DOI: 10.21917/ijsc.2016.0167

INDEXING AND QUERY PROCESSING TECHNIQUES IN SPATIO-TEMPORAL DATA

A. John[1], M. Sugumaran[2] and R.S. Rajesh[3]
[1,3]*Department of Computer Science and Engineering, Manonmaniam Sundaranar University, India*
E-mail: johnmtech@gmail.com[1], rs_rajesh1@yahoo.co.in[3]
[2]*Department of Computer Science and Engineering, Pondicherry Engineering College, India*
E-mail: sugu@pec.edu[2]

Abstract

Indexing and query processing is an emerging research field in spatio - temporal data. Most of the real-time applications such as location based services, fleet management, traffic prediction and radio frequency identification and sensor networks are based on spatio-temporal indexing and query processing. All the indexing and query processing applications is any one of the forms, such as spatio index access and supporting queries or spatio-temporal indexing method and support query or temporal dimension, while in spatial data it is considered as the second priority. In this paper, give the survey of the various uncertain indexing and query processing techniques. Most of the existing survey works on spatio-temporal are based on indexing methods and query processing, but presented separately. Both the indexing and querying are related, hence state - of - art of both the indexing and query processing techniques are considered together. This paper gives the details of spatio-temporal data classification, various types of indexing methods, query processing, application areas and research direction of spatio-temporal indexing and query processing.

Keywords:

Uncertain Data, Spatio-Temporal Index, Spatio-Temporal Queries, Skyline Query, Top-k Query, Nearest-Neighbor Query

1. INTRODUCTION

Traditional database systems are not well equipped to handle moving object databases (MOD), location information, enabling technologies because data is assumed to be constant and moving object databases are continuously changing the data [1]. The moving object databases belong to the area of the spatio-temporal databases, which derive from spatial databases and temporal databases. The difference is that moving objects databases focus on the continuous change in geographic space while spatial-temporal databases only support the discrete changing of spatial information. Considerable research has been carried out regarding the two different aspects of moving objects data management, which includes modeling and tracking of location information, uncertainty management, spatio-temporal indexing and querying issues, and data mining including traffic and location prediction.

Nowadays query processing and indexing methods have an essential one for data retrieval in spatial - temporal network. Query processing is a collection of data arranged for ease and speed of search and retrieval of data. Indexing is a data structure used to retrieve data from database tables and improve the speed of data retrieval. The indexing methods and query processing is inter-related one, based on the indexing method different types of query processing types are supported to the particular indexing method. This paper gives a brief literature about various indexing methods and query processing techniques. There are many different query processing strategies to fulfill user requirements. According to the function of the query, various classifications are made. They are location based query, moving objects and updating query, range based query, trajectory queries for moving objects and uncertain past, present and future data detection query, optimization query etc. The query processing varies based on the requirement of the users. The Location-based Services (LBS) are the combination of location-aware devices (e.g., GPS-like devices) provide personalized services to user based on their current locations and it provide answers for user queries based on the requirement of the users. The location-based queries are either snapshot or continuous queries. Examples of snapshot queries include “Where is my nearest bus station” and “Which are the schools within one mile of my location”, while examples of continuous queries include continuously “report my nearest police station” and continuously report “the gas stations within one mile of my car” [2]-[7].

Moving object and updating query based on area of the Spatio-Temporal databases, which derive from spatial databases and temporal databases. Moving objects focused only continuous changes in geographical space while support the discrete change of spatial information. Two main considerations of moving object managements are location management and moving object management. The location management focused on represents, store the data and continuously change the location moving objects in database and predict the next position of moving objects. The spatio-temporal data stores the whole history of the moving objects so that query get answer in any time [8] – [11]. Range based query a source point *S* and a value *e* and the data set *D*, a range query retrieve all the objects of *S* are within the network distance *e* from *S*. The nearest neighbor query is closest to network distance (e.g find the ATM within one kilometer) the range based and nearest neighbor (NN) are related. The range based query range and nearest neighbor query are based Euclidean Restriction [12]. Trajectories queries user will work blindly and find the data. The aim of this query is find the moving data in the trajectories, as the same time density queries are to efficiently find dense areas with high attention of moving objects. For example a driver refer the trajectories based on earlier route who have driven to the city and analyzing the motion data measured by the sensors attached to the bodies of top sport players, pick the data from the more number of cases. Trajectories queries are applied in dynamic, real traffic analysis and fuzzy nature etc. the dimensions of the queries are two dimensional and three dimensional [13]-[17]. In uncertain trajectories data semantics are “sometimes”, “always”, “possibility” and “definitely”. Uncertain applied in fuzziness in managing moving objects and deals with spatio-temporal indeterminacies [17], [18]. The optimization query is focus on minimize the cost, minimize the number of service sites, minimize the time complexities and minimize the space complexities.

Indexing methods are used to increase the speed of the data retrieval. In the spatio-temporal area the data continuously increases over a time and moving object sent their positions. The main drawback of spatio-temporal processing is keeping all the updates are impossible. Two approaches [20] are used to minimize the updates or history data: (1) Sampling (2) Update on change only. The sampling data is taken for sampling at time interval and update on change only is moving object data is sent only the data is changed (e.g., change in speed or direction). Spatio-temporal indexing methods are categorized into three ways [20]-[22]. First method deal about spatial index access methods based on location, the second method is spatio-temporal index access methods based on location and time, and the third is indexing the temporal dimensions, while in this method spatial data is considered as the second priority based on time and then location is considered.

Most of the applications are based on second and third category and we give the brief information about spatio temporal supporting indexing method and queries processing. Some of the previous achieved indexing methods, drawbacks and further investigation are mentioned in the survey papers [20], [21], [23] and we continue some of the main spatio temporal index methods mentioned in the previous papers and some of the new indexing methods are included and in this paper combind both survey and query processing. Mainly focused on Spatio-Temporal past present and future data predictions indexing methods in this paper.

The rest of this paper proceeds as follows: Definition and representation of uncertain data is discussed in Section 2. Section 3 describes the indexing methods of uncertain data. Section 4 presents the various query processing Techniques in uncertain data in section 5 Research directions and application fields of spatio-temporal uncertain data and section 6 concludes the paper.

2. DEFINITION AND REPRESENTATION OF UNCERTAIN DATA

The term certain means data are given as exact constants and uncertainty means lack of certainty having less knowledge and impossible to retrieve the existing state, future outcome or not possible to get more than one output. The values for uncertain data are given by probability measures, most notably probability distribution functions (PDFs) [24]. Uncertainty is mainly used in different field such as general public, decision theory, information theory, probability theory, finance, economics, insurance etc and uncertain data are mainly used in prediction of future data or events to already made or to be unknown [25]. For example of uncertain data is prediction of further preceding road in three way junction shown in Fig.1. Uncertain data can be generated from many applications, such as scientific measurements, sensor networks, GPS, and mobile object tracking.

Uncertain data are classified into two types: exact data and inexact data [26]. Inexact data are represented in different ways and the uncertain data models are probabilistic data, fuzzy data, approximate data, incomplete data, interval data and imprecise data [27], etc. This paper gives the details about different inexact data indexing methods and query processing techniques.

Fig.1. Example of Uncertain data

Definition: The definition of inexact uncertain data relationship, an uncertain relation R is defined as set of possible instance, $I(R) = \{R_1, R_2, R_3, \ldots, R_n\}$ where R_i is a relation instance. A data model representing an uncertain relation R, way of representing the set of instances $I(R)$.

The taxonomy of uncertain data representation and possible indexing querying techniques [28] for uncertain data are shown in Fig.2.

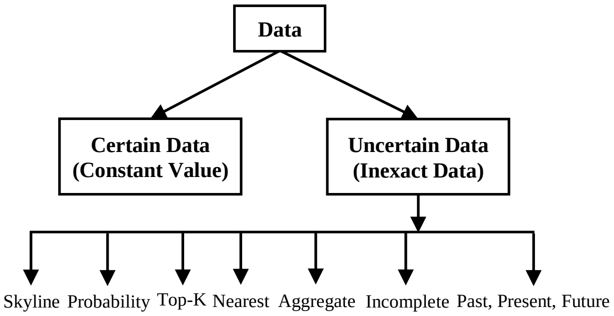

Fig.2. Taxonomy of uncertain data and Query Types

The main features of uncertain data types [27] are,

- Uncertain data types are more complex and diversities in term of query task.
- Based on application verities of data types are used in uncertain data. Uncertain data consists of structured data, semi-structured data and unstructured data. The recent days the query processing data are include graphical data, XML data, spatial data, fuzzy data etc.
- Based on the query types and usage the definition, storage and indexing, query processing and results are changed based on the application i.e. specialty of uncertain data.
- Based on the complexity difficult data models are used for uncertain data i.e. probability model, sliding Window model etc.

3. INDEXING METHODS OF UNCERTAIN DATA

Indexing is a data structure used to retrieve data from database table and improve the speed of data retrieval. Already we mentioned indexing methods are categorized into three types. Here we are going to focus about the second category manages both the spatial and temporal aspects into one structure. First going to focus some of the general spatio temporal indexing methods and spatiotemporal indexing methods for past, present and future data prediction and trajectories prediction indexing methods.

3.1 SPATIO TEMPORAL INDEXING METHODS FOR UNCERTAIN DATA

Traditionally, B^+ - Tree and R-Tree are used for indexing uncertain data, but both trees are cannot index effectively and directly. Some of the main indexing methods of uncertainty are External Interval Tree Index [29], TP Index Method [30], FUR-Tree [31], PTI Index [32], U-Grid [33], MODTN Model [11], B^x [34], U-Tree [35], LGU-Tree [36], MON-Tree [37], Gauss Tree [38], RUM-Tree [39], ST^2B-Tree [40], Segment based Index and interval based Index [41], Threshold interval Index [42] and Hbase Index [43].

3.1.1 External Interval Tree Index:

Interval trees are not specifically designed for handling uncertain data, but one-dimensional uncertain objects may be treated as intervals by using their PDF endpoints. Arge et al. [29] proposed two optimal external interval tree indexes. Both indices use a primary tree for layout and secondary structures to store the objects at each node, but one has a dynamic primary tree instead of a static one. However, the downfall of both interval indices is that if many uncertainty intervals overlap with the query interval's endpoints, then few objects are pruned from the search, and a lot of time is wasted in calculating probabilities.

3.1.2 TP Index Method:

Time parameterized Tree [TP] [30] is used in dynamic nature application and used to retrieve the actual results, current motions of the moving objects, change the expiration results. TP queries are important both as standalone methods, as well as building blocks of more complex operations. The general framework is covered time-parameterized variations of the most common spatial queries, namely window queries, k-nearest neighbors and spatial joins. In particular, each of these TP queries is reduced to nearest neighbor search where the distance functions are defined according to the query type. The TP methods can be applied with mobile queries, mobile objects or both queries suitable indexing method and this method is compared with R-Tree.

3.1.3 FUR-Tree:

Frequent updates R-Tree (FUR-Tree) is used in uncertain windows queries [31]. It is based on T-Tree but it follow Bottom up approach for insertion, deletion and updations. R-tree is the index choice for multi-dimensional data with low dimensionality. The R-tree follow top-down for update and each update one index is needed to locate and delete the data item. Depending on the amount of overlap among the bounding rectangles in the index nodes, this traversal may follow more than a single partial path from the root towards the leaf of the index. Top-down update is inherently inefficient because objects are stored in the leaf nodes and the observation that many applications exhibit locality reserving updates, the bottom-up concept proposed. The main aim of bottom-up approach is to improve the update performance and avoiding the expensive top down updates. The summary structure shows in the Fig.3. The main memory allows direct access to R-Tree index using bottom up approach. The main advantage of index method is insertion and deletions take place in underflow. The disadvantage is FUR-Tree not supports dynamic updates.

3.1.4 PTI Index:

Probability Threshold Indexing (PTI) [32] is used in probability uncertain model. A large number of items may overlap with [a; b], while in this fact only a small fraction of result contribute to the probability threshold queries

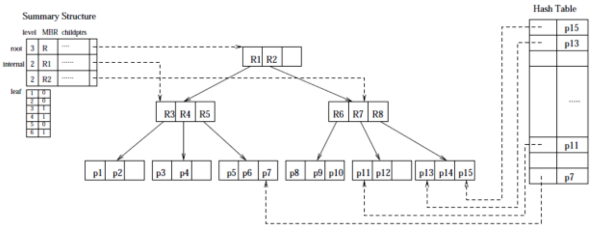

Fig.3. Structure of FUR Tree

This indexing goal is to redesign index structures so that probabilistic uncertainty information is fully utilized during an index search. This PTI structure, is based on the modification of a one-dimensional R-tree, where probability information is augmented to its internal nodes to facilitate pruning. Compared to the range query is performed on an R-tree, Starting from the root node, the query interval [a; b] is compared with the maximum bounding rectangle (MBR) of each child in the node. Only children with MBRs that overlap with [a; b] are further followed. Thus save the effort of retrieving nodes whose MBRs do not overlap [a; b]. The structure of the PTI index implementation is shown in the Fig.4. This framework is the same as R-tree, where each internal node consists of children MBRs and their corresponding pointers. In addition, each child Mj consists of a table, Mj:PT, that contains information of its x-bounds. Each entry of Mj:PT is a tuple of the form <left-x-bound, right-x-bound>. Further, a global table called TG, which contains the values of x for x-bounds.

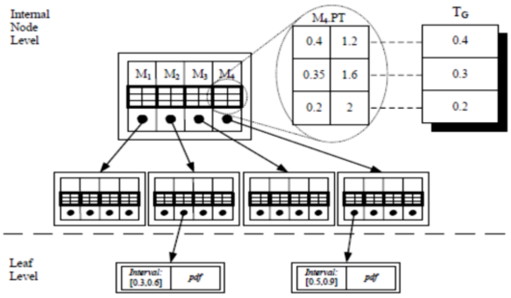

Fig.4. PTI index structure

3.1.5 U-Grid:

This index method is used in the moving object environment in uncertainty location [33]. The uncertainty in its location increases until the next update. This method supports probabilistic answers to nearest-neighbor queries is much more difficult than range queries. For range queries, the probability for each object can be determined independent of the other objects. It depends only upon the query and the uncertainty of the object in question. The uncertainty of an object can be characterized as an uncertainty region of an object, an uncertainty probability density function of an object, Probabilistic Nearest-Neighbor Query

(PNNQ) for a set of *n* objects. The solutions have the projection, pruning, bounding and evaluation phases. This method is applied into NN Queries and it is used in the moving object updating. The disadvantage is, it based on only probability.

3.1.6 MODTN:

Moving Object Dynamic Transport Network model (MODTN)[11] is used to express moving objects are modeled as moving graph points which predefined transportation to express general events of the system, such as traffic jams, temporary constructions, insertion and deletion of junctions or routes, the underlying transportation networks are modeled as dynamic graphs .it is dynamic transportation network having two steps, 1) the modeling of the underlying transportation networks on dynamic graph which allow us to express state changes such as traffic jams ,blockages caused by temporary constructions and topology changes. The basic idea is to associate a temporal attribute to every route or junction of the graph system so that the state of the route or junction at any time instant can be retrieved. The changes are discrete so it is represented in temporal data but the whole result is spatio-temporal history of the graph system can be stored and queried. 2) The second step of moving objects based on first step moving objects are modeled as moving graph points in MODTN. A moving graph point is a function from time to graph point, which can be represented as a group of moving units in the discrete model. To deal with more complicated situations where moving objects need to be modeled as moving graph lines or moving graph regions. This model defined in three ways,

Definition 1

A dynamic graph system, *GS,* is defined as a set of dynamic graphs and inter-graph junctions:

$$GS = \{G_1, G_2, \ldots, G_n, j^*_1, j^*_2, \ldots, j^*_m\}$$

where, $n \geq 1$, $m \geq 10$, G_i ($1 \leq i \leq n$) is a dynamic graph, and j^*_k ($1 \leq k \leq m$) is an inter-graph junction.

Definition 2

A dynamic graph, *G,* is defined as a pair: $G = (R, J)$ where *R* is a set of dynamic routes and *J* is a set of dynamic in-graph junctions.

Definition 3

A dynamic route of graph *G*, denote by *r*, is defined as follows:

$$r = (gid, rid, route, len, fdr, tp)$$

where, *gid* and *rid* are identifiers of *G* and *r* respectively, *route* is a polyline which describes the geometry of *r*, *len* is the length of the route, *fdr c*{0, 1, 2} is the traffic flow directions allowed in the route, and *tp* is the temporal attribute associated with *r*. MODTN model supports for insertion, deletion and updating operations. It is fully applied in dynamic environment and support location base quires and uncertainty queries.

3.1.7 B^x:

The continuous variables and the positions of mobile service users, termed moving objects is required large amount of data. The indexing of moving objects must support different queries and frequent updates. The existing index tree based on minimum bounding regions (MBRs) such as the R-tree exhibit high concurrency overheads during node splitting, and each individual update is known to be quite costly. So introduced new index is called *B^x* [34] and it is based on B+ tree. The B+ index manage moving objects locations as vectors that are time stamped based on their update time. The B+ tree that partitions values according to their timestamp and otherwise preserves spatial proximity and it supported range query, nearest query and continuous queries. The main operations are updating, insertion, deletion and the structure of the *B^x* shown in the Fig.5.

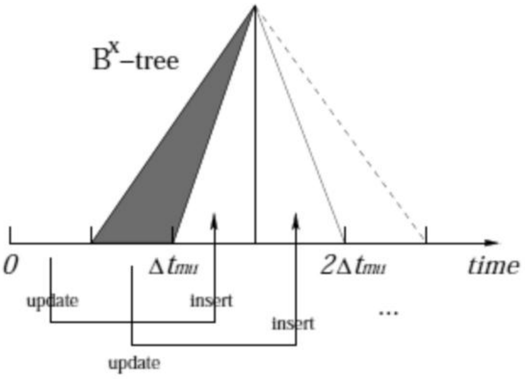

Fig.5. Structure of *B^x*

3.1.8 U-Tree:

U-Tree is used to retrieve multi-dimensional imprecise data on range based uncertain data and designed for optimized I/O and CPU time [35]. It is used in range based query and has dynamic updation. The structure of U- tree is designed for pruning subtrees that do not contain any results but required details are stored in leaf nodes. A leaf contains uncertainty region and together with a disk address and parameters are stored. Each insertion/deletion in a U-tree is performed in exactly the same way as the R*-tree, except that each metric is replaced with its summed counterpart. The main disadvantage of this index method is the I/O and CPU overhead take place.

3.1.9 LGU-Tree:

Lazy Group Update (LGU) algorithm for disk-based index structures for moving objects [36]. LGU contains two key additional structures to group “similar” updates so that they can be performed together: a disk-based insertion buffer (I-Buffer) for each internal node, and a memory-based deletion table (D-Table) for the entire tree. It supports for time interval queries, moving queries, windows queries. The operations are insertion, deletion and dynamic updations. The representations of LGU are in the Fig.6. LGU uses three key techniques to improve update throughput: It uses disk-based buffers (I-Buffers) to utilize memory wisely. I-Buffer is associated with each internal node; insertions are “pushed” from root to leaf in group through the I-Buffers. A global memory-based buffer (D-Table) is utilized to perform group deletion in the bottom up manner. It performs insertions and deletions in different ways: insertions are performed lazily in a top-down way while deletions are performed lazily bottom-up.

A hash lookup table similar to TPRK is used to enable direct access to the leaf level when performing.

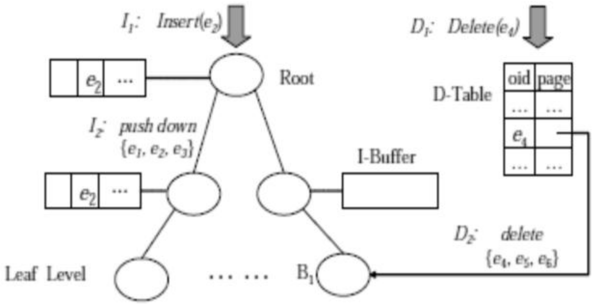

Fig.6. Structure of LGU

3.1.10 MON-Tree:

MON-Tree Index [37] method is used to query and store the position of the continuously moving objects. This moving objects indexing method deal with unconstrained 2-dimensional movement. The constrained moving objects care only important case of the object movements. So the Mon tree used in unconstrained moving objects and it deal about two network models. First one is edge oriented, i.e., the network is composed by edges and nodes and each edge has an associated polyline. The second one is more suitable for transportation networks and is route oriented, i.e., the network is composed by routes and junctions. A route has also an associated polyline as attribute. The MON-Tree showed good scalability when increasing the number of objects and time units in the index structure, and the query window and time interval in querying. This index method has polyline insertion, movement insertion and search insertion operations and it is compared with FNR tree.

3.1.11 Gauss-Tree:

This is novel index structure for efficient query processing for uncertain data using probability feature vector (PFV) [38]. The contributions of this index structure are: A model to handle uncertainty in databases that is based on the assumption that the uncertainty of feature vectors can be modeled by Gaussian distributions. Novel types of queries called k-most-likely identification queries (k-MLIQ) and threshold identification queries (TIQ). These queries are based on the probability that a query object and a data object describe the same object. The general solution is calculating the probabilities that are necessary to process the introduced queries. This method can be used in combination with several data structures and query algorithms. An index structure for efficiently processing k-MLIQs and TIQs called the Gauss-tree. The Gauss-tree belongs structurally to the R-tree family but uses novel algorithms for query processing, insertion and tree construction. This index method combined with PFV used to solve uncertain data. Gauss-tree parameter space to improve the pruning efficiency and leaf nodes store the objects with expectation and variance.

3.1.12 RUM-Tree:

RUM-tree stands for R-Tree with updates Memos [39]. This index method is used frequently updating multidimensional indexes location dependent application. It is minimizes the cost of updates. This index method updates in a memory based approaches so avoid disk accesses in updates processes. The main operations are insertion, updates, deletion and search. The RUM-tree supports to Garbage Cleaner to limit the number of obsolete entries in the tree and to limit the size of updates memos. The garbage cleaner deletes the obsolete entries lazily and in batches. Deleting lazily means that obsolete entries are not removed immediately; Deleting in batches means that multiple obsolete entries in the same leaf node are removed at the same time. The main structure of the RUM is shown in the Fig.7.

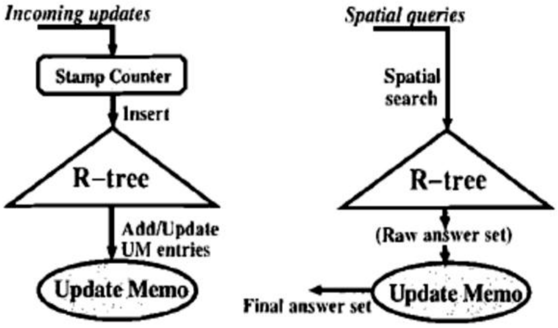

Fig.7. Structure of RUM

The main advantage of this index is reducing the cost for insertion. It is support location based frequent updates and range based queries.

3.1.13 ST^{2}B-Tree:

ST^{2}B-Tree [40] means a Self-Tunable Spatio-Temporal B+-Tree index for Moving object data. This tree rectifies the problem of MON tree which is the locations of objects change in space and time, the data distribution also changes and the answer for a same query over the same region may vary widely over time. As a result, traditional static indexes are not able to perform well and it is critical to develop self-tuning indexes that can be reconfigured automatically based on the state of the system. So the ST^{2}B-tree solve this problem. This method Turing is conducted automatically without human interventions and it performs the work of MON tree also.

3.1.14 Segment based Index and Interval based Index:

This index method is used in Range based queries in linear and nonlinear space [41]. The diagrammatic representation of segment based index are shown in the Fig.8.

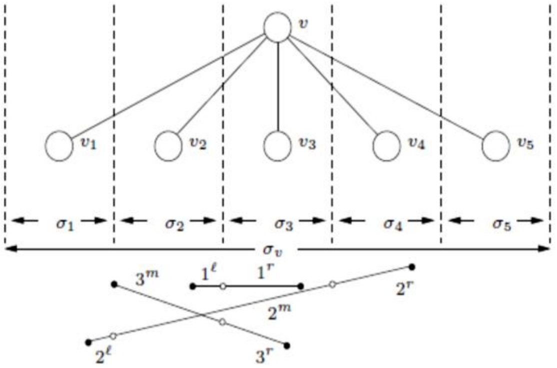

Fig.8. Segment and Interval based Index

The linear space and answers a query in $O\left(\sqrt{n}+k\right)$ time. The bootstrap is used to improve the query time to $O(\log n + k)$ while keeping the size linear. We fix a parameter r and construct a segment tree T of fan-out r – an r-ary tree that defines an r-way hierarchical decomposition of the plane into vertical slabs, each associated with a node of T. Let σ_v denote the slab corresponding to a node v. The slabs associated with the children of v are defined as follows. The partition σ_v into r vertical sub-slabs σ_1. …, σ_r, each containing roughly the same number of endpoints of segments in S. We create r children v_1. …, v_r of v and associate σ_i with v_i. A node v is a leaf if σ_v does not contain any endpoint of S in its interior.

Interval based index [41] describe a different index to store S, based on the interval tree, that uses $O(n\log n)$ space and answers a query in time $O(\log n + k)$. An interval tree T for S is built as follows. Let E be the set of endpoints of the segments of S. We first choose the median of E, and vertically divide the plane into two halves. We store this splitting line at the root of T and then build its two subtrees for the two halves recursively. Similar to the segment tree, the interval tree T also hierarchically partitions the plane into $O(n)$ canonical vertical slabs.

Hybrid Index [41] is the combination of segment based index and interval based index. It start with the interval-tree based index from the previous subsection. We stop the top down construction of the interval tree T as soon as there are $(\log^2 n)$ endpoints of S left in the slab, that is, the "atomic slab" σ_z for each leaf z of T contains $(\log^2 n)$ endpoints of S. For each internal node u, we define $S - u$ and $S + u$ as above, and let S^-, S^+ be as defined earlier. Since we have "fat" leaves, not all segments will be split those with both endpoints lying in the same atomic slab will not.

3.1.15 Threshold Interval Index (TII):

The Threshold Interval Index (TII) [42] is like a dynamic external interval tree with x-bounds borrowed from the probability threshold index. It has two advantages; first advantage is that the structure intrinsically and dynamically maintains balance all the time. The second is the interval-based structure makes all uncertain objects which fall entirely within the query interval easy to find and, therefore, possible to add to the results set without further calculation. The TII has a primary tree to manage interval endpoints. It also has secondary structures at internal nodes of the primary tree to store objects. When an object is added to the index, the endpoints of its uncertainty interval are added to the primary tree. Then, the object itself is added to the secondary structures of the appropriate tree node. The definitions of primary and secondary tree are,

1) Primary Tree: The primary tree is a weight-balanced B-tree with branching parameter $r > 4$ and leaf parameter $k > 0$. The weight of a node is the number of items (in this case, endpoints) below it. All leaves are on level 0. All endpoints are stored at the leaves, and internal nodes hold copied values of endpoints. The weight-balanced B-tree provides an effective way to dynamically manage intervals and spread.
2) Secondary Structures: Each internal node v represents an interval I_v, which spans all interval endpoints represented by children of v.

An uncertain object is stored at v if its uncertainty interval falls entirely within I_v but overlaps one or more boundaries of any child node's I_{vi}. Each object is stored at exactly one node in the tree. It is applied in the uncertain and range based quires.

3.1.16 HBase:

The present database are store only key-value databases are designed to be scalable, available and distributed, without much support for data organization including management of spatio-temporal data. HBase [43] is a novel hybrid index structure to organize data, combining a statistical based R-tree for indexing space and applying Hilbert curve for traversing approaching space. With key-value store, which insures effective querying response time and high insert rates, and generating target row key which take skewed data handing into account. The implementation of HBase is fully based on Range Query and k nearest neighbor queries. The HBase is support to NoSQL databases, which provides a simple fast access of data using a single key based on the key-value data model. It is developed a multidimensional mapping strategy for organizing moving objects on fixed networks.

3.2 UNCERTAIN PAST, PRESENT AND FUTURE DATA DETECTION QUERIES AND UNKNOWN DATA PREDICTION

In Real world data may be change various ways such as sequential data, time series, temporal, spatio- temporal, audio signal, video signal to name a few. The concept of data streams has gained a lot of practical interest in the field of spatio-temporal [44]. A data stream is an infinite sequence of data points defined usually either using time stamps or an index. So data may get change in time and change the position also. The old data is past and the present state of the data is present data and next what state of data is called future data. The past, present and future data finding purpose different indexing methods are introduced and that is mentioned in the part II of various indexing methods. Here we discuss various past, present and future detection and prediction of the data is summarizations are a mentioned here. Past, Present and Future (PPF) data is detected in fixed network and unfixed networks. Most of the works are done in the fixed networks. Another one the main work is unknown data prediction, the unknown data prediction also is called as future data prediction. The fixed networks data detection is linear representation and unknown data predictions are nonlinear data representation. The some of the main works in PPF data detection and unknown moving object data detection index methods are as follows PCFI+ Index [45], BBx [46], Rppf [47], VPMR [48], PPFI [49], aCBRB Tree [50], GPR Tree [51], FT Tree [52], MTree [53], HTPR* [54], MSMON-Tree [55], PPFN* [56].

3.3 PPF DATA DETECTION ON FIXED NETWORK

3.3.1 PCFI+-Index:

Past-Current-Future+-Index (PCFI+-Index) [45] is indexes the past, current & future information of the moving objects. The PCFI+-Index builds upon the PCFI-Index which was based on SETI-tree and TPR*-tree. The PCFI+-Index consists of two parts, in memory part with the name frontline, and disk based part. The whole region is partitioned into none-overlapping cells, and a spatial access method is used to index these cells. A set of main-memory TPR*-tree is used to index the moving objects that belong to the cells (one cell, one TPR*-tree). The large updation and current information is organized as has index file. The PCFI+-Index can handle most of the queries efficiently and provides a uniform solution for the trajectory, time-slice, internal and moving queries, and has a better performance than the SETI-Index, TPR*-Index, and PCFI Index is better performance than the SETI-Index, TPR*-Index, and PCFI Index.

3.3.2 BBx:

BBx Index Method [46] is used to find queries about the past, present and future position of moving objects. BBx index structure is used to find the position of moving objects in linear function of time. The index stores linear moving objects location on a forest of B+-tree. This index support queries based on temporal and spatial constraints such as queries are retrieved all the objects position within spatial range during set of time interval. BBx index inherit the ability of finding present and future position from the Bx-tree. Three factors are considered to the stage efficiency are Object position is represented as linear function. It is done by TPR-tree family (spatial representation), Object is based on indexing BBX, B+ -tree used for updation. The main supporting queries are Spatial Range based Location updation, Interval Queries and Predictive Queries.

3.3.3 R_{PPF}-Tree:

R_{PPF}-Tree is used to [47] obtain all the online position information from the moving e-service users. With the much slower advances in I/O speeds and many concurrent users, indexing techniques are feasible to store all the position in linear position. This work offers an R_{PPF}-tree technique capable of capturing the positions of moving objects at all points in time. The index substantially modifies partial persistence techniques, which support transaction time, to support valid time for monitoring applications. The performance of a time slice query is independent of the number of past position samples stored for an object.

3.3.4 VPMR Tree:

VPMR Tree [48] is focuses on the query problems of indexing structure for moving objects. The main focus of query processing is retrieval of approximate future movement information about moving objects that may be unavailable. First, a VPMR-tree is used for indexing the moving objects in road networks. Second, exponential smoothing method for predicting the result of queries that refer to the future is implemented. The experimental results have shown that VPMR-tree achieves a better performance for query evaluation than TPR-tree.

3.3.5 PPFI:

Past, Present and Future Index method [49] is used to predict the current and near future position of moving objects on fixed road network with an efficient update mechanism. The proposed work is PPFI and which consists of a 2DR* - tree, 1DR* -tree and hash table for updation. 2DR* tree is a static part built on managing the fixed network.1DR*-tree is a dynamic part built on indexing object movements. Hash table is used to update and predict near future position. This work is compared with FNR tree and STR-tree. This work is supports patio-temporal query, Trajectories query, topological query and predictive query.

3.3.6 aCN-RB-Tree:

aCN-RB-Tree [50] is efficient index structure for spatio-temporal aggregation of trajectory in a constrained network. It manages aggregation values of trajectories using constraint network based index. Also, it supports direction of trajectory, efficient search for traffic zone and trajectory of the time interval. The updation method of aCN-RB-tree, considering moving object features. The method updates index efficiently as it divided moving object data by area and time. It supports traffic management systems and mining systems in ubiquitous environment.

3.3.7 GPR-Tree:

Grid Partition R-Tree (GPR-Tree) [51] is used to find trajectories of moving objects on fixed network. By dividing the network space into grids of different size and indexing trajectories in each grid, GPR-Tree provides efficient query as well as updating performance for moving objects' trajectories. GPR-Tree is composed of a hash table and a 2D R*-Tree to index road network, and a forest of 2D R*-Trees to index trajectories. Algorithms for trajectory update, spatiotemporal window query are also provided.

3.3.8 FT –Tree:

This Index method is used to predict past, present and future prediction of the moving objects [52]. The moving object index for full temporal query, proposes a moving object index structure FT tree for temporal query, invites theories of various versions into TPR tree, improves the TPR tree that supports future query, and keeps various versions of TPR tree effectively to realize past, present and future temporal query.

3.3.9 M-Tree:

In this works [53] supports to pattern-matching queries for trajectories constrained by road networks. Main goal of this work is first investigate the requirements for the trajectory representation and distance measure for our target problem. The trajectory representation and the distance measure, which fulfill all the requirements. More specifically, three pattern-matching queries (whole, sub pattern, and reverse sub pattern matching) to search for similar trajectories to the given query trajectory. Though the notion of similarity varies across different types of queries, proposed a unified framework efficiently supporting range and KNN queries for all three types of matching based on M-tree and pruning rules. The validated the quality of results by visualizing the results for different types of queries over real-life road network trajectories.

3.3.10 HTPR-Tree:*

This method used to predictive queries but also partial history ones involved from the most recent update instant of each object to the last update time. This index structure, named History Time-Parameterized R-tree (HTPR*-tree)[54], which takes into account moving object creation time or update time in the leaf node entry, and supports partial history query. The HTPR* tree is bottom-up update approach referencing the R-tree update technique to support frequent update operation of the HTPR*-tree. The experiments that the update performance of the HTPR*-tree is better than that of the TD_HTPR*- and TPR*-tree.

3.3.11 MSMON:

MSMON -Tree [55] is used to retrieve the past, present and future position of network constrained moving objects.FNR & MON tree only deal with history of data and cannot support the query of moving objects in the current and future positions. Two main methods for modeling of road network structure, 1) edge based model 2) Route based model Route based model is better than the edge based model because of the smaller expression of moving objects and also data volume is large, reduce the amount of index data.MG2R* index method is proposed to retrieve past, present and future position. It is a general layered structure of the current index of moving objects on road network. It is a two tire structure and it consists of multi-Grid – R* tree used to index the road network. R* tree is used to a route in road network and also find trajectories of the moving objects in the road. This method is support the following query types Global history query, the time query, Spatial Query, spatial-temporal query and future position query.

3.3.12 PPFN:*

PPFN*-Tree [56] Indexing of moving objects grouped into two categories are deal with past information retrieval and future data prediction. This work is used to store past trajectories, present position and predict near future position of moving objects. It is hybrid structure which consists of 2DR* managing road networks and set of TB*-tree indexing objects movement history trajectories along polyline. HTPR* indexing the position of the moving objects after recent updates. Update improvement purpose hash table is used with 2DR*. This indexing methods are supported following queries Spatio-temporal range Query, Trajectory Query and Topological Query.

3.4 UNKNOWN DATA PREDICTION

Unknown data prediction is also one of the future data detection and it is applied into nonlinear environments. Main applying areas are traffic detection, price detection, weather detection etc. Some of the predicting indexing methods and supporting queries are as follows.

3.4.1 TPR-Tree:*

This TPR*-tree index [57] is the first analytical model that 1) accurately estimates the costs of predictive window queries, and 2) quantifies the performance of spatio-temporal access methods. Then it presents the TPR*-tree, a new spatiotemporal access method highly optimized for moving data. This work initiates several interesting directions for future work. Firs investigated alternative predictive queries using the TPR*-tree, in particular, nearest neighbors and joins. A predictive nearest neighbor query specifies a (moving) query point q and retrieves the database objects that will come closest to q during the query interval. A predictive spatio-temporal join will return all pairs of objects from two datasets (each indexed by a TPR*-tree) that will come within distance d from each other during the query interval.

3.4.2 STP:

This is the first a general framework [58] for monitoring and indexing moving objects. Existing work for prediction in spatial-temporal database move according to linear function. So practical movement is more difficult and overcome this problem introduced a framework for monitoring and indexing the moving objects. The proposed work accurate the capture its movement and a server side index the object location using filter refinement mechanisms. This work is support nonlinear motion pattern. The proposed index method effectively predict without false misses. The general client server arch used for answering typical spatio-temporal queries on objects with unknown. Queries are processed using filter refinement mechanism. The recursive motion function is used to find large number of movement type. Using this movement the next location can be found according to the trend of its own movement. STP-tree (Spatio-Temporal prediction tree) is an access method for indexing the expected trajectories and it is used to reduce the number of location update and false hit during the query processing. The supporting queries are Range Queries and Trajectories Queries.

3.4.3 Bx-Tree:

The majority of previous proposals assume that the position of an object is represented by a near-past position or by a linear function of time based on an exact near-past position and velocity. In contrast, this work makes the realistic assumption that the current and near-future position of an object is to be determined from a near-past position and velocity for which only a stochastic distribution is known. Thus, positions are uncertain. Meihui Zhang Su Chen [59] presents techniques that enable the efficient inferencing of current and near-future uncertain locations from past uncertain velocity and location information. This work demonstrates how it is possible to index the resulting uncertain moving objects by means of an adapted Bx-tree. And it provides techniques for processing probabilistic range and nearest neighbor queries.

3.4.4 OLSVR:

This work is proposed an online support vector regression (OLSVR) approach [60] for the prediction of short-term freeway traffic flow and compared the performance of OL-SVR to other prediction algorithms. While compared to the Gaussian maximum likelihood (GML) method is slightly better for one-step ahead short-term prediction under ‘‘normal” or non-incident conditions, OL-SVR outperforms GML and other algorithms, such as Holt exponential smoothing and neural net, at some vehicle detection stations (VDS) under atypical conditions such as holidays and incidents. It should be noted that the prediction of traffic flow under atypical conditions is evidently more challenging than doing so under typical conditions and, hence, much desired by operational agencies. Therefore, the proposed OL-SVR is found to be suitable and useful in real-world operations. This advantage is further strengthened as OL-SVR is inherently fast-paced in its data feeding and analyzing processes.

Another some of prediction methods of general motion prediction, network mobility model based prediction, Traffic prediction and GIS prediction are as follows. Another one of the proposed a motion prediction method [61] based on three motion patterns: staying, moving straight, and moving randomly to make predictive indexes for moving objects. Moreover, evaluation results showed the advantages of our methods in experiments that compared previous prediction techniques using practical trajectory data. This method supposes the trajectory data can be obtained accurately and completely; however, we should introduce a complementary method for missing trajectories.

Network mobility model [62] is used for predict concisely and effectively capturing the turning patterns of moving objects at road junctions and estimating the objects’ travel speeds on road segments. Most existing object movement prediction schemes focus on near-term predictions, e.g., to reduce update rates. These techniques are incapable of predicting the turning behaviors of moving objects at road junctions. Based on this model, develop two algorithms for predicting the future path of a mobile user moving in a road network. The *Maximum Likelihood* algorithm returns paths that maximize the travel probability among all possible paths, while the *Greedy* algorithm aims at being highly efficient while computing near maximum probability paths. This work is support a novel indexing method is based on the Greedy algorithm for supporting efficient processing of predictive range queries on the server side.

The goal of this work [63] is to predict a highly accurate and scalable method for traffic prediction at a fine granularity and over multiple time periods. The accuracy exceeds that of other published work on 15-min data, and can achieve very good accuracy on the more volatile 5-min data. In addition, accuracy remains very good up to 12 5-min time periods into the future. The method takes into account the spatial characteristics of a road network in a way that reflects not only the distance but also the average speed on the links. Because the method is designed to minimize the number of parameters needed to estimate, it remains computationally light and hence can be scaled to even large metropolitan areas. This work if the network topology is changed may fit a new model with the new spatial adjacency matrix. Another interesting aspect is to create a simultaneous prediction model for volume and speed jointly.

GIS-based prediction system [64] to analyze hotel value and estimate objective room rates. These work has two types of models, the first static model is based on hedonic pricing theory and composed of intrinsic hotel characteristics and various locational characteristics. The second dynamic model contains historical hotel room rates. The model showed that the solution can be considerably simplified by using free and open source tools such as the Java Open Street Map Editor (JOSM), R statistical package and the Weka data mining framework and also simplified the process of external spatial data acquisition by using Open Street Map data.

4. VARIOUS QUERY PROCESSING TECHNIQUES FOR UNCERTAIN DATA

The some of the main query processing techniques are already mentioned in some survey papers [27], [65], and [66]. Herein include more uncertain data query processing techniques and applying areas. The main Query processing techniques are Skyline Queries, Probability queries, Top-K queries, nearest Neighbor queries, past, present and future detection queries and trajectories queries, Aggregate queries, frequencies queries for uncertain, sum Queries and join queries etc.

4.1 UNCERTAIN SKYLINE QUERIES

Stephan Borzsonyi et al. [67] first address the skyline operator. Skyline operation filters out a set of interesting points from a potentially large set of data points. Skyline is known as the maximum vector problem. The best example for skyline query is hotel selection based on the price and distance. In hotel example traveler consider only price and distance shown in the Fig.9.

Definition: Given a set of points *a, b, c, d, e, …., n*, the skyline query returns a set of points (referred to as the *skyline points*), such that any point $p_i \in p$ is not dominated by any other point in the dataset.

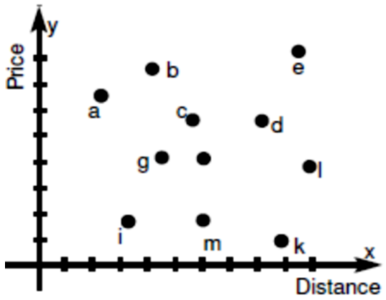

Fig.9. Example of Skyline

One of the main properties of the Skyline of a set *M* is that for any monotone scoring function $M \longrightarrow R$, *if* $p \in M$ maximizes that scoring function, then *p* is in the Skyline. In other words, no matter how you weigh your personal preferences towards price and distance of hotels, you will find your favorite hotel in the Skyline. In addition, for every point *p* in the Skyline, there exists a monotone scoring function such that *p* maximizes that scoring function. In other words, the Skyline does not contain any hotels which are nobody's favorite. The important use of skyline is multi-criteria decision making, market analysis, data mining, environment monitoring and visualization. Skyline queries aim to prune a search space of large numbers of multidimensional data items to a small set of interesting items by eliminating items that are dominated by others .The skyline query support to different models such as probability skyline model [68], continues probability model [69], incomplete data [70] etc.

4.1.1 Skyline based Probability Uncertain Model:

This probabilistic skyline model [68] is used to compute on large uncertain data sets, where an uncertain object may take a probability to be in the skyline, and a p-skyline contains all the objects whose skyline probabilities are at least *p*. this probability skyline model used two efficient algorithms, 1) Bottom-up algorithm is used to compute computes the skyline probabilities of some selected instances of uncertain objects, and uses those instances to prune other instances and uncertain objects effectively. 2) Top-down algorithm recursively partitions the instances of uncertain objects into subsets, and prunes subsets and objects aggressively. The both methods of algorithm follow three iterations bounding, pruning, refining. Using the bottom up algorithm we can find all possible dominating objects, increase the speed up the search of possible objects, and perform comparison using R-tree. Top down method each object whole data sets initially partitioned into 2 regions each region can be represented by a bounding box. Using this method we can estimate the skyline probability of bounding box into small ones until the uncertain objects can be determined against the threshold.

4.1.2 All Skyline Probabilities for Uncertain Data:

In this work [71] threshold is not desirable so low probability events cannot be ignored. In such cases it is necessary to compute skyline probabilities of all data items. This is work first sub-quadratic algorithm is used to computing all skyline possibilities, here threshold equals 0 in probabilistic skyline query. Space partition algorithm and dominance counting algorithm are used to computing all possible data. New probabilistic skyline analysis used to account different user utilities without any restriction.

4.1.3 Probabilistic Skyline Queries:

This work [69] proposed efficient and effective methods for determining the probabilistic skyline of uncertain objects, which are defined by a PDF in parametric form such as a Gaussian function or a Gaussian Mixture Model. The probabilistic skyline can be supported by an index structure for uncertain objects. Using baseline algorithm determine skyline iteration over the number n objects in the datasets, and determine skyline probability of generating objects by *S* instance. The baseline algorithm has only a linear runtime complexity with respect to the number of instances. Priority algorithm is used to avoid exact skyline probability for each object from large data sets. Indexed algorithms for storing common indexing technique uses minimal bounding rectangles (MBR). The MBR used without looking at exact objects can prune. The indexed approach is backed by a priority queue held in main memory. Priority queue is sorted on descending order and first element is inserted in the root of the index. The next element to be processed the one with the highest current skyline probability. This work experimental evaluation is based on synthetic and real world data.

4.1.4 Reverse Skyline Queries:

Reverse skyline queries are used first in the work [72]. The goal of a reverse skyline query is to identify the influence of a query object on a multidimensional dataset with respect to a vector of distances. Using Reverse Skyline able to retrieve a small subset of the database as candidates for reverse skyline queries (RSQ) and prove that all reverse skyline points with respect to q, belong to the global skyline.RSQ use two algorithms for computation purpose, Branch and Bound Reverse Skyline (BBRS) and RSSA. BBRS is used to improved customization of the original BBS algorithm and partitioning access method using R*-Tree. Reverse Skyline using Skyline Approximations (RSSA) algorithm for supporting reverse skyline queries which is based on the well known filter-refinement paradigm. The main idea is to compute the dynamic skyline for each database object and to keep a fixed-sized approximation of this skyline on disk. By using this approximation in the filter step, we are able to identify points being in the reverse skyline as well as to filter out points not being in the reverse skyline. The remaining candidate points are then further examined in the refinement step. Similar to the method presented in the previous section, a window query is issued for each candidate, but the size of the window can be substantially reduced due to the approximation again. Consequently, this also leads to substantial cost savings. The approximation scheme is to pre-compute the dynamic skyline for each object of the database and to select a fixed number of skyline points. Greedy algorithm is used to optimal purpose and experiments are done with both real-world and synthetic datasets.

4.1.5 Reverse Skyline in Monochromatic and Bichromatic:

In this work the probabilistic reverse skyline query on uncertain data, in both monochromatic and bichromatic cases [73], and proposed effective pruning methods to reduce the search space of query processing. Moreover, efficient query procedures have been presented seamlessly integrating the proposed pruning methods. Monochromatic probabilistic reverse skyline (MPRS)/bichromatic probabilistic reverse skyline (BPRS) are consists of three phases, indexing, pruning, and refinement. Indexing phase builds up a multidimensional index over uncertain database(s). Pruning phase aims to prune those objects that are not qualified as query results (i.e. Inequality). In this phase only proposed spatial and/or probabilistic pruning methods to significantly reduce the search space. The refinement phase checks Inequality and reports the qualified objects. MPRS two pruning heuristics, spatial and probabilistic pruning, the spatial pruning method only utilizes the spatial property of uncertain objects, that is, the bounds of uncertainty regions, to reduce the MPRS search space. The probabilistic pruning method, which is exactly, aims to utilize this distribution information to increase the pruning power. Experimental results shown MPRS/BPRS queries produce best effective results in various real time continuous applications.

4.1.6 Probabilistic Skyline Operator:

This work is based on continuous skyline queries over sliding windows [74] on uncertain data elements regarding given probability thresholds. This method is highly desirable to develop on-line, efficient, memory based, incremental techniques using small memory. The main process is minimum information needed in continuously computing probabilistic skyline against a sliding window. The volume of such minimum information is expected to be bounded by logarithmic size in a lower dimensional space regarding a given window size. The data retrieve purpose threshold top-k probabilistic skyline used. Skyline operator [67] the content of the database are applied in the SQL syntax for skyline queries. It also used two computation techniques based on block-nested-loop and divide-and-conquer paradigms for computation. It is applied in the continuous queries in different threshold value and it is used in the ad-hoc skyline queries.

4.1.7 Exact Skyline Probability Queries:

Previous probabilistic skyline computation only identifies those objects whose skyline probabilities are higher than a given threshold, or is useful only for 2D data sets. In this work, we develop a probabilistic skyline algorithm called PSkyline which computes exact skyline probabilities of all objects in a given uncertain data set [75]. PSkyline aims to identify blocks of instances with skyline probability zero, and more importantly, to find incomparable groups of instances and dispense with unnecessary dominance tests altogether. To increase the chance of finding such blocks and groups of instances, PSkyline uses a new in-memory tree structure called Z-tree. We also develop an online probabilistic skyline algorithm called O-PSkyline for uncertain data streams and a top-k probabilistic skyline algorithm called K-PSkyline to find top-k objects with the highest skyline probabilities.

4.1.8 Stochastic Skyline Operator:

A skyline operator, namely stochastic skyline, is proposed in [76], [27] to improve the practicability of the uncertain skyline queries. Stochastic skyline model over uncertain data can ensure to provide a minimum set of candidates to the optimal solutions over the family of multiplicative decreasing scoring functions for users to make their personal trade-offs. The operator is based on the stochastic dominance, which is proved to be NP complete regarding the dimensionality. In addition, an efficient algorithm on large set of objects based on novel filtering and verification techniques is developed for stochastic skyline computation in.

4.1.9 Distributed Skyline over Uncertain Data:

Centralized storage is one of the main issues in uncertain skyline queries. It takes large amounts of independent data storage and high rate of data generation make central assembly of data at one location be infeasible. The uncertainty is used in many distributed applications like P2P network and sensor networks, due to the lack of central control that verifies the quality of the data. Due to the large amounts of data stored and the network delay and economic cost incurred, it is often too expensive to transfer the entire dataset from each local site to the centralized data server for query processing. Thus, skyline query processing over uncertain data is very important and challenging [27], [77].

Efficient and progressive algorithms for distributed skyline queries addressed the uncertain queries in distributed environments [78]. It is proposed DSUD and e-DSUD algorithms to process the queries over uncertain data. The efficiency of the algorithms stems from a highly optimized feedback mechanism, where the central server transmits precious information to each local site to prevent the delivery of a large number of unqualified skyline tuples. Besides low-bandwidth consumption, both algorithms can achieve excellent progressiveness in returning the results, but the overall query time is relatively long, due to the sequential processing pattern [27].

An efficient and progressive algorithm for distributed skyline queries [79] is the extension of paper [78] and the same DSUD and e-DSUD algorithms used. But it is focuse on arbitrary horizontal partitioning, where a local site has all the attributes but stores only a subset of the entire tuples. As an interesting direction, vertical partitioning between distributed data still exists in the context of uncertain data. Probabilistic R-tree and continuous updating are the advantages of this work.

4.1.10 U-Skyline:

U skyline is used to uncertain databases, this work a new uncertain skyline query, called U-Skyline Query [80]. Instead of setting a probabilistic threshold to qualify each skyline tuple independently, the U-Skyline query searches for a set of tuples that has the highest probability as the skyline answer. In order to answer U-Skyline queries efficiently propose a number of optimization techniques for query processing.

1) Computational simplification of U-Skyline probability. U-Skyline aims to find a tuple set with the highest U-Skyline probability. Therefore, the main aim of this techniques is simplifying the U-Skyline probability computation is important. The U-Skyline probability for a candidate skyline S is obtained by enumerating all possible worlds. This brute force strategy is infeasible due to the exponential running time complexity O (2^N). An idea to improve the U-Skyline probability computation is to combine the probability computation for multiple possible worlds that share the same skyline sets. It is integer programming formulation.
2) Pruning of unqualified candidate skylines and early termination of query processing. In this techniques first proposed a dynamic programming algorithm to obtain U-Skyline from uncertain data sets, and then improve this algorithm with pruning and early termination techniques to avoid unnecessary computation. Recursive computation, dynamic programming and skyline pruning are used for computation.
3) Reduction of data set and partition, U-Skyline from an uncertain data set, proposed two additional techniques, namely, input data set reduction and reduced data set partition. It is used to speed up U-Skyline search by reducing the input data set and applying the strategy of divide-and-conquer in query processing. This method is compared to a commercial parallel integer programming solver and conducted experiment is 10-100 times faster than the parallel inter programming solver.

4.1.11 NSkyline Query for Incomplete data:

This work is used to find the incomplete data items [70]. Incomplete data items mean missing values in some of their dimensions. In complete data means the dominance relation is transitive, incomplete data suffer from non-transitive dominance relation which may lead to a cyclic dominance behavior. In traditionally Replacement and Bucket algorithms are used for skyline. This work used "ISkyline" specifically for incomplete data. ISkyline have two optimization techniques namely virtual points and shadow skyline are used to tolerate cyclic dominance relations. The main purpose of virtual points is to reduce the number of points in the candidate skyline list and virtual point cannot simply perform an exhaustive pairwise comparison of all points in the candidate skyline list to get the query result, so shadow skyline are introduced. The concept of shadow skylines that works together with virtual points to alleviate the problem of storing and comparing all input data. The main idea is that we do not need to store all points in each bucket, instead, we only need to store the skyline set of points not found in the local skyline list.

4.2 TOP–*k* QUERIES FOR UNCERTAIN DATA

Top-*k* Queries is used for various ranking purpose and it is applied in uncertain data also. It is one of the variant of rank queries. First it is introduced in the multimedia system [27], [81], [82], such that to find top-*k* answers from the given objects (top-*k* objects). So top-*k* queries used search highest grade for the given queries from given objects. The definition of top-*k* Queries [83] is,

Definition:

Uncertain Top-*k* Query: Let D be an uncertain database with possible worlds space $PW = \{PW^1,\ldots\ldots, PW^n\}$. Let $T = \{T^1,\ldots\ldots,T^m\}$ be a set of k-length tuple vectors, where for each $T^i \in T$: 1) Tuples of T^i are ordered according to scoring function F and 2) T^i is the top-*k* answer for a non-empty set of possible worlds $PW(T^i) \subseteq PW$. A U-Top *k* query, based on F, returns $T^* \in$ T, where, $T^* = \arg Max_{T^i \in T}\left(\sum_{w \in PW\left(T^i\right)}\left(\Pr\left(w\right)\right)\right)$.

Top-*k* queries are applied into various uncertain data models. In this part give briefly about various top-*k* queries related to the corresponding models. The first top-*k* uncertain query [83] described two types of probability top-*k* queries such as U-top-*k* and U-*k*Ranks. Continuously various top-*k* based query processing are mentioned here [27], and continuing remaining some of the main queries model related to top-*k* queries.

4.2.1 Dynamic Structure for Top-k Queries:

Dynamic structure [84] for the top-*k* problem is used to an update cost of uncertain data set $S = (S, p, f)$. In the dynamic problem, top-*k* query answer when S changes, the uncertain data set undergoes a series of updates, including insertion and deletions in the ground set S, changes in the probability function *p* and the score function *f*. The dynamic data structure handles the updating time and answers the top-*j* queries. The dynamic data structure is binary tree and the advantage is using this method solve all Top-*k* queries.

4.2.2 Polynomial U-top k Queries:

Polynomial-time algorithms for processing top-*k* queries in uncertain databases are generally adopted model of x-relations [85]. An x-relation consists of a number of x-tuples, and each x-tuple randomly instantiates into one tuple from one or more alternatives. This work provide solution for both U-Top-*k* queries and U-*k*Ranks queries, both of which are significantly faster and use much less space under the x-relation model.

4.2.3 Sliding Window Top-k Queries:

The first work provide solution for sliding window using top-*k* queries on over uncertain data [86]. Cheqing Jin and Ke Yi introduced framework for processing continuous top-*k* queries in a sliding window over uncertain data streams. This model proposed both space and time-efficient based results with the maintenance of a compact set. The results of this sliding window provide data compression, buffering, and ideas from exponential histograms.

4.2.4 Top-k Queries using Score Distribution:

Score distribution of top-*k* vectors [87] to allow the user to choose between results along this score probability dimensions. The complete distribution of all potential top-*k* tuple vectors are used instead of large data compute. The proposed work provides a number of typical vectors that effectively sample this distribution. The semantic method is used to answer of top-*k* queries on uncertain data to the applications. Using top-*k* vectors and the *c*-Typical-Top *k* provide the distribution of the total scores of top-*k* tuples. The solutions are ties the descending probabilities are used for solving ties solutions.

4.2.5 Probabilistic Threshold Top-k Queries:

Probabilistic threshold provide the solution for answering probabilistic threshold top-*k* queries on uncertain data and using threshold provide ranking [88], which computes uncertain records taking a probability of at least *P* to be in the top-*k* list where, *P* is a user specified probability threshold. Here mentioned how probability top-*k* queried and probabilistic threshold top-*k* query be answered. Probability top-*k* queries computing purpose exact algorithms are used, the dominant property sets, basic sets, generating multiple rules are used to generate top-k queries. Sampling method is used to accuracy of the given solutions. Poisson approximation based method derive a general stopping condition for query answering algorithms which depends on parameter *k* and threshold *P* only and is independent from data set size.

4.2.6 Top-k for Uncertain Data on Graphics Processing Unit:

Top-*k* queries in uncertain database, the processing speed and the degree of parallelism are significant factors people concerned [89]. In this work proposed a CPU-GPU cooperative processing framework. Using GPU brings advantages in top-*k* queries processing in uncertain database. A GPU is a collection of multiprocessors at the hardware level. And each multiprocessor has several elements to support thousands of threads simultaneously, named scalar processors (SP). GPU can not directly access RDBMS residing in the CPU host, so we propose a CPU-GPU cooperative processing framework consisting of two main parts(CPU and GPU part), and two layers(storage and processing layer). In the experimental level error detection techniques are introduced for accuracy purpose.

4.2.7 Scalable Probabilistic Top-k Similarity Ranking on Uncertain Data:

In probabilistic top-*k* similarity ranking [90] a framework based on iterative distance browsing that efficiently supports probabilistic similarity ranking in uncertain vector databases. The distances browsing incrementally retrace all the objects in order to their distance. Present a novel and theoretically founded approach for computing the rank probabilities of each object and reduce the linear-time complexity and less memory requirements compared to the other methods. Incremental accessing of the uncertain vector instances in increasing order of their distance to the reference object. The probability ranking framework is working two modules distance browsing and Probability ranking. Distance browsing modules used to incrementally retrieve all the objects in order their distance. This can be achieved using R*-Tree index and nearest search algorithms. The probability ranking used to find rank position using U-*k*Ranks and PT-*k*. This work is applicable to large database and future can apply various uncertainty models.

4.2.8 Multi Top-k Queries over Uncertain Data:

Most of the data in the real world is incomplete and particular to one query is not shared all the data. So Tao Chen and Lei Chen proposed sharing among multiple top-*k* queries over uncertain data streams based on the frequency upper bound of each top-*k* queries [91]. The main goal of this work is to share the computation queries among the possible and satisfy the frequency semantics at the same time. First it is performed single query with probability tuples and time based sliding window with size which is based on the top-*k* probability. Sharing queries is very challenging for the uncertain top-*k* queries with different frequency upper bounds and different *k* values. First grouping based on the same frequency bound is called inter grouping. The intra grouping queries with the same frequency upper bound but different *k* values. Two types of the work used for similarity search. One is combine group with different frequency upper bonds and another one is sharing among queries between groups after combination. Dynamic programming solution is used to find optimal solution in sharing between groups and overlapping sub problems. Greedy algorithm is used in single queries, it is not a optimal solution but it is more efficient than dynamic program in term of space and time.

4.2.9 Top-k Similarity Queries on Uncertain Trajectories:

It used to find the similarity queries on uncertain trajectories [92]. Top-k similar query (KSQ) is used to search similar queries. UTgrid is used for indexing uncertain trajectories and develop query processing algorithm is used to effective puring and proposed p-distance, the holistic approach that take into account all trajectories in the database. It measures the dissimilarity between two uncertain trajectories. The p-distance based result avoids result distortion based by data outliers. P-distance based KSQ retrieve the *k*-most similar trajectories to the query processing. The UT-grid (uncertain trajectories based on grid partitioning) is manage large amount of uncertain trajectories. UT grid is spatio-temporal uncertain trajectories based on grid partitioning.

4.3 NEAREST NEIGHBOR QUERIES FOR UNCERTAIN DATA

Nearest neighbor Queries (NNQ) used to many features in uncertain databases. NNQ in uncertain data have many applications such as publicly share trajectories, such as route sharing, nearest service sharing and GPS trajectories. In Many applications, the frequently data collection decrease to save resources and wireless network traffic. NNQ in Uncertain and Trajectories databases not have a common definition but E. Frentzos et al. [93] defined as given a query trajectory (or spatial point) *q* and a time interval *T*, a NN query returns either the trajectory from the database which is closest to *q* during *T* or for each $t \in T$ the trajectory which is closest to q. Nearest neighbor queries in uncertain and trajectories are used different techniques such as PNNQ [94], [95], PRNNQ [96], [77], CPNNQ [97], SNN [98]. These techniques in NNQ are applied in the both spatial and spatiotemporal applications.

4.3.1 PNNQ on Uncertain Objects:

Probability nearest neighbor is a successful method to express the distance between two uncertain objects using probability density function (PDF) [94]. The PDF is assigned a value to each possible objects and integrating the complete probabilistic distance function. The PNNQ based on the smooth distance is used to find similar objects. The PNNQ based on the discrete is used to find different dissimilar objects. The nearest neighborhood queries are used to find the minimal and maximum distance between two objects. NNQ are used predict the minimum and maximum distance between two objects.

Another work of PPNQ focused on probabilistic nearest neighbor queries in database with uncertain moving trajectories modeled by stochastic process and Marko chain model [95]. The Marko model is simple used to only temporal data dependencies. This work proposed three different semantics of the queries P8NNQ queries, PNNQ queries and PCNN queries. Bayesian inference used to give the guaranteed sampled trajectories to conform observation data stored in the database. The sampled approach can be used in monte-carlo based approximation solution. Apriori pattern mining approach is used to reduce the cardinality of the PCNNQ query.

4.3.2 PRNNQ over Uncertain Data:

Reverse nearest neighbors (RNN) queries retrieve all the objects in their nearest neighbors [96]. The probability revise nearest neighbors (PENNQ) query which obtain data objects with probability of RNNs greater than or equal to user specified threshold. The proposed method for object retrieving is called geometric pruning (GP) which is used to reduce the PRNN search space without introducing any false dismissals. The PRNN queried using sequential scan and it is a multidimensional (R-Tree, M-Tree or SR-Tree) indexing methods are used. This index method is fully applied into uncertain environments. The actual query retrieve propose PRNN is combined with heuristics of GP.

Muhammad Aamir Cheema, Xuemin Lin [77] also proposed Probability reverse nearest neighborhood queries on multidimensional uncertain data. PRNNs of an uncertain query object *Q*. The data are stored in system as follows: for each uncertain object, an R-tree is created and stored on disk that contains the instances of the uncertain object. Each node of the R-tree contains the aggregate appearance probability of the instances in its subtree. We refer these R-trees as local R-trees of the objects. Another R-tree is created that stores the MBRs of all uncertain objects. This R-tree is also called global R-tree. Several optimization methods are presented and it reduced the overall computation.

Similar work is proposed probabilistic reverse nearest neighbor (PRNN) queries, which return the uncertain objects having the query object as nearest neighbor with a sufficiently high probability. We propose an algorithm for efficiently answering PRNN queries using new pruning mechanisms taking distance dependencies into account. This work is compared with state-of the-art approaches recently proposed. Our experimental evaluation shows that our approach is able to significantly outperform previous approaches.

4.3.3 CPNNQ for Uncertain Trajectories:

Continuous Probability nearest neighbors Query (CPNNQ) is time parameterized method [97]. This method is applied in spatial and temporal network and it sense the objects in vary over time. The new index method is introduced called interval based probabilistic answer to a continuous NN query (IPAC-NN). The main working procedure of this tree identify a transformation of the uncertain trajectories, probability density function (PDF) describe the uncertainty associated with the location, construct a geometric dual of an IPAC-Tree. The crisp query objects are used to find the location of the other trajectories with some radius. This method is used to find the uncertain data in large areas. This method is compared to Top-K queries.

4.3.4 SNNS:

Superseding nearest neighbor search [98] is used on uncertain spatial databases. Each object is described in multidimensional pdf. Sometimes no objects supersede NN candidates. Such situation all the objects are outside of the SNN-core and it take best objects from all the objects in the SNN-Core.SNN search conventional R-tree is used to prune the search space to achieve I/O effectively. The Best fit (BF) algorithm is used to retrieve the precise data on NNS. Nearest neighbor on uncertain data is searched using expected distance(ED) principle and PR-principle is used to define result of NN queries on uncertain data. The full graph is used find the SNN –core and pipeline algorithm is used incrementally retrieves object instance in ascending order of their distance. The graph and pipeline algorithm is used smaller cost than sequential scan and pipeline is used R-tree can be accessed efficiently.

4.3.5 PNNQ on Uncertain Moving Objects Trajectories:

Continuous query processing take long time to evaluate. To solve this problem, Jinchuan Chen et al. [99] proposed the Constrained Nearest-Neighbor Query (C-PNN), which returns the IDs of objects whose probabilities are higher than some threshold, with a given error bound in the answers. C-PNN can be answered efficiently with verifiers. These are methods that derive the lower and upper bounds of answer probabilities, so that an object can be quickly decided on whether it should be included in the answer. The incremental refinement the data are listed in the hash table. The five verifiers, which can be used on uncertain data with arbitrary probability density functions.

Further develop a partial evaluation technique, so that a user can obtain some answers quickly, without waiting for the whole query evaluation process to be completed. In addition, we examine the maintenance of a long-standing or continuous C-PNN query R-tree is used to investigate. This query requires any update to be applied to the result immediately, in order to reflect the changes to the database. Here a continuous incremental update method based on previous query answers, in order to reduce the amount of I/O and CPU cost in maintaining the correctness of the answers to such a query.

4.4 UNCERTAIN AGGREGATE QUERIES

Uncertain aggregate query is an extension of aggregate query over uncertain data. The existing studies on uncertain aggregate queries can be categorized as two aspects: queries over static datasets and queries over dynamic data streams. The early research of aggregate queries over static datasets mainly focuses on the queries for moving objects. Predicting the future situation by collecting the trajectory of moving objects and other historical data is the focus of the early studies. As the development of probabilistic database technology, a variety of sampling methods and summary structures of the traditional aggregate queries have a certain reference value. To solve the aggregate query problem for moving objects in the moving objects database [27]. The Aggregations queries are applied into aggregation operations [100], aggregation constraints [101], uncertain probabilistic database [102,103], and range location based queries [104].

In general aggregation operator [100] that allows us to determine a probability distribution for attributes values either for a single attribute or for the cross-product of a number of attributes. Such functionality has potential for knowledge discovery in databases where we may use it to assess the validity of potential association rules. The imprecise and uncertain data should facilitate the speed and efficiency of knowledge discovery using such imperfect data using aggregation operation. Another important category of aggregation is the aggregate constraints [101]. An aggregate constraint is placed on a set of records instead of on a single record, and a real life system usually has a large number of aggregate constraints. It is a challenging task to find qualified possible worlds in this scenario, since tuple by tuple sampling is extremely inefficient because it rarely leads to a qualified possible world. Mohan Yang and Haixun Wang [101] introduced two approaches for querying uncertain data with aggregate constraints: constraint aware sampling and MCMC sampling.

Raghotham Murthy, Robert Ikeda et al. [102] introduced aggregation querying in uncertain and probabilistic database. Aggregation Query is handled in the Trio system for uncertain and probabilistic data. The “exact” aggregation in uncertain databases can produce exponentially sized results. The three alternatives: a low bound on the aggregate value, a high bound on the value, and the expected value. These variants return a single result instead of a set of possible results, and they are generally efficient to compute for both full-table and grouped aggregation queries. Mao Ye, Ken C.K. Lee [103] also applied probabilistic aggregation querying into wireless sensor networks. Here is focued on finding possible minimum values and sensor nodes that provide the possible minimum values in wireless sensor networks. Specifically, formulate two probabilistic queries, namely, Probabilistic Minimum Value Queries (PMVQ)s and Probabilistic Minimum Node Queries (PMNQ)s and their variants, and develop in-network algorithms to answer them energy efficiently in wireless sensor network.

Ying Zhang, Xuemin Lin [104] proposed the problem of uncertain location-based range aggregate query in a multidimensional space; it covers a wide spectrum of applications. To efficiently process such a query, we propose a general filtering and verification framework and two novel filtering techniques, named STF and APF, respectively, such that the expensive computation and IO cost for verification can be significantly reduced.

4.5 FREQUENT QUERY PROCESSING IN UNCERTAIN DATA

Frequent query means a sets of data discovered from uncertain data are naturally probabilistic, in order to get the results based on the frequently used [105]. A PFI is a set of attribute values that occurs frequently with a sufficiently high probability. A simple way of finding PFIs is to mine frequent patterns from every possible world, and then record the probabilities of the occurrences of these patterns. Most of the work on frequent query is based on association rules and Apriori Algorithms and also described some of the recent work on related to frequent Queries.

4.5.1 Mining Frequent Subgraph Patterns from Uncertain Data:

Many numbers of works are introduced in mining frequent items from exact database. The mining frequent Subgraph is discovering the patterns from an uncertain graph database [106]. This is a novel model of uncertain graph data has been proposed, and the frequent subgraph pattern mining problem has been formalized by introducing the concept of expected support. The computational complexity of this problem has been formally proved. The data finding from unexact data is NP-hardness of the problem, an approximate mining algorithm, called MUSE (Mining Uncertain Subgraph pattErns), and has been developed to discover an approximate set of frequent subgraph patterns with respect to the relative error tolerance.

4.5.2 Mining of Frequent Item Sets on Large Uncertain Databases:

Frequent items set from large uncertain data query [105] is based threshold based probability frequent items (PFI). It is mainly based on the some common probability model, it is quickly verified and also this method is used two incremental algorithms such as exact and approximate methods for retrieving PFIs from evolving databases. This query retrieving method is supports for both attributes and tuple uncertain data. The experimental result of this work is effective and accurate query processing in large database.

4.5.3 Frequent Query Items from Distributed Uncertain Data:

Many query processing in distributed frequent items algorithms are do not allow user to express the items according to the own constrains. The proposed tree based frequent item methods [107] handle different types of user defined constrains. In this method first the system identified domain items and that satisfy constrains in distributed items and then constrains of UF tree, which constrained frequent items recursively. The experimental results are frequent items in distributed uncertain data is effective.

4.5.4 SOLE

Scalable On-Line Execution algorithm (SOLE) is the first attempt to combine spatio-temporal continuous query processing techniques [108] with data stream management systems to sup-port continuous queries over spatio-temporal datastreams. The continuous spatio-temporal queries may encounter uncertainty areas SOLE can overcome such uncertainty areas using a conservative caching technique. A scalable framework for SOLE that modifies the commonly used shared execution paradigm to support data streaming environments.

4.6 JOIN QUERIES OVER UNCERTAIN DATA

Join operation is one of the main operations in SQL queries operations. Mainly in the join uncertain attributes are joined together. The main studies of join queries are classified into probabilistic join queries and uncertain similarity joins, according to the probabilistic predicates and distance measures. Join Queries on Uncertain data in Semantics. One of the main works is the SQL join operation [109] on uncertain attributes. Two join operations are focused such data, namely v-join and d-join. They are each useful for different applications. Using probability theory, we then devise efficient query processing algorithms for these join operations. Specifically, we use probability bounds that are based on the moments of random variables to either early accept or early reject a candidate v-join result tuple. For v-join devise the Two End Zigzag Join algorithm which is a combination of using probability theory and our novel indexing mechanism. This saves both CPU and I/O costs. We reduce a d-join to a similarity join and devise the condensed d-join algorithm and an optimal condensation scheme based on dynamic programming.

4.6.1 Similarity Join Query Processing for Uncertain data

The similarity join query processing pruning methods [110] into an efficient query procedure that can incrementally maintain the join on uncertain data stream (USJ) answers. Most importantly further design a novel strategy, namely, adaptive superset pre-join (ASP), to maintain a superset of USJ candidate pairs. ASP is in light of our proposed formal cost model such that the average USJ processing cost is minimized. We have conducted extensive experiments to demonstrate the efficiency and effectiveness of our proposed approaches. Mainly this work is used to filter the Data from false data.

4.7 SUM QUERIES OVER UNCERTAIN DATA

SUM queries are crucial for many applications that need to deal with uncertain data [111]. In this work deal with ALL_SUM queries and pseudo polynomial algorithms that are efficient in many practical applications, e.g., when the aggr attribute values are small integers or real numbers with small precision. These cases cover many practical aggregate attributes, e.g., temperature, blood pressure, needed human resources per patient in medical applications. The proposed a new recursive approach, we developed an algorithm, called Q_PSUM, which is polynomial in the number of SUM. Then, proposed a more efficient algorithm, called DP_PSUM, which is very efficient in the cases where the aggr attribute values are small integers or real numbers with small precision.

5. RESEARCH DIRECTIONS AND APPLICATION FIELDS

5.1 RESEARCH DIRECTION

Uncertain data and uncertain Query processing is one of the emerging fields in research such as database, data mining, network computing, pervasive computing and different aspects of real time applications. Even though a number of studies have been investigated the uncertain queries in various scenarios with various definitions, there are still many interesting and challenging issues about the uncertain queries that have not been studied so far in the related literature. According to the current status of research, we summarize the key of the open research directions as follows. The main further research areas in uncertain data are Indexing methods, new frame works, Uncertain Queries, parallel and distributed environment, concurrent data updating etc.

Most of the existing indexing methods are not applicable to all the types of query processing. So introduce new different types of framework for different types of queries [29]-[56].

Existing indexing methods are multi-structured and it is combination of different types of indexing methods, so it is complex index structure. The structure are very complex, the performance of the indexing method is atomically decreased [49], [54]-[56].

The existing multi-structured indexing applicable for simple updating mechanisms in moving objects environments. So no concurrent updating mechanisms are not supported. The real time applications are fully needed concurrent updations [49], [54]-[56].

In Many kinds of data are massive in scale and similar object find is needed in real applications, such as streaming data, weather data etc. So the mass data management in query processing and similar objects or group of object finding is one of the main issues for further research [108] [110], [111].

In generally nowadays the Distributed queries over decentralized data sets and Parallel query processing for massive data community have lot of issues are available and finally all the mentioned indexing methods and Query processing techniques performance improvement, new types of query processing and indexing methods are further research consideration in spatio-temporal fields.

5.2 APPLICATION FIELDS

Main application fields of spatio temporal are as follows [112]

Meteorology: all kinds of weather data, moving storms, tornados, developments of high pressure areas, movement of precipitation areas, changes in freezing level, droughts.

Biology: animal movements, mating behavior, species relocation and extinction.

Crop Sciences: harvesting, soil quality changes, land usage management, seasonal grasshopper infestation.

Forestry: forest growth, forest fires, hydrology patterns, canopy development, planning tree cutting, planning tree planting.

Medicine: patients' cancer developments, supervising developments in embryology.

Geophysics: earthquake histories, volcanic activities and prediction.

Ecology: causal relationships in environmental changes, tracking down pollution incidents.

Transportation: traffic monitoring, control, tracking vehicle movement, traffic planning, vehicle navigation, fuel efficient routes.

6. CONCLUSION

Uncertain indexing and query processing is emerging research field in spatio-temporal data and it is applied into various application fields, which is required to studying the indexing and queries over uncertain data. This survey paper gives brief information about various uncertain indexing and query processing techniques. In the indexing, three different forms indexing methods are mentioned, limitations are covered and various types of the query processing techniques and supporting queries are mentioned. The most of the indexing methods are only supports few queries and past, present and future indexing methods structure are very complex because it is integrated different indexing methods. The structure of the indexing methods are very complex so performance is decreased and main expectation of real time application is concurrent updation, it is not supported by most of the indexing methods. Real time application are need similar object find and grouping that is also not feasible in exiting indexing methods and finally but not least all the query processing techniques are not supports to all the types of indexing methods and queries. These are the few points of further research direction in spatio-temporal data environments.

REFERENCES

[1] X. Meng and J. Chen, "*Moving Objects Management: Models, Techniques and Application*", Springer, 2010.

[2] Sergio Ilarri, Eduardo Mena and Arantza Illarramendi, "Location-Dependent Query Processing: Where we are and where we are Heading", *ACM Computing Surveys*, Vol. 42, No. 3, pp. 1-79, 2010.

[3] Chi Yin Chow, Mohamed F. Mokbel and Walid G. Aref, "Casper: Query Processing for Location Services without Compromising Privacy", *ACM Transactions on Database Systems*, Vol. 34, No. 4, pp. 1-45, 2009.

[4] C.S. Jensen, "*Database Aspects of Location-Based Services*", Morgan Kaufmann, 2004.

[5] *Mohamed F. Mokbel* and Walid G. *Aref*, "*PLACE*: A *Scalable Location-Aware Database Server* for *Spatio-Temporal Data Streams*", *IEEE Data Engineering Bulletin, Vol. 28,* No. *3,* pp. 1-8, *2005.*

[6] Kyriakos Mouratidis, Dimitris Papadias and Marios Hadjieleftheriou, "Conceptual Partitioning: An Efficient Method for Continuous Nearest Neighbor Monitoring", *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pp. 634-645, 2005.

[7] O. Wolfson, B. Xu and S. Chamberlain, "Location Prediction and Queries for Tracking Moving Objects", *Proceedings of International Conference on Data* Engineering, pp. 687-688, 2000.

[8] O. Wolfson, B. Xu, S. Chamberlain and L. Jiang, "Moving Object Databases: Issues and Solutions", *Proceedings of 10th International Conference on Scientific and Statistical Database* Management, pp. 111-122, 1998.

[9] A. Civilis, C.S. Jensen, J. Nenortaite and S. Pakalnis, "Efficient Tracking of Moving Objects with Precision Guarantees", *Proceedings of the 1st Annual International Conference on Mobile and Ubiquitous Systems, Networking and Services*, pp. 164-173, 2004.

[10] A. Civilis, C.S. Jensen and S. Pakalnis, "Techniques for Efficient Road-Network-Based Tracking of Moving Objects", *IEEE Transactions on Knowledge and Data Engineering*, Vol. 17, No. 5, pp. 698-712, 2005.

[11] Z. Ding and R.H. Guting, "Managing Moving Objects on Dynamic Transportation Networks", *Proceedings of the 16th International Conference on Scientific and Statistical Database Management*, pp. 287-296, 2004.

[12] Dimitris Papadias, Jun Zhang, Nikos Mamoulis and Yufei Tao, "Query Processing in Spatial Network Databases", *Proceedings of the 29th International Conference on Very Large Data Bases*, Vol. 29, pp. 802-813, 2003.

[13] Rakesh Agrawal, Christos Faloutsos and Arun N. Swami, "An Efficient Similarity Search in Sequence Databases", *Proceedings of the 4th International Conference on Foundations of Data Organization and Algorithms*, pp. 69-84, 1993.

[14] Christos Faloutsos, M. Ranganathan and Yannis Manolopoulos, "Fast Subsequence Matching in Time-Series Databases", *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pp. 419-429, 1994.

[15] Rakesh Agrawal, King Ip Lin, Harpreet S. Sawhney and Kyuseok Shim, "Fast Similarity Search in the Presence of Noise, Scaling, and Translation in Time-Series Databases", *Proceedings of the 21st International Conference on Very Large Data Bases*, pp. 490-501, 1995.

[16] H.V. Jagadish, Alberto O. Mendelzon and Tova Milo, "Similarity-Based Queries", *Proceedings of the 14th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pp. 36-45, 1995.

[17] D. Rafiei, "On Similarity-Based Queries for Time Series Data", *Proceedings of the 15th International Conference on Data Engineering*, pp. 410-417, 1999.

[18] Dieter Pfoser and Nectaria Tryfona, "Capturing Fuzziness and Uncertainty of Spatiotemporal Objects", *Proceedings of the 5th East European Conference on Advances in Databases and Information Systems*, pp. 112-126, 2001.

[19] E. Tossebro and M. Nygard, "Uncertainty in Spatio Temporal Databases", *Proceedings of the 2nd International Conference on Advances in Information Systems*, pp. 43-53, 2002.

[20] Mohammed F. Mokbel, Thanaa M. Ghanem and Walid G. Aref, "Spatio-Temporal Access Methods", *IEEE Data Engineering Bulletin*, Vol. 26, No. 2, pp. 40-49, 2003.

[21] L.V. Nguyen-Dinh, W.G. Aref, and M.F. Mokbel ,"Spatio-Temporal Access Methods: Part 2", *IEEE Data Engineering Bulletin*, Vol. 33, No. 2, pp. 46-55, 2010.

[22] K. Appathurai and S. Karthikeyan, "A Survey on Spatio-Temporal Access Methods", *International Journal of Computer Applications*, Vol. 18, No. 4, pp. 40-42, 2011.

[23] Tamas Abraham and John F. Roddick, "Survey of Spaio-Temporal Databases", *GeoInformatica*, Vol. 3, No. 1, pp. 61-99, 1999.

[24] Charu C. Aggarwal, "*Managing and Mining Uncertain Data*", Springer, Vol. 38, 2009.

[25] Douglas W. Hubbard, "*How to Measure Anything: Finding the Value of Intangibles in Business*", 2nd Edition, John Wiley and Sons, 2010.

[26] A.D. Sarma, O. Benjelloun, A. Halevy and J. Widom, "Working Models for Uncertain Data" , *Proceedings of the 22nd international conference on data engineering*, pp. 1-7, 2006.

[27] Yijie Wang, Xiaoyong Li, Xiaoling Li, and Yuan Wang, "A Survey of Queries Over Uncertain Data", *Knowledge and Information Systems*, Vol. 37, No. 3, pp. 485-530, 2013.

[28] Christof Tannert, Horst Dietrich Elvers and Burkhard Jandrig, "The Ethics of Uncertainty in the Light of Possible Dangers, Research becomes a Moral Duty", EMBO Rep*orts*, Vol. 8, No. 10, pp. 892-896, 2007.

[29] L. Arge and J.S. Vitter, "Optimal Dynamic Interval Management in External Memory", *Proceedings of 37th Annual Symposium on Foundations of Computer Science*, pp. 560-569, 1996.

[30] Yufei Tao and Dimitris Papadias, "Time-Parameterized Queries in Spatio-Temporal Databases", *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pp. 334-345, 2002.

[31] Mong Lee, Wynne Hsu, Christian S.Jensen, Bin Cui and Keng Lik Teo, "Supporting Frequent Updates in R-Trees: A Bottom-Up Approach", *Proceedings of the 29th International Conference on Very Large Data Bases*, Vol. 29, pp. 608-619, 2003.

[32] Reynold Cheng, Yuni Xia, Sunil Prabhakar, Rahul Shah and Jeffrey Scott Vitter, "Efficient Indexing Methods for Probabilistic Threshold Queries over Uncertain Data", *Proceedings of the 13th International Conference on Very Large Data Bases*, Vol. 30, pp. 876-887, 2004.

[33] Reynold Cheng, Sunil Prabhakar and Dmitri V. Kalashnikov, "Querying Imprecise Data in Moving Object Environments", *IEEE Transaction Knowledge and Data Engineering*, Vol. 16, No. 9, pp. 1112-1127, 2004.

[34] Christian S. Jensen, Dan Lin and Beng Chin Ooi, "Query and Update Efficient B+-Tree based Indexing of Moving Objects", *Proceedings of the 13th International Conference on Very Large Data Bases*, Vol. 30, pp.768-779, 2004.

[35] Yufei Tao, Reynold Cheng, Xiaokuai Xiao, Wang Kay Ngai, Ben Kao and Sunil Prabhakar, "Indexing Multi-Dimensional Uncertain Data with Arbitrary Probability Density Functions", *Proceedings of the 31st International Conference on Very Large Data Bases*, pp 922-933, 2005.

[36] Bin Lin and Jianwen Su, "Handling Frequent Updates of Moving Objects", *Proceedings of the 14th ACM International Conference on Information and Knowledge Management*, pp. 493-500, 2005.

[37] Victor Teixeira de Almeida and Ralf Hartmut Guting, "Indexing the Trajectories of Moving Objects in Networks", *Proceedings of the 14th ACM International Conference on Scientific and Statistical Database Management*, pp 115-118, 2004.

[38] C. Bohm, A. Pryakhin and M. Schubert, "The Gauss-Tree: Efficient object Identification in Databases of Probabilistic Feature Vectors", *Proceedings of 22nd International Conference on Data Engineering*, pp. 1-12, 2006.

[39] Xiaopeng Xiong and Walid G.Aref, "R-Trees with Update Memos" Computer Science Technical Reports, Department of Computer Science, Purdue University, 2005.

[40] Su Chen, Beng Chin Ooi, Kian-Lee Tan and Mario A. Nascimento, "ST^{2}B-tree: A Self-Tunable Spatio-Temporal B$^+$-Tree Index for Moving Objects", *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pp. 29-42, 2008.

[41] Pankaj K. Agarwal, Siu Wing Cheng, Yufei Tao and Ke Yi, "Indexing Uncertain Data", *Proceedings of the 29th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, pp. 137-146, 2009.

[42] Andrew Knight, Qi Yu and Manjeet Rege, "*Threshold Interval Indexing for Complicated Uncertain Data*", Lambert Academic Publishing, 2013.

[43] Nianbing Du, Junfeng Zhan, Ming Zhao, Daorui Xiao and Yinchuang Xie, "Spatio-Temporal Data Index Model Of Moving Objects On Fixed Networks Using Hbase", *Proceedings of IEEE International Conference on Computational Intelligence and Communication Technology*, pp. 247-251, 2015.

[44] M.S.B. PhridviRaj and C.V. Guru Rao, "Data Mining Past, Present and Future a Typical Survey on Data Streams", *Proceedings of 7th International Conference Interdisciplinary in Engineering*, Vol. 12, pp. 255-263, 2014.

[45] Zhao Hong Liu, Xiao Li Liu, Jun Wei Ge and Hae Young Bae, "Indexing Large Moving Objects from Past to Future with PCFI+-Index", *Proceedings of 11th International Conference on Management of Data*, pp. 131-137, 2005.

[46] Dan Lin, Christian S. Jensen, Beng Chin Ooi and Simonas Saltenis, "Efficient Indexing of the Historical, Present, and Future Positions of Moving Objects", *Proceedings of the 6th International Conference on Mobile Data Management*, pp. 59-66, 2005.

[47] Mindaugas Pelanis , Simonas Saltenis and Christian S.Jensen, "Indexing the Past, Present, and Anticipated Future Positions of Moving Objects", *ACM Transactions on Database Systems*, Vol. 31, No. 1, pp. 255-298, 2006.

[48] Jing Guo, Wei Guo, and Dongru Zhou "Querying about the Near Future Positions of Moving Objects", *Proceedings of the First International Multi-Symposiums on Computer and Computational Sciences*, pp. 555-559, 2006.

[49] Ying Fang, Jiaheng Cao, Yuwei Peng and Liwei Wang, "Indexing the Past, Present and Future Positions of Moving Objects on Fixed Networks", *Proceedings of the International Conference on Computer Science and Software Engineering*, Vol. 4, pp. 524-527, 2008.

[50] Dong Wook Lee, Sung Ha Baek and Hae Young Bae, "aCN-RB-tree: Update Method for Spatio Temporal Aggregation of moving Object Trajectory in Ubiquitous Environment",

Proceedings of International Conference on Computational Science and Its Applications, pp. 177-182, 2009.

[51] Menglong Huang, Peng Hul and Lanfang Xia, "A Grid Based Trajectory Indexing Method for Moving Objects on Fixed Network", *Proceedings of 18th International Conference on Geoinformatics*, pp. 1-4, 2010.

[52] Yanling Zheng, "A Fast Index Method for Moving Objects on Full Temporal Query", *Proceedings of 3rd International Conference on Computer Research and Development*, Vol. 3, pp. 205-208, 2011.

[53] Gook Pil Roh, Jong Won Roh, Seung Won Hwang and Byoung Kee Yi, "Supporting Pattern-Matching Queries over Trajectories on Road Networks", *IEEE Transactions on Knowledge and Data Engineering*, Vol. 23, No. 11, pp. 1753-1758, 2011.

[54] Ying Fang, Jiaheng Cao, Junzhou Wang, Yuwei Peng and Wei Song, "HTPR*-Tree: An Efficient Index for Moving Objects to Support Predictive Query and Partial History Query", *Proceedings of International Conference on Web Age Information Management*, pp. 26-39, 2012.

[55] Zhanyu Zhu, Qun Yang and Dechang Pi, "Past, Current and Future Positions Index of Moving Objects in Networks", *Future Control and Automation*, pp. 335-342, 2012.

[56] Ying Fang, Jiaheng Cao, Yuwei Peng, Nengcheng Chen and Lin Liu, "Efficient Indexing of the Past, Present and Future Positions of Moving Objects on Road Network", *Web-Age Information Management*, pp. 223-235, 2013.

[57] Yufei Tao, Dimitris Papadias and Jimeng Sun, "The TPR*-Tree: An Optimized Spatio-Temporal Access Method for Predictive Queries", *Proceedings of the 29th International Conference on Very Large Data Bases*, Vol. 29, pp. 790-801, 2003.

[58] Yufei Tao, Christos Faloutsos, Dimitris Papadias and Bin Liu, "Prediction and Indexing of Moving Objects with Unknown Motion Patterns", *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pp. 611-622, 2004.

[59] Meihui Zhang, Su Chen, Christian S. Jensen, Beng Chin Ooi and Zhenjie Zhang, "Effectively Indexing Uncertain Moving Objects for Predictive Queries", *Proceedings of the VLDB Endowment*, Vol. 2, No. 1, pp. 1198-1209, 2009.

[60] Manoel Castro Neto, Young Seon Jeong, Myong Kee Jeong and Lee D. Han, "Online-SVR For Short-Term Traffic Flow Prediction Under Typical And Atypical Traffic Conditions", *Expert Systems with Applications*, Vol. 36 , No. 3, pp. 6164-6173, 2009.

[61] Yutaka Yanagisawa, "Predictive Indexing for Position Data of Moving Objects in the Real World", *Transactions on Computational Science VI*, pp. 77-94, 2009.

[62] Hoyoung Jeung, Man Lung Yiu, Xiaofang Zhou and Christian S. Jensen, "Path Prediction and Predictive Range Querying in Road Network Databases", *The International Journal on Very Large Data Bases*, Vol. 19, No. 4, pp. 585-602, 2010.

[63] Wanli Min and Laura Wynter, "Real-time Road Traffic Prediction with Spatio Temporal Correlations" *Transportation Research Part C: Emerging Technologies*, Vol. 19, No. 4, pp. 606-616, 2011.

[64] Slava Kisilevich, Daniel Keim and Lior Rokach, "A GIS-based Decision Support System for Hotel Room Rate Estimation and Temporal Price Prediction: The Hotel Brokers Context", *Decision Support Systems*, Vol. 54, No. 2, pp. 1119-1133, 2013.

[65] Charu C. Aggarwal and Philip S. Yu, "A Survey Of Uncertain Data Algorithms and Applications", *IEEE Transaction on Knowledge Data Engineering*, Vol. 21, No. 5, pp. 609-623, 2008.

[66] Nilesh Dalvi and Dan Suciu, "Management of Probabilistic Data: Foundations and Challenges", *Proceedings of 26th ACM Symposium on Principles of Database Systems*, pp. 1-12, 2007.

[67] S. Borzsonyi, D. Kossmann and K. Stocker, "The Skyline Operator", *Proceedings of the 17th International Conference on Data Engineering*, pp. 421-430, 2001.

[68] Jian Pei, Bin Jiang, Xuemin Lin and Yidong Yuan, "Probabilistic Skylines on Uncertain Data", *Proceedings of the 33rd International Conference on Very Large Data Bases*, pp. 15-26, 2007.

[69] C. Bohm, F. Fiedler, A. Oswald, C. Plant and B. Wackersreuther, "Probabilistic Skyline Queries", in *Proceedings of the 18th ACM conference on Information and knowledge management*, pp. 651-660, 2009.

[70] Mohamed E. Khalefa, Mohamed F. Mokbel and Justin J. Levandoski, "Skyline Query Processing for Incomplete Data", *Proceedings of International Conference on Data Engineering*, pp. 556-565, 2008.

[71] Mikhail J.Atallah and Yinian Qi, "Computing All Skyline Probabilities for Uncertain Data", *Proceedings of the 28th ACM Symposium on Principles of Database Systems*, pp. 279-287, 2009.

[72] Evangelos Dellis and Bernhard Seeger, "Efficient Computation of Reverse Skyline Queries", *Proceedings of the 33rd International Conference on Very Large Data Bases*, pp. 291-302, 2007.

[73] Xiang Lian and Lei Chen, "Monochromatic and Bichromatic Reverse Skyline Search over Uncertain Data", *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pp. 213-226, 2008.

[74] Wenjie Zhang, Xuemin Lin, Ying Zhang, Wei Wang and Jeffrey Xu Yu, "Probabilistic Skyline Operator Over Sliding Windows", *Proceedings of International Conference on Data Engineering*, pp. 1060-1071,2009.

[75] Dongwon Kim, Hyeonseung Im and Sungwoo Park, "Computing Exact Skyline Probabilities for Uncertain Databases", *IEEE Transactions on Knowledge and Data Engineering*, Vol. 24. No. 12, pp. 2113-2126, 2012.

[76] Xuemin Lin, Ying Zhang, Wenjie Zhang and Muhammad Aamir Cheema, "Stochastic Skyline Operator", *Proceedings of International Conference on Data Engineering*, pp. 721-732, 2011.

[77] Katja Hose and Akrivi Vlachou, "A Survey of Skyline Processing in Highly Distributed Environments", *The VLDB Journal – The International Journal on Very Large Data Bases*, Vol. 21, No. 3, pp. 359-384, 2012.

[78] Xiaofeng Ding X and Hai Jin, "Efficient and Progressive Algorithms for Distributed Skyline Queries over Uncertain Data", *Proceedings of 30th International Conference on Distributed Computing Systems*, pp. 149-158, 2010.

[79] Xiaofeng Ding X and Hai Jin, "Efficient and Progressive Algorithms for Distributed Skyline Queries over Uncertain

Data", *IEEE Transactions on Knowledge and Data Engineering*, Vol. 24, No. 8, pp. 1448-1462, 2012.

[80] Xingjie Liu, De Nian Yang, Mao Ye and Wang Chien Lee, "U-Skyline: A New Skyline Query for Uncertain Databases", *IEEE Transactions on Knowledge and Data Engineering*, Vol. 25, No. 4, pp. 945-960, 2013.

[81] Ronald Fagin, "Combining Fuzzy Information from Multiple Systems", *Proceedings of 15th ACM symposium on Principles of Database* Systems, pp. 216-226, 1996.

[82] Ronald Fagin, "Fuzzy Queries in Multimedia Database Systems", *Proceedings of 17th ACM Symposium on Principles of Database Systems*, pp. 1-10, 1998.

[83] Mohamed A. Soliman, Ihab F. Ilyas and Kevin Chen Chuan Chang, "Top-k Query Processing In Uncertain Data", *Proceedings of the 23rd International Conference on Data Engineering*, pp. 896-905, 2007.

[84] Jiang Chen and Ke Yi, "Dynamic Structures for Top-k Queries on Uncertain Data", *Theoretical Computer Science*, Vol. 407, No. 1-3, pp. 310-317, 2008.

[85] Ke Yi, Feifei Li, George Kollios and Divesh Srivastava, "Efficient Processing of Top-k Queries in Uncertain Databases with X-Relations", *IEEE Transactions on Knowledge and Data Engineering*, Vol. 20, No. 12, pp. 1669-1682, 2008.

[86] Cheqing Jin, Ke Yi, Lei Chen, Jeffery Xu Yu and Xuemin Lin, "Sliding-Window Top-k Queries on Uncertain Streams", *Proceedings of the International Conference on Very Large Data Bases*, pp. 411-435, 2010.

[87] Tingjian Ge, Stan Zdonik and Samuel Madden, "Top-*k* Queries on Uncertain Data: On Score Distribution and Typical Answers", *Proceedings of the 29th ACM SIGMOD International Conference on Management of Data*, pp. 375-388, 2009.

[88] Ming Hua, Jian Pei, Wenjie Zhang and Xuemin Lin, "Ranking Queries on Uncertain Data: A Probabilistic Threshold Approach", *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pp. 673-686, 2008.

[89] Haozhe Chang, Tingting Qin, Xiaoguang Liu, Gang Wang and Airu Yin, "Top-k Queries Processing with Uncertain Data on Graphics Processing Units", *Proceedings of 4th International Symposium on Parallel Architectures, Algorithms and Programming*, pp. 310-314, 2011.

[90] Thomas Bernecker, Hanes Peter Kriegel, Nikos Mamoulis, Matthias Renz and Andreas Zuefle, "Scalable Probabilistic Similarity Ranking in Uncertain Databases", *IEEE Transactions on Knowledge and Data Engineering*, Vol. 22, No. 9, pp. 1234-1246, 2010.

[91] Tao Chen, Lei Chen, M. Tamer Ozsu and Nong Xiao, "Optimizing Multi-Top-k Queries over Uncertain Data Streams", *IEEE Transactions on Knowledge and Data Engineering*, Vol. 25, No. 8, pp. 1814-1829, 2013.

[92] Chunyang Ma, Hua Lu, Lidan Shou and Gang Chen, "KSQ: Top-k Similarity Query on Uncertain Trajectories", *IEEE Transactions on Knowledge and Data Engineering*, Vol. 25, No. 9, pp. 2049-2062, 2013.

[93] Elias Frentzos, Kostas Gratsias, Nikos Pelekis and Yannis Theodoridis, "Algorithms for Nearest Neighbor Search on Moving Object Trajectories", *GeoInformatica*, Vol. 11, No. 2, pp. 159-193, 2007.

[94] Hans-Peter Kriegel, Peter Kunath and Matthias Renz "Probabilistic Nearest-Neighbor Query on Uncertain Objects", *Proceedings of International Conference on Database Systems for Advanced Applications*, pp. 337-348, 2007.

[95] Johannes Niedermayer, et al., "Probabilistic Nearest Neighbor Queries on Uncertain Moving Object Trajectories", *Proceedings of the VLDB Endowment*, Vol. 7, No. 3, pp. 205-216, 2014.

[96] Xiang Lian and Lei Chen, "Efficient Processing of Probabilistic Reverse Nearest Neighbor Queries Over Uncertain Data", *The International Journal on Very Large Data Bases*, Vol. 18, No. 3, pp. 787-808, 2009.

[97] Goce Trajcevski, Roberto Tamassia, Hui Ding, Peter Scheuermann and Isabel F. Cruz, "Continuous Probabilistic Nearest-Neighbor Queries for Uncertain Trajectories", *Proceedings of the 12th International Conference on Extending Database Technology: Advances in Database Technology*, pp. 874-885, 2009.

[98] Sze Man Yuen, Yufei Tao, Xiaokui Xiao and Jian Pei, "Superseding Nearest Neighbor Search on Uncertain Spatial Databases", *IEEE Transactions on Knowledge and Data Engineering*, Vol. 22, No. 7, pp. 1041-1055, 2010.

[99] Jinchuan Chen Reynold, Mohamed Mokbel and Chi Yin Chow, "Scalable Processing of Snapshot And Continuous Nearest-Neighbor Queries Over One-Dimensional Uncertain Data", *The International Journal on Very Large Data Bases*, Vol. 18, No. 5, pp. 1219-1240, 2009.

[100] Sally McClean, Bryan Scotney and Mary Shapcott, "Aggregation of Imprecise and Uncertain Information Databases", *IEEE Transactions on Knowledge and Data Engineering*, Vol. 13, No. 6, pp. 902-912, 2001.

[101] Mohan Yang, Haixun Wang, Haiquan Chen and Wei Shinn Ku, "Querying Uncertain Data with Aggregate Constraints", *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pp. 817-828, 2011.

[102] Raghotham Murthy, Robert Ikeda, and Jennifer Widom, "Making Aggregation Work in Uncertain and Probabilistic Databases", *IEEE Transactions on Knowledge and Data Engineering*, Vol. 23, No. 8, pp. 1261-1273, 2011.

[103] Mao Ye, Ken C.K. Lee, Wang Chien Lee, Xingjie Liu and Meng Chang Chen, "Querying Uncertain Minimum in Wireless Sensor Networks", *IEEE Transactions on Knowledge and Data Engineering*, Vol. 24, No. 12, pp. 2274-2287, 2012.

[104] Ying Zhang, Xuemin Lin, Yufei Tao, Wenjie Zhang, and Haixun Wang, "Efficient Computation of Range Aggregates against Uncertain Location-Based Queries", *IEEE Transactions on Knowledge and Data Engineering*, Vol. 24, No. 7, pp. 1244-1258, 2012.

[105] Liang Wang, David Wai Lok Cheung Reynold Cheng, Sau Dan Lee and Xuan S. Yang, "Efficient Mining of Frequent Item Sets on Large Uncertain Databases", *IEEE Transactions on Knowledge and Data Engineering*, Vol. 24, No. 12, pp. 2170-2183, 2011.

[106] Zhaonian Zou, Jianzhong Li, Hong Gao, and Shuo Zhang, "Mining Frequent Subgraph Patterns from Uncertain Graph Data" , *IEEE Transactions on Knowledge and Data Engineering*, Vol. 22, No. 9, pp. 1203-1218, 2010.

[107] Alfredo Cuzzocrea, Carson Kai-Sang Leung, and Richard Kyle MacKinnon, "Mining Constrained Frequent Itemsets from Distributed Uncertain Data", *Future Generation Computer Systems*, Vol. 37, pp. 117-126, 2014.

[108] Mohamed F. Mokbel and Walid G. Aref, "SOLE: Scalable On-Line Execution of Continuous Queries on Spatio-Temporal Data Streams", *The VLDB Journal - The International Journal on Very Large Data Bases*, Vol. 17, No. 5, pp 971-995, 2008,

[109] Tingjian Ge, "Join Queries on Uncertain Data Semantics and Efficient Processing", *Proceedings of IEEE 27th International Conference on Data Engineering*, pp. 697-708, 2011.

[110] Xiang Lian and Lei Chen, "Similarity Join Processing on Uncertain Data Streams", *IEEE Transactions on Knowledge and Data Engineering*, Vol. 23, No. 11, pp. 1718-1734, 2011.

[111] Reza Akbarinia, Patrick Valduriez and Guillaume Verger, "Efficient Evaluation of SUM Queries over Probabilistic Data", *IEEE Transactions on Knowledge and Data Engineering*, Vol. 25, No. 4, pp. 764-775, 2012.

[112] K. Venkateswara Rao, A. Govardhan and K.V. Chalapati Rao, "Spatiotemporal Data Mining: Issues, Tasks and Applications", *International Journal of Computer Science and Engineering Survey*, Vol. 3, No. 1, pp. 39-52, 2012.