Range Search on Uncertain Trajectories

†Liming Zhan, ‡Ying Zhang, †Wenjie Zhang, †Xiaoyang Wang, †Xuemin Lin
†The University Of New South Wales ‡The University of Technology, Sydney
zhanl@cse.unsw.edu.au, Ying.Zhang@uts.edu.au
{zhangw, xiaoyangw, lxue}@cse.unsw.edu.au

ABSTRACT

The range search on trajectories is fundamental in a wide spectrum of applications such as environment monitoring and location based services. In practice, a large portion of spatio-temporal data in the above applications is generated with low sampling rate and the uncertainty arises between two subsequent observations of a moving object. To make sense of the uncertain trajectory data, it is critical to properly model the uncertainty of the trajectories and develop efficient range search algorithms on the new model. Assuming uncertain trajectories are modeled by the popular Markov Chains, in this paper we investigate the problem of range search on uncertain trajectories. In particular, we propose a general framework for range search on uncertain trajectories following the filtering-and-refinement paradigm where summaries of uncertain trajectories are constructed to facilitate the filtering process. Moreover, statistics based and partition based filtering techniques are developed to enhance the filtering capabilities. Comprehensive experiments demonstrate the effectiveness and efficiency of our new techniques.

Categories and Subject Descriptors

H.2.4 [**DATABASE MANAGEMENT**]: Systems—*Query processing*

Keywords

Uncertain Trajectory; Range Search

1. INTRODUCTION

With the rapid development of positioning technologies such as radio frequency identification (RFID), wireless sensor networks, smart-phone, radar, satellite and global positioning system (GPS), massive spatio-temporal data has been mounting up. Due to physical and resource limitations of the data collection devices, it is infeasible to continuously capture the location of a moving object (e.g., vehicle, people, animal and iceberg) for each point of time, and hence the uncertainty arises between two subsequent discrete observations. For instance, to save the energy and the communication cost, a taxi may report its location at a low frequency [23]. The time period between two check-in positions might be long in Geo-social applications such as bike routes[1] and tourist routes[2] [13]. Consequently, a large volume of spatio-temporal data with low sampling rate is described by *uncertain trajectories* where the possible locations of a moving object between two subsequent observations are captured by a time-dependent random variable (i.e., a stochastic process).

CIKM'15, October 19–23, 2015, Melbourne, Australia.
Copyright is held by the owner/author(s). Publication rights licensed to ACM.
ACM 978-1-4503-3794-6/15/10 ...$15.00.
DOI: http://dx.doi.org/10.1145/2806416.2806430 .

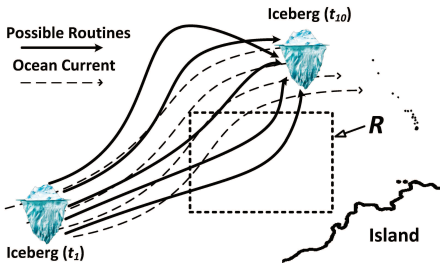


Figure 1: Motivation Example

In this paper, we investigate the problem of *range search* on uncertain trajectories, which is critical to make sense of uncertain trajectories in many key applications such as environment monitoring, location based service, traffic management, and national security. Informally, we aim to retrieve a set of moving objects (trajectories) which consistently appear within a given area with high probabilities during a particular time period. Below are two motivation examples for range search on uncertain trajectories.

Motivation Examples. In Figure 1, an iceberg is observed at times t_1 and t_{10} by satellite or radar systems, while its precise location is unknown at a time $t \in (t_1, t_{10})$ due to the resource limitation. Fortunately, according to the knowledge of nearby ocean currents as well as the historical iceberg trajectory data, an expert can derive possible locations of this iceberg based on Markov Chain model [5]. To choose a particular region for setting up an oil platform or deploying a major military exercises, we may need to evaluate to what degree a region (e.g, R in Figure 1) is affected by icebergs during a certain period in history. Since the location of an iceberg might be uncertain, we can ignore an iceberg at time t if the likelihood of this iceberg falling in the region R is smaller than a given threshold θ ($0 < \theta \leq 1$). Moreover, we may only be interested in the icebergs which consistently (say, at least η times) appear within the region with prob-

[1] http://www.bikely.com/
[2] http://www.everytrail.com/

ability at least θ. This corresponds to the range search on uncertain trajectories investigated in this paper. Similarly, in the study of wild animal migration, the range search on uncertain trajectories can help scientists to evaluate the importance of a region during a period of time, where locations of a tagged animal can be observed by wireless RFID sensors from time to time.

Challenges. Same as [5, 4, 13, 18], in this paper we assume the uncertain trajectories are described by Markov Chain model because it has solid theoretical foundation and rich applications. A straightforward approach for the problem of range search on uncertain trajectories is to calculate the *appearance probability* of each moving object o at each time within the query time interval, and count the number of times in which o appears within the search region with probability at least θ. Then an object is qualified if the number of accumulated times exceeds a duration threshold η. However, as reported in [4], the computation is still very expensive although efficient algorithm is developed in [5].

Consequently, it is desirable to follow the *filtering-and-refinement* paradigm to significantly reduce the number of candidate trajectories (i.e., moving objects) by exploiting effective filtering techniques. In particular, for any two subsequent observations of a moving object o at times t_i and t_j, denoted by $o(t_i)$ and $o(t_j)$ respectively, we aim to build a summary of the uncertain location distribution of the object. Thus, lower and upper bounds of its appearance probability can be easily derived for any time $t \in (t_i, t_j)$ when a range query is issued. Novel sub-diamonds based filtering technique is proposed in [4] to effectively support range search on uncertain trajectories. However, we observe that its performance is unsatisfactory in our empirical study, because each sub-diamond aims at bounding the appearance probability of the moving object o regarding **all** times between two subsequent observation times t_i and t_j. This motivates us to develop new filtering techniques so that the summary is constructed for a set of time intervals instead of the whole time interval (t_i, t_j) (i.e., partition along the temporal dimension). Specifically, we first introduce a simple filtering technique based on some pre-computed statistics information. Then we further enhance the filtering power by developing partition based approach to approximate the location distribution of a moving object using a set of buckets, which are generated by spatial and temporal partitions. We discuss how to effectively build the partition based summaries following some important observations.

Contributions. Our main contributions can be summarized as follows.

- We formally define the problem of range search on uncertain trajectories.
- We present a general framework for the range search on uncertain trajectories following the filtering-and-refinement paradigm, where summaries of the objects are constructed to facilitate the filtering process.
- A simple and effective filtering technique is proposed based on statistics information of the uncertain trajectories.
- A partition based filtering technique is developed to further enhance the filtering capabilities. Effective summary construction algorithm is proposed based on some important observations.
- Comprehensive experiments on real-life and synthetic datasets demonstrate the effectiveness and efficiency of our techniques.

Roadmap. The rest of the paper is organized as follows. Section 2 presents the related work in the paper. Then we formally define the problem of range search on uncertain trajectories in Section 3. Section 4 introduces a general framework for range search following the filtering and refinement paradigm. Section 5 and Section 6 propose the statistics based and partition based filtering techniques respectively. Experimental results are reported in Section 7, and Section 8 concludes the paper.

2. RELATED WORK

Recent years have witnessed the increasing amount of research on uncertain data modeling and query processing due to their importance in many applications. In this section, we briefly introduce the existing work closely related to the problem studied in this paper.

Uncertain Trajectories Modeling. A variety of models have been proposed to capture the uncertainty of the trajectory data. Early studies on uncertain trajectories employ simple geometric shapes (e.g., cylinders [16] and beads [14]) to approximate the possible locations of a moving object. Despite of its simplicity, this model suffers from an inherent drawback: the probability distribution of an object is not considered and hence cannot appropriately support probabilistic queries. In some recent work (e.g., [22]), the network-constraint model is used where the raw location of a moving object is mapped to a linear range on the road networks. In [2, 11, 17], the uncertain location of an object is captured by an independent probability density function (e.g., Gaussian distribution) at each point of time. As shown in [5, 13], the temporal dependence between two subsequent locations of an object is lost in this model. Recently, a novel evolving density model is proposed in [9] to capture the time-varying uncertainty of the moving objects where the probability density function may change over time. Markov Chain model has been widely used in the literature to capture the temporal dependency of a moving object, and hence is naturally adapted to describe the uncertainty of the trajectory data with low sampling rate in [5, 4, 13, 18]. Moreover, [5, 4] show that the Markov Chain model correctly complies with the classical possible world semantics [3]. In this paper, we employ Markov Chain model to describe the uncertain trajectories.

Range Search on Uncertain Data. Range search on uncertain data has been intensively studied in recent years. A large body of work (e.g., [15, 21]) focus on the range search on a snapshot of uncertain trajectories; that is, each object is described by a probabilistic density function, and objects with appearance probability exceeding a given threshold are retrieved. The problem of range search on uncertain trajectories has been investigated against differerent uncertain models such as cylinder model (e.g., [16]), beads model (e.g., [14]), network-constraint model (e.g., [22]), independent probabilistic density function model (e.g., [2]), evolving density model [9], segments based model [1], as well as the Markov Chain model [5, 4].

As to the best of our knowledge, [5, 4] are only two existing work which study the problem of range search on uncertain trajectories modeled by Markov Chains. Particularly, Emrich *et al.* propose efficient computation algorithm for range search in [5] without the support of indexing technique. In [4], they further improve the performance by utilizing pre-computed sub-diamonds based summaries which can significantly reduce the number of candidate objects for refinement.

Sub-diamonds based Filtering. As shown in [4], all possible valid trajectories within a segment $g(o, t_i, t_j)$ can be bounded by a diamond for each individual dimension if the maximal speed is given. Figure 2 depicts the diamond $\Diamond_1$ ($\Diamond_1 = \langle o(t_i), a, o(t_j), b\rangle$) where the horizontal axis represents the time and the vertical axis is the dimension D_1. Given a range search query, we may easily *prune* the segment g based on $\Diamond_1$. For instance, we have $P(o(t), Q_1) = 0$ for time $t \in [t_i, t_x]$ since the query region Q_1 does not overlap $\Diamond_1$ regarding the dimension D_1. Thus, g can be excluded from further computation. Similarly, we can claim $o(t)$ is enclosed by Q_2 for any $t \in [t_i, t_j]$ if Q_2 contains the diamond regarding both D_1 and D_2.

Intuitively, the filtering performance can be further enhanced by maintaining a set of sub-diamonds. For instance, $\Diamond_2$ in Figure 2 is a sub-diamond of g where $\Diamond_2 = \langle o(t_i), a, o(t_j), c\rangle$, and we know that its associated probability, denoted by $P(\Diamond_2)$, is 0.5; that is, with probability at least 0.5, $o(t)$ is bounded by $\Diamond_2$ for any $t \in [t_i, t_j]$. Consequently, the segment g can be pruned for search region Q_3 if probability threshold $\theta \geq 0.5$ because Q_3 does not overlap $\Diamond_2$. The validation of the segment can be conducted in a similar way. Together with the diamonds and their minimal bounding rectangles, the sub-diamonds of the segments are organized by an R-tree in [4], namely UST-Tree.

The sub-diamonds based filtering technique developed in [4] can significantly reduce the computational cost compared with the diamond based technique. However, as we need to enforce that the object o appears within the sub-diamond $\Diamond$ with probability at least $P(\Diamond)$ w.r.t **all** $t \in (t_i, t_j)$, this may lead to poor filtering performance. In our empirical study, we observe that the corresponding probability (i.e.,$P(\Diamond)$) of the sub-diamonds $\{\Diamond\}$ might be rather small, especially when the timespan between t_i and t_j is long. Moreover, sub-diamonds are calculated based on the projected values for each individual dimension separately. As reported in [20], this may lose the spatial correlation of the object location distribution, and hence deteriorates the filtering performance. These problems cannot be addressed by simply increasing the number of sub-diamonds.

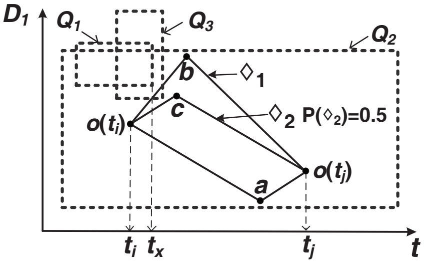


Figure 2: Sub-diamonds based Filtering

3. BACKGROUND

In this section, we formally define the problem of range search on uncertain trajectories. Table 1 summarizes the notations frequently used throughout the paper.

3.1 Capture Uncertainty by Markov Chains

In [5, 4], an uncertain trajectory is modeled as a realization of a stochastic process [10]. Markov Chains can model a discrete spatio-temporal (state-time) space with the assumption that $o(t+1)$ only depends on $o(t)$.

Definition 1 (Markov Chain model). *Given a stochastic process $o(t)$ with $t \in \mathcal{T}$ and a state $s \in \mathcal{S}$, the stochastic process is called Markov Chain iff $P(o(t+1) = s_j | o(0) = s_0, o(1) = s_1, \ldots, o(t) = s_i) = P(o(t+1) = s_j | o(t) = s_i)$.*

Notation	Definition
o ($\mathcal{O}$)	a moving object (a set of moving objects)
$\mathcal{S}$ ($\mathcal{T}$)	discrete space (time) domain
$o(t)$	location (state) of an object o at time t
q	spatio-temporal search region
R	a spatial search region
$[q.s, q.e]$	query time interval
θ	probabilistic threshold
η	duration threshold
$P(o(t), s)$	probability that $o(t)$ is located at sate s
$g(o, t_i, t_j), g$	a segment of an object o with two subsequent observations $o(t_i)$ and $o(t_j)$
$\mathcal{T}(g)$	time interval $[q.s, q.e] \cap [t_i, t_j)$
$\Delta_t(q)(\Delta_t(g))$	duration of a query (segment)
$P(o(t), R)$	probability that $o(t)$ falling in the region R
$d(o, q, \theta), d(o)$	duration (i.e., number of times) that object o satisfies the search region $q.R$
$d^-(o)$ $(d^+(o))$	lower (upper) bound of $d(o)$
$d(g)$	number of satisfied times of the object o in segment g
$d^-(g)$ $(d^+(g))$	lower (upper) bound of d(g)
$\mathcal{S}(g)$	partition based summary of segment g
$g.mbr$	minimal bounding rectangle of segment g

Table 1: The summary of notations.

For an object o moving on the space $\mathcal{S}$, we set $P_{i,j}(o) = P(o(t+1) = s_j | o(t) = s_i)$, where $P_{i,j}(o)$ represents the probability of object o moving from state s_i to s_j when the time changes from anytime time $t \in \mathcal{T}$ to its successive time $t+1$. We can store all $P_{i,j}(o)$ in a $n \times n$ matrix $M(o)$ to represent the transition probability of object o from state s_i to s_j at any time t, where n is the number of possible states (locations) and the matrix $M(o)$ is called transition matrix. Then we have $o(t+1) = o(t) \times M(o)$. Recall that $o(t)$ is the distribution vector of an object o at time t where $\sum_{s \in \mathcal{S}} P(o(t), s) = 1$. Similarly, if $M(o)^T$ is defined as a transposed Markov Chain matrix, we have $o(t) = o(t+1) \times M(o)^T$.

Given two subsequent observations $o(t_i)$ and $o(t_j)$, efficient algorithm is proposed in [5] to derive the location distribution $o(t)$ for $t \in (t_i, t_j)$ based on $M(o)$ and $M(o)^T$. Same as [5, 4], we assume objects share the same Markov Chain matrix which can be learned by domain experts in various applications.

3.2 Problem Definition

Following the common assumptions of the existing works (e.g., [5, 4, 13, 18]) which capture the uncertainty of trajectories with Markov Chain model, we assume the space and time are in discrete domain. The space $\mathcal{S}$ consists of n possible states (locations) $\{s_1, \ldots, s_n\}$ in 2-dimensional space. For a state s, $s.D_i$ denotes the coordinate value of s on i-th dimension. We use $\mathcal{T}$ to denote time domain $\{t_1, \ldots, t_m\}$. Consequently, in this paper, the trajectory of a moving object o is represented by a set of m' ($m' \leq m$) tuples $\{t_k, o(t_k)\}$, where $o(t)$ represents the state of o at time t. For a certain trajectory, $o(t)$ is a unique state $s \in \mathcal{S}$. However, it corresponds to a probability distribution when the location of the object is derived from probabilistic models. Following is a formal definition of the uncertain location for an object o at time t.

Definition 2 (Uncertain Location). *Let $P(o(t), s)$ denote the probability that an object o appears on state (loca-*

tion) s at time t. The uncertain location of an object o at time t, denoted by $o(t)$, consists of n' tuples $\{s_i, P(o(t), s_i)\}$ with $P(o(t), s_i) > 0$, where $\sum_{i=1}^{n'} P(o(t), s_i) = 1$.

Consequently, an **uncertain trajectory** is a trajectory whose location might be uncertain at each point of time. In particular, we assume the location (state) of an object is certain when it is observed (reported), while locations (states) of an object between two subsequent observation times are derived based on the Markov Chain model [4] which is introduced in Section 3.1. Therefore, the uncertain trajectory of an object o consists of a set of **segments** $\{g(o, t_i, t_j)\}$ where t_i and t_j corresponds to two subsequent observations. Each segment g records the location distribution of the object o from time t_i (inclusive) to t_j (exclusive). We use $\Delta_t(g)$ to denote the duration of the segment (a.k.a. observation interval size) where $\Delta_t(g) = t_j - t_i$.

Given a region R, we use $s \in R$ to denote that the location (state) s is within the region R. Then we define the **appearance probability** of o regarding R at time t, denoted by $P(o(t), R)$, to measure the likelihood of the object o falling in the region R at time t.

$$P(o(t), R) = \sum_{s \in R \text{ and } s \in \mathcal{S}} P(o(t), s) \qquad (1)$$

Note that we may also interpret the concept of *appearance probability* in terms of *possible world semantics*. In particular, $P(o(t), R)$ denotes the probability mass of the possible worlds at time t in which the object o falls in the region R. This also results in Equation 1.

DEFINITION 3 (SPATIO-TEMPORAL SEARCH REGION (q)). *A spatio-temporal search region q consists of three components $\langle R,\ s,\ e\rangle$, where $q.R$ represents the spatial search region, and $q.s(q.e)$ denotes the start (end) time of the query time interval. We use $\Delta_t(q)$ denote the duration of the search region (i.e., $\Delta_t(q) = q.e - q.s + 1$).*

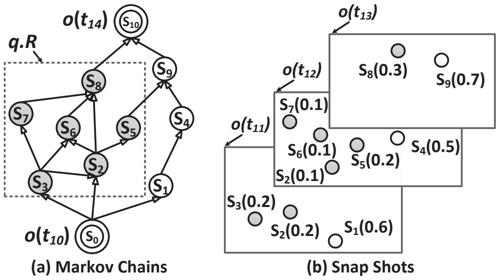
We say an object satisfies the spatio search region $q.R$ with probability at least θ at time t if $P(o(t), q.R) \geq \theta$ for $t \in [q.s, q.e]$. We use $d(o, q, \theta)$ to denote the duration (i.e., the number of times) that a moving object o satisfies $q.R$ with probability at least θ. For presentation simplicity, we use $d(o)$ to represent $d(o, q, \theta)$ whenever there is no ambiguity.

Finally we have the formal definition of the spatio-temporal range search on uncertain trajectories.

Problem Statement. In this paper, we investigate the problem of spatio-temporal range search over uncertain trajectories. Particularly, given a spatio-temporal search region q, a probabilistic threshold θ ($0 < \theta \leq 1$), a duration threshold η ($1 \leq \eta \leq \Delta_t(q)$), and uncertain trajectories of a set $\mathcal{O}$ of moving objects, we aim to identify objects $\{o|\ d(o, q, \theta) \geq \eta\}$ with $o \in \mathcal{O}$; that is, find objects which consistently (at least η times) appear within the spatial search region with probability at least θ.

Range queries on uncertain trajectories with *EXISTS* and *ALL* semantics in [4, 5] are special cases of the problem studied in this paper, which correspond to the range search with $\eta = 1$ and $\eta = \Delta_t(q)$, respectively.

EXAMPLE 1. *In Fig 3(a), $o(t_{10})$ and $o(t_{14})$ are two subsequent observations of object o; a spatio-temporal search region q is given as $\langle R, t_{11}, t_{13}\rangle$ with probabilistic threshold θ and duration threshold η. The snap shots of o are depicted in Fig 3(b) for t_{11}, t_{12} and t_{13}, where the appearance probability for each state is marked. Note that shaded states are contained by $q.R$. Then we have $P(o(t_{11}), q.R) = 0.4$, $P(o(t_{12}), q.R) = 0.5$ and $P(o(t_{13}), q.R) = 0.3$. Given $\eta = 2$, if $\theta \leq 0.4$, o meets the query constraints, otherwise o is not an answer.*



(a) Markov Chains

(b) Snap Shots

Figure 3: Range Search over Uncertain Trajectories

Thereafter of this paper, the "spatio-temporal range search" is abbreviated to "range search", and "spatio-temporal search region" are abbreviated to "search region" for presentation simplicity. We might use "location" and "state" interchangeably for better understanding of the paper.

4. FRAMEWORK

A straightforward implementation of range search on uncertain trajectories is to calculate $d(o, q, \theta)$ for each individual object $o \in \mathcal{O}$ through Markov Chains based computation technique [5]. However, as shown in [4], this is cost-prohibitive since the *refinement* cost is rather expensive. Therefore, it is desirable to develop effective and efficient *filtering* techniques to **prune** or **validate** objects such that the number of objects involving refinement can be significantly reduced. In particular, suppose we can derive the upper and lower bounds for the duration of an object o regarding the range search, denoted by $d^+(o)$ and $d^-(o)$ respectively. Then an object can be safely pruned if $d^+(o) < \eta$ or validated if $d^-(o) \geq \eta$. Moreover, for each individual time t, we can also derive lower and upper bounds of the appearance probability, denoted by $P^-(o(t), q.R)$ and $P^+(o(t), q.R)$, so we can avoid the computation of $P(o(t), q.R)$ if $P^-(o(t), q.R) \geq \theta$ (*validate*) or $P^+(o(t), q.R) < \theta$ (*prune*).

In this paper, we develop efficient algorithms to support range search on uncertain trajectories following filtering and refinement paradigm. In the sequel, we first introduce a simple minimal bounding box based filtering technique, then present a general framework for range search on uncertain trajectories.

4.1 Minimal Bounding Box Based Filtering

Given two subsequent observations $o(t_i)$ and $o(t_j)$, we may easily come up with a minimal bounding rectangle (MBR) for each segment g, denoted by *g.mbr*, which encloses all possible locations of o during time t_i and t_j if the maximal speed is pre-given. Together with the time dimension, each segment g can be enclosed by a 3-dimensional minimal bounding box (MBB), denoted by *g.mbb*. Clearly, a spatio-temporal search region q is a cube in 3-dimensional space. In this paper, we define three relations between q and *g.mbb*. We say a query q **does not overlap** a segment g if q and *g.mbb* does not overlap w.r.t spatial or temporal aspects; that is, $q.R \cap g.mbr = \emptyset$, $q.s \geq t_j$ or $q.e < t_i$. Otherwise, we say q **overlaps** g. Particularly, we say q **contains** g if g is contained by q on both spatial and temporal aspects, i.e., $g.mbr \subset q.R$, $q.s \leq t_i$, and $q.e \geq t_j$. Let $d(g)$ denote the contribution of the segment g to $d(o)$ where $0 \leq d(g) \leq \Delta_t(g)$. It is immediate that we have $d(g) = \Delta_t(g)$ if q contains g, and $d(g) = 0$ if q does not overlap g.

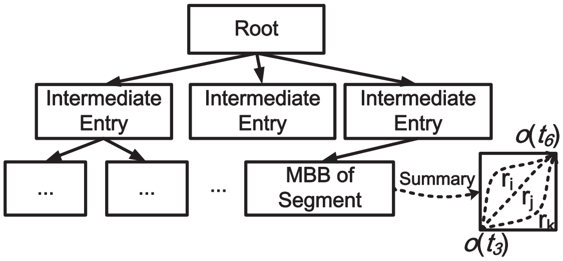


Figure 4: Segments Summaries Tree

4.2 Segments Summaries Tree (STA-tree)

MBB based filtering technique is simple and intuitive, and its filtering capability is rather limited. In Section 5 and Section 6, we introdue advanced filtering techniques based on statistics information and spatio-temporal partitions of the segment respectively where summaries of the segments are pre-computed to facilitate the filtering process.

In this paper, we assume a summary of the segment is constructed for each individual segment w.r.t the filtering technique used (e.g., sub-diamonds based filtering, statistics based filtering and partition based filtering). As shown in Figure 4, MBBs of the segments are organized by a hierarchical spatial index structure (e.g., R-tree [6]). For each segment entry, its corresponding summary is maintained to enhance the filtering performance.

With the same rationale to MBB based filtering in Section 4.1, we can easily come up with three relations between q and an intermediate entry E (i.e., segments enclosed by E). Then an intermediate entry E can be *pruned* or *validated* without further exploring its child entries.

4.3 A General Framework

Assuming the summaries of the segments are organized by an STA-tree, we present a general framework for the range search on uncertain trajectories following the filtering and refinement paradigm, and details are illustrated in Algorithm 1.

In particular, we traverse the entries in a branch and bound fashion. A FIFO queue, denoted by $\mathcal{Q}$, is used to maintain the entries to be visited. In Line 5-13, entries are processed according to their relationships with search region q. Clearly, we do not need to further explore an entry E if it does not overlap q (Line 5), because none of the segments enclosed by E contribute to the final results. On the other hand, if E is *contained* by q, we can immediately *validate* the segments $\{g\}$ enclosed by E; that is, we increase $d^-(o)$ by $\Delta_t(g)$ (Line 8) where g is a segment of the object o. Line 9 validates o if we have $d^-(o) \geq \eta$ where $\mathcal{R}$ is used to keep query results. Otherwise, i.e., E *overlaps* q but is not contained by q, Line 10-13 further explore an entry by expanding its child entries or put its corresponding segment into the set $\mathcal{F}$ which will be further processed by advanced filtering technique at Line 20.

We update $d^+(o)$ by accumulating the durations of its validated segments (Line 14) and unexplored segments, i.e., segments in $\mathcal{F}$ (Line 16). Line 17 retrieves candidate objects which are not validated but have a promising upper bound (i.e., $d^-(o) < \eta$ and $d^+(o) \geq \eta$). Then Line 18-27 further refine the candidate set by exploiting the advanced filtering techniques, in which the summary of segment may derive tighter lower and upper bounds for $d(g)$. Note that Line 21 removes a segment g from candidate segments $\mathcal{F}$ if we have $d^-(g) = \Delta_t(g)$ (i.e., o is qualified at all times $t \in [t_i, t_j)$) or $d^+(g) = 0$ (i.e., o is not qualified at any time $t \in [t_i, t_j)$).

Algorithm 1: ***Range Search***$(\mathcal{O}_T, q, \theta, \eta)$

```
   Input  : $\mathcal{O}_T$ : Uncertain trajectories of a set $\mathcal{O}$ of
            objects organized by STA-tree $T$,
            $q$ : range search, $\theta$ probabilistic threshold,
            $\eta$ : duration threshold
   Output: objects $\{o\}$ with $d(o, q, \theta) \geq \eta$
1  $\mathcal{C} := \emptyset$; $\mathcal{F} := \emptyset$; $\mathcal{R} := \emptyset$;
2  $\mathcal{Q} \leftarrow$ push root of $\mathcal{O}_T$;
3  while $\mathcal{Q} \neq \emptyset$ do
4      $E :=$ element popped from $\mathcal{Q}$;
5      if $E$ overlaps $q$ then
6          if $E$ is contained by $q$ then
7              for each segment $g \in E$ of object $o$ do
8                  $d^-(o) := d^-(o) + \Delta_t(g)$;
9                  $\mathcal{R} := \mathcal{R} \cup o$ If $d^-(o) \geq \eta$;   // validate
10         else if $E$ is an intermediate entry then
11             Push child entries of $E$ into $\mathcal{Q}$;
12         else
13             $\mathcal{F} := \mathcal{F} \cup$ corresponding segment $g$;
14 $d^+(o) = d^-(o)$ for all objects with segments in $\mathcal{F}$ ;
15 for each segment $g$ of object $o$ in $\mathcal{F}$ do
16     $d^+(o) := d^+(o) + \Delta_t(g)$;
17 $\mathcal{C} \leftarrow$ objects $\{o\}$ if $d^+(o) \geq \eta$ and $o \notin \mathcal{R}$;   // prune
18 for each candidate object $o \in \mathcal{C}$ do
19     for each candidate segment $g$ of $o$ in $\mathcal{F}$ do
20         Derive $d^-(g)$ and $d^+(g)$ from summary
           associated with $g$;
21         $\mathcal{F} := \mathcal{F} \setminus g$ If $d^-(g) = \Delta_t(g)$ or $d^+(g) = 0$;
22         $d^-(o) := d^-(o) + d^-(g)$ ;
23         $d^+(o) := d^+(o) - \Delta_t(g) + d^+(g)$ ;
24     if $d^-(o) \geq \eta$ then                          // validate
25         $\mathcal{R} := \mathcal{R} \cup o$; $\mathcal{C} := \mathcal{C} \setminus o$;
26     else if $d^+(o) < \eta$ then                        // prune
27         $\mathcal{C} := \mathcal{C} \setminus o$;
28 for each object $o$ in $\mathcal{C}$ do                 // refinement
29     $\mathcal{R} := \mathcal{R} \cup o$ If $o$ is verified;
30 return $\mathcal{R}$
```

Finally, Line 28-29 refine the remaining candidate objects by exactly computing $d(o, q, \theta)$ for each candidate object o. Note that we only need to compute $d(g)$ for candidate segments $\{g\}$ in $\mathcal{F}$.

As shown in our empirical study, the dominant cost of Algorithm 1 is the refinement cost at Line 29, since it is time consuming to calculate appearance probabilities of objects at different times. This motivates us to develop effective and efficient filtering techniques to significantly reduce the number of survived candidates with reasonable space overhead. In Section 5 and Section 6, we present advanced filtering techniques based on statistics information and spatio-temporal partitions respectively.

5. STATISTICS BASED APPROACH

In this section, we present the statistics based filtering technique. Section 5.1 introduces the motivation of the technique. Section 5.2 proposes the detailed pruning and validation rules. Performance analysis is conducted in Section 5.3.

5.1 Motivation

In this section, we develop the statistics information based filtering technique. In a nutshell, for each segment $g(o, t_i, t_j)$ we use some simple statistics to capture the uncertain location distribution of the object for each time $t \in (t_i, t_j)$. Then we can derive lower and upper bounds of the appearance probability of o at time t, denoted by $P^-(o(t), q.R)$ and $P^+(o(t), q.R)$ respectively, to *prune* or *validate* the time t. We say a time t is pruned (validated) if $P^+(o(t), q.R) < \theta$ ($P^-(o(t), q.R) \geq \theta$). Furthermore, we can merge the statistics of a set of consecutive times to reduce the summary size.

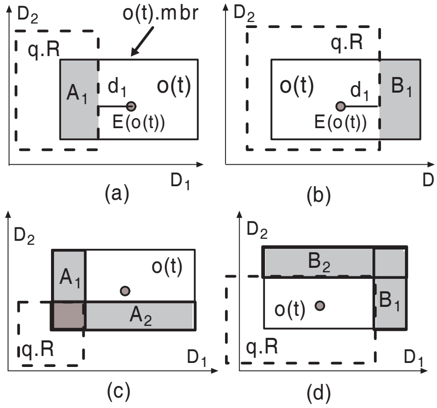


Figure 5: Motivation of Statistics based Filtering

As shown in Figure 5, suppose $o(t)$ is bounded by a minimal bounding rectangle, denoted by $o(t).mbr$. We use $P(A_i)$ ($i = 1, 2$) to denote the probability mass of the states (locations) which are *contained* by $q.R$ along the i-th dimension D_i, and $P(B_i)$ records the probability mass of the states which are *not contained* by $q.R$ along the i-th dimension. In consequence, we have $P(o(t), q.R) = P(A_1)$ in Figure 5(a), since $q.R$ contains $o(t)$ along the dimension D_2. Suppose we can derive an upper bound of $P(A_1)$, denoted by $P^+(A_1)$. Then we can safely prune the time t, if $P^+(A_1) < \theta$. Similarly, t is validated in Figure 5(b) if $1 - P^+(B_1) \geq \theta$, where $P^+(B_1)$ is the upper bound of $P(B_1)$. This observation can be easily extended to the case where the search region overlaps $o(t).mbr$ on both dimensions, e.g., Figure 5(c) and Figure 5(d). In Figure 5(c), we have $P(o(t), q.R) \leq min(P(A_1), P(A_2)) \leq min(P^+(A_1), P^+(A_2))$, and we can prune t if $min(P^+(A_1), P^+(A_2)) < \theta$. With the similar rationale, t is validated if $1 - (P^+(B_1) + P^+(B_2)) \geq \theta$.

EXAMPLE 2. *Suppose we have $P^+(A_1) = 0.1$, $P^+(B_1) = 0.1$, $P^+(A_2) = 0.2$, and $P^+(B_2) = 0.2$ in Figure 5. Then we have $P(o(t), q.R) \leq 0.1$ in Figure 5(a), $P(o(t), q.R) \geq 0.9$ in Figure 5(b), $P(o(t), q.R) \leq min(0.1, 0.2) = 0.1$ in Figure 5(c), and $P(o(t), q.R) \geq 1 - (0.1 + 0.2) = 0.7$ in Figure 5(d).*

In the next subsection, we exploit *Cantelli's inequality* [12] to derive $P^+(A_1)$, $P^+(A_2)$, $P^+(B_1)$, and $P^+(B_2)$ based on the statistics information of $o(t)$.

5.2 Statistics Based Filtering Technique

For presentation simplicity, we use a random variable X to denote the location distribution $o(t)$. Following are formal definitions of the *expectation* and the *variance* of X.

DEFINITION 4. ***Expectation**, $E(X)$. We use $E(X)$ to denote the expectation of X, where $E(X).D_i = \sum_{s \in \mathcal{S}} s.D_i \times P(X = s)$.*

Note that $P(X = s)$ denotes the probability that X resides on the state s, and $s.D_i$ is the i-th coordinate value of the state (location) s.

DEFINITION 5. ***Variance**, $\sigma^2(X)$. We use $\sigma_i^2(X)$ to denote the variance of X on each dimensions; that is, $\sigma_i^2(X) = \sum_{s \in \mathcal{S}} (s.D_i - E(X).D_i)^2 \times P(X = s)$.*

Given two values x and y where $x > 0$ and $y > 0$, we use $\delta(x, y)$ to denote a function where $\delta(x, y) = \frac{1}{1 + \frac{x^2}{y^2}}$. Following is *Cantelli's inequality* introduced in [12], which is demonstrated in Figure 6(a).

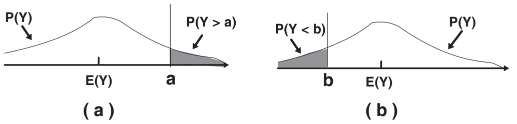


Figure 6: Example for Cantelli's Inequality

LEMMA 1 (CANTELLI'S INEQUALITY [12]). *Suppose that Y is a random variable in one dimensional space with expectation $E(Y)$ and variance $\sigma^2(Y)$, $P(Y \geq a) \leq \delta(a - E(Y), \sigma(Y))$ for any $a > E(Y)$, where $P(Y \geq a)$ denotes the probability of $Y \geq a$.*

By replacing Y with $2E(Y) - Y$, we have a variant of Lemma 1 which is illustrated in Figure 6(b).

LEMMA 2. *Suppose that Y is a random variable in one dimensional space with expectation $E(Y)$ and variance $\sigma^2(Y)$, $P(Y \leq b) \leq \delta(E(Y) - b, \sigma(Y))$ for any $b < E(Y)$, where $P(Y \leq b)$ denotes the probability of $Y \leq b$.*

We can come up with $P^+(B_1)$ in Figure 5(b) based on Lemma 1. And Lemma 2 can be used to derive $P^+(A_1)$ in Figure 5(a).

Suppose the expectation and the variance of X (i.e., location distribution $o(t)$) are readily available, we can derive the upper bound of $P(X, q.R)$ for *pruning*.

THEOREM 1. *Suppose that the search region $q.R$ overlaps $X.mbr$ but does not contain $E(X)$. We have $P^+(A_i) = 1$ if $X.mbr$ is contained by $q.R$ on i-th dimension, otherwise $P^+(A_i) = \delta(d_i, \sigma_i(X))$ where d_i denotes the distance between $q.R$ and $E(X)$ on the i-th dimension (e.g., d_1 in Figure 5(a)). Then we have $P^+(X, q.R) = min(P^+(A_1), P^+(A_2))$.*

PROOF. It is immediate that $P^+(A_i) = P(A_i) = 1$ if $X.mbr$ is contained by $q.R$ on i-th dimension according to the definition of $P(A_i)$. Otherwise, suppose $q.R$ is on the left side or bottom of $E(X)$ on i-th dimension (e.g., A_1 in Figure 5(a) and Figure 5(c), and A_2 in Figure 5(c)), then we have $P^+(A_i) = \delta(d_i, \sigma_i(X))$ according to Lemma 2. Similarly, we have $P^+(A_i) = \delta(d_i, \sigma_i(X))$ according to Lemma 1 when $q.R$ is on the right side or top of $E(X)$ on i-th dimension. Then we have $P(X, q.R)) \leq min(P(A_1), P(A_2)) \leq min(P^+(A_1), P^+(A_2))$. Thus, the theorem holds. □

With similar rationale, we have the following theorem which can obtain the lower bound of $P(X, q.R)$ for the *validation* of time t.

THEOREM 2. *Suppose that the search region $q.R$ overlaps $X.mbr$ and contains $E(X)$. We have $P^+(B_i) = 0$ if $X.mbr$ is contained by $q.R$ on i-th dimension, otherwise*

$P^+(B_i) = \delta(d_i, \sigma_i(X))$ *where* d_i *denote the distance between* $E(X)$ *and the uncovered area on i-th dimension (e.g.,* d_1 *in Figure 5(b)). Then we have* $P^-(X, q.R) = 1 - (P^+(B_1) + P^+(B_2))$.

Statistics Based Filtering. For a given segment $g(o, t_i, t_j)$, the *expectation* and *variance* information are maintained for each time $t \in (t_i, t_j)$. For a given query q, we can obtain $P^-(o(t), q.R)$ and $P^+(o(t), q.R)$ according to Theorem 1 and Theorem 2. Let $\mathcal{T}(g)$ denote the timestamps within g which satisfy query time constraints, i.e., $\mathcal{T}(g) = [t_i, t_j) \cap [q.s, q.e]$. For each timestamp $t \in \mathcal{T}(g)$, we increase $d^-(g)$ and $d^+(g)$ by one if $P^-(o(t), q.R) \geq \theta$. Otherwise, we increase $d^+(g)$ by one if $P^+(o(t), q.R) \geq \theta$.

Reduce Summary Size. To reduce the space consumption, we may keep the statistics information for a set of consecutive times $\{t_k, t_{k+1}, \ldots, t_l\}$ within a segment. Then instead of keeping the expectation and the variance information for each individual time, we maintain the minimal bounding rectangle of their expectations as well as the maximal variance value on each dimension. Then Theorem 1 and Theorem 2 can be adopted in a conservative way such that any time $t \in \{t_k, t_{k+1}, \ldots, t_l\}$ can be *pruned* or *validated* at the same time. We omit the details due to the space limitation.

5.3 Performance Analysis

Given a segment $g(o, t_i, t_j)$, we assume the location distributions of the objects for time $t \in [t_i, t_j)$ are readily available by applying Markov Chains based techniques in Section 3.1. The construction of the statistics based summary can be finished in $\mathcal{O}(n_s \times \Delta_t(g))$ time where n_s is the average number of tuples in $o(t)$. Regarding the filtering cost, it takes $\mathcal{O}(1)$ time for each time and hence the total cost is $\mathcal{O}(\Delta_t(g))$.

6. PARTITION BASED APPROACH

In this section, we introduce a new filtering approach to build summary for a segment based on both spatial and temporal partitions. Specifically, Section 6.1 introduces the motivation of the partition based filtering approach. Details of the technique are presented in Section 6.2. We discuss how to effectively construct partition based summary in Section 6.3.

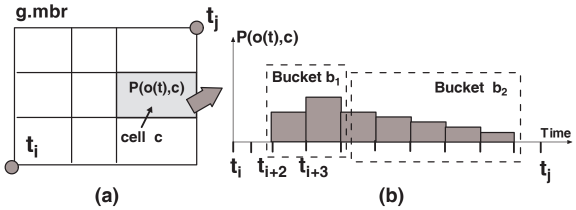


Figure 7: Motivation of Partition based Filtering

6.1 Motivation

Although statistics based filtering technique is simple and effective, a considerable number of segments will still survive from the filtering phase in our empirical study because it is difficult to precisely capture a distribution with a few statistics. This motivates us to develop more sophisticated summary technique by partitioning along both spatial and temporal dimensions, so that the filtering performance can be significantly enhanced with a reasonable space overhead.

As shown in Figure 7(a), we first partition the minimal bounding rectangle of the segment ($g.mbr$) into a set $\mathcal{S}(g)$ of cells. Then for each cell c, Figure 7(b) shows that there is a function $P(o(t), c)$ which presents the appearance probability of the object o within the cell c for times $t \in (t_i, t_j)$. We can immediately derive $P^+(o(t), q.R)$ and $P^-(o(t), q.R)$ as follows.

$$P^-(o(t), q.R) = \sum_{c \in \mathcal{S}(g) \ \wedge \ q.R \text{ contains } c} P(o(t), c) \qquad (2)$$

$$P^+(o(t), q.R) = \sum_{c \in \mathcal{S}(g) \ \wedge \ q.R \text{ overlaps } c} P(o(t), c) \qquad (3)$$

It is infeasible to explicitly keep $P(o(t), c)$ values for all $t \in (t_i, t_j)$. Consequently, we employ a set of buckets to approximate the appearance probability distribution for each cell c. For instance, in Figure 7(b) we use two buckets $\{b_1, b_2\}$ to represent $P(o(t), c)$ where the time interval size, maximal and minimal appearance probabilities of a bucket b are denoted by $\Delta_t(b)$, $b.p^+$ and $b.p^-$ respectively. In particular, we have $P^-(o(t), c) = b(t, c).p^-$ and $P^+(o(t), c) = b(t, c).p^+$ where $b(t, c)$ is the corresponding bucket of c at time t. Then, we have

$$P^-(o(t), q) = \sum_{c \in \mathcal{S}(g) \ \wedge \ q.R \text{ contains } c} b(t, c).p^- \qquad (4)$$

$$P^+(o(t), q) = \sum_{c \in \mathcal{S}(g) \ \wedge \ q.R \text{ overlaps } c} b(t, c).p^+ \qquad (5)$$

6.2 Partition Based Filtering

Given a partition based summary of a segment $g(o, t_i, t_j)$, denoted by $\mathcal{S}(g)$, we can come up with an effective computation algorithm to derive lower and upper bounds for $d(g)$ according to Equations 4 and 5. Algorithm 2 illustrates the details. Specifically, we first retrieve cells in $\mathcal{S}(g)$ which are contained by $q.R$ or overlap $q.R$, denoted by $\mathcal{C}_\cap$ and $\mathcal{C}_\cup$, respectively. Then we attempt to *validate* (Line 6-9) or *prune* (Line 11-14) a time $t \in \mathcal{T}(g)$. Line 7 calculates the lower bound of $P(o(t), q.R)$ based on Equation 4, while Line 12 derives the upper bound according to Equation 5. For the given probability threshold θ, we can *validate* a time t if $P^-(o(t), q.R) \geq \theta$ (Line 8). Similarly, t is *pruned* if $P^+(o(t), q.R) < \theta$ (Line 13).

Algorithm 2: Partition based Filter($\mathcal{S}(g)$, q, θ)

```
   Input  : $\mathcal{S}(g)$ : partition based summary of $g$,
            $q$ : range search, $\theta$ probabilistic threshold
   Output: $d^-(g)$ and $d^+(g)$
1  $d^-(g) := 0$; $d^+(g) := 0$;
2  $\mathcal{C}_\cap \leftarrow$ cells in $\mathcal{S}(g)$ which are contained by $q.R$ ;
3  $\mathcal{C}_\cup \leftarrow$ cells in $\mathcal{S}(g)$ which overlap $q.R$ ;
4  for each time $t \in \mathcal{T}(g)$ do
5      $P^-(o(t), q.R) := 0$; $P^+(o(t), q.R) := 0$;
6      for each cell $c$ in $\mathcal{C}_\cap$ do
7          $P^-(o(t), q.R) := P^-(o(t), q.R) + b(t, c).p^-$;
8      if $P^-(o(t), q.R) \geq \theta$ then                /* validate */
9          $d^-(g) := d^-(g) + 1$; $d^+(g) := d^+(g) + 1$ ;
10     else
11         for each cell $c$ in $\mathcal{C}_\cup$ do
12             $P^+(o(t), q.R) := P^+(o(t), q.R) + b(t, c).p^+$;
13         if $P^+(o(t), q.R) \geq \theta$ then            /* prune */
14             $d^+(g) := d^+(g) + 1$;
15 return $d^-(g)$ and $d^+(g)$
```

Time Complexity. Let C_r denote the cost to retrieve the cells which are contained by $q.R$ or overlaps $q.R$, and there are n_c cells in $\mathcal{C}_\cup$. Then the total filtering cost is $O(C_r + n_c \times \Delta_t(g))$.

6.3 Summary Construction

To effective construct $\mathcal{S}(g)$, we aim to address following three issues in this subsection: *i)* how to generate buckets for a given cell; *ii)* the number of cells assigned for each segment; *iii)* how to generate the cells.

(**i**) **Bucket generation**. As discussed in Section 6.1, we cannot afford to keep $P(o(t), c)$ values for all times in $[t_i, t_j)$. Thus, we use B buckets to approximate the probability distribution. Suppose each time t have the same chance to be involved in the filtering phase, the uncertainty introduced by a bucket partition $\mathcal{B}$, denoted by $C(\mathcal{B})$, is as follows.

$$C(\mathcal{B}) = \sum_{t \in (t_i, t_j)} (b(t,c).p^+ - b(t,c).p^-) \qquad (6)$$

Recall that $b(t,c)$ is the bucket used for the time t. Similar to V-optimal histogram [8], we come up with the optimal bucket partition $\mathcal{B}^*$ with cost $O(\Delta_t(g)^2 \times B)$ by applying dynamic programming technique.

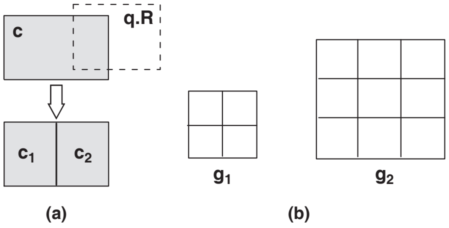


Figure 8: Motivation of Resource Allocation

In our implementation, the number of buckets for each cell is linear to the duration of the segment; that is, $B = \Delta_t(g)/l$ where l is a pre-given constant.

(**ii**) **Number of cells assigned for each segment**. Given two segments g_1 and g_2, we use $Area(g_1)$ ($Area(g_2)$) to denote the area of $g_1.mbr$ ($g_2.mbr$), and n_1 (n_2) denote the number of cells assigned to g_1 (g_2). At the first glance, we may assign a fixed number of cells to each segment (i.e., $n_1 = n_2$) or fix the area of each cell (i.e., $\frac{n_1}{n_2} = \frac{Area(g_1)}{Area(g_2)}$). However, both strategies are not cost-effective according to our observation below.

With similar spirit to [21], we use $P(c) \times W(c)$ to measure the contribution of uncertainty for a cell c, where $P(c)$ is the probability that c overlaps $q.R$ but $q.R$ doesn't contain c, while $W(c)$ denotes the probability mass within the cell (i.e., $W(c) = \sum_{t \in (t_i, t_j)} P(o(t), c)$). As shown in Figure 8(a), we can reduce the uncertainty by evenly partitioning a cell c into two parts c_1 and c_2. Assume the probability mass in c is also evenly distributed, now the overall uncertainty cost becomes $2 \times (\frac{P(c)}{2} \times \frac{W(c)}{2}) = \frac{P(c) \times W(c)}{2}$. In Figure 8(b), we assume $W(g_1) = W(g_2)$ (i.e., two segments have the same duration length), and two cells from the same segment have the same area and probability mass. Intuitive, for a cost-effective resource allocation strategy, each cell should contribute the same amount of uncertainty; that is, $\frac{P(g_1)}{n_1} \times \frac{W(g_1)}{n_1} = \frac{P(g_2)}{n_2} \times \frac{W(g_2)}{n_2}$. Consequently, we have

$$(\frac{n_1}{n_2})^2 = \frac{Area(g_1)}{Area(g_2)} \qquad (7)$$

since $\frac{P(g_1)}{P(g_2)} \approx \frac{Area(g_1)}{Area(g_2)}$. For instance, suppose we have $Area(g_1)$ = 10 and $Area(g_2) = 40$ in Figure 8(b), if 4 cells are assigned to g_1, then 8 cells should goes to g_2 because $(\frac{4}{8})^2 = \frac{10}{40}$.

(**iii**) **Cell generation**. Now we investigate how to partition the minimal bounding rectangle of a segment g into $m_1 \times m_2$ cells. Following the above argument, the uncertainty cost of a segment g is $\sum_{c \in \mathcal{S}(g)} P(c) \times W(c)$. Because $\sum_{c \in \mathcal{S}(g)} Area(c)$ and $\sum_{c \in \mathcal{S}(g)} W(c)$ are two constants regardless how the cells are generated. This implies that we should have the same $Area(c) \times W(c)$ value for each cell in $\mathcal{S}(g)$ to minimize the uncertainty cost according to Chebyshev Sum Inequality [7]. Nevertheless, it is infeasible to achieve this with a regular grid partition, and hence we resort to a simple heuristic. In particular, for a segment with $m_1 \times m_2$ cells we partition $g.mbr$ into m_i parts with the same probability mass along each dimension i, which can be finished in time $\mathcal{O}(n_s \times \Delta_t(g) + n_s \log(n_s))$ where n_s is the average number of tuples in $o(t)$ for $t \in (t_i, t_j)$.

7. EXPERIMENT

In this section, we present results of a comprehensive performance study to evaluate the efficiency and scalability of the proposed techniques in the paper. Following algorithms are evaluated.

- **UST:** The range search techniques proposed in [4] where sub-diamonds based filtering technique is employed[3].
- **STA:** Algorithm 1 in Section 4 where statistics based filtering technique (Section 5) is employed.
- **GRID:** Algorithm 1 in Section 4 where partition based filtering technique (Section 6) is employed.

In this work, we use techniques proposed in [5] to perform candidate refinement for the above three algorithms.

Datasets. We evaluate our techniques on both synthetic and real datasets using data generator from [4, 13] with following steps. We construct a two dimensions state space, consisting of n states, which are *uniformly* distributed in the domain $[0,1]^2$. For each state, we randomly choose several neighbors, and then assign random probability to each connection such that the total probability equals to 1. This builds a directed graph where the vertices represent the states and the edges represent the transition probabilities from one state to another. The graph is stored in a matrix as the transition matrix. To create an uncertain trajectory, we randomly choose one state as the start point, and then apply a directed random walk through the nonzero edges of the graph to get a moving sequence with 100 time steps. The size of time domain is set to 1,000, and the start time of an uncertain trajectory is randomly chosen between [1,900]. The observations of an object are randomly chosen from the moving sequence. In the experiment, we generate 10,000 states with a transition matrix. The number of trajectories N varies from 2,500 to 10,000 with default value 5,000. Two subsequent observations' interval size (i.e., **segment duration**) is randomly chosen from 10 to 15 by default. The probabilistic threshold θ varies from 0.1 to 1.0 with default value 0.5, and the duration threshold η varies from 1 to 10 with default value 6. The real datasets are generated from a set of trajectories of taxis in the city of Beijing [19]. We apply techniques in [13] to get the state set, transition matrix, and trajectories set, and then randomly choose 10,000 state with 583 corresponding trajectories to perform the experiment.

Workload. A workload for the range query consists of 1000 queries in our experiments. The center of a query is uniformly chosen from the domain $[0,1]^2$, its start time is randomly generated from [1,990]. The query extent, i.e, search

[3] The range search with *exists* semantics (i.e., $\eta = 1$) is investigated in [4]. Nevertheless, their technique can be easily extended to support range search with $\eta > 1$.

region of a query in each dimension varies from 0.05 to 0.25 with default value 0.1, and the duration of a query ($\Delta_t(q)$) varies from 10 to 25 with default value 10.

Same as [4], the catalog size of the UST-Tree is set to 10 in the experiments. In STA Algorithm, we compress the statistics information for every 3 consecutive times. Regarding GRID Algorithm, suppose one cell is assigned to a unit area with size 0.03×0.03. Then for each segment g with area $Area(g.mbr)$, c_n cells are assigned where $c_n = \lceil \sqrt{\frac{Area(g.mbr)}{0.03 \times 0.03}} \rceil$ according to Equation 7. Moreover, $\lceil \frac{\Delta_t(g)}{5} \rceil$ buckets are generated for each cell where $\Delta_t(g)$ is the duration of the segment (i.e., observation interval size).

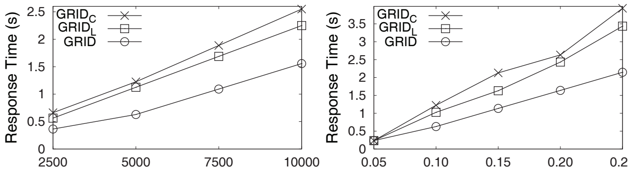
All algorithms proposed in this paper are implemented in standard C++ with STL library and compiled with GNU GCC. Experiments are run on a PC with Intel Xeon 2.40GHz dual CPU and 4G memory running Debian Linux. The disk page size is fixed to $4,096$ bytes. As the refinement phase contributes the dominant cost in three algorithms, we evaluate their performance by measuring the average number of candidate segments refined. The average query response time is also recorded to evaluate the efficiency of the algorithms.

Table 2 lists all parameters which may have impacts on our performance study, where default values are in **bold** font. In our experiments, all parameters use default values unless otherwise specified.

Notation	Definition
number of trajectories (N)	2500, **5000**, 7500, 10000
segment duration $\Delta_t(g)$	[$\mathbf{10, 15}$], $[15, 20]$, $[20, 25]$, $[25, 30]$
probabilistic threshold (θ)	0.1, 0.3, **0.5**, 0.7, 0.9, 1.0
duration threshold (η)	1, 4, **6**, 8, 10
query extent (area of $q.R$)	0.05, **0.1**, 0.15, 0.20, 0.25
query duration $\Delta_t(q)$	**10**, 15, 20, 25

Table 2: Parameter Settings

7.1 Performance Tuning



(a) *# Trajectories* (b) *Query Extent*

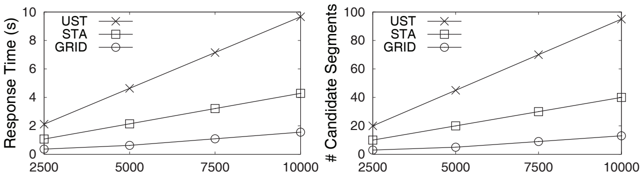
Figure 9: Performance Tuning

To evaluate effectiveness of the adaptive resource allocation strategy in Section 6.3, we also construct partition based summaries following other two alternative strategies, namely GRID$_C$ and GRID$_L$ respectively. Particularly, GRID$_C$ always allocates 3×3 cells for each segment (i.e., fix the number of cells for each segment), while each cell in GRID$_L$ has the area 0.03×0.03 (i.e., fixed the area of each cell).

Note that summaries constructed in three algorithms have similar summary size under default settings. Nevertheless, Figure 9 shows that GRID always outperforms the other two competitors by varying the number of trajectories and query extent. This confirms the effectiveness of our adaptive resource allocation strategy.

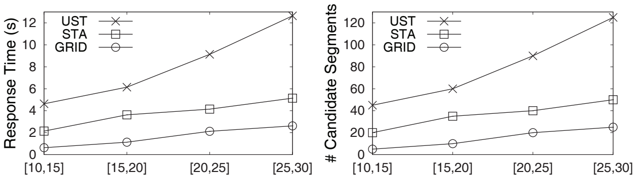
7.2 Performance Evaluation

Impact of the number of trajectories. Figure 10 evaluates the performance of three algorithms on synthetic dataset where the number of trajectories N grows from $2,500$ to $10,000$. With a larger number of trajectories, more trajectories are involved in the computation, thus incurring higher computation cost and more candidate segments. The response time and the number of candidate segments of STA and GRID grow slowly, yet the performance of UST drops more quickly with the growth of the number of trajectories. It is shown that GRID has the best scalability among three algorithms.



(a) *Response Time* (b) *# Candidate Segments*

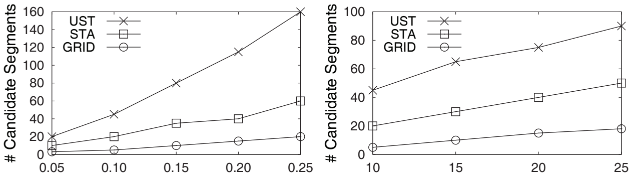
Figure 10: Impact of # Trajectories



(a) *Response Time* (b) *# Candidate Segments*

Figure 11: Impact of Segment Duration

Impact of segment duration. Figure 11 evaluates the impact of the segment duration $\Delta_t(g)$ on three algorithms where $\Delta_t(g)$ is randomly chosen from each bounded interval. The response time and the number of candidate segments are reported for three algorithms. As expected, the performance of UST degrades quickly because it is difficult to find a proper sub-diamond when the segment duration grows. Recall that a sub-diamond $\diamond$ on a segment $g(o, t_i, t_j)$ need to enforce that the appearance probability of $o(t)$ is bounded by $P(\diamond)$ regarding *all* times $t \in [t_i, t_j)$. On the contrary, STA and GRID are much less sensitive to the growth of the segment duration because of the temporal partition; that is, we build summaries for a set of time intervals in g, instead of the whole time interval.



(a) *Diff. Query Extent* (b) *Diff. Query Duration*

Figure 12: Impact of Query Extent and Duration

Impact of query extent and duration. We evaluate the impact of the query extent of $q.R$ as well as the query duration $\Delta_t(q)$ against three algorithms, where the query extent grows from 0.05 to 0.25, and the query duration varies from 10 to 25. With the grow of query extent and query duration, more trajectories are involved in the range search, thus the number of candidate segments increases as expected. Figure 12 shows that GRID has the best filtering capability among three algorithms.

Impact of probabilistic threshold. Figure 13 investigates the performance of three algorithms as a function of the probabilistic threshold θ which varies from 0.1 to 1. The performance of three algorithms is not sensitive to θ. It is

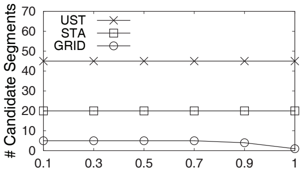


Figure 13: Diff. θ

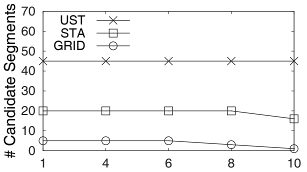
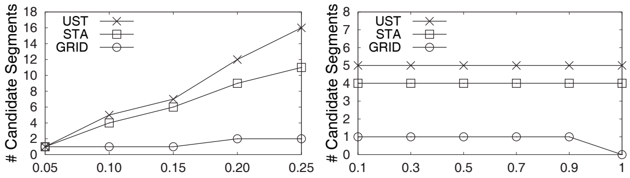


Figure 14: Diff. η

shown that GRID always has the best performance among three algorithms.

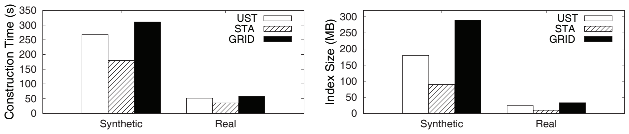
Impact of duration threshold. Figure 14 reports the number of candidate segments of the algorithms as a function of the duration threshold η which varies from 1 to 10. It is shown that the growth of η does not noticeably affect performance of three algorithms, while GRID always achieves the best performance under all settings. Recall that, when η equal 1, the range search exactly corresponds to the range search with *exists* semantics studied in [4].



(a) *Query Extent* (b) *Diff. θ*

Figure 15: Experiments on Real Data

Real data. We also perform experiments on the real-life dataset. Figure 15 reports the number of candidate segments against the growth of the query extent and probabilistic threshold θ. Similar trends are observed in Figure 15 compared with the experiments on the synthetic data.



(a) *Construction Time* (b) *Index Size*

Figure 16: Index Construction

Index construction. Figure 16 reports the index construction time and the index size of three algorithms on synthetic data and real-life data. It is interesting that STA outperforms UST in term of query response time, while STA also consumes less index size and index construction time. On the other hand, although GRID has the largest index size and construction time, it is cost effective considering its superior performance compared with other two algorithms.

8. CONCLUSION

To tame the uncertainty of trajectory data collected in a wide spectrum of applications, we have developed effective filtering and query processing techniques to support range search on uncertain trajectories which are modeled by Markov Chains. Particularly, we formally define the problem of range search on uncertain trajectories. Then we introduce an indexing structure where the summaries of the segments are organized by an STA-tree, as well as a general framework to support range search on uncertain trajectories following the filtering and refinement paradigm. To enhance the filtering capabilities, we develop effective statistics based and partition based filtering techniques. Our comprehensive experiments demonstrate the superior performance of our new techniques compared with existing work.

Acknowledgments. Ying Zhang is supported by ARC DE140100679 and DP130103245. Wenjie Zhang is supported by ARC DE120102144 and DP120104168. Xuemin Lin is supported by NSFC61232006, NSFC61021004, ARC DP120104168 and DP110102937.

9. REFERENCES

[1] L. Chen, Y. Tang, M. Lv, and G. Chen. Partition-based range query for uncertain trajectories in road networks. *GeoInformatica*, 19(1):61–84, 2015.

[2] R. Cheng, D. V. Kalashnikov, and S. Prabhakar. Querying imprecise data in moving object environments. *IEEE Trans. Knowl. Data Eng.*, 16(9):1112–1127, 2004.

[3] N. N. Dalvi and D. Suciu. Efficient query evaluation on probabilistic databases. *VLDB J.*, 16(4):523–544, 2007.

[4] T. Emrich, H.-P. Kriegel, N. Mamoulis, M. Renz, and A. Züfle. Indexing uncertain spatio-temporal data. In *CIKM*, pages 395–404, 2012.

[5] T. Emrich, H.-P. Kriegel, N. Mamoulis, M. Renz, and A. Züfle. Querying uncertain spatio-temporal data. In *ICDE*, pages 354–365, 2012.

[6] A. Guttman. R-trees: A dynamic index structure for spatial searching. In *SIGMOD Conference*, pages 47–57, 1984.

[7] G. Hardy, J. Littlewood, and G. Pólya. *Inequalities.* Cambridge University Press, 1988.

[8] H. V. Jagadish, N. Koudas, S. Muthukrishnan, V. Poosala, K. C. Sevcik, and T. Suel. Optimal histograms with quality guarantees. In *VLDB*, pages 275–286, 1998.

[9] H. Jeung, H. Lu, S. Sathe, and M. L. Yiu. Managing evolving uncertainty in trajectory databases. *IEEE Trans. Knowl. Data Eng.*, Accppted in 2013.

[10] S. Karlin and H. Taylor. *A First Course in Stochastic Processes.* Academic Press, 1975.

[11] C. Ma, H. Lu, L. Shou, and G. Chen. Ksq: Top-$(k)$ similarity query on uncertain trajectories. *IEEE Trans. Knowl. Data Eng.*, 25(9):2049–2062, 2013.

[12] R. Meester. *A Natural Introduction to Probability Theory.* 2004.

[13] J. Niedermayer, A. Züfle, T. Emrich, M. Renz, N. Mamoulis, L. Chen, and H.-P. Kriegel. Probabilistic nearest neighbor queries on uncertain moving object trajectories. *PVLDB*, 7(3):205–216, 2013.

[14] D. Pfoser and C. S. Jensen. Capturing the uncertainty of moving-object representations. In *SSD*, pages 111–132, 1999.

[15] Y. Tao, X. Xiao, and R. Cheng. Range search on multidimensional uncertain data. *ACM Trans. Database Syst.*, 32(3):15, 2007.

[16] G. Trajcevski, O. Wolfson, K. Hinrichs, and S. Chamberlain. Managing uncertainty in moving objects databases. *ACM Trans. Database Syst.*, 29(3):463–507, 2004.

[17] X. Xie, M. L. Yiu, R. Cheng, and H. Lu. Scalable evaluation of trajectory queries over imprecise location data. *IEEE Trans. Knowl. Data Eng.*, Accppted in 2013.

[18] C. Xu, Y. Gu, L. Chen, J. Qiao, and G. Yu. Interval reverse nearest neighbor queries on uncertain data with markov correlations. In *ICDE*, pages 170–181, 2013.

[19] J. Yuan, Y. Zheng, X. Xie, and G. Sun. Driving with knowledge from the physical world. In *KDD*, pages 316–324, 2011.

[20] Y. Zhang, X. Lin, W. Zhang, J. Wang, and Q. Lin. Effectively indexing the uncertain space. *IEEE Trans. Knowl. Data Eng.*, 22(9):1247–1261, 2010.

[21] Y. Zhang, W. Zhang, Q. Lin, and X. Lin. Effectively indexing the multi-dimensional uncertain objects for range searching. In *EDBT*, pages 504–515, 2012.

[22] K. Zheng, G. Trajcevski, X. Zhou, and P. Scheuermann. Probabilistic range queries for uncertain trajectories on road networks. In *EDBT*, pages 283–294, 2011.

[23] K. Zheng, Y. Zheng, X. Xie, and X. Zhou. Reducing uncertainty of low-sampling-rate trajectories. In *ICDE*, pages 1144–1155, 2012.