Interval Reverse Nearest Neighbor Queries on Uncertain Data with Markov Correlations

Chuanfei Xu [#1], Yu Gu [#1], Lei Chen [*2], Jianzhong Qiao [#1], Ge Yu [#1]

[#]*Dept. of Information Science and Engineering*
Northeastern University, China
[1]{xuchuanfei,guyu,qiaojianzhong,yuge}@ise.neu.edu.cn
[*]*Dept. of Computer Science and Engineering*
Hong Kong University of Science and Technology, Hong Kong, China
[2]leichen@cse.ust.hk

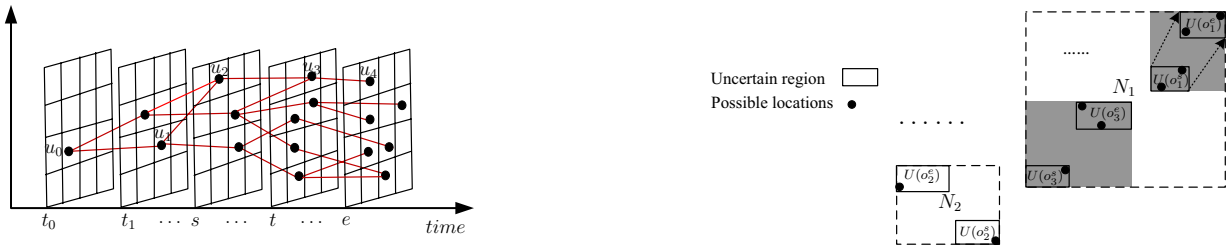
***Abstract*— Nowadays, many applications return to the user a set of results that take the query as their nearest neighbor, which are commonly expressed through reverse nearest neighbor (RNN) queries. When considering moving objects, users would like to find objects that appear in the RNN result set for a period of time in some real-world applications such as collaboration recommendation and anti-tracking. In this work, we formally define the problem of interval reverse nearest neighbor (IRNN) queries over moving objects, which return the objects that maintain nearest neighboring relations to the moving query objects for the longest time in the given interval. Location uncertainty of moving data objects and moving query objects is inherent in various domains, and we investigate objects that exhibit Markov correlations, that is, each object's location is only correlated with its own location at previous timestamp while being independent of other objects. There exists the efficiency challenge for answering IRNN queries on uncertain moving objects with Markov correlations since we have to retrieve not only all the possible locations of each object at current time but also its historically possible locations. To speed up the query processing, we present a general framework for answering IRNN queries on uncertain moving objects with Markov correlations in two phases. In the first phase, we apply space pruning and probability pruning techniques, which reduce the search space significantly. In the second phase, we verify whether each unpruned object is an IRNN of the query object. During this phase, we propose an approach termed Probability Decomposition Verification (PDV) algorithm which avoid computing the probability of any object being an RNN of the query object exactly and thus improve the efficiency of verification. The performance of the proposed algorithm is demonstrated by extensive experiments on synthetic and real datasets, and the experimental results show that our algorithm is more efficient than the Monte-Carlo based approximate algorithm.**

I. Introduction

Due to the increasing need for real time information, management of moving objects has received considerable attention in daily applications such as location based services [1], profile-based management [2], *etc*. In many real-world applications, because of limitation of measuring equipment or privacy protection, the trajectories of moving objects are incomplete, (*i.e.*, we have not observed locations of these moving objects at some points of time). For instance, the International Ice Patrol (IIP) stores the recorded locations of known icebergs in the maritime community. In IIP databases, the location of an iceberg at time t ($t \in \mathcal{T}$) may have the missed observation [3]. To analyze the iceberg activity, we have to infer the movements of icebergs in the time domain $\mathcal{T}$. Specifically, we can infer (or predict) the possible locations of each moving object at any point of time called a timestamp [4] with the help of stochastic models. In this paper, we assume a discrete space domain $\mathcal{R}$, where $\mathcal{R}$ denotes a finite set of locations. Each location represents a basic observation region (*e.g.*, a room). This location is defined by the center coordinate of the observed region in which any object can be located.

Aiming at inferring the possible locations of objects at each timestamp, we need to take the correlations into account. As described in [5], [6], [7], [3], lots of spatio-temporal data exhibit Markov correlations (*i.e.*, an object's location at a timestamp depends on its location at the previous timestamp and the locations of any two distinct objects are independent of each other). Take Figure 1(a) as an example to illustrate the trajectory of a moving object o_i. Here the last observed location of o_i is the location u_0 at time t_0, and the following locations of o_i are missed. According to expert knowledge or historical movements, we are able to build the Markov model to infer the possible locations of o_i in following timestamps and the transition probabilities. Each transition probability denotes the likelihood that the object moves from a location at the current timestamp to another location at the next timestamp (*e.g.*, the transition probability of o_i moving from u_0 to u_1 is represented by a conditional probability $\Pr(o_i^t = u_1 | o_i^{(t-1)} = u_0)$). All the conditional probabilities of moving objects are stored in *conditional probability tables* (CPTs).

Recently, reverse nearest neighbor (RNN) queries have received considerable attention [8], [9], [10], [11] from spatial database research community. Given a query object q and a set of data objects $\mathcal{D}$, RNN queries wish to find data objects that have the query object q as their nearest neighbor from $\mathcal{D}$. In many scenarios, the locations of both data objects and the query object are continuously varied during a time interval (*e.g.*, an hour), and at nearby timestamps the locations of each moving object are often correlated. In order to find potential relations between the data objects and the query object q, we need not only RNNs of q at a point of time but also the objects that maintain reverse nearest neighboring relations to q in the interval. The reason is that the objects that can be RNNs of q



(a) Moving objects with Markov correlations

(b) An illustration of TPR-tree entries

Fig. 1. Querying on Markov correlated data

for a long period are more useful. We consider a real scenario that may benefit from such interval query as follows.

In collaboration recommendation applications such as free riding or outdoor Counter-Strike (CS) activities, each player wears a sensor or an RFID tag, which is used to locate the player's locations. One player would like to invite other players who are always near to him/her to found a group or execute some task together. For the player who issues the RNN query, other candidate players may accept the invitation only if their nearest neighbor is the query player in a long period of time. As the results of existing RNN queries only reflect the reverse nearest neighboring relations at specific moment, in order to explore the constant reverse nearest neighboring relations between candidate players and the query player, we need to accumulate the RNN query results in a past period of time. This query request motivates a novel RNN query that returns results maintaining reverse nearest neighboring relations to the query player for the longest time.

As mentioned earlier, due to miss reading, the locations of moving objects (*e.g.*, players) are lost at some timestamps. In order to infer the possibly lost locations of objects and the relative transition probabilities more exactly, we find results from these imprecise data with the help of the Markov model. Besides, many other applications face similar decision making problems. *E.g.*, in the monitoring iceberg activity, it is significant to find icebergs that are possible to be nearest to a known iceberg during a period of time. To find most dangerous icebergs, we desire a query to retrieve the icebergs which have maintained reverse nearest neighboring relations to this iceberg for the longest time. The current work of RNN queries [8], [9], [12], [13], [14], however, study spatial objects in the time point case, and it is impossible to extend these works to returning results of RNN queries during a period of time because the locations of an object at different timestamps are not independent. Aiming to solve the above problem, we study a novel kind of RNN queries called interval RNN (IRNN) queries on data with Markov correlations. Given a time interval Δ, IRNN queries can retrieve objects that have kept being RNNs of the query object for the longest time in Δ.

Using the Markov model, the inferred locations of any object exist Markov correlations. For an object's location, we infer multiple possible locations at the next timestamp. Hence, there are exponential number of possible trajectories of this object. Because the inferred location of a moving object at a timestamp is not an exact one, this object is regarded as an uncertain moving object which is associated with a probability distribution over the possible locations at each timestamp. As an illustrating example, the possible trajectories of o_1 are shown in Figure 1(a), where $u_0u_1u_2u_3u_4$ denotes one possible trajectory of o_1. To handle the possible trajectories efficiently, we use TPR-tree [15] to index uncertain moving objects. The TPR-tree is essentially an extension of the R-tree by incorporating the movements of moving objects. Specifically, each TPR-tree node is described by a minimum bounding rectangle (MBR). At time $t \in [s, e]$, the MBR's edges are chosen so that the enclosed entries, be the moving objects or child TPR-tree nodes, remain inside the MBR. We define each leaf node as an uncertain region in which all possible locations of each object are bounded. As illustrated in Figure 1(b), the uncertain regions of uncertain moving objects o_1, o_2 or o_3 are denoted by $U(o_i^t)$ at time t ($1 \leq i \leq 3$). From time s to time e, the MBR of a TPR-tree node is guaranteed to bound all the objects in it. Due to uncertainty of data, we need to compute the probability that each object is an RNN of q at time t. Given a time interval $\Delta = [s, e]$, IRNN queries find each object that maintains nearest neighboring relations to q with the highest probability for the longest time (*i.e.*, accumulating the most timestamps) in the time interval Δ.

This work focuses on IRNN queries on uncertain data with Markov correlations. There are two major challenges to answer IRNN queries efficiently: (i) due to uncertainty of moving objects, we need to take into consideration not only the nearest neighboring relations between the possible locations of each moving object and the possible locations of the query object, but also the nearest neighboring relations between each moving object and other nearby objects; (ii) due to Markov correlations, we must consider not only the locations of each moving object at the current timestamp but also the locations at previous timestamps. Therefore, the time cost of the query over Markov correlated data is the exponential complexity, so that the query evaluation raises serious practicality concerns for the applicability in realistic settings. Aiming to solve this problem, we propose a general framework to answer IRNN queries on uncertain objects with Markov correlations efficiently. In detail, our contributions are summarized as follows.

- We formally define a novel type of interval RNN (IRNN) queries on uncertain moving objects with Markov correlations, which return objects that can maintain the nearest

neighboring relations to the query object for the longest time in the given time interval.
- We propose effective space pruning techniques which can reduce search space for uncertain moving objects. To further filter uncertain moving objects, we also present a probability pruning rule for improving the efficiency of IRNN queries.
- We present an efficient algorithm which verifies unpruned objects through gradually estimating and comparing the probability that any object is an RNN of the query object instead of computing the probability exactly.
- We evaluate the effectiveness and the efficiency of our proposed methods through extensive experiments with real and synthetic datasets, and the proposed methods can reduce the requirements of time by up to an order of magnitude than that of the Monte-Carlo based approximate algorithm.

The rest of the paper is organized as follows. Section 2 provides the related work. Section 3 formally defines IRNN queries on Markov correlated objects. Section 4 proposes some pruning rules for our queries. Section 5 verifies each unpruned object to return the answers of our queries. Section 6 gives the experimental results. Section 7 concludes this paper.

II. Related Work

A. Reverse Nearest Neighbor Query

There is a large volume of previous works [8], [16], [11], [12], [10] devoted to snapshot reverse nearest neighbor query. Specially, in [8], Korn *et al*. provide the first solution to the RNN problem. In [9] and [17], Lin *et al*. and Yang *et al*. further improve the performance of [8]. Moreover, Stanoi *et al*. [18] present a pruning method to partition the whole space into six equal regions. In each region, only the nearest neighbor of q can possibly be the RNN and they are selected as the candidate objects. Also, Tao *et al*. [10] develop TPL algorithms to retrieve candidates RNNs by half-plane pruning, followed by executing the refinement.

For uncertain objects, Lian *et al*. [19] first study the Probabilistic Reverse Nearest Neighbor (PRNN) query in uncertain databases. They propose a GP pruning method, which discards objects that have probabilities never greater than some threshold. GP method can significantly reduce the PRNN search space without introducing false dismissals. Furthermore, Cheema *et al*. [4] formalize the probabilistic reverse nearest neighbor queries based on the possible worlds semantics. Several effective pruning rules are proposed to significantly reduce the overall computation time. Bernecker *et al*. [20] develop a general framework for probabilistic RNN queries on uncertain data. They propose a pruning mechanism which takes distance dependencies into account. In addition, Emrich *et al*. [3] present a framework for efficiently modeling and querying uncertain spatio-temporal data. The key idea of the approach is to model possible object trajectories by stochastic processes.

On the other hand, there is a large body of research work on RNN query over moving object databases. Benetics *et al*. [21] first present the RNN monitoring over moving object. They assume that the velocities of the objects are known. Šaltenis *et al*. [15] propose the Time Parameterized R-tree (TPR-tree) which are used to index objects with future (predictive) trajectories. Given objects movement trajectories, a time-parameterized query [22] retrieves the current result set and the validity period of the result. Other researches do not make any assumption on object movement patterns. In addition, Xia *et al*. [23] define the monitoring region for the continuous RNN query based on the six-regions approach. Kang *et al*. [24] propose continuous monitoring RNN algorithm for both monochromatic and bichromatic RNN queries. Besides, Cheema *et al*. [25] propose a safe region approach to solve the problem of continuous monitor RNN. Wu *et al*. [26] issue RkNN queries in a circle region. To monitor the results, they continuously monitor the region around it that contains RNNs.

However, both these RNN query methods and continuous RNN query methods are not suitable for querying over correlated data so that they cannot apply to our problem.

B. Spatial-Temporal Query on Markov Correlated Data

Next, we summarize research work on answering query over Markov correlated data. Christopher Rè *et al*. [5] propose a framework for answering event queries over Markov data streams. They define the Markov model for data streams and propose some efficient methods to achieve events query. Also, Soliman *et al*. [6] present a new probabilistic model based on partial orders, and formulate several ranking query types. The sampling techniques are designed based on Markov chains to compute approximate query answers. In [27], Emrich *et al*. focus on studying queries on spatio-temporal objects with Markov correlations. This work presents a framework for efficiently modeling and querying uncertain spatio-temporal data. In addition, Lian *et al*. [28] define the local correlations for uncertain data, and investigate a classical spatial query on uncertain data with local correlations. To enable the fast query processing, in [29], the MC sampling approach is proposed, which samples paths of each object and outputs the fraction of the sampled paths which fulfill the query predicate. However, all these works do not consider handling the query processing in a given time interval. To the best of our knowledge, this is the first paper that aims to study interval queries on uncertain moving objects with Markov correlations.

III. PRELIMINARIES

In this section, we first model uncertain moving objects with Markov correlations. Then we formally define IRNN queries on uncertain moving objects with Markov correlations. Finally, we develop the general framework for answering IRNN queries efficiently. Table I summarizes the commonly-used symbols in this paper.

A. Interval RNN Definition

In this work, given a time domain $\mathcal{T}$ ($\mathcal{T} = \mathbb{N}_0^+$), we assume that an uncertain database $\mathcal{D}$ consists of n uncertain

TABLE I

COMMONLY USED SYMBOLS

Symbol	Description
$\mathcal{D}$	uncertain database
o_i	i-th uncertain moving object with Markov correlations
o_i^t	object o_i at time t
q^t	query object at time t
$\mathcal{C}$	candidate set
$\mathcal{T}$	time domain
Δ	time interval (subset of $\mathcal{T}$)
s	start time of time interval
e	end time of time interval
t	a time timestamp in time interval

moving objects $\{o_1, \ldots, o_i, \ldots, o_n\}$, where i-th object o_i $(1 \leq i \leq n)$ is associated with an uncertain object trajectory which denotes that o_i moves in the time domain $\mathcal{T}$. For any point of time $t \in \mathcal{T}$, o_i^t denotes an uncertain moving object o_i at time t, which is represented by a probability distribution over mutually exclusive locations. Similarly, given an IRNN query, the query object q is associated with an uncertain object trajectory as well. We define minimum bounding rectangles (MBRs) to bound uncertain moving objects and the query object, in which the possible locations of each object at one timestamp is located within its own uncertainty region of this object at time t ($U(o_i^t)$ as shown in Figure 1(b)). For a time interval $\Delta = [s, e]$, the IRNN query can take RNNs of q^t at each time $t \in \Delta$ ($s \leq t \leq e$) into consideration.

Many existing works [21] [24] for spatial-temporal query assume that the locations of each moving object are independent. In many cases, however, the probability distribution of any uncertain moving object at the current timestamp is related with that of this object at the previous timestamp. To model uncertain object trajectories, we have to infer the location distributions of uncertain moving objects according to their location information at previous timestamps in the time interval $\Delta = [s, e]$. In this paper, we study the case that the locations of each object are Markov correlated and the locations of any two distinct objects are independent of each other. The Markov correlations of objects in the uncertain database $\mathcal{D}$ are summarized as follows.

Definition 1: (Markov Correlated Uncertain Data Model) In the uncertain database $\mathcal{D}$, each uncertain moving object $o_i \in \mathcal{D}$ is associated with an uncertain object trajectory in which the locations of this object have Markov correlations if and only if

$$\begin{aligned} &\Pr(o_i^t = u | o_i^{(t-1)} = u_k, \ldots, o_i^1 = u_{(k-t)}) \\ &= \Pr(o_i^t = u | o_i^{(t-1)} = u_{(k-1)}), \end{aligned} \tag{1}$$

where u_k, …, $u_{(k-t)}$ denote the possible locations of o_i at all timestamps in the time domain $\mathcal{T}$.

As mentioned in the above definition, we require all the conditional probabilities (*e.g.*, $\Pr(o_i^t = u | o_i^{(t-1)} = u_{(k-1)})$) to represent distributions for uncertain moving objects at time $t \in \mathcal{T}$, which describes the likelihood that an object is located at each possible location. As illustrated in Table II, the conditional probabilities of uncertain moving objects are stored in *conditional probability tables* (CPTs).

Due to Markov correlations, the trajectory of each uncertain moving object is represented by a Markov chain [6], on which queries are answered. For example, in Table II, when $o_1^1 = (1, 2)$ (*i.e.*, o_1 is located at (1,2) at t=1), o_1 will appear at one of two possible locations (3,2) and (3,3) at t=2. Moreover, for different locations, the following locations at which o_1 will appear may be different. Hence, the possible locations of o_1 depends on the possible locations at the previous time. Next, we formally define interval reverse nearest neighbor queries.

Definition 2: (Interval Reverse Nearest Neighbor Query on Markov Correlated Data) Given an uncertain database $\mathcal{D}$ with Markov correlated uncertain moving objects, a time interval $\Delta = [s, e]$ and a Markov correlated uncertain query object q, the probability that any object o_i is an RNN of q at time t is computed as

$$\Pr_{\text{IRNN}}(q^t, o_i^t) = \int_{r_1}^{r_2} \Pr(dist(q^t, o_i^t) = r) \cdot \Pr(\wedge_{i \neq j} dist(o_j^t, o_i^t) > r) dr, \tag{2}$$

where o_j denotes any other object, and r_1 and r_2 denote the lower and upper bounds of the distance between q and o_i respectively. The IRNN query $IRNN(q, \Delta, \mathcal{D})$ is defined as

$$IRNN(q, \Delta, \mathcal{D}) = \{o_i | \forall o_j \in \mathcal{D} \wedge i \neq j, count(o_i) \leq count(o_j)\}, \tag{3}$$

where $count(o_i)$ is equal to the number of timestamps when $\Pr_{\text{IRNN}}(q^t, o_j^t) \leq \Pr_{\text{IRNN}}(q^t, o_i^t)$ for $\forall o_j (i \neq j)$ and $t \in \Delta$ ($\Delta = [s, e]$).

TABLE II

CONDITIONAL PROBABILITY TABLES

Symbol	Conditional probability
o_1^1	$<(1,2), 0.8>, <(2,2), 0.2>$
o_1^2	$<(3,2)\|(1,2), 0.6>, <(3,3)\|(1,2), 0.4>,$ $<(3,2)\|(2,2), 0.7>, <(2,3)\|(2,2), 0.3>$
o_1^3	…
…	…
o_2^1	$<(0,0), 0.9>, <(0,1), 0.1>$
o_2^2	$<(1,2)\|(0,0), 0.8>, <(2,2)\|(0,0), 0.2>,$ $<(3,2)\|(0,1), 0.7>, <(2,3)\|(0,1), 0.3>$
o_2^3	…
…	…
q^1	$<(1,1), 0.6>, <(2,1), 0.4>$
q^2	$<(2,2)\|(1,1), 0.6>, <(1,2)\|(1,1), 0.4>,$ $<(2,2)\|(2,1), 0.7>, <(3,2)\|(2,1), 0.3>$
q^3	…

According to the definition of IRNN query, we accumulate the timestamps when the probabilities that objects are RNNs of q are largest, and return the objects that have the most accumulating timestamps. Therefore, the objects which maintain the nearest neighboring relations to the given query objects in the most timestamps (*i.e.*, the value of the function $count(.)$ is largest) are retrieved. To answer the IRNN query, we need to compute the probability that each object is an RNN of the query object at each timestamp. For all the objects, the probabilities are evaluated by retrieving CPTs. For any object o_i, we must evaluate the value of $\Pr_{\text{IRNN}}(q^t, o_i^t)$ by Eq.(2). For

locations u and v, when $o_i^t = u$, $q^t = v$, and $r = dist(u, v)$,

$$\Pr(dist(q^t, o_i^t) = r) = \Pr(q^t = v) \cdot \Pr(o_i^t = u). \quad (4)$$

It follows that

$\Pr(o_i^t = u) = \sum_{k=1}^{K} \Pr(o_i^t = u | o_i^{t-1} = u_k) \cdot \Pr(o_i^{t-1} = u_k)$, where K denotes the number of transition probabilities to u. Likewise, $\Pr(o_i^{t-1} = u_k)$ can be evaluated by the complete probability formula as well. Consequently, we can calculate the value of $\Pr(o_i^t = u)$ using the complete probability formula iteratively. Similar to the evaluation $\Pr(o_i^t = u)$, the probability of the query object q being at each possible location is computed as follows:

$\Pr(q^t = v) = \sum_{m=1}^{M} \Pr(q^t = u | q^{t-1} = v_m) \cdot \Pr(q^{t-1} = v_m)$.

Using the complete probability formula iteratively, we can compute the value of $\Pr(dist(q^t, o_i^t) = r)$. All conditional probabilities can be searched in the CPTs. Likewise, given the value of r, $\Pr(dist(q^t, o_j^t) > r)$ can also be calculated as

$$\Pr(dist(o_j^t, o_i^t) > r) = \sum_{dist(w,u) > r} \Pr(o_j^t = w) \cdot \Pr(o_i^t = u), \quad (5)$$

where the probability is the sum of likelihood for $dist(o_j, o_i) > r$ at time t. Thereby, the probability that any object o_i is an RNN of q at this time can be computed as

$$\begin{aligned} &\Pr_{\text{IRNN}}(q^t, o_i^t) \\ &= \int_{r_1}^{r_2} \Pr(dist(q^t, o_i^t) = r) \cdot \Pr(\textstyle\bigwedge_{i \neq j} dist(o_j^t, o_i^t) > r) dr \\ &= \textstyle\sum_{dist(w,u) > dist(v,u)} (\Pr(o_j^t = w) \cdot \Pr(q^t = v) \cdot \Pr(o_i^t = u)). \end{aligned} \quad (6)$$

According to Eq. (6), we compute the probability that o_i is an RNN of q at time t by accumulating the likelihood that $dist(o_i^t, q^t)$ is less than $dist(o_j^t, o_i^t)$ for $\forall o_j (i \neq j)$. For the example in Table II, we only consider o_1 and o_2 and suppose that the interval is equal to $\Delta = [1, 3]$. For the first timestamp, $dist(q^1, o_1^1) = 1$ or $\sqrt{2}$, $dist(q^1, o_2^1) = \sqrt{2}$ or $\sqrt{5}$ and $dist(o_2^1, o_1^1) = \sqrt{2}$, 2 or $\sqrt{5}$. Consequently, the probability that o_1 is an RNN of q is larger than the probability that other objects are RNNs of q. At t=2, $\Pr(o_1^2 = (3, 2)) = 0.6 \times 0.8 + 0.7 \times 0.2 = 0.62$, $\Pr(o_1^2 = (3, 3)) = 0.8 \times 0.4 = 0.32$ and $\Pr(o_1^2 = (2, 3)) = 0.3 \times 0.2 = 0.06$. Also, $\Pr(o_2^2 = (1, 2)) = 0.8 \times 0.9 = 0.72$, $\Pr(o_2^2 = (2, 3)) = 0.3 \times 0.1 = 0.03$, $\Pr(o_2^2 = (3, 2)) = 0.7 \times 0.1 = 0.07$ and $\Pr(o_2^2 = (2, 2)) = 0.2 \times 0.9 = 0.18$. For the query object q, $\Pr(q^2 = (2, 2)) = 0.6 \times 0.6 + 0.7 \times 0.4 = 0.64$, $\Pr(q^2 = (3, 2)) = 0.3 \times 0.4 = 0.12$ and $\Pr(q^2 = (1, 2)) = 0.4 \times 0.6 = 0.24$. Using Eq. (6) to calculate the probabilities, $\Pr_{\text{IRNN}}(q^2, o_1^2) < \Pr_{\text{IRNN}}(q^2, o_2^2)$. Thus, the probability of o_2 being an RNN of q is larger than that of o_1 at t=2. Likewise, we can evaluate that the probability of o_1 being an RNN of q is larger at t=3. Since o_1 is the RNN of q at two timestamps (t=1 and 3) and o_2 is the RNN of q at only one timestamp (t=2), o_1 is the final result of the IRNN query.

Clearly, the possible locations of each uncertain moving object need to be considered with increasing time. Furthermore, it is complex not only to consider the nearest neighboring relations between any object and q, but also to evaluate these relations between this object and other nearby objects. Thus, the time cost of directly answering IRNN queries requires exponential time. In the next section, we propose a general framework to solve our query problems efficiently.

B. Query Framework

The time cost is extremely expensive in the IRNN query processing, since we must take into consideration the possible locations of the query object and nearby objects at each timestamp, and for each possible location we need to compute the probability of an object appearing at this location. To improve the efficiency, we develop a general framework which first filters the uncertain database $\mathcal{D}$ and then verifies whether each unpruned object is a result of the IRNN query. In

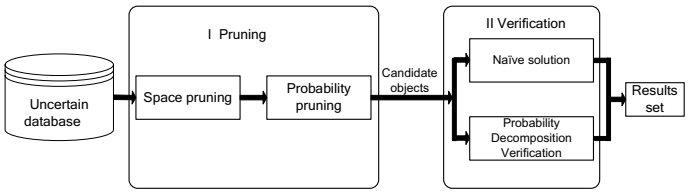


Fig. 2. The approximate query framework

detail, we highlight our pruning-verification framework for answering IRNN queries on uncertain moving objects with Markov correlations, which is illustrated in Figure 2.

- **Pruning phase.** In this phase, we first present space pruning strategies to prune uncertain moving objects that are not answers of our queries with spatial information. Then we propose probability pruning approaches to further filter objects effectively. If the objects are not RNNs of the query object q at some timestamps, they are pruned at these timestamps.
- **Verification phase.** In the verification phase, we propose two approaches to compute the probability of each candidate object (*i.e.*, an object at each timestamp when it is unpruned by our pruning rules) being an RNN of q at time $t \in \Delta$, and return the object that maintains the nearest neighboring relations to q in the most timestamps.

Given an IRNN query $IRNN(q, \Delta, \mathcal{D})$, the proposed framework first filters objects in $\mathcal{D}$ at time t ($t \in \Delta$) when the possible locations of q are far from the possible locations of these objects, and it further prunes objects by evaluating the bounds of the probability of each object being RNN of q at time t (described in Section IV). Then the framework efficiently verifies each candidate object and returns the result of the IRNN query (described in Section V).

IV. PRUNING RULES

Although the pruning strategies for traditional RNN queries have been well studied [8], [9], [10], [11], it is non-trivial to devise pruning strategies for the IRNN query processing on Markov correlated data. In this section, we first discuss space pruning for our queries on uncertain objects with Markov correlations. To further improve the query efficiency, a probabilistic pruning rule is proposed as well.

A. Space Pruning

If we use every possible location of a filtering object at each timestamp to perform bisector pruning [24], it will incur a huge cost for the IRNN query processing due to large number of possible locations during time Δ. Instead, we devise non-trivial generalization of bisector pruning for MBRs which can bound the movements of uncertain moving objects in the Δ. Next, we propose space pruning rules to find objects that are not the results of the IRNN query at each timestamp.

Lemma 1: Let U_1, U_2, U_3, U_4 and Q be MBRs in which moving objects o_1, o_2, o_3, o_4 and the query object q have been located during time Δ respectively (*p.s.*, U_1, U_2, U_3 and U_4 are located in different quadrant of the whole space, as shown in Figure 3). Then, at time t $t \in \Delta$, objects, which are definitely located outside the region generated by bisectors between vertices of Q and vertices of U_1, U_2, U_3, and U_4, are not results of IRNN query at this timestamp.

Proof: As shown in Figure 3, for time $\forall t \in \Delta$, the uncertainty region of any uncertain moving object o_i is completely located in the shadow region which is generated by bisectors between vertices of Q and vertices of U_1, U_2, U_3, and U_4. Hence, this object is located at the shadow region at time t. Since at this time the location of o_i is outside the bisectors, the minimum distance between o_i and o_1, o_2, o_3 or o_4 is not more than the distance between o_i and q at time t. According to the definition of IRNN query, there is no likelihood that o_i is an RNN of q at this timestamp. Thus, o_i is not a result of IRNN query at time t, and o_i^t can be filtered. ■

For the above lemma, we first find four MBRs (from four quadrants) in which four objects are bounded in Δ respectively. At time t, we can filter objects (*e.g.*, o_i^t in Figure 3) which are completely located in the shadow region so that the search region of the IRNN query is only the region generated by bisectors between vertices of Q and vertices of four MBRs. Note that any object (*e.g.*, o_j in Figure 3) can be pruned when it is located in the shadow region for all the timestamps during the interval Δ.

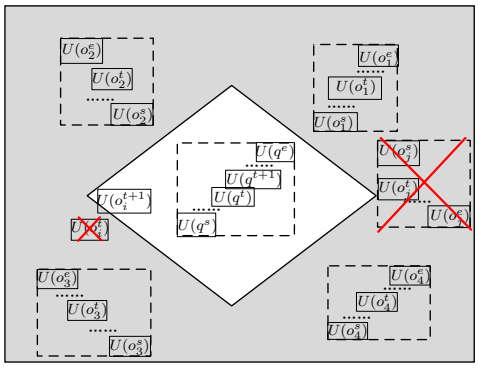


Fig. 3. Proof sketch of Lemma 1

Since both data objects and the query object are dynamic, the objects are bounded by nodes of a TPR-tree. Facing this case, we present a space pruning rule that uses the space relations to prune objects in a TPR-tree node.

Lemma 2: Let N_j be a TPR-tree node, $dist^{\perp}(o_i^t, N_j)$ be the lower bound of the distance between q and N_j and $dist^{\top}(q^t, o_i^t)$ be the upper bound of the distance between q and o_i at time t. If $dist^{\top}(q^t, o_i^t) < dist^{\perp}(o_i^t, N_j)$, then all the objects bounded by N_j cannot be RNN of q at time t. If o_j is located in N_j during time Δ, o_j can be pruned safely.

Proof: According to the condition of the lemma, if the uncertainty region $U(o_j)$ is bounded by N_j (*i.e.*, the shadow region shown in Figure 4) at time t, $dist^{\top}(q^t, o_i^t) < dist^{\perp}(o_i^t, N_j) \leq dist^{\perp}(o_i^t, o_j^t)$. Since the probability that an object o_j is an RNN of q can be computed by Eq.(2), this probability is equal to zero. Therefore, at time t, o_j is not RNN of q. Because o_j is located in N_j for all the timestamps in Δ, o_j cannot be a result of the IRNN query at each timestamp in the interval Δ. It follows that o_j can be pruned. ■

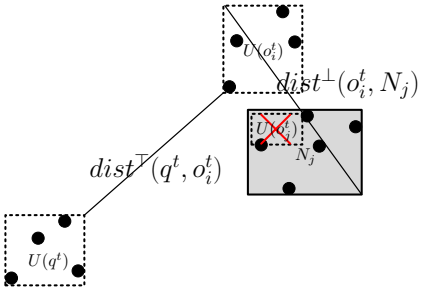


Fig. 4. Proof sketch of Lemma 2

As mentioned in Lemma 2, we can filter objects located in the MBR of a TPR-tree node, so long as the bounds of distances between objects and the query object satisfy the above pruning rule. As shown in Figure 4, o_i located in the region N_j at time t is not an RNN of the query object q according to Lemma 2. Thus, we do not need to calculate the probability of o_i^t being located at any location.

B. Probabilistic Pruning

As mentioned previously, the space pruning rules can filter any object o_i^t that is not an RNN of the query object q at time t. Using these strategies, we save the evaluation time for the probabilities of o_i^t being located at any location. To further improve efficiency of the IRNN query processing, we consider the case where exists o_j whose probability to be an RNN of q is more than that of o_i. In the case, o_i can be pruned safely. We first analyze the bounds of probabilities that an object is located at any possible location at time t as follows.

Lemma 3: Let $\Pr(o_i^t = u)$ be the probability that an uncertain moving object o_i is located at u at time t ($t \in \Delta$). Then, the upper bound of this probability is computed as

$$\Pr^{\top}(o_i^t = u) = max_{k=1}^{K}\{\Pr(o_i^t = u | o_i^{t-1} = u_k)\}, \quad (7)$$

and the lower bound of this probability is computed as

$$\Pr^{\perp}(o_i^t = u) = min_{k=1}^{K}\{\Pr(o_i^t = u | o_i^{t-1} = u_k)\}, \quad (8)$$

where $\Pr(o_i^t = v | o_i^{t-1} = u_k)$ denotes the k-th transition probability that o_i moves from the location u_k to the location u.

Proof: For any time $t \in \Delta$, the object o_i can be located at K locations. At the next timestamp, o_i must move from K locations at time t to u. Thereby, $\sum_{k=1}^{K} \Pr(o_i^t = u_k) = 1$. Because $\Pr(o_i^t = u) = \sum_{k=1}^{K} \Pr(o_i^t = u | o_i^{t-1} = u_k) \cdot \Pr(o_i^t = u_k)$, $\Pr(o_i^t = u) \leq max_{k=1}^{K}\{\Pr(o_i^t = u | o_i^{t-1} = u_k)\} \cdot \sum_{k=1}^{K} \Pr(o_i^{t-1} = u_k)$. It follows that $\Pr(o_i^t = u) \leq max_{k=1}^{K}\{\Pr(o_i^t = u | o_i^{t-1} = u_k)\}$. Thus, the upper bound of the probability that o_i is located at u at time t is $max_{k=1}^{K}\{\Pr(o_i^t = u | o_i^{t-1} = u_k)\}$. Likewise, the lower bound is $min_{k=1}^{K}\{\Pr(o_i^t = u | o_i^{t-1} = u_k)\}$. ■

When K (*i.e.*, the number of transition probabilities to u) is larger, the upper and lower bounds of the probability that any object is located at a possible location at a timestamp are closer. In this case, we are able to obtain the tight bounds. Using these bounds, we can calculate the bounds of the probability that this object is an RNN of the query object q at this time. For the object o_i, the lower bound is calculated as

$$\Pr^{\perp}_{\text{IRNN}}(q^t, o_i^t) = \sum_{dist(w,u)>dist(v,u)} (\Pr^{\perp}(o_j^t = w) \cdot \Pr^{\perp}(q^t = v) \cdot \Pr^{\perp}(o_i^t = u)), \quad (9)$$

and the upper bound is calculated as

$$\Pr^{\top}_{\text{IRNN}}(q^t, o_i^t) = \sum_{dist(w,u)>dist(v,u)} (\Pr^{\top}(o_j^t = w) \cdot \Pr^{\top}(q^t = v) \cdot \Pr^{\top}(o_i^t = u)). \quad (10)$$

According to the bounds, we summarize a probability pruning rule in the following lemma.

Lemma 4: Let o_i and o_j be any two uncertain moving objects. If $\Pr^{\top}_{\text{IRNN}}(q^t, o_i^t) \leq \Pr^{\perp}_{\text{IRNN}}(q^t, o_j^t)$ at time t, then o_i^t can be pruned safely.

Proof: Since $\Pr^{\top}_{\text{IRNN}}(q^t, o_i^t) \leq \Pr^{\perp}_{\text{IRNN}}(q^t, o_j^t)$, the uncertain moving object o_i has no likelihood to be an RNN of the query object q at time t according to the definition of IRNN query. Therefore, o_i can be pruned at this time safely. ■

As mentioned in the previous lemma, we use probability information to further prune objects. According to the proposed space and probability pruning rules, many objects can be filtered at each timestamp effectively and the unpruned objects generate the candidate set $\mathcal{C}$ of the IRNN query. The key steps of our pruning rules are illustrated in Algorithm 1. This algorithm first filters the uncertain database $\mathcal{D}$ and then takes any unpruned object at time t (*e.g.*, o_i^t) into the candidate set $\mathcal{C}$. In the next section, we will verify these candidate objects in the verification phase.

Algorithm 1: Space and Probability Pruning Algorithm

Input : uncertain database $\mathcal{D}$
Output: candidate set $\mathcal{C}$

```
1  t ⟵ s
2  find the region R generated by bisectors in Lemma 1
3  while t ≤ e do
4      for each object o_i in D do
5          if U(o_i) is non-overlapping R then
6              delete o_i^t
7          for each other object o_j in D do
8              if dist^⊤(q^t, o_j^t) < dist^⊥(o_i^t, o_j^t) then
9                  delete o_i^t
10             if Pr^⊤_IRNN(q^t, o_i^t) ≤ Pr^⊥_IRNN(q^t, o_j^t) then
11                 delete o_i^t
12         else
13             take o_i^t into C
14     t ⟵ t + 1
```

V. EFFICIENT VERIFICATION

In this section, we first present the naive solution method to verify whether each object is the result of an IRNN query. Next, a more efficient verification algorithm termed *Probability Decomposition Verification* (PDV) algorithm is proposed to verify candidate objects.

A. Naive Solution

The set $\mathcal{C}$ of candidate objects consists of unpruned objects at each timestamp generated by the first phase of our framework. In the verification phase, we verify the set of candidate objects by using the following lemma.

Lemma 5: Given a time interval $\Delta = [s, e]$ of an IRNN query, if an uncertain moving object o_i is verified to be an RNN of the query object q at x timestamps in $[s, t]$ and $x \geq (e - t)$, then o_i is the result of the IRNN query.

Proof: For an uncertain moving object o_i, we verify this object from time s to time t. Since there exists x timestamps when o_i is an RNN of the query object q, o_i can maintain the nearest neighboring relation to q for at least x timestamps. Hence, for the remainder time $(t, e]$, other objects can only maintain the nearest neighboring relation to q at most $(e - t)$ timestamps. According to the definition of IRNN query, o_i is the result of the IRNN query. ■

After pruning, we need to verify whether each object is an RNN of q at each timestamp and to accumulate the timestamps when this object is an RNN of q. Using the above lemma, we solely verify it from time s to t ($t \leq e$) so that the time efficiency of verification can be improved. Based on it, we propose a naive solution for the verification phase. In this solution, we first achieve computation for the probability that each object is located at each possible location at each timestamp. Because of the locations of the object with correlations, this probability is related with those at previous timestamps. Using Eq.(6), we evaluate the probability of each object being an RNN of q at each timestamp until satisfying the conditions of Lemma 5. Finally, the result of IRNN query can be returned. We illustrate the key steps of the naive solution in Algorithm 2.

The naive solution algorithm first calculates the probability that the query object q is located at each possible location for each timestamp in the given time interval (lines 2-4). Then this algorithm computes the exact probability of each object being the IRNN of the query and test whether it qualifies the result for the IRNN query (lines 5-7). Finally, the naive solution algorithm returns the objects which can maintain

Algorithm 2: Naive Solution Algorithm

Input : $IRNN(q, \Delta, \mathcal{C})$
Output: Interval reverse nearest neighbor of q

```
1  t ⟵ s
2  while t ≤ e do
3      for each possible location v_m of q^t do
4          compute the value of Pr(q^t = v_m)
5      for each object o_i in C do
6          compute the value of Pr(o_i^t = u_k)
7          for each other object o_j in C do
8              if Pr_IRNN(q^t, o_i^t) < Pr_IRNN(q^t, o_j^t) then
9                  o_i ⟵ o_j
10             count(o_i) ⟵ count(o_i) + 1
11         t ⟵ t + 1
12 for each other object o_j do
13     if count(o_i) ≥ count(o_j) then
14         output o_i
```

reverse nearest neighboring relations to the query object (lines 8-14). The naive solution algorithm is rather time-consuming because, in order to verify each object as IRNN, we need to take into consideration not only all the possible locations of this object but also the possible locations of query object and other nearby objects. Due to Markov correlations of objects, we compute the probability that any object is located at a possible location by retrieving a large number of conditional probabilities in CPTs. Consequently, we will present a more efficient verification algorithm in the next subsection.

B. Probability Decomposition Verification

Up to now, we have discussed the naive solution for verifying each candidate object by evaluating the probability that this object is an RNN of the query object q at time t ($s \leq t \leq e$). By using the complete probability formula to evaluate probabilities, it needs to retrieve a large number of conditional probabilities in CPTs. To improve the verification efficiency, we present a more efficient algorithm, named Probability Decomposition Verification (PDV) algorithm, which decomposes the probability of an object being located at a possible location into dominating terms and remainder terms. Each term represents the probability that this object moves to the location u in one of possible trajectories passing u. When proper dominating terms are chosen, the probability that any object is an RNN of q is estimated approximately. Along with increasing the ratio of dominating terms step by step, this algorithm can return accurate results of IRNN queries through comparing the probabilities to be RNN instead of computing these probabilities exactly.

Since there may be many possible trajectories of an uncertain moving object, it is time-consuming to calculate the probability of this object being located at a location. Therefore, we approximately estimate it by retrieving top-C (*i.e.*, C is a constant, *p.s.*, C is usually initialized smaller than 4) possible trajectories passing the location called dominating terms. For the example in Figure 5, the probability that o_i moves from u_1 or u_2 to u is 0.6 or 0.1, respectively. Since $0.1 \ll 0.6$, the likelihood that the possible trajectory of o_i passing u_2 is much smaller than that passing u_1. This kind of smaller likelihood hardly has effect on evaluating the probability of the object being located at a possible location u. We propose PDV algorithm which first ignores some smaller probability cases (remainder terms) to improve efficiency, and then evaluate the upper bound of the remainder terms to ensure that the query results are accurate. For the probability of $o_i^{t+1} = u$, we decompose this probability as follows.

$$\Pr(o_i^t = u) = D(o_i^t = u) + R(o_i^t = u), \tag{11}$$

where $R(o_i^t = u)$ denotes the remainder terms of the probability of $o_i^t = u$. At time t, o_i is located at u. For the dominating terms, $D(o_i^t = u)$ is denoted as

$$D(o_i^t = u) = \sum^{C} \Pr(o_i^t = u | o_i^{t-1} = u_c) D(o_i^{t-1} = u_c), \tag{12}$$

where $\Pr(o_i^t = u | o_i^{t-1} = u_c)$ denotes one of top-C largest conditional probabilities at time t. As mentioned in the above lemma, we approximately compute the probability that o_i is an RNN of q with dominating terms. As shown in Figure 5, we choose top-2 dominating terms for $\Pr(o_i^t = u)$ which can be approximately computed as these conditional probabilities denoted by broken paths (*i.e.*, possible trajectories). Due to the approximate computation for $\Pr_{\text{IRNN}}(q^t, o_i^t)$, the efficiency of verification can be improved.

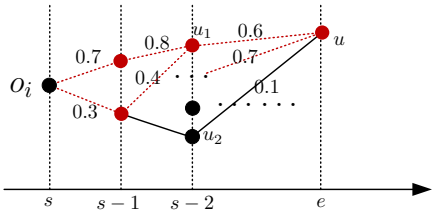


Fig. 5. An example of dominating terms

Using Eqs.(11) and (12), for the example in Table II, $\Pr_{\text{IRNN}}(q^2, o_2^2)$ is approximately computed as 0.4304 (*i.e.*, C is set to be 1). It follows that o_2's probability to be an RNN of q is larger than that of o_1 at t=2. In order to exactly verify each candidate object, we also need to consider remainder terms which is denoted as that $\Pr(o_i^t = u)$ subtracts dominating terms. We summarize the upper bound of evaluating remainder terms as follows.

Lemma 6: Let $R(o_i^t = u)$ be the remainder terms of $\Pr(o_i^t = u)$. Then, $R(o_i^t = u) \leq \sum_{k=1}^{K-C} \{\Pr(o_i^t = u | o_i^{t-1} = u_k)\}$ ($C \leq K$).

Proof: Because $\Pr(o_i^t = u) = \sum_{k=1}^{K} \Pr(o_i^t = u | o_i^{t-1} = u_k) \cdot \Pr(o_i^{t-1} = u_k) = D(o_i^t = u) + R(o_i^t = u)$, $\Pr(o_i^t = u) \leq \sum^{C} \Pr(o_i^t = u | o_i^{t-1} = u_c) \cdot \Pr(o_i^{t-1} = u_c) + \sum_{k=1}^{K-C} \{\Pr(o_i^t = u | o_i^{t-1} = u_k)\}$, where $\Pr(o_i^t = u | o_i^{t-1} = u_c)$ denotes one of

top-C largest conditional probabilities at time t. Also, since the dominating terms $D(o_i^t = u)$ of $\Pr(o_i^t = u)$ includes top-C largest conditional probabilities of o_i, $R(o_i^t = u) = \Pr(o_i^t = u) - D(o_i^t = u) \leq \sum_{k=1}^{K-C}\{\Pr(o_i^t = u|o_i^{t-1} = u_k)\}$. ■

Algorithm 3: Probability Decomposition Verification Algorithm

Input : $IRNN(q, \Delta, \mathcal{C})$
Output: Reverse nearest neighbor of q

1 $t \longleftarrow s$, $k \longleftarrow 0$
2 **while** $t \leq e$ **do**
3 **for** each possible location v_m of q^t **do**
4 compute the value of $\Pr(q^t = v_m)$
5 **while** $\mathcal{C} \neq \phi$ **do**
6 **for** each object o_i in $\mathcal{C}$ **do**
7 **for** each possible location u object o_i **do**
8 compute the top-$(C + k)$ dominating terms $D(o_i^t = u)$ and the upper bound of $R(o_i^t = u)$
9 **for** each other object o_j in $\mathcal{C}$ **do**
10 estimate $\Pr_{\text{IRNN}}(q^t, o_i^t)$ and $\Pr_{\text{IRNN}}(q^t, o_j^t)$
11 **if** $Pr_{IRNN}(q^t, o_i^t) < Pr_{IRNN}(q^t, o_j^t)$ **then**
12 $o_i \longleftarrow o_j$
13 $count(o_i) \longleftarrow count(o_i) + 1$
14 $t \longleftarrow t + 1$
15 $k \longleftarrow k + 1$
16 **for** each other object o_j **do**
17 **if** $count(o_i) \geq count(o_j)$ **then**
18 output o_i

Using the upper bounds of remainder terms to add dominating terms, we can evaluate the upper bound of the probability that any object o_i is located at a possible location. Thus, the upper bound of the probability that this object is an RNN of q is computed as well. If this upper bound is not more than the value of $\Pr_{\text{IRNN}}(q^t, o_j^t)$ estimated by dominating terms, this object is verified to be not RNN of the query object q. According to the bounds of remainder terms, the candidate objects can be verified without computing the probability to be RNN of q. We first set C to be a smaller value (smaller than 4). Then we can increase the ratio of dominating terms step by step until all objects in $\mathcal{C}$ are verified.

Based on the above processing, a more efficient verification approach PDV algorithm is presented to return accurate results of IRNN queries through estimating and comparing the probability that any object is an RNN of q instead of computing the probability exactly. We illustrate its key steps in Algorithm 3.

Similar to the naive solution, the PDV algorithm calculates the probability that the query object q is located at each possible location at each timestamp as well (lines 3-4). According to Lemma 6, the algorithm efficiently verifies each candidate object through estimating the probability that the object o_i is located at each possible location using the dominating terms and the upper bounds of remainder terms (lines 6-8). The PDV algorithm next increases the ratio of dominating terms until all objects in $\mathcal{C}$ are verified (line 15). Lastly, this algorithm returns the objects which can maintain reverse nearest neighboring relations to the query object q for the longest time (lines 16-18). The performance of the PDV algorithm and the naive solution algorithm will be compared in the experimental section.

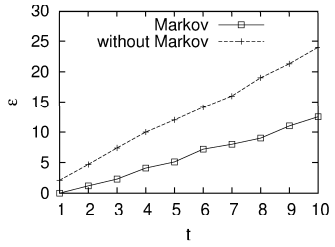


Fig. 6. Effectiveness of the Markov model

VI. Experiments

A. Experimental settings

All the experiments were conducted on a PC with a 2.6GHz Intel Processor and 2GB main memory. We used geographic information combining with synthetic moving uncertain objects as synthetic datasets. The datasets simulated moving cars which move in the maps of North Carolina and Texas[1], respectively. In any map, we set up 10,000 cars and each car was denoted by an uncertain object trajectory. The geographic information of North Carolina or Texas consists of 28,483 or 61,440 zip codes, and each representing an instance, denoted by the possible location of the car. Suppose that any car can only move from one zip code to eight nearby zip codes and the locations of cars are Markov correlated at all timestamps. We generated transition probabilities of the cars randomly.

TABLE III
Parameter settings

Parameter	Value range	Default value
Δ	[5, 15]	10
n	[500, 10,000]	3,000
$\mathcal{T}$	[0, t]	[0, 30]
C	[1, 5]	2

Moreover, we also used the International Ice Patrol (IIP)[2] as the real dataset for our experiments. The database contains a set of iceberg sighting records, each of which contains the location of the iceberg, and the number of days the iceberg has drifted. In this dataset, we selected 50,000 records as the training data and we used them to learn transition probabilities. Therefore, we can evaluate all the algorithms on the synthetic and real datasets with Markov correlations.

For the experiments, we set the time domain $\mathcal{T}$ to be [0, t] and the number of locations at next time for any car to

[1]http://www.census.gov/geo/www/tiger/
[2]http://nsidc.org/data/g00807.html

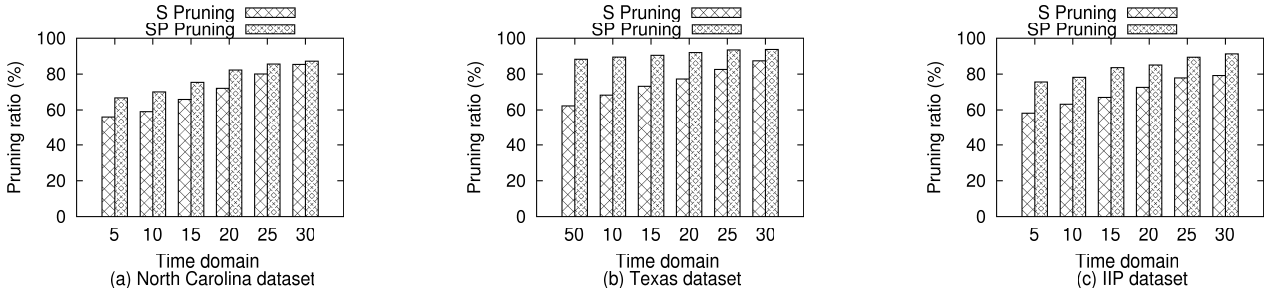


Fig. 7. The pruning rules for three datasets

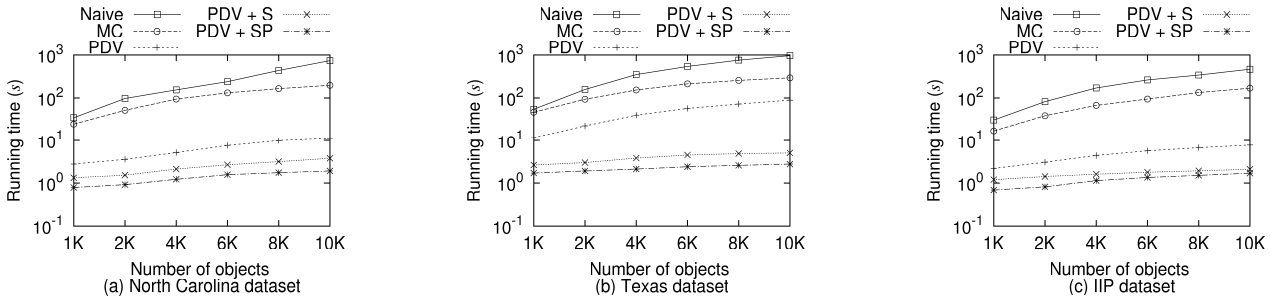


Fig. 8. The effect of object number on the time cost of different algorithms for three datasets

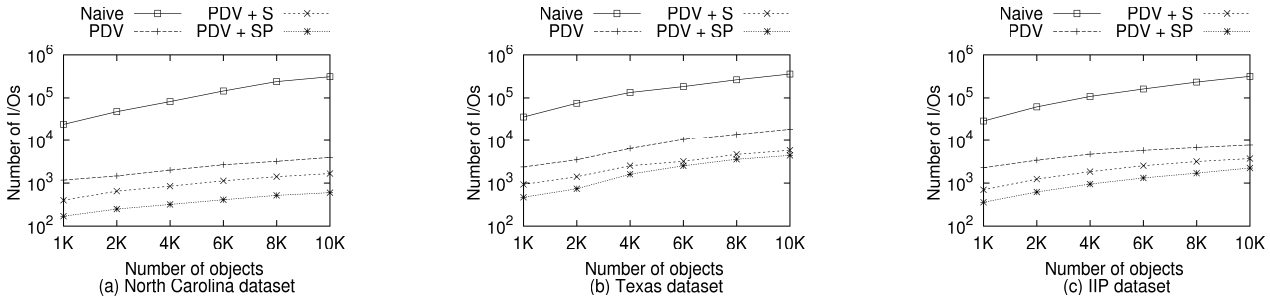


Fig. 9. The effect of object number on the I/O cost of different algorithms for three datasets

be [0, 8]. Similarly, we set the query object q to be a car as well, which is a Markov correlated moving object. The time interval Δ is included by this time domain. The parameters for our experiments are summarized in Table III.

Alternative Techniques Considered. We contrast our methods for answering IRNN queries with the Monte-Carlo based method (MC). As described in [29], the MC approach samples paths of each object and outputs the fraction of the sampled paths which fulfill the query predicate. Sampling the path of an object requires first drawing a start state from the objects distribution. According to our settings, we sample the object distribution at the first timestamp. Afterwards for each timestamp an object moving from the successor locations to the current location is chosen according to the probability distribution given by the conditional probabilities in the CPTs. We generate 10,000 samples for each uncertain moving object, and evaluate the probability that any object o_i is located at a possible location u at time t as

$$\widetilde{\Pr(o_i^t = u)} = \frac{M}{N}, \tag{13}$$

where N denotes the number of paths to possible locations of $o_i^{(t-1)}$, and M denotes the number of paths to u. Note that the MC approach only returns approximate query results, where the accuracy can be improved if more paths are sampled.

B. Experimental Results

In this subsection, the performance of all the proposed algorithms is evaluated. At first, we evaluate the effectiveness of the Markov model on the IIP dataset. We selected 50,000 records as the training data and used them to learn the Markov model. Therefore, the possible trajectories of o_i are inferred by the Markov model in $[0, t]$. Also, we can infer the locations at time t with the historical locations of o_i within the range in which o_i is located from time 0 to time t (without the Markov model). To test the accuracy of inferred locations, we define ϵ as the number of inferred locations which are not real locations from time 0 to time t. ϵ can reflect the accuracy of different inferred methods, and the lower ϵ the higher accuracy is (when $\epsilon = 0$, all inferred locations in $[0, t]$ are the real locations). As shown in Figure 6, the possible trajectories inferred based on the Markov model are more accurate than those inferred without the Markov model. Consequently, we evaluate IRNN queries based on the Markov model in the next experiments.

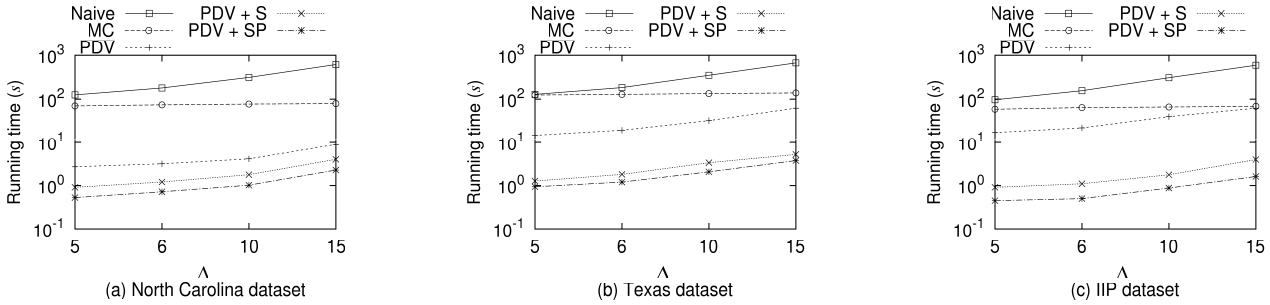


Fig. 10. The effect of Δ on the time cost of different algorithms for three datasets

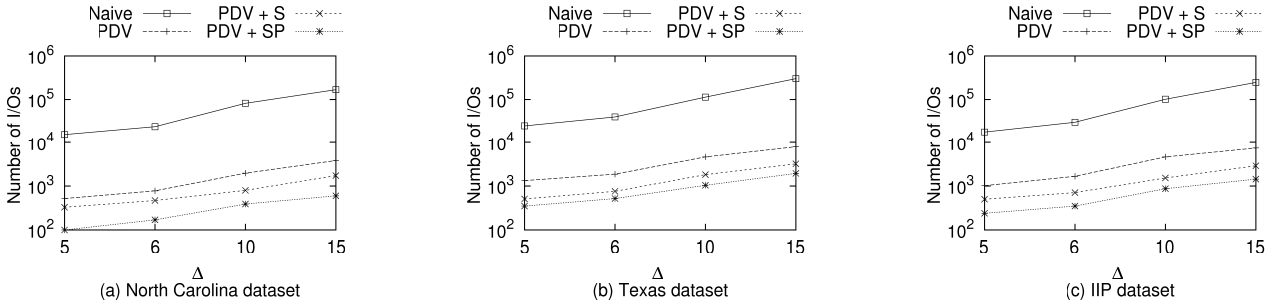


Fig. 11. The effect of Δ on the I/O cost of different algorithms for three datasets

We next study the pruning rules for IRNN queries over uncertain moving objects with Markov correlations in the above three datasets. We set the time interval Δ to 10, and vary the time domain $\mathcal{T}$ from [0, 5] to [0, 30]. We achieve space pruning rules (S Pruning) and space + probability pruning rules (SP Pruning) for different datasets. The pruning ratio (*i.e.*, number of pruning objects divided by number of total objects) is shown in Figure 7. As illustrated in Figure 7, the pruning ratio is more than 50% for any pruning rule over the datasets. Furthermore, the probability pruning method can further filter objects after performing the space pruning rules. Therefore, the effectiveness of our pruning rules is demonstrated by these experimental results.

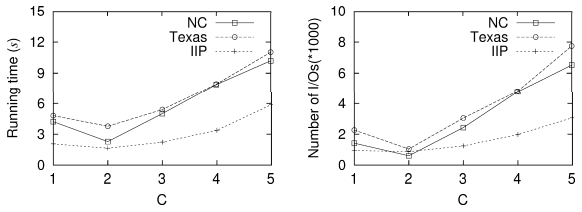
We also investigate the effect of the object number on the CPU cost of our algorithms (*i.e.*, naive solution algorithm (Naive), the PDV algorithm, PDV+S pruning algorithm and PDV+SP pruning algorithm) and MC approach. We set the time interval Δ to 10, set C to be 2 for the PDV algorithm and vary the number of moving objects from 500 to 3,000. We compute the average time efficiency for the time interval $\Delta = [1, 10], [11, 20]$ and $[21, 30]$. Figure 8 illustrates the results over the three datasets. The time cost of all the methods increases along with the increasing number of objects, and the efficiency of PDV algorithm is much higher than that of the MC approach and the naive solution algorithm. It can be explained that the naive solution algorithm requires too much time to verify each object, and the MC approach needs to generate a large number of samples, which costs much time in the query process. Furthermore, the PDV+S pruning algorithm and PDV+SP pruning algorithm can improve efficiency of the query. The reason is that our pruning methods are able to reduce the number of verified objects.

In the next experiment, we evaluate the effect of the object number on the I/O cost of these algorithms. Since the I/O cost of the MC approach depends on the number of samples, we do not need to compare its I/O cost with other methods. We set the space constraint for kNN query requirements and the error constraint for range query requirements. Likewise, we set the time interval Δ to 10, set C to be 2 for the PDV algorithm and vary the number of moving objects from 500 to 3,000. We also compute the average time efficiency for the time interval $\Delta = [1, 10], [11, 20]$ and $[21, 30]$. The results are illustrated in Figure 9. It is obvious that the I/O cost of PDV+SP pruning algorithm is lower than that of any other algorithm. Since the PDV algorithm computes the probability that any object is located at a possible location using only dominating terms, the I/O cost can be reduced by less retrieving CPTs. In addition, the pruning rules can further reduce I/O cost effectively.

Then we examine the effect of the time interval Δ on the CPU cost of our algorithms and the MC approach. We set C to be 2 for the PDV algorithm, and vary the time interval Δ from 5 to 15. As shown in Figure 10, the time cost of each method increases along with increasing Δ, and the efficiency of the PDV algorithm is much higher than that of the MC approach and the naive solution algorithm. The reason is that the PDV algorithm can evaluate the probability that any object is an RNN of q without the exact computation. It can be also seen that the proposed pruning rules improve the efficiency.

We study the effect of Δ on the I/O cost of different algorithms for three datasets. We also vary the time interval Δ from 5 to 15. The experimental results are illustrated in Figure 11. From the results, it can be seen that the I/O cost of PDV+SP pruning algorithm is lower than that of any other algorithm. Since the PDV algorithm and pruning rules can reduce the number of retrieving CPTs considerably, PDV+SP pruning algorithm answers IRNN queries over Markov corre-

lated objects efficiently.



(a) CPU cost (b) I/O cost

Fig. 12. The effect of C on the CPU and I/O costs

Finally, we test the effect of C on the costs of the PDV algorithm over different datasets (*i.e.*, NC denotes the North Carolina dataset). Figure 12 shows the CPU and the I/O costs for the three datasets. We can see that the cost is smallest when $C = 2$, and larger or smaller C would decrease the efficiency of our queries. The reason is that the larger C increases the cost of estimating the probability that each object is a reverse neighbor of q, and the smaller C only obtains the rather imprecise probability so that it increases the cost of further verification.

VII. Conclusions

In this paper, we have formally defined a new type of RNN queries, named IRNN queries, which can return RNNs of the moving query object for the longest time in a given time interval. Given a set of uncertain moving objects that exhibit Markov correlations, we have developed a pruning-verification framework to answer IRNN queries on these moving objects in two phases. In order to improve the query efficiency, in the pruning phase, we have proposed space pruning rules and probability pruning rules to reduce the number of moving objects that are certainly not results of the query respectively. Then we have presented two algorithms for verifying the unpruned objects efficiently in the verification phase. Specially, Probability Decomposition Verification (PDV) algorithm is a more efficient approach to return accurate results of IRNN queries through estimating and comparing the probability that any object is an RNN of the query object instead of computing the probability exactly. Finally, we have identified the efficiency of the proposed methods through the extensive experiments by comparing a sampling-based approximate method on synthetic and real datasets.

Acknowledgments The research is supported by the National Basic Research Program of China (973 Program) under Grant No.2012CB316201, the Hong Kong RGC GRF Project No.611411, HP IRP Project 2011, Microsoft Research Asia Grant, MRA11EG05, HKUST RPC Grant RPC10EG13, and the National Natural Science Foundation of China Nos.61003058, 61033007 and 61272177.

References

[1] R. Cheng, D. V. Kalashnikov, and S. Prabhakar, "Querying imprecise data in moving object environments," *TKDE*, vol. 16, 2004.

[2] R. C.-W. Wong, M. T. Özsu, P. S. Yu, A. W.-C. Fu, and L. Liu, "Efficient method for maximizing bichromatic reverse nearest neighbor," in *VLDB*, 2009.

[3] T. Emrich, H.-P. Kriegel, N. Mamoulis, M. Renz, and A. Züfle, "Querying uncertain spatio-temporal data," in *ICDE*, 2012.

[4] M. A. Cheema, X. Lin, W. Wang, W. Zhang, and J. Pei, "Probabilistic reverse nearest neighbor queries on uncertain data," *TKDE*, vol. 22, no. 4, 2010.

[5] M. B. Christopher R, Julie Letchner and D. Suciu, "Event queries on correlated probabilistic streams," in *SIGMOD*, 2008.

[6] M. A. Soliman, I. F. Ilyas, and S. Ben-David, "Supporting ranking queries on uncertain and incomplete data," *VLDB J.*, vol. 19, 2010.

[7] J. Letchner, C. Re, M. Balazinska, and M. Philipose, "Access methods for markovian streams," in *ICDE*, 2009.

[8] F. Korn and S. Muthukrishnan, "Influence sets based on reverse nearest neighbor queries," in *SIGMOD*, 2000.

[9] C. Yang and K.-I. Lin, "An index structure for efficient reverse nearest neighbor queries," in *ICDE*, 2001.

[10] Y. Tao, D. Papadias, and X. Lian, "Reverse knn search in arbitrary dimensionality," in *VLDB*, 2004.

[11] W. Wu, F. Yang, C.-Y. Chan, and K.-L. Tan, "Finch: evaluating reverse k-nearest-neighbor queries on location data," in *VLDB*, 2008.

[12] E. Achtert, H.-P. Kriegel, P. Kröger, M. Renz, and A. Züfle, "Reverse k-nearest neighbor search in dynamic and general metric databases," in *EDBT*, 2009.

[13] E. Achtert, C. Böhm, P. Kröger, P. Kunath, A. Pryakhin, and M. Renz, "Efficient reverse k-nearest neighbor search in arbitrary metric spaces," in *SIGMOD*, 2006.

[14] Y. Tao, M. L. Yiu, and N. Mamoulis, "Reverse nearest neighbor search in metric spaces," *TKDE*, vol. 18, 2006.

[15] S. Šaltenis, C. S. Jensen, S. T. Leutenegger, and M. A. Lopez, "Indexing the positions of continuously moving objects," *SIGMOD Rec.*, vol. 29, 2000.

[16] I. Stanoi, M. Riedewald, D. Agrawal, and A. E. Abbadi, "Discovery of influence sets in frequently updated databases," in *VLDB*, 2001.

[17] K.-I. Lin, M. Nolen, and C. Yang, "Applying bulk insertion techniques for dynamic reverse nearest neighbor problems," in *IDEAS*, 2003.

[18] I. Stanoi, D. Agrawal, and A. E. Abbadi, "Reverse nearest neighbor queries for dynamic databases," in *SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, 2000.

[19] X. Lian and L. Chen, "Efficient processing of probabilistic reverse nearest neighbor queries over uncertain data," *VLDB J.*, vol. 18, no. 3, 2009.

[20] T. Bernecker, T. Emrich, H.-P. Kriegel, M. Renz, S. Zankl, and A. Züfle, "Efficient probabilistic reverse nearest neighbor query processing on uncertain data," in *VLDB*, 2011.

[21] R. Benetis, C. S. Jensen, G. Karčiauskas, and S. Šaltenis, "Nearest and reverse nearest neighbor queries for moving objects," *VLDB J.*, vol. 15, no. 3, 2006.

[22] Y. Tao and D. Papadias, "Time-parameterized queries in spatio-temporal databases," in *SIGMOD*, 2002.

[23] T. Xia and D. Zhang, "Continuous reverse nearest neighbor monitoring," in *ICDE*, 2006.

[24] J. M. Kang, M. F. Mokbel, S. Shekhar, T. Xia, and D. Zhang, "Continuous evaluation of monochromatic and bichromatic reverse nearest neighbors," in *ICDE*, 2007.

[25] M. A. Cheema, X. Lin, Y. Zhang, W. Wang, and W. Zhang, "Lazy updates: an efficient technique to continuously monitoring reverse knn," in *VLDB*, 2009.

[26] W. Wu, F. Yang, C. Y. Chan, and K.-L. Tan, "Continuous reverse k-nearest-neighbor monitoring," in *MDM*, 2008.

[27] T. Emrich, H. Kriegel, N. Mamoulis, M. Renz, and A. Zufle, "Querying uncertain spatio-temporal data," in *ICDE*, 2012.

[28] X. Lian and L. Chen, "A generic framework for handling uncertain data with local correlations," in *VLDB*, 2010.

[29] R. Jampani, F. Xu, M. Wu, L. L. Perez, C. Jermaine, and P. J. Haas, "Mcdb: a monte carlo approach to managing uncertain data," in *SIGMOD*, 2008.