An Efficient Data Model for Sensor Networks

Janak Mathuria, Chang-Tien Lu, Jing Dai
Department of Computer Science
Virginia Polytechnic Institute and State University
Falls Church, VA22043, USA
{janakm, ctlu, daij}@vt.edu

Abstract

Processing queries "in place" over sensor networks lends itself to optimization techniques that reduce the number of messages required for fulfilling the query. This can lead to significant reduction in power consumption, a precious commodity in such networks, thereby extending the life of these networks. In this paper, we examine the efficacy of some known optimization techniques using a simulator. We also present a new "silent message" optimization technique and utilize query normalization technique from traditional databases to support queries of greater complexity.

Keywords: sensor networks, database, SQL, query optimization.

1. Introduction

Sensor networks are expected to be central to future advances in the prevailing information age. An almost countless number of such networks of varying sizes are expected to be deployed for monitoring and tracking in a host of domains such as earth and environmental sciences, traffic management, public safety and health, industrial process control to name just a few. Present day sensor networks are comprised of tiny nodes that are equipped with a radio, a processor, and some sensing devices. They run on power supplied via a battery pack, whose life effectively determines the life of the sensor. The radio allows sensors to send and receive messages over only a limited range based on broadcast. Communication is unreliable due to various disturbances and is also very expensive. Transmitting a single bit of data may consume as much battery as executing 800 instructions. A typical battery pack would be exhausted in just over 2 months if the sensor continuously transmits data. Thus minimizing communication is of paramount importance to having long-living sensor networks.

Sensors in the near future are expected to be a lot cheaper, have greater computing power and memory, and have better ability to withstand adverse environment and terrains. This is expected to lead to a huge explosion in their deployment. At the same time, improvements in communication and battery life are not expected to keep pace. Thus minimizing communication is expected to be the focus of much research in the field of sensor networks.

In most present and future models of sensor networks, data gathering remains one of the primary functions of a sensor network. Given that sensor networks have limited memory, computing power and are generally not very reliable, a common approach has been to simply feed the sensor readings into a traditional database or data-warehouse. This approach has certain drawbacks. One drawback is that simply sending all observations to a database typically requires a lot of communication since each observation has to go through multiple hops. Another drawback is that needing to communicate to a database makes the sensor network less autonomous.

In this paper, we examine the motivation, benefits, challenges, and implications of processing queries "in place" in sensor networks. We thus view the sensor network as a database. We discuss the similarities between sensor networks and traditional databases and also note the differences. We define an SQL-like query language for sensor networks and then present a "silent message" optimization technique that exploits the predictability of the next sensor reading to reduce communication. We review the efficacy of the various optimization techniques using a simulator we developed for this purpose and we finally discuss related work and conclude our work.

2. Sensor networks as databases

2.1. Similarities

One of the most distinctive characteristics of traditional database systems is that they are self

describing. This facilitates construction of non-application-specific applications such as query processing tools to learn about the database. Sensors networks may also be self-describing. Sensors can respond to queries about what attributes they observe. Each distinct type of sensor may be viewed as a distinct relation.

A traditional database provides a separation of the physical, conceptual and logical layers. The physical level is generally not visible to the user. A self-describing sensor network could easily provide separation of physical and conceptual layers. Database-like access control lists can be implemented to provide separation of the logical level by providing views of the data to different users.

Both databases and sensor networks lay great emphasis on efficient access methods. Sensor networks lay a great deal of emphasis on reducing communication to conserve power. Accessing data is by far the most common operation in databases as well as sensor networks. Thus efficient access to data is of paramount importance in both.

2.2. Differences

As compared to traditional databases, sensors are massively distributed stores. This difference manifests by making query processing in sensor networks much more communication bound than in traditional databases.

Sensors are also extremely tiny stores of data. Typical memory on sensors is 4KB and while expected to increase substantially, it would always be infeasible to near the capacity of database drives. Additionally, sensors are also highly unreliable. Given these limitations, sensors are always likely to store a limited number of past readings temporarily.

Query characteristics in sensor networks also differ from those in traditional databases. In sensor networks, the retrieval query (SELECT) is registered and returns results at specified intervals. Additional to this, the semantics of insert, update, and delete operations differ to a much greater extent. In most cases, sensor networks only read values and do not necessarily have any control over the values. A very important implication of this continuous nature is that traditional query plan approaches is not necessarily suitable for query optimization. Instead techniques that adapt to changing data patterns are required. There has been a great deal of research [10, 12, 13] in this area especially as pertains to web databases. Eddies appear particularly suitable for sensor networks. They also facilitate sharing of data and processing over multiple queries.

3. SQL-like query language

Given that other than the continuous nature of data and queries, the other basic semantics of the retrieval operation are the same, we use a SQL-like query language for querying sensor networks. We add some extensions to support the continuous nature of queries. The basic structure of the SQL-like query statement follows:

```
<select-stmt> :=
        <select-query>
        EVERY <time-interval>
        [FOR <time-span>]

<select-query> :=
    SELECT <column-list>
    FROM <relation-list>
    [WHERE <predicate>]
    [GROUP BY <group-by-list>]
    [HAVING <predicate>]
    [UNION | MINUS | INTERSECT select-query]
```

The <column-list> may contain attributes or aggregates in addition to arithmetic operations. Aggregates may be one of the standard aggregates as SQL or user-defined aggregates. The <relation-list> may contain any relation supported in the sensor network and <predicate> is the typical WHERE clause in SQL, so are the GROUP BY and HAVING clauses.

The EVERY clause and the optional FOR clause are new clauses that apply to the statement as a whole. The EVERY clause needs to be used to specify the sampling interval. The optional FOR clause may be used to specify the number of samplings to retrieve.

4. Silent messages

Given that sensor networks are likely to be widely used for monitoring, one would typically expect the readings of individual sensors to change in a predictable manner. This is especially true when the monitoring data is used for control. This predictability of readings can be exploited to reduce the number of messages required to process a continuous query. Children initially provide their readings to the parents. The child only sends its next reading to the parent in case it is not within an expected range. If the parent does not receive a reading from the child, it simply uses the expected value for its computation. Note that this hypothesizing and message elimination can bubble up all the way to the root node. For long running

continuous queries, this can lead to substantial reduction in communication. In case the expected value is not the same as the previous value, for example, it is expected to increase by a constant factor, computing expected values for aggregates such as SUM would require additional processing. Another important consideration is that receiving no messages may indicate that a sensor is no longer unavailable. Thus a mechanism is needed that ensures against this. One possible approach is to attach a probability to next reading not being within expected range. Say this probability is 0.1. In this case, the parent would wait for 10 cycles (10 / 0.1) without receiving a message. If after 10 cycles, it has not heard from a particular child, it would treat the child as unavailable and initiate a rebuild. In cases where the next expected value differs per node, as described above, the child could transmit the expected values till the next transmission along with each transmission.

5. Simulator

We have developed a simulator for studying the efficacy of various query optimization techniques, in order to have our own framework as the basis for conducting experiments for further research.

In keeping with the view to provide a framework, we allow the user to provide values for various characteristics of a sensor network, such as radio range, data transfer rate, battery consumption, available memory, relations and attributes, expected range of values and probability of within value range.

When a configuration is loaded, sensors are arranged using random numbers and a routing tree is constructed. The root node is selected at random. The root node then sends out a broadcast message to invite sensors to be its children. Each sensor that receives this message responds back to the root by accepting the invitation and in turn broadcasts its own message to invite other sensors to be its children. This process continues till all sensors have joined the routing tree. When a single sensor receives multiple invitation messages around the same time, it would select the closest sensor. The simulator displays the resulting routing tree.

5.1. Preparing queries

Users can execute queries by selecting the Tools|Query menu option. The syntax of the query is as described in section 3 with some differences. An additional OPTIMIZATIONS clause is allowed to choose optimizations among in_network, silent_messages, snooping and packet_merging to use. The interval for the EVERY clause can only be specified in terms of seconds and the value specified for the FOR clause is the number of time-points before the query is cancelled. The input query is parsed, syntactic and semantic checks are performed and if the query is valid, it is normalized.

5.2. Executing queries

The root sensor then extracts operations and passes on them to its children which in turn pass them on to their children. All sensors including the root sensor perform the necessary processing. The simulator displays the results for each time-point in a list control. A timer is used to wake up the sensors to obtain the results for the next interval. This process continues till either the specified number of cycles complete or the user cancels the query.

5.3. Results

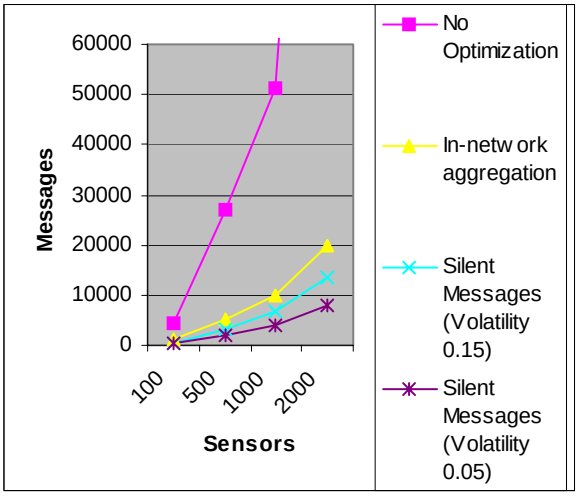

Figure 1: Results

We carried out experiments on networks of varying sizes with and without the in-network and silent messages optimizations. We used a simple query for this purpose; SELECT AVG(SampledValue) FROM Pollen EVERY 1 FOR 10. Results are presented below:

As can be seen, not performing in-network aggregation scales very poorly as the size of the network grows. Silent message optimization reduces the number the number of messages as compared to just in network processing. As volatility decreases, the silent message technique performs better as one would expect.

6. Related work

Querying sensor networks has been a fertile area of research. Using SQL-like language for querying sensor networks and various techniques for query optimization including in-network aggregation, snooping and hypothesis testing have been presented by Madden et al [1]. The directed diffusion paradigm to achieve energy savings for distributed sensing is presented by Intanagonwiwat et al [2]. Various designs for query processing in sensor networks have been proposed that discuss routing algorithms as well as optimization techniques [8]. Routing algorithms for ad-hoc networks have been studied for mobile networks and have direct applications to sensor networks [6]. Bonnet et al have also examined treating sensors as databases [7].

Adaptive query processing techniques have been applied to web and streaming databases and have been used in projects on sensor networks [5, 9]. Chandrasekaran et al study continuous query processing over data that has arrived prior to the query as well as future data [11]. Transforming queries while maintaining equivalence has been researched and used extensively in traditional databases [3, 4, 14].

7. Conclusion and future work

The results from our initial experiments support the argument for performing queries in place in sensor networks. We also see that the silent message optimization can greatly reduce number of messages required to process continuous queries when the sensor data is expected to be largely predictable. Query normalization techniques allow us to support more complex queries.

We expect to continue extending our work in this area. We have not broached the topic of clock synchronization, sensor failure, and transmission failures. We expect to continue enhancing the simulator to support correlated queries and some other query optimization techniques, and model more realistic scenarios.

8. References

[1] S. Madden, M. J. Franklin, J. Hellerstein, and W. Hong, "TAG: a Tiny AGgregation Service for Ad-Hoc Sensor Networks", 5th Annual Symposium on Operating Systems Design and Implementation (OSDI), December 2002.

[2] C. Intanagonwiwat, R. Govindan, D. Estrin, "Directed diffusion: A scalable and robust communication paradigm for sensor networks", In MobiCOM, August 2000.

[3] U. Dayal, "Of nests and trees: a unified approach to processing queries that contain nested subqueries, aggregates and quantifiers", in VLDB 1987.

[4] P. Seshadri, H. Pirahesh, and T. Y. C. Leung, "Complex query Decorrelation", in ICDE 1996.

[5] S. Madden and M. J. Franklin, "Fjording the stream: An architecture for queries over streaming sensor data", In ICDE, 2002.

[6] V. D. Park and M. J. Corson, "A highly adaptive distributed routing algorithm for mobile wireless networks", In INFOCOM, 1997.

[7] P. Bonnet, J. Gehrke, and P. Seshadri, "Towards sensor database systems", In Conference on Mobile Data Management, January, 2001.

[8] Y. Yao, J. Gehrke, "Query Processing for Sensor Networks", Proceedings of the 2003 CIDR Conference.

[9] S. Madden, M. Shah, and J. Hellerstein, "Continuously Adaptive Continuous Queries over Streams", ACM SIGMOD, June 2002.

[10] P. J. Haas and J. Hellerstein, "Ripple Joins for Online Aggregation", In ACM SIGMOD, June, 1999, pp. 287-298.

[11] S. Chandrasekaran and M. J. Franklin, "Streaming Queries over Streaming Data", Proceedings of the 28th VLDB Conference, 2002.

[12] R. Avnur and J. Hellerstein, "Eddies: Continouously adaptive query processing", in ACM SIGMOD, May 2000, pp. 261-272.

[13] T. Urhan and M. J. Franklin, "'{XJ}oin: Getting Fast Answers From Slow and Bursty Networks", University of Maryland Computer Science Technical Report, February 1999.

[14] A. Aho, Y. Sagiv, and J. Ullman, "Equivalence of relational expressions", SIAM Journal on Computing 8[2], 1979.