Feedback Control of Data Aggregation in Sensor Networks *

Tarek Abdelzaher, Tian He, John Stankovic
Department of Computer Science, University of Virginia, Charlottesville, VA 22903
{zaher,tianhe,stankovic}@cs.virginia.edu

Abstract

Sensor networks have recently emerged as a new paradigm for distributed sensing and actuation. This paper describes fundamental performance trade-offs in sensor networks and the utility of simple feedback control mechanisms for distributed performance optimization. A data communication and aggregation framework is presented that manipulates the degree of data aggregation to maintain specified acceptable latency bounds on data delivery while attempting to minimize energy consumption. An analytic model is constructed to describe the relationships between timeliness, energy, and the degree of aggregation, as well as to quantify constraints that stem from real-time requirements. Feedback control is used to adapt the degree of data aggregation dynamically in response to network load conditions while meeting application deadlines. The results illustrate the usefulness of feedback control in the sensor network domain.

1. Introduction

The work reported in this paper is motivated by the rapid emergence of sensor networks [4] as a new paradigm for writing distributed applications. These networks are composed of a large number of small wireless sensor devices, each equipped with limited processing, communication, and storage capacity. Sensor networks are especially useful in applications involving a poorly accessible, dangerous, or unfriendly environment, where it is difficult to provide a fixed monitoring infrastructure. Instead, a myriad of wireless sensor devices can be deployed (e.g., by air-dropping from a UAV) for remote monitoring and surveillance purposes. Such air-dropped networks are called *ad hoc* sensor networks to distinguish them from other types of sensor networks where nodes are laid out in some fixed predetermined pattern. Ad hoc wireless sensor networks present the most challenge to the research community due to their inherent lack of structure. Example applications include habitat monitoring, defense, border control, and emergency response systems.

This paper introduces fundamental research challenges presented by ad hoc wireless sensor networks from a feedback control perspective. These challenges lie in optimally reconciling the fundamental performance trade-offs that underlie network operation. Hence, the purpose of feedback control in this paper is not to control the dynamics of an external environment, but rather to control network performance itself. At a high level, performance of a sensor network can be viewed as a point in a three-dimensional space. These dimensions are (i) timeliness, (ii) energy consumption, and (iii) information output. It is desired to minimize energy consumption and maximize information output while maintaining timeliness. These requirements are mutually at odds; communicating more information takes more time and consumes more energy.

The nature of the trade-off among the basic sensor network performance requirements depends on current network input, which is the amount of sensory data infused into the network. For example, at low network load, timeliness can be easily achieved together with the other requirements. However, at a higher load, a decision has to be made between timeliness of delivery and the amount of deliverable information. Feedback control loops are needed to trade-off these performance requirements dynamically in a distributed fashion in response to current network conditions, essentially solving a distributed constrained optimization problem. This paper describes an instance of such a feedback control architecture, and derives some results in real-time computing that help quantify the constraints imposed on optimization.

The performance trade-offs mentioned above are fundamentally inherent to ad hoc sensor networks because they invariably arise from the main goal of such networks, namely the collection of sensory data. The most important output of a sensor network is the information it provides to external observers. One of the most limited resources in an ad hoc network is battery capacity. This is partly because advances in battery capacity have developed at a slower rate than advances in processing and communication bandwidth. Moreover, since the network is typically deployed in remote or harsh environments, changing batteries is quite expensive if not infeasible. Hence, maximizing battery lifetime by conserving energy is a predominant concern.

Omni-directional communication is the most energy-consuming operation in a sensor network due to the high degree of signal attenuation and the multipath phenomena that occur when wireless sensors are placed on the ground. Directional communication remains a big challenge since it requires sensors to know to a high degree of accuracy both their own position and orientation, as well as that of their neighbors. The fundamentally high cost of com-

*The work reported in this paper was supported in part by the National Science Foundation under grants CCR-0093144, ANI-01 05873, EHS-0208769, and MURI N00014-01-1-0576.

munication results in an important trade-off between the amount of information that the network delivers and its lifetime. A good compromise to conserve battery capacity is to perform appropriate aggregation on collected data to reduce the amount of network communication without much reduction in the information delivered.

Timeliness of delivery is a fundamental performance concern in sensor networks because such networks must react to external phenomena in real-time. An unbounded delay in the loop is unacceptable. From an application's perspective, discovering an intruding target too late is not useful for producing an effective response. Timeliness is generally at odds with energy consumption. If it is possible to delay delivery until more data can be aggregated, overhead can be saved and delivery energy can be reduced. While limited aggregation (or batching) may actually improve the overall timeliness by reducing total traffic, additional aggregation will impair timing performance due the introduced aggregation delay. The break-even point depends on the amount of data currently generated, which is a dynamic quantity that depends on activities in the environment. The timeliness-energy trade-off therefore opens a realm of opportunity for feedback control research in the sensor network domain.

An important consideration in the design of a feedback performance control framework for sensor networks is to quantify the fundamental constraints within which each sensor node operates to solve the global performance optimization problem. In the three-dimensional trade-off space introduced earlier, the basic constraints are on energy, time, and information content. To address the timing constraint, in this paper, we describe important recent results in real-time computing theory that quantify the ability of the network to communicate data in real-time. We relate global timing requirements to the local amount of traffic that can be processed by each node.

As a specific instance of performance control in sensor networks, this paper describes a data communication and aggregation framework that manipulates the degree of data aggregation to maintain specified acceptable latency bounds on data delivery while attempting to minimize energy consumption. An analytic model is constructed to describe the relationships between timeliness, energy, and the degree of aggregation, as well as to quantify constraints that stem from real-time requirements. Feedback control is used to adapt the degree of data aggregation dynamically in response to network load conditions while meeting application deadlines.

The rest of this paper is organized as follows. Section 2 describes the problem statement and the general architecture of our service, which is based on two types of data aggregation; *lossy* and *lossless*. Section 3 derives an expression for real-time system capacity that quantifies the amount of information that can be delivered through the network by the deadlines. This bound is a fundamental design constraint that must be enforced by the feedback control architecture. Section 4 investigates feedback control of lossy aggregation. It describes the conditions under which system capacity is maximized, and describes a feedback scheme that optimizes capacity subject to time constraints by adjusting the degree of aggregation. Section 5 describes local optimization using feedback control of lossless aggregation. Section 6 presents a brief performance evaluation. The paper concludes with final observations and open questions in Section 7.

2. Problem Statement and Architecture

We consider a real-time sensor network where sensory measurements should be delivered to their destinations within specified time constraints. Data is divided into multiple classes. Each class is associated with a bound on delivery time. For example, motion sensor measurements might have to be delivered within 3 seconds to allow real-time tracking of moving targets. In contrast, temperature measurements could be delivered within 30 seconds, since they exhibit slower dynamics. It is desired to deliver all data at the minimum energy cost while satisfying all time constraints. Since the environment is dynamic, the amount of data generated at any time is unpredictable and can vary considerably from time to time. We assume that some sensors report their measurements periodically at all times, while others become active only when triggered by environmental events. For example, flurries of activity in the monitored environment may generate a burst of motion sensor readings. These sensors will be silent when the environment is quiescent. Since the network load is dynamic, overload may occur which can significantly increase communication delay, possibly making it infeasible to deliver data in time. A feedback mechanism is needed to control network delay such that time constraints are met.

The main actuator "knob" that can be manipulated in our system is the degree of data aggregation. In contention-based Medium Access Control (MAC) communication protocols, packing data into larger units reduces the chances of packet collisions, hence reducing energy expenditure and improving delay.[1] Two different types of aggregation are possible; namely, *lossless* aggregation and *lossy* aggregation. Lossless aggregation refers to concatenating individual data items into larger packets, thus amortizing per-packet protocol overhead. In this case, no data is lost. The approach is especially effective in sensor networks where individual sensor readings are small in size, leaving much room for concatenation. Another type of overhead that can be amortized is the local handshake performed ahead of per-hop data transmission to reserve the channel. This handshake is common to contention-based wireless MAC protocols such as 802.11.

Lossless aggregation is effective if the load on the system is not excessive. If the total communication load approaches system capacity, the amount of communicated data must be forcibly reduced. We call the latter case, lossy aggregation. This technique is also useful for energy saving, even when the system is not heavily loaded. The best example of lossy aggregation is the averaging of sensor values. Averaging is a natural choice in many applications. For example, a user may need to know only of the average temperature in a region, as opposed the individual readings of all sensors. Similarly, it may be enough to report only the average estimated location of a target, as opposed to the exact locations of all triggered motion sensors. Lossy aggregation can be either spatial or temporal. In the former, data is averaged from multiple sensors, while in the latter, data from the same sensor is averaged

[1] Due to the difficulty in synchronizing clocks across all nodes in a sensor network, slot-based communication protocols are less practical than contention-based ones.

over time. Both spatial and temporal aggregation incur additional delay waiting for all needed data items to arrive before aggregation is performed.

The service described in this paper adaptively determines the type and amount of aggregation required such that time constraints are met. To maximize information output, lossless aggregation is performed as long as the workload is less than system capacity. Lossy aggregation is invoked only when capacity is exceeded. Two separate control loops are used to determine the amount of aggregation to be applied of each type. Note that, in applications where some degree of lossy aggregation is appropriate even at low load, a lower limit can be imposed on the lossy aggregation controller output. This limit ensures that the desired degree of aggregation is always carried out, even when the system is not overloaded.

A key to the correct operation of the system is to quantify system capacity, such that the correct type of aggregation is used in accordance with load conditions. This quantification is described below, followed by a description of both the lossless and lossy aggregation feedback loops.

3. Real-Time Capacity

The first function of the control system is to decide on the type of aggregation performed (lossy or lossless), depending on whether the network is overloaded or not. In this section, we define a notion of network capacity that is relevant to real-time applications, and relate satisfaction of end-to-end time guarantees to the local state of individual nodes.

3.1 Capacity Definition

Traditional notions of network capacity [2] quantify the amount of information that can be transmitted through the network concurrently at any point in time. In wireless networks, this amount is usually expressed as a product of bytes and meters (called byte-meters) since more data can be transmitted less distance or vice versa. These definitions have no notion of delivery latency and are therefore less suitable for applications where data that arrive after their deadline expiration have little or no value.

In this paper, we define a new notion of capacity we call, *real-time capacity*, denoted C_{RT}. Real-time capacity refers to the total byte-meters that can be delivered *by their deadlines*. To make the capacity expressions independent of the details of the workload (such as the deadline values themselves), we are interested in a normalized capacity expression that quantifies the total byte-meters that can be delivered for per unit of requested latency. It is expected that a network can have a larger byte-meter capacity if deadlines are larger, which makes the aforementioned (normalized) notion of real-time capacity more meaningful.

To illustrate the notion of real-time capacity, consider a network with two data flows, A, and B. Flow A must transfer 1000 bytes a distance of 50 meters (i.e., a total of $50,000$ byte-meters) within 200 seconds. It is said to have a real-time capacity requirement of $50,000/200 = 250$ byte-meters/second. Flow B must transfer 300 bytes a distance of 700 meters within 100 seconds. Its capacity requirement is thus $300 * 700/100 = 2100$ byte-meters/second. Hence, the total real-time capacity needed is $2100 + 250 = 2350$ byte-meters/second. Below, we establish an approximate capacity bound that quantifies the ability of the network to transfer data in time. In particular, all flows meet their deadlines as long as their collective capacity requirements do not exceed the derived capacity bound. This bound will be used to determine whether or not a system is overloaded for the purposes of applying the corresponding data aggregation technique.

3.2 Capacity Derivation and Sampling Rate

Consider a sensor network of n nodes with multiple data sources and a single data sink. The sink could be a monitoring workstation, or a relay that sends the collected data to a user. Packets traverse the network concurrently, each following a multihop path from some source to the sink. Each packet T_i has an arrival time A_i defined as the time at which the sending application injects the packet into the outgoing communication queue of its source node. The packet must be delivered to its destination no later than time $A_i + D_i$, where D_i is called the relative deadline of T_i. Different packets may generally have different deadlines. We call packets that have arrived but whose delivery deadlines have not expired *in-transit* packets. Each packet T_i has an average transmission time C_i that is proportional to its length. Any single path through the network can be thought of as a data pipeline of N stages, where N is the number of hops along the path. In a prior publication [1], we have shown that data traversing a pipeline will meet its end-to-end deadline as long as the following condition holds:

$$\sum_{j=1}^{N} \frac{U_j(1 - U_j/2)}{1 - U_j} < \alpha \qquad (1)$$

where $U_j = \sum_i C_i/D_i$ over all packets T_i in transit through node j. This quantity is called the *synthetic utilization* of node j. The parameter α depends on the scheduling policy used to order outgoing packet transmissions on the link, as discussed in [1]. The bound was derived for nodes with dedicated links. Since, in the case of contention-based protocols, the link is shared with neighboring nodes, the average packet transmission time, C_i must account for this sharing. In particular, with m neighbors, on average, only $1/m$ of link bandwidth can be used by any one node when all nodes are sending. Hence, the average packet transmission time is correspondingly increased m times to account for channel sharing. This is reflected in the values of U_j used in Equation (1). Next, we derive the real-time capacity bound in the presence of lossless aggregation. We use that bound to determine the sensor sampling rate that can be supported during normal operation.

Consider the case where aggregation is lossless. If all traffic congregates on one sink, in the absence of lossy aggregation, the total schedulable traffic generated by all sources is exactly the traffic that can be consumed by that sink. Moreover, at steady state, the sum of synthetic utilizations on all hops some fixed distance j from the sink is no larger than the total synthetic utilization at the sink. This is because the total flow of packets crossing a given perimeter cannot exceed what the destination sees, as shown in Figure 1. Observe that, assuming uniform node density, the number of nodes on a perimeter of radius j (hops) away from the destination increases approximately linearly with j. Hence, the average per-node synthetic utilization decreases linearly with distance from the destination. Assuming the synthetic utilization at the destination is U, and renumbering the hops in ascending order from destination to sources, U_j is proportional to U/j. Thus, from Equation (1):

$$\sum_{j=1}^{N} \frac{U/j(1-U/2j)}{1-U/j} < \alpha \tag{2}$$

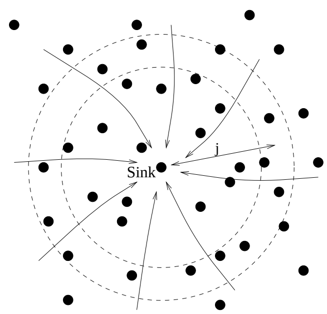

Figure 1. The single sink case

The above equation can be solved for U as a function of the average number of hops N. The equation can be rewritten as:

$$\frac{U}{2}\sum_{j=1}^{N} \frac{1}{j-U} + \frac{U}{2}\sum_{j=1}^{N}\frac{1}{j} < \alpha \tag{3}$$

Since, $U < 1$ (which can be derived from Equation (1)), for large j, $\frac{1}{j-U}$ is approximately equal to $\frac{1}{j}$, and we know that $\sum_{j=1}^{N} 1/j$ is approximately $log\ N$. Thus:

$$U < \alpha/logN \tag{4}$$

Remember that, by definition, $U = \sum_i C_i/D_i$ over all in-transit packets through a node. Since multiplying the packet transmission time, C_i, by the channel transmission speed, W_n, yields packet size, multiplying both sides of the above equation by W_n establishes the average number of bytes that can be transmitted by an average node for each unit of time of the relative deadline. Summing that quantity over the whole network is what defines its real-time capacity (in byte-hops per second). Thus:

$$C_{RT} = W_n \sum_j Uj \tag{5}$$

Since the aggregate synthetic utilization over all nodes distance j from the destination is upper bounded by that at the destination (as explained above) we can sum up the total network capacity by cutting the network into N concentric circles, where N is of the order of the average path length. The traffic through each circle is not less than that at the destination. The total real-time capacity is therefore bounded by $W_n UN$. Observe that network diameter is generally proportional to network area. Hence, assuming a uniform density network of n nodes, if N is a constant fraction of the diameter, then N is $O(\sqrt{n})$. Assuming that the MAC layer uses an actual transmission rate of W, and that node density is such that on average h nodes typically lie within the range of any one receiver, we can approximately state that $W_n = W/h$. Hence:

$$C_{RT} = \frac{\alpha NW}{hlogN} \tag{6}$$

or:

$$C_{RT} = O(\frac{\sqrt{n}}{log\sqrt{n}})\frac{W}{h} \tag{7}$$

The above expression can be used to set the sampling rate R of periodic sensors. If the size of a single sample is s bytes, its latency constraint is D seconds, and the average sensor distance from the sink is N hops, the real-time capacity requirement of the sensor is sRN/D. Summing up the requirements of all sensors, one must satisfy that the total is less than C_{RT} (given by Equation 6) for the end-to-end time constraints to be satisfied. This imposes a constraint on the maximum sampling rate, R.

Having chosen a sampling rate for periodic sensors, we proceed to the next step of the design problem. In this step, we focus on sensors that are triggered aperiodically by events in the environment. The traffic from such sensors is added to that of the periodic sensors, which may cause system overload when the environment becomes highly dynamic. Lossy aggregation must therefore be performed to maintain timeliness while maximizing information throughput.

4. Control of Lossy Aggregation

When the amount of data generated by the combination of periodic and aperiodic sensors exceeds system capacity, the lossy aggregation feedback loop is activated. The controller of this loop attempts to balance timeliness and information delivered. Its set point can be tuned for better timeliness at the expense of increased aggregation (i.e., more information loss) or lower information loss at the expense of looser timing performance. In particular, the system designer specifies the maximum data path length N for which no deadline misses may occur. The feedback loop must control the degree of aggregation such that information throughput is maximized subject to the above requirement. In the following we derive the local conditions that lead to maximization of global information throughput. We then describe how these conditions are used to design the lossy aggregation feedback loop and compute its per-node set points.

When lossy aggregation is used, the sum of synthetic utilizations of all data sources may exceed that of the sink, since more raw data may be generated than is delivered to the sink. It is desired to devise an aggregation scheme that maximizes total real-time capacity, which is proportional to $W_n \sum_j U_j$ across all nodes in the system, as stated in Equation (5). From the symmetry of the aforementioned summation, as well as the symmetry of the schedulability condition given by Equation (1), the solution that maximizes capacity must be symmetric with respect to synthetic utilization. In other words, U_j must be equal at all nodes. This is called a *load-balanced* network. Since we require that time constraints be met only for paths of length N or less, it is enough to focus on that path length. In a load-balanced network, from Equation (1), the synthetic utilization U of each single node on a communication path of length N must satisfy:

$$\frac{U(1-U/2)}{1-U} < \alpha/N \qquad (8)$$

Solving for U, we get:

$$U < 1 + \frac{\alpha}{N} - \sqrt{1+(\frac{\alpha}{N})^2} \qquad (9)$$

From Equation (5), the capacity of the network is nUW_n byte-hops per unit of relative deadline. Hence, in the optimal case of a load-balanced network, the real-time capacity of the sensor network, denoted C_{RT}, is bounded by:

$$C_{RT} < n(1 + \frac{\alpha}{N} - \sqrt{1+(\frac{\alpha}{N})^2}\,)W_n \qquad (10)$$

Some interesting observations are apparent. First, rewriting $1 + \alpha/N$ as $\sqrt{1+2\alpha/N+(\alpha/N)^2}$, observe that when N is large, the term $(\alpha/N)^2$ can be neglected leading to:

$$C_{RT} < n(\sqrt{1+\frac{2\alpha}{N}} - 1)W_n \qquad (11)$$

We know from series expansion that for a small x, the term $\sqrt{1+x}$ is approximately equal to $1 + x/2$. Hence, substituting for the square root in Equation (11) when N is large, and recalling that $W_n = W/h$, we get:

$$C_{RT} < \frac{n\alpha}{Nh}W \qquad (12)$$

If path length N is of the order of the square root of the area of the network, which in turn is of the order of the number of nodes, then:

$$C_{RT} = O(\sqrt{n})\frac{W}{h} \qquad (13)$$

To maximize real-time information throughput such that the above capacity bound is approached, the local controller at each node attempts to keep its synthetic utilization at the value indicated in the right hand side of inequality 9. Hence, the controller set point, $U_{desired}$, is:

$$U_{desired} = 1 + \frac{\alpha}{N} - \sqrt{1+(\frac{\alpha}{N})^2} \qquad (14)$$

Choosing a larger N will reduce the utilization, thereby increasing the amount of lossy aggregation. A smaller N will reduce information loss, but increase deadline misses along longer paths. The instantaneous synthetic utilization of a node is $U_{inst} = \sum_i C_i/D_i$, carried out over all outgoing packets. As explained in [1], this value is increased by C_i/D_i when a new packet, T_i, arrives. It is decreased by C_i/D_i only at the delivery deadline of the packet (and is set to zero when the link is idle). Each node maintains an exponential moving average $U_{avg}(k)$ of instantaneous synthetic utilization. This moving average is updated periodically at the controller's sampling interval. The control error $e(k)$ in the k^{th} sampling interval is defined as $e(k) = U_{desired} - U_{avg}$. This error drives an integral regulator of gain K_I whose output m determines the degree of lossy aggregation required, where:

$$\delta m = K_I(U_{desired} - U_{avg}) \qquad (15)$$

More specifically, m specifies the ratio of the number of packets after and before aggregation. For example $m = 0.66$ indicates that each 3 incoming packets must be aggregated into 2 (by averaging a pair), where $2/3 = 0.66$. A field in each packet's header keeps track of the number of original raw data items the packet's aggregated value reflects. This allows correct weights to be used when averaging the content of two packets.

Since aggregation can only *reduce* the number of packets, the maximum value of controller output m is 1, indicating that no aggregation is needed. Observe that when the system is underloaded ($U_{avg} < U_{desired}$), the controller eventually saturates at $m = 1$. Anti-windup is then invoked, thus opening the lossy aggregation control loop. Hence, only lossless aggregation is performed in an underloaded system.

Note that the instantaneous synthetic utilization of the system is proportional to m. Hence, the controlled process has a constant gain. If all data deadlines are the same, that gain is unity. The only dynamics in the loop are those that arise from the low-pass filter (i.e., exponential moving averaging) and the controller. The filter is essential to smooth bursts.

5. Control of Lossless Aggregation

When the system operates in the non-overloaded regime, only lossless aggregation is performed to optimize energy consumption and reduce delay. An architecture for application-independent data aggregation is described in [3]. In that regime, a feedback loop measures the average delay incurred to transmit a packet (which includes the contention delay on the wireless medium). This measurement is then used to adapt the degree of lossless data aggregation, called N_{aggr}. When a particular degree of aggregation is indicated, packets are not forwarded to the network device until the corresponding number of them (i.e., at least N_{aggr}) are present in the queue.

The default degree of aggregation N_{aggr} is 1, which occurs at low load. In this case, packets are delivered to the network device for transmission as soon as the device is ready. Note that if more than one packet have accumulated in the queue while the network device was busy, they will be aggregated and sent together. As network traffic builds up and contention delays increase, the feedback loop adjusts the aggregation level, N_{aggr}, to allow a greater minimum degree of aggregation. When the network device is free, packets are sent only as long as at least N_{aggr} of them are present.

Next, we derive a model for data aggregation that will be used to tune our feedback loop. The control loop operates periodically at some appropriately chosen interval, T, measuring the current MAC-layer delay $D(k)$ and adjusting the degree of aggregation, $N_{aggr}(k)$, accordingly. Let the k^{th} sampling interval of the control loop be $[(k-1)T, kT)$. The delay sensor produces its reading, $D(k)$, at the end of each interval. This reading represents the average MAC-layer delay of all packets transmitted in that last sampling interval. The average delay a packet experiences before its transmission is complete is:

$$D(k) = D_{min} + D_{collide} \tag{16}$$

where D_{min} is the minimum delay experienced when no collisions occur (which is primarily the packet tranmission delay plus some system overhead), and $D_{collide}$ is the average additional delay incurred due to collisions.

Assume that a total of $M(k)$ packets were present in interval k in the combined queues of all nodes sharing the same neighborhood, where only one node can transmit at a time. Given a degree of aggregation, $N_{aggr}(k-1)$, set at the beginning of that interval, at most $M(k)/N_{aggr}(k-1)$ data units will be transmitted on the medium. This is only an approximation, because different nodes may have different $N_{aggr}(k-1)$ values. However, since those nodes share the same medium with the same level of congestion, it is likely that their $N_{aggr}(k-1)$ will be close. Since the probability of collisions grows linearly with the number of data units available for transmission, the expected number of collisions grows with $M(k)/N_{aggr}(k-1)$. Furthermore, since most contention-based MAC-layers exhibit exponential back-off upon a collision, the average contention delay, $D_{collide}$, grows exponentially with the number of collisions. Hence:

$$D(k) = D_{min} + Ae^{bM(k)/N_{aggr}(k-1)} \tag{17}$$

where A and b are constants. This is clearly a non-linear system. We linearize the system by computing its derivative with respect to the manipulated variable (in this case, $N_{aggr}(k-1)$), which yields the small deviation model:

$$\frac{dD(k)}{dN_{aggr}(k-1)} = -A\frac{M(k)}{N_{aggr}(k-1)^2}e^{bM(k)/N_{aggr}(k-1)} \tag{18}$$

Hence, if the degree of aggregation is changed by $\delta N_{aggr}(k) = N_{aggr}(k) - N_{aggr}(k-1)$, and assuming a constant workload $M(k+1) = M(k) = M)$, it is predicted that:

$$D(k+1) = D(k) - A\frac{M}{N_{aggr}(k-1)^2}e^{bM/N_{aggr}(k-1)}\delta N_{aggr}(k) \tag{19}$$

The system model contains a nonlinear integral term. A proportional controller can therefore be used to stabilize the system and eliminate steady state error. The gain of the proportional controller can be made dynamic to compensate for part of the system nonlinearity. The controller we use is thus given by:

$$\delta N_{aggr}(k) = PN_{aggr}(k-1)^2 e(k) \tag{20}$$

where $e(k) = D(k) - D_{desired}$, and P is controller gain.

6. Experimental Evaluation

We simulate our architecture in GloMoSim [5], a scalable discrete-event simulator developed at UCLA. This software provides a high fidelity simulation for wireless communication with detailed propagation, radio, MAC, and network layer components. In our experiments, the communication parameters are chosen in accordance with Berkeley Telos mote specifications, the latest hardware platform on which sensor network research systems are currently deployed for testing.

We evaluate two types of data aggregation techniques discussed in previous sections, namely lossless and lossy aggregation, and compare them with a non-aggregation scheme. During the simulation, we adopt a typical many-to-one traffic pattern, where 10 source nodes send out CBR (Constant Bit Rate) flows to a single sink with average hop length 4 - 6 hops. The end-to-end deadline used in the experiment is 200 ms. To investigate the effectiveness of data aggregation in the presence of congestion, we incrementally increase the sending rate of 10 flows from 1.5 to 3.7 packets/second per flow. Experiments are repeated 30 runs with different seeds such that the 95% confidence intervals are within 2 - 5% of the mean.

Figure 2 demonstrates that both lossless and lossy aggregation can dramatically reduce average packet end-to-end delay in comparison with the non-aggregation scheme when the traffic becomes heavy, thanks to the fact that aggregation techniques can control the amount of information delivered in response to the timeliness requirements.

When the amount of information generated exceeds the real-time capacity, the lossy aggregation demonstrates its excellent capability of achieving low end-to-end deadline miss ratio by aggregating a small percentage of packets together. As shown in Figure 4, miss ratios for the lossy aggregation scheme under different traffic loads are always below 10%, while the lossless aggregation scheme, which doesn't take real-time capacity into account, suffers a high miss ratio penalty when traffic exceeds real-time capacity of the network.

We note that the lossy aggregation does not achieve this excellent performance for free. As shown in Figure 5, it has a non-zero

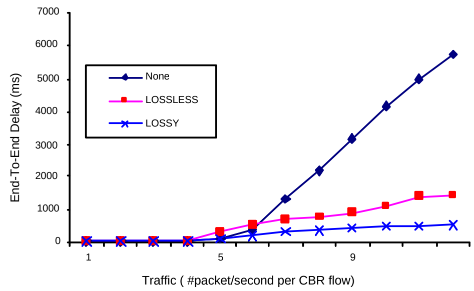

Figure 2. End-to-End Delay Vs. Traffic Load

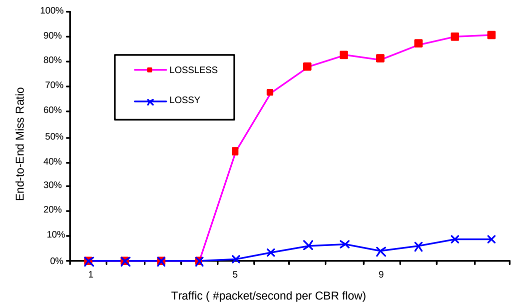

Figure 4. End2End Miss Ratio Vs. Traffic Load

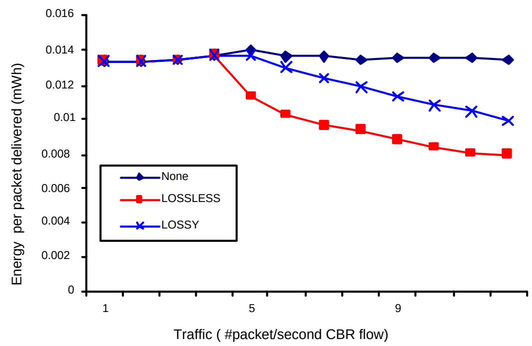

Figure 3. Energy Vs. Traffic Load

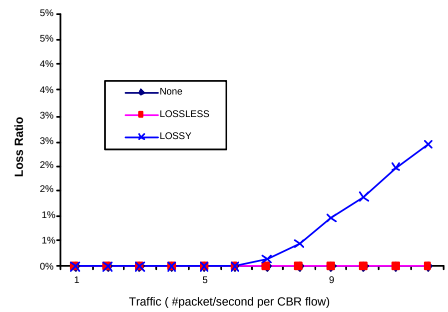

Figure 5. Loss Ratio Vs. Traffic Load

lossy ratio[2] in heavy traffic in exchange for excellent timeliness shown in Figure 4.

In addition, as shown Figure 3, both lossless and lossy aggregations can achieve energy conservation by reducing the number of control messages and the number of retransmissions in the presence of congestion.

7. Conclusions and Future Work

In this paper, we demonstrated the application of control theory to resolve fundamental performance trade-offs in sensor networks. Fundamental limits were presented on real-time network capacity. These limits where then used to derive sensor sampling rates and set points of control loops. Two different mechanisms for data aggregation were presented whose combined effect is to maximize information throughput while maintaining timing constraints and reducing protocol overhead. There are several outstanding issues that the authors hope to address in future interdisciplinary collaborations. For example, how to model non-linearities peculiar to sensor networks? How can these nonlinearities be accounted for in control? How efficient are adaptive control and robust control techniques in dealing with parameter variation and load uncertainty? What other actuators can be applied in addition to aggregation? What is the effect of routing policies? Examples, theoretical foundations, experimental evidence, and practical experience are needed in applying feedback performance control to sensor networks. This is an important focus of our research group at the present time.

[2] Lossy ratio is the percentage of packets that are aggregated with information loss

8. REFERENCES

[1] T. Abdelzaher, G. Thaker, and P. Lardieri. A feasible region for meeting aperiodic end-to-end deadlines in resource pipelines. In *ICDCS*, Tokyo, Japan, March 2004.

[2] P. Gupta and P. R. Kumar. The capacity of wireless networks. *IEEE Transactions on Information Theory*, 46(2):388–404, March 2000.

[3] T. He, B. M. Blum, J. A. Stankovic, and T. F. Abdelzaher. Aida: Adaptive application independent data aggregation in wireless sensor networks. *ACM Transactions on Embedded Computing Systems*, (to appear) 2004.

[4] J. A. Stankovic, T. Abdelzaher, C. Lu, L. Sha, and J. Hou. Real-time coomunication and coordination in embedded sensor networks. *Proceedings of the IEEE*, 91(7), July 2003.

[5] X. Zeng, R. Bagrodia, and M. Gerla. GloMoSim: A library for parallel simulation of large-scale wireless networks. In *Workshop on Parallel and Distributed Simulation*, pages 154–161, 1998.