Data Quality Driven Sensor Reporting *

Doug Hakkarinen, Qi Han
Department of Mathematical and Computer Sciences
Colorado School of Mines, Golden, CO 80401
{dhakkari,qhan}@mines.edu

Abstract

Within the field of event driven data reporting from wireless sensor networks, reducing energy consumption is an ongoing problem. Using an application's tolerance toward data imprecision (or, data quality) allows energy savings, via fewer messages sent, through the selection of how and when to send sensor readings. This paper is an early work that examines pushing or pulling data depending on which approach is expected to send fewer messages, based upon the recent history of application requests for a sensor and the changes of the sensor values predicted by application specific models. The simulation results indicate that our method is more efficient relative to the push only or pull only methods for situations where application request frequency or data change rate is variable or unknown.

1. Introduction

Wireless sensor networks have been used to support a variety of applications including those that can use approximate rather than exact sensor values, as long as the values are within a tolerable range of the true values. Due to the limitation of sensing technologies or the possibility of miscalibrated sensors, it is justified to use approximate values. Furthermore, as sensors do not query every point in the environment, their samples are already at some level of approximation[3]. The use of approximate data allows flexibility on when to send data, creating an opportunity to decrease the amount of energy used by motes.

Another need in wireless sensor networks is to be able to validate data models. A framework that enables checking whether a model is accurate would greatly help in selecting and calibrating a model. One application that has these needs is the tracking of contaminant plumes in underground water flows. Researchers in the field have noted that precise value readings are not specifically required and that data within a bounded error percentage is useful for modeling the contaminant flows. For example, a 3% error percentage would mean that if a specific reading were at 100, any value between 97 and 103 would be accurate enough. Wireless sensor networks can potentially aid the development of contaminant flow models by verifying their performance in predicting the contaminant flow.

In this paper, the above issues will be addressed through a system that transmits data in both a push and pull fashion. The objective is to minimize the amount of energy needed by pushing data from the mote to the base station (i.e., a sensor driven update) when more efficient, or to pull data from the mote by query otherwise (i.e., a query driven update). Furthermore, our approach will significantly help application domain experts; the effectiveness of the data model can now be evaluated in an online fashion, since a poor model would often mispredict and result in frequent sensor driven updates.

Most of the existing work uses either a push or pull method exclusively for data gathering. In the event-driven techniques, determining the frequency of updates based upon the current value and known value at the base station has been used [4]. In the query driven techniques, reductions in energy can be achieved by aggregation of data in queries to reduce data sent back [2] or statistically modeling the data at the base station to reduce queries [3]. In contrast, our work dynamically switches between push and pull techniques based on system conditions. This paper is an early endeavor in this area. We present our preliminary results that provide the foundations for our research pursuits. We believe that the proposed approach, suitably modified or enhanced, will provide a very efficient way to gather sensor reports for a variety of applications with different quality of data needs.

*This project is supported in part by NSF CSR grant CNS-0720875.

978-1-4244-2575-4/08/$20.00 ©2008 IEEE

2. Problem Formulation

Multiple sensors of various types may be mounted on a mote. As a result, an application's data acquisition requests may need to be converted to many individual requests for different sensors on a mote. The application may allow for a percentage of error on any value returned. Therefore, an application request contains an error percentage δ that is tied to the sensor attribute sought. As the base station will hold values that are sent by the motes, it must be decided the length of time that the value is valid. The concept of *validity*, or lifetime, takes care of this. If a sensor value is sent to the base station with a validity τ, then the base station can return that result with some confidence ϵ when the application requests it. Note that the base station will return only the result, which will implicitly have confidence greater than ϵ, where confidence is the probability the value will be within the requisite data quality allowance.

Some sensors, such as a contaminant sensor, may vary by a small amount between two periodic samplings, whereas others may only be of interest when a dramatic shift occurs, such as in the flow sensor. Thus, the needed percentage differs depending on the type of sensor for a given sampling schedule.

Overall, this means an application request will consist of a tuple of four fields: request = <mote ID, sensor ID, error percentage δ, validity confidence ϵ>. An example of a request would be <Mote 8, Sensor 1, 3%, 95%>, indicating that the application is seeking the sensor value of Sensor 1 off of Mote 8 within 3% of the value at the mote at least 95% of the time.

Pre-specified data models of the sensor data will be used to determine whether a sensor driven update should occur, as well as help in determining the validity attribute. The data model will be assumed to be based on the application and programmed on the motes before the wireless sensor network is installed. There will need to be a sensor data model for each sensor type.

Wireless sensor networks may be set up with many forms of topology. Depending on use, an application may require a network as simple as a single hop network, or multi-hop networks where motes will have their messages relayed through other motes to the base station. In terms of flexibility, a data collection system that is not specific to the network underneath is desirable. Our approach presented in this paper is irrespective of network model.

Problem statement: The problem can be stated formally as follows. Given data models for each sensor type j, error percentages δ_j, validity confidence ϵ, an unknown distribution of application requests $r = < i, j, \delta_j, \epsilon >$ for mote i, cost to transmit and receive a query ($C_{qdu.i}$), cost to transmit a sensor driven update ($C_{sdu.i}$), the objective is to decrease the aggregate cost of responding to all application requests with values meeting δ_j and ϵ for each request. The output back to the application will be a value that is within δ_j at least ϵ percent of the time. In other words, the objective is to decrease energy consumption of sensor data collection given data models and allowed error percentages. To accomplish this, we develop reasonable and efficient methods to approximate the probability of receiving a query driven update as well as to determining the validity lifetime τ of a sensor value.

3. A Hybrid Push and Pull Algorithm for Sensor Reporting

There are two different ways to get data from sensor networks. One is to push, i.e., to let motes report their readings; the other is to pull, i.e., to let the base station send out queries. The overall objective is to determine whether sending an update or waiting for a query has a lower expected energy consumption. Figure 1 shows the overall data flow in our strategy.

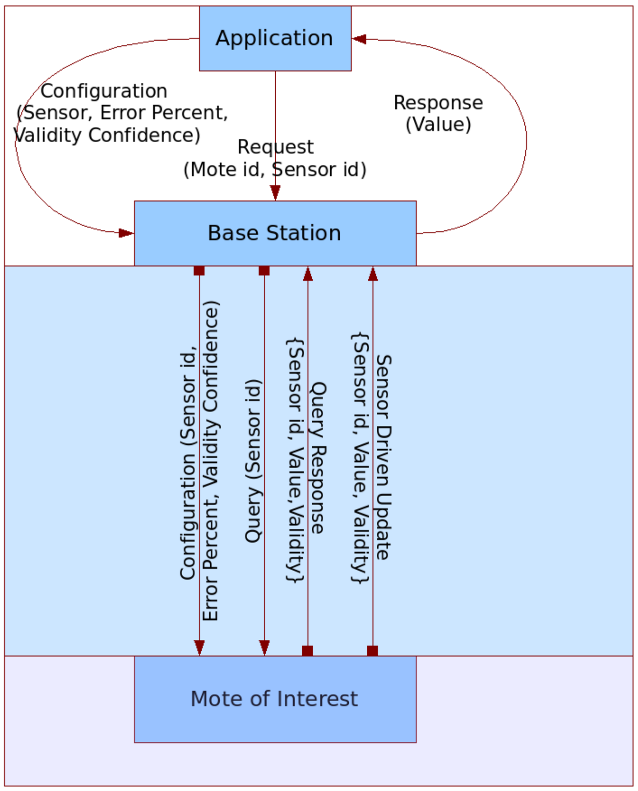

Figure 1. Flow of application requests and data updates

Theoretically, the desired approach would be to compare the cost of a sensor update multiplied by the probability of a sensor update against the cost of a query multiplied by the probability of receiving a query. This intuitively makes sense as when the number of queries from the application is high and the number of sensor updates is low, motes use sensor driven updates and save energy by not updating values that are sufficiently accurate. Alternatively, when sensors exceed the error bounds regularly and would require many updates, or when the application is not requesting data from a sensor frequently, the mote will wait for queries and only expend energy when the application actually needs the data. Therefore, if the following predicate is true, a sensor driven update is sent, where P_{sdu} is the probability of sensor driven updates and P_{qdu} is the probability of query driven updates.

$$P_{sdu} \times C_{sdu} \leq P_{qdu} \times C_{qdu} \qquad (1)$$

This decision will be based upon the error percentage given by the application and changes in sensor values. The error percentage δ_j will create error bounds based upon the current value at the base station $V_{BS.i,j}$. The error bounds for a current value v will be within $\pm\delta_j \times v$ with respect to the last reported value at the base station. If an update is to be sent, a mote can calculate the amount of time a reading will likely remain within the error bounds, and therefore send an update to the base station with a data validity. This will allow the base station to return that value on application request. If the data is no longer valid, the base station will query the mote.

As we can see from the discussions above, the main idea of our algorithm revolves around four main factors: probabilities of query driven(P_{qdu}) and sensor driven updates (P_{sdu}), and costs of query driven (C_{qdu}) and sensor driven updates (C_{sdu}).

The probabilities are non-trivial to calculate, specifically because the number of sensor driven updates are dependent on the number of queries, and vice versa. If a sensor driven update is sent to the base station, for the time that the data is valid, no queries will need to be sent. Alternatively, if a query just obtained data from a sensor, no sensor driven update will be needed until that data is no longer valid.

For the probability of a query, P_{qdu}, an approximation will be used based upon the frequency of queries received at the mote. The probability of a sensor driven update, P_{sdu}, will be approximated by a combination of the sensor data model provided by the application, the current value $V_{Mote.i,j}$, $V_{BS.i,j}$, and δ_j. Using the data model, we can derive the probability $P_{in.i,j,\eta}$, that the sensor will stay within δ_j within an amount of time η. For instance, a simple data model may be to assume that the value will change in a Gaussian fashion with a similar mean and variance to recent history. The application will provide this data model so any amount of complexity is possible. Note that P_{sdu} is approximated by 1 - $P_{in.i,j,\eta}$ as $P_{in.i,j,\eta}$ is the probability when no sensor update is needed. By looking at equal times for estimation of sensor update probability and query update probability, the two approximations can be directly compared.

As the cost of a message is predominant in wireless networks, we use the number of messages sent as an approximation of energy cost. For a query, messages must be sent both from the base station to the mote and back. For a sensor update, only a message from the sensor to the base station needs to be sent. In a typical network, these costs would be calculated as an average number of messages actually sent, including retransmission messages. In a multi-hop network this would require tracking the message cost of using intermediate motes to relay messages between the base station and the mote of interest.

As previously stated, the approximations of the two probabilities require an equal time window into the future so their estimates can be compared. This amount of time is not obvious, as the model will have higher uncertainty further into the future, but the energy savings would be minimal if the time is too short. A good heuristic is to perform the calculation until the next scheduled sampling. This would allow a time in which the model is still able to predict with reasonable certainty, but long enough as no additional messages would need to be sent.

Hence, the criteria for sending a sensor-driven update is as follows, assuming that t is the present time.

$$(1 - P_{in.i,j,\eta}) \cdot C_{sdu.i} \leq \psi_{i,j} \cdot (\eta - t) \cdot C_{qdu.i}$$

If it is decided to use a sensor driven update, the data model can again be used to predict how long the value would remain valid. The mote can send the base station a value and validity time for which the base station can assume that the data is valid. During this validity lifetime, the base station will answer any application requests directly rather than sending a query.

Discussions: The approach is suited to bounding energy costs as the ratio of query frequency against sensor data changes grows large or approaches zero. In the former case, the number of queries coming from the application for a mote is high and thus the mote will be more likely to send data via sensor update. This would be effective as queries will not have to be propagated by messages to the mote. In the latter case, the algorithm is effective in that it will not report frequent data changes when the application is not sending application requests for the mote. When the expected cost of sensor driven updates and queries are roughly equivalent each method will expected to have about the same cost. Note that our approach is independent of underlying network structures, i.e., the approach lies in the data management layer which is above the transport and network layers.

4. Performance Evaluation

A performance evaluation was done to examine the viability of the proposed push and pull hybrid approach in terms of reducing energy usage. Specifically, the evaluation was developed to determine how much energy, measured in number of messages, is used through this algorithm (referred to as 'HYBRID') relative to two more traditional approaches. We also examined under what circumstances the algorithm would perform the best and most poorly compared to traditional approaches. Specifically, the algorithm was compared against two standard approaches: a query driven update only approach (referred to as 'QDU-ONLY') and a sensor driven update only approach (referred to as 'SDU-ONLY'). In the QDU-ONLY approach, each mote only sends data when a query is received. This would only occur when an application request has received by the base station and as such there is no use for error percentage (δ) or a data model. In the SDU-ONLY approach, the sensor will send data to the base station whenever the data at the base station is no longer valid. Note that this approach does not require a specific validity τ to be reported as the mote will send the data only when a new reading is beyond δ_j.

In order to compare the three approaches a simulation was created using NesC and TOSSIM 2.0, the most widely accepted sensor network simulator. The metric used was the total number of messages sent during equal periods of simulation. This was measured at various rates of application requests, sensor change, and application error percentages. Each simulation was run 20 times for 500 simulated seconds. The application requests were modeled by periodically choosing a mote and sensor and then requesting that value from the base station. Additionally, the performance at various levels of sensor error percentages were evaluated.

Sensor values were created as a random walk with a parameterized change period. At each interval, the random walk will move up or down one unit, starting with a uniform distribution in the range of 150 to 250.

A Normal distribution was used for the data model as a random walk distribution converges to a Normal distribution with mean of the initial value and variance of number of steps multiplied by the square of the step size. The mean was taken to be the current value and the number of steps was determined by the range allowed by the sensor error percentage.

Experimental Results: Figure 2 shows the behavior of the three methods when the rate of data change is varied. The simulations showed that HYBRID sends fewer messages than QDU-ONLY in for higher application request rates and better than SDU-ONLY for faster rates of data change. As is expected, QDU-ONLY does not change significantly with the rate of data change, and SDU-ONLY decreases dramatically as the rate of data change decreases. HYBRID is affected by this rate of change, but not to the extent that the SDU-ONLY is affected. As the frequency of change slows down, there is a point at which SDU-ONLY will send fewer messages than HYBRID. One reason for this is when the sensor data change rate approaches the point that queries are never more efficient, both SDU-ONLY and HYBRID will send an update if the sensor value exceeds the error bounds, but HYBRID additionally sends an update if the validity expires.

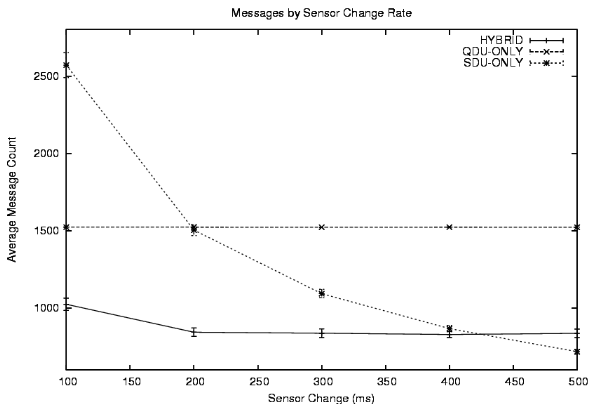

Figure 2. Impact of data change rate on the data collection overhead (query period = 1 s, application error percentage =3%) with 95% confidence interval.

Figure 3 shows the behavior of the the three methods when the rate of application requests is varied. As expected, SDU-ONLY is not affected by the application request rate, and QDU-ONLY dramatically decreases messages sent as the frequency diminishes. HYBRID also decreases greatly as the frequency diminishes, however not at the same rate as QDU-ONLY. Shown in this figure is the point where QDU-ONLY will be more efficient, however the approaches remain competitive thereafter. Both HYBRID and QDU-ONLY decrease along an exponential curve as the application request period increases, however the QDU-ONLY is a steeper exponential curve. One reason for this is that the HYBRID approach uses the average number of queries in deciding whether to send an update or not, the more queries that are sent pushes the HYBRID approach back toward using sensor driven updates.

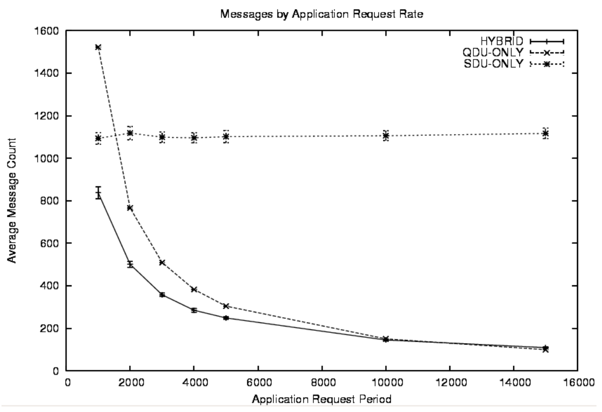

Figure 3. Impact of application request frequency on the data collection overhead (sensor change period = .3 s, application error percentage = 3%) with 95% confidence interval.

Figure 4 shows the behavior of the three methods when the application error percentage is varied. As expected, QDU-ONLY is not affected by the change. Both SDU-ONLY and HYBRID are affected by the change in the error percentage, however similarly to the change in sensor period, the SDU-ONLY mode is affected more than the HYBRID method. In fact, the HYBRID method appears to level out as the sensor percentage grows large. This indicates that for larger error percentages SDU-ONLY will outperform HYBRID. However, for more stringent percentages, such as 3% or 5%, HYBRID will outperform or be competitive with SDU-ONLY. For percentages greater than this, it would be better to use an approach such as SDU-ONLY. The reason for this is similar to the reason a longer sensor change period benefits SDU-ONLY more than HYBRID. Whether a larger error percentage or a longer average sensor change period, the length of time that the value will likely remain in range increases.

Discussions: As stated previously, our approach also works for multi-hop networks. To keep it simple, our evaluation so far has only been conducted on a one-hop network, intending to merely validate our approach. In practice, many sensor networks use clustering based or multi-hop based communication. To obtain results for non one-hop networks, a few key changes would need to be made. The relaying nodes could evaluate if adding their data to relayed packets would be efficient. Furthermore, nodes that do not communicate directly with the base station would require either to develop a protocol or to have a MAC layer that would provide an adequate estimate of energy used to send a message to or receive a message from base station. Many strategies are available for such information, however, they may require additional energy being spent due to increased packet size or administration messages. One possible future evaluation would be to examine which strategies would perform most efficiently in concert with the HYBRID approach.

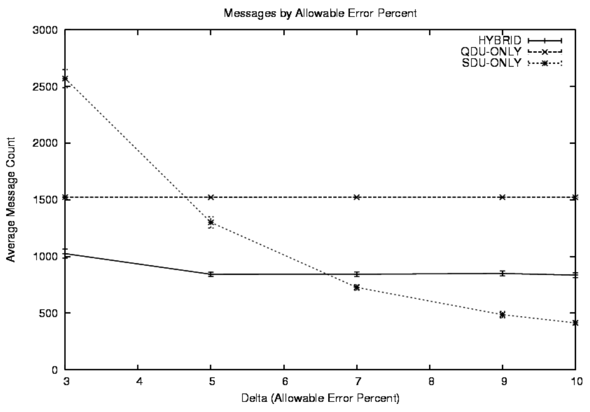

Figure 4. Impact of application error percentage on the data collection overhead (query period = 1 s, sensor data change period = .1 s) with 95% confidence interval.

Another extension of the performance studies is to use other distributions of application requests and sensor changes.

- For application requests, the distributions that we used were a uniform distribution across the motes, and a distribution that favored certain motes with a higher probability than others. While the results were similar across these two distributions, a more thorough investigation into more realistic application requests distributions would be valuable. Specifically, if application requests come in spurts, this algorithm may prove highly beneficial.

- For the sensor change distribution, the nature of the data change would ideally be captured by the application data model, however, additional tests for distributions that have no well suited model would also be of interest to examine how the HYBRID approach fares. Further examination of different distributions with the current data model may also prove valuable in determining the effectiveness of the HYBRID approach in validating the quality of a data model in an on-line fashion.

- While some investigation into the effects of sensor error tolerance was performed, additional study into the effects of changing the acceptable level of data quality should be conducted.

5. Related Work

The most relevant work to ours is KEN [1] that uses approximate queries to reduce the energy consumption. However, they do not explore reduction of energy by querying from the server when that is more efficient. Rather, they simply attempt to always keep the base station within the accuracy bounds. Yates et al. [10] develop several spatial caching and approximation policies to reduce energy use for queries. Additionally they explore quality in terms of latency and accuracy. However, they pull data as opposed to both pushing and pulling, limiting their efficiency in some cases.

Push and pull hybrid techniques have been explored before to reduce energy consumption. Some of these act by partitioning the network into push only or pull only sections [6, 8] or by deciding the push pull balance based upon comparison to a set filter [9] or rate value [5]. More relevant to this paper are those that have examined the balance between push and pull at a mote level. Silberstein [7]examines a push pull balance by query type. Our work differs as our hybrid approach is independent of the type of query. An approach of using the overall model of the data as viewed from the base station in order to reduce query necessity has also been explored. In this method, a model is kept at the base station and reduces the need to query every mote[3]. This approach assumes dense network deployment and strong spatial correlation. In Han et al. [4], the use of concrete data error bounds is explored with its interaction with various mote state models. Our paper differs in several ways: sensor state management is not explicitly considered; each sensor maintains an instantaneous value rather than an approximate range; a data model is assumed to be available for each sensor.

6. Conclusion

This paper exploits applications' quality of data tolerance to drive a choice between sensor driven updates and queries from the base station, aiming to reduce energy consumption. The incorporation of sensor data models allows for not only further energy savings, but also online evaluation of those models. Performance studies showed that our proposed approach, HYBRID, is more energy efficient than QDU-ONLY for frequent query rates, and is often more efficient than SDU-ONLY for more stringent error tolerance and faster rates of data change. The benefits of using HYBRID is even more prominent as the variance of application requests or data change across motes increases. This is because both other methods perform poorly when system conditions are more unpredictable. One exception is that in cases where the error percentage is large, the SDU-ONLY approach is more efficient than the HYBRID approach. Our proposed approach provides much higher flexibility than blindly reprogramming the motes to be either SDU-ONLY or QDU-ONLY.

Further development of the HYBRID approach includes better use of data model approximations for quality aware applications, and an exploration into the expected cost across all sensors on a mote, rather than one at a time. Thorough performance studies including the use of more realistic application and data models are also necessary to evaluate the effectiveness of the proposed approach. Further validation and testing on a physical sensor network test bed for subsurface contaminant monitoring remains part of our future work.

References

[1] D. Chu, A. Deshpande, J. M. Hellerstein, and W. Hong. Approximate data collection in sensor networks using probabilistic models. In *ICDE*, page 48. IEEE, 2006.

[2] A. Deligiannakis, Y. Kotidis, and N. Roussopoulos. Hierarchical in-network data aggregation with quality guarantees. In *EDBT*, 2004.

[3] A. Deshpande, C. Guestrin, S. Madden, J. Hellerstein, and W. Hong. Model-based approximate querying in sensor networks. *The VLDB Journal*, 14, 2005.

[4] Q. Han, S. Mehrotra, and N. Venkatasubramanian. Application-aware integration of data collection and power management in wireless sensor networks. *J. Parallel Distrib. Comput.*, 67, 2007.

[5] S. Kapadia and B. Krishnamachari. Comparative analysis of push-pull query strategies for wireless sensor networks. In *DCOSS*, pages 185–201, 2006.

[6] W. Liu, Y. Zhang, W. Lou, and Y. Fang. Managing wireless sensor networks with supply chain strategy. *qshine*, pages 59–66, 2004.

[7] A. Silberstein. Push and pull in sensor network query processing. In *SWDIM*, 2006.

[8] N. Trigoni, Y. Yao, A. J. Demers, J. Gehrke, and R. Rajaraman. Hybrid push-pull query processing for sensor networks. In *GI Jahrestagung (2)*, LNI, pages 370–374. GI, 2004.

[9] M. Wu, J. Xu, X. Tang, and W.-C. Lee. Monitoring top-k query inwireless sensor networks. In *ICDE*, page 143. IEEE, 2006.

[10] D. J. Yates, E. M. Nahum, J. F. Kurose, and P. Shenoy. Data quality and query cost in pervasive sensing systems. In *PerCom*, pages 195–205. IEEE, 2008.