Management and Prediction of Moving Objects Under Location Uncertainty

Abdullah Islam

A thesis
submitted in partial fulfillment of the
requirements for the degree of

Master of Science in Computer Science & Systems

University of Washington

2020

Committee:

Mohamed Ali

Abdeltawab Hendawi

Program Authorized to Offer Degree:
Computer Science & Systems

University of Washington

Abstract

Management and Prediction of Moving Objects
Under Location Uncertainty

Abdullah Islam

Chair of the Supervisory Committee:
Associate Professor Mohamed Ali
School of Engineering & Technology

In spatio-temporal systems, precise location data is desirable but often not available due to obfuscation, privacy, hardware inaccuracies, and other factors. Progress has been made in research which deals with the uncertainty of moving objects' location data. However, much of the existing work does not always consider factors such as constraints imposed by the topology of road networks, and harmonic integration between past movements, current, and prospective imprecise positions. In this thesis, we propose an approach that utilizes time, distance, and connectivity constraints of a road network to infer a moving object's past, present, and future locations more precisely when its exact location data is not available. The experimental results using real GPS trajectories confirm the efficiency of our proposed solution for reducing uncertainty and inferring historical, and future locations.

TABLE OF CONTENTS

LIST OF FIGURES

Chapter 1

INTRODUCTION

The availability of GPS-enabled devices has sparked the growth of the need for location-aware services such as navigation services and targeted advertisements. However, hardware inaccuracy, noise, and the obfuscation of precise locations to meet privacy concerns have expanded the need for these location-aware services to handle imprecise location data to meet users' needs.

In this thesis, we address the problem of managing and using imprecise spatial data to predict a moving object's future location in addition to estimating its past trajectory and present location more precisely. To accomplish this, we first define uncertainty as the absence of precise location data (i.e., exact latitude and longitude); instead, an object's location is represented by an imprecise circular region. To manage the described uncertainty, we propose a novel approach, which allows us to refine the past movements of an object as we track its location updates, and more precisely determine its historical trajectory, identify its current location, and predict the object's future movements.

Challenges exist in the way of effectively and efficiently querying moving objects under location uncertainty. The size of the uncertain region representing an object's possible location and the density of the road network may effect the efficacy of the proposed solution. Both of these factors introduce a large set of possible locations for the moving object which we have to account for. Another factor may be the metadata available from the road network graph. Too simple of a road network graph may not provide enough information to make an accurate inference and would leave us to resort to random guessing. Ability to use information such as edge type i.e highway, bike trail could allow us to reduce possible paths for a given moving object. Another factor may be the lack of continuity in an object's tra-

jectory. Our proposed solution does not handle piece-wise trajectories and may suffer or fail to perform entirely if an object does not follow a continuous trajectory or a large sequence of location updates are missed.

Existing techniques that query uncertain moving objects' locations suffer from inaccuracies and/or performance issues. Models that query the historical movements of objects often fail to predict their future trajectory. Similarly, models that query the future often do not consider the recent past. Approaches that support predictive, and uncertain location queries in many cases require a large set of historical data; such historical data is not easily obtainable due to privacy, confidentiality, or simply the lack of data such as in rural areas. Other models fail to leverage the constraints of road networks to their advantage even though vehicles constrained to road networks are frequently the motivation for such models.

The main idea of our proposed solution can be divided into two parts. First, we prune and refine possible historical locations with each location update. Second, we utilize the historical movements to narrow down the possible current locations. The current/latest locations are then used to predict future movements with some degree of accuracy despite imprecise location data.

At the beginning of a trip, an instance of the moving object's imprecise starting location is represented by a point and a radius encompassing a region. A set for containing imprecise regions that depict where the object was, and is, is allocated; these regions provide the mechanisms to query an object's historical trajectory. For predictive queries, the future movements of the object are derived from the object's latest imprecise region. Note, since moving objects with imprecise locations are being indexed, there is always a set of nodes denoting an object's possible position at any given time.

With subsequent updates, a probability model assigns probabilities to the nodes within the new imprecise location in the object's trip. Imprecise location updates allow us to calculate the probability of travel along edges between the previous and most recent imprecise regions in the road network. These probabilities are then used to refine the set of possible nodes the object may have traveled through in the past. In practice, various factors may

affect this model, as nodes may be eliminated through the pruning process, and of course, the probability and edge weight functions may significantly impact the effectiveness of the solution.

Our proposed solution uses information from the underlying road network and data refined from the past possible locations to aid prediction. Pruning of historical nodes allows us to make better predictions about future movements as it reduces the number of total nodes we have to account for. In addition, the probability model for both historical and predictive queries can be modified according to users' needs. This allows for an extensible solution that can be modified for particular use cases.

Our contributions in this thesis are summarized below:

- We propose a novel approach for querying and inferring possible positions of a moving object at any given time in the past, present, and future under uncertainty.

- We introduce a probability model that is used to infer the location of a moving object.

- We provide an incremental approach to building, maintaining, and refining the object's trajectory as it travels through the road network.

- We demonstrate the effectiveness of our proposed solution using real-life data.

In the following chapters, we look at existing work and how our proposed solution fits in the domain of moving object query processing. Then we define our problem and introduce the supported queries. We then describe the details of the proposed solution including the probability model, followed by algorithms and examples. Finally, we go over experimental evaluations demonstrating the effectiveness of our solution and conclude our research.

Chapter 2

RELATED WORK

In this chapter, we go over some of the existing work, which aims to solve the problem of predicting a moving object's future movements and managing uncertainty. Much work has been done in the area of movement prediction. Some of these strategies utilize motion functions [13, 23]. Though the functions can be complex, they may not fit the movements of objects in some cases. Besides, these approaches do not support prediction under uncertainty.

The predictive tree relies on road networks and the assumption of the shortest route to predict future movements and does not consider uncertainty [9]. The mobility model aims to predict future movements as well. This particular approach relies on an extensive collection of historical trajectory data [14]. The R*-Tree bases the prediction model on work done on R-Trees [28]. The adaptive multi-dimensional histogram allows for past, present, and predictive queries by constructing spatial histograms that reflect the movement of objects [22]. The TPR*-Tree attempts to solve the problem of movement prediction by building on top of existing work on the TPR tree [29] and employing a new set of insertion and deletion algorithms [24]. These approaches do not account for uncertainty in location data of moving objects.

Panda is a framework that supports generic predictive queries for moving objects but does not support uncertainty [10]. This approach models movement on Euclidean space and does not make use of road network constraints.

The following methods allow for predictive queries under uncertainty. One particular strategy relies on location and velocity constraints to aid the prediction of moving objects under uncertainty [31]. The UTR-Tree builds an index structure on top of R-trees to support predictive queries under uncertainty [5].

The U-tree makes use of probabilistically constrained rectangles, which corresponds to an object's uncertain location and helps in the pruning process. This approach supports querying under uncertainty [25]. However, this approach does not support predictive queries.

PLM is another model that aims to solve this problem. This approach utilizes adjacency matrices constructed from road network topology [15]. PLM does not support uncertainty.

The method proposed in [17] makes use of modified R-trees, which support querying moving objects under uncertainty. This method makes use of probability density functions to represent objects and custom query filtering schemes.

The uncertain trajectory hierarchy is an index structure which makes use of time-dependent probability distribution functions to evaluate range queries under uncertainty [32]. This structure does not support predictive queries.

SubSyn proposed in [30] predicts the destination of moving objects in the absence of a large number of historical trajectories. However, this method does not provide insight into the trajectory an object may take to get there nor handles uncertainty.

Work proposed in [18] uses hidden Markov-models to predict the next place to be visited. Another work proposed in [8] makes use of Markov-Chains to accomplish next place prediction. In [7], the authors employ Markov-chains to model trajectories for prediction and making use of temporal information. However, uncertainty is not taken into consideration in any of these approaches.

Another technique employing Markov-chains is described in [19]. This approach facilitates nearest-neighbor queries under uncertainty. This approach does not consider road network constraints as it makes use of Euclidean space.

The work proposed in [2] uses Representative Trajectory objects to capture movements within relevant regions, which are then refined and used to predict future movements. This method doesn't account for uncertainty.

In [16], the authors make use of fuzzy-LSTM networks to predict object trajectories. The method described in [20] proposes Continuous Time Bayesian Networks and accompanying algorithms to predict routes of moving objects. In [1], the authors propose a technique

InferTra, to predict trajectories of moving objects by utilizing a mobility model. These methods require a large amount of data to be effective and do not consider imprecise location data.

The Mobility prediction architecture described in [21] makes use of Dempster-Shafer's theory though it doesn't handle uncertainty. This approach does not rely on a large set of historical data to be effective.

The work described in [26] models uncertainty by bounding an object's location between some points. This uncertainty model allows the authors to perform range queries. This approach uses Euclidean space and velocity constraints and therefore does not consider the constraints of a road network graph.

The IPAC-NN tree described in [27] facilitates nearest-neighbor queries for moving objects under location uncertainty. It models uncertainty using shared cylinders which represent trajectories and allows nearest neighbor queries under uncertainty.

The solution proposed in [3] presents solutions for answering location based range queries under uncertainty. The authors make use of Minkowski sum to evaluate the intersection between query region and an object's uncertain region to aid pruning. However, this approach does not account for the constraints of the road network.

A solution for answering nearest-neighbor queries under uncertainty is described in [4]. The approach makes use of R-trees to to filter nearest neighbors. Then probabilistic verifiers are used to evaluate the final answer. This approach does not take into account the constraints of a road network graph.

The use of Gaussian-distribution based management of uncertainty has been discussed in [12], [11]. In [12], [11], the proposed solutions answer probabilistic range, and nearest-neighbor queries under location uncertainty respectively.

Discussion

We put forward an approach for querying moving objects under location uncertainty. It facilitates both historical location queries as well as predictive queries. Our solution takes

into consideration the constraints and topology of the underlying road network, previous movement patterns, and is extensible by allowing for arbitrary probability models. In summary, we'd like to demonstrate how the proposed work fits into the spatio-temporal query processing domain and highlight the following qualities of the solution:

- It is meant to take into consideration uncertainty in location data.
- It does not rely on large amounts of historical data to be effective.
- It uses information from previous uncertain location updates to further refine and improve subsequent inferences.
- It can be extended by custom user-defined probability models to fit specific use cases.

Chapter 3

PRELIMINARIES

In this chapter, we go over the problem definition, and the supported queries. Below we define a few keywords we will come across throughout the rest of the paper.

3.1. Definitions

Region is denoted by a central point and a radius that encompasses one or more nodes within a road network graph. A region denotes the possible location of a moving object at a given time.

Uncertainty is defined as the lack of precise location data. Instead, we assume that the information available to us is that of an imprecise region in which a moving object may reside. The size of the region, defined by its radius, can be viewed as a measure of uncertainty. In other words, smaller or larger the scope of an imprecise region is, the more or less certain we are of an object's possible state, respectively.

Steps/Updates are events received from a moving object which informs us of movement. This movement is reflected by a new uncertain region.

3.2. Problem Definition

Given a road network graph, a moving object, and a series of imprecise regions denoting the object's trajectory, we would like to propose a solution that allows us to answer queries about the object's past, as well as its future locations. To this end, our goal is to develop an approach for effectively querying moving objects under location uncertainty.

3.3. Supported Queries

The supported queries have the following input: a timestamp for which we try to predict the object's location. Alternatively, this can be the step count which represents how many steps in the future or past from the current region we are inferring the object's location.

Historical queries allow us to infer a precise location for a moving object at a given time in the past. Though the time range is limited by the earliest recorded update.

Similarly, *predictive queries* allow us to infer about an object's location at a given time in the future. As location updates are received and we approach closer to the prediction time, we can continuously refine and improve the likelihood of estimating the future location correctly using historical trajectories.

The result of these queries is a node which is predicted to be the object's location at the given time. This can be chosen according to a user defined probability model. We propose a probability model in later chapters of this thesis. Alternatively, a set of nodes and their associated probabilities can be returned by the query should it be necessary. In the end, we aim to achieve a meaningful degree of precision for both query types.

Chapter 4
PROBABILITY MODEL

Though arbitrary user-defined probability models are supported, we propose a probability model of our own, which makes use of time and distance constraints of the underlying road network. The probability models are applied to edges connecting uncertain regions from which individual nodes may derive their own probabilities. Prior to summarizing the proposed probability model, we state the following conditions to be true:

1. The sum of the edge probabilities of all the edges connecting the nodes between two regions is 1.

2. The sum of the node probabilities of all the nodes representing any uncertain region is 1.

3. The probability of the object being at a node is the sum of the probabilities for all incoming edges to that particular node.

4.1. Historical Probability Model

As an object moves between uncertain regions, the edges connecting the two regions affect the probabilities of all nodes within the new region. Note, nodes in the very first region do not inherit edges; as a result, their probabilities are uniform. Since subsequent regions in the object's trajectory inherit edges, as the object moves along its trajectory, by calculating and refining the edge probabilities between the newest and the previous imprecise regions, we can determine its actual trajectory with more precision. Below we give an example of what such a probability model could look like.

Given regions R_1 and R_2 let ΔT represent the time elapsed between the updates of the two regions and let E represent the edges between them. Suppose t_i is the time taken to travel along edge $e_i \in E$, we assume this information can be gathered from a time/distance matrix. We define a weight function for e_i as such:

$$w_i = |\Delta T - t_i| \tag{4.1}$$

Intuitively, we can say that if the inferred travel time between two nodes is close to the actual time taken to travel between two regions, then it is likely that the object traveled to and from said nodes hence the associated edge between these nodes should have a lower weight. Therefore, it is more likely that the edge with the lowest weight is the path taken by the object. Then, we can write the following:

$$P(e_i) \propto w_i \tag{4.2}$$

In the beginning of the chapter we stated in assumption 1. that the sum of all edge probabilities must be 1. The we can write the following:

$$1 = \sum_{e_i \in E} P(e_i) \tag{4.3}$$

In order derive a higher probability from edges with lower weights, we first compute the sum of all edge weights denoted by S.

$$S = \sum_{e_i \in E} w_i \tag{4.4}$$

Then, we define the following summation denoted by S':

$$S' = \sum_{e_i \in E} S - w_i \tag{4.5}$$

$$S' = S(|E| - 1) \tag{4.6}$$

Finally, we define the edge probability as:

$$P(e_i) = \frac{S - w_i}{S'} \tag{4.7}$$

Note, in order to apply probability to edges, the constants S, and S' must be calculated prior to computing individual edge probability.

4.2. Predictive Probability Model

Our predictive probability model makes use of the predictive tree proposed in [9]. The predictive tree primarily focuses on calculating predictive probabilities for moving objects, given that their location data is precise. However, under uncertainty, we represent the location of a moving object using a region made up of multiple nodes that serve as a possible location of the object. We construct multiple predictive trees rooted at each of these nodes, which then branches out to all possible nodes that can be traveled to from the respective root node. Note, during expansion, it is important to exclude circular references between nodes. We then assume the object takes the shortest route as described in [9] and calculate probabilities accordingly.

The predictive probability model follows the same principle as our historical probability model. Except the edge weights are not represented by the time difference, since we cannot receive updates from the future. Rather, we associate the distance between the source and destination nodes to be the weight of a given edge. Similar to the previous approach, lower edge weight suggests a higher probability per our assumption that travel along the shortest route is more likely than travel along a longer route. Given an edge e_i and its length d_i our weight function looks like this:

$$w_i = d_i \tag{4.8}$$

Then we apply equations (5) and (6) to compute the predictive probability of an edge.

4.3. Node Probability Assignment

So far, we've discussed the models for applying edge probabilities for both historical and future regions. However, since queries must return a node representing an object's location, we derive node probabilities from their respective incoming edges. Given a set of nodes N which reside in a given region, we first recall that:

$$1 = \sum_{n \in N} P(n) \tag{4.9}$$

Given a node $n \in N$, a set of incoming edges E_n for which n is the destination, the probability of node n can be expressed as the following:

$$P(n) = \sum_{e_i \in E_n} P(e_i) \tag{4.10}$$

And since the sum of all edge probabilities is always 1, we can say that equation 4.9, and thereby assumption 2. also holds true.

Chapter 5
QUERYING UNDER UNCERTAINTY

We describe the proposed solution in four steps: handling of location updates, pruning of historical regions, edge and node probability assignments, and maintenance of predictive regions.

Let us add to some definitions given in previous chapters and introduce new ones:

- *Imprecise location* is denoted by a central point and a radius. A moving object may reside at any point inside the *region* derived from an imprecise location.

- *Regions* are indexed by a timestamp or an integer representing the number of updates from a moving object. The nodes within regions have parent, child relations with nodes residing in regions before and after them, respectively.

- *Predictive tree(s)* are data structures covered in [9] which are used for predictive queries over moving objects. Nodes within each level of a predictive tree are associated with a corresponding predictive region.

Also, we assume the availability of a road network graph, which allows us to index and query elements such as nodes and edges.

5.1. Handling Updates and Maintenance

We outline the pseudo-code for handling location updates in **algorithm 1**. This algorithm updates the historical uncertain regions and serves as a preliminary step for the pruning process of obsolete historical nodes from an object's trajectory. We take in as input- R, a set of historical regions representing the past, uncertain regions in the object's trajectory. Note,

the historical regions may be empty if we are receiving the first update from a given object. The second input T, denotes the number of steps or updates received thus far. Depending on the implementation, this could be a timestamp at which an update is received. This parameter is used to index the uncertain regions chronologically. Finally, we take in as input L, the imprecise location of the object. This parameter defines the object's latest imprecise location from which we are to derive a new uncertain region.

We begin by gathering all available nodes within the latest imprecise region. On the first location update, we consider all available nodes to be a possible location for the moving object. If the condition fails, a reference to all nodes within the previous region is created (line 6). Then we iterate through the previous nodes and create a set containing all of its children. In line 11, we reduce the set of nodes inside the current region by only considering nodes which have a direct connection to one or more nodes in the previous region. This is accomplished by a set intersection with the children of the previous region.

The loop in line 14 iterates over the nodes within the previous region and checks if it has a connection to one or more nodes in the latest region via applying a set intersection between a node's children and the latest nodes. If it has no children, then it is considered obsolete as it has no continuous path to the latest region; as such, it is added to a set of obsolete parent nodes. However, if a node from the previous region has a connection to the latest region, it is considered a valid parent; as such, it is added to a set of valid parents.

At this point, the nodes in the latest regions must contain references to their respective parent nodes. In line 22, we iterate over the nodes inside the latest region and apply a set intersection between all incoming edges to that node and the set of valid parents computed earlier. Finally, we add a new entry to the set of regions R to contain all the new nodes now that they have been filtered and contain parental references.

The set of obsolete parents at this point remain unused. This can be used to prune and further refine the previous regions and thereby narrow down the object's possible trajectory.

Algorithm 1 Managing Movement Updates

Input: Past Uncertain Regions R, Step T, New Imprecise Location L

```
1: currentNodes ← {n: node n ∈ L}
2: if T = 0 then
3:     R[T] ← currentNodes
4:     return R
5: end if
6: pastNodes ← R[T-1]
7: pastChildren ← ∅
8: for each pastNode ∈ pastNodes do
9:     pastChildren ∪ pastNode.Children
10: end for
11: currentNodes ← currentNodes ∩ pastChildren
12: obsoleteParents ← ∅
13: validParents ← ∅
14: for each pastNode ∈ pastNodes do
15:     pastNode.Children ← pastNode.Children ∩ currentNodes
16:     if | pastNode.Children | = 0 then
17:         obsoleteParents.Add(pastNode)
18:         continue
19:     end if
20:     validParents.Add(pastNode)
21: end for
22: for each currentNode ∈ currentNodes do
23:     currentNode.Parents ← currentNode.IncomingEdges ∩ validParents
24: end for
25: R[T] ← currentNodes
26: return R
```

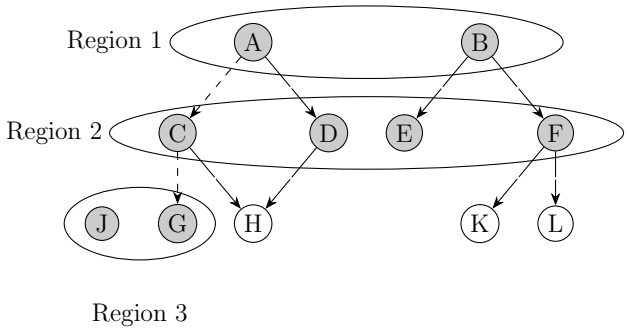


Figure 5.1: Managing uncertainty using edge connectivity and regional presence

5.2. Prune Past Regions

Pruning the past refines the past trajectory of a moving object as it travels from one uncertain region to another. This, in turn, helps us narrow down possible nodes representing an object's location.

To provide an intuitive explanation, we refer to Figure 5.1. The topmost level corresponds to the first received imprecise location, and nodes within are highlighted gray. The next level contains nodes which reside in the subsequent imprecise location. It is important to note that nodes within the second level are not only connected but also highlighted to signify their presence inside of the region. At this point, there is no pruning to be done since all nodes within the region also maintain continuity from nodes from the previous region. In the last level, we see that some nodes are highlighted and maintain continuity; in this case, node G. Some nodes maintain continuity but are not highlighted, meaning they are not within the imprecise region; in this case, nodes H, K, L. Finally, other nodes are highlighted but do not inherit edges from nodes within the previous region; in this case, node J. The algorithm described earlier would discard the nodes H, K, L since they would not be considered as we only look at nodes within the imprecise region (line 1). Also, node J would be discarded since we require continuity between regions (line 11). However, nodes D, E, F found in the

second level does not maintain continuity to the latest imprecise region. Thus, they would be added to the collection *obsoleteParents* and would need to be pruned. The dashed edges show what the trajectory may look like after pruning.

Below we describe an algorithm for recursively pruning obsolete nodes in previous regions. This algorithm takes as input a set of nodes N, which are to be pruned; an integer D denoting the depth of the region which is being pruned; set of regions R. The pseudo-code for this algorithm is given in **algorithm 2**.

Algorithm 2 Prune Past

Input: Set of obsolete nodes N, Depth D, Regions R

```
1: for each node n ∈ N do
2:     R[D].Remove(n)
3: end for
4: if D = 0 then
5:     return
6: end if
7: obsoleteParents ← ∅
8: for each node n ∈ N do
9:     for each parent p ∈ n.Parents do
10:        p.Children.Remove(n)
11:        if | p.Children | = 0 then
12:            obsoleteNodes.append(p)
13:        end if
14:    end for
15: end for
16: return PrunePast(obsoleteNodes, D-1, R)
```

We begin by removing references to the obsolete nodes from the given region. Then we check the base case, which tells us if we are at the topmost level or the first region. If the base case fails, we initialize a new set of obsolete nodes, which we will recursively prune after populating.

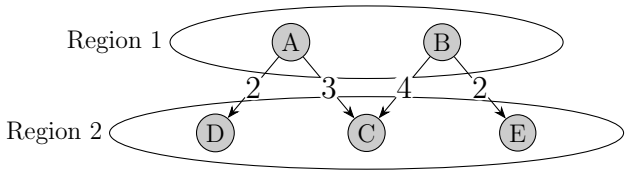


Figure 5.2: Region of depth 2 with edge weights applied

We then iterate through the obsolete nodes N and notify their parents to remove references. If the number of children for any of these parent nodes falls below 1, it must be considered for pruning as it implies an absence of a direct path between the two regions. Finally, we recursively prune our new set of obsolete nodes.

5.3. Historical Probability Computation

It is important to note that after pruning a node, the probabilities associated with its respective region must be updated. We do this in order to update any stale probabilities that may arise from the modification of the region; this is sensitive to the probability model used. Since our proposed probability model relies on a computed sum of edge weights, in the event of pruning, it must be recomputed, and probabilities need to be reassigned.

Before describing the algorithm for probability assignment, let us go over an example to provide intuition. In Figure 5.2 we describe a regional graph of depth two with assigned edge weights. Since the first region has no incoming nodes, the node probabilities are assigned uniformly. However, for the subsequent region, we may compute node probabilities using the weights of incoming edges. Note, we assume the weights are computed as described in equation 1.

We first have to compute the edge probabilities. According to equation 4, $S = 3 + 2 + 4 + 2 = 11$. Then, as described in equation 5, we compute $S' = 11(4 - 1) = 33$. Using these

two constants, we can compute the following edge probabilities as described in equation 7:

$$P(e_{A\to C}) = \frac{11-3}{33} \approx 0.24$$
$$P(e_{A\to D}) = \frac{11-2}{33} \approx 0.27$$
$$P(e_{B\to C}) = \frac{11-4}{33} \approx 0.21$$
$$P(e_{B\to E}) = \frac{11-2}{33} \approx 0.27$$

Following this, we can compute the following node probabilities using equation 10.:

$$P(C) = P(e_{A\to C}) + P(e_{B\to C}) \approx 0.45$$
$$P(D) = P(e_{A\to D}) \approx 0.27$$
$$P(E) = P(e_{B\to E}) \approx 0.27$$

These probabilities describe the likelihood of being at any of the nodes within a region at the time of the update. In this case, we computed the likelihood of our moving object being at the nodes in the second region at the time when we received the second update.

The pseudo-code for applying the historical probability model is described in **algorithm 3**. We take in as input the following: *Origin, Destination*, which are set of nodes denoting two consecutive regions, and ΔT, which is the time difference between the updates of the regions.

We begin by initializing an empty lookup table, which will store edge probabilities associated with each edge connecting the two regions *Origin, Destination*. Next, we gather all the connecting edges between the regions, which we then iterate over. On each iteration, we retrieve the estimated time taken to travel along an edge. Then we calculate the edge probability using equation 7 and store the probability associated with that edge into the lookup table defined above.

Next, we iterate over the nodes in *Destination*, and we gather the incoming edges that are associated with a particular node by applying set intersection with the connected edges between the two regions and the incoming edges to the particular node. The probabilities of these edges are then summed up. Finally, we return the region with updated probabilities.

Algorithm 3 Historical Probability Computation

Input: Region Origin, Region Destination, Time ΔT

1: $P_E \leftarrow \emptyset$
2: connectedEdges $\leftarrow$ GetConnectedEdges(Origin, Destination)
3: **for each** edge e $\in$ connectedEdges **do**
4: t $\leftarrow$ GetEdgeTravelTime(e)
5: w $\leftarrow |\Delta T - t|$
6: edgeProbability $\leftarrow$ CalculateEdgeProbability(w, connectedEdges)
7: P_E[e] $\leftarrow$ edgeProbability
8: **end for**
9: **for each** node n $\in$ Destination **do**
10: incomingEdges $\leftarrow$ $P_E \cap$ n.IncomingEdges
11: n.Probability $\leftarrow \sum_{e \in incomingEdges} P_E[e]$
12: **end for**
13: **return** Destination

5.4. Managing Predictive Trees

To facilitate predictive queries, we use a set of predictive trees [9] rooted at nodes within the latest imprecise region. The trees are then expanded to a user-defined depth, which denotes how many steps in the future they wish to predict. As the object moves along its trajectory and gets closer to the predicted step, we continue to refine the predictive trees.

The pseudo-code described below helps maintain and update the set of predictive trees as we receive location updates from the moving object(s). It takes in the following as input: a set of uncertain regions R, a lookup table containing 0 or more predictive trees, and the current number of steps or updates received thus far.

We start by gathering all the roots of the p-tree nodes inside the lookup table *PTrees*. Note, these trees represent the predictive trajectories rooted in the previously received imprecise location. Then, we gather all the nodes inside the latest imprecise region. In line 3, we check if *rootNodes* is empty, which would imply we are at the very first update in which case we would populate *PTrees* with all the available nodes in the latest region and expand

Algorithm 4 Managing Predictive Trees

Input: Regions R, Lookup Table PTrees, Step T

```
1: rootNodes ← GetRootNodes(PTrees)
2: nodes ← {n: node n ∈ Regions[T]}
3: if rootNodes = ∅ then
4:     for each node n ∈ nodes do
5:         PTrees[n] ← PredictiveTree(n)
6:     end for
7:     return PTrees
8: end if
9: validNodes ← GetChildren(rootNodes) ∩ nodes
10: PTrees ← ∅
11: for each node n ∈ validNodes do
12:     PTrees[n] ← PredictiveTree(n)
13: end for
14: return PTrees
```

them accordingly.

In the case we are not on the first update, in line 9, a set of *validNodes* is populated. This set contains all nodes in the latest region, which can be traveled to from the previous region. In other words, only the nodes with a direct connection to the previous imprecise location are considered to be valid. Then we re-initialize *PTrees* and populate it with new p-trees rooted at each of the valid nodes inside the newest uncertain region. Finally, *PTrees* is returned. Note, the expansion of the p-trees is dependent on the user and how many steps or time units in the future one may want to predict movements for.

Chapter 6
EXPERIMENTAL EVALUATION

In this chapter we explore the effectiveness of our work in terms of accuracy and efficiency through experimental results obtained using real GPS trajectories from Porto, Portugal.

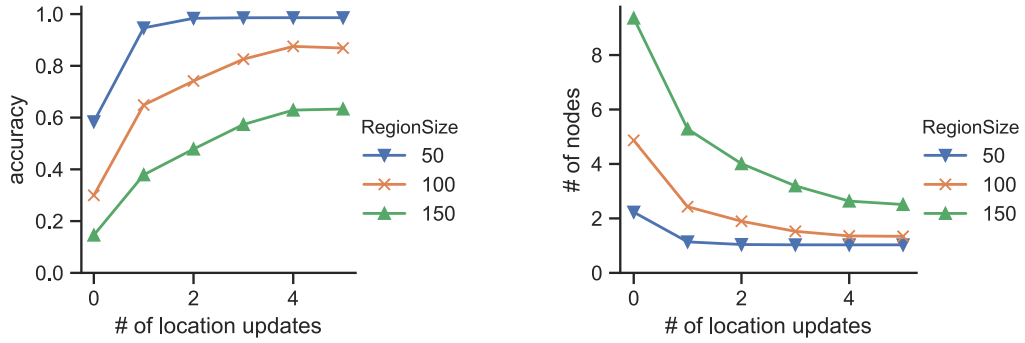
6.1. Experimental Setup

The moving object data used to run the experiments are that of taxi trajectories in Porto, Portugal. This dataset is publicly available on Kaggle.com [6]. We used road network data retrieved from OpenStreetMap, and map-matched the trajectories to fit the road network. The number of trajectories varies from 15,000 to 120,000, depending on the length of the trip. The algorithms were implemented in C# and was run on a machine with an i5-4670 and 8 GB of memory.

6.2. Historical Location Queries

We split the historical queries into two parts. One of which involves inferring a moving object's location at any time in the past. In our case, this past region is set to the 0^{th} region. In other words, we try to determine the moving object's origin. The second type of historical query, which we will call the present location query, determines an object's location in the latest uncertain region or at the time of its latest update.

To measure the accuracy of a given query, we retrieve all nodes within a region and their associated probabilities. Then, we retrieve the node representing the object's real location and report the probability as the measurement of accuracy. We run the queries for up to 5 location updates. Note, in the experiments, we publish location updates once the moving object has hopped to a different edge on the road network. The queries are also run over the



(a) accuracy rates for inferring the first location

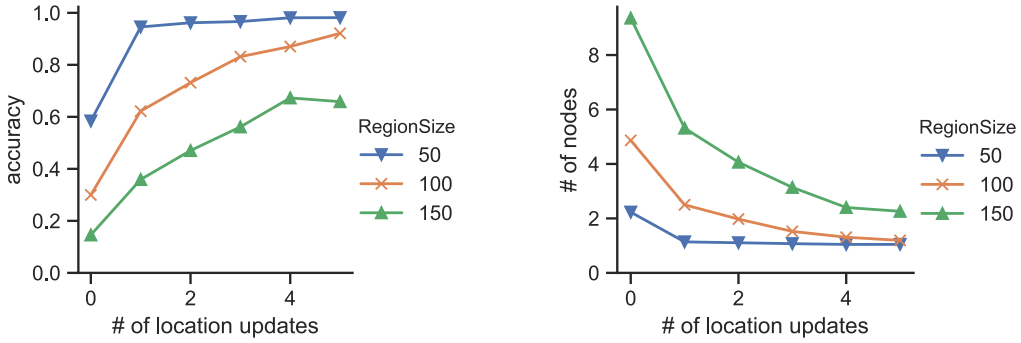
(b) number of nodes inside the first uncertain region

Figure 6.1: Inferring historical location

varying degree of uncertainty as defined by the region size/radius measured in meters.

In figure 6.1a, we can see the accuracy of our query results for historical queries. We see that the accuracy of the query increases with the number of updates because we continuously refine the past regions through pruning. We see a similar pattern in figure 6.1b where the number of nodes within a historical region decreases over time with more updates; again, this is due to pruning of obsolete nodes.

The second set of historical queries we evaluate are those for determining the object's latest location. This is part of the historical query processing because intuitively, we can say the object may have already traveled past the latest region by the time we receive its update. Unlike the previously mentioned historical query, we do not infer the object's location within a static region. Instead, with each update of an imprecise location, we try to determine the object's location within the latest region. As with the previous results, we see that the accuracy improves as we receive updates in figure 6.2a. Figure 6.2b shows the number of nodes in the region decreasing over time due to pruning, which aids the likelihood of estimating the object's location correctly.



(a) accuracy rates for inferring most recent location

(b) number of nodes inside the most recent uncertain region

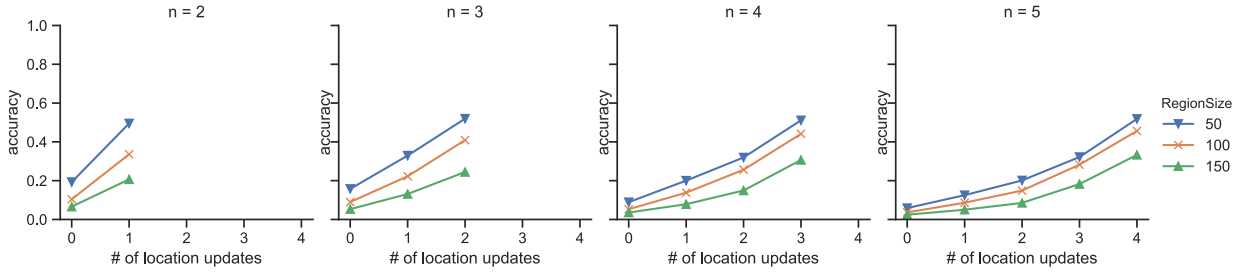
Figure 6.2: Inferring present location

With both of the historical queries running for five updates, we approach near 100% accuracy rate after the second or third update if our region size is 50 meters. For the largest region size, 150 meters, our accuracy approaches to about 60% after about four updates. The node counts help us visualize the effectiveness of the pruning method employed. Using the information gathered we can say that more nodes being pruned lead to a higher chance of estimating the object's actual location correctly.

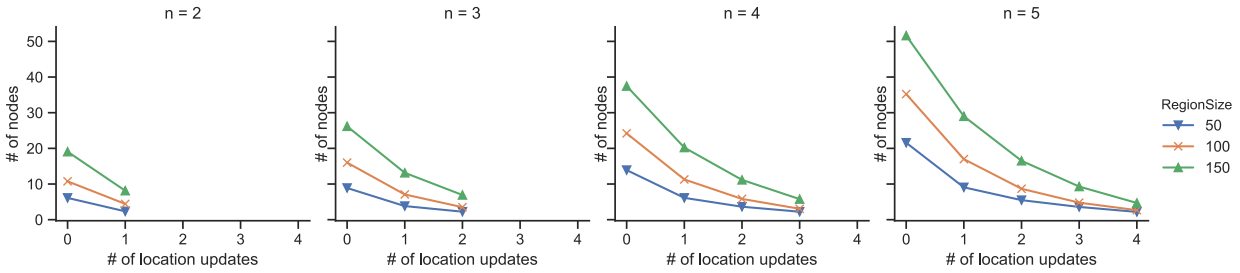
6.3. Predictive Location Queries

Figure 6.3a illustrates the accuracy of predictive queries of varying steps in the future. The accuracy is measured the same way described for historical evaluations. The predictive queries were carried out for 2 to 5 steps in the future from an object's initial region. The degree of uncertainty is determined by the region sizes denoting the radius of each uncertain region. As with our results above, smaller region sizes yield better accuracy. Also, as we travel closer to the predictive region, with subsequent updates, the accuracy increases.

In figure 6.3b we see the number of nodes in each subsequent predictive region decrease.



(a) accuracy rates for predicting n^{th} step into the future



(b) number of nodes at the n^{th} region with each update

Figure 6.3: Predicting n^{th} step into the future from origin

Since the predictive trees are rooted in the latest region, reduction of nodes through incremental pruning affects the number of predictive trees and their children.

For region size of 50 meters, our predictive queries achieve around 60% accuracy at one step before the predictive region. With a region size of 150 meters, this drops down to around 30%. The accuracy rate also suffers when we are predicting far ahead into the future because of the large number of leaf nodes in the predictive trees. This could be further improved through heuristics or other means such that the number of nodes in the predictive regions can be reduced.

6.4. Resource Overhead

To measure the resource overhead of our solution, we consider the time taken to rebuild the predictive regions, and update historical regions after each location update. We use the

trajectory data of about 15,000 to 120,000 trips. During the evaluations, we expanded the predictive regions to the depth as defined by the predictive step. For example, if we were predicting two steps into the future, at step 1, we would build predictive regions up to step 3, at step 3, we would build the predictive regions up to step 5 and so on.

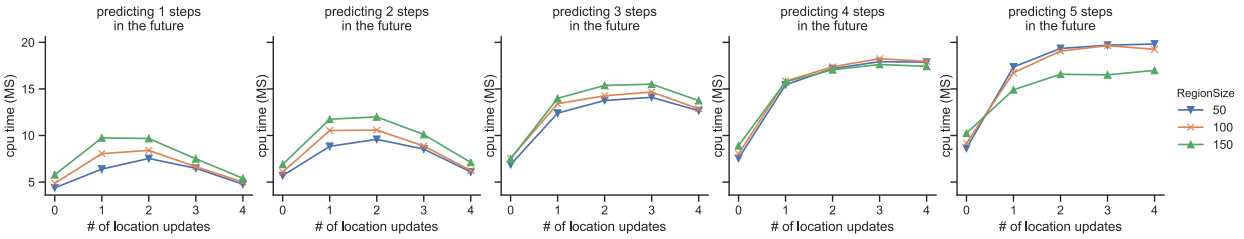


Figure 6.4: Time taken to update regional nodes and rebuild predictive trees

In figure 6.4, we see the average time taken to rebuild the uncertain regions increase as we receive first few location updates. The time needed to rebuild goes up as we increase the predictive depth. However, region size seems to have less of an effect. Also, the time seems to taper off for the instances with lower depths after a few updates. We assume this behavior is also present in the instances with higher depth as they receive more and more updates. As it takes a few rounds of updates to prune the number of nodes to make a significant impact, we assume this causes a decrease in rebuild time.

6.5. Summary

For the historical and present queries, we achieve very high accuracy for regions sizes 50 and 100 at over 90% after four rounds of updates. We can see the predictive queries follow a similar pattern. Though we do not achieve a high accuracy rate as we did for historical queries. From the resource usage evaluations, we can see that the solution is quite low cost in terms of time. Though it is important to note these metrics can always be improved since the probability models can be entirely user-defined, designed to take advantage of additional constraints and heuristics.

Chapter 7

CONCLUSION

Managing imprecise moving object locations has been an important problem and continues to be a challenge in location-aware services in general. In addition, the growth of GPS-enabled devices has increased the need for predictive analysis of moving objects for location-based recommendation systems, advertisements, ride sharing applications, and traffic management and prediction systems. In this thesis we propose an approach for querying moving objects' past, present, and future locations under uncertainty. Our approach can work with any arbitrary road network. The ability to inject user defined probability models according to one's needs makes our solution extensible. Scalability and efficiency of our proposed solution comes from the ability to significantly reduce the number of nodes in a given imprecise region. The experimental evaluations on real data demonstrate that our solution achieves near perfect results for historical queries with region sizes under 100 meters.

BIBLIOGRAPHY

[1] P. Banerjee, S. Ranu, and S. Raghavan. Inferring uncertain trajectories from partial observations. In *2014 IEEE International Conference on Data Mining*, pages 30–39, Dec 2014.

[2] Bertil Chapuis, Arielle Moro, Vaibhav Kulkarni, and Benoundefinedt Garbinato. Capturing complex behaviour for predicting distant future trajectories. In *Proceedings of the 5th ACM SIGSPATIAL International Workshop on Mobile Geographic Information Systems*, MobiGIS '16, page 64–73, New York, NY, USA, 2016. Association for Computing Machinery.

[3] J. Chen and R. Cheng. Efficient evaluation of imprecise location-dependent queries. In *2007 IEEE 23rd International Conference on Data Engineering*, pages 586–595, April 2007.

[4] R. Cheng, J. Chen, M. Mokbel, and C. Chow. Probabilistic verifiers: Evaluating constrained nearest-neighbor queries over uncertain data. In *2008 IEEE 24th International Conference on Data Engineering*, pages 973–982, April 2008.

[5] Zhiming Ding. Utr-tree: An index structure for the full uncertain trajectories of network-constrained moving objects. In *The Ninth International Conference on Mobile Data Management (mdm 2008)*, pages 33–40, 05 2008.

[6] ECML/PKDD. Taxi trip time prediction (ii) competition. https://www.kaggle.com/c/pkdd-15-taxi-trip-time-prediction-ii, April 2015.

[7] T. Emrich, H. Kriegel, N. Mamoulis, M. Renz, and A. Zufle. Querying uncertain spatio-temporal data. In *2012 IEEE 28th International Conference on Data Engineering*, pages 354–365, April 2012.

[8] Sébastien Gambs, Marc-Olivier Killijian, and Miguel Núñez del Prado Cortez. Next place prediction using mobility markov chains. In *Proceedings of the First Workshop on Measurement, Privacy, and Mobility*, MPM '12, New York, NY, USA, 2012. Association for Computing Machinery.

[9] Abdeltawab M Hendawi, Jie Bao, Mohamed F Mokbel, and Mohamed Ali. Predictive tree: An efficient index for predictive queries on road networks. *Fuzzy Sets and Systems*, 2009.

[10] Abdeltawab M. Hendawi and Mohamed F. Mokbel. Panda: A predictive spatio-temporal query processor. In *Proceedings of the 20th International Conference on Advances in Geographic Information Systems*, SIGSPATIAL '12, page 13–22, New York, NY, USA, 2012. Association for Computing Machinery.

[11] Y. Iijima and Y. Ishikawa. Finding probabilistic nearest neighbors for query objects with imprecise locations. In *2009 Tenth International Conference on Mobile Data Management: Systems, Services and Middleware*, pages 52–61, May 2009.

[12] Y. Ishikawa, Y. Iijima, and J. X. Yu. Spatial range querying for gaussian-based imprecise query objects. In *2009 IEEE 25th International Conference on Data Engineering*, pages 676–687, March 2009.

[13] Hoyoung Jeung, Qing Liu, Heng Tao Shen, and Xiaofang Zhou. A hybrid prediction model for moving objects. In *Proceedings of the 2008 IEEE 24th International Conference on Data Engineering*, ICDE '08, pages 70–79, Washington, DC, USA, 2008. IEEE Computer Society.

[14] Hoyoung Jeung, Man Lung Yiu, Xiaofang Zhou, and Christian S Jensen. Path prediction and predictive range querying in road network databases. *The VLDB Journal*, 19(4):585–602, 2010.

[15] Hassan A Karimi and Xiong Liu. A predictive location model for location-based services. In *Proceedings of the 11th ACM international symposium on Advances in geographic information systems*, pages 126–133. ACM, 2003.

[16] Mingxiao Li, Feng Lu, Hengcai Zhang, and Jie Chen. Predicting future locations of moving objects with deep fuzzy-lstm networks. *Transportmetrica A: Transport Science*, 0(0):1–18, 2018.

[17] Rui Li, Bir Bhanu, Chinya Ravishankar, Michael Kurth, and Jinfeng Ni. Uncertain spatial data handling: Modeling, indexing and query. *Computers & Geosciences*, 33(1):42–61, 2007.

[18] Wesley Mathew, Ruben Raposo, and Bruno Martins. Predicting future locations with hidden markov models. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*, UbiComp '12, page 911–918, New York, NY, USA, 2012. Association for Computing Machinery.

[19] Johannes Niedermayer, Andreas Züfle, Tobias Emrich, Matthias Renz, Nikos Mamoulis, Lei Chen, and Hans-Peter Kriegel. Probabilistic nearest neighbor queries on uncertain moving object trajectories. *arXiv preprint arXiv:1305.3407*, 2013.

[20] Shaojie Qiao, Changjie Tang, Huidong Jin, Teng Long, Shucheng Dai, Yungchang Ku, and Michael Chau. Putmode: prediction of uncertain trajectories in moving objects databases. *Applied Intelligence*, 33(3):370–386, Dec 2010.

[21] N. Samaan and A. Karmouch. A mobility prediction architecture based on contextual knowledge and spatial conceptual maps. *IEEE Transactions on Mobile Computing*, 4(6):537–551, Nov 2005.

[22] Jimeng Sun, Dimitris Papadias, Yufei Tao, and Bin Liu. Querying about the past, the present, and the future in spatio-temporal databases. In *Proceedings. 20th International Conference on Data Engineering*, pages 202–213. IEEE, 2004.

[23] Yufei Tao, Christos Faloutsos, Dimitris Papadias, and Bin Liu. Prediction and indexing of moving objects with unknown motion patterns. In *Proceedings of the 2004 ACM SIGMOD international conference on Management of data*, pages 611–622. ACM, 2004.

[24] Yufei Tao, Dimitris Papadias, and Jimeng Sun. The tpr*-tree: an optimized spatio-temporal access method for predictive queries. In *Proceedings of the 29th international conference on Very large data bases-Volume 29*, pages 790–801. VLDB Endowment, 2003.

[25] Yufei Tao, Xiaokui Xiao, and Reynold Cheng. Range search on multidimensional uncertain data. *ACM Trans. Database Syst.*, 32(3):15–es, August 2007.

[26] G. Trajcevski, A. Choudhary, O. Wolfson, L. Ye, and G. Li. Uncertain range queries for necklaces. In *2010 Eleventh International Conference on Mobile Data Management*, pages 199–208, May 2010.

[27] Goce Trajcevski, Roberto Tamassia, Hui Ding, Peter Scheuermann, and Isabel F. Cruz. Continuous probabilistic nearest-neighbor queries for uncertain trajectories. In *Proceedings of the 12th International Conference on Extending Database Technology: Advances in Database Technology*, EDBT '09, page 874–885, New York, NY, USA, 2009. Association for Computing Machinery.

[28] Simonas Šaltenis, Christian S. Jensen, Scott T. Leutenegger, and Mario A. Lopez. Indexing the positions of continuously moving objects. In *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*, SIGMOD '00, pages 331–342, New York, NY, USA, 2000. ACM.

[29] Simonas Šaltenis, Christian S. Jensen, Scott T. Leutenegger, and Mario A. Lopez. Indexing the positions of continuously moving objects. In *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*, SIGMOD '00, page 331–342, New York, NY, USA, 2000. Association for Computing Machinery.

[30] Andy Yuan Xue, Rui Zhang, Yu Zheng, Xing Xie, Jin Huang, and Zhenghua Xu. Destination prediction by sub-trajectory synthesis and privacy protection against such prediction. In *2013 IEEE 29th international conference on data engineering (ICDE)*, pages 254–265. IEEE, 2013.

[31] Meihui Zhang, Su Chen, Christian S. Jensen, Beng Chin Ooi, and Zhenjie Zhang. Effectively indexing uncertain moving objects for predictive queries. *Proc. VLDB Endow.*, 2(1):1198–1209, August 2009.

[32] Kai Zheng, Goce Trajcevski, Xiaofang Zhou, and Peter Scheuermann. Probabilistic range queries for uncertain trajectories on road networks. In *Proceedings of the 14th International Conference on Extending Database Technology*, EDBT/ICDT '11, page 283–294, New York, NY, USA, 2011. Association for Computing Machinery.