Interfacing Sensor Network to Grid Computing with Database Support

RICHARD A. WASNIOWSKI
Computer Science Department
California State University
Carson, CA 90747, USA

Abstract: - The widespread distribution and availability of small-scale sensors, actuators, and embedded processors is transforming the physical world into a computing platform. Sensor networks that combine physical sensing capabilities such as temperature, light with networking and computation capabilities are becoming ubiquitous. Applications range from environmental control, warehouse inventory, and health care to scientific and military scenarios. In this paper, we discuss a model for database support for interfacing sensors to grid computing. Stored data are represented as relations. We also discuss the design and implementation of experimental implementation the sensor database support system.

Key-Words:- Database, sensors, query, data streams, grid computing.

1 Introduction

In this paper we discuss the integration of sensor networks with existing grid computing services. Specifically we are looking at how we could extend grid services which focuses on computation resource usage rather than data resource usage with sensor networks and database support. As an integration mechanism, the Globus [6] toolkit is used to interface between Grid and sensors. We have developed a prototype to interface grid computing with a sensor network. Our prototype performs all the coding and the addressing that are required between the two systems, thus allowing the system to work with many types of sensors. A sensor network is a distributed sensing technology with limited processing ability and communications. Grid computing is similar to a web service but with more control due to the extra services it provides. Both grid computing and sensor networks are ongoing areas of research with collaboration from around the world. Grid computing looks at utilizing the vast amount of resources that are spread across networks, while sensor networks utilize the collections of simple devices to cover large areas at a low cost. While sensor networks use the grid resources, the grid uses the data collection by the sensor networks. Different sensor networks format the data being sent in different ways requiring integrating schema. Schema is created in this project for defining the format of the data, and for the XML parser to be used. The Globus toolkit is used to develop the grid side of the system, and the sensor network is simulated using simulators.

2 Motivation

Integrating sensor networks with the traditional grid poses several challenges. The technical challenges are related to the development of sensors and sensor network infrastructure, including the need to comply with emerging APIs for grid and Web services. Process-driven challenges, which are related to the development and adoption of new models and applications, are driving this technology. The challenges within the integration project relate to that fact that sensor networks and grid computing are very different systems. For instance, grid is not concerned about how processor intensive the service is. It uses complex protocols to solve interoperability issues that are acceptable for the targeted environment with no restrictions on power supplies. On the other hand, sensor networks are designed with very energy efficient protocols, and the data being sent is in a compressed form. Therefore, less data is sent overall thus a decrease in power usage overall. CPU usage is kept to a bare minimum to save power, and interoperability is of little concern due to increasing overheads in both CPU usage and bandwidth requirements. Most of sensor network systems involve monitoring answers to continuous queries over data streams produced at physically distributed locations, and most previous approaches require streams to be transmitted to a single location for centralized processing. Unfortunately, the continual transmission of a large number of rapid data streams to a central location can be impractical or expensive. TinyDB allow users to extract useful information from a sensor network using aggregation queries. These systems use in-network aggregation to reduce transmission cost, hence reduce the energy consumptions of the network. See the references [1-10].

3 System Overview

Sensor networks consist of very large numbers of low-cost devices, each of which is a data source, measuring some quantity the object's location, or for example the ambient temperature (see Fig 1)

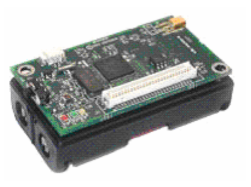

Fig.1 Sensor element from Crossbow Inc.

These networks provide important data sources and create new data management requirements. For instance, these sensors are generally self powered, wireless devices. Such a device draws far more power when communicating than when computing. Thus, when querying the information in the network as a whole, it will often be preferable to distribute as much of the computation as possible to the individual nodes. In effect, the network becomes a new kind of database, whose optimal use requires operations to be pushed as close to the data as possible. Query execution on sensor networks requires a new capacity: the ability to adapt to rapidly changing configurations, such as sensors that die or disconnect from the network. In sensor networks individual sensor nodes are connected to other nodes in their neighborhood through a wireless network, and they use a multihop routing protocol to communicate with nodes that are spatially distant. Sensor nodes also have limited computation and storage capabilities: a node has a general-purpose CPU to perform computation and a small amount of storage space to save program code and data. A sensor node has one or more sensors attached that are connected to the physical world. Example sensors are temperature sensors or light sensors. Thus each sensor is a separate data source that generates records with several fields such as the id and location of the sensor that generated the reading, a time stamp, the sensor type, and the value of the reading. Records of the same sensor type from different nodes have the same schema, and collectively form a distributed table. The sensor network can be considered a large distributed database system consisting of multiple tables of different types of sensors. Sensor data might contain noise, and it is often possible to obtain more accurate results by data fusion from several sensors. For example, when monitoring the concentration of a dangerous bio-chemical in an area, one possible query is to measure the average value of all sensor readings in that region, and report whenever it is higher than some predefined threshold.

3.1 TinyOS

We consider the sensor network as a large distributed database system, namely sensor database. Recent development of sensor database systems has attracted more and more interests in the querying performance for sensor network. TinyDB is a sensor database system developed at Berkeley for the project called TinyOS. The contribution of TinyDB is the design of an acquisition query processor for data collection in sensor networks. They use in-network aggregation and are able to significantly reduce power consumption over traditional passive systems. A simple extension to SQL has been done for controlling data acquisition, and they show how acquisition issues influence query optimization, dissemination, and execution. For example, in the TinyDB system, there is a base station directly connected to a sensor designated as the root node. Aggregate queries over the sensor data are formulated using a simple SQL-like language, and then distributed across the network. As the query is distributed across the network, a spanning tree is formed for the sensors to return data back to the root node. At each node in the tree, the sensor combines its own values with the data received from its children, and sends the aggregate to its parent. TinyDB performs reordering on the query predicate to optimize the query process. They also propose other ways of optimizing query execution plan for sensor database. If there are no failures, this technique works extremely well for decomposable aggregates, namely distributive and algebraic aggregates such as MIN, MAX, COUNT and AVG. A collection of motes all running TinyOS/TinyDB can be easily deployed in an environment and then effectively programmed. This is suitable for any environmental monitoring application. TinyDB works not only for sensing, but can be used for actuation by making use of its triggering capabilities.

3.2 TinyDB

TinyDB is an application that runs over TinyOS on the Berkeley Mica mote platform. This combination creates a network of low powered nodes, which communicate in an adaptive, multi-hop, ad-hoc wireless network. It provides a SQL-like interface for querying the network for data without the need to load application specific code into each node. TinyDB takes care of issues to do with minimizing radio communication through the use of in-network processing and aggregation, putting the CPU into a low-power mode whilst not communicating or processing, as well as optimizing queries for power efficiency in the network. TinyDB is a query processing architecture for TinyOS sensor networks. It is an SQL like interface and the functionality to respond to the queries distributed in the network. It has the ability to efficiently use network resources and to perform limited operations on the data values - either with in network or at the interface. The major aspects of this framework are that queries are processed with a query optimizer that generates a query that is efficient on the particular network, reducing the use of network resources. The query optimizer determines how data flows between nodes and establishes aggregation filters that allow an optimization between computation and communication. In addition TinyML and TinyVM can be used. TinyML is a lightweight implementation following some of the SensorML ideas that are built on XML. The basic platform components relate to physical devices (such as the Mica2). There are a few fundamental elements for TinyML. The first is the concept of a Platform. A platform consists of a basic infrastructure with some type of processor, an energy source and a radio or other communication device. In addition to the infrastructure, platforms have sensors and/or actuators. Basic sensors are such things as microphones for sound detection and thermistors for temperature readings. A sensor field, a collection of sensor nodes is made up of a collection of platforms. The platforms may be uniform or different in their capabilities and attributes. A sensor network also has elements that could provide data to link the field to an external reference points. An example is a sensor field that has platforms with self organizing location data and a field reference description that has a link to GPS data at one or more points. Consider for example a relation *sensors* containing readings of temperature and light from N sensors located in different geographical locations described by coordinates
(i, j), in M sets and defined as:

$$sensor1 = \sigma location ij(sensors)$$
$$\ldots$$
$$sensorM = \sigma location ij(sensors)$$

This can be transformed into a selection operation as illustrated in the following (TinyDB) SQL statement:

```
SELECT nodeij, light, temperature
FROM sensors
WHERE location = “(1,7)”
SAMPLE PERIOD is for 30s
```

This statement collects light and temperature readings from location (1,7).

3.3 Grid Computing

A sensor network is a distributed sensing technology with limited processing ability and communications. Grid computing is similar to a web service but with more control due to the extra services it provides. Both grid computing and sensor networks are ongoing areas of research with collaboration from around the world. Grid computing looks at utilizing the vast amount of resources that are spread across networks, while sensor networks utilize the collections of simple devices to cover large areas at a low cost. While sensor networks use the grid resources, the grid uses the data collection by the sensor networks. Different sensor networks format the data being sent in different ways requiring integrating schema. Schema is created in this project for defining the format of the data, and for the XML parser to be used. The Globus toolkit is used to develop the grid side of the system, and the sensor network is simulated using simulators.

4 Integration Frameworks

Given the view of a sensor network as a huge distributed database system, we would like to adapt existing techniques from distributed and heterogeneous database systems for a sensor network environment. However, there are major differences between sensor networks and traditional distributed and heterogeneous database systems. Because of the large scale and dynamic nature of a sensor network, we cannot assume that a centralized optimizer maintains global knowledge and thus precise meta-information about the whole network.

We have developed a prototype to interface grid computing with a sensor network. Our prototype performs all the coding and the addressing that are required between the two systems, thus allowing the system to work with many types of sensors. The grid - sensors framework we developed features the following modules: Inter-process communication: the processes communicate and synchronize each other, resource discovery; the resources needed to execute the program are automatically discovered at run time, remote commanding, callbacks, are used to implement application monitoring, and authentication. All intermediate data are stored in a database using database. We wanted our database to be free to implement so we wanted a free-to-use database manager ant that is why our database is implemented in mySQL. We are using declarative querie*s* are the preferred way of interacting with a sensor network. Rather than deploying application-specific procedural code expressed in programming language, we believe that sensor network applications are naturally data-driven, and thus we can abstract the functionality of a large class of applications into a common interface of expressive queries. It also helped us identify a set of challenging issues that we are addressing with our ongoing research. Due to the large scale of a sensor network, it is highly probable that some of the sensors and some of the communication links will fail at some point during the processing of a long-running query. We are currently studying how sensor database systems can adjust to communication failures and return a more accurate answer at the cost of increased response time and resource usage.

5 Concluding Remarks

Sensor networks are becoming ubiquitous, and the database community needs to be prepared to address the new challenging problems. In this paper we introduce a framework for integrating sensors with grid computing. We described a vision of processing queries over sensor networks, and discussed some initial steps in system implementation. We have described an architecture based on the Globus Toolkit to integrate devices and sensors into the Grid.

There are still many improvements that need to be made to the system. A few of these issues are discussed below. Results from the TinyDB query to the sensor network are currently sent back to the backend and then stored directly in a database which can then be analyzed separately. This does have its benefits, but detracts from the overall TinyDB application. We will need some kind of ODBC driver for TinyDB. With an ODBC driver the sensor network can be made to appear as a standard table. We will also focus on querying wide area sensor databases, containing (XML) data derived from sensors. The basic idea is to use XPATH queries on XML database.

References

[1] Cohen, P.R., and Feigenbaum, E.A., editors (1982) The Handbook of Artificial Intelligence - Vol. 3 AddisonWesley Publishing Co: Reading, MA, U.S.A.

[2] Duntsch, I., and Gediga, G. (1998) "Uncertainty measures of rough set prediction", Artificial Intelligence, no. 106, pp. 109-137.

[3] Efron, B. (1982) The Jackknife, the Bootstrap and Other Resampling Plans. CBMS-NSF Regional Conference Series in Applied Mathematics, no. 38. Society for Industrial and Applied Mathematics: Philadelphia, PA, U.S.A.

[4] Fayyad, U.M., Piatetsky-Shapiro, G., and Smyth, P. (1996) "The KDD process for extracting useful knowledge from volumes of data", Communications of the ACM, v. 39, no. 11, pp. 27-34.

[5] Frawley, W.J., Piatetsky-Shapiro, G., and Matheus, C. (1992) "Knowledge discovery in databases: An overview." AI Magazine, pp. 57-70.

[6] The Globus toolkit, www.globus.org/toolkit

[7] R. Motwani and P. Raghavan. Randomized Algorithms. Cambridge University Press, 1995.

[8] Serge Abiteboul, et al, The Lowell Database Research Self-Assessment, COMMUNICATIONS OF THE ACM, May 2005/Vol. 48, No. 5, pp 111-115

[9] R. Wasniowski, Interfacing of sensor networks to grid computing, report 2004.

[10] Y. Yao and J. Gehrke. The cougar approach to in-network query processing in sensor networks. SIGMOD Record 3(3), 2002.