Sensor Data Collection with Expected Reliability Guarantees

Qi Han, Iosif Lazaridis, Sharad Mehrotra, Nalini Venkatasubramanian
Department of Computer Science, University of California, Irvine, CA 92697
{qhan,iosif,sharad,nalini}@ics.uci.edu

Abstract

Due to the fragility of small sensors, their finite energy supply and the loss of packets in the wireless channel, reports from sensors may not reach the sink node. In this paper we consider the problem of sensor data collection in the presence of faults in sensor networks. We develop a data collection protocol, which provides expected reliability guarantees while minimizing resource consumption by adaptively adjusting the number of retransmissions based on current network fault conditions.

1. Introduction

With the advances in computational, communication, and sensing capabilities, large scale sensor-based distributed environments are becoming a reality. Such distributed sensor environments allow us to continuously monitor and record the state of the physical world, which can be used for a variety of purposes. Applications who are interested in sensor readings reside at powerful servers/sinks outside of the sensor network. Sensors are expected to be inexpensive and can be deployed in a large number to inhospitable environments, which implies that sensors are typically operating unattended. Therefore, sensor networks are subject to high fault rate: connectivity between nodes can be lost due to environmental noise and obstacles; nodes may die due to power depletion, environmental changes or malicious destruction. As mentioned in [8], fault tolerance is one of the metrics used to evaluate sensor applications in addition to energy efficiency/system lifetime, latency, accuracy and scalability. As a motivating application, we consider event detection such as chemical leakage diagnosis. If several (say 5) possible spots can cause leaking, we need to identify the exact spot where the toxic gas originates from, so a certain percentage (say 90%) of the reports from each of the 5 sensors are needed; after the leaking spot is identified, only 90% is needed from that particular sensor, to make sure that the leaking is effectively controlled.

Increasingly, researchers are realizing that sensors are more than passive beacons, but can perform useful work, both to conserve their own resources and to meet application goals. The failure prone nature of sensor networks implies that in order to ensure reliability requirements from certain applications, both the sink (where applications are injected into the sensor network) and sensors need to put more effort. This paper deals with how the sink and the sensor collaborate to provide answers with additional *reliability guarantees*. The application must now specify its *reliability requirements*, expressing the minimum tolerable reliability degree. The system will then work towards producing such a guarantee: sometimes due to the occurrence of many faults, it might be impossible to achieve this goal; then the "best possible" answer will be given.

Problem Definition: We consider typical sensor applications involving the reliable detection and/or estimation of event features based on reports from the sensor node observing the event. For example, in order to detect and fix toxic chemical leaking, the sink must decide on the chemical density every τ time units. Here, τ represents the duration of a decision interval and is fixed by the application. At the end of the decision interval, the sink makes an informed decision based on reports received from the sensor node during that interval. Typically, in order to make a fair decision, the sink needs to receive at least R_q (*desired reliability*) of the data sent by the sensor, the ratio of the required number of received values to the number of transmitted values, required for reliable collection. We define the *observed reliability* r as the actual reliability measured in τ time units, which measures the ratio of number of data items the sink actually receives, to the number of data items a sensor injects into the network. Our objective is to minimize the communication overhead involved in maintaining that $r \geq R_q$.

Equivalently, let N be the number of items to be transmitted by the sensor, N_q be the number of items expected at the sink. Let $\mathcal{P}$ be a plan of transmission, defined as $\mathcal{P} = (n_1^{\mathcal{P}}, n_2^{\mathcal{P}}, \ldots, n_N^{\mathcal{P}})$ where $n_i^{\mathcal{P}}$ is the number of times that item i is transmitted. Let the expected number of items $N_r^{\mathcal{P}}$ reaching the sink at least once for plan $\mathcal{P}$ is $E(N_r^{\mathcal{P}})$, The cost of plan $\mathcal{P}$, assuming that each message has a uniform cost, is:

Proceedings of the 3rd Int'l Conf. on Pervasive Computing and Communications Workshops (PerCom 2005 Workshops)
0-7695-2300-5/05 $20.00 © 2005 IEEE

$$C(\mathcal{P}) = \sum_{i=1}^{N} n_i^{\mathcal{P}}.$$

Our objective is to minimize $C(\mathcal{P})$ while maintaining that $E(N_r^{\mathcal{P}}) \geq N_q$.

2. Data Collection Protocol with Expected Reliability Guarantees (PERG)

In this section, we present a Protocol with Expected Reliability Guarantees (PERG). In order to ensure the reliability requirement at the end of τ time units, we divide the whole time period into rounds, where each round contains the same number of data items sent by the sensor. Note that we assume sensors might not report their values periodically, but triggered by events. This implies that the duration of each round may be different. This round-based protocol facilitates improving observed reliability from round to round. Assuming that the observed reliability in round i is r_i, if fault rate is so high that R_q is not achieved ($r_i < R_q$), we can raise reliability requirement for round $i+1$ - $R_q(i+1)$, aiming to compensate the loss in the previous round; On the other hand, if $r_i > R_q$, we can lower $R_q(i+1)$ so that communication overhead is saved. Let N_i be the number of items generated in round i, $R_q(i)$ is the desired reliability of round i, r_i is the observed reliability of round i, then

$$N_i \cdot r_i + N_{i+1} \cdot R_q(i+1) \geq (N_i + N_{i+1}) \cdot R_q.$$

We get $R_q(i+1) \geq \frac{(N_i+N_{i+1})\cdot R_q - N_i \cdot r_i}{N_{i+1}}$. If $N_i = N_{i+1}$, then $2R_q - r_i \leq R_q(i+1) \leq 1$. For example, let us assume that the desired reliability of an application is 0.8, the sensor generates 100 data items in each round. If the actual reliability of round i is 0.7, then the desired reliability for next round is 0.9.

The basic idea of PERG is to use re-transmission to achieve user required reliability. At the end of each round, the sensor sends the server the information about the number of data items sent in this round, and the number of messages sent in this round. Based on these information, the server estimates current network situation, as well as the actual reliability achieved in this round. The sensor derives re-transmission times for data items generated in the next round based on the feedback from the server. We now proceeds to discuss the details of the protocol.

2.1. Inferring Current Fault Severity

Since the sensor (not the sink) can keep track of the number of data items or messages sent, and only the sink (not the sensor) is aware of the number of data items or messages received, it is necessary for the sink and the sensor to exchange these information in order to infer current fault severity. More specifically, We are interested in message delivery rate P_r and observed reliability r. The information is exchanged as follows.

•The information exchange is started by a FEEDBACK-REQUEST-MSG sent from the sensor to the sink at the end of a round. The sensor repeat sending this message until a FEEDBACK-RESPONSE-MSG is received from the sink or until this message is repeated N_{fr} times. The FEEDBACK-REQUEST-MSG includes the number of items sent from the sensor N_{ds}, the number of messages sent from the sensor N_{ms}. Note that N_{ms} can be different from N_{ds} since one item may be sent multiple times.

•Upon hearing FEEDBACK-REQUEST-MSG, the sink calculates P_r and r, using its knowledge of the number of items received N_{dr} and the number of messages received N_{mr}: $P_r = \frac{N_{mr}}{N_{ms}}$, $r_i = \frac{N_{dr}}{N_{ds}}$, where $N_{ds} = N$. The sink then sends out FEEDBACK-RESPONSE-MSG, which contains P_r and r, either N_{fb} times or until FEEDBACK-ACK-MSG is received.

• Upon receiving FEEDBACK-RESPONSE-MSG, the sensor responds with a FEEDBACK-ACK-MSG.

Figure 1 describes the communication between the server and the sensor. For example, if the sensor sent out 100 data items using 150 messages, and the server received 100 messages which contain 80 distinguished data items, then the message delivery rate is $\frac{100}{150} = 67\%$ and the reliability achieved is 80%.

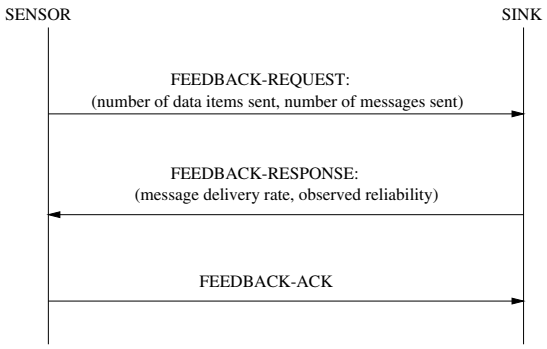

Figure 1. Feedback Protocol

The number of re-transmissions for feedback acknowledgment N_{fb} is adjusted so that the probability of the sensor receiving the feedback is above a certain threshold. Let P_{fb} be the probability of sensor receiving the feedback at least once after it is transmitted N_{fb} times, then $P_{fb} = 1 - (1 - P_r)^{N_{fb}}$. If we would like to have $P_{fb} > \alpha$, then $N_{fb} \geq \frac{\log(1-\alpha)}{\log(1-P_r)}$. We can compute the number of re-transmissions for feedback request N_{fr} in a similar way. The newly computed/received P_r and r serves as an indicator of network fault severity for the next round, where parameters such as N_{fb}, N_{fr} are computed. However, it

is still possible for those feedback messages to get lost, we therefore enforce a timeout mechanism. In other words, if no feedback message is received within certain time duration, we use the parameters of the previous round.

2.2. Re-Transmission Based Plan

The feedback message described above provides an indicator of current network fault rate. Let P_f be the probability that a transmitted item will not reach the sink if it is sent once, then $P_f = 1 - P_r$. In the presence of failure in the sensor network, data needs to be re-transmitted in order to guarantee the expected reliability. Assuming that P_f is approximately constant within a round, the expected number of items $N_r^{\mathcal{P}}$ reaching the sink at least once for plan $\mathcal{P}$ is:

$$E(N_r^{\mathcal{P}}) = \sum_{i=1}^{N}(1 - P_f^{n_i^{\mathcal{P}}}).$$

We aim to design an optimal transmission plan $\mathcal{P}$ which minimizes $C(\mathcal{P})$ while maintaining that $E(N_r^{\mathcal{P}}) \geq N_q$.

Assuming that we have a transmission plan, in which an item is sent either k times, or $k+1$ times. Let N_k be the number of items that are sent out k times, and N_{k+1} be the number of items that are sent out $k+1$ times, then

$$E(N_r^{\mathcal{P}}) = N_k \cdot (1 - P_f^k) + (N - N_k) \cdot (1 - P_f^{k+1}) \quad (1)$$

We now need to determine the **optimal** k, which should meet the following statement: the expected number of items received should be less than N_q can be received if all N items are transmitted k times, and it should be at least N_q if all N items are transmitted $k+1$ times. In other words, $N \cdot (1 - {P_f}^k) < N_q \leq N \cdot (1 - {P_f}^{k+1})$. From this, we get

$$\frac{\log(1 - \frac{N_q}{N})}{\log P_f} - 1 \leq k < \frac{\log(1 - \frac{N_q}{N})}{\log P_f},$$

$$\text{i.e.,} k = \lceil \frac{\log(1 - \frac{N_q}{N})}{\log P_f} - 1 \rceil.$$

We also need to derive the N_k. From Equation(1) and $E(N_r^{\mathcal{P}}) \geq N_q$, we get

$$N_k \leq \frac{N_q - N \cdot (1 - P_f^{k+1})}{P_f^{k+1} - P_f^k},$$

$$\text{i.e.,} N_k = \lfloor \frac{N_q - N \cdot (1 - P_f^{k+1})}{P_f^{k+1} - P_f^k} \rfloor \quad (2)$$

We have proved, via the following lemma, that this transmission plan is the optimal one.

Lemma: For a plan $\mathcal{P}$ with $E(N_r^{\mathcal{P}}) \geq N_q$, the cost $C(\mathcal{P})$ is minimized only if

$$\forall i, j \in \{1, 2, \ldots, N\} : |n_i^{\mathcal{P}} - n_j^{\mathcal{P}}| \leq 1 \quad (3)$$

Proof: Interested readers may refer to [3].

Given that we have decided that for all the N items in one round, we transmit part of them k times and the remaining part $k+1$ times, we now need to decide how to send out these messages. For example, the sensor generates data item A, B, C and D. One option is to send them interspersedly, that is ABCDABCD.... However, this would require that the sensor store all of generated items until the transmission time; in addition, the application needs to wait until at least one round period in order to receive the first item generated at the beginning of the round. Due to the memory limitations of sensor nodes and the possible timeliness requirements of the applications, we choose to send the items consecutively immediately after it is generated. This strategy brings another benefits that the sensor can switch off its radio after re-transmitting as many times as computed.

3. Performance Evaluation

The objective of the performance study is to validate our proposed protocol PERG, and evaluate its performance under different system conditions. We compare PERG against a simplistic policy LAZY, which ignores failures and the sensor sends out each of its values only once. The performance of LAZY is purely dependent on the status of the sensor network, and provides a baseline of our performance comparison. Two performance metrics are used to compare policies: (i) observed reliability and (ii) the normalized cost, defined as the total number of messages exchanged divided by the total number of data items the sensor generates.

We simulated the protocols LAZY and PERG on GlomoSim [12], a scalable discrete-event simulator for wireless networks which provides detailed radio and MAC layers. We chose communication parameters using the Berkeley mote specification [6], where CSMA is used as MAC layer protocol.

Experimental Results: We evaluate the system's behavior by varying application desired reliability and sensor network fault rate. All the results are averaged over multiple(10) runs.

Figure 2 demonstrates the impact of reliability adjustment in PERG for the first 10 rounds with application desired reliability(R_q) 0.9 and network fault rate 20%. Since protocol LAZY is not adaptive to either network conditions or application requirement, both the observed reliability and the cost remains the same in each round. However, in PERG, the desired reliability varies over the round with the ultimate goal of achieving the application desired reliability. For example, in the first round, without the knowledge of the network conditions, PERG exhibits the same performance as LAZY; however, in the second round, PERG raises its desired reliability to 0.99 in order to compensate the loss in the first round. Subsequently, PERG lowers its

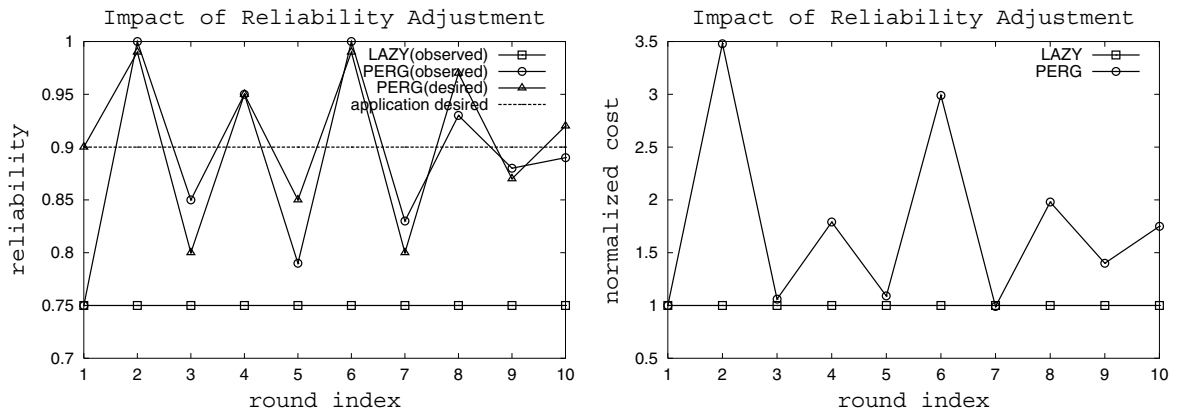

Figure 2. Impact of Reliability adjustment

desired reliability if more data items are received during previous rounds.

Figure 3 depicts the protocol performance as network fault rate varies. We observe that with the increase of fault rate, the observed reliability provided by LAZY decreases dramatically, while the observed reliability provided by PERG drops only slightly. Without surprise, this performance comes at the price of sending more messages, i.e., re-transmitting part of the data items. In order to achieve the desired reliability, the cost increases as the network fault gets more severe.

Figure 4 shows the impact of application requirements. The observed reliability of PERG is very close to the desired reliability of applications; while LAZY only provides the same reliability regardless of application requirements.

In summary, PERG provides the expected reliability guarantees via re-transmitting sensor data, which involves higher communication overhead. In contrast, even though LAZY incurs minimal overhead, it is unable to react to different reliability requirements from applications or sensor network fault conditions.

4. Related Work and Concluding Remarks

Despite a considerable amount of research on sensor networking and databases, the problems of reliable delivery of sensor data in the presence of faults and losses have not been studied until recently. The work most relevant to ours is TAG [4] and SKETCH [1], which attempt to provide robust aggregation of data in sensor networks. In TAG, a routing tree is formed during query dissemination phase. Later, a sensor node selects a new parent if (1) the quality of the link with his parent is significantly worse than that of another potential parent, or (2) it has not heard from its parent for some fixed period of time. SKETCH uses a DAG instead of a tree for data delivery. Given that most nodes have multiple parents in a DAG, an individual link or node failure has limited effects. A robust technique for computing duplicate sensitive aggregates was proposed by combining multi-path routing and duplicate insensitive sketches. In contrast, our work in this paper deals with event detection applications where reliability is imposed on a series of sensor data generated over time, and we also provide quality guarantees taking into consideration the lossy/faulty nature of the sensor network.

Congestion is one of factors causing faults in sensor networks. ESRT (Event-to-Sink Reliable Transport) [7] aims to provide congestion control in sensor networks by adjusting sensor reporting frequency based on current network congestion and application specific reliability requirements. Orthogonal to this upstream (data delivery from sensors to server) reliability, there is also research ensuring downstream reliability. PSFQ (Pump Slowly, Fetch Quickly) [9] is a hop-by-hop error recovery scheme. Driven by the purpose of controlling, managing or re-tasking sensors, PSFQ aims to provide in-sequence data delivery from the sink to the sensors. Along the same line, GARUDA [5] also provides sink-to-sensors reliability.

Providing reliable data delivery has also been addressed by routing protocols. Braided Diffusion [2] maintains multiple "braided" paths as backup. When a node on the primary path fails, data can go on an alternate path. GRAB (Gradient Broadcast) [11] ensures a robust data delivery through controlled mesh forwarding. It controls the "width" of the forwarding mesh, thus the degree of redundancy in forwarding data. We argue that reliable routing does not differentiate data and enforce reliable delivery of each piece of data, which is neither efficient nor necessary. In addition, interesting studies have been conducted in terms of the impact of link quality estimation and neighborhood table management on reliable routing in sensor networks [10].

Concluding Remarks: In this paper we considered the problem of supporting event detection applications in sen-

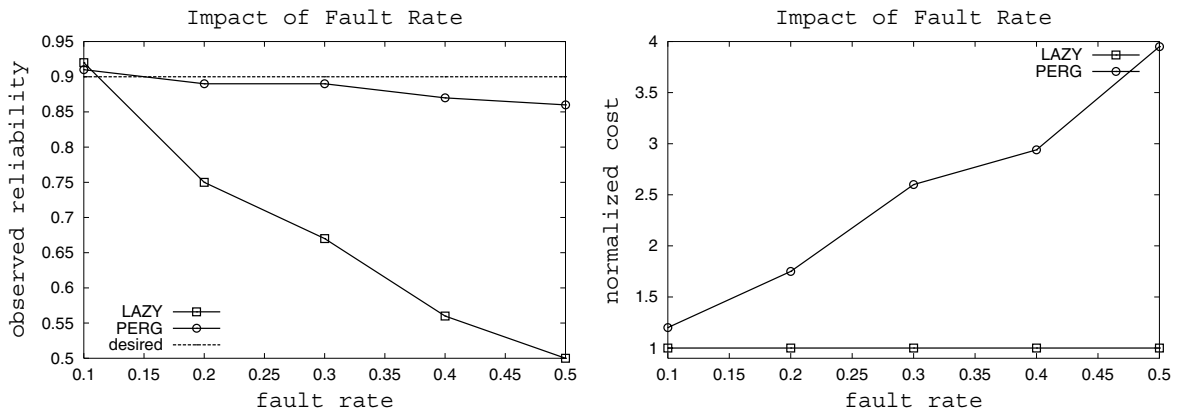

Figure 3. Impact of Fault Rate

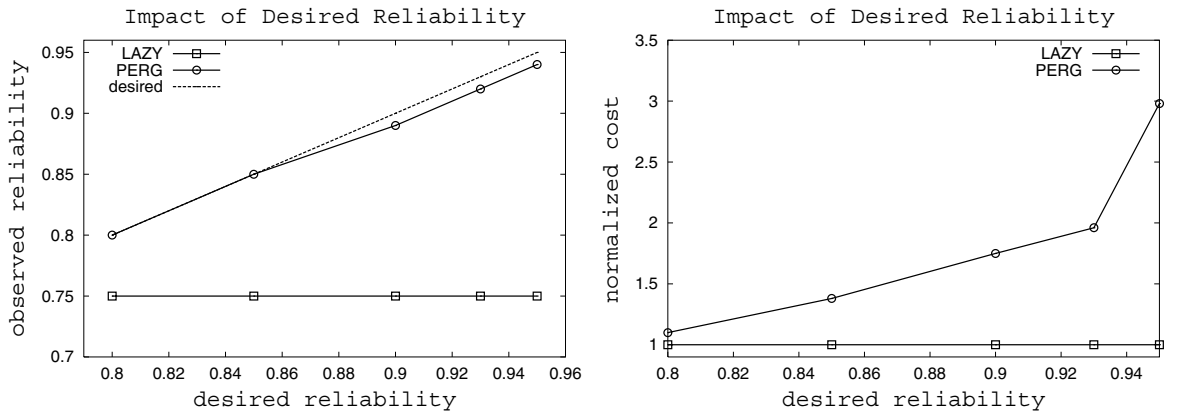

Figure 4. Impact of Desired Reliability

sor networks when faults are likely to occur. We developed a protocol for trying to achieve expected reliability guarantees efficiently, evaluated the proposed protocol PERG in the presence of varying fault rates and under different application reliability requirements.

References

[1] J. Considine, F. Li, G. Kollios, and J. Brers. Approximate aggregation techniques for sensor databases. In *IEEE ICDE*, 2004.

[2] D. Ganesan, R. Govindan, S. Shenker, and D. Estrin. Highly-resilient, energy-efficient multipath routing in wireless sensor networks. *ACM Mobile Computing and Communications Review*, 1(2), October 2002.

[3] Q. Han, I. Lazaridis, S. Mehrotra, and N. Venkatasubramanian. Sensor data collection with expected reliability guarantees. Technical report, UCI, 2004. http://www.ics.uci.edu/ qhan/techrep/perg-techrep.pdf.

[4] S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong. Tag: a tiny aggregation service for ad-hoc sensor networks. In *USENIX OSDI*, 2002.

[5] S. J. Park, R. Vedantham, R. Sivakumar, and I. F. Akyildiz. A scalable approach for reliable downstream data delivery in wireless sensor networks. In *ACM MobiHoc*, 2004.

[6] RT Monolithics Inc., http://www.rfm.com/. *ASH Transceiver TR1000 Data Sheet*.

[7] Y. Sankarasubramaniam, O. B. Akan, and I. F. Akyildiz. Esrt: Event-to-sink reliable transport in wireless sensor networks. In *ACM MobiHoc*, 2003.

[8] S. Tilak, N. B. Abu-Ghazaleh, and W. Heinzelman. A taxonomy of wireless micro-sensor network models. *Mobile Computing and Communications Review*, 6(2), 2002.

[9] C. Y. Wan, A. T. Campbell, and L. Krishnamurthy. Psfq: A reliable transport protocol for wireless sensor networks. In *ACM WSNA*, 2002.

[10] A. Woo, T. Tong, and D. Culler. Taming the underlying challenges of reliable multihop routing in sensor networks. In *ACM SenSys*, 2003.

[11] F. Ye, G. Zhong, S. Lu, and L. Zhang. Gradient broadcast: A robust data delivery protocol for large scale sensor networks. *ACM WINET*, 11, 2003.

[12] X. Zeng, R. Bagrodia, and M. Gerla. Glomosim: a library for parallel simulation of large-scale wireless networks. In *PADS*, 1998.