Computing System*, 2003

[13] Yang Yu, Bhaskar Krishnamachari and Viktor K. Prasanna, "Energy-Latency Tradeoffs for Data Gathering in Wireless Sensor Networks," in *IEEE Infocom* 2004.

[14] J. Hou and B. Wang, "*Multicast routing and its QoS extension: Problems, algorithms and Protocols*," IEEE Networks, Jan./Feb. 2000.

[15] V. P. Kompella et al., “Multicast Routing for multimedia communication,” IEEE/ACM Transactions on Networking, pp. 286-292, 1993.

[16] K. Akkaya and M. Younis, "Energy-aware routing of time-constrained traffic in wireless sensor networks," in the *International Journal of Communication Systems,* Special Issue on Service Differentiation and QoS in Ad Hoc Networks (to appear)

[17] M. Younis et al.,, “Energy-Aware Routing in Cluster-Based Sensor Networks”, in the *Proceedings of IEEE/ACM MASCOTS2002*, Fort Worth, Texas, October 2002.

[18] J.B. Andresen et al., “Propagation Measurements and Models for Wireless Communications Channels,” *IEEE Communications Magazine*, Vol. 33, No. 1, January 1995.

[19] A. Chandrakasan, et al., "Power Aware Wireless Microsensor Systems", Keynote Paper ESSCIRC, Italy, 2002.

[20] K. Danilidis et al., “Real-time Tracking of Moving Objects with an Active Camera,” Real-Time Imaging Journal, Vol. 4, No.1 pp.3-20. February 1998.