Managing Uncertainty of Large Spatial Databases

Reynold Cheng
Department of Computer Science, the University of Hong Kong, Hong Kong, China
Email: ckcheng@cs.hku.hk

Abstract

Spatial data are prevalent in location-based services (LBS), sensor networks, and RFID monitoring systems. Data readings collected in these applications are often imprecise. The uncertainty in the data can arise from multiple sources, including measurement errors due to the sensing instrument and discrete sampling of the measurements. It is often important to record the imprecision and also to take it into account when processing the spatial data. The challenges of handling the uncertainty in spatial data includes modeling, semantics, query operators and types, efficient execution, and user interfaces. Probabilistic models have been proposed for handling the uncertainty. In this paper, we examine the modeling and querying issues of this kind of databases.

1 Introduction

Data uncertainty is an inherent property in applications that deal with spatial data. In the Global-Positioning System (GPS), the location collected from the GPS-enabled devices (e.g., PDAs) often has measurement and sampling error [14, 8]. The location data transmitted to the system may further encounter some network delay. Hence, the data collected in these applications are often imprecise, inaccurate, and stale. Similar problems also occur in sensor networks and RFID monitoring systems. Consider a habitat monitoring system used in scientific applications, where data such as temperature, humidity, and wind speed are acquired from a sensor network. Due to physical imperfection of the sensor hardware, the data obtained are often inaccurate [7]. Moreover, a sensor cannot report its value at every point in time, and so the system can only obtain data samples at discrete time instants. Recent works also propose to inject uncertainty to a user's location for location privacy protection [2]. Services or queries that base their decisions on these data can produce erroneous results. There is thus a need to manage these data errors more carefully.

In this paper, we investigate how to manage uncertainty in large spatial databases. Particularly, we examine probabilistic models for uncertain spatial databases. We discuss *probabilistic spatial queries* (or PSQ), which consider the models of the spatial data uncertainty (instead of just the data value reported), and augment probabilistic guarantees to the query results. For example, a PSQ asking who is the nearest neighbor of a given point q can tell the user that John is the answer, with a probability of 0.8. The probabilities reflect the degree of correctness of query results, thereby facilitating the system to produce a more confident decision.

The rest of this paper is arranged as follows. In Section 2, we describe spatial uncertainty models commonly used in the research community. We discuss algorithms for processing an important probabilistic query on uncertain spatial models in Section 3. We conclude in Section 4.

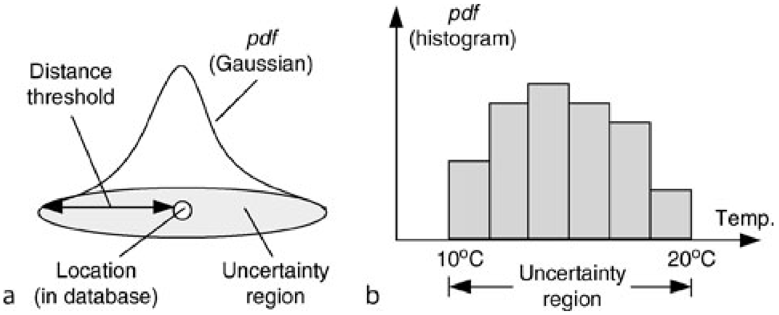

Figure 1: Spatial uncertainty models.

2 Spatial Uncertainty Models

Uncertainty in spatial data is often the result of either inherent limitations in the accuracy with which the sensed data is acquired or limitations imposed by concerns such as efficiency and battery life. Consider for example, a moving object application that uses GPS devices to determine the locations of people as they move about. Although GPS accuracy has improved significantly, it is well known that the location reported by a GPS sensor is really an approximation – in fact, the actual location is likely to be distributed with a Gaussian probability distribution around the reported location. This is an example of uncertainty due to the limitation of the measurement instrument.

Since most location sensors are powered by batteries that can be quickly depleted, most applications that rely on these sensors take great pains to conserve battery power. A common optimization is to not measure and transmit readings continuously. Instead, the data are sampled at some reasonable rate. In this case the exact values are only known at the time instances when samples are taken. Between samples, the application can only estimate (based on the earlier samples) the values. Data uncertainty can happen even when location readings are precise and frequently sampled. For example, if a given sensor is suspected of being faulty or compromised, the application may only partially trust the data provided by the sensor. In these cases, the data are not completely ignored but their reliability can be reduced. In these cases, the unreliability of the raw or processed sensor data can be captured as uncertain data. Each of these examples shows that sensor readings are not precise.

In [10], piecewise linear functions are used to approximate the cdf of an uncertain item. Sometimes, point samples are derived from an item's pdf [9, 12]. In the *existential uncertainty model*, every object is represented by the value in the space, as well as the probability that this object exists [6].

Let us now discuss a commonly-used model of spatial data uncertainty. This model assumes that the actual data value is located within a closed region, called the *uncertainty region*. In this area, a non-zero probability density function (*pdf*) of the value is defined, where the integration of pdf inside the region is equal to one. The cumulative density function (*cdf*) of the item is also provided. In an LBS, a normalized Gaussian pdf is used to model the measurement error of a location stored in a database [14, 8] (Fig. 1). The uncertainty region is a circular area, with a radius called the "distance threshold"; the newest location is reported to the system when it deviates from the old one by more than this threshold (Fig. 1). The figure also shows the histogram of temperature values in a geographical area observed in a week. The pdf, represented as a histogram, is an arbitrary distribution between 10 and 20^{o}C.

3 Probabilistic Query Algorithms

A logical formulation of queries for the above uncertainty model, called *probabilistic queries* (or PSQ), has been recently studied in [11, 13]. In [4], Cheng et al. proposed a classification scheme for different types of

PSQs. In that scheme, a PSQ is classified according to the forms of answers. An *entity-based query* is one that returns a set of objects (e.g., list of objects that satisfy a range query or join conditions), whereas a *value-based query* returns a single numeric value (e.g., value of a particular sensor). Another criterion is based on whether an *aggregate* operator is used to produce results. An aggregate query is one where there is interplay between objects that determines the results (e.g., a nearest-neighbor query). Based on these two criteria, four different types of probabilistic queries are defined. Each query type has its own methods for computing answer probabilities. In [4], the notion of *quality* has also been defined for each query type, which provides a metric for measuring the ambiguity of an answer to the PSQ.

Next, we study the probabilistic nearest-neighbor query, which is an important PSQ.

3.1 Probabilistic Nearest-Neighbor Queries

A common PSQ for uncertain spatial data is the probabilistic nearest-neighbor queries (or PNNQ). This query returns the non-zero probability of each object for being the nearest neighbor of a given point q [4]. A PNNQ can be used in an LBS, where enquires such as: “Please show me the nearest restaurant” can be asked. It can also be used in a sensor network, where sensors collect the temperature values in a natural habitat. For data analysis and clustering purposes, a PNNQ can find out the district(s) whose temperature values is (are) the closest to a given centroid. Another example is to find the IDs of sensor(s) that yield the minimum or maximum wind-speed from a given set of sensors [7, 4]. A minimum (maximum) query is essentially a special case of PNNQ, since it can be characterized as a PNNQ by setting q to a value of $-\infty$ (∞).

Evaluating a PNNQ is not trivial. In particular, since the exact value of a spatial data item is not known, one needs to consider the item’s possible values in its uncertainty region. Moreover, since the PNNQ is an entity-based aggregate query [4], an item’s probability depends not just on its own value, but also on the relative values of other objects. If the uncertainty regions of the objects overlap, then their pdfs must be considered in order to derive their corresponding probabilities.

To evaluate PNNQ, one method is to derive the pdf and cdf of each item’s distance from q. The probability of an item for satisfying the PNNQ is then computed by integrating over a function of distance pdfs and cdfs [7, 4, 5]. In [5], an R-tree-based solution for PNNQ was presented. The main idea is to prune items with zero probabilities, using the fact that these items’ uncertainty regions must not overlap with that of an item whose maximum distance from q is the minimum in the database. The *probabilistic verifiers*, proposed in [3], are algorithms for efficiently computing the lower and upper bounds of each object’s probability for satisfying a PNNQ. These algorithms, when used together with the probability threshold defined by the user, avoid the exact probability values to be calculated. In this way, a PNNQ can be evaluated more efficiently.

3.2 Answering PNNQ with the UV-Diagram

Next, we discuss a recent PNNQ evaluation algorithm [15, 16, 17]. This solution is based on the Voronoi Diagram [18], which is primarily designed for evaluating nearest-neighbor queries over two-dimensional spatial points. Conceptually, the Voronoi diagram partitions the data space into disjoint “Voronoi cells”, so that all points in the same Voronoi cell have the same nearest neighbor. The task of finding the nearest neighbor of a query point is then reduced to a point query. Figure 2(a) illustrates a Voronoi diagram of seven points. Since the query point q is located in the Voronoi cell of O_2, O_2 is the nearest neighbor of q.

In [15, 16], the idea of extending the Voronoi diagram to support PNN execution has been explored. The authors propose the *Uncertain-Voronoi* diagram (or *UV-diagram*), where the nearest-neighbor information of every point in the data space is recorded, based on the uncertain objects involved. Figure 2(b) illustrates an example UV-diagram for seven uncertain objects, where the space is divided into disjoint regions called *UV-partitions*. Each UV-partition P is associated with a set S of one or more objects. For any point q located inside P, S is the set of answer objects of q (i.e., each object in S has a non-zero probability for being the nearest

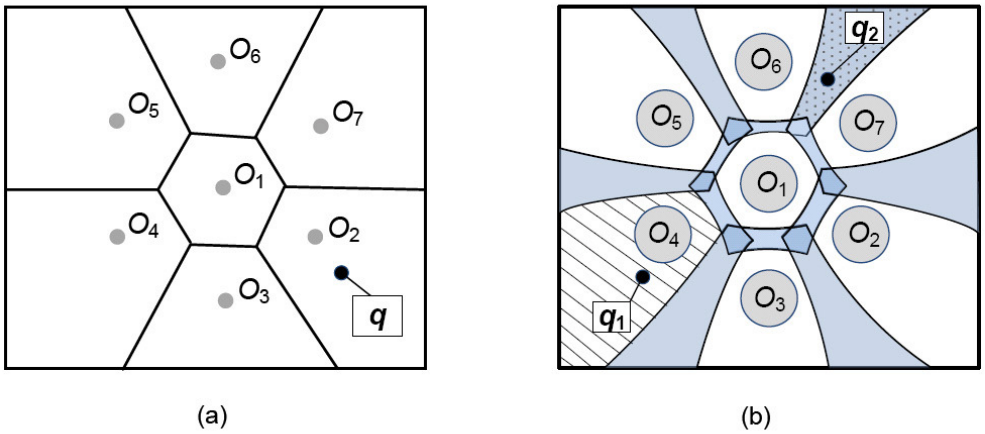

Figure 2: (a) Voronoi Diagram. (b) UV-Diagram.

neighbor of q). The highlighted regions contain points that have two or more nearest neighbor objects. As an example, since q_1 is inside the dashed region, O_4 has a non-zero probability for being the nearest neighbor of q_1; on the other hand, q_2 is located inside the dotted region, and O_6 and O_7 are the answer objects for the PNN with q_2 as the query point. Observe that the Voronoi diagram, which indexes on spatial points, is a special case of the UV-diagram, since a point can be viewed as an uncertainty region with a zero radius. Figure 2 compares the two diagrams.

Developing a UV-diagram is not simple. Notice that the UV-partitions are produced based on uncertainty regions, which may not be points. Unfortunately, efficient computational geometry methods for generating the Voronoi diagram (e.g., line-sweeping [19]) cannot be readily used for creating a UV-diagram, since these methods are primarily designed for spatial points, rather than uncertainty regions. In particular, a UV-partition can be irregular in shape, and contains different answer objects. In general, given a set of uncertain regions, an exponential number of UV-partitions can be created, and the number of edges of each UV-partition can also be exponentially large [15, 16]. This makes it computationally infeasible to generate and store these partitions. It is also difficult to find out which of these irregular UV-partitions contain a given query point. In [15, 16], a scalable method for constructing a UV-diagram has been developed.

Instead of computing UV-partitions, the authors in [15, 16] have developed a PNNQ solution for two-dimensional uncertain spatial data, The main idea is to interpret the UV-diagram in a different manner. Specifically, for every object O_i, they consider the extent a_i such that O_i can be the nearest neighbor of any point selected from a_i. They call this extent the *UV-cell* of O_i. They examine some basic properties of a UV-cell (e.g., its size and number of edges). They show how to represent a UV-cell as a set of objects, and develop novel methods to find this object set efficiently. For example, their *batch-construction* algorithm allows the UV-cells of objects that are physically close to each other to be swiftly obtained. They further propose a polynomial-time method for constructing an index for the UV-partitions, called the UV-index. They adopt an adaptive-grid indexing scheme, which has the advantage of adapting to different distributions of uncertain objects' positions. Their experimental results show that on both synthetic and real dataset, this index can be constructed in a much shorter time. The same authors in [17] have recently extended the UV-index to support multi-dimensional uncertain data. They have also examined how to update the UV-index to quickly, in order to reflect the insertion (deletion) of uncertain objects to (from) the spatial database.

4 Conclusions

Data uncertainty is found in virtually all applications that acquire spatial data. In some situations, it may be acceptable to ignore the uncertainty and treat a given value as a reasonable approximation of the reading. For others (e.g., road traffic monitoring and wireless sensor networks), such approximations and the resulting errors in query answers are unacceptable. In order to provide correct answers for these applications it is necessary to handle the uncertainty in spatial data. Recent works that propose to inject uncertainty to a user's location for location privacy protection also requires the use of a PSQ [2]. An system prototype, called Orion [20], has also been developed to handle uncertain spatial data.

4.1 Future Directions

Much work remains to be done in the area of uncertain spatial data processing. It will be interesting to study the development of data mining algorithms for spatial uncertainty. Another direction is to study probabilistic spatio-temporal queries over historical spatial data (e.g., trajectories of moving objects). Developing scalable algorithms in distributed computing environments (e.g., MapReduce or Spark) to support PSQ could be important. It would be crucial to implement these solutions in existing probabilistic database systems. Other works include revisiting query cost estimation, query plan evaluation, as well as designing user interfaces that convenient input and visualisation of uncertain data. A long term goal is to consolidate these research ideas and develop a distributed spatio-temporal database system that provides uncertainty management facilities.

References

[1] Böhm C., Pryakhin A., and Schubert M. The Gauss-Tree: Efficient object identification in databases of probabilistic feature vectors. In Proc. 22nd Int. Conf. on Data Engineering, 2006.

[2] Chen J. and Cheng R. Efficient evaluation of imprecise location-dependent queries. In Proc. 23rd Int. Conf. on Data Engineering, 2007.

[3] Cheng R., Chen J., Mokbel M., and Chow C. Probabilistic verifiers: Evaluating constrained nearest-neighbor queries over uncertain data. In Proc. 24th Int. Conf. on Data Engineering, 2008.

[4] Cheng R., Kalashnikov D., and Prabhakar S. Evaluating probabilistic queries over imprecise data. In Proc. ACM SIGMOD Int. Conf. on Management of Data, 2003, pp. 551–562.

[5] Cheng R., Kalashnikov D.V., and Prabhakar S. Querying imprecise data in moving object environments. IEEE Trans. Knowl. and Data Eng., 16(9), 2004.

[6] Dai X., Yiu M. L., Mamoulis N., Tao Y., and Vaitis M. Probabilistic spatial queries on existentially uncertain data. In Proc. 9th Int. Symp. Advances in Spatial and Temporal Databases, 2005, pp. 400–417.

[7] Deshpande A., Guestrin C., Madden S., Hellerstein J., and Hong W. Model-driven data acquisition in sensor networks. In Proc. 30th Int. Conf. on Very Large Data Bases, 2004.

[8] Pfoser D. and Jensen C. Capturing the uncertainty of moving-objects representations. In Proc. 11th Int. Conf. on Scientific and Statistical Database Management, 1999.

[9] Kriegel H., Kunath P., and Renz M. Probabilistic nearest-neighbor query on uncertain objects. In Proc. 12th Int. Conf. on Database Systems for Advanced Applications, 2007, pp. 337–348.

[10] Ljosa V. and Singh A. APLA: Indexing arbitrary probability distributions. In Proc. 23rd Int. Conf. on Data Engineering, 2007, pp. 946–955.

[11] Parker A., Subrahmanian V., and Grant J. A logical formulation of probabilistic spatial databases. IEEE Trans. Knowl. and Data Eng., 19(11), 2007.

[12] Pei J., Jiang B., Lin X., and Yuan Y. Probabilistic skylines on uncertain data. In Proc. 33rd Int. Conf. on Very Large Data Bases, 2007.

[13] Singh S., Mayfield C., Shah R., Prabhakar S., Hambrusch S., Neville J., and Cheng R. Database support for probabilistic attributes and tuples. In Proc. 24th Int. Conf. on Data Engineering, 2008.

[14] Sistla P.A., Wolfson O., Chamberlain S., and Dao S. Querying the uncertain position of moving objects. In Temporal Databases: Research and Practice. Springer Verlag, 1998.

[15] Cheng R., Xie X., Yiu M. L., Chen J., and Sun L.. UV-diagram: A Voronoi Diagram for Uncertain Data. In the IEEE Intl. Conf. on Data Engineering (IEEE ICDE 2010), Long Beach, USA, Mar, 2010.

[16] Xie X., Cheng R., Yiu M. L., Sun L., and Chen J. UV-Diagram: A Voronoi Diagram for Uncertain Spatial Databases. In the Very Large Databases Journal (VLDBJ), 22(3), pp. 319-344, June 2013.

[17] Zhang P., Cheng R., Mamoulis N., Renz M., Zuefle A., Tang Y., and Emrich T. Voronoi-based Nearest Neighbor Search for Multi-Dimensional Uncertain Databases. In Intl. Conf. on Data Engineering (IEEE ICDE 2013), Brisbane, Apr 2013.

[18] Okabe A., Boots B., Sugihara K., and Chiu S. Spatial Tessellations: Concepts and Applications of Voronoi Diagrams. Wiley, second edition, 2000.

[19] de Berg M., van Kreveld M., Overmars M., and Schwarzkopf O.. Computational Geometry: Algorithms and Applications. Springer-Verlag, 1997.

[20] The Orion Uncertain Database Management System. Available at: `http://orion.cs.purdue.edu/`