L. Leonardi, S. Orlando, A. Raffaetà, A. Roncato, C. Silvestri

Frequent Spatio-Temporal Patterns in Trajectory Data Warehouses
Frequent Spatio-Temporal Patterns in Trajectory Data Warehouses

L. Leonardi, S. Orlando, A. Raffaetà, A. Roncato, C. Silvestri

Dipartimento di Informatica Università Ca’ Foscari Venezia, Italy

Abstract. In this paper we present an approach for storing and aggregating spatio-temporal patterns by using a Trajectory Data Warehouse (TDW). In particular, our aim is to allow the analysts to quickly evaluate frequent patterns mined from trajectories of moving objects occurring in a specific spatial zone and during a given temporal interval. We resort to a TDW, based on a data cube model, having spatial and temporal dimensions, discretized according to a hierarchy of regular grids, and whose facts are sets of trajectories which intersect the spatio-temporal cells of the cube. The idea is to enrich such a TDW with a new measure: frequent patterns obtained from a data-mining process on trajectories. As a consequence these patterns can be analysed by the user at various levels of granularity by means of OLAP queries. The research issues discussed in this paper are (1) the extraction/mining of the patterns to be stored in each cell, which requires an adequate projection phase of trajectories before mining; (2) the spatio-temporal aggregation of patterns to answer roll-up queries, which poses many problems due to the holistic nature of the aggregation function.

1 Introduction

Nowadays an increasing number of technologies, ranging from location based services to location aware devices, produces temporally annotated data about positions and movements of many kinds of objects. Such data can be continuously logged at a high level of detail for service purposes. However, due to their huge volume or for privacy reasons, raw data concerning the trajectories of a multitude of moving objects cannot be stored and maintained for a long time. Nevertheless, analysts may be interested in the knowledge hidden in such data. Our proposal is thus to compute and store particular spatio-temporal (ST) aggregates concerning trajectories, and use Data Warehouse technologies to manage them and answer analytical multidimensional queries. To this aim, we extend the Trajectory Data Warehouses (TDWs) defined in [10,8], which are based on spatio-temporal (ST) multidimensional cube models. The facts of interest are the trajectories of moving objects, to be analysed by means of many associated properties, like the measures average speed, traveled distance, maximum acceleration, presence. In this paper we propose to go beyond numerical measures about trajectory data. We are interested in aggregate properties obtained through a knowledge discovery process. In particular we focus on the results extracted by
a ST Frequent Pattern Mining tool. Apart from the transformation phase of ST raw data, the above problem can be reduced to the well-known Frequent Itemset Mining (FIM) [1]. The extraction of frequent patterns is a time consuming task, and the original raw data may no longer be available when the analyst poses a query about the frequent patterns occurring in a certain spatial area during a given time interval. Therefore, as soon as data arrive, we transform data, extract patterns, and load the base cells of our TDW with the mined patterns. Finally, such patterns are aggregated in order to answer roll-up queries about patterns occurring on larger ST cells.

The need of extracting useful and actionable knowledge to store in the TDW requires to introduce the concept of spatial regions of interest (ROIs): this means that the patterns stored in the TDW do no longer refer to spatial coordinates, i.e., those used in the trajectory sampling, but to the identifiers associated with ROIs. This implies that the trajectory properties we store and retrieve from the TDW are frequent sets of ROIs, which are visited, in any order, by a large number of trajectories (beyond a certain threshold). This knowledge can be exploited in many ways. For example, a tourist, in order to plan a visit, can look at the common sets of regions/sightseeing visited by a multitude of people, or s/he can derive an association rule that, given a set of regions already visited, states which are the most visited other regions. Moreover, these patterns can be useful to an analyst interested in understanding social behaviour or to a public manager to make decisions concerning the traffic circulation. The main research challenges, discussed in the following, are thus the Extraction, Transformation & Loading (ETL) of the TDW, and the ST aggregation functions used to answer roll-up queries. In particular, some approximations are introduced in aggregate computation, in order to give a feasible solution to the problem posed by the holistic nature of the pattern aggregation functions.

It is worth noticing that the ETL phase implies a lossy transformation of raw trajectory data, so that it is impossible to return to the original input data starting from the measures stored in the base cells of our TDW. For example, we store the average speed of a set of trajectories traversing a cell, not the speed of a single trajectory. Likewise, from the TDW measures we can derive the sets of ROIs traversed by many trajectories but not the crossing order of such ROIs.

The problem of generating global patterns from sub-aggregates, such as the ones stored in our base cells, has some affinity with the Frequent Pattern Mining problem in a distributed [6] or stream [12, 3] settings. In both cases, even if for different reasons, the database is partitioned and each partition/chunk is mined separately. The referred items are the same in each partition. The models extracted, though locally coherent and accurate, pose several problems and complexities when combined/aggregated to infer a global model. The main issue is the correct computation of the support $f(p)$ of a pattern $p$, since it may be infrequent in a partition/chunk, and be frequent in another. In our case, since patterns refer to trajectories that can span several spatial or temporal partitions, this poses additional and peculiar challenges to the task of aggregating local models/patterns.
The maintenance of a DW of frequent itemsets has been already presented in [9]. Nonetheless, in our case data and patterns refer to trajectories instead of generic categorical data. Moreover, our TDW has both spatial and temporal dimensions, which require adequate aggregation functions. Finally, in [5] a non-traditional DW model is presented, designed and tailored for logistic applications, where RFID technologies are used for goods tracing. Unlike our TDW, however, this DW has to preserve object transition information, while allowing multi-dimensional analysis of path-dependent aggregates.

The rest of the paper is structured as follows. Section 2 introduces and formalises the ST data collection that is the subject of our analysis. In Section 3 we discuss the transformation and the restriction process of our trajectory dataset to the base cells of our TDW. Section 4 states which kinds of frequent patterns we plan to compute. In Section 5 we describe the ETL process of our TWD, and the issues and the algorithm used to aggregate measures and answer roll-up queries along either the spatial and/or temporal dimensions. Section 6 discusses the results of several experiments conducted by using both synthetic and real-world datasets. Finally, Section 7 draws some conclusions.

2 Trajectories

In real-world applications the movement of an object, i.e., its trajectory, is often sampled. This produces a finite set of ST observations, taken from the actual continuous trajectory. It is reasonable to expect that observations are taken at irregular rates for each object, and that there is no temporal alignment between the observations of different objects.

Let $T$ be a set of 2D trajectories, $T = \{T_{i}\}_{i \in \{1,\ldots,n\}}$. In particular, $T_{i} = (I_{i}, L_{i})$ is the sampling of the trajectory, identified by $I_{i}$, of a moving object, and $L_{i} = \langle L_{1i}, L_{2i}, \ldots, L_{Mi} \rangle$ is a sequence of observations. Each observation $L_{i} = (t_{j}, \nu_{j})$ represents the presence of an object in the geographic location $(\nu_{j})$ at time $t_{j}$. The observations are temporally ordered, i.e., $t_{j} < t_{j+1}$.

Trajectory Reconstruction. In many situations, e.g., when one is interested in computing the cumulative number of trajectories in a given area, an (approximate) reconstruction of each trajectory from its sampling is needed.

Among the several possible solutions, in this paper we will use linear local interpolation, i.e., objects are assumed to move straight between two observed points with constant speed. This interpolation seems to be a quite standard approach to solve the problem, and yields a good trade-off between flexibility and simplicity. In particular, it is a reasonable choice if we do not have any information about the context in which the objects are moving, for example the underlying road network.

Regions of Interest. In order to extract meaningful patterns from trajectories, a preprocessing phase is required: given a set of ROIs, the trajectories are transformed from sequences of point-based observations into sequences of region-based observations. This data transformation makes possible to mine more valuable
patterns, i.e., patterns that turn out to occur frequently in the input trajectories, since their spatial references are looser than the original observations.

Let $\mathcal{R} = \{r_1, \ldots, r_k\}$ be the set of all spatial ROIs occurring in our spatial domain. Such regions can be indicated by users, or extracted by a tool from the trajectories themselves on the basis of the popularity of some areas, e.g., a clustering tool could discover that a given region turns out to be of interest since it corresponds to a very dense area in which many trajectories observations have been recorded.

Let us describe the transformation from a point-based trajectory to a region-based trajectory. First we notice that a trajectory can traverse a ROI $r_i$, even if no observation falls into $r_i$. For instance, consider Figure 1(a): the moving object will likely cross region $r_4$, but no observation is present in $r_4$. By reconstructing the trajectory by local linear interpolation, we build a region-based observation $\langle t, r_i \rangle$, where $t$ is the (interpolated) time at which the trajectory presumably entered the region $r_i$ (see Figure 1(b)). Once all the interpolated region-based observations have been added, the original observations can be transformed as follows. Given a trajectory observation $\langle t, x, y \rangle$, if point $(x, y)$ falls into a region $r_i \in \mathcal{R}$, it is transformed into $\langle t', r_i \rangle$ (see below for the choice of the timestamp), otherwise it is removed from the trajectory.

In order to choose the timestamp $t'$ of the region-based observation $\langle t', r_i \rangle$, we use the following criteria [4]:

- if the starting observation of a given trajectory is $\langle t, x, y \rangle$, and point $(x, y)$ falls into region $r_i$, the transformation simply yields the region-based observation $\langle t, r_i \rangle$;
- in all other cases, take the (interpolated) entering times $t'$ of the trajectory in each ROI, and associate it with the region identifier.

If consecutive trajectory observations $o_1, \ldots, o_j$ fall into the same region $r_i$, this criteria can generate duplicates. In this case, in the transformed trajectory we will keep only the region-based observation corresponding to $o_1$.

**Example 1.** Figure 1(a) shows a trajectory crossing different regions, and Figure 1(b) illustrates the transformed region-based trajectory. Notice that the point-based observations timestamped with $t_2$, $t_5$, and $t_6$ have been removed. In addition, the two observations with timestamps $t_3$ and $t_4$ have been collapsed into a single region-based observation. The timestamps associated with the region-based observations are $t'$ and $t''$ for regions $r_2$ and $r_4$, corresponding to the entering time of the trajectory into these regions, while the timestamp associated with $r_1$ is the original starting time of the trajectory.

We will use the term observation to denote a region-based observation, and trajectory to refer to a sequence of region-based observations:

$$\text{Tr} = \{ID, \langle (t_1, r_1), \ldots , (t_n, r_n) \rangle \}$$

where $ID$ is the identifier of the trajectory, $t_1 \leq \ldots \leq t_n$, and $\{r_1, \ldots, r_n\} \subseteq \mathcal{R}$ are ROIs.
3 Spatio-Temporal Restriction of Trajectories

Our TDW is based on a cube model composed of spatio-temporal cells. In particular, by subdividing the spatial dimension(s), we obtain $s$ geographical zones $\{Z_1, \ldots, Z_s\}$, which can be observed along a discretized temporal dimension, composed of $l$ consecutive intervals $\{I_1, \ldots, I_l\}$. In conclusion, we have $s \times l$ base cells $C^j_i = (Z_i, I_j)$, which we aim to populate with interesting patterns concerning the trajectories crossing each cell.

A measure associated with a cell models a property of the set of trajectories intersecting such a cell. Since each trajectory usually spans over different cells, in order to correctly compute the measures for the base cells, we define adequate operations that restrict our trajectories to these cells.

Without lack of generality, we can assume that, for each ROI $r \in \mathcal{R}$, there exists only one spatial zone $Z$ that covers $r$, denoted by $Z \triangleright r$. Let us now define the restrictions of a trajectory to a temporal interval and to a spatial zone.

**Definition 1.** (Restriction to a temporal interval) Let $\mathcal{Tr} = \{ID, \langle (t_1, r_1), \ldots, (t_n, r_n) \rangle\}$ be a trajectory and let $I$ be a temporal interval. The trajectory $\mathcal{Tr}$ restricted to $I$, denoted by $\mathcal{Tr}|_I$, is the maximal sequence of observations in $\mathcal{Tr}$:

$$\mathcal{Tr}|_I = \{ID, \langle (t_h, r_h), (t_{h+1}, r_{h+1}), \ldots, (t_{h+m}, r_{h+m}) \rangle\},$$

such that $[t_h, t_{h+m}] \subseteq I$.

Notice that $\mathcal{Tr}|_I$ is a subsequence of consecutive observations of the trajectory $\mathcal{Tr}$.

---

1 We can eventually split the region into sub-regions in order to satisfy such a constraint.
Definition 2. (Restriction to a spatial zone) Let $Tr = \{ID, \langle (t_1, r_1), \ldots, (t_n, r_n) \rangle \}$ be a trajectory and let $Z$ be a spatial zone. The trajectory restricted to $Z$, denoted by $Tr|_Z$, is the maximal sequence of observations in $Tr$:

$$Tr|_Z = \{ID, \langle (t_{i_1}, r_{i_1}), \ldots, (t_{i_m}, r_{i_m}) \rangle \},$$

such that $1 \leq i_1 \leq \ldots \leq i_m \leq n$ and $Z \owns r_{i_j}$ for all $j = 1, \ldots, m$.

Remark that the restriction to a spatial zone $Z$ can consist of several trajectory subsequences of $Tr$. This is due to the fact that an object can exit from $Z$, visit other zones, and then go back to $Z$, thus entering the zone several times.

Definition 3. (ST Restriction to base cells) Let $Tr = \{ID, \langle (t_1, r_1), \ldots, (t_n, r_n) \rangle \}$ be a trajectory, let $Z$ be a spatial zone and let $I$ be a temporal interval. The restriction of $Tr$ to the base cell $(Z, I)$ is obtained by restricting the trajectory to $Z$, and to $I$ in any order. We denote the result as: $Tr|_{Z,I}$.

Observe that the order is not relevant, i.e., $(Tr|_Z)|_I = (Tr|_I)|_Z$ by definition of temporal and spatial restriction. Thus the definition above is well given.

Finally, we say that a trajectory $Tr = \{ID, \langle (t_1, r_1), \ldots, (t_n, r_n) \rangle \}$ traverses a set of ROIs $R$ if $R \subseteq \bigcup_{i=1}^{n} \{r_i\}$.

Example 2. Figure 2 illustrates two trajectories $ID_0$ and $ID_1$. The ROIs are \{r_1, \ldots, r_4\} and $ID_0$ traverses the ROIs $r_1$, $r_2$ and $r_3$ while $ID_1$ traverses $r_1$ and $r_2$.

We consider two spatial zones $Z_1$ and $Z_2$ and two time intervals $I_1$ and $I_2$, i.e., four base cells: $(Z_1, I_1)$, $(Z_1, I_2)$, $(Z_2, I_1)$, $(Z_2, I_2)$, where $Z_1 \owns r_1, r_2$ and $Z_2 \owns r_3, r_4$. Concerning the timestamps, $t_1$, $t_2$, and $t_3$ range over the time interval $I_1$, whereas $t_4$ and $t_5$ over $I_2$. The ST restriction of $ID_0$ and $ID_1$ to the TDW base cells partitions $ID_0$ and $ID_1$ into the sub-trajectories shown in Table 1.

4 Mining Frequent Sets of ROIs

In this section we will introduce the kind of patterns, modelling significant behaviour, we want to mine through a knowledge discovery process starting from...
the trajectories intersecting each base cell of our TDW. In order to extract patterns we plan to exploit a Frequent Itemset Mining (FIM) tool. The FIM problem [1], introduced in the context of Market Basket Analysis (MBA), deals with the extraction of all the frequent itemsets from a database \( D \) of transactions. Given a set of items \( I = \{a_1, ..., a_M\} \), each transaction \( t \in D \), associated with a transaction identifier \( TID \), contains a subset of items in \( I \). Let \( i \subseteq I \) be a \( k \)-itemset, and \( f(i) \) denote its support, defined as \( f(i) = |\{t \in D \mid i \subseteq t\}| \). Mining all the frequent itemsets from \( D \) requires to discover all the \( k \)-itemsets \( i \) (for \( k = 1, 2, ..., \)) having a support greater than or equal to \( \sigma \cdot |D| \), i.e., \( f(i) \geq \sigma \cdot |D| \) where \( 0 \leq \sigma \leq 1 \) is a fixed threshold stating which is the minimal fraction of the transactions in \( D \) that must support a given itemset \( i \).

In the context of trajectories, interesting itemsets to be extracted can be frequent sets of ROIs (FSROIs), i.e., collections of ROIs which are visited together, even if in different orders, by a large number of trajectories.

**Definition 4.** (Frequent Sets of ROIs) Let \( p = \{r_1, ..., r_n\} \) be a pattern, i.e., a subset of ROIs \( p \subseteq R \) and let \( \sigma \), where \( 0 \leq \sigma \leq 1 \), be the minimum support threshold. Let \( T_p \subseteq T \) be the set of all trajectories \( Tr \in T \) which traverse \( p \) and let \( s(p) \) be the set of trajectory identifiers in \( T_p \) (tidSets).

Then the frequency of \( p \) is \( f(p) = |T_p| \) (or, equivalently, the cardinality of \( s(p) \)). We say that the pattern \( p \) is frequent w.r.t. \( \sigma \) iff \( f(p) \geq \sigma \cdot |T| \).

In general, in order to mine the FSROIs starting from the raw trajectories, several steps are needed:

1. Restrict trajectories to the spatio-temporal range requested by the user.
2. Select the ROIs contained in the spatial range.
3. Intersect the restricted trajectories with the selected ROIs, in order to obtain for each trajectory the set of ROIs the trajectory traverses. In this way a dataset of transactions \( (ID, SetOfROIs) \) is built.
4. Apply a FIM tool to the obtained transactions with a given \( \sigma \).

We can implement this process by using a Moving Object Database (MOD) where we store the raw point-based trajectories. MODs provide a great flexibility, since we can ask the system any spatio-temporal query, and we can easily modify the ROIs. However, some preliminary tests with Hermes [11], a MOD implementation running on Oracle, reveal that the execution time of the steps 1-3 is huge. Our TDW solution is much more efficient than the MOD approach

<table>
<thead>
<tr>
<th>Base cells</th>
<th>Restricted Trajectories</th>
</tr>
</thead>
<tbody>
<tr>
<td>((Z_1, I_1))</td>
<td>{(ID_0, ((t_1, r_1), (t_3, r_2)))} {(ID_1, ((t_2, r_1)))}</td>
</tr>
<tr>
<td>((Z_1, I_2))</td>
<td>{(ID_1, ((t_5, r_2)))}</td>
</tr>
<tr>
<td>((Z_2, I_2))</td>
<td>{(ID_0, ((t_4, r_3)))}</td>
</tr>
</tbody>
</table>

Table 1. Restricted trajectories in base cells.
in answering such spatio-temporal queries, even if we trade flexibility for performance. In fact, in the TDW some pre-computations are carried out during the ETL phase, and important decisions, concerning the allowed granularities of range queries and the ROIs to be taken into account during the mining, are made at TDW design time. In particular, the spatio-temporal grid and the associated hierarchy constrain the possible range queries, and, established the set of ROIs, we can perform the following transformations in order to prepare data for the mining algorithm:

1. Transform the restricted point-based trajectories into region-based ones, by linearly interpolating the trajectories. Usually this transformation strongly reduces the original size of data: we only maintain a single observation per each crossed ROI, and the number of ROIs is usually small, even if their definition is application-dependent.
2. Transform region-based trajectories into transactions, by removing timestamps associated with visited ROIs in the region-based observations. Note that while in a trajectory a given ROI can be visited multiple times, in the corresponding transaction this ROI can appear only once.

These transformations will be described in the next section.

5 TDW: Storing and Aggregating Patterns

In this section we discuss the ETL process and the aggregate function for computing FSROIs.

Unfortunately, the function to aggregate FSROIs is holistic, i.e., the super-aggregate cannot be computed from sub-aggregates, not even using any finite number of auxiliary measures. A common solution consists in computing holistic functions in an approximate way. Thus our approach can introduce some approximations during aggregation.

5.1 ETL

In Section 2 and 3 we already discussed the first part of the ETL process: how to transform raw trajectories into region-based ones, and how to restrict them to the various base cells \( C_i \). This process can be done on-the-fly, during the reception of the stream of observations, when all the points recorded in a given temporal interval \( I_j \) have been received.

The ETL process hence performs the restriction of the set of trajectories \( T = \{ T_{Rh} \}_{h \in \{1,...,n\}} \) to each \( C_i \), thus obtaining the set \( T|_{Z_i,I_j} \) (see, e.g., Table 1). Then it continues by dropping from these region-based trajectories the timestamps. This produces distinct transactions of the form \( (ID, SetOfROIs) \), one for each trajectory crossing the base cell, where \( ID \) is the trajectory identifier associated with the sets of ROIs traversed by the trajectory.

The transactional dataset obtained for each base cell \( C \) is stored in vertical form, and it is thus composed of pairs \( \langle \{r\}, s(\{r\}) \rangle \), where \( r \) is a ROI in \( C \) and
s(\{r\}) is the non-empty set of trajectory identifiers crossing r in C. We call these pairs 1-regionsets. As an example of this further ETL step, consider the restricted trajectories in Table 1. Figure 3 shows the resulting vertical dataset after removing the timestamps associated with each region-based observation.

Finally, we can complete the process of ETL by producing the collection of FSROIs to store in each base cell. This requires to fix a given threshold $\sigma$, and apply a FIM algorithm to the 1-regionsets.

Besides the FSROIs measure, which contains all the frequent patterns and associated supports that hold in the base cell, we also store an auxiliary measure, used to compute the super-aggregate of FSROIs during spatio-temporal roll-ups. This auxiliary measure is called 1-pattern and contains the 1-regionsets. Unlike the FSROIs measure, the 1-pattern also maintains the knowledge about the identities of the trajectories (crossing the associated ROI). Note that a crucial design decision is the choice of the support threshold $\sigma$. By only storing 1-regionsets that are frequent with respect to $\sigma$, we cannot use the TDW to extract sets of ROIs that are frequent with respect to a threshold smaller than $\sigma$.

Even if the stored 1-patterns are used to extract aggregates using a support threshold larger than $\sigma$, some approximation can occur. In fact, a 1-regionset may be infrequent in a base cell $(Z, I_j)$ and be frequent in another one $(Z, I_h)$. Notice that this can happen for two base cells corresponding to the same spatial zone and different time intervals, since these cells share the same set of regions. Since nothing is stored for infrequent patterns, this may generate errors during the roll-up. In order to avoid producing too large errors during aggregation, in accordance with [3], we also store some infrequent 1-regionsets in each cell, i.e., not only frequent ones but also sub-frequent ones.

**Definition 5.** Let $p = (\{r\}, s)$ be a 1-regionset with $r$ being a region inside cell C, let $\sigma$ be the minimum support threshold, let $\epsilon$ be the maximum support error s.t. $0 \leq \sigma - \epsilon \leq 1$, and let $W$ be the number of trajectories crossing C. We say that

- $p$ is frequent in C if $|s| \geq \sigma \cdot W$.
- $p$ is sub-frequent in C if $(\sigma - \epsilon) \cdot W \leq |s| < \sigma \cdot W$.
- $p$ is infrequent in $C$ if $|s| < (\sigma - \epsilon) \ast W$.

Summing up, in each base cell of our TDW the measure $1$-pattern stores both sub-frequent and frequent $1$-regionsets.

### 5.2 Spatio-Temporal Aggregation

The next step is to define an aggregate function to answer roll-up queries, i.e., to determine which are the FRSROIs occurring in a specific spatial zone and during a given temporal interval, where either the zone or the interval can be larger than the granularity of base cells of our data cube. This spatio-temporal aggregate operator cannot be simply defined in a distributive way starting from the FSRQIs measure. In fact, if we join the FRSROIs contained in the base cells, the support of some patterns can be wrongly estimated and some patterns spanning over several base cells can be lost. For example, consider the trajectory $ID_1$ in Figure 2 that traverses regions $r_1$ and $r_2$ both covered by zone $Z_1$. Since $r_1$ is visited at timestamp $t_2 \in I_1$ and $r_2$ is visited at timestamp $t_5 \in I_2$ the pattern \{r$_1$, r$_2$\} correctly does not appear to be supported by trajectory $ID_1$ either in $(Z_1, I_1)$ or in $(Z_1, I_2)$. However, when we join $I_1$ and $I_2$ and we want to extract the patterns in $(Z_1, I_1 \cup I_2)$. $ID_1$ should be counted in the support of \{r$_1$, r$_2$\}. This problem is related to the fact that our trajectories may be split and assigned to more than one base cell $(Z, I)$.

In order to obtain the aggregate value for the FSROI measure for spatially and/or temporally adjacent cells of the TDW we exploit the $1$-patterns stored in each of these cells. We first roll-up the $1$-pattern measure (Algorithm 1) and then we compute the FRSROIs by applying to the resulting $1$-regionsets the same FIM algorithm used in the loading phase. As discussed in Section 6, we approximate the exact collection of FSRQIs in a very accurate way, without resorting to the raw data $T$.

In Algorithm 1 we present the pseudo code for the spatio-temporal aggregation over the interval $I = I_i \cup I_{i+1} \ldots \cup I_{i+v}$ and $Z = Z_{j_0} \cup Z_{j_1} \ldots \cup Z_{j_m}$. We denote by $W$ the number of distinct trajectories (i.e., restricted trajectories) occurring in the cell $(Z, I)$. This is a measure that can be modelled in our TDW as we did in [10]. Our goal is to obtain all the $1$-regionsets occurring in the ST-cell $(Z, I)$, whose supports are not less than $(\sigma - \epsilon) \ast W$, i.e. the frequent and sub-frequent $1$-regionsets occurring in the larger cell $(Z, I)$.

Algorithm 1 proceeds by selecting each spatial zone $\bar{Z}$ composing $Z$ (line 2) and aggregating it along the temporal dimension, i.e., $(\bar{Z}, I_i) \cup (\bar{Z}, I_{i+1}) \ldots \cup (\bar{Z}, I_{i+v})$. Line 3-6 implements the temporal aggregation: for each region $r$ in the spatial zone $\bar{Z}$ (denoted by $R_{\bar{Z}}$), we build the set of trajectory identifiers $S$ crossing $r$ in any temporal interval belonging to $I$. To efficiently implement the union of the (ordered) list of identifiers, we use a heap (line 6). Moreover, in order to remove infrequent $1$-regionsets, two checks are performed one before and the other after the union. The first one (line 5) is based on the sum of the cardinalities of the tidSets associated with $r$ in the different time intervals. If such a sum is less than $(\sigma - \epsilon) \ast W$ the region can be safely removed since this
Algorithm 1 Algorithm for spatio-temporal aggregation of 1-regionsets.

INPUT: \( P(\bar{Z}, \bar{I}) \) is the set of frequent and subfrequent 1-regionsets in \((Z, I)\) where \( Z = \{Z_{j_0}, Z_{j_1}, \ldots, Z_{j_m}\} \) and \( I = \{I_i, I_{i+1}, \ldots, I_{i+v}\}\) and \( \bar{Z} = Z_{j_0} \cup Z_{j_1} \cup \ldots \cup Z_{j_m} \) is a rectangle and \( \bar{I} = I_i \cup I_{i+1} \cup \ldots \cup I_{i+v} \) is a convex interval; \( \sigma \) is the minimum support; \( \epsilon \) is the maximum support error; \( W = (Z, I) \) presence.

OUTPUT: \( P \), the set of frequent and subfrequent 1-regionsets in \((Z, I)\).

1: \( P \leftarrow \emptyset \)
2: for all \( \bar{Z} \in \{Z_{j_0}, Z_{j_1}, \ldots, Z_{j_m}\} \) do
3:    for all \( r \in R_{\bar{Z}} \) do
4:        let \( \langle\{r\}, s_{i+k}\rangle \in P(\bar{Z}, I_{i+k}) \text{ with } k \in \{0, \ldots, v\} \)
5:        if \( \Sigma_{k=0}^v |s_{i+k}| \geq (\sigma - \epsilon) \ast W \) then
6:            \( S \leftarrow \bigcup_{k=0}^v s_{i+k} \)
7:            if \( |S| \geq (\sigma - \epsilon) \ast W \) then
8:                \( P \leftarrow P \cup \langle\{r\}, S\rangle \)
9:        end if
10:    end if
11: end for
12: end for
13: return \( P \)

The third one (line 4) verifies if the effective cardinality of the union and a 1-regionset is inserted into the result \( P \) only if it is at least sub-frequent (line 8).

It is worth noticing that the ROIs in the 1-patterns occurring in the cells with different \( \bar{Z} \) are distinct, hence once inserted in the output \( P \) the set of identifiers supporting a region \( r \) will not change anymore.

As in the case of base cells, once computed the frequent and subfrequent 1-regionsets for the larger cell, they are given as input to a FIM algorithm with a given threshold \( \sigma \), in order to extract the collection of all the FSROIs.

6 Experimental evaluations

In our experiments we used several datasets. Some of them are real-world sets of trajectories, others are synthetic ones, generated by the traffic simulator described in [2]. Here we report the results obtained for one of the synthetic datasets and two real-world ones. In our approach, the ROIs to exploit must be decided at TDW design time. For these tests the ROIs are different for each dataset, and are computed on the basis of the density of trajectories sampling [4]. The method proposed in [4] identifies ROIs having rectangular shape. First it subdivides the space according to a regular grid, with cells having a small size (microcells), and selects those microcells that are crossed by a large number of trajectories (\( \geq \delta \)). Then, for each of the selected microcells that is not already contained in a ROI, a new ROI is created and expanded repeatedly. Every enlargement step is performed in the direction, if any, that maximizes the average density of the resulting ROI, and such that the ROI does not intersect the zone borders of
our TDW grid. ROIs are thus completely contained in a zone and they do not overlap.

| Dataset | Extent (x × y × t) | | T | | Σ | | L | | Z size (dx × dy) | | #Z | | µx × µy | | δ | | #ROI | | Σ | | SReg |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Athens | 8.4k × 10.5k × 2.6k | | 273 | | | 12k | | 1.2k × 1.5k | | 49 | | 60 × 60 | | 15 | | 64 | | 2k |
| Milan | 70k × 90k × 86k | | 10k | | | 64k | | 2k × 2k | | 1575 | | 100 × 100 | | 25 | | 2672 | | 130k |
| Tiger | 720k × 780k × 300 | | 151k | | | 10M | | 60k × 60k | | 156 | | 48 × 64 | | 2415 | | 156 | | 733k |

Table 2. Datasets

**Athens dataset.** The first real-world dataset contains the trajectories of trucks in Athens (Greece). The dataset consists of 273 trajectories, for a total of 112203 point-based observations, covering an interval of 2616 time units. The observation density analysis leads to 64 ROIs. The leftmost image in Figure 4 visually illustrates the path of trajectories in this dataset as well as the ROIs, indicated in all the images as grey rectangles (red in the colour version of the paper). The area interested by the object movement is divided into 49 spatial zones, corresponding to the subdivision of the spatial dimension of our TDW.

**Milan dataset.** The second real-world dataset contains the trajectories of real vehicles in Milan (Italy). Those data, recorded in 2008 as part of a traffic flow study, consist of 10785 trajectories, for a total of 64749 point-based observations, covering an interval of 86400 time units. Density based ROIs are 2672. The central image in Figure 4 shows the trajectories and ROIs in this dataset. However, due to the high density of trajectories, ROIs appear as a darker area surrounded by several spaghetti-like trajectories. In this case the TDW spatial grid consists of 1575 spatial zones.

**TIGER dataset.** Finally, a synthetic dataset was generated using a map of the county of San Joaquin road network (source: US Census TIGER/LINE, county code 06077), and contains the trajectories of 151000 distinct objects regularly monitored for 300 time units. Unlike the case of the real-world data, all points are equally distant in time. In this case, density based ROIs are 156. The rightmost image in Figure 4 visually illustrates the path of trajectories in this dataset as well as the ROIs. The TDW spatial grid consists of 156 spatial zones.

Table 2 summarises the properties of the datasets, as well as some information about the spatial grid of the TDW, and the ROIs. The TDW grid and the parameters to determine the ROIs were chosen for making more significant and interesting the subsequent tests. The column extent contains the range of x, y and t coordinates of observations, |T| indicates the number of trajectories of the dataset, |L| denote the number of observations of all trajectories, Z size is the extent of spatial zones, #Z is the number of spatial zones, µx and µy are the sizes of initial microcells used for ROI detection, δ is the trajectory density threshold for ROI detection, #ROI is the number of ROIs detected, and, finally, |SReg| is the overall number of region-based observations stored in the region-based trajectories.
6.1 Experiments

The proposed method for the computation of frequent patterns over the union of a set of base cells makes use of precomputed data (1-pattern) stored in those cells. It is natural to wonder whether such precomputation pays, that is, if the possible advantages justify our TDW approach. In this subsection we analyse the performances of our approach from a computational and storage usage perspective, and assess the approximation quality of the aggregate function.

Storage size. Figure 5 shows that the size of the storage, required to maintain all the 1-pattern measures in our TDW, is largely smaller than the size of original dataset. Note that no data compression was considered to compute such statistics. For example, 1-patterns are (ordered) lists of identifiers associated with ROI identifiers. Such numerical lists can strongly be compressed employing the same methods as the ones used in Information Retrieval for compressing postings lists [13].

Looking at the figures, we can observe how such large storage reduction mainly depends on the region-based transformation. Many original observations
that do not fall into any ROI are in fact discarded. The region-based transac-
tional datasets, after the removal of observation timestamps, corresponds to the 
curves labelled as “region based dataset”. Note that the relatively large number 
of ROIs, used for the Milan dataset, makes worse the storage reduction.

As expected, when we increase the minimum support, the storage reduction 
increases, because infrequent 1-regionsets disappear from the stored 1-pattern 
measure.

Finally, each curve corresponds to TDW grids with different temporal gran-
ularities. Since the time ranges of the datasets are not homogeneous, we first 
selected the overall number of time slices, and then determined the time granu-
larities for each dataset accordingly. Thus, for the Milan dataset, 30 time slices 
implies that the time granularity is \( dt = \frac{86400}{30} = 2880 \). From Figure 5 we 
can observe that the smallest storage reduction occurs for the smaller granularity 
(60 time slices), since some redundancies occur in the 1-regionsets stored in 
the 1-pattern measure.

Running time. In order to answer a FSROI query over the union of a set 
of spatio-temporal cells, we first need to roll-up the 1-patterns (Algorithm 1), 
and then to extract the FSROIs from the resulting dataset. For this extraction, 
we exploited kDCI, a very fast FIM algorithm [7].

As we are interested in analysing the cost of Algorithm 1, we considered the 
worst case, i.e. the global roll-up over the complete spatio-temporal domain 
of our TDW. Of course, since Algorithm 1 starts from the 1-patterns stored in the 
various cells, the final datasets obtained will only contain the frequent and sub-
frequent 1-regionsets. Applying kDCI over such resulting dataset is obviously 
faster than applying it to the complete vertical dataset, which also contains in-
frequent 1-regionsets. However, the cost of Algorithm 1 could exceed the possible 
benefits in the running time of kDCI. Hence, it is interesting to evaluate this 
trade-off.

For each dataset and for several values of the minimum support threshold 
and time granularity we computed the query processing time of the proposed 
method on the union of all spatio-temporal cells, and the running time of kDCI 
on the original (region-based) data. The plots in Figure 6 shows the results.

We can observe that a coarser time granularity (i.e., a smaller number of 
time slices) entails a better running time. This can be explained by the reduced 
number of temporal aggregations needed to answer a query (inner loop of Algo-
rithm 1). Similarly, more restrictive minimum support values are associated with 
a smaller running time as a consequence of the decreased number of frequent 
1-regionsets and FSROIs. Apart for the Athens dataset, whose query processing 
times are so small that differences are mainly determined by the resolution of the 
timer used for measures, the experiments show that the proposed method 
outperforms the computation from scratch for medium and high support values, 
and behaves similarly for small support values.

Approximation quality. The above experiments highlight the advantages 
of starting the computation of FSROIs from the 1-patterns (frequent and sub-
frequent 1-regionsets) stored in the cells, instead of starting from the complete
Fig. 5. Storage reduction by varying the minimum support threshold and number of slices involved: storage size of all the 1-patterns of the TDW base cells vs. storage size of all the original trajectory observations.

Fig. 6. FIM query running time by varying the minimum support threshold and number of slices involved.
region-based dataset. Such a choice, however, may entail some approximation in the computation of the aggregate function, in particular either the loss of some of the frequent patterns or the under-estimation of their supports. To assess the quality of the results of FSROI queries we used the metric Similarity [12], which accounts for missing patterns or patterns having incorrect support. When computing Similarity the patterns are weighted according to the difference between the correct support value and the resulting one. A Similarity value equal to 1 indicates that the exact and the approximate result sets exactly contain the same patterns with the same support values, whereas a Similarity value equal to 0 indicates that the two result sets are completely different. Similarity was measured for several values of minimum support and time granularity, the same used in the previously described experiments. It is worth noticing that the support thresholds used for loading the TDW are the same as the ones used for the roll-up queries, while the queries span over the complete spatio-temporal TDW domain. In the case of Milan and Tiger datasets the results are always identical to the exact ones. For this reason, in Table 3, we just report the Similarity for the Athens dataset. We stress that in most cases Similarity is greater than 0.95 and often larger than 0.999. This means not only that nearly all patterns are found, but also that the difference between exact and resulting support is small, and thus the patterns can be reliably used to build association rules.

<table>
<thead>
<tr>
<th>Similarity</th>
<th>Number of time slices</th>
</tr>
</thead>
<tbody>
<tr>
<td>min support (%)</td>
<td>5</td>
</tr>
<tr>
<td>3.69</td>
<td>0.99983</td>
</tr>
<tr>
<td>5.53</td>
<td>0.99801</td>
</tr>
<tr>
<td>7.38</td>
<td>0.99997</td>
</tr>
<tr>
<td>9.22</td>
<td>0.97949</td>
</tr>
<tr>
<td>11.07</td>
<td>0.9601</td>
</tr>
<tr>
<td>12.91</td>
<td>0.99819</td>
</tr>
<tr>
<td>14.76</td>
<td>0.94779</td>
</tr>
<tr>
<td>18.45</td>
<td>0.99793</td>
</tr>
</tbody>
</table>

Table 3. Similarity Results for the Athens dataset

7 Conclusion

In this paper we have discussed the ETL process for storing, in each base cell of a TDW, patterns concerning sets of ROIs, which are visited, in any order, by large numbers of moving objects. We have also presented the problem of spatio-temporal aggregation of such patterns, needed to answer analytical multidimensional queries, asking for FSROIs occurring in a given area during a certain time interval.
Acknowledgements

This work has been partially supported by the FP6-14915 IST/FET Project GeoPKDD funded by the EU.

References