CHAPTER 8

Probabilistic Counting in Uncertain Spatial Databases Using Generating Functions

Andreas Züfle

8.1 Introduction

Our ability to unearth valuable knowledge from large sets of spatial data is often impaired by the uncertainty of the data, which geography has named the “the Achilles heel of GIS” [Goodchild 1998]. The uncertainty is caused by several reasons: (1) imprecision caused by physical limitations of sensing devices and connection errors; (2) data records may be obsolete; (3) data can be obtained from unreliable sources, such as volunteered geographic information; and (4) data may be deliberately obfuscated to preserve the privacy of users. These issues introduce the notion of uncertainty in the context of spatiotemporal data management. Many algorithms have been proposed in the last decade to handle different spatial query predicates (such as distance range, $k$NN, and distance ranking) described in various tutorials [Renz et al. 2010, Cheng et al. 2014, Züfle et al. 2017, 2020] and surveys [Aggarwal and Philip 2009, Züfle 2021]. Many query predicates require counting the number of uncertain objects that satisfy a given query predicate, such as being located in a query region or being closer to a query object than another object. Computing the probability mass function of such a count requires computing for each integer $n$ the probability of having exactly $n$ objects satisfy the query predicate. A commonly used technique that allows many of these algorithms to run efficiently
leverages the technique of generating functions to efficiently aggregate an exponential number of possible worlds in polynomial time. This technique is described, along with examples and implementation, in this spatial gem.

An example of an uncertain (toy) database shown in Figure 8.1 has six uncertain objects \{A, B, C, D, E, F\}. Rather than having a single unique (crisp) location, an object in an uncertain database may have multiple alternatives, each associated with a corresponding probability of being the true location. For example, object B has five possible locations and object D has four possible locations. There exist multiple models for uncertain data, either describing uncertain objects by discrete (and finite) sets of alternatives or by describing uncertain objects by continuous distributions of (uncountably infinite) possible locations [Züfle 2021]. The most prominent systems for uncertain relational data management are MayBMS [Antova et al. 2008], MystiQ [Boulos et al. 2005], Trio [Agrawal et al. 2006], and BayesStore [Wang et al. 2008], which allow efficiently answering traditional queries that select subsets of data based on predicates or join different datasets based on conditions. While these existing systems efficiently support simple projection–selection–join queries, they offer no support for complex queries and data mining tasks. A likely reason for this gap is the theoretic result of Dalvi and Suciu [2007] that shows that the general problem of query processing in uncertain databases is \#P-hard in the number of database objects.

While this result implies that general query processing on uncertain data is hard, it does not rule out the possibility of efficient solutions for specific query types. And in fact, many important classes of spatial queries have efficient (polynomial time) solutions including range count queries [Follmann et al. 2011], nearest

![Figure 8.1](image)

**Figure 8.1** Example of an uncertain ε-range query. Object A is a true hit; objects B, C, and D are possible hits.
neighbor queries [Cheng et al. 2004, 2008, Iijima and Ishikawa 2009], k-nearest neighbor queries [Ijosa and Singh 2007, Beskales et al. 2008, Cheng et al. 2009] and, (similarity-) ranking queries [Hua et al. 2008, Yi et al. 2008, Cormode et al. 2009, Li et al. 2009, Soliman and Ilyas 2009, Li and Deshpande 2010].

All these classes of spatial queries have in common is that they count the number of spatial objects that fall within a region. A range count query directly returns the distribution of the number of objects within a specified query region. To decide if an object $A$ is a $k$NN of a query object $Q$, a $k$NN query computes the probability that less than $k$ objects are closer to $Q$ than $A$, thus counting a number of objects within a distance of less than the distance between $Q$ and $A$; and for a distance ranking query, the probability that an object $A$ has the $k$-th nearest objects of $Q$ is the probability that exactly $k - 1$ objects (other than $A$) have a distance to $Q$ less than the distance between $Q$ and $A$.

**Example 8.1** As an example of counting the number of uncertain objects within a region, reconsider Figure 8.1, and assume a query that counts the number of uncertain objects within a distance of $\epsilon$ from a query object $q$. We first note that this query answer is a random variable, which depends on the locations of the uncertain objects (which are random variables, too). We observe that objects $E$ and $F$ are guaranteed to be outside the range, such that we can prune them from our computation. We also note that object $A$ is guaranteed to be in the range, allowing us to increment the query result by one without having to further consider this object. For objects $B$, $C$, and $D$, the events of being located inside the query range are random variables with probabilities of 0.3, 0.2, and 0.9, respectively. Thus, the result of this query is a random variable having a sample space of $\{1, 2, 3, 4\}$, and mapping each of these possible results to their probability. For example, the probability of having exactly one object in the range is $0.7 \cdot 0.8 \cdot 0.1 = 0.056$. For the probability that exactly two objects are inside the range, we can add the probabilities on the three possible worlds where exactly one object out of $\{B, C, D\}$ is inside the range.

In the general case of computing the probability that exactly $k$ out of $n$ uncertain objects are inside the query range, we need to aggregate the probabilities of $\binom{n}{k}$ combinations of objects to be inside the query range. Straightforward approaches require to enumerate all the $\binom{n}{k}$ possible worlds the number of which is in $O(n^k)$.

Yet, for this problem of counting the (distribution of the) number of uncertain objects within a query range two efficient solutions based on (1) the Poisson–binomial recurrence [Hua et al. 2008, Yi et al. 2008, Bernecker et al. 2010] and (2) based on generating functions [Li et al. 2009] have been proposed independently in the literature. These solutions allow aggregating the probabilities of an exponential number of possible combinations of objects in polynomial time, allowing us to
answer many important spatial query types efficiently. This spatial gem describes
how the generating function technique, which was first presented in the context of
distance ranking by Li, Saha, and Deshpande in the best paper of VLDB 2009 [Li
et al. 2009], can be used to efficiently answer spatial queries on uncertain data.

## Generating Functions for Probabilistic Counting

Let \( \mathcal{X} = \{X_1, ..., X_N\} \) denote the set of objects having a nonzero probability of being
located in the query region, and let \( p_i \) denote the probability of object \( X_i \) to be
located inside the query region. We can model each \( X_i \) as a Bernoulli distributed
random variable that has a probability of \( p_i \) being 1 and a probability of
\( 1 - p_i \) being 0. With this model, the count of objects inside the query region is
the sum \( \sum_{i=1}^{N} b(p_i) \). We note that since the probabilities \( p_i \) are not identical, this
random variable does not follow a binomial distribution but is instead known as a
Poisson–binomial distribution having parameters \( \{p_1, ..., p_N\} \) [Hua et al. 2008].

Our goal is to evaluate this random variable \( \sum_{i=1}^{N} b(p_i) \) efficiently, that is, for each
\( 0 \leq k \leq N \) we want to derive the probability \( P(\sum_{i=1}^{N} b(p_i) = k) \).

For this purpose, represent each random variable \( X_i \) by a polynomial \( \text{poly}(X_i) = p_i \cdot x + (1 - p_i) \). Consider the generating function
\[
\mathcal{F}^N = \prod_{i=1}^{N} \text{poly}(X_i) = \sum_{i=0}^{N} c_i x^i.
\] (8.1)

The coefficient \( c_i \) of \( x^i \) in the expansion of \( \mathcal{F}^N \) equals the probability \( P(\sum_{n=1}^{N} X_n = i) \)
[Li et al. 2009]. For example, the monomial \( 0.25 \cdot x^4 \) implies that with a probability of
0.25 the sum of all Bernoulli random variables equals four.

The expansion of \( N \) polynomials each containing two monomials leads to a total
of \( 2^N \) monomials, one monomial for each sequence of successful and unsuccessful
Bernoulli trials, that is, one monomial for each possible world. To reduce this
complexity, an iterative computation of \( \mathcal{F}^N \) can be used by exploiting that
\[
\mathcal{F}^k = \mathcal{F}^{k-1} \cdot \text{poly}(X_k). \] (8.2)

This rewriting of Equation (8.1) allows inductively computing \( \mathcal{F}^k \) from \( \mathcal{F}^{k-1} \). The
induction is started by computing the polynomial \( \mathcal{F}^0 \), which is the empty product
that equals 1, the neutral element of multiplication, that is, \( \mathcal{F}^0 = 1 \). To under-
stand the semantics of this polynomial, the polynomial \( \mathcal{F}^0 = 1 \) can be rewritten
as \( \mathcal{F}^0 = 1 \cdot x^0 \), which we can interpret as the following tautology: “with a proba-
bility of one, the sum of all zero Bernoulli trials equals zero.” After each iteration,
we can unify monomials having the same exponent, leading to a total of at most
$k + 1$ monomials after each iteration. This unification step allows the removal of the combinatorial aspect of the problem since any monomial $x^i$ corresponds to a class of equivalent worlds, such that this class contains only and all of the worlds where the sum $\sum_{k=1}^{N} X_k = 1$. In each iteration, the number of these classes is at most $k$, and the probability of each class is given by the coefficient of $x^i$.

**Example 8.2** As an example, consider again the running example of Figure 8.1. For each object, we first obtain the probability of being located inside the query region (which can be done in linear time using a range query and aggregating the probabilities of instances inside the query region). For the six objects $A, B, C, D, E,$ and $F$, we obtain probabilities of being inside the query region of $1.0, p_1 := 0.3, p_2 := 0.2, p_3 := 0.9, 0,$ and $0$, respectively. We can safely prune objects $E$ and $F$ since they cannot affect the query result. We can also prune object $A$ by increasing the result by 1 since we know $A$ must be inside the query range. Given the probabilities $p_1 = 0.3, p_2 = 0.2$, and $p_3 = 0.9$, we obtain the three generating polynomials $poly(X_1) = (0.3x + 0.7)$, $poly(X_2) = (0.2x + 0.8)$, and $poly(X_3) = (0.9x + 0.1)$. We trivially obtain $F^0 = 1$. Using Equation (8.2), we get

$$F^1 = F^0 \cdot poly(X_1) = 1 \cdot (0.3x + 0.7) = 0.3x + 0.7.$$ 

Semantically, this polynomial implies that out of the first one Bernoulli trials, the probability of having a sum of one is 0.3 (according to monomial $0.3x = 0.3x^1$), and the probability of having a sum of zero is 0.7 (according to monomial $0.7 = 0.7x^0$).

Next, we compute $F^2$, again using Equation (8.2):

$$F^2 = F^1 \cdot poly(X_2) = (0.3x^1 + 0.7^0) \cdot (0.2x^1 + 0.8^0) =$$

$$0.06x^1x^1 + 0.24x^1x^0 + 0.14x^0x^1 + 0.56x^0x^0.$$ 

In this expansion, the monomials have deliberately not been unified to give an intuition of how the generating function technique is able to identify and unify equivalent worlds. In the above expansion, there is one monomial for each possible world. For example, the monomial $0.14x^0x^1$ represents the world where the first trial was unsuccessful (represented by the 0 in the first exponent) and the second trial was successful (represented by the 1 in the second exponent). The above notation allows the identification of the sequence of successful and unsuccessful Bernoulli trials, clearly leading to a total of $2^k$ possible worlds for $F^k$. However, we know that we only need to compute the total number of successful trials; we do not need to know the sequence of successful trials. Thus, we may treat worlds that have
the same number of successful Bernoulli trials equivalently to avoid the enumeration of an exponential number of sequences. This is done implicitly by polynomial multiplication, exploiting that

\[ 0.06x^1 x^1 + 0.24x^1 x^0 + 0.14x^0 x^1 + 0.56x^0 x^0 = 0.06x^2 + 0.24x^1 + 0.14x^1 + 0.56x^0. \]

This representation no longer allows us to distinguish the sequence of successful Bernoulli trials. This loss of information is beneficial as it allows the unification of possible worlds having the same sum of Bernoulli trials

\[ 0.06x^2 + 0.24x^1 + 0.14x^1 + 0.56x^0 = 0.06x^2 + 0.38x^1 + 0.56x^0. \]

The remaining monomials represent an equivalence class of possible worlds. For example, monomial \(0.38x^1\) represents all worlds having a total of one successful Bernoulli trial out of the first two trials. This is evident since the coefficient of this monomial was derived from the sum of both worlds having a total of one successful Bernoulli trial. In the next iteration, we compute:

\[ F^3 = F^2 \cdot poly(X_3) = (0.06x^2 + 0.38x^1 + 0.56x^0) \cdot (0.9x + 0.1) \]

\[ = 0.054x^2 x^1 + 0.006x^2 x^0 + 0.342x^1 x^1 + 0.038x^1 x^0 + 0.504x^0 x^1 + 0.056x^0 x^0. \]

This polynomial represents the three classes of possible worlds in \(F^2\) combined with the two possible results of the third Bernoulli trial, yielding a total of 32 monomials. Unification yields

\[ 0.054x^2 x^1 + 0.006x^2 x^0 + 0.342x^1 x^1 + 0.038x^1 x^0 + 0.504x^0 x^1 + 0.056x^0 x^0 = \]

\[ 0.054x^3 + 0.348x^2 + 0.542x^1 + 0.056x^0. \]

This polynomial describes the PDF of \(\sum_{i=1}^{3} X_i\) (having \(X_1 = B, X_2 = C, X_3 = D\)) since each monomial \(c_i x^i\) implies that the probability, that out of all three Bernoulli trials the total number of successful events equals \(i\), is \(c_i\). Thus, we get \(P(\sum_{i=1}^{3} X_i = 0) = 0.0056, P(\sum_{i=1}^{3} X_i = 1) = 0.542, P(\sum_{i=1}^{3} X_i = 2) = 0.348,\) and \(P(\sum_{i=1}^{3} X_i = 3) = 0.054.\)

### 8.3 Complexity Analysis

The generating function technique requires a total of \(N\) iterations (as in the worst case all uncertain objects have a nonzero non-one probability of being in the query region). In each iteration \(1 \leq k \leq N\), a polynomial of degree \(k - 1\), and thus of
maximum length $k$, is multiplied with a polynomial of degree 1, thus having a length of 2. This requires computing a total of $(k + 1) \cdot 2$ monomials in each iteration, each requiring a scalar multiplication. This leads to a total time complexity of $\sum_{i=1}^{N} 2k + 2 \in O(N^2)$ for the polynomial expansions. Unification of a polynomial of length $k$ can be done in $O(k)$ time, exploiting that the polynomials are sorted by the exponent after expansion. Unification at each iteration leads to a $O(n^2)$ complexity for the unification step. This results in a total complexity of $O(n^2)$, similar to the Poisson–binomial recurrence approach.

Many spatial query predicates do not require computing the full probability mass of the distribution of the number of objects within the query range but only require the probability of having less or equal than a specified parameter $K$ of objects in the query range. For example, to find the probability that an object is among the $K$-nearest neighbors of a query object, it is sufficient to compute the probability that at most $K − 1$ objects are closer (within a shorter range). In this case, all monomials having an exponent greater or equal to $K$ can be pruned from the computation. In this case, the length of the expanded polynomial in each iteration is bounded by $K$, thus yielding a run-time complexity of $O(k \cdot n)$. This efficient computation can also be leveraged for the case of distance ranking, where the challenge is to find the probability that exactly $K − 1$ other objects are closer to a query object for an object to be exactly the $K$-th nearest neighbor (i.e., having a distance rank of $K$). This task requires only finding the coefficient $c_{K-1}$ of the expanded monomial $c_{K-1}x^{K-1}$ having an exponent of $K − 1$. Since in each iteration of multiplying and expanding monomials the coefficient $c_{K-1}$ only depends on the coefficients $c_{K-2}$ and $c_{K-1}$ of previous iterations, we may also discard monomials with an exponent of $K$ or greater to answer distance ranking queries.

To summarize, using generating functions we can compute the distribution of the number of objects within a query range in $O(n^2)$, where $n$ is the number of database objects. In cases such as KNN or distance ranking queries where we only need to know the probability of having at most or exactly $K$ objects within the query range, we can reduce this complexity to $O(K \cdot n)$ by truncating intermediate polynomials.

### 8.4 Implementation

A Python implementation can be found in the following GitHub repository [https://github.com/azufle/generating_functions](https://github.com/azufle/generating_functions). This implementation, both as a Jupyter notebook and a classic .py script, defines a function that efficiently computes the probability mass function of a probabilistic count given a list of probabilities. Additional documentation can be found in the repository.
8.5 Variants, Extensions, and Improvements

This section surveys a variety of extensions and improvements of the classic generating functions.

8.5.1 Acceleration Using Discrete Fourier Transform

An advantage of the generating function approach is that this naive polynomial multiplication can be accelerated using Discrete Fourier Transform (DFT). This technique allows the reduction of the total complexity of computing the sum of $N$ Bernoulli random variables to $O(N\log^2 N)$ [Li et al. 2011]. This acceleration is achieved by exploiting that DFT allows expanding two polynomials of size $k$ in $O(k\log k)$ time. Equi-sized polynomials are obtained in the approach of Li et al. [2011] by using a divide and conquer approach that iteratively divides the set of $N$ Bernoulli trials into two equi-sized sets. Their recursive algorithm then combines these results by performing a polynomial multiplication of the generating polynomials of each set. More details of this algorithm can be found in Li et al. [2011].

8.5.2 Extension to Uncertain Counts

In many applications, a probabilistic event may not only have two possible outcomes (Yes/No, Success/Failure), but may have a third outcome that represents an unknown/undecided/uncertain state. For example, given incomplete trajectories of objects and their resulting uncertainty regions at a time (for example, described by a bounding box of possible locations), there may be possible worlds where (1) an object is within the query region, (2) an object is outside the query region, and (3) containment of the object within the query region cannot be decided due to uncertainty.

For such cases, the addition of an unknown state has been proposed to be represented by generating function in Bernecker et al. [2011]. For each object $X_i \in X = \{X_1, ..., X_N\}$ having a probability of $p_i$ to satisfy the query condition (such as being located within the query region), a probability of $\overline{p}_i$ to not satisfy the query condition and a probability of $1 - p_i - \overline{p}_i$ being in an unknown state, we can consider the generating function:

$$F^N = \prod_{i=1}^{N} p_i \cdot x + (1 - p_i - \overline{p}_i) \cdot y + \overline{p}_i.$$

Intuitively, the anonymous variable $x$ denotes an event of satisfying the query condition, and the anonymous variable $y$ denotes the event of an undecided satisfaction of the query condition. In the expanded polynomial, a monomial such as $c_{i,j} x^i \overline{y}^j$ corresponds to a possible world having a probability of $c_{i,j}$ having $i$ objects
guaranteed to satisfy the query condition and $j$ additional objects possibly satisfying the query condition. For example, a monomial $0.13x^3y^2$ corresponds to a possible world having a probability of 0.13 and having at least 3 but no more than $3 + 2 = 5$ objects satisfy the query predicate.

### 8.5.3 Dynamic Polynomials

In many applications, the probability of an object being inside a query range may change dynamically. For example, mobile objects may send update location information to change their uncertainty region. In this case, the probability distribution of the number of objects inside a query range changes as well. To update the probability distribution, we may recompute from scratch, using all objects having nonzero probability of being inside the query range. However, such an approach may be inefficient when there is a large number of such objects having frequent updates. To update the probability distribution of a probabilistic count, we can use polynomial division as described in Hubig et al. [2012]. Thus, having a database $\mathcal{X} = \{X_1, ..., X_N\}$ of objects each with a probability of $p_i$ to be inside the query and given the polynomial $\mathcal{F}^N$ that describes the probability distribution of the number of objects inside the query range, assume that an object $X_j$ changes its probability from $p_j$ to $p_j'$. We can update the $\mathcal{F}^N$ by removing the old effect of the old probability $p_i$ through polynomial division of polynomial $\text{poly}(X_j) = p_i \cdot x + 1 - p_i$ and by including the effect of the new probability through multiplication with polynomial $\text{poly}(X_j') = p_i' \cdot x + 1 - p_i'$. Combining both steps, we obtain the updated polynomial $\mathcal{F}^{N'}$:

$$\mathcal{F}^{N'} = \mathcal{F}^N \frac{\text{poly}(X_j')}{{\text{poly}(X_j)}} = \mathcal{F}^N \frac{p_i' \cdot x + 1 - p_i'}{p_i \cdot x + 1 - p_i}.$$

### References

C. C. Aggarwal and S. Y. Philip. 2009. A survey of uncertain data algorithms and applications. IEEE Trans. Knowl. Data Eng. 21, 5, 609–623. DOI: https://doi.org/10.1109/TKDE.2008.190.

P. Agrawal, O. Benjelloun, A. Das Sarma, C. Hayworth, S. Nabar, T. Sugihara, and J. Widom. 2006. Trio: A system for data, uncertainty, and lineage. In Proceedings of the 32nd International Conference on Very Large Data Bases (VLDB 2006), demonstration description. VLDB Endowment, 1151–1154.

L. Antova, T. Jansen, C. Koch, and D. Olteanu. 2008. Fast and simple relational processing of uncertain data. In 2008 IEEE 24th International Conference on Data Engineering. IEEE, 983–992. DOI: https://doi.org/10.1109/ICDE.2008.4497507.

T. Bernecker, H.-P. Kriegel, N. Mamoulis, M. Renz, and A. Zuefle. 2010. Scalable probabilistic similarity ranking in uncertain databases. IEEE Trans. Knowl. Data Eng. 22, 9, 1234–1246. DOI: https://doi.org/10.1109/TKDE.2010.78.
T. Bernecker, T. Emrich, H.-P. Kriegel, N. Mamoulis, M. Renz, and A. Züfle. 2011. A novel probabilistic pruning approach to speed up similarity queries in uncertain databases. In 2011 IEEE 27th International Conference on Data Engineering. IEEE, 339–350. DOI: https://doi.org/10.1109/ICDE.2011.5767908.

G. Beskales, M. A. Soliman, and I. F. Ilyas. 2008. Efficient search for the top-k probable nearest neighbors in uncertain databases. Proc. VLDB Endow. 1, 1, 326–339. DOI: https://doi.org/10.14778/1453856.1453895.

J. Boulos, N. Dalvi, B. Mandhani, S. Mathur, C. Re, and D. Suciu. 2005. MYSTIQ: A system for finding more answers by using probabilities. In Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data (SIGMOD '05). ACM, 891–893. DOI: https://doi.org/10.1145/1066157.1066277.

R. Cheng, D. V. Kalashnikov, and S. Prabhakar. 2004. Querying imprecise data in moving object environments. IEEE Trans. Knowl. Data Eng. 16, 9, 1112–1127. DOI: https://doi.org/10.1109/TKDE.2004.46.

R. Cheng, J. Chen, M. F. Mokbel, and C.-Y. Chow. 2008. Probabilistic verifiers: Evaluating constrained nearest-neighbor queries over uncertain data. In 2008 IEEE 24th International Conference on Data Engineering. IEEE, 973–982. DOI: https://doi.org/10.1109/ICDE.2008.4497506.

R. Cheng, L. Chen, J. Chen, and X. Xie. 2009. Evaluating probability threshold k-nearest-neighbor queries over uncertain data. In Proceedings of the 12th International Conference on Extending Database Technology: Advances in Database Technology (EDBT ’09). ACM, 672–683. DOI: https://doi.org/10.1145/1516360.1516438.

R. Cheng, T. Emrich, H.-P. Kriegel, N. Mamoulis, M. Renz, G. Trajcevski, and A. Züfle. 2014. Managing uncertainty in spatial and spatio-temporal data. In 2014 IEEE 30th International Conference on Data Engineering. IEEE, 1302–1305. DOI: https://doi.org/10.1109/ICDE.2014.6816766.

G. Cormode, F. Li, and K. Yi. 2009. Semantics of ranking queries for probabilistic data and expected ranks. In 2009 IEEE 25th International Conference on Data Engineering. IEEE, 305–316. DOI: https://doi.org/10.1109/ICDE.2009.75.

N. Dalvi and D. Suciu. 2007. Efficient query evaluation on probabilistic databases. VLDB J. 16, 4, 523–544. DOI: https://doi.org/10.1007/s00778-006-0004-3.

A. Follmann, M. A. Nascimento, A. Züfle, M. Renz, P. Kröger, and H.-P. Kriegel. 2011. Continuous probabilistic count queries in wireless sensor networks. In International Symposium on Spatial and Temporal Databases, Vol. 6849: Lecture Notes in Computer Science. Springer, 279–296. DOI: https://doi.org/10.1007/978-3-642-22922-0_17.

M. F. Goodchild. 1998. Uncertainty: The Achilles heel of GIS. Geo. Info. Systems 8, 11, 50–52.

M. Hua, J. Pei, W. Zhang, and X. Lin. 2008. Ranking queries on uncertain data: A probabilistic threshold approach. In Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data (SIGMOD ’08). ACM, 673–686. DOI: https://doi.org/10.1145/1376616.1376685.

N. Hubig, A. Züfle, T. Emrich, M. A. Nascimento, M. Renz, and H.-P. Kriegel. 2012. Continuous probabilistic sum queries in wireless sensor networks with ranges.
References

In *Proceedings of the International Conference on Scientific and Statistical Database Management (SSDBMˈ12)*, Vol. 7338: Lecture Notes in Computer Science. Springer, 96–105. DOI: https://doi.org/10.1007/978-3-642-31235-9_6.

Y. Iijima and Y. Ishikawa. 2009. Finding probabilistic nearest neighbors for query objects with imprecise locations. In *2009 Tenth International Conference on Mobile Data Management: Systems, Services and Middleware*. IEEE, 52–61. DOI: https://doi.org/10.1109/MDM.2009.16.

J. Li and A. Deshpande. 2010. Ranking continuous probabilistic datasets. *Proc. VLDB Endow.* 3, 1–2, 638–649. DOI: https://doi.org/10.14778/1920841.1920923.

J. Li, B. Saha, and A. Deshpande. 2009. A unified approach to ranking in probabilistic databases. *Proc. VLDB Endow.* 2, 1, 502–513. DOI: https://doi.org/10.14778/1687627.1687685.

J. Li, B. Saha, and A. Deshpande. 2011. A unified approach to ranking in probabilistic databases. *VLDBJ.* 20, 2, 249–275. DOI: https://doi.org/10.1007/s00778-011-0220-3.

V. Ljosa and A. K. Singh. 2007. APLA: Indexing arbitrary probability distributions. In *IEEE 23rd International Conference on Data Engineering*. IEEE, 946–955. DOI: https://doi.org/10.1109/ICDE.2007.367940.

M. Renz, R. Cheng, and H.-P. Kriegel. 2010. Similarity search and mining in uncertain databases. *Proc. VLDB Endow.* 3, 1–2, 1653–1654. DOI: https://doi.org/10.14778/14778.1920841.1921066.

M. A. Soliman and I. F. Ilyas. 2009. Ranking with uncertain scores. In *2009 IEEE 25th International Conference on Data Engineering*. IEEE, 317–328. DOI: https://doi.org/10.1109/ICDE.2009.102.

D. Z. Wang, E. Michelakis, M. Garofalakis, and J. M Hellerstein. 2008. BayesStore: Managing large, uncertain data repositories with probabilistic graphical models. *Proc. VLDB Endow.* 1, 1, 340–351. DOI: https://doi.org/10.14778/1453856.1453896.

K. Yi, F. Li, G. Kollios, and D. Srivastava. 2008. Efficient processing of top-k queries in uncertain databases with x-relations. *IEEE Trans. Knowl. Data Eng.* 20, 12, 1669–1682. DOI: https://doi.org/10.1109/TKDE.2008.90.

A. Züfle. 2021. Uncertain spatial data management: An overview. In *Handbook of Big Geospatial Data*. Springer, 355–397. DOI: https://doi.org/10.1007/978-3-030-55462-0_14.

A. Züfle, G. Trajecvski, D. Pfoser, M. Renz, M. T. Rice, T. Leslie, P. Delamater, and T. Emrich. 2017. Handling uncertainty in geo-spatial data. In *Proceedings of the 2017 IEEE 33rd International Conference on Data Engineering (ICDE)*. IEEE, 1467–1470. DOI: https://doi.org/10.1109/ICDE.2017.212.

A. Züfle, G. Trajecvski, D. Pfoser, and J.-S. Kim. 2020. Managing uncertainty in evolving geo-spatial data. In *Proceedings of the 2020 21st IEEE International Conference on Mobile Data Management (MDM)*. IEEE, 5–8. DOI: https://doi.org/10.1109/MDM48529.2020.00021.