INTRODUCTION

Sensors are often used to monitor the status of an environment continuously. The sensor readings are reported to the application for making decisions and answering user queries. For example, a fire alarm system in a building employs temperature sensors to detect any abrupt change in temperature. An aircraft is equipped with sensors to track wind speed, and radars are used to report the aircraft’s location to a military application. These applications usually include a database or server to which the sensor readings are sent. Limited network bandwidth and battery power imply that it is often not practical for the server to record the exact status of an entity it monitors at every time instant. In particular, if the value of an entity (e.g., temperature, location) monitored is constantly evolving, the recorded data value may differ from the actual value. Querying the database can then produce incorrect results. Consider a simple example where a user asks the database: “Which room has a temperature between 10°F and 20°F?” If we represent temperature values of rooms A and B stored in the database by $A_0$ and $B_0$ respectively, we can see from Figure 1(a) that the answer to the user query is “Room B”. In reality, the temperature values of both rooms may have changed to newer values, $A_1$ and $B_1$, as shown in Figure 1(b), where the true query answer should be “Room A”. Unfortunately, because of transmission delay, these newest pieces of information are not propagated in time to the system to supply fresh data to the query, and consequently the query is unable to yield a correct answer.

In general, the incorrectness of query results is due to sensor uncertainty, an inherent property of any sensor database where each recorded data item is only an older, approximate version of the corresponding entity monitored. Apparently, providing meaningful answers in face of sensor uncertainty appears to be a futile exercise. In many situations, however, the values of sensors cannot change drastically in a short period of time – the degree and/or rate of change of a sensor value is constrained. This helps us to alleviate the problem. In the previous example, if we can guarantee that the actual temperatures of room A and B are no more than some deviations from $A_0$ and $B_0$ respectively, as in Figure 1(c), then we can state with confidence that room A does not satisfy the query.

Whether room B satisfies the query is less obvious. In Figure 1(c), it is not clear whether B has a temperature between 10°F and 20°F. However, the fact that the uncertainty associated with B’s temperature is bounded makes it possible to decide the degree of likelihood that B satisfies this query. In general, bounded uncertainty allows us to augment different levels of confidence with each answer (e.g., as a probability), instead of providing a definite answer. Queries that augment answers with probability values, based on uncertain information, are known as probabilistic queries. In the previous example, a probabilistic query produces answers such as: “Room A has a probability of 0 being between 10°F and 20°F, while Room B has a probability of 0.7”. Contrast with a query that gives incorrect answers because of stale data, a probabilistic query provides more confidence in the answers since uncertainty is considered.

Depending on the nature of the probabilistic query, confidence in a probabilistic query answer can be expressed in different forms. We will study different classes of probabilistic queries, which have different...
forms of answers and evaluation techniques. We also examine the concept of “quality” of a probabilistic query result which is related to the ambiguity of the result.

The rest of this article is organized as follows. We first present the background and related works. Then we examine in detail how sensor data uncertainty can be modeled. Based on the data models, we present a classification of probabilistic queries. For each query class, we examine factors that determine quality of probabilistic query answers. Finally, we outline future research issues.

BACKGROUND

Approximate answers to queries based on a subset of data have been well studied. Vrbsky and Liu (1994) studied approximate answers for set-valued queries (where a query answer contains a set of objects) and single-valued queries (where a query answer contains a single value). An exact answer \( E \) can be approximated by two sets: a certain set \( C \) which is the subset of \( E \), and a possible set \( P \) such that \( C \cup P \) is a superset of \( E \). Other techniques like precomputation (Poosala & Ganti, 1999), sampling (Gibbons & Matias, 1998), and synopses (Acharya, Gibbons, Poosala & Ramaswamy, 1999) are used to produce statistical results. While these efforts investigate approximate answers based on a subset of (exact) values of the data, this article addresses imprecise answers that assume all (uncertain) values of the data.

There are a number of works about evaluation of intervals. Olston et al. discuss the problem of balancing the trade-off between precision and performance for querying replicated data (Olston, Loo & Widom, 2001; Olston & Widom, 2000, 2002). In their model, the cache in the server cannot keep track of the exact values of sensor sources due to limited network bandwidth. Instead of storing the actual value in the server’s cache, an interval for each item is stored, within which the current value must be located. A query is then answered by using these intervals, together with the actual values fetched from the sources. The problem of minimizing the update cost within an error bound specified by aggregate queries is studied.


SENSOR UNCERTAINTY MODELS

Uncertainty of sensor values can be represented in three forms, from no attention to uncertainty at all, to the highest resolution of uncertainty information (Cheng & Prabhakar, 2003). For notational convenience, let us assume that a real-valued attribute \( a \) of a set of database objects \( T \) is queried. We name the \( i \)th object of \( T \) as \( T_i \), and the value of \( a \) for \( T_i \) as \( T_i.a \) (where \( i = 1, \ldots, |T| \)).

1. **Point Uncertainty:** This is the simplest model, where we assume there is no uncertainty associated with data at all. Each data item, \( T_i.a \), is supposed to be a correct representation of the external entity being monitored. Queries use these exact values to evaluate results. Although manipulating real values is relatively easy for a query, the example in Figure 1 illustrates how such data can lead to incorrect query results.

2. **Interval Uncertainty:** Instead of representing the exact sensor value in the database, an uncertain interval, denoted by \( U(t) \), is stored. Specifically, \( U(t) \) is a close interval \([l(t),u(t)]\), where \( l(t) \) and \( u(t) \) are real-valued functions of \( t \), bounding the value of \( T_i.a \) at time \( t \). This model represents imprecision of data in the form of an interval. An example model of \( U(t) \) is an interval bounding all values within a distance of \((t-t_{update}) \times r \) of \( T_i.a \), where \( t_{update} \) is the time that \( T_i.a \) is last updated, and \( r \) is the current rate of change of \( T_i.a \). Thus, \( U(t) \) expands linearly with time until the next update of \( T_i.a \) is received. Another realization of this model can be found in Wolfson et al. (1999).

3. **Probabilistic Uncertainty:** This model is proposed by Cheng et al. (2003). Compared with interval uncertainty, it requires one more piece of information—the probability density function (pdf) of \( T_i.a \) within \( U(t) \). We call this function an uncertain pdf of \( T_i.a \) at time \( t \), denoted by \( f(x,t) \). Notice that \( \int_{U(t)} f(x,t)dx = 1 \) and \( f(x,t) \) equals 0 if \( x \notin U(t) \).

Further, the exact form of \( f(x,t) \) is application-dependent. For example, in modeling sensor measurement uncertainty, where each \( U_i \) is an error range containing the mean value, \( f(x,t) \) can be a normal distribution around the mean value. Another example is the modeling of one-dimensional moving objects, where at any point in time, the actual location is within a certain bound, \( d \), of its last reported location value. If the actual location changes further than \( d \), then the sensor reports its new location value to the database and possibly changes \( d \). Then \( U(t) \) contains all the values within
Related Content

Towards Google Earth: A History of Earth Geography
[www.igi-global.com/chapter/towards-google-earth/7923?camid=4v1a](www.igi-global.com/chapter/towards-google-earth/7923?camid=4v1a)

Semantic Enrichment of Geographical Databases
[www.igi-global.com/chapter/semantic-enrichment-geographical-databases/11209?camid=4v1a](www.igi-global.com/chapter/semantic-enrichment-geographical-databases/11209?camid=4v1a)

Discovering Quality Knowledge from Relational Databases
M. Mehdi Owrang O. (2009). *Selected Readings on Database Technologies and Applications* (pp. 95-111).
[www.igi-global.com/chapter/discovering-quality-knowledge-relational-databases/28548?camid=4v1a](www.igi-global.com/chapter/discovering-quality-knowledge-relational-databases/28548?camid=4v1a)

A New Approach to Secure Federated Information Bases Using Agent Technology
[www.igi-global.com/article/new-approach-secure-federated-information/3290?camid=4v1a](www.igi-global.com/article/new-approach-secure-federated-information/3290?camid=4v1a)