Attraction and Avoidance Detection from Movements

Zhenhui Li[*]
Pennsylvania State University

Bolin Ding
Microsoft Research

Fei Wu
Pennsylvania State University

Tobias Kin Hou Lei
University of Illinois at Urbana-Champaign

Roland Kays[†]
North Carolina Museum of Natural Sciences

Margaret C. Crofoot[‡]
University of California, Davis

ABSTRACT

With the development of positioning technology, movement data has become widely available nowadays. An important task in movement data analysis is to mine the relationships among moving objects based on their spatiotemporal interactions. Among all relationship types, attraction and avoidance are arguably the most natural ones. However, rather surprisingly, there is no existing method that addresses the problem of mining significant attraction and avoidance relationships in a well-defined and unified framework.

In this paper, we propose a novel method to measure the significance value of relationship between any two objects by examining the background model of their movements via permutation test. Since permutation test is computationally expensive, two effective pruning strategies are developed to reduce the computation time. Furthermore, we show how the proposed method can be extended to efficiently answer the classic threshold query: given an object, retrieve all the objects in the database that have relationships, whose significance values are above certain threshold, with the query object. Empirical studies on both synthetic data and real movement data demonstrate the effectiveness and efficiency of our method.

1. INTRODUCTION

Rapid advances of sensors, wireless networks, GPS, satellites, smart-phone, and web technologies have provided us with tremendous amount of time- and space-related data. Mining patterns from spatiotemporal data has many important applications in human mobility understanding, smart transportation, urban planning, biological studies, environmental and sustainability studies.

[*]The corresponding author, email: JessieLi@ist.psu.edu

[†]This author is also affiliated with North Carolina State University.

[‡]This author is also affiliated with Max Planck Institute for Ornithology and Smithsonian Tropical Research Institute.

Proceedings of the VLDB Endowment, Vol. 7, No. 3

An important and interesting question people often ask about movement data is: What is the relationship between two moving objects based on their spatiotemporal interactions? Relationships between two moving objects can be classified as attraction, avoidance or neutral. In an *attraction* relationship, the presence of one individual causes the other to *approach* (*i.e.*, reduce the distance between them). As a result, the individuals have a higher probability to be spatially close than expected based on chance. On the other hand, in an *avoidance* relationship, the presence of one individual causes the other to *move away*. So the individuals have a lower probability to be spatially close than expected. Finally, with a *neutral* relationship, individuals *do not alter* their movement patterns based on the presence (or the absence) of the other individual. So the probability that they are being spatially close is what would be expected based on independent movements.

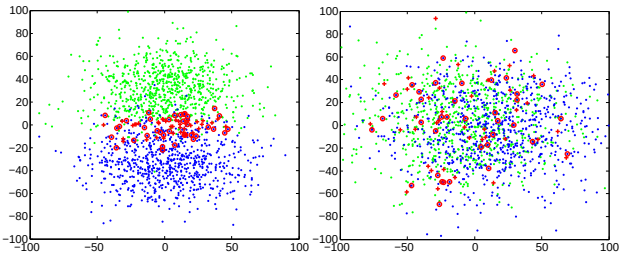
The attraction relationship is commonly seen, for example, in animal herds or human groups (*e.g.*, colleague and family). In addition, the avoidance relationship also naturally exists among moving objects. In animal movements, prey try to avoid predators, and different animal groups of the same species tend to avoid each other. Even in the same group, subordinate animals often avoid their more dominant group-mates. In human movements, criminals in the city try to avoid the police, whereas drug traffickers traveling on the sea try to avoid the patrol.

In real applications, however, people often want to know more than just the relationship type. Given the answer to the previous question (*i.e.*, attraction, avoidance, or neutral), people may immediately ask: What is the *degree* of the relationship? In other words, we need to know the confidence in a given type of relationship. To answer such problem, in principle, one needs to test all possible hypotheses and examine the statistical significance of each hypothesis.

In the literature, study of moving object relationship has been largely restricted to attraction relationship only. In particular, various measures [30, 29, 6, 5], such as Euclidean distance and dynamic time warping, have been proposed to calculate the distance between two trajectories. Meanwhile, moving object clusters, such as flock [1], convoy [16], and swarm [21], are detected by counting the frequency of objects being spatially close, i.e., the *meeting frequency*. All these studies make a common assumption that, the smaller the distance is or the higher the meeting frequency is, the stronger the attraction relationship is.

Unfortunately, as we will see soon, such assumption is often violated in real movement data. Consequently, none

of the existing work can provide a definite answer to our questions regarding the *type and degree* of moving object relationships. For example, two animals may be observed to be spatially close for 10 out of 100 timestamps. But is this significant enough to determine the attraction relationship? Further, another two animals are within spatial proximity for 20 out of 100 timestamps. Does this mean that the latter pair has a more significant attraction relationship than the former pair? Finally, if two animals are never being spatially close, do they necessarily have an avoidance relationship? We use the following example to further illustrate this problem.



(a) Attraction (b) Avoidance

Figure 1: Attraction and avoidance relationships

EXAMPLE 1. *In Figure 1(a), the green points and blue points show the locations of two moving objects, respectively. The red points indicate the locations at which the two objects are spatially close at the same time. There are 40 co-locating (red) points, which means that the meeting frequency is 40. As we can see from Figure 1(a), these two objects have their own territories but are* attracted *to meet in the overlapped region. In Figure 1(b), we show another pair of moving objects which also meet 40 times in the same period of time. However, since these two objects share the same territories, they are expected to meet more often than 40 times. Therefore, they are likely to have an* avoidance *relationship. In the real world, Figure 1(a) may correspond to two monkeys who are attracted by time-specific food resources, thus show up in the same region at the same time. Figure 1(b) could be the case where a wolf and a deer share the same territory but the dear is trying to avoid the wolf. Comparing Figure 1(a) and Figure 1(b), we see that meeting frequency is not a good measure for attraction and avoidance relationships.*

Motivated by the example above, we argue that it is necessary to look into the *background territories* to mine relationships between two objects. In other words, we detect the relationships through the *comparison* between how frequent two objects are *expected* to meet and the *actual* meeting frequency they have. Intuitively, if the actual meeting frequency is smaller (or larger) than the expectation, the relationship is likely to be avoidance (or attraction).

However, such comparison does not tell us *the degree of a relationship*. To evaluate the significance value of the relationship, we propose to use *permutation test*, a popular non-parametric approach to performing hypothesis tests and constructing confidence intervals. In our problem, the null hypothesis is that two movement sequences are independent. Under this hypothesis, if we randomly shuffle orders in the movement sequence, the meeting frequency should remain a similar value.

In addition, efficient discovery of significant relationships in a large moving object database is a nontrivial task. The major challenge lies in the *exponential number* of permutations needed to calculate the significant value (*i.e.*, p-value). In fact, obtaining the exact significance value is a #P-hard problem. Nevertheless, we observe that, in practice, the significance value converges quickly after a few hundred rounds of permutations. In this paper, we further design two pruning rules that greatly speed up the permutation test on real data.

Finally, we use the proposed method to address a classic threshold query: given one query object, retrieve the objects that have relationships, whose significance values are above certain threshold, with the query object. A straightforward solution to this problem is to compute the significance value for each object and check whether it is above the threshold. But this could be time-consuming because of the large number of moving objects. We design an adaptive algorithm which reduces the number of permutations needed while providing the same accuracy guarantee for the answers.

In summary, the contributions of the paper are as follows.

- We propose a novel and unified framework to mine significant attraction and avoidance relationships among moving objects.
- Since computing significance value is a #P-hard problem, we design an approximate counting algorithm and provide the approximation ratio given a limited number of permutations. Pruning techniques are further developed to speed up the permutation test.
- We propose an adaptive algorithm to efficiently answer threshold queries and retrieve significant relationships.
- The effectiveness and efficiency of our methods are demonstrated on both real and synthetic moving object databases.

The remainder of the paper is organized as follows. We formally define our problem in Section 2, and introduce the permutation-based method for computing significance value in Section 3. Threshold query processing is discussed in Section 4. Experimental results on both synthetic and real datasets are shown in Section 5. Finally, we describe the related work in Section 6, discuss future work in Section 7, and conclude our study in Section 8.

2. A NEW MEASURE OF RELATIONSHIP

2.1 Preliminaries

We have m *moving objects* $\mathcal{D} = \{o^1, o^2, \ldots, o^m\}$. The *trajectory* of a moving object o^i can be represented as a sequence of locations each associated with a timestamp: $\text{traj}^i = (\text{loc}_1^i, t_1^i)(\text{loc}_2^i, t_2^i)\cdots(\text{loc}_n^i, t_n^i)$. Each *location* loc_k^i could be a two-dimensional vector of longitude and latitude or a vector from a multi-dimensional feature space. For simplicity of presentation, we assume the tracking time of all the trajectories are synchronized and have the same number of tracking timestamps as n, that is $t_k^i = t_k^j$, $\forall i, j \in \{1, \ldots, m\}$ and $\forall k \in \{1, \ldots, n\}$. From now on, the i-th trajectory is denoted as $\text{traj}^i = \text{loc}_1^i \text{loc}_2^i \cdots \text{loc}_n^i$.

We will focus on two moving objects when defining the relationship. So we simplify the input as two trajectories, $R = r_1 r_2 \cdots r_n$ and $S = s_1 s_2 \cdots s_n$, where r_i and s_i are the i^{th} location in trajectories R and S, respectively.

We then define the *distance* between two locations r_i and s_j: distance(r_i, s_j). In spatial databases, the distance function could be the Euclidean distance between two spatial points, or the graph distance between two nodes in the transportation network. More generally, a location could also be a vector from a multi-dimensional feature space, and then the distance function may be defined as the L^p distance between two vectors.

An intuitive measure of the correlation between two moving objects is the frequency of their co-occurrences within certain distance threshold d. That is, when the locations of two objects are within distance d at certain timestamp, they are said to *meet* each other. Let $\tau(r_i, s_j)$ be the indicator of whether two locations are within distance d:

$$\tau(r_i, s_j) = \begin{cases} 1, & \text{distance}(r_i, s_j) \leq d; \\ 0, & \text{otherwise.} \end{cases}$$

We define the meeting frequency as follows.

DEFINITION 2.1. *(Meeting Frequency) The meeting frequency between R and S is defined as the number of timestamps when their spatial locations are within distance d:*

$$\text{freq}(R, S) = \sum_{i=1}^{n} \tau(s_i, r_i).$$

The value of proximity threshold d varies for different types of moving objects and for different conditions. For example, humans and animals may have quite different proximity values. Some animals may sense other animals even when the distance is over 100 meters; but the proximity value may be much smaller for humans to be considered as "being together". Even for the same type of moving objects, there could be different levels of proximity which all make sense in different scenarios. For example, people within 10 feet could have very close contact, such as family members living together or friends hanging out together. But people within 100 feet could also have loose contact, such as attending the same football game or traveling together on a train. When setting d at lower value, we are giving a more strict definition toward "being together". On the other hand, when d is getting larger, such definition of "being together" is getting looser.

2.2 Probabilistic Background Model

As discussed in Section 1, it is inappropriate to directly measure the relationship between two trajectories R and S using meeting frequency. In this section, we introduce a *probabilistic background model* to measure to what degree two objects (their trajectories) attract/avoid each other.

We use *permutation test* to estimate the probabilistic background model. The permutation test is a model-free and computationally-intensive statistical technique for hypothesis testing [26]. The distribution of the test statistic under the null hypothesis is obtained by calculating all possible values of the test statistic under rearrangements of the labels on the observed data points. The value of the observed data points is then compared to the distribution of the test statistic. If it is significantly high (or low) in this distribution, the observed value will be deemed significant.

In our problem, the null hypothesis is that two movement sequences R and S are independent. Under this hypothesis, if we randomly shuffle orders in the movement sequence, the meeting frequency between random permutations of trajectories R and S should remain similar value. If the meeting frequency between R and S is higher or lower than certain percentage (*e.g.*, 95%) of the randomized results, we reject the hypothesis and claim that R and S have significantly non-independent relationship (*i.e.*, attraction or avoidance).

To be more specific, let σ and σ' denote two (independent) random permutations of sequence $\{1, 2, \ldots, n\}$. The meeting frequency of randomly permuted trajectory sequences $\sigma(R)$ and $\sigma'(S)$ is $\text{freq}(\sigma'(R), \sigma(S))$. In addition, assuming the distributions of the two random permutations are independent and both uniform, computing the frequency between the two randomly permuted sequences is essentially the same as computing the frequency between one fixed sequence, R, and one random sequence, $\sigma(S)$. This is because the probability distribution of $\text{freq}(\sigma'(R), \sigma(S))$ is exactly the same as the one of $\text{freq}(R, \sigma(S))$, as formalized in Lemma 1. So we will focus on the distribution of $\text{freq}(R, \sigma(S))$ later on.

LEMMA 1. *Let σ and σ' be two independent uniformly random permutations of sequence $(1, 2, \ldots, n)$. For any two trajectories, $R = r_1 r_2 \cdots r_n$ and $S = s_1 s_2 \cdots s_n$, we have*

$$\begin{aligned} \forall y: \ \Pr\left[\text{freq}(\sigma'(R), \sigma(S)) = y\right] &= \Pr\left[\text{freq}(\sigma'(R), S) = y\right] \\ &= \Pr\left[\text{freq}(R, \sigma(S)) = y\right]. \end{aligned}$$

PROOF. The proof is directly from the symmetry. □

We note that the measures and methods developed in this work can also be generalized to the cases when σ and σ' are drawn from non-uniform distributions. But a thorough discussion on this issue is beyond the scope of this paper.

2.3 Avoidance and Attraction Relationships

Let $\mathcal{F} = \{\text{freq}(R, \sigma(S)) \mid \sigma\}$ be the multiset of all randomized meeting frequencies, we aim to define the relationship between two moving objects both *qualitatively* and *quantitatively*. We first describe how to measure *the degree or the significance* of relationship between two moving objects.

DEFINITION 2.2. (Significance Value of Relationships) *We define the significance value of attraction (or avoidance) between two moving objects R and S as the fraction of values in $\mathcal{F}$ which are* smaller *(or* larger*) than the actual meeting frequency* $\text{freq}(R, S)$*, plus one half of the fraction of values in $\mathcal{F}$ which are* equal *to* $\text{freq}(R, S)$*, and denote it as* $\text{sig}_{\text{attract}}(R, S)$ *(or* $\text{sig}_{\text{avoid}}(R, S)$*). That is,*

$$\begin{aligned} \text{sig}_{\text{attract}}(R, S) &= \Pr\left[\text{freq}(R, S) > \text{freq}(R, \sigma(S))\right] \\ &\quad + \frac{1}{2}\Pr\left[\text{freq}(R, S) = \text{freq}(R, \sigma(S))\right], \\ \text{sig}_{\text{avoid}}(R, S) &= \Pr\left[\text{freq}(R, S) < \text{freq}(R, \sigma(S))\right] \\ &\quad + \frac{1}{2}\Pr\left[\text{freq}(R, S) = \text{freq}(R, \sigma(S))\right]. \end{aligned}$$

Here, the cases where $\text{freq}(R, S) = \text{freq}(R, \sigma(S))$ contribute equally to $\text{sig}_{\text{attract}}(R, S)$ and $\text{sig}_{\text{avoid}}(R, S)$. Obviously, for any trajectories R and S, we have

$$\text{sig}_{\text{avoid}}(R, S) = 1 - \text{sig}_{\text{attract}}(R, S).$$

Based on the above definition of significance value, we now provide our definition of significant attraction and avoidance relationships with respect to a user-defined significance value threshold Λ.

DEFINITION 2.3. (Significant Attraction/Avoidance) *Two moving objects R and S are said to have an* attraction *(or* avoidance*) relationship if the significance value of attraction (or avoidance) is greater than a user-defined threshold Λ:* $\text{sig}_{\text{attract}}(R,S) > \Lambda$ *(or* $\text{sig}_{\text{avoid}}(R,S) > \Lambda$*).*

Threshold Λ is typically very close to 1 (*e.g.*, 0.95). Also, it is obvious that Λ is meaningful only when $\Lambda > 0.5$. Consequently, two objects are said to have no relationship (independent) if they have neither an attraction relationship nor an avoidance relationship.

2.3.1 An Alternative Way to Define Relationships

Before proceeding, it is worth noting that an alternative way to define the attraction and avoidance relationships is to compare the actual meeting frequency $\text{freq}(R,S)$ with the *expected meeting frequency* $\text{E}[\text{freq}(R,\sigma(S))]$. If the actual meeting frequency $\text{freq}(R,S)$ is less than the expected meeting frequency $\text{E}[\text{freq}(R,\sigma(S))]$, the moving objects are likely to have an avoidance relationship, and vice versa.

One advantage of using expected meeting frequency is that, unlike the significance value, it can be easily computed, according to the following lemma.

LEMMA 2. *The* expected meeting frequency *is:*

$$\text{E}[\text{freq}(R,\sigma(S))] = \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{n}\tau(r_i,s_j).$$

PROOF. Let y_i be the indicator of whether r_i meets the corresponding point in $\sigma(S)$ at timestamp i. Then

$$\text{E}[\text{freq}(R,\sigma(S))] = \sum_{i=1}^{n}\text{E}[y_i] = \sum_{i=1}^{n}\sum_{j=1}^{n}\frac{1}{n}\tau(r_i,s_j),$$

based on the linearity of expectation. □

However, by comparing the actual meeting frequency with the expected meeting frequency, one cannot determine a universal *degree of relationship*. Indeed, one can set a threshold for the difference between actual and expected meeting frequency to define attraction/avoidance relationships (as in Definition 2.3), but such thresholds are highly problem-dependent and have no statistical implication.

To remedy this issue, in [9], Doncaster proposes to perform a χ^2 test to measure the statistical significance of the difference between the actually meeting frequency and the expected meeting frequency, assuming a binomial probability model for the meeting frequency under the null hypothesis. Instead of making such assumption, our method directly computes the significance value of relationships by enumerating all possible permutations.

2.4 Problem Definition

In the rest of this paper, we will focus on the following two problems: (1) computing the significance value of relationship between two trajectories, and (2) finding all trajectories in a database which have significant relationships with respect to a given query trajectory.

PROBLEM 1. (Measuring Significant Relationships) *Given two trajectories R and S, our goal is to compute the significance value of relationship between them* $\text{sig}_{\text{attract}}(R,S)$ *or* $\text{sig}_{\text{avoid}}(R,S)$*, based on which we can determine if they have an significant attraction or avoidance relationship.*

PROBLEM 2. (Querying Significant Relationships) *In a database $\mathcal{D}$ of trajectories, for a user-given threshold Λ of significance and a query trajectory Q, to determine which trajectories in $\mathcal{D}$ significantly attract or avoid Q.*

Since computing the probability distribution of the background model is not trivial (see Theorem 1 in Section 3.1), it is challenging to answer the above two questions. In the following two sections, we discuss how to solve these two problems both effectively and efficiently, respectively.

3. COMPUTING SIGNIFICANCE VALUE

To determine if a significant attraction or avoidance relationship exists, we need to compute the significance value of a relationship. In this section, we first show that the computation of the true significance value is hard, and then develop an efficient approximate counting algorithm to estimate it.

3.1 Hardness of Computing Significance Value

We now prove that computing the exact significance value is a #P-hard problem. Simply put, #P-hardness for counting problems is an analogy to NP-hardness for decision problems. For a #P-hard problem, there is unlikely to be any efficient (polynomial-time) algorithm which solves it exactly.

THEOREM 1. *The problem of computing* $\text{sig}_{\text{avoid}}(R,S)$ *or* $\text{sig}_{\text{attract}}(R,S)$ *is #P-hard.*

PROOF. We reduce the problem of *counting perfect matchings in bipartite graph* to the problem of computing $\text{sig}_{\text{attract}}$. As counting perfect matchings is a #P-complete problem [28], the proof of our hardness result can be completed.

Consider a bipartite graph $G(U,V,E)$, where U and V are the vertex sets each having size n, and $E \subseteq U \times V$ is the edge set. From $G(U,V,E)$, create two sets of locations U and V, s.t. for any $u \in U$ and $v \in V$, $uv \in E$ if and only if $\text{distance}(u,v) \leq d$, *i.e.*, $\tau(u,v) = 1$ (note: the two sets of locations created here might be from a high-dimensional space). Let R and S be the orderings of U and V, respectively, s.t. $r_i s_i \in E$. In other words, $\{r_i s_i \mid i \in \{1,2,\ldots,n\}\}$ is a perfect matching in $G(U,V,E)$. We have $\text{freq}(R,S) = n$, and from the definition of significance value $\text{sig}_{\text{attract}}$,

$$\#\text{perfect matchings in } G = 2\cdot(1-\text{sig}_{\text{attract}}(R,S))\cdot n!$$

So the proof is completed. □

3.2 Randomized Approximation Counting

Since computing the significance value $\text{sig}_{\text{attract}}(R,S)$ or $\text{sig}_{\text{avoid}}(R,S)$ is #P-hard, we now focus on approximate counting algorithms.

3.2.1 Basic Monte Carlo Estimator Algorithm

Our counting algorithms are based on the following basic Monte Carlo scheme. Let U be a finite set of known size, and $G \subseteq U$ be a subset of unknown size. The objective is to estimate the ratio $\rho = |G|/|U|$. The classical Monte Carlo scheme works as follows: choose N independent (uniformly distributed) samples from U, denoted by $u_1, u_2, \ldots, u_N$, and let $Y_i = 1$ if $u_i \in G$, and 0 otherwise, $\forall i = 1,\ldots,N$. The ratio ρ is estimated as

$$\hat{\rho} = \sum_{i=1}^{N}\frac{Y_i}{N}.$$

LEMMA 3. *(Estimator Theorem) [25] Assuming $\rho \geq 0.5$, the above Monte Carlo algorithm yields an ϵ-approximation to ρ, i.e.,*

$$(1-\epsilon)\rho \leq \hat{\rho} \leq (1+\epsilon)\rho$$

with probability at least $1-\delta$, provided $N \geq \frac{4}{\epsilon^2 \rho} \ln \frac{2}{\delta}$.

In Lemma 3, we can assume that $\rho \geq 0.5$, because it is obvious that $|G|/|U| + |U-G|/|U| = 1$. So estimating $\rho = |G|/|U|$ is the same as estimating $|U-G|/|U|$. The number of permutations N depends on the larger value of $|G|/|U|$ and $|U-G|/|U|$.

One issue in the above Monte Carlo algorithm is that the number of samples needed, N, is dependent on the real value ρ itself. Below we show how this issue can be easily fixed when applying the algorithm to our problem.

3.2.2 Computing Significance Value

A direct way of applying the above basic Monte Carlo algorithm is to let U be the set of all permutations, G_1 be the subset of permutations $\{\sigma \mid \text{freq}(R,S) < \text{freq}(R,\sigma(S))\}$, and G_2 be permutations $\{\sigma \mid \text{freq}(R,S) = \text{freq}(R,\sigma(S))\}$. We can apply Lemma 3 to estimate $\rho_1 = |G_1|/|U|$ and $\rho_2 = |G_2|/|U|$. Then, $\text{sig}_{\text{avoid}}(R,S) = \rho_1 + \frac{1}{2}\rho_2$. However, based on Lemma 3, the number of samples needed to estimate $\text{sig}_{\text{avoid}}(R,S)$ accurately depends on its value (or ρ_1 and ρ_2), which is unknown yet. So we develop the following alternative algorithm which estimates at least one of $\text{sig}_{\text{avoid}}(R,S)$ and $\text{sig}_{\text{attract}}(R,S)$ accurately while the number of necessary samples is independent of their values.

ApproxCount(R,S,N)
1: Randomly select $N = \frac{8}{\epsilon^2}\ln\frac{2}{\delta}$ permutations $\sigma_1,\ldots,\sigma_N$.
2: Let $Y_< = |\{\sigma_i \mid \text{freq}(R,S) < \text{freq}(R,\sigma_i(S))\}|$
$\quad + \frac{1}{2}|\{\sigma_i \mid \text{freq}(R,S) = \text{freq}(R,\sigma_i(S))\}|$.
3: Let $Y_> = |\{\sigma_i \mid \text{freq}(R,S) > \text{freq}(R,\sigma_i(S))\}|$
$\quad + \frac{1}{2}|\{\sigma_i \mid \text{freq}(R,S) = \text{freq}(R,\sigma_i(S))\}|$.
4: Output $Y_</N$ as estimation of $\text{sig}_{\text{avoid}}(R,S)$
5: and $Y_>/N$ as estimation of $\text{sig}_{\text{attract}}(R,S)$.

Algorithm 1: Computing approximate significance value

THEOREM 2. *Let $N = \frac{8}{\epsilon^2}\ln\frac{2}{\delta}$ in* ApproxCount(R,S,N)*: if $\text{sig}_{\text{avoid}}(R,S) \geq 0.5$, then $Y_</N$ is an ϵ-approximation of $\text{sig}_{\text{avoid}}(R,S)$ with probability $1-\delta$; and if $\text{sig}_{\text{attract}}(R,S) \geq 0.5$, then $Y_>/N$ is an ϵ-approximation of $\text{sig}_{\text{attract}}(R,S)$ with probability $1-\delta$.*

PROOF. We only prove the case when $\text{sig}_{\text{avoid}}(R,S) \geq 0.5$. Similar argument applies to the case when $\text{sig}_{\text{attract}}(R,S) \geq 0.5$. Define U as the set of all permutations, and

$$G_1 = \{\sigma \mid \text{freq}(R,S) < \text{freq}(R,\sigma(S))\}.$$

Let's first assume that $G_2 = \{\sigma \mid \text{freq}(R,S) = \text{freq}(R,\sigma(S))\} = \emptyset$. Then, we have $\text{sig}_{\text{avoid}}(R,S) = |G_1|/|U|$. From Lemma 3 that, since $\text{sig}_{\text{avoid}}(R,S) \geq 0.5$, $Y_</N$ is an ϵ-approximation of $\text{sig}_{\text{avoid}}(R,S)$.

When $G_2 \neq \emptyset$, the analysis is more involved and we only provide an outline here. First, we color half of permutations in G_2 as red, denoted as G_2^-, and the other half as green, denoted as G_2^+. Then, we can rewrite $\text{sig}_{\text{avoid}}(R,S)$ as $|G_1 \cup G_2^-|/|U|$. If the coloring is known in advance, directly sampling from $G_1 \cup G_2^-$ and applying Lemma 3 yield the desired bound. However, in practice, we do not know the coloring. Nevertheless, one can show that the estimator in ApproxCount (*i.e.*, $Y_</N$) is at least as good as this coloring estimator, hence the proof is completed. □

3.3 An Efficient Algorithm

In this section, we describe how to speed up the basic ApproxCount algorithm. Given trajectories R and S, the time complexity of ApproxCount is $O(N \cdot n)$, since it uses N random permutations to calculate the significance value, and takes $O(n)$ to generate a permutation σ for a sequence with length n and compare $\text{freq}(R,S)$ with $\text{freq}(R,\sigma(S))$. When we need to compute significance value for every pair in a dataset consisting of m trajectories, the time complexity is $O(m^2 \cdot N \cdot n)$. Such time complexity could be too high for some real applications when the length n of each trajectory or the number of trajectories m is large.

We observe that, while the number of permutations N required in ApproxCount needs to be large enough to provide accuracy guarantee, the major bottleneck of the complexity lies in testing whether $\text{freq}(R,S)$ is smaller/equal/larger than $\text{freq}(R,\sigma(S))$, *i.e.*, lines 2-3 in ApproxCount, for each random permutation σ. We propose two pruning techniques below that can greatly speed up this test. The key idea is that, as we only care about the *comparison* result ($<$, $>$, or $=$) between $\text{freq}(R,S)$ and $\text{freq}(R,\sigma(S))$, instead of the actual value of $\text{freq}(R,\sigma(S))$, in many cases, it is *not* necessary to generate the complete permutation sequence $\sigma(S)$ and/or compute the *exact* value of $\text{freq}(R,\sigma(S))$. So we propose to use Knuth shuffle to generate a permutation and stop shuffling as soon as the comparison result is clear.

The function Compare is presented in Algorithm 2 with the two pruning techniques. It outputs $<$, $>$, or $=$ as the comparison result between $\text{freq}(R,S)$ and $\text{freq}(R,\sigma(S))$ (used in lines 2-3 of ApproxCount).

Pruning I: Eliminating non-overlapping locations

In real scenarios, we observe that most moving object pairs have *small portions of overlapping locations*. A location r_i in R is said to *overlap* with trajectory S if there exists a location s_j in S *s.t.* the distance between r_i and s_j is less than the distance threshold d (*i.e.*, $\tau(r_i,s_j)=1$). Let $R' \sqsubseteq R$ denote the *maximal subsequence* $r'_1 r'_2 \cdots r'_{n'}$ of R, where each r'_i is a location in R that overlaps with S, and $|R'| = n'$.

Recall that σ is a random permutation, or in other words, a one-to-one mapping $[n] \to [n]$ (where $[n] = \{1,2,\ldots,n\}$). Let σ' be the first n' elements of each random permutation σ, *i.e.*, a one-to-one mapping $[n'] \to [n]$. We have that *the probability distribution of* $\{\text{freq}(R,\sigma(S)) \mid \sigma\}$ *is identical to the one of* $\{\text{freq}(R',\sigma'(S)) \mid \sigma\}$. The reason is obvious: a point in the set $R - R'$ will not have any impact on the meeting frequency $\text{freq}(R,\sigma(S))$, so we only need to consider the points in R'. Therefore, comparing $\text{freq}(R,S)$ and $\text{freq}(R,\sigma(S))$ in lines 2-3 of algorithm ApproxCount can be replaced with comparing $\text{freq}(R,S)$ and $\text{freq}(R',\sigma'(S))$. And since the size of R' is usually much smaller than that of R (*i.e.*, $n' \ll n$) in real data, this pruning technique can greatly save the computation time.

Pruning II: Early termination conditions

Another pruning technique is to early stop a permutation test. Let $R'_{[i]}$ denote the first i elements in R', and let $\sigma'_{[i]}$ denote the first i elements of σ'. Obviously, if $\text{freq}(R'_{[i]},\sigma'_{[i]}(S))$

Compare(R, S)

1: $R' \leftarrow r'_1 r'_2 \cdots r'_{n'}$, *s.t.* $\forall r'_i \in R\ \exists s_j \in S:\ \tau(r'_i, s_j) = 1$
2: freq $\leftarrow 0$
3: **for** $i = 1 \rightarrow n$ **do** $\sigma'(i) = i$
4: **for** $i = 1 \rightarrow |R'|$ **do**
5: rand$_i \leftarrow$ a random number in $[i, n]$
6: Switch $\sigma'(i)$ and $\sigma'($rand$_i)$
7: **if** $\tau(r'_i, s_{\sigma'(i)}) = 1$ **then**
8: freq $\leftarrow$ freq $+ 1$
9: **if** freq$(R, S) <$ freq **then return** $<$
10: **if** freq$(R, S) >$ freq $+ |R'| - i$ **then return** $>$
11: **return** $=$

Algorithm 2: Generate one permutation $\sigma(S)$ and compare freq(R, S) with freq$(R, \sigma(S))$.

is already larger than freq(R, S), there is no need to compare the rest part of R' and $\sigma'(S)$, because we have:

$$\underline{\text{freq}(R, S) < \text{freq}(R'_{[i]}, \sigma'_{[i]}(S))} \quad \leq \text{freq}(R', \sigma'(S)).$$

Also, note that freq$(R'_{[i]}, \sigma'_{[i]}(S)) + n' - i$ is an upper bound of freq$(R', \sigma'(S))$ after comparing the first i elements. So one can stop if the upper bound is smaller than freq(R, S):

$$\underline{\text{freq}(R, S) > \text{freq}(R'_{[i]}, \sigma'_{[i]}(S)) + n' - i} \quad \geq \text{freq}(R', \sigma'(S)).$$

By checking these two conditions (*i.e.*, the underlined parts of the above formula), we could early stop the permutation generation and thus save the running time.

In Algorithm 2, we first get all the overlapped locations in R, denoted as R' (line 1). This is a pre-processing step and can be speeded up using R-tree. It only needs to be executed once for all the N permutation tests. It takes $O(n \log n)$ time to obtain R' by querying each location r_i in the R-tree of S. If r_i overlaps with S, it is inserted into R'. The random one-to-one mapping $\sigma' : [n'] \rightarrow [n]$ is generated sequentially in lines 4-6. The variable freq maintains the value of freq$(R'_{[i]}, \sigma'_{[i]}(S))$, and is updated in each iteration (lines 7-8). *Pruning II* is then implemented in lines 9-10. Even with the pruning rules, the complexity of Compare is still $O(n)$ in the worst case. However, as we will show in Section 5, the two pruning techniques are indeed very effective in practice.

It is worth noting that if R and S do not have any overlapped location, we will have $|R'| = 0$ and function Compare will immediately return "=". We could further speed up the computation by first filtering the *obviously* non-overlapped trajectories. We can check whether the minimum bounding boxes of two trajectories overlap. This checking step only takes $O(1)$ time and could save a lot of time if most trajectory pairs in database do not overlap.

4. THRESHOLD QUERY PROCESSING

In this section, we tackle the problem that, given a threshold Λ and a query trajectory Q, determine which trajectories in the database are significantly attracting or avoiding Q. For presentation simplicity, we only consider the significance value of attraction, $\text{sig}_{\text{attract}}$, and use "sig" to denote it for short. The algorithms and analysis for $\text{sig}_{\text{avoid}}$ are quite similar to the ones for $\text{sig}_{\text{attract}}$ and thus are omitted here.

Formally, for a trajectory database $\mathcal{D}$, a *threshold query* (Q, Λ) is to find all trajectories S's in $\mathcal{D}$ such that the significance value of avoidance or attraction between Q and S is no less than Λ. The answer to this query is:

$$\mathcal{A}_{Q,\Lambda} = \{S \in \mathcal{D} \mid \text{sig}(Q, S) \geq \Lambda\}.$$

A naive query-processing algorithm is to directly apply the ApproxCount algorithm to compute $\text{sig}(Q, S)$ with certain accuracy parameters ϵ and δ for each trajectory S in the database $\mathcal{D}$; and generate answers to threshold queries by simply scanning all trajectories. Of course, the answers obtained are approximate. Following are the guarantees we aim to provide for answers to threshold queries.

Definition 4.1. *(ϵ-approximate threshold query) An answer $\mathcal{A}_{Q,\Lambda}$ to a threshold query (Q, Λ) is ϵ-approximate iff: i) there are only a constant number (i.e., $O(1)$ in expectation) of items in $\mathcal{A}_{Q,\Lambda}$ such that* $\text{sig}(Q, S) < \Lambda/(1+\epsilon)$*; ii) there are only a constant number (i.e., $O(1)$ in expectation) of items NOT in $\mathcal{A}_{Q,\Lambda}$ such that* $\text{sig}(Q, S) > \Lambda/(1-\epsilon)$.

In other words, if an answer $\mathcal{A}_{Q,\Lambda}$ is ϵ-approximate, we have no guarantee on whether a trajectory S with $\Lambda/(1+\epsilon) \leq \text{sig}(Q, S) \leq \Lambda/(1-\epsilon)$ is included in/excluded from $\mathcal{A}_{Q,\Lambda}$. However, if $\text{sig}(Q, S) < \Lambda/(1+\epsilon)$, S is excluded from $\mathcal{A}_{Q,\Lambda}$ with high probability; and if $\text{sig}(Q, S) > \Lambda/(1-\epsilon)$, it is included in $\mathcal{A}_{Q,\Lambda}$ with high probability.

Theorem 3. *Applying algorithm* ApproxCount*, we can get an ϵ-approximate answer to a threshold query with a total of $(8m \ln 2m)/\epsilon^2$ random permutations, where $m = |\mathcal{D}|$ is the total number of trajectories in the database.*

Proof. For each trajectory $S \in \mathcal{D}$ in the database, we use algorithm ApproxCount to compute $\text{sig}(Q, S)$ with $N = \frac{8}{\epsilon^2} \ln 2m$ random permutations. Let $\text{sig}'(Q, S)$ be the estimated result returned by ApproxCount. From Theorem 2, if $\text{sig}(Q, S) < \Lambda/(1+\epsilon)$, then with probability at least $1 - \frac{1}{m}$, we have $\text{sig}'(Q, S) \leq (1+\epsilon)\text{sig}(Q, S) < \Lambda$. So only with probability $\frac{1}{m}$, S is put into $\mathcal{A}_{Q,\Lambda}$ (incorrectly). Hence, the total number of such trajectories S's is at most $m \cdot \frac{1}{m} = 1$ in expectation. Similarly, we can show that, if $\text{sig}(Q, S) > \Lambda/(1-\epsilon)$, we will miss such S in $\mathcal{A}_{Q,\Lambda}$ with probability at most $\frac{1}{m}$ (S should have been put into the answer set as $\text{sig}(Q, S) \geq \Lambda$). Again, we have at most 1 such trajectory in expectation. Applying algorithm ApproxCount for each $S \in \mathcal{D}$, we need a total of $m \cdot \frac{8}{\epsilon^2} \ln 2m$ random permutations. □

The cost of the query-processing algorithm is dominated by the number of random permutations needed. Next, we introduce the AdaptSample algorithm (Algorithm 3) which reduces the number of permutations needed while providing the same accuracy guarantee for the answers.

The basic idea is as follows. Let $\text{sig}_i(Q, S)$ be the significance value computed using ApproxCount(Q, S, i) with i random permutations. Clearly we have $\lim_{i \rightarrow \infty} \text{sig}_i(Q, S) = \text{sig}(Q, S)$. More specifically, according to Lemma 3, $N = \frac{4 \ln 2m}{\epsilon^2 \text{sig}(Q, S)}$ permutations are needed to get an ϵ-approximation of $\text{sig}(Q, S)$ with probability at least $1 - \frac{1}{m}$. However, since our goal is just to compare $\text{sig}(Q, S)$ with Λ, such an ϵ-approximation is often unnecessary. In particular, for any trajectory S with $\text{sig}_i(Q, S)$ either *far below or above* Λ, a much smaller number of permutations should suffice to determine if we should exclude or include S in the query result.

AdaptSample($\mathcal{D}, Q, \epsilon, \Lambda$)

1: Let $N = \frac{4 \ln 2m}{\epsilon^2 \Lambda}$;
2: **for** $i = 1$ to N **do**
3: **for** each trajectory S in $\mathcal{D}$ **do**
4: $v \leftarrow$ Compare(Q, S)
5: (Compare freq(Q, S) and freq($Q, \sigma_i(S)$))
6: Calculate $\text{sig}_i(Q, S)$ from v and $\text{sig}_{i-1}(Q, S)$
7: **if** $\text{sig}_i(Q, S) < l(i, \epsilon, \Lambda)$ **then**
8: $\mathcal{D} \leftarrow \mathcal{D} - \{S\}$;
9: **if** $\text{sig}_i(Q, S) > u(i, \epsilon, \Lambda)$ **then**
10: $\mathcal{D} \leftarrow \mathcal{D} - \{S\}$ and $\mathcal{A} \leftarrow \mathcal{A} + \{S\}$;
11: **for** each trajectory S in $\mathcal{D}$ **do**
12: **if** $\text{sig}_N(Q, S) \geq \Lambda$ **then** $\mathcal{A} \leftarrow \mathcal{A} + \{S\}$;
13: Output $\mathcal{A}$.

Algorithm 3: Threshold query processing with adaptive sampling.

Based on this observation, we design the bounds $l(i, \epsilon, \Lambda)$ and $u(i, \epsilon, \Lambda)$ in AdaptSample to make such decisions "on the fly", hence greatly reduce the computation time.

Next, we show that AdaptSample produces ϵ-approximate answers to threshold queries. Although the algorithm has the same $O\left(\frac{m}{\epsilon^2} \ln m\right)$ complexity as ApproxCount in the worst case, we show that under certain conditions, the number of random samples is much smaller than the worst case on average, with an improvement of a factor of $O(1/\epsilon)$.

THEOREM 4. *In* AdaptSample *algorithm, let* $m = |\mathcal{D}|$,

$$l(i, \epsilon, \Lambda) = \left(1 - \sqrt{\frac{4}{i\Lambda} \ln \frac{8m \ln 2m}{\epsilon^2 \Lambda}}\right) \Lambda, \text{ and}$$

$$u(i, \epsilon, \Lambda) = \left(1 + \sqrt{\frac{4}{i\Lambda} \ln \frac{8m \ln 2m}{\epsilon^2 \Lambda}}\right) \Lambda.$$

Then the output $\mathcal{A}$ *is an* ϵ*-approximate answer to the threshold query* (Q, Λ). *Suppose* $\text{sig}(Q, S)$ *is uniformly distributed in* $[0, 1]$, *the total number of random permutations needed is* $O\left(\frac{m}{\epsilon} \ln \frac{m}{\epsilon^2}\right)$ *in expectation.*

PROOF. To prove this theorem, we first show that the output $\mathcal{A}$ is indeed ϵ-approximate, and then prove the efficiency, *i.e.*, the number of permutations needed.

For a trajectory S, if $\text{sig}(Q, S) \geq \Lambda$, using Lemma 3, we can prove that we may have $\text{sig}_i(Q, S) < l(i, \epsilon, \Lambda)$ in an iteration i only with probability at most

$$\Pr[\text{fail}] = \frac{1}{m} \cdot \frac{\epsilon^2 \Lambda}{4 \ln 2m} = \frac{1}{mN}.$$

So the probability of *not making a mistake* for this trajectory S (having $\text{sig}_i(Q, S) \geq l(i, \epsilon, \Lambda)$ for all iterations) is at least

$$(1 - \Pr[\text{fail}])^N \approx 1 - 1/m.$$

Similarly, for a trajectory S with $\text{sig}(Q, S) < \Lambda$, we can prove that the probability of not making a mistake (having $\text{sig}_i(Q, S) \leq u(i, \epsilon, \Lambda)$ for all iterations) is at least $1 - 1/m$.

There are a total of m trajectories. Therefore, in expectation, we make at most $m \cdot \frac{1}{m} = 1$ mistakes (missing S when $\text{sig}(Q, S) \geq \Lambda$ or putting S into $\mathcal{A}$ when $\text{sig}(Q, S) < \Lambda$). For all the other trajectories left in $\mathcal{D}$ after lines 2-9, from Lemma 3, lines 10-11 ensure that the final output $\mathcal{A}$ is an ϵ-approximate answer to the query (Q, Λ).

Assuming that $\text{sig}(Q, S)$ is uniformly distributed in [0,1] for all trajectories S in $\mathcal{D}$, let's analyze how many random permutations are needed in total. Let's consider the case when $\Lambda = 1$, as it is not hard to show that, in this case, we need the max number of permutations (for $\Lambda \in [0.5, 1]$). The average (expected) number of permutations needed for each trajectory $S \in \mathcal{D}$ in the database (note that $\text{sig}(Q, S)$ is uniformly distributed in [0,1]) can be calculated as:

$$\begin{aligned} & 4 \ln \frac{8m \ln 2m}{\epsilon^2} + \sum_{i=4 \ln \frac{8m \ln 2m}{\epsilon^2}}^{\frac{4 \ln 2m}{\epsilon^2}} \sqrt{\frac{4}{i} \ln \frac{8m \ln 2m}{\epsilon^2}} \\ \approx\ & 4 \ln \frac{8m \ln 2m}{\epsilon^2} + \int_{4 \ln \frac{8m \ln 2m}{\epsilon^2}}^{\frac{4 \ln 2m}{\epsilon^2}} \sqrt{\frac{4}{x} \ln \frac{8m \ln 2m}{\epsilon^2}} \mathrm{d}x \\ =\ & O\left(\frac{m}{\epsilon} \ln \frac{m}{\epsilon^2}\right). \end{aligned}$$

Therefore, the total number of random permutations needed is at most $O\left(\frac{m}{\epsilon} \ln \frac{m}{\epsilon^2}\right)$ in expectation. □

In a large-size dataset, R may be clearly irrelevant with most moving objects. We could first index the bounding boxes of all objects in R-tree and only consider those objects with bounding boxes overlapped with that of R as candidates. This preprocessing step can filter many irrelevant candidates before query processing.

5. EXPERIMENT

In this section, we present a comprehensive performance study of the proposed methods on both real and synthetic datasets. All the experiments are conducted on a 2.9 GHz Intel Core i7 system with 16GB memory.

Note that, to simplify the presentation, we will only show the values of $\text{sig}_{\text{attract}}(R, S)$ in all the experiments, since we have $\text{sig}_{\text{attract}}(R, S) + \text{sig}_{\text{avoid}}(R, S) = 1$. Therefore, if $\text{sig}_{\text{attract}}(R, S)$ is close to 1, then R and S have significant *attraction* relationship; If $\text{sig}_{\text{attract}}(R, S)$ is close to 0 (*i.e.*, $\text{sig}_{\text{avoid}}(R, S)$ close to 1), then R and S have significant *avoidance* relationship. Parameter N is determined by parameters ϵ and Λ. For simplicity of experiment, we study the performance by directly tuning N.

5.1 Experiments on Synthetic Data

We first conduct experiments on synthetic dataset to demonstrate the effectiveness and efficiency of our method. Note that, for this experiment, it is not necessary to generate the trajectories R and S, as long as we have the pairwise distance matrix, denoted as M, of the two trajectories. More specifically, $M(i, j)$ takes value 1 if $\text{distance}(r_i, s_j) < d$, and 0 otherwise. For a pair of trajectories of the same length n, we generate the pairwise distance matrix as follows. (1) Generate a vector of length n by setting $\alpha \cdot n$ $(0 < \alpha < 1)$ uniformly chosen entries of the vector to 1 and the rest to 0. Use this vector as the diagonal of M. (2) Draw two distinct random integers i, j uniformly from $[1 : n]$ and set $M(i, j)$ to 1. Repeat this till the number of 1's in M reaches $\beta \cdot n^2$ (including its diagonal), where $0 < \beta < 1$.

In this way, we ensure that the resulting synthetic trajectory pair has $\text{E}[\text{freq}(R, S)] = \beta \cdot n$, and $\text{freq}(R, S) = \alpha \cdot n$.

5.1.1 Computing Significance Value

In this section, we study how fast the significance value converges to its true value w.r.t. the number of permutations

N. We generate trajectory pairs with $\beta = 0.2$, $n = 1000$, and vary α to get different significance values. Since computing the exact significance value given M is #P-hard (Theorem 1), we use the significance value estimated using a sufficiently large number of permutations ($N = 1000$) as the ground truth, and denote it as $\text{sig}^*_{\text{attract}}$.

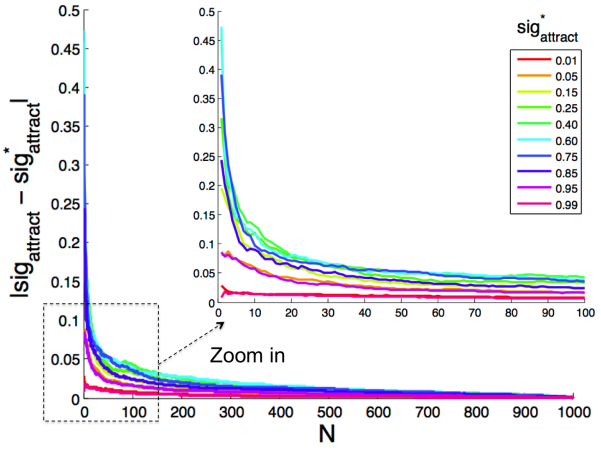


Figure 2: Difference between significance value and ground truth using N permutations

In Figure 2, we show the difference between the ground truth $\text{sig}^*_{\text{attract}}$ and the significance value calculated after N permutations. Each line corresponds to one significance value and the result is the average over the 100 trials.

We can see from Figure 2 that the estimated significance value converges fast to its true value. For example, when $N = 100$, the differences are below 0.05 for all significant values. Further, in the zoomed-in figure, we can see that the estimated significance value converges faster as $\text{sig}^*_{\text{attract}}$ approaches 0 or 1, and slower when $\text{sig}^*_{\text{attract}}$ is close to 0.5. This observation is consistent with Lemma 3, which suggests that, to achieve the same ϵ-approximation to ρ (assuming $\rho \geq 0.5$), the lower ρ is, the more permutations are needed.

5.1.2 *Efficiency of Computing Significance Value*

Now we study the effectiveness of our pruning rules developed in Section 3.3 for computing the significance values. In this experiment, we compare the original ApproxCount algorithm proposed in Algorithm 1 with the one which combines ApproxCount with the pruning rules, denoted as ApproxCount+.

Here, we define the ratio of overlapped locations as $\sigma = \frac{|R'|}{|R|}$ ($\beta < \sigma < 1$), and denote the number of trajectory pairs as n_p. By default, we use the following parameters: $\alpha = 0.05, \beta = 0.05, \sigma = 0.07, n = 1000, n_p = 2000, N = 1000$. In this set of experiments, we examine the total running time needed for all n_p trajectory pairs.

Running time w.r.t. trajectory length n. Figure 3(a) shows that the running time grows roughly linearly in n for both methods. However, pruning rules reduce the computation time. And it becomes more obvious as the trajectory length grows. For example, when $n = 1000$, ApproxCount+ takes less than 15 seconds, whereas ApproxCount takes 160 seconds.

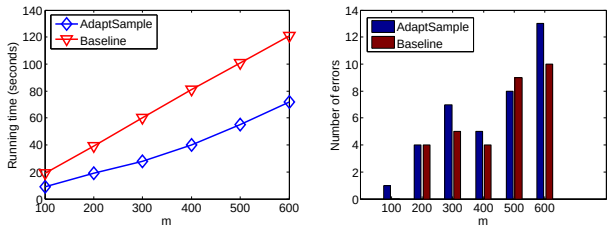
Running time w.r.t. number of pairs n_p. Figure 3(b) shows that as the size of moving objects in a dataset becomes larger, our pruning rules are saving more computation time. For example, when the number of pairs $n_p = 10,000$, ApproxCount+ takes about 60 seconds, while ApproxCount takes about 800 seconds.

Running time w.r.t. the ratio of overlapped location σ. In this experiment, we set the number of pairs $n_p = 10,000$. Figure 3(c) shows that, as $\sigma = \frac{|R'|}{|R|}$ decreases from 0.1 to 0.06, the running time of ApproxCount+ decreases from about 90 seconds to 55 seconds. This is because *Pruning I* is more effective when σ is smaller. Meanwhile, the running time of ApproxCount is not affected by σ (about 800 seconds in this case), and is omitted in Figure 3(c).

5.1.3 *Efficiency of Threshold Query Processing*

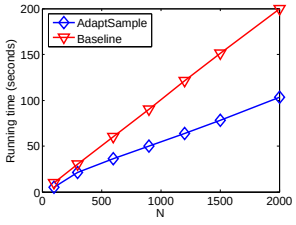
Given a query moving object, a threshold query is to retrieve all the objects with significance values above certain threshold Λ. In this section, we compare AdaptSample method proposed in Section 4 with Baseline. Baseline computes the significance value for each object and then return the objects satisfying threshold Λ. We show that AdaptSample can reduce query time without sacrificing accuracy.

We use the following parameters by default: $\beta = 0.2$, $\sigma = 1$, $n = 1000$, $m = 500$, $N = 1000$, and the threshold $\Lambda = 0.9$. Further, we vary α during the data generation process so that the significance values are uniformly distributed in $[0, 1]$. We use the significance value computed with $N = 10,000$ permutations as the ground truth.

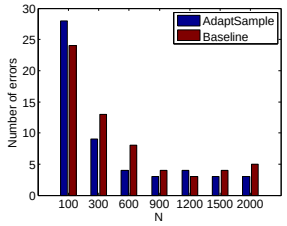


(a) Running time w.r.t. number of objects m

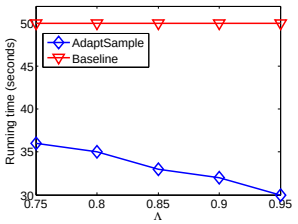
(b) Number of errors w.r.t. number of objects m



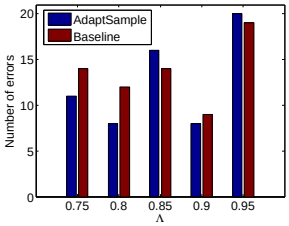
(c) Running time w.r.t. number of permutations N



(d) Number of errors w.r.t. number of permutations N

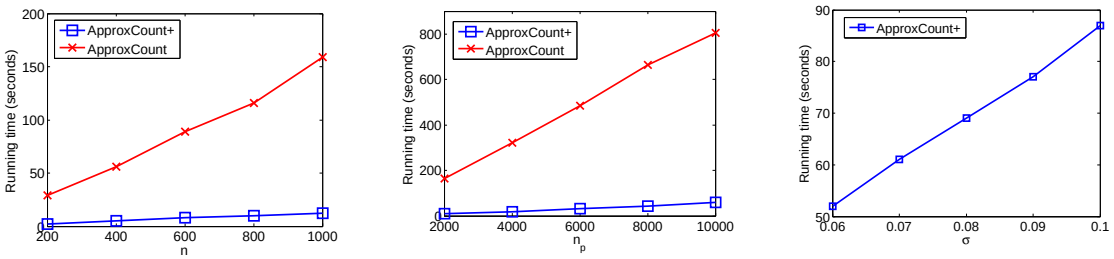


(e) Running time w.r.t. threshold Λ



(f) Number of errors w.r.t. query threshold Λ

Figure 4: Threshold query evaluation



(a) Running time w.r.t. trajectory length n

(b) Running time w.r.t. number of object pairs n_p

(c) Running time w.r.t. overlapped locations ratio σ

Figure 3: Efficiency study on synthetic data

Performance w.r.t. number of objects m. In Figure 4(a), we show the running time as a function of m. We can see the difference between AdaptSample and Baseline is more obvious as m increases. For example, when $m = 600$, AdaptSample takes about 70 seconds, while Baseline takes 120 seconds. In Figure 4(b), we further show the number of errors made by both methods. One can see that, as m increases, the number of errors remains roughly constant. This is consistent with Theorem 3 and 4.

Performance w.r.t. number of permutations N. As one can see in Figure 4(c), AdaptSample is again significantly faster than Baseline, and the difference becomes more obvious when N is large. For example, when $N = 2000$, Baseline takes twice of the time of AdaptSample. In Figure 4(d), one can see that the number of errors decreases as the number of permutations increases. Again, comparing two methods, the difference in accuracy is small.

Performance w.r.t. query threshold Λ. In this experiment, we set $N = 500$ and vary Λ. The running time of Baseline is not affected by Λ, as shown in Figure 4(e). Meanwhile, as Λ approaches to 1 (assuming $\Lambda > 0.5$), AdaptSample is more effective in filtering out unqualified candidates. In Figure 4(f) we again observe that the difference in number of errors between these two methods is insignificant.

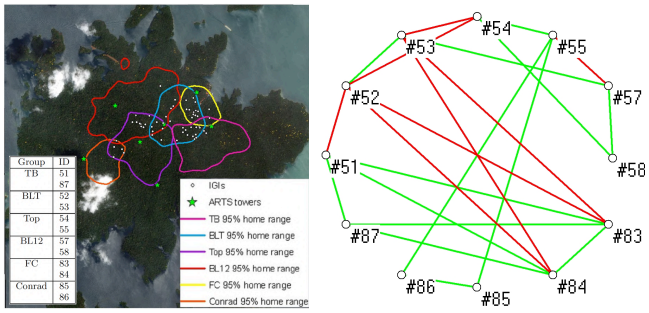
In summary, we have shown that AdaptSample can speed up the process of threshold query. And the speed-up becomes more significant when the data size is bigger, number of permutations is larger, and the threshold is higher. At the same time, AdaptSample achieves the same accuracy statistically as Baseline.

5.2 Experiments on Animal Movement Data

In this section, we use a real movement dataset. The dataset contains trajectories of 12 capuchin monkeys (cebus capucinus) with tracking time from 11/10/2004 to 04/18/2005. The average sampling rate for this dataset is about 15 minutes, and the number of locations for one monkey varies from 1,500 to 11,000. The monkeys belong to six groups. Figure 5(a) plots home ranges of the six monkey groups. Please refer to [8] for more detailed description of this dataset. By default, we set distance threshold $\epsilon = 100(meters)$, and number of permutations $N = 1000$ for our experiments.

5.2.1 Mining Significant Relationships

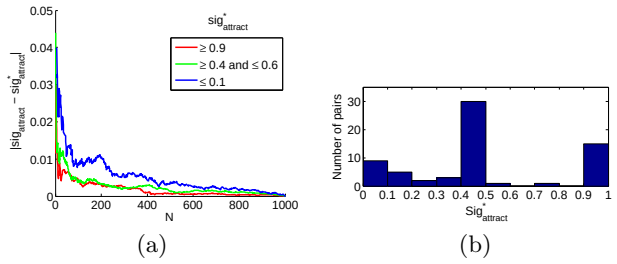
Figure 5(b) shows the pairwise significant relationships for the monkey dataset. A green line represents significant attraction relationship ($\text{sig}_{\text{attract}} > 0.95$). A red line represents significant avoidance relationship ($\text{sig}_{\text{attract}} < 0.05$, *i.e.*, $\text{sig}_{\text{avoid}} > 0.95$). There are some interesting findings from this network graph. First, the monkeys in the same group all have significant attraction relationships, which is expected. Then, let us look at FC group (#83, #84). We can see that FC group has a significant avoidance relationship with BLT group (#52, #53), but a significant attraction relationship with TB group (#51, #87). Looking at the home range plot in Figure 5(a), we can see that FC and BLT have much larger shared home range compared with that shared by FC and TB group. But the meeting frequency between FC and BLT is similar to that between FC and TB, *e.g.*, $\text{freq}(\#83, \#52) = 11$ and $\text{freq}(\#83, \#51) = 8$. According to a report from animal scientist [8], there have been 13 fights reported between FC and BLT group, but only 3 fights reported between FC and TB group. This well explains the avoidance relationship between FC and BTL detected by our method. An explanation of attraction relationship between FC and TB group could be the mutual attraction to some time-specific resource in their overlapped territories.



(a) (b)

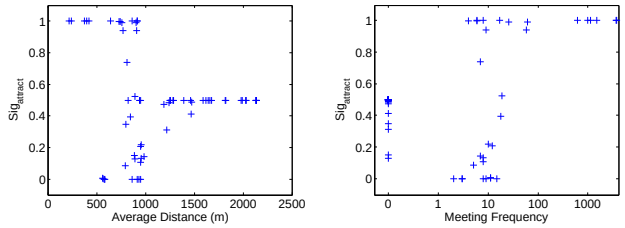
Figure 5: (a) Home ranges of the six study groups. (b) Significant relationships for monkey data (green line: attraction, red line: avoidance).

Now we study how fast the significance value converges to its true value on the monkey dataset. We again take the significance value computed using 1000 permutations as the ground truth, denoted as $\text{sig}^*_{\text{attract}}$. In Figure 6(a), we show the difference between $\text{sig}^*_{\text{attract}}$ and $\text{sig}_{\text{attract}}$ calculated using N permutations for various significance values. As we can see, the estimated significance values converge very fast and the differences are all less than 0.01 after 200 permutations regardless of the true values. However, note that the green line ($0.4 \leq \text{sig}^*_{\text{attract}} \leq 0.6$) converges faster



(a) (b)

Figure 6: (a) Convergence rate of significance values. (b) Histogram of significance values.



(a) (b)

Figure 7: Comparison with traditional measures

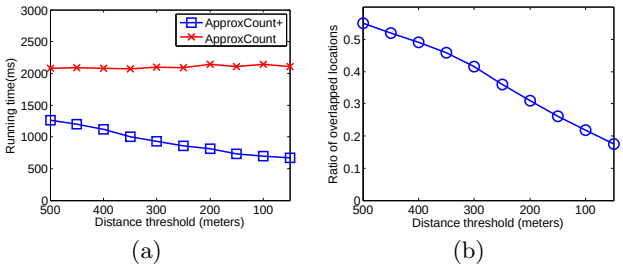
than the blue line ($\text{sig}^*_{\text{attract}} \leq 0.1$). This is in contrast to the synthetic data case and Lemma 3, which suggest that when $\text{sig}^*_{\text{attract}}$ approaches to 0 or 1, less number of permutations are needed to achieve the ϵ-approximation. To understand this, we show the histogram of number of object pairs w.r.t. to different significant values in Figure 6(b). We find that among the pairs whose $\text{sig}^*_{\text{attract}}$ values are in the range of $[0.4, 0.6]$, there are 15 pairs which do not have any overlapped location in their trajectories (*i.e.*, $|R'| = 0$). For those pairs, the estimated significance values converge to exactly 0.5 in one iteration, which greatly affects the average convergence rate for pairs in this group.

5.2.2 *Comparing Significance Value with Traditional Measures*

As we discussed before, previous work [30, 29, 6, 5] often use (1) the Euclidean distance or its variants and (2) the meeting frequency to measure the relationship between moving objects. Their assumption is that smaller Euclidean distance or larger meeting frequency indicates stronger attraction relationship. We now demonstrate why this assumption is not necessarily true.

In Figure 7(a), we plot the significance value and the average Euclidean distance for each pair of monkeys. We can see that, if the average distance between two monkeys is larger than $1km$, they usually have independent relationship ($\text{sig}_{\text{attract}}$ close to 0.5). This often occurs when two monkeys have their own territories and their trajectories do not overlap. Meanwhile, when the average distance is less than 0.5km, the pair is very likely to have attraction relationship. However, when the average distance is in the middle range of 0.5km-1km, the relationship can either be attraction, independent or avoidance. As a result, by using the average distance, one cannot correctly identify the relationship in these cases.

Similarly, in Figure 7(b), we show that meeting frequency is also not a good measure for attraction and avoidance relationships. As one can see, if the meeting frequency is high (> 20), two monkeys are very likely to have attraction relationship. However, when the frequency is in the middle range from 1 to 20, the relationships can either be attraction, independent or avoidance. Further, if two monkeys have zero meeting frequency, they could either be independent or having an avoidance relationship.



(a) (b)

Figure 8: Efficiency study on synthetic data

5.2.3 *Efficiency of Computing Significance Value*

Next, we verify the effectiveness of the pruning rules we developed in Section 3.3 on the monkey dataset. Figure 8(a) plots the running time of our methods ApproxCount and ApproxCount+. We can see that ApproxCount+ is much faster than ApproxCount. For example, when the distance threshold $d = 100(meters)$, ApproxCount spends about 2200 milliseconds to compute the significance value. On the other hand, ApproxCount+ only takes 700 milliseconds to finish. To understand why this is the case, in Figure 8(b) we plot the average ratio of overlapped locations between monkey pairs as a function of the distance threshold d. As one can see, the ratio is typically very small. For example, when $d = 100$, that number of overlapped locations (*i.e.*, $|R'|$) is only 20% of the total number of points in R (*i.e.*, n). In addition, as the distance threshold decreases, $|R'|$ becomes smaller, and therefore *Pruning I* is more effective.

5.2.4 *Efficiency of Threshold Query*

Lastly, we demonstrate that AdaptSample method can speed up the threshold query processing without sacrificing the accuracy on the monkey dataset. In this experiment, we use one monkey as the query object and retrieve all the other monkeys having significant values $\text{sig}_{\text{attract}}$ above threshold Λ. We repeat the process for every monkey in the dataset and record the total time in Figure 9 w.r.t. the threshold Λ.

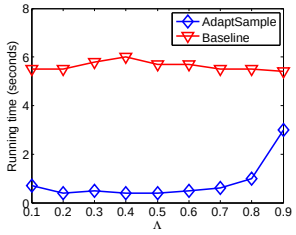


Figure 9: Threshold query on monkey data set

We can see that AdaptSample is significantly faster than Baseline. For example, with $\Lambda < 0.7$, AdaptSample takes less than 1 second to process the threshold query, while Baseline

takes about 5.5 seconds. Note that, however, as Λ increases from 0.5 to 0.9, the computation time for AdaptSample also increases, which contradicts the result on synthetic dataset. The reason is that for the monkey dataset, the true significance values are not uniformly distributed in $[0, 1]$, as shown in Figure 6(b). In particular, there is a large number of pairs with significance values in $[0.9, 1]$, compared to the number of pairs with significance values in $[0.5, 0.9]$. So AdaptSample takes more iterations to process these pairs for bigger Λ.

In terms of accuracy, both methods retrieve identical objects for all threshold queries, except when $\Lambda = 0.1$ and 0.3, AdaptSample has 2 and 1 false positives, respectively.

5.3 Experiment on Human Movement Data

In this section, we use Reality Mining dataset (`http://reality.media.mit.edu`) to evaluate the effectiveness and efficiency of our method on human movements. The dataset contains movement data of 95 persons. A person's movement data includes a sequence of timestamps and the corresponding cell tower IDs.

We use friendship survey, affiliation and group information as the ground truth in this experiment. Pairs having certain relationship (*i.e.*, being friends, having the same affiliation, or being in the same group) are denoted as 1 (a positive pair), and 0 otherwise (a negative pair). Out of the 4465 pairs in total, there are 68 friend pairs, 74 pairs who are in the same group, and 670 pairs that have same affiliation.

	Group	Affiliation	Friend
Significance Value	0.6221	0.6538	0.7001
Meeting Frequency	0.3104	0.3371	0.5053
Dynamic Time Warping	0.1967	0.2627	0.5576

Table 1: Cosine similarity score with ground truth

We use cosine similarity to compute the correlation between ground truth and different measures including the significance value, meeting frequency, and dynamic time warping distance. The number of permutations N is set to 1000 for our method. Since the number of negative pairs is much more than that of positive ones, we sample the negative pairs to make the number balanced and report the average cosine similarity over 1000 sampling trials in Table 1. As one can see, the cosine similarity scores between significance value and ground truth are always above 0.62, and are consistently higher than the scores for meeting frequency or dynamic time warping.

The time complexities for calculating the meeting frequency, dynamic time warping and significance value for one pair are $O(n)$, $O(n^2)$, and $O(N \cdot n)$, respectively. To compute the values for all pairs, it takes 1 second for meeting frequency and 153 seconds for dynamic time warping. The running time for computing the significance values is 118 seconds with pruning, and 233 seconds without pruning. It worths noting that persons tracked in this dataset are all faculties, staffs, and students at MIT. So they share significant amount of overlapped locations in their movements. But our pruning techniques are still effective in this case.

6. RELATED WORK

In data mining literature, the problem of mining moving object clusters, which aims to detect attraction relationships among moving objects, has been extensively studied. Generally speaking, a moving object cluster is "a set of objets that move together". In particular, several measures have been proposed to measure the similarity between trajectories, including the Euclidean Distance, Dynamic Time Warping (DTW) [30], Longest Common Subsequences (LCSS) [29], Edit Distance on Real sequence (EDR) [6], and Edit distance with Real Penalty (ERP) [5]. The intuition behind these measures is that, the more similar two trajectories are, the closer their relationship is. An alternative approach is to count the number of timestamps at which a set of objects are located together. If the objects are frequently co-located, they form a cluster. Representative methods include moving cluster [17], flock [20, 12, 11, 1], convoy [16], swarm [21], and gathering pattern [31]. However, all the methods above do *not* consider *the background model* of the moving objects. Instead, they measure the degree of relationship based on the similarity of trajectories or co-location frequencies. As we have previously shown, such measures are not necessarily valid for mining significant relationships.

Methods have also been proposed to characterize the temporal patterns in order to detect semantic relationships. Miklas *et al.* [24] find that "friends meet more regularly and for longer duration whereas the strangers meet sporadically" and Eagle *et al.* [10] shows that friend demonstrates distinctive temporal and spatial patterns. Then, temporal and spatial features are extracted to build the semantic relationship classifier in a supervised framework [7, 23]. But in this paper we only focus on unsupervised methods.

In this paper, we use *permutation test* to calculate the significance values. The permutation test, also known as randomization test or shuffle test, is a standard statistical approach, and has been applied to measure the significant correlation on social network [2, 3, 19], graph [13], and time series [4, 18]. The significance values of avoidance and attraction relationships we define in this paper have rigorous statistical semantics. However, computing the exact significance values is proved to be #P-hard. So we employ Monte Carlo sampling methods to compute the significance values and process the threshold queries approximately. Similar ideas can be found in other areas of uncertain databases, such as the works on query evaluation in uncertain databases [27, 14, 15]. In particular, [27] evaluate the top-k queries which output tuples satisfying a query and with the top-k highest probabilities. A Monte Carlo sampling algorithm is used to compute the probabilities approximately and a lower-higher bound shrinking scheme is developed to process top-k queries efficiently. Although sharing some similarities, techniques in our work and the works on query evaluation in uncertain database cannot be applied interchangeably. The major reason is that *uncertainty in our work arises from the statistic model for defining significance*, while uncertainty in those works arises from data itself, not to mention the differences on the formulations of "probabilities" and "queries" for addressing different application interests.

7. DISCUSSION

In this section, we discuss two interesting extensions of our work. First, our current permutation method may generate "impossible" trajectories. For example, two consecutive points in a randomized trajectory might be too far away for the object to travel within a time unit. It is therefore desirable for our method to preserve *certain spatiotemporal*

properties in the trajectories while conducting the permutation test. Such property could be the distance constraint on any two consecutive points, or some landscape constraints induced by a river, a mountain, or the road network. The property can also come from certain known trajectory patterns of the moving objects, such as daily periodic patterns or seasonal migration patterns [22].

Second, in our current framework, we simply count the meeting frequency without considering the semantics in the meeting events. For example, meeting for 10 consecutive hours and meeting for 1 hour each day for 10 days obviously carry different semantic meanings and therefore could indicate different types of relationships. Also, some places are visited more frequently in general, such as a public park in down town, whereas locations like a private property are visited much less frequently. Meeting events at different locations also carry different semantics. In the future, we plan to use the weighted meeting frequency to incorporate the semantics into the relationship detection framework.

8. CONCLUSION

In this paper, we propose a unified framework to detect significant attraction and avoidance relationships in movement data. The idea of our method is that, in order to mine significant relationships, one needs to look into the background model of the movement data. Based on this idea, we propose to use permutation test to evaluate the significance value of the relationships. Two pruning techniques are proposed to speed up the permutation test. In addition, we discuss how to answer threshold queries for movement database and retrieve all the objects having significant relationships with the querying object. An early-stop strategy is employed for efficient query processing.

9. REFERENCES

[1] G. Al-Naymat, S. Chawla, and J. Gudmundsson. Dimensionality reduction for long duration and complex spatio-temporal queries. In *SAC'07*.

[2] A. Anagnostopoulos, R. Kumar, and M. Mahdian. Influence and correlation in social networks. In *KDD'08*.

[3] S. Aral, L. Muchnik, and A. Sundararajan. Distinguishing influence-based contagion from homophily-driven diffusion in dynamic networks. *PNAS'09*.

[4] N. Castro and P. J. Azevedo. Time series motifs statistical significance. In *ICDM'11*.

[5] L. Chen and R. T. Ng. On the marriage of lp-norms and edit distance. In *VLDB'04*.

[6] L. Chen, M. T. Özsu, and V. Oria. Robust and fast similarity search for moving object trajectories. In *SIGMOD'05*.

[7] J. Cranshaw, E. Toch, J. I. Hong, A. Kittur, and N. Sadeh. Bridging the gap between physical location and online social networks. In *Ubicomp'10*.

[8] M. Crofoot, I. Gilby, M. Wikelski, and R. Kays. Interaction location outweighs the competitive advantage of numerical superiority in cebus capucinus intergroup contests. *PNAS'08*.

[9] C. P. Doncaster. Non-parametric estimates of interaction from radio-tracking data. *Journal of Theoretical Biology'90*.

[10] N. Eagle, A. Pentland, and D. Lazer. Inferring friendship network structure by using mobile phone data. In *PNAS'09*.

[11] J. Gudmundsson and M. van Kreveld. Computing longest duration flocks in spatio-temporal data. In *GIS'06*.

[12] J. Gudmundsson, M. J. van Kreveld, and B. Speckmann. Efficient detection of motion patterns in spatio-temporal data sets. In *GIS'04*.

[13] S. Hanhijärvi, G. C. Garriga, and K. Puolamäki. Randomization techniques for graphs. In *SDM'09*.

[14] M. Hua, J. Pei, W. Zhang, and X. Lin. Ranking queries on uncertain data: a probabilistic threshold approach. In *SIGMOD'08*.

[15] R. Jampani, F. Xu, M. Wu, L. L. Perez, C. M. Jermaine, and P. J. Haas. Mcdb: a monte carlo approach to managing uncertain data. In *SIGMOD'08*.

[16] H. Jeung, M. L. Yiu, X. Zhou, C. S. Jensen, and H. T. Shen. Discovery of convoys in trajectory databases. In *VLDB'08*.

[17] P. Kalnis, N. Mamoulis, and S. Bakiras. On discovering moving clusters in spatio-temporal data. In *SSTD'05*.

[18] J. Kawale, S. Chatterjee, D. Ormsby, K. Steinhaeuser, S. Liess, and V. Kumar. Testing the significance of spatio-temporal teleconnection patterns. In *KDD'12*.

[19] T. La Fond and J. Neville. Randomization tests for distinguishing social influence and homophily effects. In *WWW'10*.

[20] P. Laube and S. Imfeld. Analyzing relative motion within groups of trackable moving point objects. In *GIS'02*.

[21] Z. Li, B. Ding, J. Han, and R. Kays. Swarm: Mining relaxed temporal moving object clusters. In *VLDB'10*.

[22] Z. Li, B. Ding, J. Han, R. Kays, and P. Nye. Mining periodic behaviors for moving objects. In *KDD'10*.

[23] Z. Li, C. X. Lin, B. Ding, and J. Han. Mining significant time intervals for relationship detection. In *SSTD'11*.

[24] A. G. Miklas, K. K. Gollu, K. K. W. Chan, S. Saroiu, P. K. Gummadi, and E. de Lara. Exploiting social interactions in mobile systems. In *Ubicomp'07*.

[25] R. Motwani and P. Raghavan. *Randomized Algorithms*. Cambridge University Press, 1995.

[26] E. W. Noreen. Computer intensive methods for testing hypotheses. *Journal of Theoretical Biology*, 1989.

[27] C. Re, N. N. Dalvi, and D. Suciu. Efficient top-k query evaluation on probabilistic data. In *ICDE'07*.

[28] L. G. Valiant. The complexity of computing the permanent. *Theor. Comput. Sci.*, 8, 1979.

[29] M. Vlachos, D. Gunopulos, and G. Kollios. Discovering similar multidimensional trajectories. In *ICDE'02*.

[30] B.-K. Yi, H. V. Jagadish, and C. Faloutsos. Efficient retrieval of similar time sequences under time warping. In *ICDE'98*.

[31] K. Zheng, Y. Zheng, N. J. Yuan, and S. Shang. On discovery of gathering patterns from trajectories. In *ICDE'13*.