Warehousing and Mining Aggregate Measures Over Trajectories of Moving Objects

Fernando José Braz

Supervisor
Salvatore Orlando

PhD Coordinator
Annalisa Bossi

January, 2009
Author’s e-mail:  fbraz@dsi.unive.it

Author’s address:
Dipartimento di Informatica
Università Ca’ Foscari di Venezia
Via Torino, 155
30172 Venezia Mestre – Italia
tel. +39 041 2348411
fax. +39 041 2348419
web: http://www.dsi.unive.it
To my wife
To my daughter
To my family
To my friends

To those who are not here with me, but from some place are seeing this conquest:
my father Adério and my friends Edson and Irapuan.
God bless you
Abstract

The development of new technologies for mobile devices and low-cost sensors results in the possibility of storing large data volumes about trajectories of moving objects. In this thesis we propose a multi-dimensional data model to store aggregate measures computed over such huge data volumes. This allowed us to define a Trajectory Data Warehouse (TDW) that is loaded by managing and transforming a data stream of spatio-temporal observations of moving objects, arriving in an irregular and unbounded way. We discuss how standard data warehousing tools can be used to store trajectories and to compute OLAP operations over them. We construct a data cube where the dimensions are the spatial coordinates and the time, discretized according to a regular grid.

The stream nature of our huge input data, concerning moving object observations, may make it difficult to build and maintain a data warehouse. The transformation and loading phase of our TDW is particularly challenging, since it has to work under resource constraints (memory, processing), by accommodating possible irregular arrival rate of input data. In our TDW the identifier of the trajectories are abstracted in favour of aggregate information concerning global properties of a set of moving objects. Each TDW record defined by spatio-temporal coordinates stores measures (average speed, traveled distance, etc.) that represent properties of a set of objects traversing that cell. The design issues of our trajectory data warehouse are: the loading phase and the rolling-up of aggregate measures for different spatio-temporal granularities.

The data about trajectories stored in our TDW can be mined to extract knowledge and reveal patterns associated with the occurrence of real phenomena. For example, if stored data are concerned with moving vehicles in a read network, a traffic manager could be interested in analyzing occurrence of Traffic Jams. Therefore, the previous knowledge about the conditions that cause (or co-occur with) a Traffic Jam phenomenon may be used to forecast the occurrence and allows the user to take suitable actions. To this end, we have investigated the use of Data Mining tasks to extract patterns and models, useful for forecasting phenomenon occurrences. In particular, we have focused on the co-location patterns, a type of Spatio Temporal Data Mining technique. We have developed an algorithm (Target Event Co-location Pattern) to extract relevant and interesting co-location patterns from aggregates.
The goal is to find, in a delimited area defined by the user, occurrences of *co-location patterns* among events characterizing multiple moving objects. In order to reduce the computational cost involved in this process, the algorithm allows the analyst to define a constraint over the *co-location pattern* by using a so-called *target event* defined by the user. In other words, the algorithm can be used to forecast the occurrence of specific events related to moving objects in a given area by analyzing the occurrences of other related events that occur in the neighborhood area.
Acknowledgments

This project would not be possible without the support received from many people. There is not space enough to thank them all. Therefore, I will only mention some of them.

Salvatore Orlando was a tutor, a professor, and an indescribable friend. I cannot describe his importance during these three years. I will never be able to reward his infinite patience with my difficulties and limitations. I learned much more that scientific concepts and theories. Salvatore taught me to maintain the hope and to believe in myself, and that is priceless.

Gian Maria Zuppi is another special person. He is a visionary, a man who can make anything to help another ones. His huge scientific knowledge is only overcome by his infinite will to aid whoever it needs.

I also would like to thank the Dipartimento di Informatica from Ca’ Foscari Universitys personnel for their support. I know I have found a lot of new friends during this season. Special thanks to Andrea and Samuel, fundamental ones to conclude this task.

To UNIVILLE, I would like to thank the complete staff of the Divisao de Tecnologia de Informacao (IT Division Staff). They were fantastic and very patient to attend and help solving my several requests. Special thanks to professors Paulo Ivo, Sandra Furlan, Ivanilda and Rosalvo too, for all support. I would like to thank my colleagues of the Departamento de Informatica.

All the gratitude for my family, which didn’t measure efforts to lean in this project. They gave me much more than I needed. They were fundamentals to complete this project. It is not possible to relate all of them here, they are too many. They are very special people.

Finally, I would like to thank (in a very special way) my wife and my daughter, Mariceia and Laura. From the beginning until the conclusion of this project I received the unconditional and unrestricted support. I know that they abandoned personal projects to lean me in this project. I know that this period was very hard for them. We had many very difficult moments, however, for each difficulty they presented the solution. I just can’t imagine how this project would be concluded without them. Today I feel better and more completed because of them. Briefly, the PhD degree is an important achievement, however, the real conquest is to understand the importance of them in my life.
Contents

1 Introduction 1
  1.1 Data Warehouse Technology .............................. 4
  1.2 Data Mining Concepts .................................. 6
  1.3 Contributions ........................................... 7
  1.4 Thesis overview ......................................... 8

I First Part 11

2 Trajectory Data Warehouse 13
  2.1 Related Work ............................................ 14
  2.2 Problem Description .................................... 18
  2.3 Star Model - Trajectories ............................... 19
  2.4 Aggregate Measures .................................... 20
  2.5 Load Phase .............................................. 23
  2.6 Presence Measure ....................................... 26
  2.7 The Prototype ........................................... 28
    2.7.1 Assessments of the Prototype ....................... 32
  2.8 TDW: Events Occurrences Patterns .................... 35
    2.8.1 Traffic Jam Patterns on TDW ....................... 36
    2.8.2 Pre-processing of the TDW data .................... 37
    2.8.3 Statistical Analysis ................................ 40
    2.8.4 Data Mining Experiment Results ................... 44
  2.9 Conclusions ............................................. 46

II Second Part 49

3 Pattern Mining applied to Trajectories 51
  3.1 General Framework ...................................... 51
    3.1.1 Intra-zone non sequential patterns ............... 52
    3.1.2 Intra-zone sequential patterns .................... 52
    3.1.3 Inter-zone non-sequential patterns ............... 53
    3.1.4 Inter-zone sequential pattern ...................... 54
  3.2 Sequential Pattern ..................................... 54
  3.3 Clustering .............................................. 56
  3.4 Co-location .............................................. 58
3.5 Conclusions .......................................................... 66

4 Trajectory Data Warehouse Patterns .......................... 69
4.1 Spatial co-location patterns ........................................ 70
4.2 Target Spatio-Temporal Co-location Pattern Algorithm ........ 73
4.3 Execution Trace of Target Spatio-Temporal Co-location Pattern Algorithm .......................................................... 78
4.4 Experimental Evaluation ........................................... 84
4.4.1 Number of Timestamps ........................................ 84
4.4.2 Spatio-Temporal Prevalence Index Threshold .................. 86
4.4.3 Target Event Size ............................................. 87
4.4.4 Co-location Patterns ........................................... 88

Conclusions .................................................................. 91

Bibliography ................................................................. 95
List of Figures

1.1  KDD Process .................................................. 2

2.1  TDW environment ............................................. 13
2.2  Trajectory with a sampling .................................. 19
2.3  Star Model ...................................................... 20
2.4  Spatial or temporal hierarchy ................................. 20
2.5  Correctly counted .............................................. 23
2.6  Duplicate counting ............................................. 23
2.7  Linear interpolation ........................................... 23
2.8  Interpolated trajectory (spatial-temporal points) ......... 23
2.9  Results Interface .............................................. 28
2.10 Loading Settings ................................................ 30
2.11 Granularity 125 ............................................... 33
2.12 Granularity 250 ............................................... 33
2.13 Granularity 500 ............................................... 34
2.14 Loading .......................................................... 35
2.15 Correlation Time vs Distance ................................ 41
2.16 Correlation Time vs dPres .................................... 41
2.17 Correlation Time vs lvlSpeed ................................ 42
2.18 Correlation Distance vs dPres ................................. 42
2.19 Correlation Distance vs lvlSpeed ............................ 43
2.20 Correlation dPres vs lvlSpeed ................................. 43
2.21 Number of Cells (%) vs TJ Density - Free Traffic ....... 45
2.22 Number of Cells (%) vs TJ Density - Traffic Jam ........ 46

3.1  Co-location Patterns - example .............................. 61

4.1  Spatial Co-location Patterns - TDW ......................... 70
4.2  Target Cell ...................................................... 74
4.3  Level 1 .......................................................... 75
4.4  Level 2 .......................................................... 75
4.5  Target Spatio-Temporal Colocation Concepts .............. 76
4.6  Neighborhood t - 2 ............................................. 79
4.7  Neighborhood t - 1 ............................................. 80
4.8  Neighborhood t .................................................. 81
4.9  Execution Time .................................................. 85
4.10 Range Cells ..................................................... 85
4.11 Candidate Events ............................................... 86
<table>
<thead>
<tr>
<th>Figure Number</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.12</td>
<td>Patterns</td>
<td>86</td>
</tr>
<tr>
<td>4.13</td>
<td>Prevalence vs Pattern</td>
<td>86</td>
</tr>
<tr>
<td>4.14</td>
<td>Prevalence vs Time</td>
<td>86</td>
</tr>
<tr>
<td>4.15</td>
<td>Size vs Execution Time</td>
<td>87</td>
</tr>
<tr>
<td>4.16</td>
<td>Size vs Range Cells</td>
<td>87</td>
</tr>
<tr>
<td>4.17</td>
<td>Size vs Patterns</td>
<td>88</td>
</tr>
<tr>
<td>4.18</td>
<td>Size vs Candidate</td>
<td>88</td>
</tr>
</tbody>
</table>
## List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Cells representation - for each observation</td>
<td>25</td>
</tr>
<tr>
<td>2.2</td>
<td>Sequence of segments composing the interpolated trajectory, and the base cells that completely include each segment.</td>
<td>25</td>
</tr>
<tr>
<td>2.3</td>
<td>Fact Table</td>
<td>29</td>
</tr>
<tr>
<td>2.4</td>
<td>X or Y Dimension Table</td>
<td>29</td>
</tr>
<tr>
<td>2.5</td>
<td>T Dimension Table</td>
<td>29</td>
</tr>
<tr>
<td>2.6</td>
<td>Buffer Table</td>
<td>30</td>
</tr>
<tr>
<td>2.7</td>
<td>Spatio-temporal Granularity</td>
<td>32</td>
</tr>
<tr>
<td>2.8</td>
<td>Trajectories Log</td>
<td>38</td>
</tr>
<tr>
<td>2.9</td>
<td>Labeled Table</td>
<td>39</td>
</tr>
<tr>
<td>2.10</td>
<td>TJ Density Discretization</td>
<td>39</td>
</tr>
<tr>
<td>2.11</td>
<td>Training Set Scheme</td>
<td>40</td>
</tr>
<tr>
<td>2.12</td>
<td>Correlation - Pearson coefficient</td>
<td>41</td>
</tr>
<tr>
<td>2.13</td>
<td>Accuracy Results (025TJ, 050TJ, 075TJ, TJ)</td>
<td>44</td>
</tr>
<tr>
<td>4.1</td>
<td>Spatial Co-location Size 1</td>
<td>71</td>
</tr>
<tr>
<td>4.2</td>
<td>Spatial Co-location Size 2</td>
<td>72</td>
</tr>
<tr>
<td>4.3</td>
<td>Prevalence Time x Participation Index</td>
<td>83</td>
</tr>
<tr>
<td>4.4</td>
<td>Dataset Records</td>
<td>84</td>
</tr>
<tr>
<td>4.5</td>
<td>Target Event Size</td>
<td>87</td>
</tr>
<tr>
<td>4.6</td>
<td>Co-location Patterns</td>
<td>88</td>
</tr>
</tbody>
</table>
The recent development of mobile and location aware technologies (like cellular phones equipped with GPS), allow for tracking the movement of objects. This fact contributes for increasing the amount of stored data related to moving objects. A singular characteristic of a moving object is the possibility of its position to change over time. Therefore, an time ordered set of object positions can represent the trajectory of that object. There are several domains where it is possible to find the trajectory concept, for example: traffic management, migratory habits, mobile phone, forest management, natural phenomena (hurricanes) etc.

The analysis of trajectory data can be fundamental for understanding the behavior of objects in a given environment. Patterns of migratory habits may be useful to understand the behavior of an animal species and to increase survival chances for endangered species. In traffic management area, a pattern may reveal the occurrence of a phenomenon (traffic jam) in a given area of a road network map. Analyzing trajectory data allows, for example, deriving behavioral patterns of birds and humans. Patterns may enable understanding the spreading of some diseases, inducing suitable measures to protect populations and prevent further spreading of the disease, or in animal monitoring to increase survival chances for endangered species.

However, before the analysis of data, it is necessary to receive, process and store the data volume. Sometimes, in some domains, it is not possible to forecast neither the data volume nor the data flow. This characteristic increases the difficulty to process trajectories data, thus making necessary to adapt the data processing to the available computational resources. Therefore, the development of techniques to process and store trajectories data is a very important point of research in the moving objects area.

Moving objects represent a stimulant and new area of research. In this work we present an effort to investigate techniques to reveal the hidden knowledge on trajectory data. This effort involves both storage and analysis of trajectory data. In summary, we are focused on to study the use of technologies of knowledge discovery in databases (KDD) applied to trajectory data. The terms knowledge discovery in databases (KDD) and data mining are often used interchangeably. In this thesis we consider KDD as a process for finding useful information and patterns in data. KDD
is a process composed of many steps. *Data mining* is just one of these steps. *Data mining* is the use of algorithms to extract the information and patterns derived by the KDD process [17], [16], [18].

Figure 1.1 presents the complete KDD process, in the following we detail each KDD step:

**Selection:** The data needed for the data mining process may be obtained from many different data sources. This first step obtains the data from various databases and/or files.

**Preprocessing:** The data to be used by the process may have incorrect or missing data. There may be anomalous data from multiple sources involving different data types and metrics. There may be many different activities performed at this time. The goal is to correct or remove the erroneous data.

**Transformation:** Data from different sources must be converted into a common format for processing. Data reduction may be used to reduce the number of possible data values being considered.

**Data mining:** In this step happens the use of the data mining algorithms on the transformed data set.
Interpretation/evaluation: In this step happens the interpretation of the knowledge revealed by the previous steps. Various visualization and GUI strategies are used at this step.

A complete KDD process may involve a set of different research areas: databases, machine learning, pattern recognition, statistics, artificial intelligence, data visualization, information retrieval, and high-performance computing. The database concept provides the infrastructure to store, access, and manipulate data.

In the same way, data warehouse technology offers an environment for storing data. It refers to the current business trend of collecting and cleaning transactional data to make them available for online analysis and decision support. Data Warehouse is a traditional well-established technology that supports critical business decision-making. Traditional databases contain operational data that represent the day-to-day needs of a company. Meanwhile, a Data Warehouse contains informational data, it can be used to support functions as planning and forecasting. Therefore, Data Warehouse can be considered as a component of the KDD process. Data Warehouse technology presents summarized data along of different dimensions. Besides, Data Warehouse measures represent features of a given fact and different views of the same fact can be obtained along the various dimension and hierarchies. This offers a powerful tool for data analysis.

The intensive use of databases results in very large data volumes. These data volumes store a crucial knowledge about different domains. Data Mining is defined as a process to find hidden information stored in huge amounts of data. Traditional database queries access those data volumes using a well-defined query through a specific language such as SQL (Structured Query Language). However, the output of a SQL query is, basically, a subset of the database. The proposal of the Data Mining concept is to present not a subset of a database, but to reveal the hidden knowledge on a data volume. This knowledge can be expressed as patterns, rules, clusters, classifications, trends, etc.

In this thesis we present a discussion about the KDD process applied to analysis of trajectory data. Therefore, the use of a traditional Data Warehouse technology and Data Mining techniques is a natural choice to start analyzing trajectory data. We discuss how standard data warehousing tools can be used to store trajectories and to compute OLAP operations over them. We construct a data cube where the dimensions are the spatial coordinates and the time, discretized according to a regular grid.

In the following sections we recall the main Data Warehouse and Data Mining concepts. Besides, we describe the main achievements and the structure of the Thesis.
1.1 Data Warehouse Technology

The traditional operational databases are used, most of the time, to store very large volume of detailed information. Operational databases work in environments where the goal is to record a large volume of transactions. In these environments the data are not redundant. They present the details of each transaction performed in the database. A database that stores individual sales records is an example of an operational database. In that case, each sales record may represent an item bought by a customer. Meanwhile, a customer table stores the customer information. Therefore, an operational database has a low level of redundant information and a high level of individual record details. It is a very efficient data format to provide, with accuracy and safety, an environment to receive, store and update a very large data volume of an intensive transaction environment. However, it is a very inefficient data format for decision support and knowledge discovery. Data Warehouse model was introduced to solve this problem. A data warehouse system is not transaction intensive, and the goal is to present summarized values originated from different sources.

The traditional data warehouse is defined as a subject-oriented data collection integrated from various operational databases. \[32\] defines a data warehouse as a subject-oriented, integrated, time-varying, non-volatile collection of data that is used primarily in organizational decision making. A Data warehouse provides an integrated environment by extracting, filtering, and integrating relevant information from various data sources. According to a data warehouse model, data can be summarized and aggregated in a multidimensional way in order to facilitate access and data analysis.

A data warehouse architecture includes tools for extracting data from multiple operational databases and external sources. The goal of those tools is to clean, transform and integrate the data volume in order to load data into the data warehouse. Another function of those tools is to refresh the warehouse to reflect updates at the sources and to purge data from the warehouse.

The multidimensional paradigm involves the following concepts: facts, dimensions and measures. A measure is the attribute of a fact, which represents the state of a situation with regards to the dimensions of interest. A dimension has members, each member represents a position on the dimensional axis. The members of a dimension may be structured in a hierarchical manner, creating the different levels of granularity of information. Each of the numeric measures depends on a set of dimensions, which provide the context for the measure. For example, in a sales data warehouse, time of sale, sales district, salesperson, and product might be some of the dimensions of interest. These dimensions can be hierarchical. Therefore, time of sale may be organized as a day-month-quarter-year hierarchy and the product dimension can be organized as a product-category-industry hierarchy. Typical OLAP operations include rollup (increasing the level of aggregation) and drill-down (decreasing the level of aggregation or increasing detail) along one or more dimen-
1.1. Data Warehouse Technology

The multidimensional paradigm can be modeled by using one of the three data structures: *Star Schema*, *Snowflake Schema* and the *Fact Constellation*. In this thesis we are focused on the *star schema*. It is composed of one central *fact table* and some *dimension* tables. The fact table has measures and dimension keys in order to link to the *dimension* tables. Each tuple in the fact table consists of a pointer to each of the dimensions that provide its multidimensional coordinates, and stores the numeric measures for those coordinates. Each dimension table consists of columns that correspond to attributes of the dimension. Therefore, a data warehouse can be composed of a number of dimensions and each dimension may have multiple levels. There could be a very large number of intermediate aggregated data cubes (called *cuboids*) to be computed.

When the goal is to analyze data which contains spatial components (*measures* or *dimensions*), the traditional DW technology is very deficient. Nowadays, the solution in order to provide an environment to analyze very large spatial data volumes is to merge both technologies of *Geographical Information Systems - GIS* and *DW*. It is the source of the *Spatial Data Warehouse - SDW* technology. The GIS were developed to store, manipulate and display spatial data, these tasks can be completed by a GIS. However, the GIS are transaction-oriented systems and cannot compute a lot of crucial operations to analyze large data volumes (e.g. summarized information, cross-referenced information, interactive exploration of data, etc). Besides, GIS are not suited for temporal data, are very slow to aggregate data and hardly deal with multiple levels of data granularity.

On the other hand, the simple merging between SIG and OLAP it is not enough to allow the analysis of spatial data. In most of the cases the spatial data are treated like a descriptive data and the spatial analysis is constrained by nominal locations (ex: name of city, name of state, name of country, etc).

In this thesis we investigate a classical data warehouse model to store trajectory information in a multidimensional model. The proposal is to use a traditional data warehouse product to store trajectory data. Our *Trajectory Data Warehouse* stores information about sets of trajectories in a given area at a given timestamp. Therefore, our *Trajectory Data Warehouse* must be able to manage *spatio-temporal* data characteristics and associated dimensions. Besides, the TDW has to be able to work in a data stream environment, where the trajectory data observations are received in a continuous way. Our TDW has to receive, process and store the trajectory data, produced in an unbounded way and arriving in stream, by coping with space and time constraints.
1.2 Data Mining Concepts

The large data volumes stored in databases present an interesting problem: the difficulty to extract and reveal the knowledge of that data volumes. The traditional database technology can show the data items, but it cannot reveal tendencies or patterns of relationships between occurrences of data items. The output of traditional query is a subset of the database, just a set of registers, not a knowledge of the database.

Data mining can be executed on various data types: relational databases, data warehouses, multimedia databases, spatial databases, text documents and world wide web (www), this list grows day by day. A Data Mining task can involve many algorithms to reveal the knowledge in a given data volume. The different algorithms can be used in order to accomplish several tasks. The goal of these algorithms is to fit a model to the data. The algorithms examine the data and determine a model that is closest to the characteristics of the data being examined. The created model can be either predictive or descriptive. The predictive model makes a prediction about data values by using known results found from different data. In this model the data source can be a historical dataset. Predictive model data mining tasks include classification, regression and time series analysis. Meanwhile, a descriptive model identifies patterns or relationships in data volume. Therefore, a descriptive model can be used as a way to explore the properties of the data. In this model the goal is not to predict new properties. The most common descriptive model data mining tasks are: clustering, association rules, and sequence discovery. In the following we briefly present some of the most common data mining tasks.

[Classification] This task maps data into predefined classes. It is a kind of supervised learning because the classes are determined before examining the data. Classification algorithms require that the classes be defined based on data attribute values. An input pattern is classified into one of several classes based on its similarity to predefined classes.

[Regression] The goal in this task it is to map a data item to a real valued prediction variable. It can be done by the learning of the function that does the mapping.

[Time Series Analysis] This task allows to examine the value of an attribute along the time. Different series of the attribute value are obtained along the time points (daily, weekly, etc.). By using those time series, it is possible, for example, to investigate the level of similarity between different series. Another possibility is to examine the line structure and to classify its behavior.

[Clustering] This task has some similarity with classification task. The prime difference is that, in this task, the groups (classes) are not predefined. In Clustering task the groups are defined by the data alone. Clustering is a kind of unsupervised learning, the proposal is to segment the dataset into groups. It can be done evaluat-
ing the similarity among the data on predefined attributes. The clusters will contain the data items in agreement with the level of similarity among of them.

[Association Rules] Association rules are used to show the relationships among data items. It is one of the most popular tasks of data mining. The prime goal of association rules is to identify patterns of co-occurrence of items \cite{2, 20, 22, 3}. The purchasing of one product when another product is purchased is an example of an association rule.

Association rules work with two main values: support and confidence. The values of support and confidence are used to measure the importance of the rule. Confidence measures the strength of the rule, whereas support measures how often it should occur in the database. For example, in the rule “Customer who buy a Car also buy a CD in 80\% cases”, the value 80\% is the confidence and represents how much that rule can be trusted. The support value represents how many times the rule happens in the database. Many associations can be found by association rule task, but just some of them are interesting. Support and confidence can be used to make this selection. A minimum threshold for support and confidence are defined to do it. The rules with values of support and confidence larger than the threshold may be considered interesting. In this case, the rule has a significant number of cases of usage and a few cases in which it is not valid.

[Sequence Discovery] The goal of this task it is to find sequential patterns in data. The patterns are based on a time sequence of actions. In a market basket scenario, a traditional environment of association rules, there is the assumption that the items are purchased at the same time. Meanwhile, in sequence discovery the items are purchased along the time in some order.

In this thesis we investigate the use of data mining tasks applied to aggregated data of trajectories. We exploit classification and extraction of pattern mining techniques. We investigate the use of classification technique to verify whether it is possible to obtain a high level of accuracy to foresee a traffic occurrence by using the TDW measures. Besides, we present an investigation about the use of a specific pattern extraction method for Spatio Temporal data.

1.3 Contributions

In this thesis, we present original contributions in two related area: data warehouse of trajectories (Trajectory Data Warehouse) and Spatio-Temporal Data Mining applied to trajectories.

The original contribution in the Trajectory Data Warehouse field is the definition of a model of Spatio-Temporal Data Warehouse able to receive, process, compute and store trajectory data. We consider that the Trajectory Data Warehouse works in a data stream environment. This environment presents some characteristics that can hinder the modeling and the maintenance of a data warehouse. The data arriving
in an unpredictable and continuous rate, the large data volume and the resource constraints (memory, processing) are some of these main characteristics. The TDW was proposed in agreement with two basic assumptions:

- The identity of the moving objects and trajectories will be abstract, since we are interested in studying global properties of a set of such objects.

- The base cuboid will be composed of spatio-temporal cells, consisting of regions and time intervals which we are interested in.

The classical star schema was used, with three dimensions: two spatial dimensions $X$ and $Y$, and one temporal dimension $T$. We divided the spatio-temporal into a set of cells $(x,y,t)$, each of them storing measures that represent characteristics of the set of trajectories involved by the cell. The proposed multidimensional modeling of trajectory observations introduces a new measure (presence) in order to solve the problem of duplicate counting. This problem can happen when it is necessary to count the number of distinct trajectories crossing a cell. It is a holistic measure ([26]), and the value is computed by using two proposed auxiliary measures (distributive and algebraic). We have published about these topics in [7] and [56]. Therefore, this thesis presents a proposal to analyze spatio-temporal data based on a classical multidimensional model. The TDW can be implemented by using the star schema, the classical multidimensional model does not need to be extended.

Another original contribution is the research of Spatio-Temporal Mining applied to trajectories. We have investigated the possibility of usage of Spatio-Temporal Mining to find patterns of events of movement. We consider the proposed TDWA as the data source of the mining process. We have published about this investigation in [6] and [5]. We have done a review of the proposals of Spatial and Spatio-Temporal Pattern Mining. The goal is to present an assessment of them and to discuss the requisites of a mechanism to find spatio-temporal patterns on Trajectory Data Warehouse data. We have introduced an algorithm of co-location pattern to be applied in the analysis of TDW data. The proposed algorithm (Target Spatio-Temporal Co-location Pattern) adapts and specializes previous proposals appeared in the literature to extract patterns from spatial data [12], [11] and [31]. The algorithm allows us to forecast the occurrence of events of movement in a given area investigating the occurrences of another related events that occur into the neighborhood area. To the best of our knowledge, it is the first attempt to use co-location patterns to mine a spatio-temporal TDW environment.

1.4 Thesis overview

This thesis is divided into chapters. For the sake of readability, since parts of the citations are common to more chapters, the references are listed at the end of the thesis.
1.4. Thesis overview

The first part of the thesis includes one chapter that presents a general discussion about the usage of Spatio Temporal Data Warehouse to receive, process, compute and store data of a set of moving objects. The first chapter also introduces the problem of modeling a Trajectory Data Warehouse (TDW). Besides, in this chapter we detail our proposal of a TDW and present a prototype. In the same chapter we present an investigation about the possibility of finding occurrences of events by simply analyzing the TDW data. We present a statistical analysis on the data stored in the various cells to discover correlation in such data. It is useful to study co-occurrence of measures in cells without supervised knowledge. We also study the extraction of a classification model, extracted from a labelled TDW. In that investigation we focus on traffic jam patterns.

The second part of the thesis is totally dedicated to present an investigation about the usage of data mining techniques to find patterns of occurrences of events related to movements of objects. We focus on reviewing and discussing proposals of Spatial and Spatio-Temporal Data Mining applied to trajectories. Besides, we introduce the Target Spatio-Temporal Co-location Pattern algorithm. We present details of the algorithm and some results obtained. Finally, the last chapter describes some future works and draws some conclusion. In particular, we discuss some aspects related to time constraint: the goal is to extend the proposed algorithm in order to push into the algorithm gap constraints over the occurrence of consecutive events.
First Part
The traditional data warehouse can be defined as a subject-oriented data collection integrated from various operational databases. In a data warehouse, data can be summarized and aggregated in a multidimensional way in order to facilitate access and data analysis. A Data warehouse provides an integrated environment by extracting, filtering, and integrating relevant information from various data sources. A data stream environment presents some characteristics that can hinder the modelling and the maintenance of a data warehouse. The data arriving in an unpredictable and continuous rate, the large data volume and the resource constraints (memory, processing) are some of these main characteristics. The traditional data warehouse model must be adapted in order to work in agreement with these constraints.

<table>
<thead>
<tr>
<th>Xid</th>
<th>Yid</th>
<th>Time</th>
<th>Speed</th>
<th>Presence</th>
<th>Distance</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>20</td>
<td>5</td>
<td>135</td>
<td>20</td>
<td>50</td>
<td>20</td>
</tr>
<tr>
<td>10</td>
<td>20</td>
<td>10</td>
<td>180</td>
<td>25</td>
<td>70</td>
<td>80</td>
</tr>
<tr>
<td>15</td>
<td>10</td>
<td>5</td>
<td>120</td>
<td>10</td>
<td>80</td>
<td>30</td>
</tr>
<tr>
<td>15</td>
<td>10</td>
<td>10</td>
<td>100</td>
<td>40</td>
<td>30</td>
<td>20</td>
</tr>
</tbody>
</table>

Figure 2.1: TDW environment

The development of new technologies for mobile devices and low-cost sensors results in the possibility of storing larger data volumes about trajectories of moving objects. These data volumes can be stored in a multidimensional model in order to allow an accurate analysis. This model of storage can be defined as a trajectory
data warehouse. The goal is to store, manage and analyze the trajectories data in a multidimensional way. The trajectory can be represented by position \((X \text{ and } Y)\) and time data. A set of observations represents data about several moving objects positions. The trajectory data warehouse has two main problems: the loading phase and the computing of measures. The loading phase has to receive and process the data volume considering the available resources and the data stream characteristics. The aggregated information stored in each cell of the \(DW\) model can be used to reveal knowledge of the objects. It can be done by the usage of the \(OLAP\) operators, these results can be used as input for subsequent analysis.

Figure 2.1 presents the trajectory data warehouse environment. Trajectories data are received, computed and stored in a multidimensional way, each cell (cuboid) represents the characteristics of a set of trajectories involved by the cell coordinates \((x,y,t)\).

In this chapter we present a proposal to implement a Trajectory Data Warehouse. We consider a \(TDW\) as a kind of Spatio Temporal Data Warehouse. We present a prototype which implements the proposed \([7]\) \(TDW\). Besides, Section 2.8 presents a discussion on whether the \(TDW\) stored aggregates can be used to forecast Traffic Jam occurrences. It was done in two steps, in the first one the goal is to investigate the relationship among the \(TDW\) measures. A statistical analysis was conducted by using the Pearson correlation index. In the second step we have used a data mining task in order to forecast the occurrence of a movement phenomenon by using the \(TDW\) measures. We focused on a traffic jam occurrence.

### 2.1 Related Work

There are some proposals of spatial data warehouses \([29, 50, 39, 55]\), some of these proposals \([29]\) work with a data cube model with spatial and non-spatial dimensions and measures, but none of these proposals work with objects moving in a continuous way in time. The actual research on moving objects could be classified into three areas:

- data modeling and query languages \([41]\)
- standard operators for spatio-temporal aggregation \([37]\)
- implementation of spatio-temporal operators \([61, 58]\)

The modeling of trajectory data can be resumed in proposals for querying positions (past, current and future) of moving objects \([47], [44], [1]\).

A classification and formalization of spatio-temporal aggregations can be found in \([37]\). An aggregate function, when applied over a set of tuples, return a single value. In order to generate the set of tuples to which the operators (aggregate) will be applied, the authors suggest three methods: group composition, partition composition and sliding window composition.
There are some proposals in order to implement spatio-temporal operators. The proposal of [13], presents an idea which uses a combination between indexes and materialization of aggregate measures. The aRB-tree is a structure which is composed by a host index and some measure indexes. The host index is a R-tree which associates the region extents with an aggregated information over all the timestamps of interest. For each entry of the host index, a time-varying aggregate data is defined on a B-tree. But, these types of indexes can present a problem: if an object remains in the query region for several timestamps during the query interval, it could be counted multiple times in the result. This happens because the identifiers of the objects not are stored, just the aggregated information.

A proposal to solve this problem was presented on [61]. The idea is to combine spatio-temporal indexes with sketches, an traditional counting technique based on probabilistic counting [21]. The sketches allows to estimate the number of distinct objects in an area. [61] use an R-tree index in order to manage the regions of interest, and the B-tree records the historical sketches on the corresponding region. That proposal exploits the property that the sketch for the union of several datasets is equal to the $OR$ of the individual sketches of each dataset.

However, in the building of a warehouse for trajectories it is important to exploit the dependency between data. The above works do not take it into account, the spatio-temporal observations are treated as unrelated points.

The recent works extend the traditional Data Warehouse models adding spatio-temporal dimensions in order to try modeling and storing data about trajectories of moving objects. In agreement with [60] the spatial dimensions can be classified into three categories considering the dimension of the hierarchy:

**Non-geometric** This dimension contain only non-geometric (nominal) data and use this configuration to locate a phenomenon in a space. The dimension $city < state < country$ is an example of this type. There are not any associated geometry.

**Geometric-to-non-geometric** The primitive-level data is geometric but whose generalization, starting at a certain high level, becomes non-geometric. An example could be a dimension where a polygon represents a city in a map, in this level the dimension is geometric, however in a high level the dimension can evolute to a non-geometric, the description of the state related to the city, for example.

**Fully geometric** In this classification the primitive level and all of its high-level generalizations are geometric. An example of this type is a dimension whose the finest level is a polygon delimiting a area and the high levels also are geometric types storing the ranges of altitude.

A multidimensional model was defined as a finite set of dimensions and fact table relationships [38]. The authors have proposed a multidimensional model for
the analysis of spatial data detailing three major concepts: Spatial dimensions, Spatial Fact Relationships and Spatial Measures.

A Spatial dimension is a dimension where at least one level is spatial and different spatial data types may be associated with the levels of a hierarchy, the non-spatial dimensions are called thematic. In [60] the spatial dimensions are based on spatial references of hierarchy members. [38] extends the model considering that a dimension can be spatial with only a basic hierarchy. Another characteristic of that model is the possibility of sharing hierarchy among different dimensions.

In a traditional multidimensional model a fact relationship relates leaf members from all dimensions which are involved in the relationship. In a non-spatial multidimensional model it is represented by the relational join operator. Then, [38] define a Spatial Fact Relationships as a fact relationship that requires a spatial join between the spatial dimensions. A object is joined with other object if their geometrics intersect ([48]). The resulting object has the descriptive attributes of both participating objects, its geometry is the intersection of the geometries of the objects which are participating in the join operation.

[38] distinguish two types of spatial measures:

Spatial measures represented by a geometry In this case, the measures are represented by a geometry and a spatial-function must be defined to be used for computing the aggregations along the hierarchies;

Spatial measures as result of spatial computation The measures classified by this type does not to be represented by a geometry, it may be obtained by applying spatial or topological operators.

The definition of the level of granularity of a spatial measure is a issue which can be crucial in a TDW. In a TDW, in most of cases, we do not have the complete set of data representing all positions of the object along the time, it is approximated by using interpolation methods. A possible solution to the problem of granularity could be to define a same spatial measure at different levels of spatial granularity. [14] have proposed a model which allows a measure to represent the location of a fact at multiple levels of spatial granularity.

[65] proposes an approach for enabling spatial data manipulation into OLAP systems. A spatial index mechanism is employed to derive pre-aggregation and materialization of spatial hierarchies, which in turn are leveraged by OLAP system to efficiently answer OLAP queries along spatial hierarchies. However, this model has a very important constraint: it does not support spatial measures.

In [52] the authors propose a logical multidimensional model for a SDW. It is implemented on the top of an object-relational database system with support for spatial data. The database system used in the model allows the storage, spatial aggregation, retrieval and manipulation of spatial data. However it does not support spatial analysis for multidimensional data. The star schema is the base of the model, the authors propose two additional extensions: object-relational concepts
and structures and spatial components. Some spatial OLAP operations can be computed by the concomitant usage of the standard SQL constructions and the Spatial aggregate functions and operators provided by the database system. The authors propose an algorithm which receives the OLAP queries and transform it into aggregate-aware SQL. The algorithm can solve the problems of Spatial aggregate design, Spatial aggregate maintenance and Spatial aggregate exploitation.

A new OLAP language, denoted Spatial OLAP or SOLAP [4], has been defined to meet spatio-temporal analysis needs. The same work discuss the need to integrate the visualization of the spatial component with that of tabular and diagram displays. In [49] the authors have proposed topological and metric operators enhancing the semantics of SOLAP queries.

[29] build a spatial data cube by using a star/snowflake model. They propose the idea of spatial measures with a method to select spatial objects for materialization. In [45] the proposal is to use the classical star schema with spatial dimensions, besides presents methods to process arbitrary aggregations. The work present in [15] introduces a proposal to extend the multi-dimensional data model employed in data warehouses allowing to cope correctly with changes in dimension data: a temporal multi-dimensional data model allows the registration of temporal versions of dimension data. Mappings are provided to transfer data between different temporal versions of the instances of dimensions and enable the system to correctly answer queries spanning multiple periods and thus different versions of dimension data.

In [19] the authors discuss some problems concerning the integration of data in a spatio-temporal data warehouse. The authors argue that data specifications can evolve along the time. Therefore, in this case data sources have temporal, spatial and semantic heterogeneity. The work proposes two approaches to model heterogeneous data in a multidimensional structure. In the first approach an unique temporally integrated cube will be build, this cube contains all the data of all epochs. The second proposal is to create a data mart for each specific view that users want to analyze.

In [13] the authors present an approach of a Spatio-Temporal Data Warehouse. The proposal assumes that the spatial dimension at the finest granularity consists of a set of regions (e.g., road segments in traffic supervision systems, areas covered by cells in mobile communication systems etc.). For each timestamp, the raw data provide the set of objects that fall in each region. (e.g., cars in a road segment, users serviced by a cell). Therefore, in this environment, queries are defined in order to compute aggregate data over regions that satisfy some spatio-temporal condition. According the authors, the main difference between a spatio-temporal and a traditional OLAP is the lack of predefined hierarchies (e.g., product types). Besides, in some environments, the spatial dimensions may be volatile, i.e., the regions at the finest granularity may evolve in time. For instance, the area covered by a cell may change according to weather conditions, extra capacity allocated etc. Those characteristics can be a very important constraint in the development of spatiotemporal data warehouses. The authors propose to solve these problems by
using indexing solutions. Basically, the proposed solution has two steps: the first step involves static spatial dimensions and maintain the focus on queries that ask for aggregated data in a query window over a continuous time interval. An example would be to find the number of objects in a given spatial area during a given time interval. For such queries the proposal is to develop multi-tree indexes that combine the spatial and temporal dimensions. In contrast with traditional OLAP solutions, the index structure is used to define hierarchies. Besides, preaggregated data are stored in internal nodes. Second step is devoted to handle volatile regions. The approach does not aim at simply indexing, but rather replacing the data cube for spatio-temporal data warehouses.

In [53] the authors argue that the fundamental characteristic of the data warehouse technology is its multidimensional paradigm. However, when the goal is to use a spatio-temporal data set, it can not be presented within a conventional multidimensional database. In that work is presented a design of spatio-temporal data warehouse schema in order to include spatiotemporal information and handle computing operations on it. The proposal explores the space-time activity survey and to produce statistics on flexible groups. The authors present an implementation of a spatio-temporal data warehouse by using the traditional snowflake schema. The fact table represents human activity and mobility varying in space and time. Dimension tables are from one hand Space dimension and Time dimension, and from the other hand, the activity of a person. This activity is structured at different levels from the most detailed description (Journey dimension) to its generalisation by trip to the most general description (Person dimension). Each level contains specific attributes that may have a hierarchical description (such as Mode, Activity or Profession). The Space and Time dimensions also define a hierarchy with different levels of granularity. The fact table has trip-duration as its measure which signifies a duration when a person stays in a space unit at a time interval. Hence, if one wants to calculate the exposure indice in a risky zone, just an addition of all durations in this zone is needed. Moreover, if one wants to calculate the person number crossed in a given time and a given space, just the distinct count of all the records within the time and space interval is needed.

2.2 Problem Description

The movement of a spatio-temporal object - i.e., it trajectory - represents the position of the object in a \( d \) - dimensional space \((d \in \{2,3\})\) at a given time instant \(t\).

The movements of the objects can be represented by a finite set of observations, i.e. a finite subset of points taken from the actual continuous trajectory. This finite set is called a sampling. Figure 2.2 shows a trajectory of a moving object in a two dimensional space and a possible sampling of such a trajectory.

Formally, let \(TS\) be a stream of samplings of 2D trajectories \(TS = \{T_i\}_{i \in \{1,...,n\}}\). Each \(T_i\) is the sampling for a simple moving object: \(T_i = (ID_i, L_i)\), where \(L_i = \)}
The model of the data warehouse for trajectories is build in agreement with two basic assumptions:

- The identity of the moving objects and trajectories will be abstract, since we are interested in studying global properties of a set of such objects, e.g., the number of trajectories crossing a given spatio-temporal cell.

- The base cuboid will be composed of spatio-temporal cells, consisting of regions and time intervals which we are interested in.

The classical star schema is used, with three dimensions: two spatial dimensions $X$ and $Y$, and one temporal dimension $T$. The concept hierarchy allows data to be handled at varying levels of abstraction by performing roll-up and drill-down OLAP operations. In this case, we are interested in associating concept hierarchies with
spatial and temporal attributes, it can be carried out by discretizing the corresponding values, resulting in a set-grouping hierarchy dimensional attribute, as illustrated in Figure 2.4.

The spatial cube [26] is built as the lattice of cuboids, where the lowest one (base cuboid) references all the dimensions at the primitive abstraction level, while the others are obtained by summarizing on different subsets of the dimensions, and at different abstraction levels along the concept hierarchy. The component of the base cuboid is denoted by the term base cell, while we simply use cell for a component of a generic cuboid. Figure 2.3 shows the referred star model.

2.4 Aggregate Measures

In a spatio-temporal data warehouse a typical query could be to find the aggregate measure regarding the trajectories falling into a given area and time interval, this measure is any interesting property about the trajectories.

In agreement with to Gray et al. [26] the aggregate functions can be categorized into three classes. These classes characterize the functions with regard to the space complexity of computing a super-aggregate starting from a set of sub-aggregates previously computed.

1. **Distributive.** In this class, a set of sub-aggregates will be computed, then cumulate them by using a suitable function to produce the super-aggregate. Some examples are \( \text{MIN}() \), \( \text{MAX}() \), \( \text{SUM}() \), \( \text{COUNT}() \).
2.4. Aggregate Measures

2. **Algebraic.** The difference with respect the previous function is that, in this case, an $M$-tuple is needed to store the sub-aggregates. For example, the \texttt{AVG()} function needs to store, for each sub-aggregate, a pair made of the \textit{count} and the \textit{sum} computed over the given set of values. The final cumulative function has to add the two components of each sub-aggregates, and then divide the results to produce the global average. Therefore, to compute the super-aggregates of an algebraic functions we need these intermediate results rather than just the raw sub-aggregates, as for distributive functions.

3. **Holistic.** These aggregate functions constitutes a big issues for data warehouses, and when applied to either huge data sets or unbounded ones, like a stream data source, are often computed in an \textit{approximate} way. The problem of holistic functions is that there is no constant $M$, such that an $M$-tuple can be used to store the results of each sub-aggregation. Examples of these functions are \texttt{MEDIAN()}, \texttt{RANK\_SELECTION()}, \texttt{MOST\_FREQUENT()}, \texttt{QUANTILES()}, \texttt{COUNT\_DISTINCT()} etc.

In this work we are interested in studying an aggregated function called \textit{Presence}. This function represents the count of the \textit{distinct} trajectories crossing a given cell. Since this function has to deal with the issues related to counting \textit{distinct} trajectories, it is a sort of \texttt{COUNT\_DISTINCT()} aggregate, and thus a \textit{holistic} one. We will show how to exploit alternative and \textit{non-holistic} aggregate functions to compute \textit{Presence} values that \textit{approximate} to the exact ones. These alternative functions only need a few/constant memory size for maintaining the information – i.e., the $M$-tuple – to be associated with each base cell of our data warehouse, from which we can start to compute a super-aggregate. The two approximate functions we will consider are the following:

1. **Presence\textsubscript{Distributive}:** We assume that the only measure associated with each base cell is the exact (or approximate) count of all the \textit{distinct} trajectories crossing the cell. Therefore, the super-aggregate corresponding to a roll-up operation is simply obtained by summing up all the measures associated with cell. This aggregate function may produce very inexact approximation of the true \textit{Presence}. Because we may count multiple times the same trajectory. We do not have enough information in the base cell that permit us to perform a \textit{count distinct} when rolling-up.

2. **Presence\textsubscript{Algebraic}:** Each base cell stores an $M$-tuple of measures. One of these is the exact (or approximate) count of all the \textit{distinct} trajectories touching the cell. The other measures are used when we compute the super-aggregate, in order to correct the errors introduces by function \textit{Presence\textsubscript{Distributive}} due to the duplicated count of trajectory presences.
More formally, let $C_{x,y,t}$ be a generic base cell of our cuboid, where $x$, $y$, and $t$ identify intervals of the form $[l, u)$, in which we have subdivided the spatial and temporal dimensions. The associated measures are thus $C_{x,y,t}.presence$, $C_{x,y,t}.crossX$, $C_{x,y,t}.crossY$, and $C_{x,y,t}.crossT$.

$C_{x,y,t}.presence$ is the count of all the distinct trajectories crossing the cell.

$C_{x,y,t}.crossX$ is the number of distinct trajectories crossing the spatial border between $C_{x,y,t}$ and $C_{x+1,y,t}$.

$C_{x,y,t}.crossY$ is the number of distinct trajectories crossing the spatial border between $C_{x,y,t}$ and $C_{x,y+1,t}$.

Finally, $C_{x,y,t}.crossT$ is the number of distinct trajectories crossing the temporal border between $C_{x,y,t}$ and $C_{x,y,t+1}$.

In order to compute the super-aggregate corresponding to two adjacent cells with respect to a given dimension, namely $C_{x',y',t'} = C_{x,y,t} \cup C_{x+1,y,t}$, we can compute it as follows:

\[
\text{Presence}_{\text{Algebraic}}(C_{x,y,t} \cup C_{x+1,y,t}) = \]
\[
= C_{x',y',t'.presence} = \]
\[
= C_{x,y,t}.presence + C_{x+1,y,t}.presence - C_{x,y,t}.crossX
\]

Moreover, if we need to update the other measures associated with the $C_{x',y',t'}$ for subsequent aggregations, we have:

\[
C_{x',y',t'.crossX} = C_{x+1,y,t}.crossX \]
\[
C_{x',y',t'.crossY} = C_{x,y,t}.crossY + C_{x+1,y,t}.crossY \]
\[
C_{x',y',t'.crossT} = C_{x,y,t}.crossT + C_{x+1,y,t}.crossT
\]

Note that Equation 2.1 can be thought as a simple application of the well know Inclusion/Exclusion (IE) principle: given two sets $A$ and $B$, we have that $|A \cup B| = |A| + |B| - |A \cap B|$. Suppose that the elements of $A$ and $B$ are just the distinct trajectories occurring in cells $C_{x,y,t}$ and $C_{x+1,y,t}$ respectively. Hence, their cardinalities $|A|$ and $|B|$ exactly correspond to $C_{x,y,t}.presence$ and $C_{x+1,y,t}.presence$. Moreover, $|A \cap B|$ should correspond to $C_{x',y',t'.crossX}$. Unfortunately, in some cases $C_{x,y,t}.crossX \cong |A \cap B|$, and this may introduce errors in the values returned by $\text{Presence}_{\text{Algebraic}}$. Figure 2.5 shows a trajectory that will be correctly counted, since it crosses the border between the two cells to be rolled-up. Conversely, Figure 2.6 shows a very agile and fast trajectory, which will be counted two times during the roll-up, since it is not considered in $C_{x,y,t}.crossX$, even if it should appear in $|A \cap B|$. In fact, the trajectory touches both cells $C_{x,y,t}$ and $C_{x+1,y,t}$, but does not cross the border between the two cells.
2.5 Load Phase

The feeding of the data warehouse is a very important step in order to obtain correct results using aggregate functions. The feeding begins at the base cells of the base cuboid, with suitable sub-aggregate measures, from which starting to compute super-aggregated functions. A stream of trajectory observations can arrive at different rates and in an unpredictable and unbounded way. In this manner, is necessary to limit the amount of buffer memory needed to store information about active trajectories. The system module that is responsible for feeding data can consider a trajectory as inactive when, for a long time interval, no further observations relative to a given object have been received. The inactive trajectories could be removed from the buffer, it is a manner to free memory space.

Some examples of sub-aggregate numeric measures to be stored in each base cell
are:

1. number of observations;
2. number of trajectories, i.e., our Presence;
3. total distance covered by trajectories;
4. number of trajectories that covered a distance larger than a given $d$;
5. number of trajectories that followed a closed path, i.e., which started and ended on the same point (with tolerance $d$).

Some measures require a pre-computation, and can be updated in the data warehouse as soon as single observations of the various trajectories arrive. We suggest a classification, ordering the measures according to an increasing amount of pre-computation effort:

a) *no pre-computation:* the measure can be updated in the data warehouse by directly using each single observation;

b) *per trajectory local pre-computation:* the measure can be updated by exploiting a simple pre-computation, which only involves a few and close observations of the same trajectory;

c) *per trajectory global pre-computation:* the measure update requires a pre-computation which considers all the observations of each trajectory;

d) *global pre-computation:* it is required the complete re-computation of the aggregate on all the input raw data / trajectories. Thus, these measure cannot be treated and computed by using a data warehouse.

Referring to the above examples, measure 1 is of type (a), while measure 2 is of type (b), even if in the experimental part we will try to threat it as a measure of type (a), thus introducing several errors in the sub-aggregates stored in the base cell of our data warehouse. Finally, measure 3 is of type (b), and measures 4 and 5 are of type (c).

The amount of pre-computation associated with each type of measure impact upon the amount of memory required to buffer incoming trajectory observations. Measures of type (a) are the less expensive in terms of space and time, since it is enough to consider observations one at a time, without buffering anything. Therefore a measure can be updated as soon as each single observation $L^j_i = (x^j_i, y^j_i, t^j_i)$ of the various trajectories arrives. Conversely, for type (b) the measure must be computed starting from a finite set of neighbors of each observation $L^j_i = (x^j_i, y^j_i, t^j_i)$. For example, this could require a $k$-window of observations $L^j_{i-k-1}, \ldots, L^j_i$ to be considered and stored in a buffer.
2.5. Load Phase

If we need to interpolate the observations to store correct measures in the base cells of our data warehouse, the aggregate measure we plan to compute should be, at least, of type (b). As a matter of fact, when the observation \( L_i^j = (x_i^j, y_i^j, t_i^j) \) arrives, in order to interpolate, thus inferring the trajectory route, we have to maintain available in the buffer the previous trajectory observation \( L_i^{j-1} = (x_i^{j-1}, y_i^{j-1}, t_i^{j-1}) \).

Given the observations for a trajectory shown in Figure 2.2, a possible reconstructed trajectory using linear interpolation is shown in Figure 2.7, where we also illustrate the discretized 2D space.

If we updated the data warehouse on the basis of each single observation, the measures \((M_1, \ldots, M_k)\), possibly corresponding to the four observations of our example, could naturally be stored in the following cells:

<table>
<thead>
<tr>
<th>Time label</th>
<th>X</th>
<th>Y</th>
<th>T</th>
<th>( M_1 )</th>
<th>\ldots</th>
<th>( M_k )</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>[30, 60)</td>
<td>[30, 60)</td>
<td>(0.30)</td>
<td>\ldots</td>
<td>\ldots</td>
<td>\ldots</td>
</tr>
<tr>
<td>65</td>
<td>[60, 90)</td>
<td>[30, 60)</td>
<td>[60, 90)</td>
<td>\ldots</td>
<td>\ldots</td>
<td>\ldots</td>
</tr>
<tr>
<td>75</td>
<td>[90, 120)</td>
<td>[90, 120)</td>
<td>[60, 90)</td>
<td>\ldots</td>
<td>\ldots</td>
<td>\ldots</td>
</tr>
<tr>
<td>120</td>
<td>[120, 150)</td>
<td>[90, 120)</td>
<td>[60, 120)</td>
<td>\ldots</td>
<td>\ldots</td>
<td>\ldots</td>
</tr>
</tbody>
</table>

Table 2.1: Cells representation - for each observation

This approach will be correct only if consider measure is of type (a), otherwise several errors will be introduced. Other cells might be crossed by the moving object, without reconstructing the full trajectory and adding further intermediate points, we are not able to store any information about the trajectory in these other cells.

<table>
<thead>
<tr>
<th>((t_i, t_{i+1}))</th>
<th>X</th>
<th>Y</th>
<th>T</th>
<th>( M_1 )</th>
<th>\ldots</th>
<th>( M_k )</th>
</tr>
</thead>
<tbody>
<tr>
<td>(10, 30)</td>
<td>[30, 60)</td>
<td>[30, 60)</td>
<td>(0.30)</td>
<td>\ldots</td>
<td>\ldots</td>
<td>\ldots</td>
</tr>
<tr>
<td>(30, 32)</td>
<td>[30, 60)</td>
<td>[30, 60)</td>
<td>[30, 60)</td>
<td>\ldots</td>
<td>\ldots</td>
<td>\ldots</td>
</tr>
<tr>
<td>(32, 60)</td>
<td>[90, 120)</td>
<td>[30, 60)</td>
<td>[30, 60)</td>
<td>\ldots</td>
<td>\ldots</td>
<td>\ldots</td>
</tr>
<tr>
<td>(60, 65)</td>
<td>[90, 120)</td>
<td>[30, 60)</td>
<td>[60, 90)</td>
<td>\ldots</td>
<td>\ldots</td>
<td>\ldots</td>
</tr>
<tr>
<td>(65, 67)</td>
<td>[90, 120)</td>
<td>[30, 60)</td>
<td>[60, 90)</td>
<td>\ldots</td>
<td>\ldots</td>
<td>\ldots</td>
</tr>
<tr>
<td>(67, 70)</td>
<td>[90, 120)</td>
<td>[90, 120)</td>
<td>[60, 90)</td>
<td>\ldots</td>
<td>\ldots</td>
<td>\ldots</td>
</tr>
<tr>
<td>(70, 73)</td>
<td>[120, 150)</td>
<td>[90, 120)</td>
<td>[60, 90)</td>
<td>\ldots</td>
<td>\ldots</td>
<td>\ldots</td>
</tr>
<tr>
<td>(73, 75)</td>
<td>[120, 150)</td>
<td>[120, 150)</td>
<td>[60, 90)</td>
<td>\ldots</td>
<td>\ldots</td>
<td>\ldots</td>
</tr>
<tr>
<td>(75, 90)</td>
<td>[120, 150)</td>
<td>[120, 150)</td>
<td>[60, 90)</td>
<td>\ldots</td>
<td>\ldots</td>
<td>\ldots</td>
</tr>
<tr>
<td>(90, 99)</td>
<td>[120, 150)</td>
<td>[120, 150)</td>
<td>[90, 120)</td>
<td>\ldots</td>
<td>\ldots</td>
<td>\ldots</td>
</tr>
<tr>
<td>(99, 120)</td>
<td>[150, 180)</td>
<td>[120, 150)</td>
<td>[90, 120)</td>
<td>\ldots</td>
<td>\ldots</td>
<td>\ldots</td>
</tr>
</tbody>
</table>

Table 2.2: Sequence of segments composing the interpolated trajectory, and the base cells that completely include each segment.

On the basis of the linearly interpolated trajectory of Figure 2.7, we can thus add further interpolated points for each cell crossed by a trajectory. We add points that cross the borders of each crossed cell, considering all its three dimensions. The choice
of adding all such intermediate points simplifies the computation of several measures to associate with each base cell. Figure 2.8 shows the resulting interpolated points as white and gray circles. The white interpolated points, associated with temporal labels 30, 60, and 90, have been added to match the granularity of the temporal dimension. In fact, they correspond to cross points of a temporal border of some 3D cell. The gray points, labeled with 32, 67, 70, 73, and 99, have been instead introduced to match the spatial dimensions. They correspond to the cross points of the spatial borders of some 3D cell, or, equivalently, the cross points of the spatial 2D squares shown in Figure 2.8.

After the including of these additional interpolated points, we have further 3D base cells in which we can now store significant measures associated with the trajectory of the given object. The new points subdivide the interpolated trajectory into small segments, each one completely included in some 3D base cell. Therefore we can now update a cell measure on the basis of a single trajectory segment. The Table 2.2 shows the sequence of edges composing the interpolated trajectory of Figure 2.8 and the base cell which the edge belongs to. For the sake of simplicity, each edge is identified by a pair of timestamps \((t_i, t_{i+1})\), associated with the starting and ending points.

2.6 Presence Measure

On Section 2.4 we defined two approximate aggregate functions we would like to exploit in order to approximate to the holistic aggregate Presence, namely Presence\(_{\text{Distributive}}\) and Presence\(_{\text{Algebraic}}\), of Presence, defined as as the number of distinct trajectories present in a given spatio-temporal cell. In order to update the sub-aggregate measures stored in each base cell \(C_{x,y,t}\), namely \(C_{x,y,t}^{\text{presence}}, C_{x,y,t}^{\text{crossX}}, C_{x,y,t}^{\text{crossY}},\) and \(C_{x,y,t}^{\text{crossT}}\), we have different options.

1. **single** observations (type a). In this case we can only update/increment a measure \(C_{x,y,t}^{\text{presence}}\). Since we do not use any buffer, we cannot remember the previous points of each trajectory. Thus in this case we cannot update \(C_{x,y,t}^{\text{crossX}}, C_{x,y,t}^{\text{crossY}},\) and \(C_{x,y,t}^{\text{crossT}}\).

2. a **pair** of observations (type b). In particular, the currently received observation \(L_i^t\) of trajectory \(T_i\), along with the previous buffered \(L_{i-1}^t\) one. Using this pair of points, we can linearly interpolate the trajectory. If we buffer non only the previous observation \(L_{i-1}^t\), but also the last \(C_{x,y,t}^{\text{presence}}\) that was modified on the basis of \(L_{i-1}^t\), we can avoid most of the duplicate presence counts of trajectories when the two consecutive points fall in the same base cell.

Moreover, by exploiting linear interpolation, we can also identify the cross
2.6. Presence Measure

points of each base cell, and can accordingly update the various \(C_{x,y,t}.crossX\), \(C_{x,y,t}.crossY\), and \(C_{x,y,t}.crossT\).

3. a window of \(k\) observations (type b). In particular, let \(L_j^i = (x_j^i, y_j^i, t_j^i)\) be the currently received observation of trajectory \(T_i\). The \(k\) window thus includes \(L_{j-k-1}^i, L_{j-k-2}^i, \ldots, L_{j-1}^i, L_j^i\). The window size \(k\) is dynamically adjusted according to the following constraints: (1) All \(t_{j-k-1} < \ldots < t_j < u\) must fall within the same temporal interval \([l, u)\), characterizing a base cells of our cuboid. More formally, \(l \leq t_{j-k-1} < t_{j-k-2} < \ldots < t_j < u\). (2) In addition, \(L_{j-k}^i\) must not be included in the window, because \(t_{j-k} < l\).

Buffering all these points (and some related information) guarantees the linear interpolation of the associated trajectory, and permits us to completely avoid duplicates when we update an aggregate measure \(C_{x,y,t}.presence\). To this end, it is enough to maintain the information about which are the cells whose presence measures have been updated on the basis of the window points. It is straightforward to show that if we encounter a new observation \(L_{j+1}^i\), such that \(t_{j+1} \geq u\), we can forget (un-buffer) all the points of the window, since we are surely going to update new cells, associated with a different and successive temporal interval.

Moreover, using the window of \(k\) points, we can also update \(C_{x,y,t}.crossX\), \(C_{x,y,t}.crossY\), and \(C_{x,y,t}.crossT\) without duplicates. For example, think about a trajectory that, in the same base time interval, quickly goes ahead and back, crossing multiple times the same spatial border of two cells. By maintaining information about which measures have been updated on the basis of the window points, we can avoid these duplicate counts of the same crossing trajectory.

Given the three methods for loading the base cell measures, the corresponding aggregate functions are:

1. single observations: We can use this loading method for approximating the presence measures associated with each base cell. These approximate sub-aggregates can then be used by our distributive aggregate function, denoted as \(Presence_{1Distributive}\). It is worth noting that we cannot exploit our aggregate algebraic function (\(Presence_{1Algebraic}\)). This is because, using only single observations, we cannot compute the counts of the trajectories crossing the borders of the base cells.

2. observation pairs: We can use this loading method for approximating all the needed measures associated with each base cell, both presence and cross counts. These approximate sub-aggregates can then be used by our distributive/algebraic aggregate functions, denoted as \(Presence_{2Distributive}\) and \(Presence_{2Algebraic}\).
3. **Observation windows**: We can use this loading method for exactly computing all the needed measures associated with each base cell, both presence and cross counts. These exact sub-aggregates can then be used by our distributive/algebraic aggregate functions, denoted as $\text{Presence}^k_{\text{Distributive}}$ and $\text{Presence}^k_{\text{Algebraic}}$.

In the prototype we have used the method *pair of observations* in order to update the sub-aggregate measures for each cell.

### 2.7 The Prototype

We developed a prototype in order to implement the proposed *TDW*. The prototype can implement a *TDW* in agreement with the concepts presented in the sections above. Besides, the computation of aggregated measures and a mechanism to compute roll-up operations were implemented in the prototype. The *TDW* was populated by using the synthetic datasets generated by the traffic simulator described in [8]. The measures stored in the *TDW* can be used to discover interesting phenomena of the trajectories. The application tries to solve the both problems: loading and aggregation, which were presented in Sections 2.5 and 2.4.

![Figure 2.9: Results Interface](image)

Figure 2.9 shows the interface to visualize the values of the measures computed considering a cell selected by the user. The result presents the evolution of the values of measures in a range of time. In the same visualization is possible to define different
values of roll-up operations. The roll-up operation can be defined by the usage of the slider controls over the map, a more detailed explanation will be presented in the next sections.

The TDW was implemented in a traditional data warehouse tool. We used the MS SQL SERVER 2005. The TDW was modeled in agreement with the star model [34]. We defined a fact table and three dimension tables (X and Y spatial dimensions and T temporal dimension). Tables 2.3, 2.4 and 2.5 present the structure of these tables (fact and dimensions). The level of granularity of the trajectory data warehouse can be defined in the prototype. It allows to control the loading of the TDW and computation of the measures and aggregations. Each tuple stored in the fact table represents a summarization of the measures that are delimited by the borders of the cell. The base cell are delimited by the tid, xid and yid values. The measures presented in Table 2.3 are detailed in Sections 2.5 and 2.4. The measures presence, xborder, yborder and tborder are necessary in order to compute the holistic presence measure. These measures are specially important when it is necessary to compute the roll-up operations.

Table 2.3: Fact Table

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tid</td>
<td>time foreign key</td>
</tr>
<tr>
<td>xid</td>
<td>X spatial foreign key</td>
</tr>
<tr>
<td>yid</td>
<td>Y spatial foreign key</td>
</tr>
<tr>
<td>numobs</td>
<td>Number of observations in the cell</td>
</tr>
<tr>
<td>trajinit</td>
<td>Number of trajectories starting in the cell</td>
</tr>
<tr>
<td>vmax</td>
<td>Maximum speed of trajectories in the cell</td>
</tr>
<tr>
<td>distance</td>
<td>Total distance covered by trajectories in the cell</td>
</tr>
<tr>
<td>time</td>
<td>Total time spend by the trajectories in the cell</td>
</tr>
<tr>
<td>presence</td>
<td>Number of trajectories in the cell - distributive</td>
</tr>
<tr>
<td>xborder</td>
<td>Number of trajectories crossing the x cell border</td>
</tr>
<tr>
<td>yborder</td>
<td>Number of trajectories crossing the y cell border</td>
</tr>
<tr>
<td>tborder</td>
<td>Number of trajectories crossing the t cell border</td>
</tr>
<tr>
<td>speed</td>
<td>Average speed of trajectories in the cell</td>
</tr>
</tbody>
</table>

Table 2.4: X or Y Dimension Table

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>xid</td>
<td>primary key</td>
</tr>
<tr>
<td>xl1</td>
<td>first level of hierarchy</td>
</tr>
<tr>
<td>xl2</td>
<td>second level of hierarchy</td>
</tr>
</tbody>
</table>

Table 2.5: T Dimension Table

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tid</td>
<td>primary key</td>
</tr>
<tr>
<td>tl1</td>
<td>first level of hierarchy</td>
</tr>
<tr>
<td>tl2</td>
<td>second level of hierarchy</td>
</tr>
</tbody>
</table>

The prototype allows to set the environment in order to receive the data volume. It is done loading the data volume into a buffer table (see Table 2.6). Considering that the trajectory data are produced in an unpredictable and unbounded way, we have to store data into a buffer table. This procedure allows to release space in the buffer table. It can be done by the exclusion of the tuples of the ended trajectories.
Table 2.6: Buffer Table

<table>
<thead>
<tr>
<th>oid</th>
<th>Object Identifier</th>
</tr>
</thead>
<tbody>
<tr>
<td>xvalue</td>
<td>X spatial value</td>
</tr>
<tr>
<td>yvalue</td>
<td>Y spatial value</td>
</tr>
<tr>
<td>tvalue</td>
<td>T time value</td>
</tr>
<tr>
<td>dift</td>
<td>Time variation between two consecutive positions</td>
</tr>
<tr>
<td>difx</td>
<td>X spatial variation between two consecutive positions</td>
</tr>
<tr>
<td>dify</td>
<td>Y spatial variation between two consecutive positions</td>
</tr>
<tr>
<td>dist</td>
<td>Distance covered between two consecutive positions</td>
</tr>
<tr>
<td>vel</td>
<td>Speed between two consecutive positions</td>
</tr>
<tr>
<td>idrow</td>
<td>Identifier of the row</td>
</tr>
<tr>
<td>timestamp</td>
<td>Timestamp of the observation</td>
</tr>
</tbody>
</table>

Through the loader component it is possible to define some characteristics of the environment and to compute the interpolation procedure. In order to compute the interpolation and to load the $TDW$ two parameters must be defined: Granularity level and Dimension hierarchical level.

The definition of the granularity level is necessary to build the regular grid which divides the spatio-temporal environment. This procedure is done before the loading the $TDW$. After the definition of the granularity level it is possible to define the hierarchy level of the dimension tables. This procedure also can be done by the loader component.

Figure 2.10: Loading Settings

Figure 2.10 shows a visualization of the interface available to define those settings. These procedures are just executed once, before the beginning of the loading the data warehouse. After the definition of the settings explained above, the inter-
2.7. The Prototype

face allows to start the process for receiving the data stream values and loading the TDW.

Algorithm 1 presents the basic procedures to complete the loader process. \( C_{\text{cur}} \) represents the current base cell, \( C_{\text{prev}} \) the previous base cell stored in the buffer related to the same trajectory, and \( IP \) represents the set of base cells computed by the interpolation process. Using the setting values already defined, the loader component gets the active values in the buffer table and performs the procedures to load the TDW.

Algorithm 1 Loading

Input: \{Stream SO of observations - id, x, y, t\}

Output: \{Fact Table FT\}

\[ FT \leftarrow \emptyset \]

\[ \text{buffer} \leftarrow \emptyset \]

repeat

\[ \text{obs} \leftarrow \text{next}(SO) \]

\[ C_{\text{cur}} \leftarrow \text{findCell}(\text{obs.x, obs.y, obs.t}) \]

if \( \text{obs.id} \notin \text{buffer} \) then

\[ \text{insertbuffer}(\text{obs.id}) \]

else

\[ C_{\text{prev}} \leftarrow \text{findCell}(\text{obs.x, obs.y, obs.t}) \]

\[ IP \leftarrow \text{interp}(C_{\text{cur}}, C_{\text{prev}}) \]

\[ ct \leftarrow 1 \]

end if

repeat

if \( IP[ct] \notin FT \) then

\[ \text{insert}(IP[ct], FT) \]

else

\[ \text{update}(IP[ct], FT) \]

end if

\[ ct \leftarrow ct + 1 \]

until \( ct < IP.numpoint \)

We described the concept of interpolation (see Section 2.5) used in order to load the TDW. The buffer table is used in order to implement that procedure. For each active row of the buffer table the application has to find the related cell in the TDW. If exists a row in the fact table for that cell, the values of distributive measures (e.g numobs, trajinit) can be updated, otherwise a new row will be inserted in the fact table. It is the procedure when is not necessary to compute the interpolation.

When is necessary to compute the interpolation, the procedure must use the identifier of the active trajectories, their last processed point, the cell such point belongs to, and the speed in the segment ending at such a point. Using that set of values it is possible to determine the additional points in order to represent the
intersections among the trajectory and the borders of the cells. The additional points can be computed considering the constant speed of the trajectory and the spatio-temporal granularity.

In Algorithm 1, the function \textit{interp} computes the additional points of the interpolation process. For each new point computed by the interpolation process it is executed a search to find the related \textit{cell} in the \textit{TDW}. Whether there is the base \textit{cell} in the \textit{fact table}, the related measures must be updated. Otherwise, the procedure is to insert a tuple with the new values of the measures.

In the prototype the pre-aggregated values are stored in the \textit{TDW}. These pre-aggregated values are computed in the \textit{loading} phase. However, when is necessary to compute some query or to perform a roll-up operation, the application uses the \textit{aggregation component}. Figure 2.9 shows the results of a query defined by the boundaries of the cell selected on the map. The results are represented by the charts (right side) where it is possible to verify the evolution of the measures for each value of time dimension. The user can choose the query \textit{base cell} either by the usage of the combo-boxes or by clicking on the map. The map is divided into a regular grid, this division is done in agreement with the granularity defined by the user.

2.7.1 Assessments of the Prototype

In this section we present a report of the experiments in order to evaluate the \textit{TDW} prototype. We conducted tests to measure the time taken for loading the \textit{TDW} Data Cube, and to measure the accuracy of the roll-up operations.

We conducted the tests using a \textit{TDW} defined with three different values of spatio-temporal granularity. The granularity was defined considering the length of the borders of the spatio-temporal cell. Table 2.7 presents the length values for each border of spatio-temporal cell.

<table>
<thead>
<tr>
<th>Granularity</th>
<th>Dimensions Length</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>( X )</td>
</tr>
<tr>
<td>125 G</td>
<td>125000</td>
</tr>
<tr>
<td>250 G</td>
<td>250000</td>
</tr>
<tr>
<td>500 G</td>
<td>500000</td>
</tr>
</tbody>
</table>

For each level of granularity we computed roll-up operations (three levels: 1g, 2g and 3g) on the measure \textit{presence}. We used traditional roll-up operations (summing up the sub-aggregate values) and our proposed model (algebraic \textit{presence}). Each result obtained was compared with the correct values. We consider the number of distinct trajectories crossing the area as the correct value.
\[ \text{Error} = 100 \times \frac{\|C - R\|}{\|R\|} \]  

(2.2)

The error level was computed by using Equation 2.2. \(C\) represents the computed value and \(R\) the real value of distinct trajectories crossing the area.

Figures 2.11, 2.12 and 2.13 present the error values for each granularity along the roll-up levels. For all levels of granularity (125, 250 and 500) our proposal presents the lower level of error.

Figure 2.11: Granularity 125

Figure 2.12: Granularity 250

Also it is possible to verify that the level of error of our proposal remains almost stable along the roll-up level (1g, 2g and 3g). The error level of the distributive model decreases while the level of granularity increases. It can be explained by the
fact that the number of trajectories crossing a base cell decreases in agreement with the growth of the granularity level.

In order to measure the time taken to load the cube, we simulated several data volumes of stream observations, with different number of observations for each of them. For each data volume, it was verified the ratio between the time taken to load the observations into the cube, and the number of observations. In the loading we consider the loading of the buffer table, the computation of the interpolated points, the computation of the sub-aggregate measures and the consequent loading to the data cube.

\[ T_{\text{obs}} = \frac{T_{\text{tot}}}{N_{\text{obs}}} \]  

(2.3)

\[ Thr = \frac{1}{T_{\text{obs}}} \]  

(2.4)

The time taken to load the cube, by observation, was computed by Equation 2.3. \( T_{\text{tot}} \) represents the total time considering the receiving, interpolation and computation of the sub aggregates values. \( N_{\text{obs}} \) represents the number of observations loaded. The throughput (\( Thr \)) is computed by Equation 2.4 and it represents the number of observations loaded into the cube by second.

Figure 2.14 presents the results of this test expressed in \textit{seconds / number of observations}. We considered several data sets composed by 21017, 39208, 62219, 107865 and 156113 observations. The time taken to load the cube is stable, it remains in the same level (\( T_{\text{obs}} \approx 0.005 \text{ seconds/observation} \cdot Thr \approx 200 \)) for all the different packages of observations.
2.8 TDW: Events Occurrences Patterns

In this section we perform some statistical analysis on the data stored in the various cells, to discover correlation in such data. The statistical analysis was conducted by using the Pearson correlation index. Besides, we investigate the use of traditional data mining techniques to reveal patterns of occurrences of events. We focus on a traffic jam occurrence. It was done adding supervised knowledge to data records associated with cells, by labeling them with respect to the occurrence of a traffic jam phenomena, and then investigating whether it is possible to extract good (classification) models from all the labeled records associated with the cells. The goal is to verify whether it is possible conclude something about the occurrence of a TJ in a spatio-temporal cell by only looking at the measures stored in the cells.

We consider a set of objects moving in a network, abstracted by an undirected graph. The area is divided into a regular grid, composed of a set of cells delimited by (x,y,t) spatio-temporal coordinates. In this environment a singular cell may involve several different edges, each of them presenting different characteristics. For example, consider a given cell involving six edges: three edges are considered crowded, while the three other ones are not crowded. In this case, what is the most adequate representation of a TJ occurrence for that cell? In the following section we present a proposal to solve this problem. We argue that by using this model (an area divided into a regular grid, composed of a set cells) also is possible to individualize singular edges. It can be done defining an adequate level of granularity of the cells. In this
way it is possible to investigate movement occurrences at individual edges and in an area composed of different edges. We expect to obtain best results of precision in the analysis of the events occurrences patterns by using low levels of granularity. High levels of granularity may involve a high number of different edges (with different characteristics), therefore the accuracy of the results tends to decrease.

The most part of the section itself is concerned with the labeling of the cells. It was done because the use of a simulator to generate the dataset. By using a simulator, it is possible to manipulate some variables in order to generate an environment. Therefore, the cells were labeled considering the results of the simulator. In real environments, maybe it is possible to know from external sources whether an event (a Traffic Jam in this case) occurred (or not) in a given cell. In the following we present some details about the experiments and the obtained results.

### 2.8.1 Traffic Jam Patterns on TDW

We are interested in the investigation of the use of the aggregate measures stored in our TDW to discover the occurrence of specific phenomena. Actually, in this section we consider a Traffic Jam occurrence. We have used the fact table rows in order to mine patterns revealing TJ. Table 2.11 shows the measures that we have considered to complete this experiment. Each fact table row stores the summarized values of measures for a given cell \((x,y,t)\).

Our goal is to use a classification mining task in order to extract a classification model able to reveal TJ occurrences. The TDW fact table rows were labeled to represent the occurrence of a TJ phenomenon for each cell \((x,y,t)\).

In order to solve the problem introduced in the last section (a same cell composed of several different edges) we have proposed to measure the TJ occurrence with a value that represents the density of TJ occurrences for each cell. Equation 2.5 was defined in order to compute the TJ density value \(dTJ_{x,y,t}\) for each cell \((x,y,t)\). \(NedTJ_{x,y,t}\) represents the number of edges labeled with Crowded=Yes for a given cell \((x,y,t)\), and \(Ned_{x,y,t}\) represents the number of edges involved by cell \((x,y,t)\).

\[
dTJ_{x,y,t} = \frac{NedTJ_{x,y,t}}{Ned_{x,y,t}} \tag{2.5}
\]

The value of \(dTJ_{x,y,t}\) represents the probability of the occurrence of a traffic jam phenomenon in a given cell \((x,y,t)\). The measure \(dTJ_{x,y,t}\) is not available in the TDW. We have used it to label TDW cells, thus preparing a training set for the classification method. The goal of the experiment is to confirm our proposal to identify a serious Traffic Jam phenomenon in our TDW:

*A traffic jam occurrence can be identified by the simultaneous occurrence of the two conditions in the measures stored in the TDW:*
1. the speed of an object in a given edge at a given timestamp value is considerably smaller than the minimum value between the maximum speed of the object and the maximum speed allowed in the edge.

2. the number of objects in a given edge at a given timestamp value is very close, or higher than the maximum capacity of the edge.

The above items 1 and 2 can be represented by the two new measures to represent the context of a given cell: level of speed \((lSp_{x,y,t})\) and density of presence \((dPres_{x,y,t})\). These new measures can be computed on the basis of those stored in the TDW along with the statically available measures \((SEdge\) and \(WAVGSp\)).

\[
dPres_{x,y,t} = \frac{Pres_{x,y,t}}{SEdge_{x,y,t}} \quad (2.6)
\]

\[
lSp_{x,y,t} = \frac{AVGSp_{x,y,t}}{WAVGSp_{x,y,t}} \quad (2.7)
\]

Density of presence (Equation 2.6) represents the ratio between the number of distinct trajectories crossing a cell \((Pres_{x,y,t})\) and the sum of capacities of the edges for each cell \((SEdge_{x,y,t})\). Level of speed (Equation 2.7) represents the ratio between the average speed of the trajectories crossing the cell \((AVGSp_{x,y,t})\) and the value \(WAVGSp_{x,y,t}\). Value \(WAVGSp_{x,y,t}\) represents the average among the values of maximum speed defined for each edge (involved by cell).

### 2.8.2 Pre-processing of the TDW data

We have generated a synthetic dataset by using Brinkhoff - Trajectory Generator \[8\]. The dataset simulates the occurrence of several traffic jam phenomena. The generator uses a discrete time model: the whole period is divided into \(n\) timestamps. At each timestamp, new moving objects are generated and existing objects are moved or are deleted because they have reached their destination. Each moving object belongs to a class that specifies the behavior of the object. For example, the (maximum) speed is defined by such a class. Each edge of the network belongs to an edge class which defines the speed limit and the capacity of an edge. If the number of objects traversing an edge at a timestamp exceeds the specified capacity, the speed limit on this edge will be decreased. The traffic jam occurrence can be simulated considering that if the number of vehicles using a street exceeds a threshold, the maximum and the average speed of the objects reduce.

We have generated a synthetic dataset simulating some occurrences of traffic jams at edges. The datasets was generated to represent different environments, it was done considering different hours of the day, different number of the objects, etc. The dataset was generated based on the road network of the City of Oldenburg, which contains of 6105 nodes and 7035 edges. In the following a briefly description (and values) for each characteristic of the synthetic dataset:
• **73520 Moving Objects** is the number of objects traveling on the network;
• **657281 Points** represents the number of positions reported in the dataset;
• **2592445 Traversed Nodes** is the number of nodes traversed by objects traveling on the network;
• **74483 Routes** represents the number of routes defined in the simulation;
• **5789 AVG route length** represents the average length of the routes.

For generated dataset we have studied the logs created by the simulator. We have modified the simulator to include new attributes into the original log: *Edge id, Edge Capacity* and *Maximum Speed*. It was done in order to investigate the traffic jam occurrence. The synthetic dataset was generated considering that a traffic jam happens when the number of objects traversing an edge at a timestamp exceeds its capacity. Therefore, by using the additional information (*edge id, edge capacity, maximum speed*) we can identify an occurrence of traffic jam: where (edge) and when (timestamp) it occurs. Table 2.8 presents the attributes of the log of the generated dataset.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Status</td>
<td>Status of point</td>
</tr>
<tr>
<td>Id</td>
<td>Identifier of point</td>
</tr>
<tr>
<td>Seq</td>
<td>Sequence number</td>
</tr>
<tr>
<td>Class Object Id</td>
<td>Identifier of objects class</td>
</tr>
<tr>
<td>Time</td>
<td>Timestamp</td>
</tr>
<tr>
<td>X</td>
<td>X-coordinate</td>
</tr>
<tr>
<td>Y</td>
<td>Y-coordinate</td>
</tr>
<tr>
<td>Speed</td>
<td>Current speed of object</td>
</tr>
<tr>
<td>Edge Id</td>
<td>Identifier of the edge</td>
</tr>
<tr>
<td>Edge Capacity</td>
<td>Max. capacity of objects</td>
</tr>
<tr>
<td>Speed Maximum</td>
<td>Max. speed in the edge</td>
</tr>
</tbody>
</table>

The counting of the different objects by grouping by *Edge Id* (Identifier of the edge) and *Time* (timestamp) allows to identify the edges and timestamps where a traffic jam phenomenon may be happened. The following SQL expression was used in our experiments in order to identify the possible occurrences of traffic jam.

```
Select EdgeID, Timestamp, Count(ID) From TrajectoryLog
Group By EdgeID, Timestamp
Having Count(ID) >= EdgeCapacity
```

---

1Each object *obj* moves between two timestamps $t_i$ and $t_{i+1}$ from its current position *obj.loc$_i$* to its new position *obj.loc$_{i+1}$* according to its computed route.
The SQL expression only uses the edge capacity as constraint to select the records whose a traffic jam is likely happening. It was done because it is the only method to simulate a traffic jam occurrence by using the simulator.

The result set of the SQL expression allows us to label the records of the generated dataset in which a traffic jam (TJ) is likely happening. Table 2.9 describes the additional attributes inserted in the Log Table in order to compose the Labeled Log Table. We have added the binary attribute Crowded to represent the situation of the capacity of the edge. A record whose the number of trajectories crossing the edge is above the value of the maximum capacity of the edge will be represented by Yes value, the other records will be labeled with the value No. Therefore, the records obtained considering the True value of the above SQL predicate were labeled with Yes value (Crowded attribute), and the other records are labeled with No value.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class</td>
<td>Class of Moving Object</td>
<td>{0, 1, 2, 3, 4, 5}</td>
</tr>
<tr>
<td>Cap</td>
<td>Capacity Of Edge</td>
<td>{1, 2, 3, 4, 5}</td>
</tr>
<tr>
<td>Crowded</td>
<td>Situation of capacity of Edge</td>
<td>{Yes, No}</td>
</tr>
</tbody>
</table>

We have defined different levels of TJ occurrence classified according to $dT_{J_{x,y,t}}$ value (see Equation 2.5), each of them representing a kind of density of TJ (considering the number of edges involved in the cell). In this manner a given cell may be classified into a TJ class in a given timestamp and to evolve in this classification along the time dimension.

<table>
<thead>
<tr>
<th>Density Range</th>
<th>Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>[0, 0.25)</td>
<td>025TJ</td>
</tr>
<tr>
<td>[0.25, 0.50)</td>
<td>050TJ</td>
</tr>
<tr>
<td>[0.50, 0.75)</td>
<td>075TJ</td>
</tr>
<tr>
<td>[0.75, 1.00]</td>
<td>TJ</td>
</tr>
</tbody>
</table>

We have implemented a discretization process of the range of TJ density values. It was done to facility the classification mining task, Table 2.10 presents the levels of TJ density discretization.

We have presented our proposal to identify a Traffic Jam phenomenon considering the speed of the objects and the number of objects occurring during a given timestamp (see Section 2.8.1). We have prepared two subsets in order to investigate that proposition. The first one is the full dataset, composed of all attributes (see Table 2.11), and the second subset was composed just using two new attributes used to compute two new measures to represent the context of the cell:

We have discretized the complete range of the values of $dPres_{x,y,t}$ and $lSp_{x,y,t}$. It was done to facilitate the discovery of the patterns, it considers each interval of...
### 2. Trajectory Data Warehouse

#### Table 2.11: Training Set Scheme

<table>
<thead>
<tr>
<th>Set 1</th>
<th>Set 2</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>•</td>
<td></td>
<td>Disc numobs</td>
<td>Discretized $%$ of observations</td>
</tr>
<tr>
<td>•</td>
<td></td>
<td>Disc trajinit</td>
<td>Discretized $%$ of trajectories starting</td>
</tr>
<tr>
<td>•</td>
<td>•</td>
<td>Disc presence</td>
<td>Discretized $%$ of trajectories</td>
</tr>
<tr>
<td>•</td>
<td>•</td>
<td>Disc time</td>
<td>Discretized $%$ time taken</td>
</tr>
<tr>
<td>•</td>
<td>•</td>
<td>Disc distance</td>
<td>Discretized Total distance covered by trajectories</td>
</tr>
<tr>
<td>•</td>
<td>•</td>
<td>Disc speed</td>
<td>Discretized Average speed of trajectories</td>
</tr>
<tr>
<td>•</td>
<td>•</td>
<td>Disc $v_{\text{max}}$</td>
<td>Discretized Maximum speed of trajectories</td>
</tr>
<tr>
<td>•</td>
<td>•</td>
<td>Disc acceleration</td>
<td>Discretized Acceleration of trajectories crossing the cell</td>
</tr>
<tr>
<td>•</td>
<td>•</td>
<td>Disc Sum Edges Capacity</td>
<td>Sum of the capacities of the edges</td>
</tr>
<tr>
<td>•</td>
<td>•</td>
<td>Disc Avg Edges Capacity</td>
<td>Average of the capacities of the edges</td>
</tr>
<tr>
<td>•</td>
<td>•</td>
<td>Disc dPres</td>
<td>Discretized Density of Presence</td>
</tr>
<tr>
<td>•</td>
<td>•</td>
<td>Disc lvlSp</td>
<td>Discretized Level of Speed</td>
</tr>
<tr>
<td>•</td>
<td>•</td>
<td>Label</td>
<td>Discretized Density Range</td>
</tr>
</tbody>
</table>

discretization as an item in a frequent pattern processing.

#### 2.8.3 Statistical Analysis

In this experiment we have used a dataset generated by the simulator, not a real dataset produced by a set of moving objects. Therefore, we expect that the synthetic dataset presents a behavior similar to a real environment. A possible way to investigate this behavior is to compute the relationship among the measures. A popular statistical measure to find the level of relationship between two measures is the **correlation**. Besides, **correlation** can be used to investigate co-occurrence of measures in cells. It is a very important point in order to verify whether it is possible to conclude something about the occurrence of an event ($T_J$) in a spatio-temporal cell by only looking at the measures stored in the cells.

In this section we present the results of a statistical analysis by using the complete synthetic dataset. We consider the relationship among the measures **distance**, **time**, **dPres** and **lvlSpeed**. We used the **correlation** value in order to identify the possible relationships. It is evident that the correlation among the measures can be influenced by several features. Dataset, road network and base granularity of the cells are some of the possible characteristics.

The **Pearson** coefficient is a measure statistic which estimates the correlation of the two given random variables. The coefficient values ranges from -1 to 1. A value of 0 shows that there is no linear relationship between the variables. A value of 1 shows that there is a positive linear relationship between the variables. When the **Pearson** coefficient is -1 it represents that there is a negative linear relationship between the variables. Table of critical values for **Pearson** correlation was used to verify the correlation among the measures.
Table 2.12 presents the results of Pearson coefficient. By using the Table of critical values for Pearson correlation, it is possible to conclude that there is a significative linear correlation among the measures. In the following we discuss the results, the dispersion graphs are also presented for each relationship.

<table>
<thead>
<tr>
<th></th>
<th>Time</th>
<th>Distance</th>
<th>dPres</th>
<th>lvlSpeed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time</td>
<td>1</td>
<td></td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Distance</td>
<td>0.917</td>
<td>0.551</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>dPres</td>
<td>0.438</td>
<td>-0.277</td>
<td></td>
<td>-0.113</td>
</tr>
<tr>
<td>lvlSpeed</td>
<td>-0.318</td>
<td></td>
<td></td>
<td>1</td>
</tr>
</tbody>
</table>

[Time vs Distance] and [Time vs dPres] (Figures 2.15 and 2.16) are considered positive linear relationships. Measure Time represents the time taken by trajectories into the cell. This relationship suggests that when the trajectories remain into the cell for a large time, the distance traveled by trajectories also will increase. A similar behavior is observed between Time and Presence. When the number of distinct trajectories into a cell increase, it results in an increase in the time taken by the trajectories. Those relationships may reveal a behavior similar to a real environment. Large cells containing roads that allow objects to move by staying in the same cell it is an example of a real environment where those relationships could be found. Another possibility is an environment with a large number of objects, in this case the individual trace of each object can contribute to improve the distance traveled by the complete set of objects.

[Time vs lvlSpeed] (Figure 2.17) represents a moderate negative linear relationship. In this case, the level of speed decreases when happen an increase in the
time taken by the trajectories. This behavior can be explained considering the relationship between \textit{time} and \textit{presence}. An increase in the \textit{time} results in a consequent increase in the number of distinct trajectories into the cell, it contributes to improve the probability of occurrence of a traffic jam and, consequently, decreases the level of speed into the cell. Another evident cause of this relationship could be a cell crossed by a lot of very agile objects. In this case, the objects cross the borders of the cell without any occurrence of traffic jam. The consequence will be the reduction in the time taken by the objects into the cell.

![Figure 2.17: Correlation Time vs lvlSpeed](image)

\textbf{Distance vs dPres} (Figure 2.18) is considered a strong positive linear relationship. When the number of distinct trajectories increase, the same behavior can be expected to the distance traveled by the trajectories into the cell. In this case, the behavior describes a cell with a large number of distinct trajectories. However, in this situation, the number of distinct trajectories is not sufficient to start a \textit{traffic jam} phenomenon.

![Figure 2.18: Correlation Distance vs dPres](image)
2.8. TDW: Events Occurrences Patterns

[Distance vs lvlSpeed] (Figure 2.19) is a negative linear relationship. It can be explained considering that an increase in the distance it is a consequence of the increase in the number of distinct trajectories into the cell (see Distance vs dPres). Therefore, this relationship may be an indicator of a Traffic Jam occurrence. It is a consequence of the relationship between Distance vs dPres, when the number of distinct objects exceeds the capacity of the set of edges involved by cell.

![Figure 2.19: Correlation Distance vs lvlSpeed](image)

[dPres vs lvlSpeed] (Figure 2.20) represents a weak negative linear relationship. However, it is an expected behavior: when the number of distinct trajectories increases, the level of speed of the trajectories decreases. It is an expected behavior in the occurrence of a traffic jam phenomenon.

![Figure 2.20: Correlation dPres vs lvlSpeed](image)

The presented results suggest that the synthetic dataset used in the experiments has a behavior similar to a real environment of moving objects. However, in this statistical analysis we have used a linear correlation measure. Therefore, the results obtained allow to investigate the behavior of the dataset considering a linear relationship. In this way, a weak linear correlation or a zero-linear correlation can not allow to affirm that there is not a relationship between two measures. In this
case, a non-linear correlation could be found between the variables. However, in this experiment we consider enough to focus on *linear* correlation.

Besides, the statistical analysis allows to conclude that it is possible to use the co-occurrence of measures in *spatio-temporal* cells in order to investigate the a given movement event. However, it is not sufficient in order to conclude a relationship among measures and occurrences. It is only a step in this process that must be complemented with additional techniques (*e.g.* supervised knowledge).

### 2.8.4 Data Mining Experiment Results

Classification was the data mining task used to conduct the tests. Test set values were obtained after a discretization process executed on the measures of the fact table rows (see Table 2.11). The goal is to verify if is possible to obtain a high level of accuracy to foresee a *Traffic Jam* occurrence by using the new proposed measures ($dPr_{x,y,t}$ and $lSp_{x,y,t}$).

The experiment was conducted by using the *Weka* suite, it is a data mining environment which offers several data mining tasks with various algorithms. We have used the classification mining considering three algorithms: *ID3*, *BayesNet* and *J48*. It was done in order to analyze the results independently of the algorithm.

Table 2.13 presents the results obtained with the experiment, it represents the values obtained considering the complete set of density of *traffic jam* values: {0.25TJ, 0.50TJ, 0.75TJ, TJ}. Column *Set* shows the datasets detailed in Table 2.11 and *Time* column presents, in seconds, the time to build the mining model. Column *Instances* shows the percentage of instances *Unclassified*, *Correctly* and *Incorrectly* classified. Finally, column *Precision* presents the percentage of the accuracy for each level of density of *traffic jam*.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Set</th>
<th>Time</th>
<th>Correct</th>
<th>Incorrect</th>
<th>Unclassified</th>
<th>025TJ</th>
<th>050TJ</th>
<th>075TJ</th>
<th>TJ</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID3</td>
<td>1</td>
<td>0.09</td>
<td>53.02%</td>
<td>43.630%</td>
<td>3.34%</td>
<td>61.70%</td>
<td>17.00%</td>
<td>27.70%</td>
<td>59.20%</td>
</tr>
<tr>
<td>ID3</td>
<td>2</td>
<td>0.00</td>
<td>62.51%</td>
<td>37.35%</td>
<td>0.13%</td>
<td>66.30%</td>
<td>0.00%</td>
<td>45.50%</td>
<td>60.40%</td>
</tr>
<tr>
<td>BayesNet</td>
<td>1</td>
<td>0.08</td>
<td>58.22%</td>
<td>41.77%</td>
<td>0.00%</td>
<td>70.40%</td>
<td>17.60%</td>
<td>33.60%</td>
<td>64.70%</td>
</tr>
<tr>
<td>BayesNet</td>
<td>2</td>
<td>0.00</td>
<td>61.29%</td>
<td>38.71%</td>
<td>0.00%</td>
<td>67.00%</td>
<td>0.00%</td>
<td>41.70%</td>
<td>62.70%</td>
</tr>
<tr>
<td>J48</td>
<td>1</td>
<td>0.16</td>
<td>61.15%</td>
<td>38.84%</td>
<td>0.00%</td>
<td>68.10%</td>
<td>20.00%</td>
<td>36.50%</td>
<td>62.20%</td>
</tr>
<tr>
<td>J48</td>
<td>2</td>
<td>0.02</td>
<td>62.55%</td>
<td>37.44%</td>
<td>0.00%</td>
<td>66.40%</td>
<td>0.00%</td>
<td>41.50%</td>
<td>60.50%</td>
</tr>
</tbody>
</table>

The first point to analyze is the time to build the mining model, by using the set 2 (just $dPr_{x,y,t}$ and $lSp_{x,y,t}$ measures) we can obtain a significant reduction in the value of the time independently of the algorithm. The same reduction can be observed considering the value of the *instances unclassified*, the value is very close to 0.00% in both cases (Set 2). In the same way, we can observe an increasing in the percent of *instances correctly* classified. The value of *instances incorrectly* classified
also decreases for the two sets. In resume, by using the set 2 we have observed an increasing of the accuracy to classify the instances. We have obtained this gain in the same time that we have obtained an increase in the performance to build the model.

About precision, we can observe that the best values happen in the $025TJ$ and $TJ$ values of the density of traffic jam. In the intermediate values ($050TJ$ and $075TJ$) we did not obtain a reasonable accuracy. In the same way that happened in the analysis of the classification of the instances, in the precision measure we can observe an increase in the accuracy of the results by using the set 2. Similarly, this gain happened with the reduction of the time to build the model.

Another point to analyze it is related to the rules produced by decision tree based mining techniques. The set of trees produced by using the Set 1 as data source is composed of the attributes (see Table 2.11) Disc presence, Disc distance, Disc speed, Disc $v_{\text{max}}$, Disc acceleration and Disc Sum Edges Capacity. The number of levels of the trees, produced by Set 1, ranges between 5 and 6 levels. Meanwhile, the number of levels of the all trees produced by using the Set 2 is 2 (Disc dPres, Disc lSp). By using a number significantly smaller of tree levels it is possible to obtain better results, considering the accuracy of the foresee of the traffic jam event.

We have done another experiment by using the Set 1 in order to verify the distribution of the data regarding the threshold values defined as density of presence ($d\text{Pres}_{x,y,t}$) and level of speed ($l\text{Sp}_{x,y,t}$). By using the results obtained with the data mining experiments, we have plotted the Figures 2.21 and 2.22.

![Figure 2.21: Number of Cells (%) vs TJ Density - Free Traffic](image)

The results confirm that when the density of presence $\leq 0.20$ and level of speed $\geq 0.80$, a typical environment of Free Traffic, 100% of the cells are classified in the $025TJ$. Similarly, when the density of presence $\geq 0.80$ and level of speed $\leq 0.20$, a typical environment of Traffic Jam, almost 80% of the cells are classified in the $TJ$. These thresholds are efficient to identify a traffic jam.
occurrence in a given cell. More than 80% of the objects are selected considering these values.

![Figure 2.22: Number of Cells (%) vs TJ Density - Traffic Jam](image)

### 2.9 Conclusions

Our **TDW** offers an environment to receive, process and store trajectory data by using the multidimensional concept. We proposed a data warehouse based on star schema: one fact table and three dimension tables. Each fact table record store measures representing characteristics of a set of trajectories involved by cell coordinates (*dimension tables*).

We consider that the **TDW** works on a data stream environment. Therefore, the **TDW** must be able to offer a solution adequate to the data stream characteristics. The proposed **TDW** introduces measures and techniques of loading and aggregation considering the data stream environment.

In order to investigate the **TDW**, we developed and implemented a prototype. The prototype considers the constraints to load the data warehouse and compute the aggregation values. The application is a first step in order to solve the problems of loading and aggregation in a **TDW** environment. The data cube model adopted is very simple, it is just a contribution in order to improve the discussion about the problems to implement a **TDW**. However, this model can be extended to more general situations.

The current stage of the prototype can solve some problems of a trajectory data warehouse environment. A possible future work may be to sophisticate the hierarchy of the dimensions. The loading phase is an opened problem. We limited
the loading phase considering a linear interpolation. However, it is possible to find some topological situations (e.g. roads, bridges) where the proposed interpolation is not efficient, because of the some constraints in the movement of the object.

In this work we used the roll-up operation. However, may be interesting to offer mechanisms in order to compute other operators such as drill-down, pivot, slice and dice. Therefore, the development of a query language using OLAP operators also is a possible point to research.

The results obtained in the experiments allow to consider the measures density of presence and level of speed enough to investigate the occurrence of TJ events. The best results of accuracy (classification of instances) and precision (TJ density levels) were obtained in the Set 2 (density of presence and level of speed). Besides, the time taken to compute these patterns are also reduced by using the Set 2. Therefore, considering the Traffic Jam event, the proposed TDW can be used to identify occurrences in an environment of trajectories.

We have obtained a high level of accuracy to find TJ occurrences considering the cells whose level of traffic jam is 0.25TJ or TJ. Therefore, a possible future research may be to investigate new methods/algorithms to improve the accuracy of results analyzing mixed cells.

The results of the statistical analysis (correlation) suggest that it is possible to investigate the use of co-location patterns among the measures of the TDW. The proposal is to use the co-location patterns to investigate the occurrence of movement events inter-region. In other words, an occurrence of an event (composed of a subset of measures) at a given area (region) could affect the occurrence of a different consequent event in another different area. The second part of this Thesis is focused in this investigation, the goal is to verify whether traditional data mining techniques can be used to reveal the hidden knowledge on trajectory data.
Second Part
The development of new technologies for mobile computation and remote sensors need huge databases to store the data received concerning multitude of sampled moving objects. Basically, in the most of the cases, the data report the location (2-dimensional space) of an object at a given timestamp. Therefore, by using the records it is possible to reconstruct the trajectory of a moving object, even if some level of uncertainty in the reconstruction may occur. These trajectory databases can be the source to discover a very important knowledge about moving objects: movement patterns.

There are a lot of environments where this knowledge can be applied: management of road traffic, monitoring of movement of animals in order to investigate the migratory phenomenon, air traffic, enemy movement in a battlefield and many others. In this chapter we present the related work to investigate the movement patterns, all of them are based on a trajectory database to store the trajectory data. The goal is to discuss the requisites of a mechanism to find spatio-temporal patterns on Trajectory DataWarehouse data. We focus on co-location patterns.

3.1 General Framework

On [57] the authors present a discussion about the use of data mining techniques to find interesting spatio-temporal patterns from Earth Science data. In that work, the data consist of time series measurements for various Earth science and climate variables (e.g. soil moisture, temperature and precipitation), along with additional data from existing ecosystem models. Those measurements represent a knowledge related to a given region. The presented environment has some similarities with the environment of TDW. In both environments we have a spatio-temporal area divided into a set of grid cells, and for each cell there is a set of measures representing characteristics of that region. As in the TDW environment, the ecologists are interested in a variety of spatio-temporal association patterns involving sequences of events abstracted from the measures stored at each cell. The authors present four types of patterns: Intra-zone non sequential patterns, Inter-zone non-sequential
3. Pattern Mining applied to Trajectories

We argue that those patterns can be applied on the TDW environment. In the following we present a description of those types of patterns and a possible framework of use of the concepts applied to trajectories.

3.1.1 Intra-zone non sequential patterns

By using this pattern the goal is to find relationships among events in the same cell (or set of cells), ignoring the temporal aspects of the data. Association rule mining algorithms (e.g. Apriori [2]) could be used to reveal intra-zone non-sequential associations among events occurring at the same spatial location. In our TDW, each record can be considered as a transaction identified by cell coordinates. The records are composed of different items (discretized measures). Therefore, on a TDW environment an association rule \( \text{antecedent} \rightarrow \text{consequent} \) represents a relationship among discretized values of measures. This type of pattern may reveal different relationships among measures at different locations. The following association rules are examples of possible patterns.

- \((\text{Area } A) - 10\text{densitypresence} \rightarrow 01\text{levelSpeed}\)
- \((\text{Area } B) - 10\text{levelspeed} \rightarrow 01\text{time}, 01\text{densitypresence}\)

The rules reveal two different behaviors. The rule found at area \(A\) represents a possible occurrence of a traffic jam event. When the number of distinct objects increases, the level speed of them decreases. However, the rule at area \(B\) represents an opposite situation, when the level speed of the objects is high, the value of time spent by the objects in that area is low, and the value of the density of presence is also low. It may represent the behavior of fast objects crossing areas with high capacity of traffic.

Therefore, this type of pattern can identify interesting behavior of moving objects at a given area by using the traditional mining algorithms (Apriori, for example). However, we can not observe the evolution of the behavior along the time, and it is a crucial knowledge when the goal is to mine trajectories of moving objects.

3.1.2 Intra-zone sequential patterns

Temporal relationships among events occurring within the same cell. In this type of pattern the temporal information is used to reveal the evolution of the behavior. The time dimension allows to find intra-zone sequential associations among the events. Traditional algorithms to find frequent sequential patterns in market-basket data can be used in this case. In [3] the authors consider a database of sequences, where each sequence is a list of transactions ordered by transaction-time, and each transaction is a set of items. The goal is to discover all sequential patterns with a user-specified minimum support, considering time constraints that specify a minimum and/or
maximum time period between adjacent elements in the pattern. The support of a pattern is the number of data-sequences that contain the pattern. In the same work (3) the authors propose to relax the restriction that establishes that the items in an element of a sequential pattern must come from the same transaction. Instead, allowing the items to be present in a set of transactions whose transaction-times are within a user-specified time window.

The problem of finding sequential patterns in a database of sequences it is similar to the problem of finding sequential patterns of events in a TDW. The TDW records can be considered as transactions. Each of them identified by the cell coordinates. The discretized measures are the items of transaction. Therefore, an occurrence of an event of movement may be represented by a sequence of discretized measures.

For example, an event of traffic jam may be represented by a sequence like \( \langle 10\text{dpresence}, 10\text{time}, 01\text{levelspeed} \rangle \), where \( 10\text{dpresence} \) represents a high level of density of presence, \( 01\text{levelspeed} \) represents a low level of speed of the objects and \( 10\text{time} \) represents a high level of time spent by the trajectories into the cell.

Also on a TDW there is a time constraint which can be used to restrict the patterns. In a TDW environment a time constraint may be the definition of a maximum time interval between successive elements in the sequence.

The sequential pattern \( \langle 10\text{dpresence}, 01\text{levelspeed}, 10\text{time} \rangle \) may not be enough to represent the occurrence of a traffic jam event. We do not have the time information between the events. Therefore, the sequential pattern \( \langle 10\text{dpresence}, (5 \text{ minutes}), 10\text{time}, (10 \text{ minutes}), 01\text{levelspeed} \rangle \) represents the event more precisely. In this case, the sequence pattern represents a traffic jam considering a time-window of 15 minutes. However, by using a time-window of 10 minutes, the same sequence pattern does not represent a traffic jam event.

This type of pattern allows to reveal sequence patterns by using a time constraint. In a TDW environment it represents a useful knowledge to understand the evolution (creation, stabilization and dissipation) of the movement events.

### 3.1.3 Inter-zone non-sequential patterns

This pattern reveals the relationships among events happening in different cells, ignoring temporal aspects of the data. In a TDW environment it allows to investigate whether the occurrence of an event at a given area can be related with the occurrence of another event at a different area. In [54] the authors introduce spatial co-location rules based on the frequent co-occurrences of events within the same spatial window. However it is not possible to investigate how this relationship develops along the time. In an environment of TDW the temporal dimension is a very important characteristic. We have to be able to identify the relationship among the occurrences of events at different areas. However, we also have to know how it happens along the time. It is a crucial knowledge to forecast the occurrence of some events (e.g. traffic jam) and to manage the environment.
3.1.4 Inter-zone sequential pattern

Temporal relationships among events occurring at different spatial locations. We considered this the most complete pattern for mining a TDW. By using this pattern it is possible to identify the relationship among the occurrence of events at different areas. Besides, it is possible to understand the evolution of the relationship among the occurrences. This knowledge allows to take actions to solve a problem that may occur considering the previous occurrence of another event at a different area. We classified our proposed algorithm (Target Spatio-Temporal Co-location Pattern) into this class of pattern.

3.2 Sequential Pattern

The problem of Frequent Sequential Pattern (FSP) was introduced in [3]. Considering a database of sequences $D$ composed of a set of sequences where each sequence is a set of timestamped items, and the timestamps ordering the elements in the sequence, the Frequent Sequential Pattern (FSP) problem is to find all the sequences that are frequent in the database sequence $D$. A frequent sequence is a subsequence of a large percentage of sequences of database $D$. PrefixSpan [46] and SPADE [64] are some of the approaches proposed to solve this problem. The most of the proposals of Frequent Sequential Pattern were defined to work with discrete values. Therefore, when the goal is to find Frequent Sequential Patterns in a Spatio Temporal environment some adaptations in the transactional proposals must be done. It is necessary to adapt the concepts to the spatio-temporal characteristics, mainly focusing on the temporary dimension of the values and events. The goal of usage of Sequential Pattern Mining to analyze a set of trajectories it is to discover usable knowledge about movement behavior. This knowledge can be useful for the understanding of the phenomena of the mobility. The proposals consider a trajectory as a type of transaction composed of several items. The location of the objects for each timestamp are the itemset of transactions. In [24] the authors suggest to investigate the mobility phenomena by using the concept of trajectory patterns. It is a description of frequent behaviors considering space and time dimensions. The Space dimension is represented by regions of space visited during the movement. Meanwhile, Time dimension represents the duration of movements. Therefore, the proposed pattern (trajectory pattern) represents a set of individual trajectories that share the property of visiting the same sequence of places with similar travel times. Actually, a trajectory pattern is a sequence of spatial regions that, on the basis of the source trajectory data, it is frequently visited in the order specified by the sequence. The transition between two consecutive regions in such a sequence is annotated with a typical travel time that emerges from the input trajectories. The approach works with two basic concepts:
• Regions of Interest in the given space.

• Typical Travel Time of moving objects from region to region.

An example of a trajectory pattern, according to the proposal, may be the following:

Bus Station $\xrightarrow{30\text{min}}$ Central Park $\xrightarrow{45\text{min}}$ Empire State

It is interesting to observe that the pattern does not specify any route among two consecutive regions. The pattern presents a typical travel time, which approximates the travel time of each individual trajectory represented by the pattern. Another interesting characteristic is that the individual trajectories aggregated in a pattern are not necessary simultaneous. The only requisite is that the trajectories visit the same sequence of places with similar transition times.

[Mining-LSP]. The proposal presented in [25] maintains the focus in the investigation about the use of data mining techniques to find patterns of long and sharable frequent routes of the moving objects. In agreement with the authors, Spatio-temporal rules can be found by constructing spatio-temporal baskets, from which traditional association rule mining methods can discover spatio-temporal rules. When the items in the baskets are spatio-temporal identifiers and are derived from trajectories of moving objects, the discovered rules represent frequently traveled routes. The proposal is to find the frequent routes that are long and sharable. The approach presents a database projection based method for efficiently extracting such long, sharable frequent routes. The method prunes the search space by making use of the minimum length and sharable requirements and it avoids the generation of the exponential number of sub-routes of long routes.

The proposal to find long and sharable patterns (LSPs) in trajectories filters the trajectories to contain only sub-trajectories that are frequent. It is done by using a defined support criterion. Besides, it removes trajectories that do not meet the minimum length criterion. Then, it alternates two steps until there are undiscovered LSPs. The first step entails the discovery of a LSP. The second step entails the filtering of trajectories by the previously discovered pattern. An advantage of the proposed method it is the ease of implementation in commercial relational database management systems (RDBMSes).

[FSTP]. In the proposal presented by [19] the goal is to find sequential patterns defined as routes frequently followed by a moving object. In that approach, pattern elements were defined as spatial regions around frequent line segments. Basically, the method transforms the original sequence into a list of sequence segments in order to detect frequent regions. The Apriori based algorithm proposes the usage of a substring tree structure to search the sequential patterns.
3.3 Clustering

The proposals based on Clustering data mining techniques are focused in the analysis of the trajectories based on the similarity among of them. The criteria of similarity were proposed by using spatio and/or temporal characteristics. The basic idea is to investigate the behavior, along the time, of a set of similar trajectories.

[STPMine]. Another approach of Spatio Temporal Data Mining applied to trajectories was presented in [40]. The authors define the spatio temporal periodic pattern mining problem and present a mining algorithm to retrieve maximal periodic patterns. The proposal is to discover periodic patterns of movement of the objects that follow approximately the same routes, considering regular time intervals. The proposal considers that the locations of moving objects are sampled over a dimension time. Therefore, the movement of an object can be represented as an n-length sequence $S$ of spatial locations, one for each timestamp. Given a sequence $S = \{(l_0, t_0), (l_1, t_1), \ldots, (l_{n-1}, t_{n-1})\}$, where $l_i$ is the location of the object at time $t_i$, a minimum support $\text{min sup}$, and an integer $T$, called period (day, week, month), the problem is to discover movement patterns that repeat themselves every $T$ timestamps. The discovered pattern $P$ is a $T$-length sequence of the form $r_0\ldots r_{T-1}$, where $r_i$ is a spatial region or the special character *, indicating the whole spatial universe. Then, a pattern $AB*C$ represents that at the beginning of the cycle the object is in region $A$, at the next timestamp it reaches the region $B$, then it moves irregularly and goes to region $C$.

[Flock]. In [28] the goal is to find patterns of movement of objects. These patterns are large enough subgroups of the moving point objects that exhibit similar movement in the sense of direction, heading for the same location, and/or proximity. The proposal is to investigate the movement patterns of a uniform group of moving objects. The proposal investigated the patterns proposed in [36]: Flock, Leadership, Convergence and Encounter. These patterns are based on similar direction of motion, change of direction and location. Each pattern can occur for a subset of the moving objects at a given timestamp. The flock pattern describes moving objects in the same direction while being close to each other. The expression being close denotes a circle of some specified radius $r$, whose position is initially not known. A set of moving objects can have many flock patterns and even one single entity can be involved in several flock patterns. The difference between flock and leadership pattern is that, in a leadership pattern, one of the entities was already heading in the specified direction for some time before the flock pattern occurs. The convergence represents a pattern of movement where the moving objects have the same destination local, therefore, in this pattern the direction of motion does not change. The moving objects do not need arrive at the same time. The destination local is as a circle whose radius can be specified and whose position is unknown. The encounter pattern is a convergence pattern where the moving objects arrive at the same time. The proposed algorithms compute the patterns considering a variation
of the values of radius (to denote the area) and the minimum number of moving objects (to define a cluster). Therefore, by the proposal it is possible to find patterns considering a defined threshold of approximation.

[MC1]. The concept of Moving Cluster was presented by [33]. The authors define a moving cluster as a set of objects that move close to each other for a long time interval. Then, a moving cluster is a sequence of spatial clusters that appear in consecutive snapshots of the object movements, such that two consecutive spatial clusters share a large number of common objects. The proposal is to discover moving clusters in a trajectory database. In that approach the identity of a moving cluster remains unchanged while its location and content may change over time. Therefore, the proposal considers patterns in the form of moving regions (clusters) within time intervals.

[Flock-Time]. The proposal presented in [27] introduces a method to find specific spatio-temporal patterns in trajectories: flocks and meetings. The proposal extends the method presented in [28]. The patterns are large enough subgroups of the moving point objects that exhibit similar movement and proximity for a certain amount of time. The proposal investigates a method to compute a longest duration flock or meeting. In following, the definitions of flock and meeting (and sub-types) adopted by the authors.

- Given a set of \( n \) trajectories of entities in the plane, where each trajectory consists of \( \tau \) line segments, a flock in a time interval \( I \), where the duration of \( I \) is at least \( k \), consists of at least \( m \) entities such that for every point in time within \( I \) there is a disk of radius \( r \) that contains all the \( m \) entities.
  - Fixed-flock is the flock where the same \( m \) entities stay together during the entire interval.
  - Varying-flock, is the flock where the entities in the flock change during the interval. In this case the disk of radius \( r \) changes location in a continuous way.

- Given a set of \( n \) trajectories of entities in the plane, where each trajectory consists of \( \tau \) line segments, a meeting in a time interval \( I \), where the duration of \( I \) is at least \( k \), consists of at least \( m \) entities that stay within a stationary disk of radius \( r \) during \( I \).
  - Fixed-meet is the meet where the same \( m \) entities stay together during the entire interval.
  - Varying-meet is the meet where the entities in the meeting region change during the interval.

In this case, a set of at least \( m \) entities within a circular region of radius \( r \) is a flock if they move in the same direction during a given time interval. The possibility of
the use of a time interval it is the main contribution of this proposal. It extends the approach presented in [28], by using the time interval in order to properly represent the flock and meeting patterns.

[Operators]. In [12] the goal is to investigate the problem of trajectory similarity search in Trajectory Databases. The trajectories are represented by a sequence of locations of the objects with respect to time. The authors define that problem as a method to detect and quantify (dis-)similarity among the trajectories of moving objects. The similarity among the trajectories is measured by the usage of the distance concept. The proposal presents four types of similarity search:

(Time-relaxed) Spatial Trajectory Similarity Search Find clusters of objects that follow similar routes (i.e., projections of trajectories on 2D plane) during the same time interval (e.g. co-location and co-existence from 3pm to 6 pm)

(Time-aware) Spatiotemporal Trajectory Similarity Search Find clusters of moving objects taking only their route into consideration (i.e., irrespective of time, direction and sampling rate)

Speed-pattern based Similarity Search Find clusters of objects that follow similar routes and, additionally, move with a similar speed pattern

Directional Similarity Search Find groups of objects that follow a given direction pattern (e.g. NE during the first half of the route and subsequently W), concurrently or not

For each type of similarity query the authors introduce distance operators. Besides, propose respective query processing algorithms based on primitive (space and time) and derived parameters (speed, direction) of trajectories.

3.4 Co-location

In this section we focus on the use of co-location patterns in the set of TDW data. We have to verify whether it is possible to forecast the occurrence of an event in a given area (cell or set of cells) considering the previous occurrence of another event into the neighborhood of the cell. Co-location pattern is the process to find subsets of boolean spatial features frequently located together. The analysis of a trajectory dataset may reveal, for example, a frequent co-location pattern of a traffic jam occurrence in a given cell with the decreasing of average speed of the moving objects located in the neighborhood of the cell. The spatial co-location concept has similarity with association rules concept. However, spatial co-location patterns present an additional difficult regarding to the traditional frequent patterns: there is not the notion of transactions. In the traditional association rules concept, by using a market basket scenario, the transactions represent sets of item bought together
by customers. Traditional association rules are a representation of the item sets bought together. However, in the spatial co-location rules concept, transactions are not explicit. Another difference among the concepts (association rules and spatial co-location rules) is related to the independence of transactions. In an environment of association rules the transactions are independent of each other. Meanwhile, in a spatial co-location rules environment the transactions are embedded in a space area and share several spatial relationships with each other.

Therefore, a very important step in order to implement the spatial co-location rules, considering the difficulty to create explicit disjoint transactions from continuous spatial data, it is to model the co-location rules. In [54] the authors suggest three different approaches to model the spatial co-location rules:

Reference Feature Centric Model this model [35] is relevant to application domains where the focus is a specific boolean spatial feature. In this case the model enumerates neighborhoods materializing a set of transactions around instances of the reference spatial feature.

Window Centric Model The goal is to predict sets of spatial features likely to be discovered in a land parcel given that some other features have been found in the same area. This model enumerates all possible windows as transactions. Considering a space discretized by a uniform grid, several windows of size $l \times l$ will be enumerated and materialized. In this case, each transaction contains a subset of spatial features of which at least one instance occurs in the corresponding window.

Event Centric Model This model is adequate to applications where there are many types of boolean features. The goal is to find subsets of spatial features likely to occur in a neighborhood around instances of given subsets of event types.

In the following we present some proposals of Co-location Pattern Mining.

[Co-location Miner] On [54], [31] the authors present a method to find subsets of features frequently located together. The proposal introduces a notion of user-specified neighborhoods in place of transactions to specify groups of items. The definitions of the method are based in the Event Centric Model.

**Definition 3.4.0.1** A co-location is a subset of boolean spatial features

**Definition 3.4.0.2** A co-location rules is of the form: $C_1 \rightarrow C_2 (p, cp)$ where $C_1$ and $C_2$ are co-locations, $p$ is a number representing the prevalence measure and $cp$ is a number measuring conditional probability.

**Definition 3.4.0.3** Given a reflexive and symmetric neighbor relation $R$, a neighborhood of a location $l$ is a set of locations $L = \{l_1, ..., l_k\}$ such that $l_i$ is a neighbor of $l$. 

60 3. Pattern Mining applied to Trajectories

The relation \( R \) may be defined using topological relationships (e.g. connected, adjacent), metric relationships (e.g. Euclidian distance, Manhattan distance etc) or a combination (e.g. shortest-path distance in a graph such as road-map).

**Definition 3.4.0.4** For a subset of locations \( L' \) if \( L' \) is a neighborhood of every location in \( L = \{l_1, ..., l_k\} \) then \( L' \) is a neighborhood of \( L \).

**Definition 3.4.0.5** \( I = \{i_1, ..., i_k\} \) is a row instance of a co-location \( C = \{f_1, ..., f_k\} \) if \( i_j \) is an instance of feature \( f_j \) (\( \forall j \in 1, ..., k \)) and \( I \) is a neighborhood of \( I \) itself.

Therefore, if elements of \( I \) are neighbors to each other, then \( I \) is an instance of \( C \). In Figure 3.1 \( \{(5,2),(5,3)\} \) is an instance of co-location \( \{\times, *\} \), considering a 9-neighbor relationship over cells of a grid.

**Definition 3.4.0.6** The table instance of a co-location \( C = f_1, ..., f_k \) is the collection of all its row instances.

**Definition 3.4.0.7** Participation ratio \( pr(C, f_i) \) for feature type \( f_i \) of a co-location \( C = \{f_1, ..., f_k\} \) is the fraction of instances of \( f_i \) which participate in any row instance of co-location \( C \).

An example can be presented by using Figure 3.1 \( \{(5,2),(5,3)\}, \{(5,2),(6,3)\}, \{(5,2),(6,2)\}, \{(5,4),(5,3)\}\) and \( \{(2,3),(3,3)\} \) are instances of co-location \( \{\times, *\} \). However, only the instances \( (5,2),(5,4) \) and \( (2,3) \) of spatial feature \( \times \) out of six participates in co-location \( \{\times, *\} \). Notice that the project operation remove duplicates, e.g. \( 5.2 \) occur three times in the patterns above, however it is just counted once. Therefore, in this case, the participation ratio for feature type \( \{\times\} \) of the co-location \( \{\times, *\} \) it is computed by using the following formule: \( pr(\{\times, *\}, \times) = \frac{3}{6} = 0.50 \). Similarly, the value of participation ratio for feature type \( \{\times\} \) of the co-location \( \{\times, *\} \) is \( pr(\{\times, *\}, \times) = \frac{4}{4} = 0.57 \).

**Definition 3.4.0.8** Participation index of a co-location \( C = f_1, ..., f_k \) is \( \min_{i=1}^{k} pr(C, f_i) \).

The participation index for co-location \( \{\times, *\} \) can be computed by using the participation ratio values for both features \( \{\times\} \) and \( \{\times\} \) computed in the example above. Therefore, participation index for co-location \( \{\times, *\} \) is \( \min\{0.50, 0.57\} = 0.50 \).

**Definition 3.4.0.9** The conditional probability of a co-location rule \( C_1 \rightarrow C_2 \) is the probability of finding \( C_2 \) in a neighborhood of \( C_1 \).

\[
\frac{|\pi_{C_1}(\text{all row instances of } C_1 \cup C_2)|}{|\text{instances of } C_1|}
\]
3.4. Co-location

In this model the **data items** are the **boolean feature types** (traffic jam or another events). **Transactions** are defined by **neighborhoods of instances of feature types** (stored measures for each cell of the *TDW*) and the **interest measures** are defined as following:

- **Prevalence**: participation index of \( C_1 \rightarrow C_2 \)
- **Conditional Probability**: \( Pr(C_2 \text{ in a neighborhood of } C_1) \)

The proposal introduces the **Co-location Miner** algorithm (see Algorithm 2) to generate all the co-location rules with prevalences and conditional probabilities above defined thresholds.

**Algorithm 2 Co-location Miner Algorithm**

1: **Input:**
2: \( K \) boolean spatial instance types and their instances
3: A symmetric and reflexive neighbor relation \( R \)
4: A user specified minimum threshold prevalence measure (\( \text{min\_prevalence} \))
5: A user specified minimum conditional probability (\( \text{min\_cond\_prob} \))
6: **Output**: \{co-location patterns with partition index > \( \text{min\_prevalence} \) and conditional probability > \( \text{min\_cond\_prob} \)\}
7: **Method:**
8: prevalent size 1 co-location set along with their table instances
9: Generate size 2 co-location rules
10: for size of co-locations in (1,2,3,..., K - 1) do
11: Generate candidate prevalent co-locations by using an *apriori-based* algorithm.
12: Generate table instances and prune based on neighborhood
13: Prune based on prevalence of co-locations
14: Generate co-location rules
15: end for
[Partial Join]. The proposal presented in [62] suggests the occurrence of a computational bottleneck in the execution time of the proposal discussed previously [51] [31]. The problem happens when it is necessary to compute joins to identify instances of candidate co-location patterns. The authors propose a novel partialjoin approach for mining co-location patterns efficiently. It transactionizes continuous spatial data while keeping track of the spatial information not modeled by transactions. Besides, it uses a transaction-based Apriori algorithm and adopts the instance join method for residual instances not identified in transactions. One of the goals is to reduce the computational cost to identify instances of co-locations split across explicit transactions. It is based in the fact that just event instances having at least one cut neighbor relationship are related to the neighborhood instances split over transactions.

The authors suggest four new concepts: neighborhood transaction, intra$X$ row instance, inter$X$ row instance and cut neighbor relation. A Neighborhood Transaction is composed of the instances that are involved by the same neighborhood area. There is a difference between the approach presented in [54] where the transactions were generated by using rectangular grids. In that case, a transaction that belongs to a common neighborhood, may be divided into two transactions. A intra$X$ row instance is a row instance belong to a common transaction $T$. A cut neighbor relation represents a neighbor relation between two instances where the instances are neighbors of each other but belong to distinct transactions. Meanwhile, a inter$X$ row instance are the instances that have at least one cut neighbor relation.

The identification of the all intra$X$ instances it is performed by using a transaction-based Apriori algorithm. Inter$X$ instances are generated by using the join-based co-location mining algorithm [54]. In this method all instances in the transaction are neighbors of each other and no spatial operation and combinatorial operation it is required to find the intra$X$ instances. The computation cost of instance join operations for generating only inter$X$ instances not identified in the transactions is relatively cheaper than one for finding all instances of co-locations.

[Joinless]. Another proposal in the same area of the presented above it is [63]. The approach is focused on reduce the computational cost of finding co-location patterns by reducing the number of join operations. The proposal presents two models of neighborhood partition: Star and Clique neighborhood partitioning. The proposed joinless co-location mining algorithm is based on the star neighborhood partition model. The star neighborhood of an object it is a set of the center object and objects in its neighborhood whose feature types are greater than the feature type of the center object in a lexical order. The proposal presents an algorithm composed of three steps:

**Step 1:** converts an input spatial data set into a set of disjoint star neighborhood.

**Step 2:** gathers the star instances of candidate co-locations from the materialized
set and coarsely filters the candidates using the prevalence values of their star instances.

**Step 3:** filters their co-location instances from the star instances, finds prevalent co-locations and generates co-location rules.

The joinless proposal does not require expensive spatial joins or instance joins to identify co-location instances. It can be done by using an instance lookup scheme.

All the proposals presented previously are defined to find global spatial co-location patterns based on a fixed interest measure. Those proposals assume that patterns are uniformly distributed over the space. However, it is not a characteristic found in real environments. A frequent global pattern may not be frequent in a given delimited area. An occurrence of traffic jam, for example, may be globally frequent, but in a given area the traffic jam does not occur. Therefore, a more complete proposal of spatial co-location pattern has be able to find local co-location patterns, presenting patterns with a higher level of accuracy.

[Zoloc-Miner]. In [11] the authors present a proposal to find local co-location patterns named Zonal co-location patterns. The proposal considers that a pattern are not uniformly distributed over the space. Therefore, different regions of the space may present different patterns. The idea is to delimit the area (define a zone) in order to find local co-location patterns. In that approach, zonal co-location patterns represent subsets of feature types that are frequently located in a subset of space. Besides, the patterns can be computed by using dynamic parameters: repeated specification of zone and interest measure values according to user preferences.

A fundamental step in the proposal is to index space while storing co-location patterns. According the authors, a index structure for zonal co-location patterns must be able to discover cross-neighboring co-locations, solve the problem of overlapping user-defined mining zones and to be efficient computationally.

The proposed solution (clQuad-tree) is based on classical index structure Quad-tree [51]. Basically, the proposed index structure is built in three phases:

- Initial quads are created within the clQuad-tree
- Points belonging to the quads’ buffer regions are assigned
- Co-locations are generated and stored for each quad in the clQuad-tree.

The Zoloc-Miner algorithm computes the patterns, by using the clQuad-tree index structure, in two phases. In the first phase a set of potential candidates are retrieved for a given zone, then, in the second phase, the patterns that satisfy a prevalence threshold are selected.

**Phase 1**: co-location instances for a given zone are retrieved from the tree. Each quad that intersects with the area of research (zone) is verified. A quad is composed of a set of patterns and their instances. When the instances of a pattern intersect a zone the pattern is added to the candidate set.
Phase 2: In this phase the participation index (see Definition 3.4.0.8) for each pattern of the candidate set is calculated. If the participation index of the pattern is no less than the previously defined threshold, then the pattern is selected and chosen to compose the final result.

[TopologyMiner]. In [59] the proposal is to find topological (co-location) patterns by using a summary-structure that records the instances’ count information of a feature in a region within a time window, instead of instances themselves. The reduced size allows to kept the proposed structure in the main memory. Therefore, it may be a very important characteristic for reducing the computational cost to find co-location patterns. The proposed TopologyMiner algorithm uses the summary-structure to find the patterns without a step of candidate generation. It avoids the generation of many candidates and multiple scans of the database. Unlike of the traditional proposals, whose are based in the candidate-generation-and-test methodology, the proposed algorithm discovers the co-location patterns in the depth-first manner and follows the pattern growth methodology.

The TopologyMiner algorithm divides the search space into a set of partitions. In each partition, it uses a set of locally frequent features to grow patterns. This happens in two phases:

1. The space and time dimensions are divided into a set of disjoint cubes. The summary-structure is built through a scan database operation. Besides, indexes (in summary-structure) also are built to facilitate the mining operation.

2. The count information obtained in the previous database scan and stored in the summary-structure are used to discover the frequent co-location patterns in the depth-first manner.

The summary-structure is the main innovation in the proposal. Considers \( \mathcal{D} \) be the spatio-temporal database, \( \mathcal{R} \) be the distance threshold, and \( \mathcal{W} \) be the time window threshold. The database \( \mathcal{D} \) is divided into a set of disjoint cubes \( \{ \langle c_{x_1,y_1}, w_1 \rangle, ... \langle c_{x_p,y_p}, w_q \rangle \} \) where \( \{c_{x_1,y_1}, ..., c_{x_p,y_p}\} \) are 2D cells with width \( \frac{\mathcal{R}}{\sqrt{2}} \), and \( \{w_1,w_2, ..., w_q\} \) are 1D time periods with width \( \frac{\mathcal{W}}{2} \). The proposal considers, for the instances in a cube \( \langle c_{x_{1k},y_k}, w_t \rangle \), that the close neighbors are in the Neighbor-set \( N_{c_{x_{1k},y_k},w_t} = \{ \langle c_{x_{1i},y_i},w_s \rangle | | y_k - y_i | \leq 2 \text{ and } | x_k - x_i | \leq 2 \} \).

After the division of the dimensions (space and time) into a set of cubes, and the consequent definition of the Neighbor-set, it is executed the database scan operation in order to hash the instances of the features into the cubes. For each cube that contain at least one feature instances, it is kept (in the summary-structure) the instances’ count of a feature. The proposal uses two hash-based indexes in order to retrieve information in the summary-structure: Cube-Feature Index (CFI) and Feature-Cube Index (FCI). By using CFI index it is possible to obtain the features
that occur in a given cube, and retrieve their correspondingly instances’ count in the
cube in constant time. FCI index is used to determine the corresponding cubes in
which a feature occurs and obtain its instances count in constant time. Considering
that two instances are near in the position if and only if their cubes are neighbors, the
instances of a feature $f_i$ in a cube and the instances of a feature $f_j$ in the neighboring
cube form the valid instances of the co-location pattern $\langle f_i, f_j \rangle$. Therefore, the
instances’ count of a feature in a co-location pattern can be obtained from the
summary-structure directly.

[Co-location Episode]. In [10] the proposal is, given a trajectory database, to
find co-location episodes in order to represent the inter-movement regularities among
different types of moving objects. The proposal considers a co-location episode as
a sequence of spatiotemporal co-location events. These events are sets of objects
moving close to each other for a given time period. Besides, there is a particular
object type (centric feature) which participates in a sequence of co-locations. Ac-
cording the proposal, in a valid instance of the episode the object that instantiates
the common feature should be the same in all co-location instances. The proposal
requires a temporal duration for a valid co-location event and searches for temporal
episodes of such events. Basically, the approach to find the co-location episodes is
composed of two steps:

1. By using a hash-based technique, to retrieve the object pairs, whose trajec-
tories are close during some periods and identify these intervals.

2. Through an Apriori-based algorithm to mine the co-location patterns.

The patterns relate two or more trajectories, instead of searching for trajectories
that follow a route defined by a temporal sequence of regions.

[MDCOP]. Mixed-drove spatio-temporal co-occurrence pattern (MDCOP) algo-
rithm was proposed in [12]. The goal is to mine patterns that represent subsets of
object-types that are located together in space and time. The environment of the
proposal is a mixed group of moving objects. The proposal uses the participation
index concept (see definitions 3.4.0.8, 3.4.0.7), presented in [54], in order to compute
the spatial prevalence measure. However, in [54] the goal was just to find spatial co-
location patterns, the dimension time was not considered. In the MDCOP proposal
the authors propose to extend that approach by using a time prevalence measure.

The prevalence measure was presented by the following three definitions:

**Definition 3.4.0.10** Given a spatio-temporal pattern and a set $T$ of time slots, such
that $T = \{T_0, ..., T_{n-1}\}$, the time prevalence measure of the pattern is the fraction of
time slots where the pattern occurs over the total number of time slots.

**Definition 3.4.0.11** Given a spatio-temporal dataset $ST$, and a spatial prevalence
threshold $\theta_p$, the mixed-drove prevalence measure of a spatio-temporal pattern $P_i$
is a composition of the spatial prevalence measure and the time prevalence measure.

\[
\text{Prob}_t \text{time_slot}(s \text{prev(pattern } P_i, \text{ time_slot } t_m) \geq \theta_p)
\]
where \( \text{Prob} \) is the probability of overall prevalence time slots and \( s_{\text{prev}} \) is the spatial prevalence measure.

**Definition 3.4.0.12** Given a spatio-temporal dataset \( ST \) and a threshold pair \((\theta_p, \theta_{\text{time}})\), \( \text{MDCOP} \) \( P_i \) is a mixed-drove prevalent pattern, if its mixed-drove prevalence measure satisfies the following.

\[
\text{Prob}_{\text{t}=\text{all}\text{-time slot}}[s_{\text{prev}}(\text{pattern } P_i, \text{ time slot } t_m) \geq \theta_p] \geq \theta_{\text{time}}.
\]

where \( \text{Prob} \) is the probability of overall prevalence time slots and \( s_{\text{prev}} \) is the spatial prevalence measure, \( \theta_p \) is the spatial prevalence threshold, and \( \theta_{\text{time}} \) is the time prevalence threshold.

The algorithm, first, will discover all size \( k \) spatial prevalent \( \text{MDCOPs} \) and then will apply a time-prevalence based filtering to discover \( \text{MDCOPs} \). Finally, the algorithm will generate size \( k + 1 \) candidate \( \text{MDCOPs} \) using size \( k \) \( \text{MDCOPs} \). The participation index is used as a spatial prevalence interest measure to check if the pattern is spatial prevalent at a time slot. The time prevalence is used as a time prevalence interest measure. The dimension time is the main difference of this proposal. The goal is to discover persistent patterns that co-occur in most but not all spatio-temporal intervals, therefore consecutive occurrences are not mandatory.

### 3.5 Conclusions

The presented approaches are designed to solve the problem of finding a pattern of movement in an environment of moving objects. All the proposals are efficient to find those patterns, each one of them focusing the problem in a specific way. The proposals presented in Sections 3.2 and 3.3 have some common factors:

**Trajectories are represented by a sequence of locations:** Solutions begin with the same initial consideration where the trajectories are represented by a sequence of locations along the time. Those locations are the centers of neighborhood which are used to compute the constraints of the pattern (support, density). For the totality of the proposals it is necessary to adjust a variation around the spatial and time values. That adjustment must be done because is highly unlikely that an object will repeat an identical sequence of locations precisely. Even if the spatial route is precise, the location transmissions at each timestamp are unlikely to be perfectly synchronized. Thus, the object will not reach the same location at the same time every day, and as a result the sampled locations at specific timestamps will be different.

**The patterns does not present a complete route of movement:** All the proposals present a solution where the pattern of the movement does not represent a complete route. The pattern is composed of the locations (neighborhood areas) where it is likely that an object (or a set of objects) will pass at a given
timestamp interval. Therefore, the accuracy of the pattern is related to the size of the neighborhood area and to the timestamp range.

The patterns represent a frequent behavior of movement of a set of trajectories. Some of the presented proposals focus on finding a pattern of movement of a set of trajectories, considering an area and a timestamp range. Those proposals (Sections 3.2 and 3.3) introduce methods to mine directly trajectory data.

However, our TDW does not maintain memory of the original trajectories. The TDW stores summarized values (maximum speed, average speed, traveled distance, acceleration, number of distinct objects) that represent some characteristics of a set of trajectories crossing a given cell defined by \((x,y,t)\) coordinates. Therefore, those proposals could not be used to analyze the data stored in the TDW.

The TDW values may represent some events of the set of trajectories in a given cell. For example, a traffic jam occurrence may be represented by the increase in the number of the objects and the decrease in the average speed of these objects in a given cell \((x,y,t)\).

Therefore, a spatio-temporal data mining technique to be used on a TDW must be able to find pattern of occurrences of events related to a set of trajectories. The proposal is to discover the relationship among occurrences of events considering a given spatial area and a range of timestamp.

The proposals Co-location Miner \[54\] \[31\], Partial Join \[02\], Joinless \[63\], Zoloc-Miner \[11\], TopologyMiner \[59\], Co-location Episode \[10\] and MD-COP \[12\] introduce methods to mine co-location patterns. We consider that these methods, applied to Spatio-temporal features, can be used to mine our TDW. Therefore, we are focused on the use of co-location patterns applied to spatio-temporal environments. An example of that knowledge could be the understanding of the evolution of the traffic jam occurrence. It may be used in order to manage the traffic flow, taking actions to prevent the occurrence of the event. In the next chapter we present a investigation about the use of spatio temporal co-location patterns in order to analyze the TDW data. Besides, we present our proposal of algorithm (Target Spatio-Temporal Co-location Pattern) to mine co-location patterns by using stored TDW values. The proposed algorithm is a first attempt to implement some concepts introduced in \[57\]. In the current version, our algorithm is based on Inter-zone sequential pattern. By using the proposed algorithm it is possible to identify the relationship among the occurrence of events at different areas.
3. Pattern Mining applied to Trajectories
Trajectory Data Warehouse Patterns

We proposed a *Trajectory Data Warehouse* (see Chapter 2) to receive, store and compute trajectory data. The multi-dimensional data model stores aggregate measures computed over huge data volume. A *Trajectory Data Warehouse* represents, through of the measures, characteristics of a set of trajectories crossing a given cell delimited by $x,y,t$ coordinates. The measures are computed by using aggregate functions summarizing the values for each cell($x,y,t$). The values stored in a *Trajectory Data Warehouse* can be used to analyze phenomenon of movement of a set of objects by mining patterns of event occurrence. For example, if stored data are concerned with moving vehicles in a road network, a traffic manager could be interested in analyzing occurrence of Traffic Jam phenomena. Therefore, the previous knowledge about the conditions correlated to a Traffic Jam phenomenon may be used to forecast the occurrence itself, thus allowing a traffic manager to take actions to decrease the effects of the phenomenon. A traffic jam is an example of an event related to the movements of the objects. It may be identified by the decrease in the average speed and the increase in the number of the objects crossing a given cell. In this case we want to investigate the occurrence of a traffic jam phenomenon in a given cell considering the occurrence of another events in the neighborhood of that cell. It allows to forecast the occurrence of the event analyzing the occurrences of related different events and to take actions to solve the problem. In this investigation we focus on developing a method to process the TDW data and finding occurrence of co-location patterns considering an area of search delimited by neighborhood concepts.

This chapter introduces our proposal of algorithm to implement the concept of spatio temporal co-location on TDW data. Before introducing our algorithm, we present an example of the proposed algorithm (see Algorithm 3.4) to mine co-location patterns from TDW data. It was done because all the other proposals of co-location pattern presented in last chapter are based on that algorithm. Therefore, considering that the other proposals introduce, basically, mechanisms to improve the performance of the method, the final result of the example could be similar to the all proposals. Besides, after to introduce our algorithm, we present the results of an experimental evaluation of the algorithm by using the synthetic dataset as data source.
4.1 Spatial co-location patterns

Figure 4.1 illustrates a distribution of data based on our TDW. The dataset is used to present the approach ([54], [31]) to compute the co-location patterns and co-location rules. The grid represents the set of cells at time \( t \) where we have stored several measures in order to represent the characteristics of the set of trajectories crossing that area.

In this example we use three different shapes in order to represent different spatial features types. These spatial features were defined considering the discretized values of the Average speed (see Table 2.11) measure stored in the TDW. We consider only three levels of discretization:

- \( \{\times\} \): \( 0.00 < \text{Disc speed} \leq 0.33 \)
- \( \{+\} \): \( 0.33 < \text{Disc speed} \leq 0.66 \)
- \( \{\ast\} \): \( 0.66 < \text{Disc speed} \)

We have defined the 9 adjacent cells of a given cell \( c \) (including \( c \)) as the neighborhood of \( c \). Table 4.1 presents the size 1 spatial co-locations and the table instances of \( \times \), + and \( \ast \).

![Figure 4.1: Spatial Co-location Patterns - TDW](image)

Table 4.2 presents the size 2 co-locations obtained from the size 1 co-locations, in this case we have three different candidate co-locations: \( \{\times \ast\} \), \( \{\times +\} \) and \( \{\ast +\} \). For each candidate co-locations we have computed participation ratio and participation index values, by using Definitions 3.4.0.7 and 3.4.0.8.
Table 4.1: Spatial Co-location Size 1

<table>
<thead>
<tr>
<th>×</th>
<th>*</th>
<th>+</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1,2)</td>
<td>(1,6)</td>
<td>(1,5)</td>
</tr>
<tr>
<td>(1,3)</td>
<td>(3,3)</td>
<td>(2,5)</td>
</tr>
<tr>
<td>(2,3)</td>
<td>(5,3)</td>
<td>(2,6)</td>
</tr>
<tr>
<td>(4,5)</td>
<td>(6,2)</td>
<td>(3,2)</td>
</tr>
<tr>
<td>(5,2)</td>
<td>(6,3)</td>
<td>(3,4)</td>
</tr>
<tr>
<td>(5,4)</td>
<td>(7,4)</td>
<td>(4,1)</td>
</tr>
<tr>
<td>—</td>
<td>(8,5)</td>
<td>(4,2)</td>
</tr>
</tbody>
</table>

\[ \text{pr}(\{\times\}) = 1 \]
\[ \text{pr}(\{+\}) = 1 \]

The participation ratio of the candidate \{×∗\} was obtained in two steps: the first one computes the participation of the event \{×\} by the division between the number of occurrences of the size 2 co-locations (\{×∗\},×), cells: \{(2,3)(3,3), (5,2)(5,3), (5,4)(5,3), (5,2)(6,2), (5,2)(6,3), (5,4)(6,3)\} and the number of occurrences of size 1 co-location \{×\}, cells: \{(1,2),(1,3),(2,3),(4,5),(5,2),(5,4)\}. Therefore,

\[ \text{pr}(\{\times \}, \times) = \frac{6}{6} = 1.00. \]

Similarly, the second step computes the participation of the event \{∗\} by the division between the number of occurrences of the size 2 co-locations (\{×∗\},∗), cells: \{(3,3)(2,3), (5,3)(5,2), (5,3)(5,4), (6,2)(5,2), (6,3)(5,4), (6,3)(5,2)\} and the number of occurrences of size 1 co-location \{∗\}, cells: \{(1,6),(3,3),(5,3),(6,2),(6,3),(7,4),(8,5)\}. Therefore,

\[ \text{pr}(\{\times \}, \ast) = \frac{6}{7} = 0.85. \]

Finally, the participation index of the candidate co-location \{×∗\} was computed by using the two above computed participation ratio values. Therefore,

\[ \text{pi}(\{\times \ast\}) = \min\{1.00,0.85\} = 0.85. \]

By using a minimum threshold prevalence measure of 0.90, the candidate co-locations \{×∗\} and \{×+\} will be pruned. The algorithm stops at this point,
because it is no longer possible to compose a size 3 candidate co-location starting from the selected candidate co-location \{\ast +\} (anti-monotone property).

Therefore, we can compute two co-location rules starting from the co-location pattern \{\ast +\}:

- \{\ast \Rightarrow +\} with conditional probability \(\text{pr}(\{\ast +\}, \ast) = \frac{7}{7} = 1.00\)
- \{+ \Rightarrow \ast\} with conditional probability \(\text{pr}(\{\ast +\}, +) = \frac{16}{17} = 0.94\)

Considering our example, the above presented co-location rules present a relationship between the occurrence of average speed at level 3 (\ast\) in a given cell and the occurrence of the same measure at level 2 (+) in the neighborhood of the cell.

However, by using this method, we cannot verify the occurrence of events in a given area at given time, considering the occurrence of the another events into the defined neighborhood at another time.

<table>
<thead>
<tr>
<th>(\times \ast)</th>
<th>(\times +)</th>
<th>(\ast +)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(2,3) (3,3)</td>
<td>(2,3) (3,4)</td>
<td>(1,1) (1,5)</td>
</tr>
<tr>
<td>(5,2) (5,3)</td>
<td>(2,3) (3,2)</td>
<td>(1,1) (2,5)</td>
</tr>
<tr>
<td>(5,4) (5,3)</td>
<td>(4,5) (3,4)</td>
<td>(1,6) (2,6)</td>
</tr>
<tr>
<td>(5,2) (6,2)</td>
<td>(4,5) (4,4)</td>
<td>(3,3) (3,2)</td>
</tr>
<tr>
<td>(5,2) (6,3)</td>
<td>(5,2) (6,1)</td>
<td>(3,3) (4,2)</td>
</tr>
<tr>
<td>(5,4) (6,3)</td>
<td>(5,2) (5,1)</td>
<td>(3,3) (4,3)</td>
</tr>
<tr>
<td>pr({\times \ast}, \times) = 1.00</td>
<td>pr({\times \ast}, \ast) = 0.85</td>
<td>pi({\times \ast}) = 0.85</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(5,2) (4,1)</td>
<td>(3,3) (4,4)</td>
</tr>
<tr>
<td>pr({\times \ast}, \ast) = 0.85</td>
<td>pi({\times \ast}) = 0.85</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(5,2) (4,2)</td>
<td>(3,3) (3,4)</td>
</tr>
<tr>
<td></td>
<td>(5,2) (4,3)</td>
<td>(5,3) (4,3)</td>
</tr>
<tr>
<td></td>
<td>(5,4) (4,3)</td>
<td>(5,3) (4,4)</td>
</tr>
<tr>
<td></td>
<td>(5,4) (4,4)</td>
<td>(5,3) (6,4)</td>
</tr>
<tr>
<td></td>
<td>(5,4) (6,4)</td>
<td>(6,2) (6,1)</td>
</tr>
<tr>
<td></td>
<td>(5,4) (6,5)</td>
<td>(6,2) (7,2)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(6,2) (7,3)</td>
</tr>
<tr>
<td></td>
<td>pr({\times +}, \times) = 0.66</td>
<td>pr({\times +}, \ast) = 0.53</td>
</tr>
<tr>
<td></td>
<td>(6,3) (7,2)</td>
<td>(6,3) (7,3)</td>
</tr>
<tr>
<td></td>
<td>(6,3) (7,4)</td>
<td>(7,4) (6,4)</td>
</tr>
<tr>
<td></td>
<td>(7,4) (6,5)</td>
<td>(7,4) (6,4)</td>
</tr>
<tr>
<td></td>
<td>(7,4) (7,3)</td>
<td>(7,4) (8,4)</td>
</tr>
<tr>
<td></td>
<td>(7,4) (8,4)</td>
<td>(8,5) (8,4)</td>
</tr>
<tr>
<td>pr({\ast +}, \ast) = 1.0</td>
<td>pr({\ast +}, \ast) = 0.94</td>
<td></td>
</tr>
<tr>
<td></td>
<td>pi({\ast +}) = 0.94</td>
<td></td>
</tr>
</tbody>
</table>
4.2 Target Spatio-Temporal Co-location Pattern Algorithm

We propose an algorithm (Target Spatio-temporal Co-location Patterns) to identify Spatio-temporal co-location patterns considering an environment of TDW. Consider a set of spatial events \( E = \{e_1, e_2, ..., e_n\} \) representing ST events which may happen in the cells of our TDW. We have to investigate the occurrence of a co-location pattern into a neighborhood area considering an interval of time, starting at time \( t-i \) and finishing at time \( t \).

In order to solve this problem we compute partial prevalence measures \( (pPr_t(E)) \) of the candidate co-locations for each pair of timestamp values \([t - i, t]\), and to compute a global prevalence measure \( (gPr(E)) \) by using the partial prevalences.

\[
gPr(E) = \prod_{j=t-i}^{t} pPr_j(E)
\]

However, this procedure can be very expensive computationally, considering the number of distinct timestamps involved in the process. We propose to reduce this cost limiting the number of candidate co-locations by using the concepts of Candidate and Target Events.

Definition 4.2.0.13 Given a set of events, a set of candidate event is any subset of the events that occurs in a given neighborhood area.

Definition 4.2.0.14 Given a set of events, a target event is a previously defined subset of the co-location pattern.

For example, considering the set of events \( \{\spadesuit, \diamondsuit, *, +\} \) and given that the event \( \spadesuit \) was defined as the target event, the subset \( \{\diamondsuit, *, +\} \) represents the set of candidate events. Therefore, considering a k-size = 1, the possible co-location patterns are the three combinations \( \{\diamondsuit, \spadesuit\}, \{*, \spadesuit\} \) and \( \{+, \spadesuit\} \). All the co-location patterns are composed of an occurrence of a candidate event and another occurrence of target event. The goal of using the target events is to reduce the number of possible co-location event in order to find the co-location pattern. Therefore, the target event concept is a kind of constraint, a general method used to reduce the computational cost of an algorithm [23], [13], [30]. In the previous example, without considering the target event, we could find 6 co-location patterns:\( \{\diamondsuit, \spadesuit\}, \{*, \spadesuit\}, \{+, \spadesuit\}, \{+, *\}, \{+, \diamondsuit\} \) and \( \{*, \diamondsuit\} \). The reduction in the number of the events to combine in order to build the co-location pattern increases the computational performance to find all the co-location patterns in a given area.

Besides, we propose to use another constraint to limit the area investigated by the algorithm. We are considering 2-dimensional space divided into a regular grid composed of a set of 2D cells, according the TDW proposed in [7]. Each cell represents a neighborhood area delimited by the regular grid. The values of measures
stored in the TDW are discretized in order to represent the set of spatial features. In this manner the same cell can store several different boolean spatial features.

The proposed algorithm uses the concept of neighborhood in order to reduce the spatial area of research.

**Definition 4.2.0.15** Given a target event defined by the user, a target cell is the cell where is possible to identify at least one occurrence of the target event. Therefore, it is possible to have many target cells, one for each instance of the target event.

**Definition 4.2.0.16** The value of neighborhood radius represents the number of levels of the neighborhood area.

**Definition 4.2.0.17** The neighborhood radius set is the set of adjacent cells to the given target cell considering the value of neighborhood radius.

**Definition 4.2.0.18** Given a 2-Dimensional space, a MBR search area is the Minimum Bounding Rectangle defined on the space grid in order to limit the area where the algorithm will execute the research.

The following figures represent the concepts of target cell, neighborhood radius and neighborhood radius set. Considering that the cell defined by coordinates \{(5,3)\} in Figure 4.2 is a target cell, Figures 4.3, 4.4 show the neighborhood radius set for a neighborhood radius of 1 and 2, respectively.

![Figure 4.2: Target Cell](image-url)
4.2. Target Spatio-Temporal Co-location Pattern Algorithm

Figure 4.3: Level 1

Figure 4.4: Level 2
After defining and computing the Target Events, the procedure uses the definitions proposed in [54]. We have to compute participation ratio and participation index values for each pair of timestamps. This computation is reduced by using the Target Events, we just have to compute the participation ratio and index values for the Target Events. The algorithm has two main phases: (i) Definition of the search spatio-temporal area and (ii) Computation of the space and time prevalence values.

In the first step we delimitate the area where the search of the patterns will be executed. Besides, we delimitate the target neighborhood areas. Figure 4.5 presents a view of fundamental concepts (MBR Search Area, 2 Target Cells and 2 Target Neighborhood Area) of the algorithm on an area divided according to a regular grid. The target neighborhood area is a set of disjoint areas obtained from the neighborhood of the cells where happens an occurrence of a target event. Target Events are events defined by the user in order to execute the algorithm. For example, when the goal is to find a co-location occurrence composed of traffic jam event and a set of other events, traffic jam is defined as a target event. The computation of the prevalence values (space and time) by using the values stored in cells involved by the target neighborhood area it is performed in the second step.

Figure 4.5: Target Spatio-Temporal Colocation Concepts

Algorithm 3 presents the pseudo-code of the Target Spatio-Temporal Co-location Pattern Algorithm. In the following we discuss the details of each step.
Algorithm 3 Target Spatio-Temporal Co-location Pattern Algorithm

Input
- A relation $R$ where each record represent a specific cell $x, y, t$ and the discretized values for each TDW measure.
- User defined Target Events Set
- A user defined Neighborhood radius
- Timestamp Values
- MBR search area
- A user specified minimum threshold prevalence measure ($\text{min\_prevalence}$)

Output
- Co-location patterns with partition index $> \text{min\_prevalence}$

Method
1: Generate search area
2: Find the cells (Target Cells) with at least an occurrence of the defined Target Events
3: Find the neighborhood area related for each Target Cells
4: Generate Candidate Events size 1
5: for size of co-locations in $(1,2,3,..., K - 1)$ do
6: Generate candidate prevalent co-locations by using the previous co-location pattern stored in the Frequent Pattern Table.
7: Compute the space and time prevalence measures for each candidate co-location pattern and prune by using the (min\_prevalence) value.
8: end for

Step 1 By using the input values, the first step is to generate the search area that will be used to limit the space area to find co-location patterns. The coordinates that limit the research area are stored in the Search Area Table, it is a structure used to search the delimited area for the complete timestamp interval.

Step 2 Once defined and stored the boundaries of the research area, in this step it is executed the process to select the Target Cells. This process, by using the target timestamp and the boundaries of the Search area, searches the cells where at least one occurrence of the Target Event happens. These values (cell coordinates) will be stored in a structure named Target Cells Table.

Step 3 In this step we define the neighborhood area for each record stored in Target Cells Table. To this end, we use the Neighborhood radius value to limit the neighborhood area for each target cell. These values are stored in a structure named Target Neighborhood Area Table.

Steps 4-5 In these steps we complete the process of finding Candidate Events size 1. The first point is to select (for each timestamp) the candidate events for each neighborhood area. These candidate events are the source to compute the prevalence measures for each candidate co-location composed by using
the Target Events and the candidate events. The candidate co-location patterns size 1 will be selected considering the user-defined prevalence threshold (\text{min\_prevalence}). The candidate co-location patterns size 1 with prevalence value above or equal to the \text{min\_prevalence} will be stored in a structure named \textit{Frequent\_pattern}. These values are the data source to the next steps of the algorithm.

Steps 5-8 A loop to compose the candidate co-location patterns starting from the candidate co-location size 1. Steps 6 and 7 are complementary, each record produced in the Step 6 it is complemented by an execution of the Step 7. Frequent Pattern Table is ordered by size of co-location (k) and co-location pattern. In Step 6 each value of co-location pattern with co-locationsize = k−1 it is combined with the consequent co-location pattern in order to compose a candidate co-location pattern with co-locationsize = k. Then, for each candidate co-location pattern size k, the Step 7 computes the space and time prevalence measures. This computation is executed considering the limited search area and the neighborhood area. Considering that the Frequent Pattern Table is ordered, the number of items used to compute each execution of the Step 6 decreases after each execution.

In the next Section we present an example of execution of the algorithm. It considers three timestamps and three different TDW values. For simplicity, the example uses just one TDW measure for cell.

### 4.3 Execution Trace of Target Spatio-Temporal Co-location Pattern Algorithm

Figures 4.6, 4.7 and 4.8 represent the MBR search area at three different timestamps, they are used to explain our method. We consider an environment composed of a set of events \{A, B, C, T\} \textit{(Candidate Events)} and their instances over the spatio-temporal area: \text{A}\{A.1, A.2, A.3, A.4\}, \text{B}\{B.1, B.2, B.3, B.4, B.5, B.6, B.7\} and \text{C}\{C.1, C.2, C.3, C.4, C.5\}.

In this case, the event (T) is the Target Event. We are interested in finding a co-location pattern composed of different events occurring in the time interval [t−i, t]. The distance between events are defined in terms of both time and space. Figures 4.6, 4.7 and 4.8 present the environment at times (t−2), (t−1) and (t), respectively. They present the area divided into a regular grid. In this example we consider only one instance of the event for each cell. The line between two instances represents a neighbor relationship. The cells where the Target Event (T) happens at time t (Step 11) are used to define the neighborhood area (Step 12). The bold lines around the cells with an occurrence of the instance T (Target cell) represent the boundaries of the neighborhood radius set area (\text{AreaI}, \text{AreaII} and \text{AreaIII}). Besides,
4.3. Execution Trace of Target Spatio-Temporal Co-location Pattern Algorithm

figures present three boxes with the computed value of participation ratio for each candidate co-location pattern. In this case the goal is to find a co-location pattern by using the $T$ event at time $t$ as the Target Event. We are considering a threshold measure of prevalence of 0.55.

For each figure, the space distance component represents the 9 adjacent cells around each Target cell. The figures present three Target cells and three neighborhood radius set areas, one for each Target cell, considering area composed of 9 adjacent cells.

In Figure 4.6, the time distance component represents the environment at time $(t-2)$. Considering the event $T$ as the Target Event, it is possible to find three different candidate co-location: $\{T, A\}$, $\{T, B\}$ and $\{T, C\}$. In the following the candidate instances for each area of Figure 4.6:

AreaI $\{A.1, B.1, B.2, T\}$

AreaII $\{A.3, B.5, B.6, B.7, T\}$

AreaIII $\{C.1, T\}$

The computation of Participation ratio for each candidate co-location it is presented in the following:
4. Trajectory Data Warehouse Patterns

- $Pr(\{T, A\}, A) = \frac{2}{4}$, 2 out of 4 instances of A (A.1,A.3) is contributing to the co-location $\{T, A\}$

- $Pr(\{T, B\}, B) = \frac{5}{7}$, 5 out of 7 instances of B (B.1,B.2,B.5,B.6,B.7) is contributing to the co-location $\{T, B\}$

- $Pr(\{T, C\}, C) = \frac{1}{5}$, 1 out of 5 instances of C (C.1) is contributing to the co-location $\{T, C\}$

In Figure 4.7, the time distance component represents the environment at time $(t-1)$. Considering the event $T$ as the Target Event, it is possible to find three different candidate co-location: $\{T, A\}$, $\{T, B\}$ and $\{T, C\}$. In the following the candidate instances for each area of Figure 4.7:

**AreaI** $\{A.1, A.2, B.1, T\}$

**AreaII** $\{A.3, B.5, B.6, B.7, T\}$

**AreaIII** $\{C.1, C.2, T\}$

The computation of Participation ratio for each candidate co-location it is presented in the following:
4.3. Execution Trace of Target Spatio-Temporal Co-location Pattern Algorithm

- \( Pr(\{T, A\}, A) = \frac{3}{4} \), 3 out of 4 instances of A (A.1, A.2, A.3) is contributing to the co-location \( \{T, A\} \)
- \( Pr(\{T, B\}, B) = \frac{4}{7} \), 4 out of 7 instances of B (B.1, B.5, B.6, B.7) is contributing to the co-location \( \{T, B\} \)
- \( Pr(\{T, C\}, C) = \frac{2}{3} \), 1 out of 5 instances of C (C.1, C.2) is contributing to the co-location \( \{T, C\} \)

In Figure 4.8, the time distance component represents the environment at time \((t)\). Considering the event \( T \) as the Target Event, it is possible to find three different candidate co-location: \( \{T, A\}, \{T, B\} \) and \( \{T, C\} \). In the following the candidate instances for each area of Figure 4.8:

**AreaI** \( \{A.1, A.2, B.1, T\} \)

**AreaII** \( \{A.3, B.5, B.6, B.7, T\} \)

**AreaIII** \( \{C.1, C.2, T\} \)

The computation of Participation ratio for each candidate co-location it is presented in the following:
4. Trajectory Data Warehouse Patterns

- $Pr(\{T, A\}, A) = \frac{3}{4}$, 3 out of 4 instances of $A$ (A.1,A.3,A.4) is contributing to the co-location $\{T, A\}$

- $Pr(\{T, B\}, B) = \frac{6}{7}$, 6 out of 7 instances of $B$ (B.1,B.2,B.4,B.5,B.6,B.7) is contributing to the co-location $\{T, B\}$

- $Pr(\{T, C\}, C) = \frac{2}{5}$, 2 out of 5 instances of $C$ (C.2,C.3) is contributing to the co-location $\{T, C\}$

At this point, the algorithm has three different values of participation ratio for each candidate co-location. Each of them computed for a different timestamp, in this example we considered three timestamps: $\{t-2, t-1, t\}$. The next step is to compute the value of participation index for each candidate co-location ({$T, A$}, {$T, B$}, {$T, C$}). According to the proposed algorithm it can be computed by using the minimum value among of them (see Definition 3.4.0.8). In the following we present the computation of the participation index for each candidate co-location pattern:

- $Pi(\{T, A\}) = \min(Pr_{t-2}(\{T, A\}, Pr_{t-1}(\{T, A\}, Pr_{t}(\{T, A\}))$ = $\frac{2}{4}$

- $Pi(\{T, B\}) = \min(Pr_{t-2}(\{T, B\}, Pr_{t-1}(\{T, B\}, Pr_{t}(\{T, B\}) = \frac{4}{7}$

- $Pi(\{T, C\}) = \min(Pr_{t-2}(\{T, B\}, Pr_{t-1}(\{T, B\}, Pr_{t}(\{T, B\}) = \frac{1}{5}$

The following step is to compute the time component of the participation index (prevalence time) for each candidate co-location pattern. It is computed by the division between the number of timestamps where the candidate co-location pattern occurs and the total number of timestamps. This value represents a fundamental knowledge about the pattern, it is possible to understand the weight of the pattern along the time interval.

- $Pt\{T, A\} = \frac{3}{3} = 1.0$

- $Pt\{T, B\} = \frac{3}{3} = 1.0$

- $Pt\{T, C\} = \frac{3}{3} = 1.0$

Finally, by using the values of participation index and prevalence time it is possible to compute the Global Participation Index ($GPI$) for each candidate co-location pattern.

- $GPI\{T, A\} = Pi(\{T, A\}) \times Pt\{T, A\} = \frac{2}{4} \times \frac{3}{3} = \frac{2}{4} = 0.50$

- $GPI\{T, B\} = Pi(\{T, B\}) \times Pt\{T, B\} = \frac{4}{7} \times \frac{3}{3} = \frac{4}{7} = 0.57$

- $GPI\{T, C\} = Pi(\{T, C\}) \times Pt\{T, C\} = \times \frac{1}{5} \frac{3}{3} = \frac{1}{5} = 0.20$
After computing the *Global Participation Index* for each candidate co-location, the next step is pruning. It is done comparing the threshold value of prevalence with the *global participation index* computed for each candidate co-location. In this example the threshold value of presence is 0.55. Therefore, the only candidate co-location selected is \{T, B\}, its GPI is 0.57.

That pattern means that an B event at timestamp t - 2 and t - 1 has a relationship (into a neighborhood) with a T pattern starting at time t with a prevalence level of 0.57. However, we can not affirm that all the occurrences of pattern T at timestamp t are originated at timestamps t-2 or t - 1.

In this example we have defined a Target Event composed by only one TDW measure. We have done it in order to facilitate the explanation of the method. However, the Target Event can be composed of several TDW measures.

Therefore, by using this model it is possible, for example, to investigate the behavior of a Traffic Jam Event along the time considering a given area of research. A Traffic Jam event (in this case a Target Event) may be defined by using, for example, two discretized TDW measures: *level of speed* (See Equation 2.7) and *density of presence* (See Equation 2.6). In this manner the algorithm may reveal the evolution of a Traffic Jam occurrence by using the TDW measures.

Table 4.3: Prevalence Time × Participation Index

<table>
<thead>
<tr>
<th>TimeStamp</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
</tr>
</thead>
<tbody>
<tr>
<td>t - 1</td>
<td>0.90</td>
<td>0.90</td>
<td>0.60</td>
<td>0.80</td>
<td>0.20</td>
<td>0.60</td>
<td>0.90</td>
</tr>
<tr>
<td>t - 2</td>
<td>0.50</td>
<td>0.80</td>
<td>0.90</td>
<td>0.90</td>
<td>0.80</td>
<td>0.80</td>
<td></td>
</tr>
<tr>
<td>t - 3</td>
<td>0.70</td>
<td>0.50</td>
<td>0.90</td>
<td>0.70</td>
<td>0.80</td>
<td></td>
<td></td>
</tr>
<tr>
<td>t - 4</td>
<td>0.40</td>
<td>0.90</td>
<td>0.90</td>
<td>0.80</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>t - 5</td>
<td></td>
<td>0.90</td>
<td>0.80</td>
<td>0.70</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>t - 6</td>
<td></td>
<td></td>
<td>0.90</td>
<td>0.80</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>t - 7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| Participation Index (Pi) | 0.90 | 0.50 | 0.60 | 0.40 | 0.20 | 0.60 | 0.70 |
| Prevalence Time (Pt)     | 0.14 | 0.29 | 0.43 | 0.57 | 0.71 | 0.86 | 1.00 |

| Global Participation Index (Pi × Pt) | 0.13 | 0.14 | 0.25 | 0.22 | 0.14 | 0.51 | 0.70 |

*Prevalence time* is a fundamental concept of the proposed algorithm. This value measures the occurrence of the pattern along the time dimension. It is the quotient between the number of timestamps that the pattern occurs and the number of timestamps. Table 4.3 presents an example to explain the importance of the prevalence time. The example presents values of participation ratio for 7 hypothetic co-location patterns (A,B,C,D,E,F and G) by using 7 different timestamps. Given a prevalence measure of 0.30, and considering just the Participation Index measure, cases A, B,C,D,F and G are considered valid patterns. However, when *Time prevalence* is computed the situation is totally different, in our example, in this case, only cases F and G are considered valid. The row Global Participation Index presents
the final participation index computed by using both Space and Time prevalence. That simple example presents the importance of the prevalence time to compute the global participation index.

4.4 Experimental Evaluation

In this section, we discuss the results of some experiments, aimed at evaluating the performance of the proposed algorithm. Besides, in Section 4.4.4 we present some co-location patterns obtained in the same experiment. We consider three topics of evaluation: Number of Timestamps, Spatio-Temporal Prevalence Index Threshold and Target Event Size. We conducted the experiments on an Intel Core2 1.66 GHz computer with 1.0 GB of RAM. The dataset which we have used to complete the experiments is based on the proposed TDW where each record represents characteristics of a set of trajectories crossing a cell. The dataset is composed of 17417 records (cells), each of them storing the following attributes: distance (102 different discretized values), time (44 discretized values), acceleration (259 discretized values), density of presence (19 discretized values) and level of speed (13 discretized values). Table 4.4 presents an example of the records stored in the dataset. Columns X coordinate and Y coordinate represent the spatial coordinates of the cell, column TimeStamp represents the timestamp value of the cell and, finally, the Measures - Discretized Values column represents the discretized values of the different measures (distance, time, acceleration, density of presence, level of speed) for the cell.

<table>
<thead>
<tr>
<th>X coordinate</th>
<th>Y coordinate</th>
<th>TimeStamp</th>
<th>Measures - Discretized Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>13000</td>
<td>20500</td>
<td>1</td>
<td>dist1 time0 accel108 dpre0 lspe1</td>
</tr>
<tr>
<td>12500</td>
<td>20000</td>
<td>2</td>
<td>dist8 time2 accel13 dpre1 lspe0</td>
</tr>
<tr>
<td>12000</td>
<td>19000</td>
<td>3</td>
<td>dist16 time5 accel7 dpre1 lspe0</td>
</tr>
<tr>
<td>11500</td>
<td>18500</td>
<td>4</td>
<td>dist13 time5 accel7 dpre1 lspe0</td>
</tr>
<tr>
<td>11000</td>
<td>18500</td>
<td>5</td>
<td>dist2 time0 accel3 dpre0 lspe0</td>
</tr>
</tbody>
</table>

We defined five different areas (area 1, area 2, ..., area 5) of the 2-Dimensional space in order to execute the experiments. These areas represent the MBR search area. It was done in order to decrease the possibility of finding a valid relationship on a restrict area and to generalize it for the complete area.

4.4.1 Number of Timestamps

The goal is to evaluate the behavior of the algorithm by varying the number timestamps. We consider the following relationships:
4.4. Experimental Evaluation

**Number of timestamps vs time of execution** The goal is to investigate how the number of timestamps can be related to the execution time of the algorithm.

**Number of timestamps vs Number of Range Cells** Investigate the relationship between the number of timestamps and the number of the cells involved by the *neighborhood radius sets*.

**Number of timestamps vs Number of Candidate Events** Investigate the relationship between the number of timestamps and the number of candidate events involved by the *neighborhood radius sets*.

**Number of timestamps vs Number of co-location patterns** Investigate whether there exists some relationship between the number of timestamps and the number of found co-location patterns.

Figure 4.9 shows the relationship between the number of *timestamps* and *time of execution* of the algorithm. The *execution time* increases when the number of timestamps increase. It is an expected conclusion, considering that the prevalence value of the algorithm has a prevalence time component.

In Figure 4.10 is possible to observe a similar behavior, the number of cells involved by the *neighborhood radius sets* increases when the number of timestamps increases.

The same behavior was identified by Figure 4.11 where the growth in the number of timestamps it is related to an increase in the number of *candidate events*. It is an expected behavior, considering that different timestamps show different snapshots of the environment at a given area. Therefore, the growth of the number of timestamps, considering spatio-temporal cells, implies in the growth of the number of *Range Cells*. Finally, Figure 4.12 presents a different behavior: when the number of timestamps increases it happens a decrease in the number of found patterns. It can be explained considering that the patterns are not constant along the time dimension. It is a very important concept, the time dimension is a fundamental dimension in order to analyze the summarized data of a *TDW*. The decrease of the number of the
patterns along the time dimension it confirms the evolution of the environment and justifies the temporal component of the prevalence index. A valid pattern at a given timestamp $t$ may be invalid at a timestamp $t + i$. Therefore, an increment in the number of timestamps may result in the decrease of the probability of finding a pattern. Another interesting conclusion is that the execution time of the algorithm does not have relationship with the number of patterns found by the algorithm.

### 4.4.2 Spatio-Temporal Prevalence Index Threshold

In this case the goal is to investigate the relationship among the value of Spatio-Temporal Prevalence Index Threshold and the evolution of the value of time execution and the number of patterns. An evident conclusion is that the value of Spatio-Temporal Prevalence Index Threshold does not have any relationship with Range Cells and Candidate Events. Therefore, we did not investigate the relationship among Spatio-Temporal Prevalence Index Threshold and those items.

Figures 4.13 and 4.14 allow to verify that low values of spatio-temporal prevalence measure are related to an increase in the number of patterns found by the algorithm. Another effect of the low values of prevalence measure it is an increase
in the execution time of the algorithm. Therefore, it is possible to conclude that the performance of the algorithm has a strong relationship with the number of the patterns found by the algorithm, it is also an expected behavior of the algorithm.

### 4.4.3 Target Event Size

In this topic the goal is to investigate the relationship among the size of the Target Event and the following features: Execution Time, Range Cells (Number of the cells involved by the neighborhood radius sets), Candidate Events and Number of Patterns. We consider the number of different events that compose the Target Event as the size of the Target Event. Table 4.5 presents some examples of the size for the different Target Events.

<table>
<thead>
<tr>
<th>Target Event</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>{dpre01}</td>
<td>1</td>
</tr>
<tr>
<td>{dpre01,lvlspeed10}</td>
<td>2</td>
</tr>
<tr>
<td>{dist8, time2, acce13}</td>
<td>3</td>
</tr>
</tbody>
</table>

Figure 4.15 shows the evolution of the execution time along the different values of the Target Event Size. The behavior is the same for the all areas: the Execution Time decreases (↓) when the size of the Target Event increases (↑).

![Figure 4.15: Size vs Execution Time](image)

![Figure 4.16: Size vs Range Cells](image)

It can be explained considering the behavior presented in Figures 4.16 and 4.17. The probability of finding a large number of pattern increases when the size of the pattern decreases (Figure 4.17). Considering that the Range cells set has a linear relationship with the number of Target Cells, it is possible to conclude that a growth in the Target Event Size results in a decrease in the number of Range Cells (Figure 4.16).
Another interesting behavior is presented in Figure 4.18. The number of Candidate Events remains stable along the evolution of the size of the Target Event. It is an expected behavior considering that we search the same area (MBR Search area) for the different size values of the Target Event.

### 4.4.4 Co-location Patterns

This section presents some co-location patterns obtained by using the proposed algorithm. We have used the same dataset described in Section 4.4. The goal is to investigate whether it is possible to find some co-location patterns considering two different movement phenomena: Traffic Jam and Free Traffic. We have used a neighborhood radius of 1 and a timestamp range composed of three different timestamps. Table 4.6 presents some obtained patterns.

<table>
<thead>
<tr>
<th>Candidate Events</th>
<th>Target Event</th>
<th>GPI</th>
</tr>
</thead>
<tbody>
<tr>
<td>{dist0 dpre0}</td>
<td>{Free}</td>
<td>0.83</td>
</tr>
<tr>
<td>{dist12 dpre0}</td>
<td>{Free}</td>
<td>0.60</td>
</tr>
<tr>
<td>{dist19 dpre0}</td>
<td>{Free}</td>
<td>0.66</td>
</tr>
<tr>
<td>{time0 dpre0}</td>
<td>{Free}</td>
<td>0.77</td>
</tr>
<tr>
<td>{time1 dpre0}</td>
<td>{Free}</td>
<td>0.60</td>
</tr>
<tr>
<td>{075TJ lspe0}</td>
<td>{TJ}</td>
<td>1.00</td>
</tr>
<tr>
<td>{time11 dpre16}</td>
<td>{TJ}</td>
<td>0.66</td>
</tr>
<tr>
<td>{time12 dpre16}</td>
<td>{TJ}</td>
<td>0.66</td>
</tr>
<tr>
<td>{dist56 dpre16}</td>
<td>{TJ}</td>
<td>0.66</td>
</tr>
<tr>
<td>{dpre5 acce3 lspe0}</td>
<td>{TJ}</td>
<td>0.66</td>
</tr>
<tr>
<td>{time7 acce3 lspe0}</td>
<td>{TJ}</td>
<td>0.50</td>
</tr>
</tbody>
</table>

It is evident that the co-location patterns can suffer changes regarding the area of search. Patterns may be modified considering different values of neighborhood radius and timestamp range. However, at this point we only are interested in investigate
whether the algorithm is able to present co-location patterns. In the following we remember the discretization of the measures of the dataset:

- distance: 102 different discretized values
- time: 44 discretized values
- acceleration: 259 discretized values
- density of presence: 19 discretized values
- level of speed: 13 discretized values

Analyzing the co-location patterns when the target event is Free Traffic, it is possible to notice that low values of density of presence \((dpre0)\) combined with low values of time \((time0, time1)\) can be considered an indication of a Free Traffic event. It can describe an environment composed of a reduced number of fast moving objects. In this case few objects remains into the region during short time intervals. It is an evident indication of a possible occurrence of Free Traffic.

Another pattern also presents low values of density of presence \((dpre0)\), however in this case, the co-location pattern is composed of low values of distance \((dist0, dist19)\). This pattern may be a representation of an environment composed of few slow objects. Also in this case the objects could movement without any constraint. The reduced average value of traveled distance can be explained considering an environment composed of slow objects.

Meanwhile, the co-location patterns related to a Traffic Jam target event reveals different behavior. It is possible to identify environments composed of a high number of moving objects \((dpre16)\). Those objects remain during a reasonable time interval \((time11, time12)\) into the area, traveling a significative distance \((dist56)\) into the same cell. This behavior may be the source of an event of Traffic Jam. Besides, it is possible to identify two new patterns of Traffic Jam occurrence. The first one is represented by a significative number of slow objects \((lspe0)\), with low values of acceleration \((acce3)\) moving into a given area, in this case is possible to affirm that this environment can be the source of a Traffic Jam event. The second environment is represented by slow objects \((lspe0)\) moving with a reduced acceleration \((acce3)\) during a considerable time interval \((time7)\). Also in this case it may be considered as an indication of a possible Traffic Jam occurrence.

With the preliminary results is possible to conclude that the proposed algorithm is able to search and find co-location patterns in an environment composed of Spatio-temporal cells. The previous results present some co-location patterns obtained through an inter-zone analysis. Therefore, the algorithm is able to find relationship among occurrences of events located in different zone \((cell)\). It is a very important knowledge when the goal is to forecast the occurrence of events considering the previous occurrence of another events at different zones.
Conclusions

The recent development of mobility technology (mobile computation, cellular phone, remote sensors etc) allows the monitoring of the behavior of moving objects. The movement of an object can be described by using its trajectory. The analysis of a trajectory database allows to reveal hidden knowledge about some phenomena of movement. The knowledge discovery area has been deeply developed in the last years. Techniques, concepts and methods to store and manage data are fundamental to allow the analysis of the large data volumes. Therefore, database and data warehouse technologies are the base to execute a process of knowledge discovery. Besides, the data mining technology is another indispensable tool to implement a complete process to reveal hidden knowledge from data volumes. In the recent years, some researches in the usage of the data mining techniques applied to trajectory databases were produced. Those researches assume that the trajectory database stores the memory of the movement of the object. Each record of the database represent a spatio-temporal position of the moving object.

However, in this thesis we have analyzed the problem to mine and aggregate measures concerning trajectories of moving objects. We have assumed that the database does not maintain the memory of the trajectories for each moving object. We have proposed a multi-dimensional data model to store aggregate measures computed over trajectory data. This allowed us to define a Trajectory Data Warehouse (TDW) that is loaded by managing and transforming a data stream of spatio-temporal observations of moving objects, arriving in a irregular and unbounded way. We have used a traditional star schema composed of two spatial dimensions and one temporal dimension. The base cuboid is composed of spatio-temporal cells, consisting of regions and time intervals. We have associated concepts hierarchies with spatial and temporal attributes. The spatial cube was built as the lattice of cuboids, where the lowest one references all the dimensions at the primitive abstraction level. The others are obtained by summarizing on different subsets of the dimensions, and at different abstraction levels along the concept hierarchy. The Trajectory Data Warehouse allows to find aggregate measures regarding the trajectories involved in a given area and time interval. Those measure represents properties about the trajectories. We have introduced an aggregated function called Presence. This function represents the count of the distinct trajectories crossing a given cell, it is a holistic function. This type of function, when applied to a data stream source are often computed in an approximated way. We introduced two approximate functions in order to compute the measure Presence: Distributive and Algebraic. These alternative functions only need a few/constant memory size for maintaining the information.

We have developed a prototype in order to validate the proposed TDW and the
associated measures. The prototype implements all the concepts and features of the Trajectory Data Warehouse. We have conducted some experiments in order to measure the accuracy of the presence measure. Besides, we also have investigated the time taken to load the cube of the TDW, it is a fundamental characteristic in order to implement the TDW schema by using commercial data warehouse products. All the obtained results allow to affirm that the proposed Trajectory Data Warehouse offers an environment to receive, process and store trajectory data by using the multidimensional concept. Besides, considering that the TDW works on a data stream environment, it is an adequate solution to the data stream features. Future works may investigate the loading phase of the TDW. It is an opened problem, we have limited the loading phase considering a linear interpolation. However, it is possible to find topological constraints where the proposed interpolation does not solve the problem. Besides, the improvement of the current dimensions is a point to be investigate in the future works. The idea is to investigate the implementation of the complete spatial dimensions defined in [60]. It could be useful to answer queries involving a geometric operation such as a spatial union, a spatial merge or a spatial intersection. Another investigation is the possibility of the development of a query language using OLAP operators.

We have done an investigation to verify whether the proposed TDW allows, by analyzing the stored measures, to reveal the occurrence of events in a given area. Actually, we have considered a Traffic Jam (TJ) occurrence. We have proposed to investigate a TJ occurrence by using two measures to represent the context of a given cell: level of speed and density of presence. These measures can be computed on the basis of those stored in the TDW along with the statically available measures. The results obtained in the experiments allow to consider the measures density of presence and level of speed enough to investigate the occurrence of TJ events. We have obtained a high level of accuracy to find TJ occurrences considering the cells whose level of traffic jam is into the following intervals: [0.00%, 25.00%] or [75.00%, 100.00%]. Therefore, a possible future research may be to investigate new methods/algorithms to improve the accuracy of results analyzing cells whose level of TJ is into the interval (25.00%, 75.00%).

We proposed an algorithm to mine co-location patterns of events by using the Trajectory Data Warehouse model as data source. We considered that an event in an environment of trajectories of moving objects can be represented by a combination of the TDW measures. The algorithm is based on the proposals presented in [12], [11] and [31]. The proposed Target Event Co-location Pattern algorithm was defined to work by using the data stored in a Trajectory Data Warehouse. The goal is to find, in a delimited area defined by the user, occurrences of co-location patterns among events of movement of objects. In order to reduce the computational cost involved in this process the algorithm finds the complement of a co-location pattern by using the concepts of target and candidate event. We used the concept of MBR search area in order to reduce the area of search. It was done in order to reduce the computational cost to execute the algorithm. The algorithm allows to forecast the
occurrence of events of movement in a given area investigating the occurrences of another related events that occur in the neighborhood area.

We done several experiments in order to investigate the relationship among several features: Number of Timestamps, Spatio-Temporal Prevalence Index Threshold, Target Event Size, Execution Time, Number of Range Cells, Number of Candidate Events and Number of co-location patterns. The results confirm the importance of the temporal component in the Spatio-temporal prevalence measure. The temporal component allows to investigate the occurrence of a co-location pattern along the time dimension. It is possible to define a time-window to investigate the pattern, and find patterns that are prevalent at a given area and also prevalent along the define time-window. Besides, it reduces the computational cost to execute the algorithm reducing the number of candidate events to compose the final co-location pattern.

The mined patterns are useful to improve the capability of analysis of a TDW environment. The patterns allow to understand the relationship among the events along the time at different areas. These patterns may reveal, for example, different relationships among antecedent and consequent (target) events at different areas. The antecedent event for a same target event may be different at different areas or different timestamps.

We implemented the algorithm by using the measures defined in the our proposed TDW. However, another different measures may be used in the algorithm, one of them it is correlation. Correlation indicates the strength and direction of a linear relationship between two variables. An example of the usage of the correlation measure may be to identify an occurrence of a traffic jam event when the values of density of presence increases (↑), and the values of level of speed decreases (↓). The correlation measure may be useful to find more detailed relationship among the measures of the TDW.

Another research point to be investigated it is the time constraint. We are focus on the definition of the time interval between successive events occurrences. This constraint allows to improve the capability of anticipating the occurrence of a given pattern. A same complete pattern (antecedent and target events) may have different behavior considering the time gap between them. For example: the pattern \( \langle A, BC \rangle \) at a given area may have time gap between the events of 10 minutes. Meanwhile, at a different area the same pattern may have a time gap of 70 minutes. In this case, the same pattern can reveal different events. Therefore, the time constraint is a very important point to be exploited in the future research.
Bibliography


List of PhD Theses

TD-2004-1 Moreno Marzolla
  "Simulation-Based Performance Modeling of UML Software Architectures"

TD-2004-2 Paolo Palmerini
  "On performance of data mining: from algorithms to management systems for
data exploration"

TD-2005-1 Chiara Braghin
  "Static Analysis of Security Properties in Mobile Ambients"

TD-2006-1 Fabrizio Furano
  "Large scale data access: architectures and performance"

TD-2006-2 Damiano Macedonio
  "Logics for Distributed Resources"

TD-2006-3 Matteo Maffei
  "Dynamic Typing for Security Protocols"

TD-2006-4 Claudio Silvestri
  "Distributed and Stream Data Mining Algorithms for Frequent Pattern Dis-
covery"

TD-2007-1 Marco Giunti
  "Secure Implementations of Typed Channel Abstractions"

TD-2007-2 Francesco Lelli
  "Bringing Instruments to a Service-Oriented Interactive Grid"

TD-2007-3 Matteo Mordacchini
  "Grid and Peer-to-Peer Resource Discovery Systems"

TD-2008-1 Claudio Lucchese
  "High Performance Closed Frequent Itemsets Mining inspired by Emerging
Computer Architectures"

TD-2008-2 Giulio Manzonetto
  "Models and theories of lambda-calculus"

TD-2009-1 Fernando José Braz
  "Warehousing and Mining Aggregate Measures Over Trajectories of Moving
Objects"
TD-2009-2 Andrea Marin
"On the relations among product-form stochastic models"

TD-2009-3 Samuel Rota Bulò
"A game-theoretic framework for similarity-based data clustering"