A Two-Phase Anomaly Detection Model for Secure Intelligent Transportation Ride-Hauling Trajectories

Asma Belhadi, Youcef Djenouri, Gautam Srivastava (*Senior Member, IEEE*), Djamel Djenouri, Alberto Cano (*Senior Member, IEEE*), and Jerry Chun-Wei Lin (*Senior Member, IEEE*)*

***Abstract*—This paper introduces a new approach to identify both individual and group trajectory outliers from ride-hauling (taxi) trajectory databases. It proposes two approaches for efficiently solving this problem: i) a *Two Phase-based algorithm*, which consists of two main phases, the former determines the individual trajectory outliers by computing the distance of each point in each trajectory, whereas the latter identifies the group trajectory outliers by exploring the individual trajectory outliers using both feature selection and sliding windows strategies, and ii) a *GPU-based* approach, which benefits from the massively GPU computing and the sliding windows strategy to boost the runtime performance of the two phase-based algorithm. Extensive experiments have been carried out to thoroughly demonstrate the usefulness of our methodology on both synthetic and real trajectory databases. Experimental results show the efficiency of the GPU approach compared with the sequential approach by reaching an speedup of 341 using large synthetic databases. Moreover, we show the usefulness of the proposed methodology to detect both individual and group trajectory outliers on a real-world taxi trajectory database. We also compare both the serial and the parallel approaches with the baseline trajectory outlier and group detection algorithms. The results are very promising in terms of both computational time and the quality of returned outliers. Finally, we prime our methodology and results for future refinement using deep learning methodologies.**

***Index Terms*—Trajectory Database, Outlier Detection, Individual Trajectory Outliers, Group Trajectory Outliers, Taxi Frauds, GPU Computing.**

I. Introduction

In recent years, due to the growth of database systems and advances in high-support GPS technologies, countless numbers of sequence points represented by trajectory databases are generated, stored, and analyzed. These trajectory databases simulate various behaviours of different objects in several real-world applications such as intelligent transportation [1]–[6], mobile traffic network [7]–[10], and climate change analysis [11], [12]. Specifically, in the case of intelligent transportation, data analysts face a myriad of trajectories derived from mobility of people, cars, buses, and taxis through inter or intra smart cities.

Outlier detection, also known to anomaly detection, is broadly used in the data mining community where unusual observations, objects, and/or points are derived from normal observations [13]–[18]. Finding the outliers is considered as a major problem for providing security for intelligent and transportation systems. In the context of trajectory analysis, outlier detection aims to discover trajectories or sub-trajectories that do not conform with the rest of trajectories in a database [19]–[21]. Current solutions to trajectory outlier detection only consider single view of outlierness in a whole trajectory or a sub-trajectory. However, in real-world scenarios, different types of trajectory outliers and useful features could be identified such as **ITF**: Individual taxi Trajectory Fraud, **GTF**: Group of taxi Trajectory Fraud, **ICP**: Individual Change Point, and **GCP**: Group Change Point.

Example 1: Group of Taxi trajectory Fraud. Consider the example of ride-hauling (taxi) trajectories illustrated in Fig. 1. Here, each trajectory is mapped to the road map network of the United States. Expected trajectories are illustrated in black color. Traditional taxi trajectory fraud algorithms [20], [22], [23] may detect outliers illustrated in green color, where a *taxi* goes from *Minneapolis, Minnesota* to *Denver, Colorado* and deviates through *South Dakota and Wyoming*. The deviation may be legit, e.g. due to road maintenance, or be a fraud.

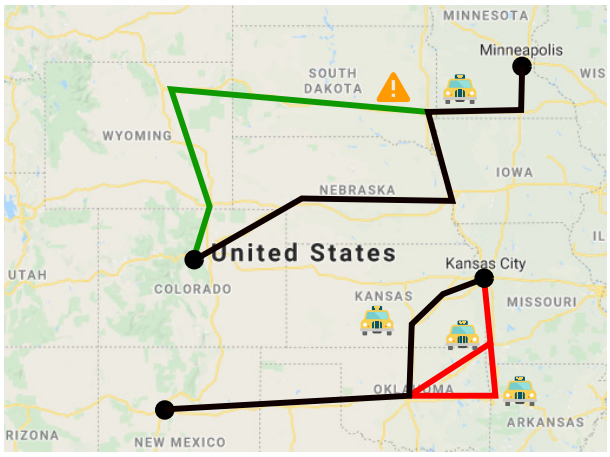


Fig. 1. Motivating Example I: Group of Taxi trajectory Fraud

A. Belhadi is with the Department of Computer Science, USTHB, Algiers, Algeria.

Y. Djenouri is with the SINTEF Digital, Forskningsveien 1, 0314, Oslo, Norway.

G. Srivastava is with the Department of Mathematics & Computer Science, Brandon University, Manitoba, Canada
and with the Research Centre for Interneural Computing, China Medical University, Taichung, Taiwan.

D. Djamel is with the Department of Computer Science & Creative Technologies, Computer Science Research Centre, University of the West of England, Bristol, UK.

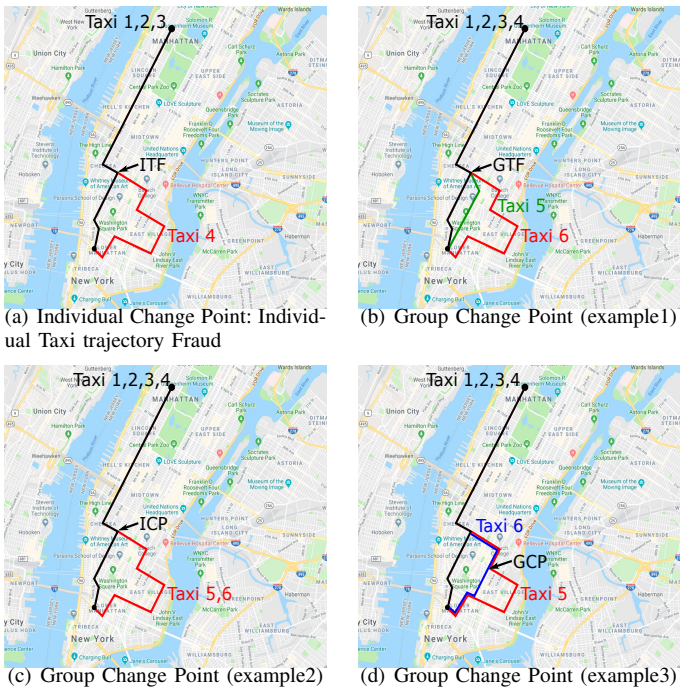
A. Cano is with the Department of Computer Science, Virginia Commonwealth University, Richmond, VA, USA.

J. C. W Lin is with the Department of Computing, Mathematics, and Physics, Western Norway University of Applied Sciences, Bergen, Norway. (*Corresponding author)

Manuscript received May, 2020.

It is relatively easy to identify the anomaly when a single trajectory deviates from the historic trajectories for the same route. However, traditional algorithms cannot identify outliers presented in red color, where a group of taxis deviate from their usual trajectory, for the trip from *Kansas City, Missouri* to *Albuquerque, New Mexico*. Detecting these trajectory outliers where the group collectively deviate, not necessarily though the same paths, is not straightforward. Detecting such outliers could help planners to study the different correlations between these trajectories to deduce useful and pertinent information. For instance, taxi trajectories shown in *red* color could indicate that the taxis are partners in a potential taxi fraud, whereas for the *green* trajectory only *individual* deviation is observed. In this work, we focus on studying and determining outliers where multiple taxis may partner in fraud as a group. To perform this, we first define a new problem called *Group Trajectory Outlier Detection*. Next, we propose different approaches for finding these kind of anomalies through analysing real-world ride-hauling datasets.

Example 2: Change points detection. Consider the four sketched examples of taxi trajectories illustrated in Fig. 2. Each taxi trajectory starts from the source point and ends at the destination point. Traditional trajectory outlier detection algorithms [24]–[27] may detect outliers illustrated in Fig. 2(a) in red color, where *taxi4* follows the normal trip from the source to the destination until a given point, in which it deviates from the taxis *taxi1*, *taxi2*, and *taxi3*. Traditional trajectory outlier detection algorithms cannot determine the individual change point for this case. Moreover, the whole process is not able to identify outliers presented in red and green colors in Fig. 2(b), 2(c), and 2(d). In Figure 2(b), different taxis (*taxi5* and *taxi6*) deviate from their normal trips at the same group change point but follow different trajectories (green trajectory for *taxi5* and red trajectory for *taxi6*). In the case illustrated in Figure 2(c), two taxis deviate from their normal trip at the **same** group change point and follow the **same** trajectory shown in red color. Moreover, the last case sketched in Figure 2(d), taxis (namely *taxi5* and *taxi6*) deviate from their normal trip at different group change points but follow the same trajectory. Detecting these different kinds of outliers and change points could help city planners to extract patterns and discover relevant knowledge through careful analysis of the trajectories. For instance, the case of Figure 2(a) may allow the determination of individual taxi fraud through detecting frequently individual taxi fraud at the same individual change point. These actions can support city planners making good decisions such as putting surveillance cameras at known common change points. In the case of Figs. 2(b), 2(c), and 2(d), detecting group trajectory outliers allows city planners to make fair decisions regarding taxis outliers. Taxis which deviate from the same group change point but follow different trajectories have a strong probability that their aim is to avoid malicious circumstances like traffic jams rather than committing taxi fraud. However, group taxis outliers at the same or different group change points with the same trajectory have a strong probability that they are partners in taxi fraud. The problem related to the existing taxi fraud detection algorithms is how to derive individual and group change points efficiently.



(a) Individual Change Point: Individual Taxi trajectory Fraud

(b) Group Change Point (example1)

(c) Group Change Point (example2)

(d) Group Change Point (example3)

Fig. 2. Motivating Example II: Change Points Detection.

Contribution: To answer the previous questions, this paper presents a new approach that allows identifying taxi trajectory outliers with different contexts by detecting taxi individual outliers and groups of taxi outliers. We also investigate on detecting the individual and group change points of different taxis. The main contributions of this work are summarized as:

1) We propose a general two phase-based algorithm for taxi frauds. The first phase aims to identify taxi trajectory outliers based on the distance of each point in each taxi trajectory. The second phase aims to explore the taxi trajectory outliers to derive the group of taxis outliers by using feature selection.
2) We incorporate a GPU-based version of the two phase-based algorithm with the sliding windows strategy to boost the performance and scalability of the outlier detection process on large-scale taxis trajectories and point spaces.
3) We evaluate our novel methodology in several well-known trajectory databases and study the scalability of the algorithms in terms of runtime, the number of outliers, and speedup by analyzing the performance under different parameters. We illustrate the performance of the proposed algorithms on a real-world taxi trajectory database. We compare both the serial and the parallel approaches with the baseline trajectory outlier and group detection algorithms. The results are very promising in terms of both computational time and the quality of returned outliers. We also link our results gathered in this paper to future enhancements using deep learning methods under future work.

Outline: The remainder of the paper is organized as follows. Section II reviews the main existing trajectory outlier detection algorithms, followed by a detailed explanation of our framework of multi-view trajectory outliers in Section III. Section IV presents the performance evaluation. Finally, Section V concludes the paper and outlines our future work.

II. Related Work

Qin et al. [28] developed a probability-based taxi fraud detection algorithm which consider both the behavior of taxi drivers and the traffic variability. The set of route choice of all taxi drivers is first generated from the taxi trajectory database. The choice probability of each route is then calculated by joining all taxi driver choice taking this route. The anomaly score of each taxi driver is finally determined using the probability values of all routes visited by such taxi driver. Ammar et al. [29] developed a smart system which integrate the images databases in detecting anomalies. The convolution neural network is applied to extract both the trip, and engineering features. The random finite set algorithm is then performed to derive outliers. Zhang *et al.* [30] proposed the iBAT (Isolation-Based Anomalous Trajectory) algorithm. The "*few and different*" properties of anomalous trajectories are then exploited where a random tree is generated by dividing the trajectories until almost all of them are isolated. This generation produces shorter paths for anomalous trajectories which are isolated faster than normal trajectories isolated in longer path. Chen *et al.* [31] proposed the iBOAT (Isolation-Based Online Anomalous Trajectory) algorithm. This algorithm aims to find anomalous taxi sub-trajectories in real time. It is used to automatically detect fraud implied by rapacious taxi drivers who take unnecessary diversions during the trip. When the outlier sub-trajectory is determined, the notification of possible fraud could be suggested to the passenger, even the taxi is still in use. Ge et al. [32] introduced a route-based structure to represent the taxi driving paths. This allows to expand the generative statistic approach to analyze the distribution of taxi driving, and identify the taxi frauds. The same authors [33] investigated the use of data mining in developing a taxi business intelligence system to help taxi companies in understating the behavior of taxi drivers. The taxi drivers are grouped into similar clusters, where on each cluster the taxi frauds are identified by exploring heterogeneous trajectory GPS traces using Demptser-Shafer theory. In the same context, Yuan et al. [34] suggested a smart taxi fraud detection system by exploring a large taxi GPS logs. Both occupied and unoccupied taxi trajectories are mapped to the road network. The source-destination trajectories are retrieved from the mapped trajectory database. The iBAT algorithm is then applied to derive the taxi frauds. The framework also ale to rank taxi drivers and distinguish the best drivers from the drivers causes frauds. Leng et al. [35] studied the two Chinese taxi system, Didi and Kuaidadi, in analyzing taxi frauds from large taxi trajectories database. The correlation among spatio-temporal traveling patterns are determined using density values, this allows to detect hot spots, and isolated regions, which helps to deduce the taxi frauds. Zhang et al. [36] investigate prediction of taxi destination in the taxi fraud system. The time-related feature pre-processing step is first processed to accurately embed the data. A data-driven ensemble learning solution, which combines the merits of the support vector machine, and the deep learning to identify segment deviation of different taxi trajectories. Wang et al. [37] analyzed different anomalous patterns from taxi trajectory data. The difference and intersection metric set is calculated between each two pair of taxi trajectories, in order to define the anomaly scores. Anomalous trajectory detection, and classification algorithm is then proposed to analyze the different taxi trajectory frauds. Xudong et al. [38] applied a probabilistic tensor factorization algorithm to measure the expected and predicted probability of each taxi trip. Each taxi trip is assumed as an observation on multivariate data distribution. The tucker decomposition with an accurate expectation maximization are used to approximate such distribution of taxi trips. Smolyak et al. [39] addressed the lack of ground truth in finding the trajectory anomalies, and developed a hybrid generative adversarial network with infinite Gaussian mixture model to generate synthetic and realistic trajectory data and therefore facilitate trajectory anomaly detection.

As can be seen from the literature overview, existing solutions for taxi frauds focus on discovering individual taxi trajectory frauds. In addition; there is no work which identifies the change points of taxi trajectory frauds. Therefore, in this paper paper we propose the first dedicated algorithms to detect both individual and group taxi trajectory frauds and also to detect the change points.

III. Methodology

A. Problem Formulation

To introduce the multi-view taxi frauds problem we need a few preliminary definitions. A taxi trajectory is a sequence of taxi location points in space. We will denote by p a single spatial location point, where each p is a tuple of two values, the latitude and the longitude of this location.

Definition 3.1 (Taxi Trajectory Database): We define a taxi trajectory database $\Lambda = \{\Lambda_1, \Lambda_2, \ldots, \Lambda_m\}$, where each raw taxi trajectory Λ_i is a sequence of spatial taxi locations points with timestamps $\{p_{i1}, p_{i2}, \ldots, p_{in}\}$ obtained by localization techniques such as GPS.

Definition 3.2 (Candidate Taxi Sub-Trajectory Outlier): We define a candidate taxi sub-trajectory outlier Λ_i^{xy} as a sub-sequence of points $p_{ix}, p_{ix+1}, \ldots, p_{iy}$ in the whole trajectory Λ_i.

Definition 3.3 (Order Relation): A point p_1 is highly ordered with respect to p_2 in the trajectory database if and only if the time of p_1 comes before the time of p_2.

A recent survey in the literature [40] details how the location points which are similar enough are aggregated into regions. Let us denote by R a location *region* in space.

Definition 3.4 (Mapped Taxi Trajectory Database): We define a mapped taxi trajectory database $\Lambda = \{\Lambda_1, \Lambda_2 ... \Lambda_m\}$, where each mapped taxi trajectory Λ_i is a sequence of spatial location regions $\{R_{i1}, R_{i2} ... R_{in}\}$, obtained by replacing each point p_{ik} in Λ_i with its region R_{ik}.

Definition 3.5 (Taxi Trajectory Similarity): The similarity between two taxi trajectories Λ_i and Λ_j, denoted as $d(\Lambda_i, \Lambda_j)$, is defined by their symmetric difference, i.e., the number of all regions in the two taxi trajectories minus the number of shared regions in the two taxi trajectories. Formally:

$$d(\Lambda_i, \Lambda_j) = n - \{|(R_{il}, R_{jl})|, \forall\ l \in [1..n]\} \quad (1)$$

Definition 3.6 (Group Taxi Frauds): We define the set of the candidate group taxi frauds $\mathcal{G} = \{\Lambda_1^{s_1,e_1}(\mathcal{G}), \Lambda_2^{s_2,e_2}(\mathcal{G})...\Lambda_{|\mathcal{G}|}^{s_{|\mathcal{G}|},e_{|\mathcal{G}|}}(\mathcal{G})\}$*, where the starting point* s *of each taxi sub-trajectory* $\Lambda_i^{se}(\mathcal{G})$ *should be highly ordered an ending point* e *of at least one another taxi sub-trajectory in the same group* $\mathcal{G}$*.*

Definition 3.7 (Density Point): We define the density of the point p_{ij} *as follows:*

$$\mathcal{DP}(p_{ij}) = \frac{\sum_{l=1}^{m} distance(p_{ij}, p_{lj})}{m} \quad (2)$$

where m *is the number of all taxi trajectories. Note that the distance between two points* p_1*, and* p_2 *is computed using the spatial information (latitude:* $p_1.x$*,* $p_2.x$ *and longitude:* $p_1.y$*,* $p_2.y$*) in the Euclidean space as follows:*

$$distance(p_1, p_2) = \sqrt{(p_1.x - p_2.x)^2 + (p_1.y - p_2.y)^2} \quad (3)$$

The density of a group is an important concept in our analysis. Intuitively, it is defined as the number of shared regions between all the taxi trajectories of the group.

Definition 3.8 (Group Density): We define the density of the candidate group of taxi trajectory frauds, $\mathcal{G}$, as

$$Density(\mathcal{G}) = |\{R|\forall \Lambda_i \in \mathcal{G}, R \in \Lambda_i\}| \quad (4)$$

Definition 3.9 (Individual Taxi Trajectory Fraud Problem): The Individual Taxi Trajectory Fraud Problem aims to discover for a given density threshold μ*, the set of all individual taxi trajectory frauds* $\mathcal{I}^*$ *such as:*

$$\mathcal{I}^* = \{\Lambda_i^{se} | \forall j \in [s, \ldots, e], p_{ij} \in \Lambda_i^{se}, \mathcal{DP}(p_{ij} \leq \mu)\} \quad (5)$$

Definition 3.10 (Group Taxi Trajectory Fraud Problem): The Group Taxi Trajectory Fraud Problem aims to discover a group taxi trajectory fraud $\mathcal{G}^*$ *such as:*

$$\mathcal{G}^* = \max_{\mathcal{G}} \{\mathcal{DG}(\mathcal{G})\} \quad (6)$$

B. Two Phase Approach

Before presenting the Two Phase-based approach, we define some useful concepts.

Definition 3.11 (ITF: Individual Taxi trajectory Fraud): We define ITF by the set of individual taxi trajectory fraud where each taxi trajectory in ITF satisfies the requirements in Eq. 5.

Definition 3.12 (GTF: Group Taxi trajectory Fraud): We define GTF by the set of group taxi trajectory fraud where the trajectories in GTF satisfy the requirements in Eq. 6.

Definition 3.13 (ICP and GCP: Individual and Change Points): We define ICP (Individual Change Point) by the point that causes frauds of the set ITF. Similarly, GCP (Group Change Point) is the point that causes frauds of the set GTF.

Algorithm 1 Two Phase-based algorithm

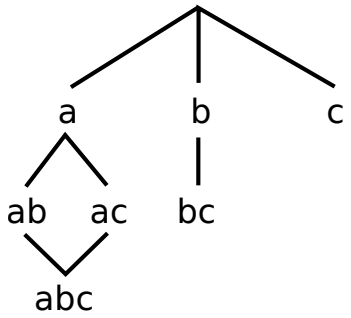
Input:
$\Lambda = \{\Lambda_1, \Lambda_2..., \Lambda_m\}$: The taxi trajectory database.
μ: The density point threshold.
γ: The density group threshold.
Output:
ICP: Individual Change Point.
ITF: Individual Taxi trajectory Fraud.
GCP: Group Change Point.
GTF: Group Taxi trajectory Fraud.
{**First Phase: Determine individual taxi trajectory frauds**}
for i=1 to m **do**
  for j=1 to n **do**
    $d_{ij} \leftarrow \mathcal{DP}(p_{ij})$; {See Def. 3.7}
  end for
end for
$ICP \leftarrow \emptyset$;
$ITF \leftarrow \emptyset$;
$flag \leftarrow false$;
$c \leftarrow 1$;
for *i*=1 to *m* **do**
  for *j*=1 to *n* **do**
    if $d_{ij} \leq \mu$ **then**
      if flag=false **then**
        $AddElementToList(ICP[i], p_{ij})$;
        $ICP \leftarrow ICP \cup p_{ij}$;
        $CreateList(ITF_c[i], p_{ij})$;
        $flag \leftarrow true$;
      else
        $AddElementToList(ITF_c[i], p_{ij})$;
      end if
    else
      $flag \leftarrow false$;
      $ITF \leftarrow ITF \cup ITF_c[i]$;
      $c \leftarrow c + 1$;
    end if
  end for
end for
{**Second Phase: Determine group taxi trajectory frauds**}
$Open \leftarrow \emptyset$;
for *i*=1 to *m* **do**
  if $ICP[i] \neq \emptyset$ **then**
    for *j*=1 to *c* **do**
      $AddElementToList(Open, ITF_j[i])$;
    end for
  end if
end for
while $Open \neq \emptyset$ **do**
  $node \leftarrow RemoveFirstElement(Open)$;
  $\mathcal{DG}(node)$; {See Def. 3.8}
  $AddElementsToList(Open, GPN(node))$; {With respect to Def. 3.6}
  $Best \leftarrow SaveBest(Open)$;
end while
$GTF \leftarrow Best$;
$GCP \leftarrow LastPoint(ICP)$;
return (ICP, ITF, GCP, GTF);

Proposition 3.1: A sub-trajectory Λ_i^{xy} *belongs to a group trajectory outliers if and only if* Λ_i^{xy} *is an individual trajectory outlier.*

Based on Proposition 3.1, this approach is performed into two main phases, first determining individual trajectory outliers, and then determining group trajectory outliers as follows:

1) **Determining individual trajectory outliers.** In this phase, the individual trajectory outliers are derived. The process starts by computing the density of each point in the whole set of trajectories Λ using the point density measure (See Def. 3.7). This allows to derive the individual change point, if it exists, for each trajectory. From the individual change point, the process continues to determine the individual trajectory outlier. For each point highly ordered to the individual change point, if the density of this point is less than a density threshold μ, then this point is added to the individual trajectory

(a) First Phase



(b) Second Phase

Fig. 3. Two Phase Illustration

outlier started by such individual change point. This process is repeated until a normal point is found (its density is greater than μ). The whole process is repeated until all points of all trajectories are scanned.

2) **Determining group trajectory outliers** After determining the individual trajectory outliers, the next step is to derive the group trajectory outliers. A feature selection technique is evaluated, where each individual trajectory outlier is considered as one feature, and the aim is to select the most relevant features from the whole individual trajectory outliers. The relevant set of features is then considered as group trajectory outliers. The evaluation of the selected features is computed using the group density measure (See Def. 3.8), in which the aim is to maximize this measure. In this context, a depth first strategy is used by starting with the empty node $\emptyset$ that contains any trajectory, to the full node that contains all individual trajectory outliers. The group change point is finally obtained, which is the last individual change point of all trajectories in each group trajectory outliers.

Algorithm 1 presents the pseudo-code of the Two Phase-based algorithm for solving the multi-view trajectory outlier detection problem. The input consists of the trajectory database Λ, the density point, and the density group thresholds. The output comprises the sets ICP, ITF, GCP, and GTF. The algorithm starts by computing the density of each point in the trajectory database (from line 4 to line 8). It then constructs the sets ICP and ITF using the μ threshold (from line 13 to line 30). The depth-first search strategy is then applied to each ITF_j to determine the best features (group taxi trajectory fraud) with the group change point (from line 33 to 47). The algorithm uses some pre-defined methods:

1) *AddElementToList(L, e)*: Add an element *e* at the end of the list L.
2) *AddElementstoList(L, E)*: Add all elements in E at the end of the list L.
3) *CreateList(L, e)*: Create a new list L, and assign an element *e* as the head of this list.
4) x $\leftarrow$ *RemoveFirstElement(L)*: Assign to x the first element of the list L before removing it.
5) *SaveBest(L))*: Return the current best element in the list L regarding the $\mathcal{DG}$ formula.
6) *LastPoint(L)*: Return the last point of the list L, regarding the order relation definition (Def. 3.3).

The complexity of this algorithm depends on the number of trajectories m, the number of points n, and the number of the individual trajectory outliers c generated in the first stage. The cost is equal to the sums of costs of the first and the second phases. In the first phase, the density computation requires $m^2 \times n$ operations, the construction of the individual trajectory outliers is performed on m operations, the total cost of this stage is $(m^2 \times n) + m$. For the second phase, the feature selection is performed on the c individual trajectory outliers, which generates $(2^c - 1)$ nodes, and each node is evaluated on c operations cost. The total cost of this phase is $c \times 2^c - c$. The total complexity cost of the Two Phase-based algorithm is $O(m^2 \times n + c \times 2^c)$. From this complexity analysis, we can remark that two phase-based algorithm is quadratic on the number of trajectories, polynomial on the number of points, and exponential on the number of individual trajectory outliers. The overall performance of such algorithm is reduced when the number of trajectories becomes too large. In the next section, we propose a GPU-based sliding window algorithm for boosting the performance of the two phase-based algorithm.

C. GSW-TP: GPU-based Sliding Window for Two Phase-based algorithm

The aim of this approach is to improve the overall performance of the two phase-based algorithm proposed in the previous section using both a sliding window strategy and the GPU architecture. Graphical Processing Units (GPU) have been recently used for solving complex problems [41]–[45]. The GPU programming model consists of many GPU threads that are logically grouped into several blocks of threads. Each thread in a block shares a memory space with the other threads in the same block. All blocks have also access to constant and global memories. Threads are grouped into warps of 32 threads, and thread blocks of up to 1024 threads. Taxi trajectories database is first divided into k sliding windows,

where k is the number of the GPU-blocks used in the mining process. These sliding windows are then sent to the shared memory of the GPU, where each block b_i is mapped to one sliding window SW_i. The j^{th} thread in b_i, th_{ij}, determines the local individual trajectory outliers by performing the lines 4 to 30 on the trajectory Λ_j. In the case of the density of the ending point of the trajectory Λ_j on the sliding window SW_{i-1} and the density of the starting point of Λ_j on the sliding window SW_i are less than μ, the part of the individual trajectory outlier Λ_j on SW_{i-1} is concatenated with the part of Λ_j on SW_i, and the average point will be the average of both parts of Λ_j. Afterwards, the threads of each block compute the local average points of each individual trajectory outliers. Each block then finds the local group trajectory outlier at each sliding window. A global group trajectory outlier will be a local trajectory outlier that maximizes a function described in Def. 3.10. From a theoretical standpoint, GSW-TP improves the sequential version of the two phase-based algorithm by exploiting the massively threaded computing of GPUs while determining both individual and global trajectory outliers. GSW-TP also minimizes the CPU/GPU communication by defining only two points of CPU/GPU communication. The first one takes place when the trajectory database is loaded into the GPU in the beginning, and the second one when the individual and the global trajectory outliers are returned to the host memory. Moreover, GSW-TP minimizes threads divergence, which is a typical issue in GPU-based computing. Threads divergence only takes place when the threads of different blocks and belong to the same wrap process different number of individual trajectory outliers. However, this issue needs several GPU synchronization points.

IV. Performance Evaluation

Thorough experimental analyses have been carried out to evaluate the proposed framework using both synthetic and real-world trajectory databases. The synthetic trajectory databases are first used to study the behavior of the two proposed algorithms (the two phase-based and the GSW-TP) with varying the number trajectories, the number of points, and the number of GPU blocks/threads. A sketch of individual and group trajectory outliers are shown on a real-world case study on a trajectory database [46], which contains $1,710,671$ different taxi trajectories. Regarding the quality, a common problem of outlier detection techniques is the evaluation procedure of the quality of returned outliers, in particular for new applications such group trajectory outliers, where a ground truth is not defined or does not exist. To facilitate a quantitative evaluation, we inject synthetic group trajectory outliers as follows:

Injecting individual trajectory outliers: Individual trajectory outliers are generated by adding noise *several times* with a certain probability $p \sim \mathcal{U}(0.8, 1.0)$ and a given threshold μ.

Injecting group trajectory outliers: From the individual trajectory outliers, noise are added *few times* with a certain probability $p \sim \mathcal{U}(0.0, 1.0)$ and a given μ.

For both injections, each point p_{il} in the trajectory Λ_i is changed as in Equation 7.

$$p_{il} = \begin{cases} p_{il} + n \sim \mathcal{N}(0,1) & \text{if } p \geq \mu \\ p_{il} & \text{otherwise.} \end{cases} \quad (7)$$

Thus, the evaluation of the returned outliers is performed using F-measure, and mAP (mean Average Precision), which are common measures for the evaluation of taxi fraud methods. They are defined as:

1) **F-measure.** It combines the precision and recall measures as follows:

$$Fmeasure = \frac{2 \times Recall \times Precision}{Recall + Precision} \quad (8)$$

where $Recall = \frac{|CRF|}{|F|}$ is the ratio of the number of correct retrieved fraud (CRF) to the total number of all frauds (F), and $Precision = \frac{|CRF|}{|RF|}$ is the ratio of the number of correct retrieved frauds (CRF) to the total number of retrieved frauds (RF).

2) **mAP.** It is computed as:

$$mAP = \frac{\sum_{i=0}^{n} AvgP(i)}{n}, \quad (9)$$

where n is the number of all frauds to be identified, and $AvgP(i)$ is the precision at rank i, i.e., the first i ranked frauds considered while the remaining frauds are ignored.

The sequential experimentation were run on a computer with 64 bit core i7 processor running Windows 10 and 16 GB of RAM. The parallel code has been implemented in the CUDA language. The CPU is an Intel Xeon E5520 2.27 GHz with 2 GB RAM. The GPU is a NVIDIA Tesla C2075 with 448 CUDA cores (14 multiprocessors with 32 cores each), a clock speed of 1.15 GHz, 2.8 GB of global memory, 48 KB of shared memory. Both the CPU and GPU are used in single precision mode. The parallel algorithm was evaluated in terms of the speedup compared to the sequential version.

A. Synthetic Trajectory Databases

Fig. 4. (a) presents the runtime performance of the two phase-based pattern mining algorithm using synthetic trajectory databases by varying the number of points from 100 to $1,000$, the number of trajectories from $1,000$ to $50,000$, and with different grid sizes ($100 * 100$, $200 * 200$, and $500 * 500$). The results reveal that by increasing the parameters values, the runtime of the two phase-based algorithm is highly increased. Thus, for 100 points, and $1,000$ trajectories, and with grid size $100 * 100$, the Two Phase-based algorithm needs 12 seconds, however, for mining big trajectory databases, with $1,000$ points, $50,000$ trajectories, and $500 * 500$ as grid size, the two-phase based algorithm needs $2,000$ seconds to retrieve the taxi frauds, and the change points. The reason of obtaining these results is the complexity of the mining process to derive both individual and group taxi trajectory frauds, where two expensive steps are needed to do such task efficiently. Moreover, the recursive process in retrieving change points is also high time consuming, where all points in the trajectories need to be checked.

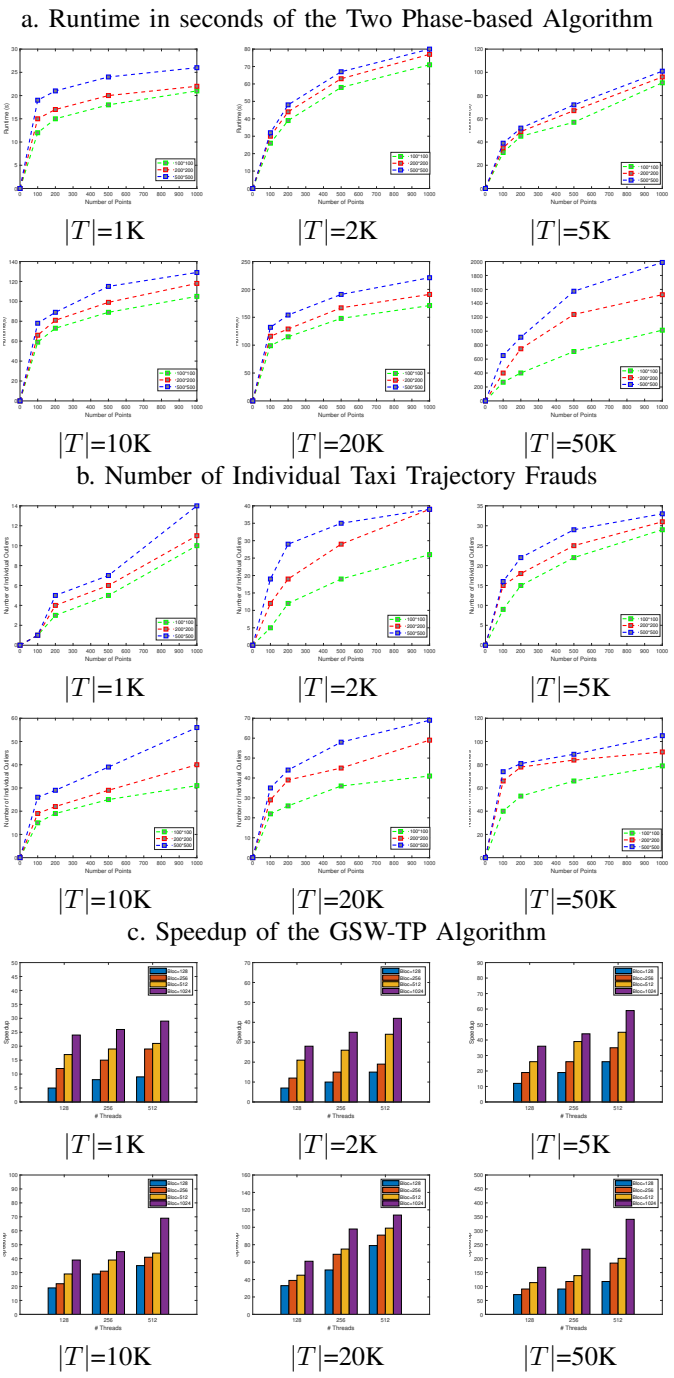


Fig. 4. Runtime in seconds, Number of Individual Taxi Trajectory Frauds of the Two Phase-based Algorithm, and Speedup of the GSW-TP Algorithm using Synthetic Trajectory Databases

Fig. 4. (b) presents the quality performance of the two phase-based algorithm using synthetic trajectory databases by varying the number of points from 100 to $1,000$, the number of trajectories from $1,000$ to $50,000$, and with different grid sizes ($100 * 100$, $200 * 200$, and $500 * 500$). The results reveal that by increasing the parameters values, the number of individual taxi trajectory frauds returned by the two phase-based algorithm increased. Thus, for 100 points, and $1,000$ trajectories, and with grid size $100 * 100$, the number of individual taxi trajectory frauds retrieved by the Two Phase-based algorithm is 15, however, for mining big trajectory databases, with $1,000$ points, $50,000$ trajectories, and $500*500$ as grid size, the number of individual taxi trajectory frauds returned by the two-phase based algorithm is 102. Moreover, the number of individual taxi frauds is very sensitive to the number of points in the grid. Dealing with large number of individual taxi frauds trajectories may reduce the runtime performance of the two phase-based algorithm in the second phase to derive the group taxi trajectory frauds. For instance, when we deal with $50,000$ trajectories, $1,000$ points, and grid of $500 * 500$, 102 individual taxi trajectory frauds are found. This could generate $2^{102} - 1$ potential nodes in the feature selection phase, which is very prohibitive as cost to derive the group taxi trajectory frauds. Consequently, in the next experiment, we show the effect of the GPU and the sliding window strategies to boost the runtime performance of the two phase-based algorithm.

Fig. 4. (c) presents the speedup of GSW-TP algorithm using synthetic trajectory databases by varying the number of trajectories from $1,000$ to $50,000$, the number of blocs from 128 to $1,024$, and the number of threads of each block from 128 to 512. The number of points and the grid size are set to $1,000$ and $500 * 500$, respectively. The results reveal that by increasing the parameters values, the speedup of the GSW-TP increased. Indeed, when the number of blocks and threads per block is 128, the GSW-TP algorithm is 25 times faster than the two phase-based algorithm, and this for dealing with $1,000$ trajectories. However, when $1,024$ blocks with 512 threads per block are used, the GSW-TP algorithm is more than 300 times faster than the two phase-based algorithm for dealing $50,000$ trajectories. These results are obtained thanks to the massively threaded GPU model and the efficient mapping of the trajectories among different GPU blocks. Moreover, with accurate management of the different GPU levels memories help in improving the performance of the GSW-TP.

The next experiment aims to show the effect of the sliding windows on the overall performance of the mining process. Several experiments have been carried out using synthetic trajectory databases containing $50,000$ but with different number of points 100, $1,000$, and $10,000$. The results are reported in Table I. By varying the sliding windows from 1 to $1,024$, the number of individual taxi trajectory frauds per block is highly reduced. For instance, when the number of sliding windows is set to 1 (which simulates the sequential version of the two phase-based algorithm), the maximum number of individual taxi trajectory frauds returned is 75, 181, and 614 for 100, $1,000$, and $10,000$ points respectively, however, when the number of sliding widows is set to $1,024$, the maximum number of individual taxi trajectory frauds returned is 2, 4, and 11 for 100, $1,000$, and $10,000$ points respectively. This allows to prune the search of the group trajectory outliers in the next phase. For instance, the two phase-based approach needs $2^{614} - 1$ nodes exploration to deal with $10,000$ points, where with $1,024$ sliding windows, only 2^{11} nodes exploration to deal with the same number of points, and all this to find the group taxi trajectory frauds.

The last experiment of this part is to show the quality of the returned taxi trajectory frauds of our algorithm with different

TABLE I
NUMBER OF INDIVIDUAL TAXI TRAJECTORY FRAUDS PER GPU BLOCK BY VARYING BOTH THE THE NUMBER OF SLIDING WINDOWS AND THE NUMBER OF POINTS

$\|SW\|$	Number of Points	Min. Indiv. Frauds	Avg. Indiv. Frauds	Max. Indiv. Frauds
1	100	57	63	75
	1,000	149	164	181
	10,000	369	451	614
2	100	32	35	39
	1,000	74	89	105
	10,000	191	221	376
4	100	19	25	33
	1,000	63	66	74
	10,000	150	201	336
8	100	16	21	25
	1,000	55	57	61
	10,000	121	182	302
16	100	12	15	17
	1,000	41	42	44
	10,000	101	115	119
32	100	10	12	13
	1,000	33	35	36
	10,000	85	89	93
64	100	9	10	11
	1,000	21	28	30
	10,000	61	63	64
128	100	7	8	9
	1,000	15	21	23
	10,000	50	51	52
256	100	5	5	5
	1,000	10	11	12
	10,000	31	33	36
512	100	3	4	5
	1,000	7	7	7
	10,000	21	23	24
1,024	100	1	2	2
	1,000	4	4	4
	10,000	7	9	11

TABLE II
QUALITY OF RETURNED TAXI FRAUDS BY VARYING BOTH THE THE NUMBER OF SLIDING WINDOWS AND THE NUMBER OF POINTS

$\|T\|$	Number of Points	Fmeasure	mAP
1K	100	0.74	0.76
	1,000	0.75	0.76
	10,000	0.83	0.79
10K	100	0.79	0.84
	1,000	0.82	0.85
	10,000	0.84	0.87
50K	100	0.86	0.88
	1,000	0.91	0.92
	10,000	0.94	0.95

trajectory database sizes. Several experiments have been carried out using synthetic trajectory databases containing $1,000$, $10,000$, and $50,000$ but with different number of points 100, $1,000$, and $10,000$. The results are reported in Table II. By increasing the number of trajectories from $1,000$ to $50,000$, the quality of the returned frauds, in terms of F-measure and mAP are highly increased, from 0.74 to 0.94 for F-measure value, and from 0.76 to 0.95 for mAP value. Again, while increasing the number of points from 100 to $10,000$, the quality of the returned frauds is also enhanced. These results are explained by the fact that our approach is very appropriate for sparse and big trajectory databases, as the case of real-world applications.

B. Real Taxi Trajectory Databases

The second experiment aims at demonstrating the usefulness of our framework in a real-world scenario of taxi trajectories. the taxi trajectory service of Porto[1] is used. The dataset contains real taxi trajectories retrieved from $01/07/2013$ to $30/06/2014$ of the 442 taxis running in the city of Porto, in Portugal. This allows to recuperate more than *3 GB* of data stored in one single CSV file. Each row contains information related to one trip including: TripID, CallType, TaxiID. The last component of the row contains a list of GPS coordinates. This list contains one pair of coordinates for each *15* seconds of trip. The last list item corresponds to the trip's destination while the first one represents its start. Useful information about this trajectory data could be found in [46].

Fig. 5(a) presents the number of taxi trajectory frauds with different number of trajectory sizes, and different density point threshold values. By increasing the number of trajectories from 1 to $7,000$, the number of individual taxi trajectory frauds increased. Indeed, the number of taxi trajectory frauds is 35 for 700 trajectories, and reaches 72 for $7,000$ trajectories. However, when the density point threshold increased from 0.1 to 1.0, the number of individual taxi trajectory frauds decreased. Indeed, the number of trajectory frauds is 72 for density set = 0.1, and decreases to 60 for density set = 1.0.

Fig. 5(b) contains the results of both individual and group taxi trajectory frauds by applying the two phase-based algorithm on the same trajectory database. From this figure, we remark that the two phase-based algorithm is able to detect group taxi trajectory frauds (marked by red color) against normal trajectories (marked by black color). These group taxi trajectory frauds represent different taxi trips. One of the reason that these taxis deviate from the normal trajectories is the high traffic jam of Porto city, in a peak hours, where the flow rate is too high.

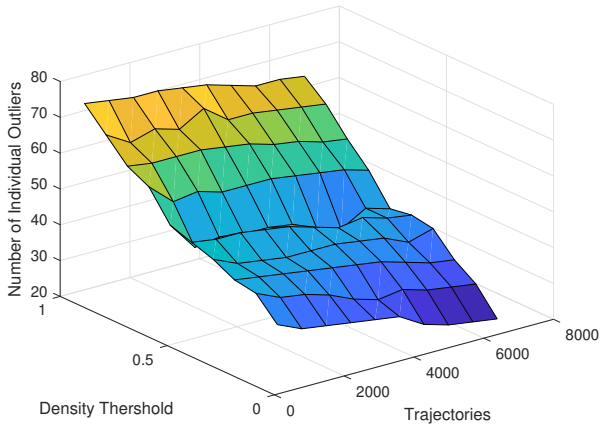
C. Comparison with the State-of-the-art Approaches

In this part, we compare our approaches with the existing baseline approaches using the real taxi trajectory database. We consider the baseline AGJFD [47] and FGM [48] algorithms for our serial approach, and SolvingSet+ [49] for our GPU-based version.

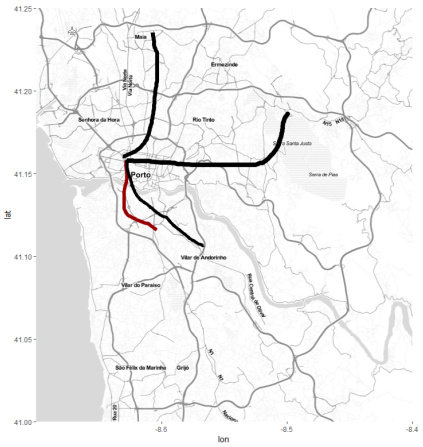
Serial Approach Table III presents the accuracy of the proposed serial approach and the baseline algorithms (AGJFD, FGM) in terms of F-measure value. By varying the percentage of trajectories used as input from 10% to 100%, our solution outperforms the baseline for almost all cases. This comes from the fact that our solution uses more accurate strategies by incorporating feature selection for identifying both individual and group taxi trajectory frauds. However, the baseline approaches employ statistical distribution to find the group taxi trajectory frauds. In real scenarios, it is hard to fit the whole trajectory data to the corresponding distributions.

To validate the previous results, we conduct the Z-test for Two Phase-based approach compared with AGJFD and FGM. The validation process can be modelled as follows:

[1] http://www.geolink.pt/ecmlpkdd2015-challenge/dataset.html



(a) Number Of Individual Outliers



(b) Map of Individual Outliers

Fig. 5. Case Study of Real Taxi Trajectory Database

TABLE III
QUALITY OF THE SERIAL APPROACH AND THE BASELINE GROUP DETECTION APPROACHES

% Trajectories	Two Phase	AGJFD	FGM
10	0.94	0.91	0.92
20	0.92	0.91	0.92
30	0.92	0.91	0.92
40	0.91	0.91	0.91
50	0.90	0.90	0.91
60	0.89	0.88	0.88
70	0.89	0.88	0.87
80	0.89	0.87	0.85
90	0.88	0.86	0.84
100	0.88	0.84	0.81

1) Each approach is viewed as a normal variable.
2) Each case among the ten cases (10% to 100%) represents an observation.
3) Each case's result is a sample.

Two estimators, $\widehat{E_1}$, and $\widehat{E_2}$ are used as follows,
$\widehat{E_1} = Mean(TwoPhase) - Mean(AGJFD)$
$\widehat{E_2} = Mean(\widehat{E_1}) - Mean(FGM)$
First, the normality of three approaches is checked using the Shapiro-Wilk test which is available on XLSTAT. Therefore, the first hypothesis, H_0, and the alternative hypothesis, H_a are defined as follows:

1) H_0: The approaches follow a Normal Distribution.
2) H_a: The approaches do not follow a Normal Distribution.

The used significance level α was set to 0.01. The results of the Shapiro-Wilk test indicate that the unilateral p-values for all approaches are greater than α. We conclude from this table that H_0 cannot be rejected. Hence, the approaches follow the normal distribution. In other words, the non-normality is not significant. Afterwards, we performed the Z-test, it is first used with $\alpha = 0.02\%$, to compare the three approaches. XLSTAT gives $\widehat{E_1} = 0.008$, and $\widehat{E_2} = 0.009$. These results confirm that the proposed approach statistically outperforms the two other approaches (AGJFD, and FGM) by Z-test.

TABLE IV
RUNTIME (SEC) OF THE GSW-TP APPROACH AND THE BASELINE SOLVINGSET+ APPROACH

% Trajectories	GSW-TP	SolvingSet+
10	157	284
20	233	335
30	259	448
40	274	512
50	291	597
60	334	614
70	394	678
80	475	710
90	574	814
100	681	928

GPU-based Approach Table IV shows the runtime in seconds of the GSW-TP with the baseline SolvingSet+ algorithm. By increasing the percentage of trajectory sizes from 10% to 100%, the GSW-TP outperforms SolvingSet+ in terms of processing time, whatever the case used as input. Thus, GSW-TP's runtime is 681 seconds to deal with the whole trajectory data, while SolvingSet+ needs 928 seconds for processing the same trajectory data. These promising results are obtained due to the efficient mapping between trajectories and the GPU threads/blocks.

Again, to validate the previous results, we conduct the Z-test for GSW-TP approach compared with SolvingSet+. The validation process can be modelled as follows:

1) Each approach is viewed as a normal variable.
2) Each case among the ten cases (10% to 100%) represents an observation.
3) Each case's result is a sample.

An estimator $\widehat{E_1} = Mean(GSW - TP) - Mean(SolvingSet+)$ is used. First, the normality of the two approaches is checked using the Shapiro-Wilk test which is available on XLSTAT. Therefore, the first hypothesis, H_0, and the alternative hypothesis, H_a are defined as follows:

1) H_0: The approaches follow a Normal Distribution.
2) H_a: The approaches do not follow a Normal Distribution.

The used significance level α was set to 0.015. The results of the Shapiro-Wilk test indicate that the unilateral p-values for all approaches are greater than α. We conclude from this table that H_0 cannot be rejected. Hence, the approaches follow the normal distribution. In other words, the non-normality is not significant. Afterwards, we performed the Z-test, it is first used with $\alpha = 0.025\%$, to compare the two approaches. XLSTAT gives $\widehat{E_1} = 0.014$. These results confirm that the GSW-TP approach statistically outperforms the SolvingSet+ by Z-test.

D. Discussions

In this section, we provide the insights on several open research issues regarding the application of the proposed algorithms in taxi trajectory frauds: Several directions could be investigated to improve the quality of the detected taxi trajectory frauds:

1) It may be by studying the different dependencies between the historical taxi trajectories. Incorporating pattern-mining approaches, and exploring the discovered patterns with the existing taxi trajectory frauds is a challenging problem and may improve the quality of the returned taxi frauds.

2) It could be by adapting more specific advanced methods such as spatial data, graph data, or time series and sequence data. All these special scenarios are somehow related to possible scenarios in tackling taxi trajectory databases.

3) Using high performance computing tools for handling taxi frauds in real time environments is a challenging issues. Several questions should be addressed. For example, which architectures should be used? how do we efficiently partition the data among the different jobs? how can we design a parallel approach respecting the high performance computing challenges such as reducing communication and synchronization cost or increasing load balancing and optimizing memory management?.

4) Solutions of taxi trajectory frauds could identify different anomalous patterns from the same taxi trajectory data. The problem is how to decide which patterns are useful for the city planners. To improve the usefulness of the detected patterns, a crowdsourcing approach may be applied, where different taxi trajectory frauds approaches should work together to identify the best anomalous patterns delivered to the city planners.

V. Conclusion

This paper introduced a new approach whose goal is to discover and extract individual and group taxi trajectory frauds, and also the change points. In order to solve such problem efficiently, two approaches have been introduced: i) The *Two Phase-based algorithm* explored two main phases, it first determines the individual taxi trajectory frauds by computing the distance of each point in each trajectory, and a second phase identifies the group taxi trajectory frauds by exploring the individual taxi trajectory frauds using both feature selection, and sliding windows strategy, and ii) The *GSW-TP* approach, which exploits the merits of the GPU architecture with the sliding windows strategy to improve the runtime of the phase-based algorithm. To demonstrate the usefulness and efficiency of the proposed framework, several experiments have been carried out on synthetic and real trajectory databases. Experimental results reveal the scalability of the parallel approach compared to the sequential version by reaching a speedup of up to 341 when dealing $50,000$ trajectory database. Moreover, the results reveal the usefulness of exploring different kinds of taxi trajectory frauds when dealing with real taxi trajectory database. The results also reveal the superiority of our approaches compared to the baseline methods in terms of both computational time and the quality of returned frauds.

References

[1] L. Zhu, F. R. Yu, Y. Wang, B. Ning, and T. Tang, "Big data analytics in intelligent transportation systems: A survey," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 1, pp. 383–398, 2019.

[2] I. Kalamaras, A. Zamichos, A. Salamanis, A. Drosou, D. D. Kehagias, G. Margaritis, S. Papadopoulos, and D. Tzovaras, "An interactive visual analytics platform for smart intelligent transportation systems management," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 2, pp. 487–496, 2018.

[3] W. Li, H. Song, and F. Zeng, "Policy-based secure and trustworthy sensing for internet of things in smart cities," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 716–723, 2018.

[4] N. Cao, C. Lin, Q. Zhu, Y.-R. Lin, X. Teng, and X. Wen, "Voila: Visual anomaly detection and monitoring with streaming spatiotemporal data," *IEEE Transactions on Visualization and Computer Graphics*, vol. 24, no. 1, pp. 23–33, 2018.

[5] G. Ajaeiya, I. H. Elhajj, A. Chehab, A. Kayssi, and M. Kneppers, "Mobile apps identification based on network flows," *Knowledge and Information Systems*, vol. 55, no. 3, pp. 1–26, 2018.

[6] Y. Djenouri and A. Zimek, "Outlier detection in urban traffic data," in *8th International Conference on Web Intelligence, Mining and Semantics*. ACM, 2018, pp. 3:1–3:12.

[7] H. Senaratne, M. Mueller, M. Behrisch, F. Lalanne, J. Bustos-Jiménez, J. Schneidewind, D. Keim, and T. Schreck, "Urban mobility analysis with mobile network data: A visual analytics approach," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 5, pp. 1537–1546, 2018.

[8] T. Kieu, B. Yang, and C. S. Jensen, "Outlier detection for multi-dimensional time series using deep neural networks," in *19th IEEE International Conference on Mobile Data Management (MDM)*. IEEE, 2018, pp. 125–134.

[9] M. Ghahramani, M. Zhou, and C. T. Hon, "Mobile phone data analysis: A spatial exploration toward hotspot detection," *IEEE Transactions on Automation Science and Engineering*, vol. 16, no. 1, pp. 351–362, 2018.

[10] K. Lu, J. Liu, X. Zhou, and B. Han, "A review of big data applications in urban transit systems," *IEEE Transactions on Intelligent Transportation Systems*, 2020.

[11] G. Atluri, A. Karpatne, and V. Kumar, "Spatio-temporal data mining: A survey of problems and methods," *ACM Computing Surveys (CSUR)*, vol. 51, no. 4, pp. 83:1–83:4, 2018.

[12] M. Das and S. Parthasarathy, "Anomaly detection and spatio-temporal analysis of global climate system," in *3rd International Workshop on Knowledge Discovery from Sensor Data*. ACM, 2009, pp. 142–150.

[13] M. Gupta, J. Gao, C. C. Aggarwal, and J. Han, "Outlier detection for temporal data: A survey," *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 9, pp. 2250–2267, 2014.

[14] R. Conforti, M. La Rosa, and A. H. ter Hofstede, "Filtering out infrequent behavior from business process event logs," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 2, pp. 300–314, 2017.

[15] M. Salehi, C. Leckie, J. C. Bezdek, T. Vaithianathan, and X. Zhang, "Fast memory efficient local outlier detection in data streams," *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 12, pp. 3246–3260, 2016.

[16] Y. Djenouri, A. Zimek, and M. Chiarandini, "Outlier detection in urban traffic flow distributions," in *IEEE International Conference on Data Mining (ICDM)*. IEEE, 2018, pp. 935–940.

[17] Y. Djenouri, A. Belhadi, J. C.-W. Lin, and A. Cano, "Adapted k-nearest neighbors for detecting anomalies on spatio–temporal traffic flow," *IEEE Access*, vol. 7, pp. 10 015–10 027, 2019.
[18] K. Thiyagarajan, S. Kodagoda, R. Ranasinghe, D. Vitanage, and G. Iori, "Robust sensor suite combined with predictive analytics enabled anomaly detection model for smart monitoring of concrete sewer pipe surface moisture conditions," *IEEE Sensors Journal*, 2020.
[19] Z. Zhu, D. Yao, J. Huang, H. Li, and J. Bi, "Sub-trajectory-and trajectory-neighbor-based outlier detection over trajectory streams," in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 2018, pp. 551–563.
[20] J. Mao, P. Sun, C. Jin, and A. Zhou, "Outlier detection over distributed trajectory streams," in *SIAM International Conference on Data Mining*. SIAM, 2018, pp. 64–72.
[21] A. H. Milaghardan, R. A. Abbaspour, and C. Claramunt, "A dempster-shafer based approach to the detection of trajectory stop points," *Computers, Environment and Urban Systems*, vol. 70, pp. 189–196, 2018.
[22] Y. Yu, L. Cao, E. A. Rundensteiner, and Q. Wang, "Detecting moving object outliers in massive-scale trajectory streams," in *20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2014, pp. 422–431.
[23] C.-L. Yu, Yanwei, E. A. Rundensteiner, and Q. Wang, "Outlier detection over massive-scale trajectory streams," *ACM Transactions on Database Systems (TODS)*, vol. 42, no. 2, pp. 10:1–10–33, 2017.
[24] J.-G. Lee, J. Han, and X. Li, "Trajectory outlier detection: A partition-and-detect framework," in *IEEE International Conference on Data Engineering*. IEEE, 2008, pp. 140–149.
[25] Y. Ge, H. Xiong, Z.-h. Zhou, H. Ozdemir, J. Yu, and K. C. Lee, "Top-eye: Top-k evolving trajectory outlier detection," in *19th ACM International Conference on Information and Knowledge Management*. ACM, 2010, pp. 1733–1736.
[26] Z. Liu, D. Pi, and J. Jiang, "Density-based trajectory outlier detection algorithm," *Journal of Systems Engineering and Electronics*, vol. 24, no. 2, pp. 335–340, 2013.
[27] Y. Zheng, "Trajectory data mining: an overview," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 6, no. 3, pp. 29:1–29:41, 2015.
[28] G. Qin, Z. Huang, Y. Xiang, and J. Sun, "Probdetect: A choice probability-based taxi trip anomaly detection model considering traffic variability," *Transportation Research Part C: Emerging Technologies*, vol. 98, pp. 221–238, 2019.
[29] A. M. Kamoona, A. K. Gostar, R. Tennakoon, A. Bab-Hadiashar, D. Accadia, J. Thorpe, and R. Hoseinnezhad, "Random finite set-based anomaly detection for safety monitoring in construction sites," *IEEE Access*, vol. 7, pp. 105 710–105 720, 2019.
[30] D. Zhang, N. Li, Z.-H. Zhou, C. Chen, L. Sun, and S. Li, "iBAT: detecting anomalous taxi trajectories from GPS traces," in *13th International Conference on Ubiquitous Computing*. ACM, 2011, pp. 99–108.
[31] C. Chen, D. Zhang, P. S. Castro, N. Li, L. Sun, S. Li, and Z. Wang, "iboat: Isolation-based online anomalous trajectory detection," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 2, pp. 806–818, 2013.
[32] Y. Ge, H. Xiong, C. Liu, and Z.-H. Zhou, "A taxi driving fraud detection system," in *2011 IEEE 11th International Conference on Data Mining*. IEEE, 2011, pp. 181–190.
[33] Y. Ge, C. Liu, H. Xiong, and J. Chen, "A taxi business intelligence system," in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2011, pp. 735–738.
[34] Y. Yuan, K. Miao, D. Zhang, L. Sun, and C. Chen, "An osgi-based smart taxi service platform," in *2012 IEEE Asia-Pacific Services Computing Conference*. IEEE, 2012, pp. 173–178.
[35] B. Leng, H. Du, J. Wang, L. Li, and Z. Xiong, "Analysis of taxi drivers' behaviors within a battle between two taxi apps," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 1, pp. 296–300, 2015.
[36] X. Zhang, Z. Zhao, Y. Zheng, and J. Li, "Prediction of taxi destinations using a novel data embedding method and ensemble learning," *IEEE Transactions on Intelligent Transportation Systems*, 2019.
[37] J. Wang, Y. Yuan, T. Ni, Y. Ma, M. Liu, G. Xu, and W. Shen, "Anomalous trajectory detection and classification based on difference and intersection set distance," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 3, pp. 2487–2500, 2020.
[38] X. Wang, A. Fagette, P. Sartelet, and L. Sun, "A probabilistic tensor factorization approach to detect anomalies in spatiotemporal traffic activities," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. IEEE, 2019, pp. 1658–1663.
[39] D. Smolyak, K. Gray, S. Badirli, and G. Mohler, "Coupled igmm-gans with applications to anomaly detection in human mobility data," *ACM Transactions on Spatial Algorithms and Systems (TSAS)*, vol. 6, no. 4, pp. 1–14, 2020.
[40] Y. Djenouri, A. Belhadi, J. C.-W. Lin, D. Djenouri, and A. Cano, "A survey on urban traffic anomalies detection algorithms," *IEEE Access*, vol. 7, pp. 12 192–12 205, 2019.
[41] A. Cano, "A survey on graphic processing unit computing for large-scale data mining," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 8, no. 1, p. e1232, 2018.
[42] Y. Djenouri, A. Belhadi, P. Fournier-Viger, and H. Fujita, "Mining diversified association rules in big datasets: A cluster/GPU/genetic approach," *Information Sciences*, vol. 459, pp. 117–134, 2018.
[43] Y. Djenouri, D. Djenouri, A. Belhadi, and A. Cano, "Exploiting GPU and cluster parallelism in single scan frequent itemset mining," *Information Sciences*, vol. 496, pp. 363–377, 2018.
[44] V. Roberge, M. Tarbouchi, and G. Labonté, "Fast Genetic Algorithm Path Planner for Fixed-Wing Military UAV Using GPU," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 54, no. 5, pp. 2105–2117, 2018.
[45] G. Zhou, R. Bo, L. Chien, X. Zhang, S. Yang, and D. Su, "GPU-accelerated algorithm for online probabilistic power flow," *IEEE Transactions on Power Systems*, vol. 33, no. 1, pp. 1132–1135, 2018.
[46] L. Moreira-Matias, J. Gama, M. Ferreira, J. Mendes-Moreira, and L. Damas, "Predicting taxi–passenger demand using streaming data," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 3, pp. 1393–1402, 2013.
[47] C. Sun, Z. Yan, Q. Li, Y. Zheng, X. Lu, and L. Cui, "Abnormal group-based joint medical fraud detection," *IEEE Access*, vol. 7, pp. 13 589–13 596, 2019.
[48] L. Xiong, B. Póczos, and J. G. Schneider, "Group anomaly detection using flexible genre models," in *Advances in neural information processing systems*, 2011, pp. 1071–1079.
[49] F. Angiulli, S. Basta, S. Lodi, and C. Sartori, "GPU strategies for distance-based outlier detection," *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 11, pp. 3256–3268, 2016.

Asma Belhadi obtained the PhD in Computer Engineering from the University of Science and Technology USTHB Algiers, Algeria, in 2016. She is working on topics related to artificial intelligence and data mining, with focus on logic programming. Dr. Belhadi participated in many international conferences worldwide, and she has been granted short-term research visitor internships to many renowned universities including IRIT in Toulouse. She has published over 10 refereed research articles in the areas of artificial intelligence.

Youcef Djenouri obtained the PhD in Computer Engineering from the University of Science and Technology USTHB Algiers, Algeria, in 2014. In 2017, he was postdoctoral research at Southern Denmark University, where he has working on urban traffic data analysis. Currently, he is granted a post-doctoral fellowship from the European Research Consortiu on Informatics and Mathematics (ERCIM), and he worked at the Norwegian University of Science and Technology (NTNU), in Trondheim, Norway. He is working on topics related to artificial intelligence and data mining, with focus on association rules mining, frequent itemsets mining, parallel computing, swarm and evolutionary algorithms and pruning association rules. Dr. Djenouri has published more than 60 refereed research papers, in the areas of data mining, parallel computing and artificial intelligence. Current information can be found at https://sites.google.com/site/youcefdjenouri.

Gautam Srivastava was awarded his B.Sc. degree from Briar Cliff University in the U.S.A. in the year 2004, followed by his M.Sc. and Ph.D. degrees from the University of Victoria in Victoria, British Columbia, Canada in the years 2006 and 2012, respectively. Dr. G, as he is popularly known, is active in research in the field of Cryptography, Data Mining, Security and Privacy, and Blockchain Technology. In his 5 years as a research academic, he has published a total of 45 papers in high-impact conferences in many countries and in high-status journals (SCI, SCIE). He is an IEEE Senior Member.

Djamel Djenouri obtained the PhD in Computer Science from the University of Science and Technology (USTHB), Algiers, Algeria, in 2007. From 2008 to 2009, he was granted a post-doctoral fellowship from the European Research Consortium on Informatics and Mathematics (ERCIM), and he worked at the Norwegian University of Science and Technology (NTNU), Norway. He is currently an Associate Professor at the University of the West England in Bristol. He is working on topics related Internet of things, wireless and mobile networks, machine learning and application for smart cities and green applications. He has been conducting several research projects with international collaborations as the principal investor for many of them. He participated in many international conferences worldwide and gave many keynotes and plenary-session talks. He published more than 100 papers in international peer-reviewed journals and conference proceedings, two books, and he is holding two national patents.

Alberto Cano is an Assistant Professor with the Department of Computer Science, Virginia Commonwealth University, USA, where he heads the High-Performance Data Mining Lab. He obtained his BSc degrees in Computer Engineering and in Computer Science from the University of Cordoba, Spain, in 2008 and 2010, respectively, and his MSc and PhD degrees in Intelligent Systems and Computer Science from the University of Granada, Spain, in 2011 and 2014, respectively. His research is focused on machine learning, data mining, general-purpose computing on graphics processing units, Apache Spark, and evolutionary computation. He has published over 45 articles in high-impact factor journals, 50 contributions to international conferences, two book chapters, and one book in the areas of machine learning, data mining, and parallel, distributed, and GPU computing. His research is supported by an Amazon AWS Machine Learning award (2018) and the VCU Presidential Research Quest Fund (2018). Dr. Cano is Associate Editor of IEEE Access and Applied Intelligence. Dr. Cano is IEEE Senior Member.

Jerry Chun-Wei Lin received his Ph.D. in Computer Science and Information Engineering from National Cheng Kung University, Tainan, Taiwan, in 2010. He is now working as an associate professor at Department of Computing, Mathematics, and Physics, Western Norway University of Applied Sciences (HVL), Bergen, Norway. His research interests include data mining, privacy-preserving and security, Big Data analytics, and social networks. He has published more than 200 research papers in peer-reviewed international conferences and journals, which have received more than 1900 citations. He is the co-leader of the popular SPMF open-source data-mining library and the Editor-in-Chief of *Data Mining and Pattern Recognition* (DSPR), Associate Editor of *Journal of Internet Technology* and *IEEE Access*, and a member of the Editorial Board of *Intelligent Data Analysis*.