Data Management Issues in Disconnected Sensor Networks

Wolfgang Lindner
MIT CSAIL
wolfgang@csail.mit.edu

Samuel Madden
MIT CSAIL
madden@csail.mit.edu

Abstract: The possibility of disconnection is one of the fundamental new networking problems presented by sensor networks. The goal of this paper is to address the problem of *result collection* in *periodically disconnected* sensor networks, focusing, in particular, on opportunities for in-network processing. Our hypothesis is that, through careful augmentation of the networking layer underneath the declarative query processor, it will be possible to expose information that will make the query processor able to adapt to and handle disconnections.

1 Introduction

There are many types of sensor-network deployments in which disconnections will be frequent and expected: in networks where the nodes are mobile, for example, nodes will become occasionally disconnected from each other. Similarly, in remote sensing environments, intermittent connections may be the only practical way to get data out of a network – e.g., when a satellite passes overhead or when a mobile node occasionally visits the remote site. To support operation in such environments, we propose extending the networking capabilities of current sensor network query processors with explicit support for disconnection. In particular, we focus on the subclass of periodically disconnected networks where a connection recurs on a frequent bases.

2 Related Work

To the best of our knowledge, no one has proposed to augment a query processing system in this way before. There has been some work on solutions for handling disconnection in sensor networks, though no specific efforts at integrating such support into a query processing system. For example, in the ZebraNet [JOW+02] project, the goal is to develop mobile sensornet software and hardware that can be used to track Zebras as they roam about a wildlife preserve in Africa. Each node periodically collects GPS data, and logs it to local non-volatile storage. When zebra-based nodes come into contact with each other, they (selectively) exchange data. Eventually, a zebra will wander into the range of a base station, where the data from its sensor and any other sensors it came into contact with can be offloaded. A similar concept is that of a Data MULE [SRJB03]. Both systems do no in-network processing, and are not seeking to build a reusable infrastructure for disconnected networking; rather, they are focused on solving their particular mobile-node problem in an application specific manner.

There has also been some work in the networking community on disconnected operation, like predicting the location of a mobile host in an ad-hoc network [LR00] so that data can be efficiently delivered to that host, despite occasional disconnections. Research on Delay-tolerant Networks (DTNs) [Fa03] addresses a number of issues surrounding adding disconnection support to traditional networks, but does not discuss any form of in-network processing.

3 Challenges in Disconnected Sensor Networks

When the possibility of disconnection is introduced, the process of how queries are processed and optimized must be changed in several fundamental ways, introducing a number of new requirements and challenges. These include:

Dissemination: How can we ensure that every node in the network receives a query despite intermittent disconnections? This is related to issues of reliable broadcast in sensor networks [HJB03, LS04], but requires additional techniques to deal with longer-duration partitionings.

Result Collection: In a disconnected network, a node may not be able to communicate its readings to a preselected parent. One option would be for nodes to simply buffer results until the network is reconnected, but this could be problematic due to storage shortages and suboptimal given that it may be possible to combine readings with those of connected neighbors.

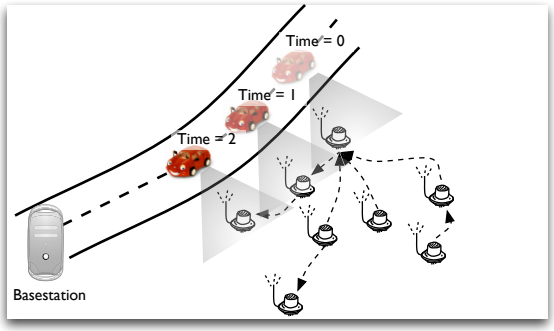

Figure 1: As the car drives along the road, different sensors come into contact with it. Deciding how to stage data for deliver to mobile nodes on vehicles is an important research challenge in disconnected sensor networks.

Duplicate Elimination: One of the problems with network disconnection is that it may cause a node to not know whether the data it attempted to send was properly delivered. This can happen, for example, if disconnection happens while a node is in the middle of sending a message. One option is to simply retransmit these indeterminate packets, but this can lead to the introduction of duplicate results that skew query answers. This problem is especially tricky when performing in-network aggregation of results. Since we do not know what records have already been aggregated common techniques, like eliminating duplicates on layer 2, are not a solution. Therefore, to perfectly eliminate duplicates, each aggregate record must be tagged with the sensors that contributed to it (dramatically increasing the size of the transmitted data at mitigating the benefits of in-network aggregation.)

Periodic Disconnections: A final challenge related to disconnection arises in networks that disconnect and reconnect at periodic, known intervals. In such networks, it is desirable to *stage* results at the parts of the network where the reconnection will happen. In some cases, reconnection may happen in a known temporal pattern, across several nodes as, for example, a data-collection vehicle drives through the network. Figure 1 illustrates this – as the car drives along the road, different sensors with the network are in contact with it.

4 High-level Facilities for Result Collection

In this section, we summarize the high-level facilities we have begun to design to address the problem of *result collection* in *periodically disconnected* sensor networks with in-network processing capabilities. Because of the limited space we do not address issues of query dissemination or duplicate detection and elimination within this paper.

Collaborative Data Storage: The traditional TinyDB [MHHF] model assumes that nodes can offload data as soon as it is produced, requiring no storage. With disconnections, nodes may need to buffer data for substantial periods of time, requiring them to store data. Because the storage on nodes is limited, sensors on one side of a partition may need to pool their storage to provide sufficient capacity for all readings.

Proactive Aggregation: When disconnection occurs, nodes on one side of the partition should attempt to partially aggregate their results as much as possible so that a minimum of data must be stored. This reduces storage load and also minimizes the amount of data that must be transmitted when reconnection occurs. Both collaborative storage and proactive aggregation require facilities for identifying neighboring sensors with available storage or data to be aggregated, and for optimizing the placement of storage and aggregation operators. Figure 2 shows an example of proactive aggregation; nodes B and C begin routing their data to node A (rather than directly to the basestation) after a disconnection. When a reconnection occurs, A can transmit the already aggregated data directly to the basestation. Note that there are many other possible strategies; for instance, the aggregation could have been done at nodes A, B, and C so that, no matter which node reconnected first, the aggregate data could be immediately delivered.

Multiresolution Data Transmission: Because re-connections may be short lived, we believe it is important to explore data transformations that produce multi-resolution data structures. An example of such a transform is the *Haar wavelet* transform. Because it encodes data as a binary tree of differences from a mean, such that the root of the tree is the mean, the left child, l, is the average difference of the left half of the data from the mean, the left child l' of l is the average difference of the left-most quarter of the data from the value at l, and so on it allows us to get the most significant coefficients out of the network. Figure 3 illustrates a simple graphical example of the Haar wavelet; notice that the 8-reading signal at the bottom of the figure can be reconstructed from the $u, d_1, d_2, d_3, d_4, d_5, d_6, d_7, d_8$ sequence, and that any prefix of this sequence can be used to reconstruct a coarser approximation of signal. There are many other examples of such transforms, ranging from other types of wavelets, to query driven specifications of ranges of readings that are most important.

Multilocation Data Staging: In situations with periodic, predictable disconnections, we would like to store data at several locations to maximize the probability that certain readings are delivered or ensure that as much data as possible is brought out of the network during each reconnection. For example, looking at Figure 1, the strategy that will maximize the amount of data delivered is one that spreads data across the three sensors adjacent to the road.

Choosing where data should be staged and how much data to store at each node presents an interesting optimization problem. If the connection schedule and amount of data to be delivered is known *apriori*, a simple greedy approach that stores as much data at each sensor as that sensor can deliver will suffice. However, when the exact schedule or duration of reconnection is unknown or only partially known, there are a number of possible tradeoffs. For example, it may be desirable to stage high-priority data at several locations to maximize its probability of delivery. Or, probabilistic approaches that assign data to sensors based on their expected probability of being able to deliver it may come into play.

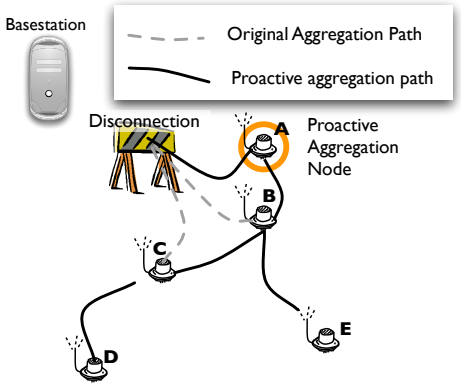

Figure 2: Proactive aggregation. Before disconnection, nodes A,B, and C routed directly to the basestation. After disconnection, node A is chosen to receive all of the aggregation results.

Predictive Modeling: Probabilistic modeling has a second important role in disconnected networks. In a highly dynamic, frequently disconnected network, it may be preferable to predict sensor readings at unknown locations or times rather than bring every piece of data out of the network. This prediction might be done using historical data – for instance, by tracking the correlations between pairs of sensors and using readings from correlated nodes when actual readings are not available. Alternatively, it could take the form of building temporal models ("trajectories") that make it possible to guess the reading from a sensor into the near future. We have begun to study the use of such predictive models in other aspects of our work [GBT+04, DHGM04], and they are widely used in the machine learning and AI communities [CDLS99], but have not yet been applied in this kind of disconnected environment.

5 Conclusion

Thus, there are a number of challenging research problems related to query processing in disconnected environments. In particular, there are a number of algorithmic alternatives that must be explored to understand the possible tradeoffs, such as:

1. Which probabilistic model is most effective at predicting values of disconnected parts of the network?
2. What is the appropriate policy for choosing where to stage data given a regular but

not completely predictable reconnection schedule?

3. What is the best scheme for proactively aggregating data? How well does proactive aggregation work compared to simple store-and-forward techniques?

To verify our hypothesis we are currently building disconnection-sensitive features into TinyDB and plan to deploy the system, both under controlled laboratory settings and in the field. Although most of the techniques described in this paper are implementable on today's hardware platforms, one of the risks we face is that tiny, embedded microprocessors will not gain sufficient compute power to support, e.g. complex wavelet transforms, in the next few years. However, based on the hardware advancements made in the motes over the past two years (8x improvement in memory capacity, 2x improvement in processor speed), we expect that this will not be as large of a problem in the future.

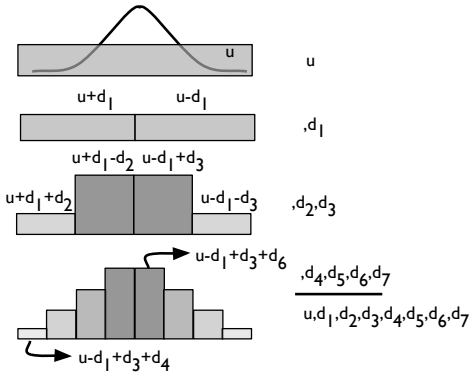

Figure 3: An example of a Haar wavelet. Notice that the coefficients $u, d_1, d_2, d_3, d_4, d_5, d_6, d_7, d_8$ are enough to reconstruct the bottom-most histogram, and that any prefix of those coefficients provides an approximation of the signal.

References

[CDLS99] Cowell, R., Dawid, P., Lauritzen, S., and Spiegelhalter, D.: *Probabilistic Networks and Expert Systems*. Spinger. New York. 1999.

[DHGM04] Deshpande, A., Hong, W., Guestrin, C., and Madden, S.: Exploting Correlated Attributes in Acquisitional Query Processing. In Submission. 2004.

[Fa03] Fall, K.: A Delay-Tolerant Network Architecture for Challenged Internets. In: *ACM SIGCOMM 2003*. August 2003.

[GBT+04] Guestrin, C., Bodik, P., Thibaux, R., Paskin, M., and Madden, S.: Distributed Regression: an Efficient Framework for Modeling Sensor Network Data. In: *IPSN*. March 2004.

[HJB03] Hull, B., Jamison, K., and Balakrishnan, H.: Bandwidth Management in Wireless Sensor Networks. In: *ACM SenSys*. October 2003.

[JOW+02] Juang, P., Oki, H., Wang, Y., Martonosi, M., Pehand, L., and Rubenstein, D.: Energy-Efficient Computing for Wildlife Tracking: Design Tradeoffs and Early Experiences with ZebraNet. In: *ACM ASPLOS*. October 2002.

[LR00] Li, Q. and Rus, D.: Sending Messages to Mobile Users in Disconnected Ad-hoc Wireless Networks . In: *ACM MOBICOM*. August 2000.

[LS04] Levis, P. and Shenker, S.: Epidemic Algorithms for Code Distribution in Sensor Networks. In: *NSDI*. April 2004.

[MHHF] Madden, S., Hong, W., Hellerstein, J. M., and Franklin, M. TinyDB web page. http://telegraph.cs.berkeley.edu/tinydb.

[SRJB03] Shah, R. C., Roy, S., Jain, S., and Brunette, W.: Data MULEs: Modeling a Three-tier Architecture for Sparse Sensor Networks. In: *IEEE Workshop on Sensor Network Protocols and Applications (SNPA)*. May 2003.