Towards Controlling the Transmission of Diseases: Continuous Exposure Discovery over Massive-Scale Moving Objects

Ke Li 1, Lisi Chen1, Shuo Shang1,2∗, Haiyan Wang 1, Yang Liu1, Panos Kalnis3 and Bin Yao 4
1University of Electronic Science and Technology of China, Chengdu, 611731, China
2Sichuan Artificial Intelligence Research Institute, Yibin, 644000, China
3King Abdullah University of Science and Technology, Saudi Arabia
4Shanghai Jiao Tong University, Shanghai, 200240, China

like_like@std.uestc.edu.cn, {chenlisi.cs, jedi.shang}@gmail.com, imwhyyyy@gmail.com,
jane_liuyang@126.com, panos.kalnis@kaust.edu.sa, yaobin@cs.sjtu.edu.cn

Abstract
Infectious diseases have been recognized as major public health concerns for decades. Close contact discovery is playing an indispensable role in preventing epidemic transmission. In this light, we study the continuous exposure search problem: Given a collection of moving objects and a collection of moving queries, we continuously discover all objects that have been directly and indirectly exposed to at least one query over a period of time. Our problem targets a variety of applications, including but not limited to disease control, epidemic pre-warning, information spreading, and co-movement mining. To answer this problem, we develop an exact group processing algorithm with optimization strategies. Further, we propose an approximate algorithm that substantially improves the efficiency without false dismissal. Extensive experiments offer insight into effectiveness and efficiency of our proposed algorithms.

1 Introduction
With the continued growth of location-tracking devices (e.g., vehicle navigation systems and smartphones) and GPS-enabled services (e.g., Google Maps), the volume of trajectory data is skyrocketing. Among diverse trajectory query enabled services (e.g., Google Maps), the volume of trajectory search problem: Given a collection of moving objects and a collection of moving queries, we continuously discover all objects that have been directly and indirectly exposed to at least one query over a period of time. Our problem targets a variety of applications, including but not limited to disease control, epidemic pre-warning, information spreading, and co-movement mining. To answer this problem, we develop an exact group processing algorithm with optimization strategies. Further, we propose an approximate algorithm that substantially improves the efficiency without false dismissal. Extensive experiments offer insight into effectiveness and efficiency of our proposed algorithms.

Figure 1: Example of cases of exposure

close contacts can help us limit human transmissions and analyze individual infectious risk.

Existing studies focus on the contact tracing for a single query object (or query trajectory) based on the spatial proximity between query object and data objects (e.g., [Faloutsos et al., 1994; Assent et al., 2009]), some studies apply similarity computation among trajectory segments (e.g., [Rong et al., 2017; Xie et al., 2017]). However, during the disease outbreaks, massive-scale individuals may be infected in a short period of time. It is of great importance to enable continuous contact search among a huge number of moving objects based on their trajectories. An effective contact search should not only be capable of handling massive-scale moving objects simultaneously, but also guarantee the real-time detection regarding each object. To the best of our knowledge, existing studies formulate the problem as a standalone query processing problem, trajectory search problem [Alarabi et al., 2021; Chao et al., 2021], or a search problem with multiple queries [He et al., 2020; Ojagh et al., 2021]. These studies either target to process a small number of queries, or consider a static collection of trajectories as input, making them ineffective to process massive-scale moving objects over a large number of queries in practice. Meanwhile, they are incompetent to maintain real-time detection results for each object.

In this light, we define and study the Continuous Exposure Search (CES) problem: Given a collection of moving objects \( O \), and a set of queries \( Q \in O \) denoting existing cases of exposure, we target to continuously detect new cases of ex-
pose by updating $Q$ in a real-time fashion. In our settings, new cases of exposure are the objects exposed to query objects within a distance threshold $\epsilon$, and they have been traveling together over a period of time $k$. Let us consider Figure 1 as a toy example. Let $o_1, o_2, o_3$ be three moving objects and initial query set be $Q = \{o_1\}$. We set the distance threshold $\epsilon$ to be 2 meters, and set contact duration threshold $k$ to be 3 minutes. We use red, blue, and purple points to denote locations of exposed (query) objects, close contacts, and up-to-date cases of exposures, respectively. Object $o_2$ closely contacts query object $o_1$ during a consecutive period of time $[t_1, t_3]$, thus we update the query set $Q$ by adding $o_2$ (i.e., $Q = \{o_1, o_2\}$) and return updated $Q$ at $t_3$ as real-time result. Similarly, object $o_3$ closely contacts query objects during period $[t_4, t_6]$, thus we update the query set to $Q = \{o_1, o_2, o_3\}$ at time $t_6$.

It is non-trivial to develop an effective and efficient solution to answering our CES problem. The straightforward method is incapable of handling massive-scale moving objects and ensuring real-time results. To address the challenge, we propose an efficient exact real-time search algorithm with various optimization techniques. Additionally, we propose an approximate real-time search algorithm that further improves the search efficiency without false dismissal. The main contributions of this paper can be summarized as follows.

- We define a novel CES problem that is able to continuously maintain up-to-date cases of exposure over a large number of moving objects. The CES problem is helpful to monitor and control the transmission of epidemics, such as SARS, Ebola, and COVID-19.
- We develop an exact real-time search algorithm and an approximate real-time search algorithm, namely Exact Group Processing (EGP) algorithm and Approximate Group Processing (AGP) algorithm, respectively, to answer the CES problem. We develop group filtering and pre-checking strategies to improve the search efficiency.
- The effectiveness and efficiency of our proposals are evaluated by extensive experiments over two real-life datasets. The experimental results show that our proposal is capable of detecting cases of exposure over massive-scale moving objects in a real-time fashion.

Related work. A similar problem to ours is co-movement pattern mining. Existing studies have discussed several types of interesting co-movement patterns [Jeung et al., 2008; Li et al., 2010; Vieira et al., 2009; Tang et al., 2012; Zheng et al., 2013]. Specifically, flock [Vieira et al., 2009] is the first proposed co-movement pattern, which captures all groups of in moving objects that travel together at $k$ consecutive time intervals. Among the literature, [Zheng et al., 2013] proposed a gathering pattern in the front of traffic congestion discovery. Besides, [Tang et al., 2012] discovered co-movement groups in a streaming manner. In contrast, the constraint on the number of objects in a group is not required in our CES problem. Note that objects outside the group may be considered infected under our problem definition. Therefore, solutions of these studies cannot be used for our problem.

To effectively prevent the transmission of infectious diseases such as SARS, Ebola, and COVID-19, a host of studies were developed [Eusuf et al., 2020; Xu et al., 2020; He et al., 2020], in which [Xu et al., 2020] is the first to introduce the contact tracing problem. However, it only focuses on simulating the disease and not capable of handling real-time queries. From another perspective, [Eusuf et al., 2020] presented a contact tracing query (CTQ) that identifies individuals who have direct or indirect contact with the query. [Chao et al., 2021] proposed a generalised trajectory contact search (TCS) query. An iterative algorithm is proposed to answer the TCS query. Another similar work [Alarabi et al., 2021] proposed a technique that traces all contacts exposed to a patient and searches for potentially infected persons. However, their solution is based on trajectory segments, and it is designed for indoor applications. To the best of our knowledge, none of the existing methods can be directly applied to solve our problem because they either target static trajectories or different definitions of cases of exposure.

2 Problem Formulation

We take a set of moving objects $O = \{o_1, \ldots, o_o\}$ and a set of query objects $Q (Q \subseteq O)$ as input. In real-life scenarios, for example, the moving objects can be pedestrians and the query objects can be people exposed to Coronavirus. We aim to detect cases of potential exposure in a real-time fashion. We proceed to define our important concepts in remaining parts of this section.

Location streams of moving objects. A location of a moving object $o$ at time $t$ is denoted by $L^o_t = ([x, y], t)$, where tuple $[x, y]$ is the spatial coordinates and $t$ is the timestamp. Note that $L^o_t$ is dynamically updated. We use $L^o_t = \{L^o_t \mid o \in O\}$ to denote the locations of moving objects.

Close contact. An object $o_1$ closely contacts another object $o_2$ if the distance between $o_1$ and $o_2$ is smaller than a distance threshold $\epsilon$.

Case of exposure. Given a duration threshold $k$, object $o$ $(o \in O \setminus Q)$ is a case of exposure if the duration of continuous close contact between $o$ and any object in $Q$ exceeds $k$.

Continuous Exposure Search problem. Given a collection of moving objects $O$, a set of exposed (query) objects $Q$, a social distance threshold $\epsilon$, and a contact duration threshold $k$, the Continuous Exposure Search (CES) problem targets to detect cases of exposure by updating $Q$ in a real-time fashion.

3 Exact Traversal Algorithm

A straightforward exact solution (ET) for answering our CES problem works as follow. At each timestamp, we check whether each object $o_i \in O \setminus Q$ closely contact each query object $o_j \in Q$. Specifically, if the distance between object $o_i$ and object $o_j \in Q$ is smaller than $\epsilon$, we record this object $o_i$ as a close contact. We periodically perform the aforementioned operations on incoming locations of all objects, and keep track of the continuous close contacts by maintaining a hash set at each time. We reset the contact duration of an object whenever it fails to keep being a close contact, namely, there is an interruption before its contact duration reaches $k$. We consider objects that have been consecutively recorded as
close contacts over a period of time \( k \) to be cases of exposure. Next, we update query set \( Q \) by adding up-to-date cases of exposure, and return updated \( Q \) as real-time results. Finally, the updated \( Q \) is used for subsequent round of detection.

**Time complexity.** Let \( |Q| \) be the number of query objects and \( |O| \) be the number of non-query objects. At each timestamp, we need to run ET accordingly. The time complexity of running each round of ET is \( O(|Q| \times |O|) \). It is computationally prohibitive of handling massive-scale moving objects and maintaining their real-time results.

### 4 Exact Group Processing Algorithm

To efficiently answer the CES problem while maintaining an exact up-to-date results for each moving object, we propose an Exact Group Processing (EGP) algorithm. The EGP algorithm does not compute all-pair distance between each object and each query. Rather, it partitions query objects and non-query objects based on their current location respectively and form query groups and non-query groups. Next, it discovers cases of exposure in two phases. In Phase 1, we compute all-group distances and generate potential cases of exposure in the group level. In Phase 2, we evaluate each potential cases of exposure in object level. In addition, we propose pre-checking strategies to enhance efficiency of EGP algorithm.

**Group distance.** To avoid all-pair distance computation, a host of distance metrics have been proposed to measure how far two groups of locations are from each other, in which Hausdorff distance [Taha and Hanbury, 2015] is one of the representative distance metrics. However, for two groups consisting of \( m \) and \( n \) locations, a single calculation of Hausdorff distance requires \( O(m \times n) \) time, making the overall computation expensive, especially for massive-scale locations.

**Group partition.** To enable fast discovery of close contacts, we partition all objects into groups based on their current locations. Objects in a group are spatially close to each other, making it possible for us to filter out far-away objects at an early stage. Specifically, we partition the underlying space by a grid index \( C \), each cell is a square with equal size \( \frac{\epsilon}{\sqrt{2}} \times \frac{\epsilon}{\sqrt{2}} \), where \( \epsilon \) denotes the distance threshold of close contact. Query objects and non-query objects falling into cell \( c \in C \) are recorded as a query group and a non-query group, which is denoted by \( O^q_c \) and \( O_c \), respectively. Note that query groups and non-query groups are stored and indexed separately. Figure 2 illustrates how we find object groups that have potential cases of exposure given a query group located in cell \( c_{ab} \) (marked with a star). From Figure 2 we see that only these objects falling into red cells (i.e., \( SC = \{ c_{ij} \in C || i - a| \leq 2\sqrt{2} \sqrt{\epsilon} \} \) are possible to closely contact a query object within \( \epsilon \), we regard these objects as potential cases of exposure. Note that \( |SC| \) is a constant, namely 21. For these object groups not in red cells (e.g., gray cells), they cannot closely contact any query object in \( c_{ab} \), because the distance exceeds \( \epsilon \).

**Pre-checking strategies.** To discover all close contacts in a non-query group \( O^q_c \) to a query group \( O^q_c \), it requires \( O(|O^q_c| \times |O^q_c|) \) time, which is time-consuming. Here, some pre-checking strategies are developed to avoid unnecessary computation based on our group partition. We first scan all locations of \( O^q_c \) and \( O^q_c \) to get the minimal bound range rectangles \( MBR(O^q_c) \) and \( MBR(O^q_c) \). Equation 1 defines a upper bound \( UB \) and a lower bound \( LB \) of \( \{ dist(l^o_i, l^o_j) | o_i \in O^q_c, o_j \in O^q_c \} \). Compared with Hausdorff distance, our pre-checking computation only requires a linear time complexity of \( O(|O^q_c| + |O^q_c|) \). If the upper bound \( UB(O^q_c, O^q_c) \) is smaller than \( \epsilon \), we regard all objects of \( O^q_c \) as close contacts. Meanwhile, we safely prune \( O^q_c \) if \( LB(O^q_c, O^q_c) > \epsilon \).

\[
UB(O^q_c, O^q_c) = \max (\{dist(MBR(O^q_c), MBR(O^q_c))\}) \\
LB(O^q_c, O^q_c) = \min (\{dist(MBR(O^q_c), MBR(O^q_c))\})
\]

**Algorithm details.** Algorithm 1 presents the pseudo code of our Exact Group Processing (EGP) algorithm. The input is a set of objects \( O \), a stream of locations \( L_t(O) = \{ l^o_i | o_i \in O \} \) consisting of locations of \( O \) at each timestamp \( t \), a set of query objects, a social distance threshold \( \epsilon \), and a contact duration threshold \( k \). The output is updated query object set, which consists of up-to-date cases of exposure. Initially, the contact duration \( cd(o) \) of all non-query objects is set to 0 (line 1). During the search process, we use \( \Pi \) to store these cells that cover at least one query object. We denote current close contacts by \( A \). At the beginning, \( \Pi \) and \( A \) are set to empty sets (line 2). The algorithm consists of two phases: 1) group partition (lines 3–11), and 2) group-level close contacts discovery (lines 12–20). During the group partition process, for each incoming location \( l^o_i \) of object \( o \), we obtain cell \( c \) that covers \( l^o_i \). If \( o \) is a query object, we store it in a query group \( O^q_c \) and add \( c \) to \( \Pi \); Otherwise, we store \( o \) in a non-query group \( O_c \) (lines 3–11). Up to this point, all query groups and non-query groups are obtained. During the close contacts discovery, we perform our pre-checking strategies regarding each query group and its nearby non-query groups with potential cases of exposure (lines 12–20). In particular, if the upper bound (cf. Equation 1) of the distance between a query group \( O^q_c \) and a non-query group \( O^q_c \) is not greater than threshold \( \epsilon \), we add all objects \( o \in O^q_c \) to current close contact set \( A \). Meanwhile, if the lower bound exceeds \( \epsilon \), we can safely prune \( O^q_c \) since any object in \( O^q_c \) is unable to make a close contact with an object in the query group. After the group-level discovery phase, we proceed to compute the pairwise distances of objects between each query group and its nearby non-query groups with potential cases of exposure (line 17). At the end
of each processing, we update the contact duration of all objects (line 21). Specifically, we reset the contact duration of these objects \( o \in \mathcal{O} \setminus \mathcal{A} \) to 0, indicating there is an interruption before their contact duration reaches \( k \). We update the contact duration of these objects \( o \in \mathcal{A} \) (i.e., up-to-date discovered close contacts) and update query set \( \mathcal{Q} \) by inserting up-to-date cases of exposure, where the cases of exposure are the objects with contact duration \( cd(o) \geq k \) (line 22). Finally, we output updated \( \mathcal{Q} \) as our real-time results (line 23).

**Time complexity analysis.** The time complexity for obtaining all query and non-query groups is \( O(|\mathcal{O}|) \). Let \( p \) and \( q \) be the number of objects in each query group and each non-query group. Discovering close contacts requires a time complexity of \( O(|\mathcal{P}| \times |\mathcal{S}| \times p \times q) = O(|\mathcal{P}| \times p \times q) \), where \( |\mathcal{S}| \) is a constant (cf. group partition). Note that in most cases, \( p \) and \( q \) are much smaller than \( |\mathcal{Q}| \) and \( |\mathcal{O}| \).

5 Approximate Group Processing Algorithm

To further improve the efficiency of discovering cases of exposure, we propose an Approximate Group Processing (AGP) algorithm.

**Head-tail-first checking strategy.** A non-query object cannot be a case of exposure if it fails to be a close contact before its contact duration reaches \( k \). Motivated by this, we propose the following Head-tail-first checking strategy. Instead of discovering close contacts chronologically, we first get the intersection results of \( \mathcal{A}_t \) (i.e., tail) and \( \mathcal{A}_{t-k} \) (i.e., head) to get the candidates if there are close contacts at timestamp \( t \), where \( \mathcal{A}_t \) denotes the close contact set at time \( t \). For period \( [t-k+1, t] \) we only need to find close contacts in the intersection result, which improves space and time efficiency.

**Hop checking.** Given a contact duration \( k \), if object \( o \) is a potential cases of exposure during \( k \) consecutive period of time \( [t_i, t_{i+k-1}] \), and it closely contacts at least one query at timestamp \( t_i \) and \( t_{i+k-1} \), we regard \( o \) as an approximate case of exposure. Based on approximate cases of exposure, we propose our hop checking strategy: Instead of processing locations of all timestamps, we only process locations for certain timestamps that \( t \) is outside \( T \), we assume that all objects are close contacts. The objective of hop checking is to further reduce computational effort without false dismissal.

**Algorithm details.** Algorithm 2 presents the pseudo code of our AGP algorithm. Compared with EGP, AGP additionally takes the hop length \( m \) as input. We use \( cd'(o) \) to represent the duration of \( o \) being an potential case of exposure. We initialize \( cd'(o) \) for every non-query object (line 1). Here, we only detect potential case of exposure only for certain specified timestamp, namely \( t \% m = 0 \) (line 2). We denote close contacts and potential case of exposure by \( B \) and \( A' \), respectively. For location \( l_t^o \in L_t(\mathcal{O}) \) of object \( o \), we perform the
same operations as the EGP to obtain the query group $O_q^c$, and the non-query group $O_c$ (lines 4–12). Next, we regard these objects in $SC(c)$ as potential case of exposure, and we increase their contact duration accordingly (line 13–17). Next, we reset the contact duration of objects $o / \notin A'$ (line 18). For objects with $cd'(o) \geq k$, we further check whether they are close contacts at timestamp $t$ and $t - k + 1$. If they meet the conditions, we regard them as approximate cases of exposure. We update $Q$ by adding these approximate cases of exposure (line 19). Finally, we return $Q$ as real-time results.

**Time complexity analysis.** The group partition requires $O(|O|)$ time. In addition, discovering potential case of exposure requires $O(|\Pi| \times |SC|) = O(|\Pi|)$ time. Thus, the time complexity of running each round of AGP algorithm is $O(|O| + |Q|)$. Note that the maximum value of $|\Pi|$ is $|Q|$.

6 Experimental Study

6.1 Experimental Setup

**Data preparation.** Our experiments are conducted on two real-world trajectory datasets. The first dataset is generated from the Beijing taxi data [Yuan et al., 2011], which contains trajectories of 10,357 cabs tracked over a period of one week in Beijing. The total number of points in this dataset is about 15 million. The second is collected in the city of Porto, Portugal (PT) over 19 months, which contains over 1.7 million trajectories. The average sampling interval in BJ is 177 seconds with a distance of about 623 meters, while in PT each taxi reports its location at 15-second intervals. The value of a timestamp is set to be within the range of 24h. We limit the range of longitude and latitude such that the sampled locations have sufficiently high density. In addition, we discard these trajectories with few sampled locations. More specifically, we only use trajectories with no less than 10 sampled locations in our experiments. To enable distance calculation among objects (e.g., taxis, pedestrians) all the time, we align all trajectories through linear interpolation, such that each object returns a real-time location with a fixed frequency 10s and 5s for BJ and PT. As a result, the datasets contain 296,364,033 and 23,767,470 valid locations in BJ and PT, respectively. Without loss of generality, we randomly select some objects $Q \in O$ as initial query objects.

**Compared algorithms.** We compare the performance of our proposal algorithms, namely exact traversal algorithm (ET), group processing algorithm (EGP), and the approximate algorithm (AGP). For AGP, the default set of hop length is 2. The performance metrics for efficiency evaluation and efficacy evaluation are CPU time and the total number of discovered (approximate) cases of exposure. For simplicity, we use $|CE|$ to denote the number of (approximate) cases of exposure. The CPU time is the averaged run time of running each round. Besides, we study the accuracy of AGP, which is calculated by the ratio of the number of cases of exposure to the number of approximate cases of exposure.

**Parameter settings.** The default parameter settings are listed in Table ??, which is based on specific data distribution of the two datasets. All methods were implemented in

<table>
<thead>
<tr>
<th></th>
<th>Beijing</th>
<th>Porto</th>
</tr>
</thead>
<tbody>
<tr>
<td>Latitude</td>
<td>[39.830, 40.030]</td>
<td>[40.953, 41.307]</td>
</tr>
<tr>
<td>$\epsilon$</td>
<td>{2, 4, 6, 8, 10}</td>
<td>{2, 4, 6, 8, 10}</td>
</tr>
<tr>
<td>$k$</td>
<td>{5, 10, 15, 20, 25}</td>
<td>{5, 7, 9, 11}</td>
</tr>
<tr>
<td>$</td>
<td>Q</td>
<td>$</td>
</tr>
</tbody>
</table>

Table 1: Parameter Settings
Java and evaluated on Windows 10 platform equipped with an AMD Ryzen 5 CPU (3.6GHz) and 32GB memory. Unless stated otherwise, the experimental results are averaged over 20 independent trials with different query objects as inputs.

Effect of the number of query objects. First, we investigate the effect of the number of query objects $|Q|$ on the performance of the proposed algorithms with the default setting. A significantly increasing trend of the number of cases of exposure is observed in Figure 3(c), while the trend on BJ is not obvious. It is clear that the number of discovered cases of exposure of ET is the same as EGP, for they are both exact algorithms. The results of the three algorithms are reasonably close. As shown in Figure 3(b) and Figure 3(d), the group processing algorithm performs slightly worse than the approximate solution in CPU time. The required CPU time of ET exceeds 1,000 ms for processing Beijing datasets, demonstrating that ET cannot be used for answering real-time queries. Compared with ET, EGP and AGP reduce the required CPU time by at least one order of magnitude, which demonstrates the superiority of our proposal.

Effect of the distance threshold $\epsilon$. Intuitively, a larger value of $\epsilon$ means that two objects are easier to make close contact with a query object. Figure 4 shows the performance when we vary the distance threshold $\epsilon$ from 2 meters to 10 meters. As expected, the number of cases of exposure increases with the increase of $\epsilon$ on both BJ and PT for all algorithms. From Figure 4(a), the difference of the results of ET and AGP narrows as we increase $\epsilon$ on BJ. However, the difference becomes more significant as we increase $\epsilon$ on PT. Such case may attribute to their different data distribution. It is worth noting that there is no obvious trend of runtime for ET algorithm, because the required computation effort of ET greatly depends on the number of queries.

Effect of the contact duration $k$. We study the effect of the contact duration threshold $k$. A smaller $k$ indicates that objects are easier to become cases of exposure, which results in more cases of exposure, thus much more CPU time is required at each snapshot. From Figure 5(a) and 5(a), we can see that the number of cases of exposure greatly decreases on both Beijing and Porto for all algorithms, as we vary $k$ from 10 to 40. In addition, a slightly decreasing trend of CPU time is observed in Figure 5(b) and 5(d).

Evaluation on the approximate algorithm. Figure 6 shows the accuracy when we vary the number of query objects $|Q|$ and the social distance threshold $\epsilon$. The AGP results in a high accuracy on Beijing dataset, it does not show obvious performance degradation when we vary $|Q|$ and $\epsilon$. However, the accuracy degrades on Porto datasets.

Evaluation on pre-checking strategies. The EGP solution without pre-checking strategy is denoted by “EGP*PC”. In Figure 7, as we vary the number of query objects, we observe that the required CPU time of EGP is consistently less than those of EGP*PC.

7 Conclusion
We proposed and investigated a novel exposure search problem that finds cases of exposure for a stream of trajectory locations (CES problem). To address the problem, two efficient algorithms were proposed, in which pre-checking strategies were developed to enhance the efficiency. Extensive experiments confirmed that our proposal was capable of achieving high efficiency and high effectiveness, and the pre-checking strategies were helpful to avoid unnecessary computation.

Acknowledgments
This work was supported by the NSFC (U21B2046, U2001212, 62032001, and 61932004) and Sichuan Science and Technology Program (No.2021YFS0007). Bin Yao was supported by the NSFC (61922054, 61872235, 61832017).
References


[Li et al., 2010] Zhenhui Li, Bolin Ding, Jiawei Han, and Roland Kays. Swarm: Mining relaxed temporal moving object clusters. Proc. VLDB Endow., 3(1):723–734, 2010.


[Tang et al., 2012] Lu-An Tang, Yu Zheng, Jing Yuan, Jiawei Han, Alice Leung, Chih-Chieh Hung, and Wen-Chih Peng. On discovery of traveling companions from streaming trajectories. In IEEE 28th International Conference on Data Engineering (ICDE 2012), pages 186–197. IEEE Computer Society, 2012.


