Knowledge Discovery on Trajectory Data Warehouses: Possible usage of the Data Mining Techniques

Fernando J. Braz[1,2]

[1]Department of Computer Science – Ca' Foscari University - Venice - Italy

[2]Department of Informatics – University of Region of Joinville
Joinville, Brazil.

fbraz@dsi.unive.it

Abstract. *In this paper we are interested in discussing the possibility of the usage of Data Mining tasks in order to reveal knowledge resident in Trajectory Data Warehouses (TDW). We consider a data stream environment where a set of mobile objects send the data about its location in a irregular and unbounded way. The data volume is stored in a centralized and traditional DW with pre-computed aggregations values (preserving the trajectories privacy). Through of analysis of the TDW measures (pre-computed aggregation values) we can reveal some characteristics about trajectories in a given spatio-temporal area. The revealed knowledge can be useful in order to describe or show the occurrence of a real phenomenon. We present a review of a proposed structure of a TDW and discuss the use of Data Mining tasks to improve the analysis of the trajectory data warehouse environment.*

1. Introduction

The data warehouse traditional model can be resumed as a subject-oriented data collection integrated from various operational databases. On a data warehouse model, the data are 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 may improve the difficulty to build and maintain a data warehouse. The data arriving on an unpredictable and continuous rate, the larger data volume and the resource constraints (memory, processing) are some of these main characteristics. The data warehouse traditional model must be adapted in order to work in agreement with these constraints.

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 could be stored on a multidimensional model in order to allow an accuracy analysis, it 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 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 trajectory data of moving objects arrive in an unbounded and unpredictable way, it is a characteristic that must be considered in the building of a multidimensional data warehouse model.

The data about trajectories stored in a TDW are the source of a very interesting knowledge which can reveal patterns of occurrence of phenomenons represented by a given subset of the measures of the fact table. Therefore, the Data Mining tasks could be used in order to improve the analysis process. However, considering the characteristics of the trajectory environment, some adaptations in the traditional methods of mining can be necessary.

In the Section 2 we present a briefly review about a proposal by the implementation of a Trajectory DW model, the main problems and the DW model used in order to store the trajectories are also presented. A proposed application ([Braz and Orlando 2007]) in order to allow the implementation of the proposal reviewed in the Section 2 is commented in the Section 3. The Section 4 presents our discussion about of usage of Data Mining techniques in order to reveal the resident knowledge in TDW. Finally, in the Section 5 we draw some conclusions and possible future research topics.

2. The Trajectory DW Model

There are some proposals of spatial data warehouses [Han et al. 1998],[Rivest et al. 2001],[Marchant et al. 2004],[Shekhar et al. 2001], but none of these proposals work with objects moving in a continuous way in time. However, in the building of a warehouse for trajectories, it is a crucial issue. The movement of a spatio-temporal object *o* - i.e., it *trajectory* - 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*.

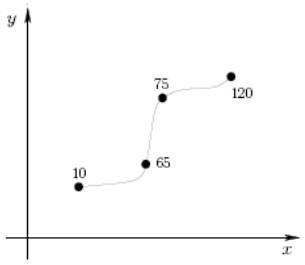

Figure 1. Trajectory with a sampling

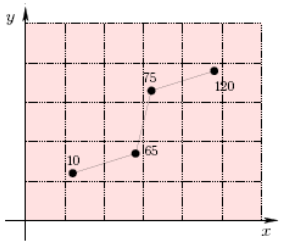

Figure 2. Linear interpolation

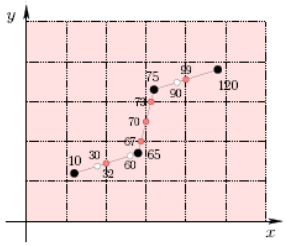

Figure 3. Interpolated trajectory with spatial and temporal points

The Figure 1 represents a trajectory of a moving object in a two dimensional space. Each point of the trajectory is represented by a tuple *(id, x, y, t)* corresponding to an object *id* in a location *(x, y)* at time *t*. There are some situations where we have to reconstruct the trajectory of the moving object from its sampling, e.g., when one is interested in computing the cumulative number of trajectories in a give area. In [Braz et al. 2007] the proposal is to use *linear local interpolation* in order to do it, assuming the movement of the objects between the observed points happens with constant speed, in a straight way. A Trajectory Data Warehouse (*TDW*) has to capable to store a stream of samplings, process the data volume, compute and store the measures in order to provide an environment to analyze the information about the objects. The aggregations measures are crucial in order

to do it. However, in a data stream environment, where the data arrive in an irregular and unpredictable way it is specially difficulty. In the *loading phase* the available resources are a very important constraint, it is necessary to develop some mechanism in order to limit the consumption of the resources and to improve the performance of the process. Besides, there is another important phase: *computing measures*, several measures can be computed in agreement with different complexity spaces. Therefore, the *TDW* must be modeled considering all these issues. In this article we are considering the the model proposed in [Braz et al. 2007]. In the next subsections we will present the solution proposed by [Braz et al. 2007] to solve the problem of loading and computing the data volume into the TDW.

2.1. Loading Problem

The loading begins at the base cells of the base cuboid, with suitable sub-aggregate measures, from which starting to compute super-aggregated functions. Considering the data stream characteristic, the proposal is to limit the amount of buffer memory needed to store information about active trajectories. In agreement with [Braz et al. 2007] the authors consider a trajectory ended when, for a long time interval, no further observation has been received. Given the observations for a trajectory shown in the Figure 1, a possible reconstructed trajectory using linear interpolation is shown by the Figure 2, where also is illustrated the discretized 2D space. With the linear interpolation is possible to infer additional spatio-temporal locations of intermediate points, these points occur between two known trajectory observations. Updating 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, the Table 1 shows the base cells.

Table 1. Cells representation - for each observation

Time label	X	Y	T	M_1	...	M_k
10	[30,60)	[30,60)	[0,30)	...	...	...
65	[60,90)	[30,60)	[60,90)	...	...	...
75	[90,120)	[90,120)	[60,90)	...	...	...
120	[120,150)	[90,120)	[60,120)	...	...	...

Before the interpolation some base cells could be traversed by the trajectories but, since no observation falls in them, they not appear in the *fact* table, the solution proposed in [Braz et al. 2007] is to consider the additional interpolated points for each cell traversed by a trajectory. The interpolation is computed considering the intersections between the trajectory and the border of the spatio-temporal cells. The Figure 3 shows a trajectory considering the additional interpolated points. The 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 points labeled with 32, 67, 70, 73, and 99, have been instead introduced to match the spatial dimensions.

After the including of these additional interpolated points, there are additional 3D base cells is possible to 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. Now is possible to update a cell

Table 2. Sequence of segments composing the interpolated trajectory, and the base cells that completely include each segment.

$(\mathbf{t}_i, \mathbf{t}_{i+1})$	X	Y	T	M_1	...	M_k
(10,30)	[30,60)	[30,60)	[0,30)	...	...	...
(30,32)	[30,60)	[30,60)	[30,60)	...	...	...
(32,60)	[90,120)	[30,60)	[30,60)	...	...	...
(60,65)	[90,120)	[30,60)	[60,90)	...	...	...
(65,67)	[90,120)	[30,60)	[60,90)	...	...	...
(67,70)	[90,120)	[90,120)	[60,90)	...	...	...
(70,73)	[120,150)	[90,120)	[60,90)	...	...	...
(73,75)	[120,150)	[120,150)	[60,90)	...	...	...
(75,90)	[120,150)	[120,150)	[60,90)	...	...	...
(90,99)	[120,150)	[120,150)	[90,120)	...	...	...
(99,120)	[150,180)	[120,150)	[90,120)	...	...	...

measure on the basis of a single trajectory segment. The Table 2 shows the sequence of edges composing the interpolated trajectory of Figure 3, and the base cell which the edge belongs to.

2.2. Aggregation Problem

A typical measure in a trajectory data warehouse can represent any interesting property about the trajectories in a spatio-temporal interval. In [Gray et al. 1997] the authors present a classification of aggregate functions based on the space complexity for computing a super-aggregate starting from a set of sub-aggregates previously computed:

- *Distributive*: The super-aggregates can be computed from the sub-aggregates.
- *Algebraic*: The super-aggregates can be computed from the sub-aggregates together with a finite set of auxiliary measures .
- *Holistic*: The super-aggregates cannot be computed from the sub-aggregates, not even using any finite number of auxiliary measures.

Table 3. Numeric measures

m1	*numobs*	*Number of observations in the cell*
m2	*trajinit*	*Number of trajectories starting in the cell*
m3	*presence*	*Number of trajectories in the cell*
m4	*distance*	*Total distance covered by trajectories in the cell*
m5	*speed*	*Average speed of trajectories in the cell*
m6	v_{max}	*Maximum speed of trajectories in the cell*

The Table 3 shows the measures that are considered in [Braz et al. 2007]. In agreement with the authors, the computation of the super-aggregates for the measures *m1, m2, m4* and *m6* uses *distributive* aggregate functions. After the loading of the base cells with the exact measures is possible to accumulate the measures by usage the function *sum* (*m1, m2* and *m4*) and *max* (*m6*). However, the super-aggregate for the measure *m5* is *algebraic*, it is necessary to compute an auxiliary measure in order to compute the aggregate function. A pair $\langle distance, time \rangle$ must be considered, where *distance* is the measure *m4* and *time* is the total time spent by trajectories in the cell. For a cell *C* arising as

the union of adjacent cells, the cumulative function performs a component-wise addition, producing a pair $\langle distance_f, time_f \rangle$, therefore the average speed in *C* is computed by $distance_f / time_f$. The aggregate function for *m3* is *holistic*, then is necessary to compute the measure in an approximated way. In this application we have used the approach presented in [Braz et al. 2007], the approach will be presented in the following.

The *Presence* function represents the count of the *distinct* trajectories crossing a given cell. Since this function has to deal with the issues related to counting *distinct* trajectories, it is a sort of `COUNT_DISTINCT()` aggregate, and thus a *holistic* one. The authors have used alternative and *non-holistic* aggregate functions to compute *Presence* values that *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 is possible to compute a super-aggregate. The two approximate functions which are considered are the $Presence_{Distributive}$ and $Presence_{Algebraic}$. A more detailed explanation about these functions can be found in [Braz et al. 2007]. Here we are considering that the $Presence_{Distributive}$ represents the *approximate* number of distinct trajectories crossing a given cell, however the usage of this function does not solve the problem of duplicate counting, where a same trajectory can be counted multiple times when happens a roll-up operation. The function named $Presence_{Algebraic}$ is used in order to compute the super-aggregation, each base cell stores an M-tuple of measures. One of these is the exact (or approximate) count of all the *distinct* trajectories touching the cell. The other measures are used to compute the super-aggregate, in order to correct the errors introduces by function $Presence_{Distributive}$ due to the duplicated count of trajectory presences.

Then, the idea is to compute a *holistic* measure by the usage of the *distributive* and *algebraic* measures. Therefore, the final result of the measure is an approximated value, computed in agreement with the limited resources presented in a data stream environment.

3. The Application

In this section we will summarize the application presented in [Braz and Orlando 2007]. The application allows to receive, compute the measures and store the data in a TDW. The TWD model was implemented in agreement with the model detailed in the Section 2. 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 the Section 2.1 and 2.2.

The Figure 4 shows the interface where is possible 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 have used the *MS SQL SERVER 2005*. The *TDW* was modeled in agreement with the *star* model [Kimball 1996], with a fact table and three dimension tables (*X* and *Y* spatial dimensions and *T* temporal dimension). The structure of these tables can be found in the Tables 4, 5 and 5. The application allows the user to access the *DTW* and to define a lot of settings:

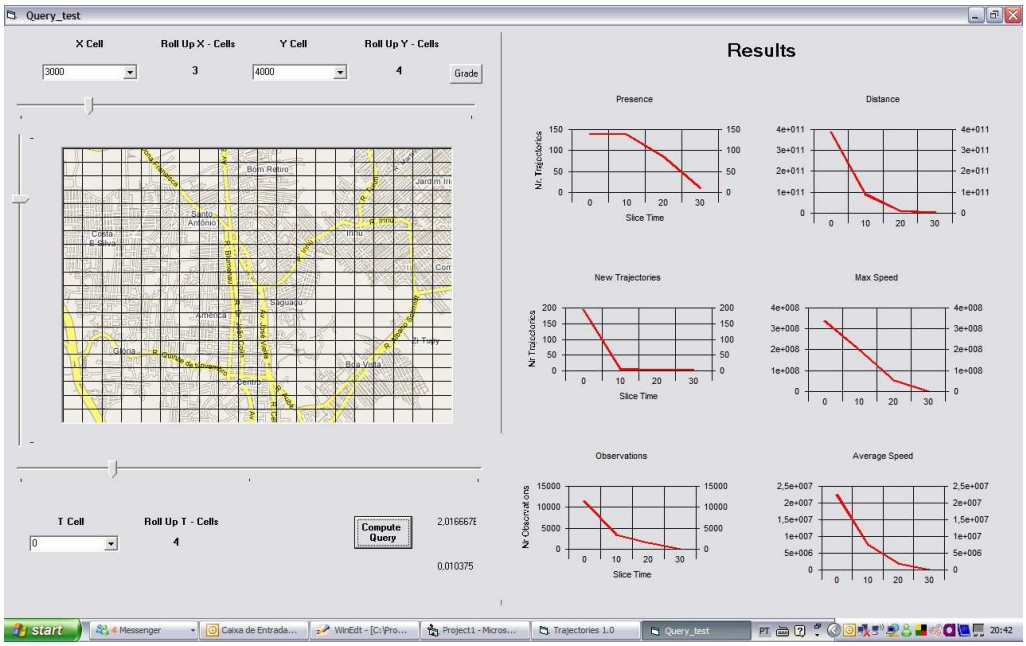

Figure 4. Results Interface

the definition of the level of the granularity of the trajectory data warehouse, to control the loading of the *TDW* and computation of the measures and aggregations are some of the possibilities of the usage.

Table 4. Fact Table

tid	*time foreign key*
xid	*X spatial foreign key*
yid	*Y spatial foreign key*
numobs	*Number of observations in the cell*
trajinit	*Number of trajectories starting in the cell*
vmax	*Maximum speed of trajectories in the cell*
distance	*Total distance covered by trajectories in the cell*
time	*Total time spend by the trajectories in the cell*
presence	*Number of trajectories in the cell - distributive*
xborder	*Number of trajectories crossing the x cell border*
yborder	*Number of trajectories crossing the y cell border*
tborder	*Number of trajectories crossing the t cell border*
speed	*Average speed of trajectories in the cell*

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 the Table 4 are detailed in the Section 2.2. The measures *presence, xborder, yborder* and *tborder* are necessary in order to compute the *holistic presence* measure.

The application allows to the user to define some settings of the TDW, for example to define the granularity of the grid, in this case we are considering the definition of the granularity for each cell of the TDW . These procedures are executed just one time,

Table 5. X or Y Dimension Table

xid	primary key
xl1	first level of hierarchy
xl2	second level of hierarchy

Table 6. T Dimension Table

tid	primary key
tl1	first level of hierarchy
tl2	second level of hierarchy

before the beginning of the loading the data warehouse. After the definition of the settings, the interface permits to start the process of the receiving the data stream values and loading the *TDW*. Using the setting values already defined, the loader component gets the active values in the *buffer table* and performs the procedures in order to load the *TDW*. A detailed explanation about the mechanism used to make the interpolation can be found in [Braz and Orlando 2007].

The pre-aggregated values are stored in the *TDW*. These pre-aggregated values are computed in the *loading* phase. However, when is necessary to compute some query or to perform a roll-up operation, the application uses the *aggregation component*. The Figure 4 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 is possible to verify the evolution of the measures for each value of time dimension.

The user can choose the query *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. The results of the queries are presented in the *Results* area in the query form of the application (see Figure 4).

4. Usage of Data Mining

The knowledge discovery in databases is the non-trivial process of identifying valid, novel, potentially useful and ultimately understandable patterns in data [Dunham 2003]. Data mining is a step in this process. The knowledge discovery is the prime goal of Data Mining technology.

In this discussion we are interested in the *Association Rules* mining task. Association rules are used to show the relationships between data items [Dunham 2003]. A database can be viewed as a set of tuples where each tuple contains a set of attributes. If, for instance, each tuple represents a sale transaction and each attribute represents an item purchased, an association rule can reveal the association between the purchased items.

Association rules work with two principal values: *support* and *confidence*. The *support* is the percentage of transactions in which each item occurs. Next, we give a formal definition taken from [Rakesh Agrawal and Swami 1993].

Definition 4.0.1 *Given a set of* items $I = I_1, I_2, ..., I_m$ *and a database of transactions* $D = t_1, t_2, ..., t_n$ *where* $t_i = I_{i1}, I_{i2}, ..., I_{ik}$ *and* $I_{ij} \in I$*, an* ***association rule*** *is an implication of the form* $X \Longrightarrow Y$ *where* $X, Y \subset I$ *are sets of items called* itemsets *and* $X \cap Y = \emptyset$.

Definition 4.0.2 *The* $support(s)$ *for an association rule* $X \Longrightarrow Y$ *is the percentage of transactions in the database that contain* $X \cup Y$.

Definition 4.0.3 *The* $confidence(\alpha)$ *for an association rule* $X \Longrightarrow Y$ *is the ratio of the number of transactions that contain* $X \cup Y$ *to the number of transactions that contain* X.

Many associations can be found by association rule task, but just some of them are interesting, ***support*** and ***confidence*** can be used to do this selection. The ***confidence*** measures the strength of the rule and ***support*** measures how often it occur in the database. Usually just some rules are interesting. In order to select which rules are interesting, a minimum threshold for support and confidence are defined.

The *association rules* task could be used in order to reveal the relationship among the measures stored for each cell. In this case we have two possibilities:

1. to consider the patterns resident in a given cell (or a set of cells) without using the time value;
2. to consider the patterns for a set of timestamps

In the first case (1), the goal is to reveal the stronger relationships, it is possible because the time dimension is not considered in the building of the rules, the set of transactions is constrained just by the values of the boundaries of the cells. However, it is an incomplete result because the trajectory data warehouse is totally dependent of the time dimension. The definition of the trajectory (see Section 2) presents the trajectory as the representation of the movement of a spatio-temporal object, this movement is related with the positions of the object in several timestamps. Therefore, we consider the item 2 the more representative knowledge for the environment, the time dimension permits to evaluate the evolution of the knowledge for the cell, or set of cells.

In the Section 3 the Figure 4 presents the evolution of the measures along the the time value. However, it represents just the evolution of the measures, it does not represent the evolution of the knowledge resident in the cell, the knowledge could be an *association rule* discovered among the measures of the cell. The knowledge is represented by a pattern which is discovered by the data mining tasks, the patterns can reveal the occurrence of some phenomenon related by a given cell. The evolution of a pattern along the several *timestamps* can be useful in order to understand the occurrence of the phenomenon of trajectories. For example, a pattern related to some measures (*average speed, acceleration, number of distinct trajectories...*) can reveal the occurrence of a *traffic jam* (*real phenomenon*), then it can be used to anticipate the occurrence of another related phenomenon and to execute actions to prevent the consequent occurrence.

Considering the TDW model proposed in [Braz et al. 2007], some transformations in the records of the fact table must be done. In the traditional *association rules* algorithms the goal is to identify the items which are related, the source are the records which represent the occurrence of the items for each transaction. In the fact table we have data about cells and its measures (*number of trajectories, distance traveled, average speed, acceleration...*). In this case the proposal is to transform each row of the fact table in a transaction in order to execute the *association rules* task.

Each record stored into the fact table has a triple (X_i, Y_i, T_j) identifying a zone (X_i, Y_i) during a given time interval T_j. The our proposal is to transform each record in order to be used by the *association rules* task, where the idea is to compute the relationship among the items of a data set. In order to do the transformation the first step is the discretization of the domain of the attributes of the records. In the our model the attributes are the measures of the fact table, therefore a measure can be the source of a lot of items. For example, the domain of attribute *presence* can be discretized as follows: [0,100),[100,200),...,[900,1000), in this manner each interval of discretization represents

a item in the transaction identified by the triple (X_i, Y_i, T_j). The discretization allows to consider each record of the fact table as a transaction in order to use some algorithm of *association rule*.

However, this method has some problems related to discretization: a lower range of discretization can produce a high number of items decreasing the performance of finding *patterns* or *association rules*, besides it could be the origin of very generic *patterns* involving a lot of records without to reveal a real knowledge. On the other hand, a high range of discretization can generate a high number of *patterns* or *association rules* which could be merged into a lower number without to lose the accuracy of the knowledge;

How to determine an adequate level of discretization seems to be a very important point of research, a possible solution could be the usage of a threshold to define an research area around each transformed measures (*items*) in order to limit the scope of the algorithm to find the *patterns*, therefore the idea is to use some constraints to limit the mechanisms of finding the knowledge.

In the current stage of our research we are implementing the data mining tasks. We have started by the implementation of the *association rules* task. We have implemented the data mining module using the *LCM* algorithm ([T. Uno and Arimura 2003]). Actually, we still are not considering the dimension time in the current implementation. The our first goal is to identify the best model in order to adapt the *TDW* data to use the *association rules* techniques. In agreement with the preliminary results we can confirm the problem of the number of *patterns* related with the width of the discretization: a high value of range discretization results in a high number *patterns*, however a low value corresponds to a low number of very generic *patterns* and a decreasing in the performance of the mining process. The next steps of the research are measuring the relationship among the level of discretization and the performance of the data mining task, besides to try to solve the problem of discretization and to implement the data mining tasks considering the time dimension.

5. Conclusions and Future Work

In this paper we have presented an review about the model proposed to implement a *TDW* and an application to define, implement and analyze the *TDW* environment. Besides, we are starting a discussion about the possibility of the usage of data mining techniques in order to reveal the knowledge resident in a *TDW*.

The application is a first step in order to try solving 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* model. However, this model can be extended to more general situations.

The current stage of the application can solve some problems of a trajectory data warehouse environment. However, the dimensions that we have used are very simple, a possible future work could be to sophisticate the hierarchy of the dimensions. The loading phase is an open problem, we have limited the loading phase considering a linear interpolation. However is possible to find some topological situations (e.g roads, bridges) where is very difficult to do the interpolation because of the some constraints in the movement of the object.

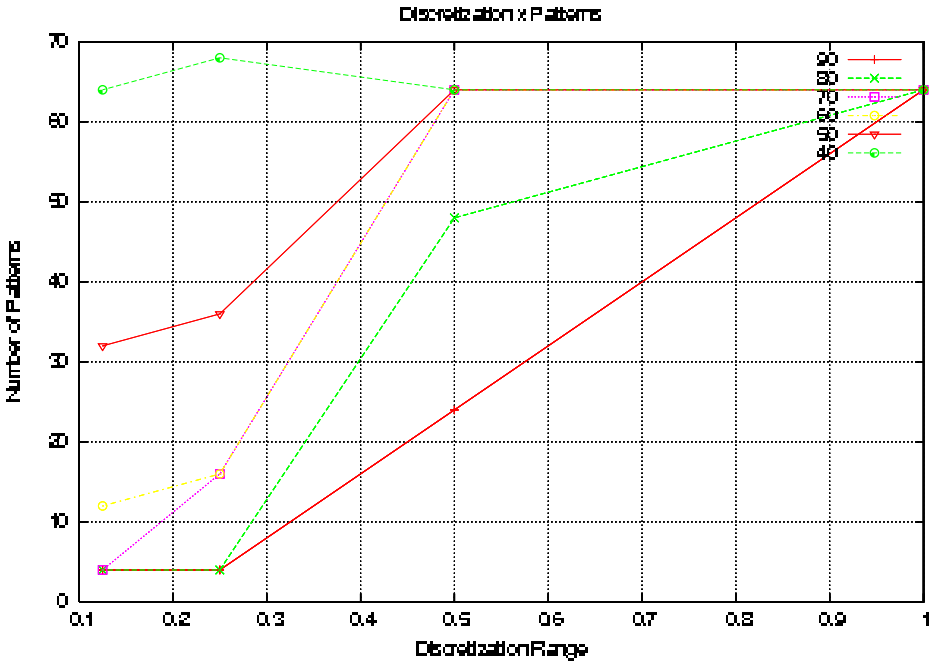

Figure 5. Discretization vs Patterns

Another interesting area of additional research is to develop more complex measures in order to provide values to discover patterns or trend of the trajectories. To compute values to support the data mining tasks is a very interesting point of future research. The traditional methods of data mining must be adapted in order to work in a *TDW* environment, however the possibility of the usage of these techniques seems to be very promisor.

References

Braz, F. and Orlando, S. (2007). Trajectory data warehouses: Proposal of design and application to exploit data. In *GEOINFO 2007: Proceedings of the 9th Brazilian Symposium on Geoinformatics*, Campos do Jordao,Sao Paulo,Brazil. SBC Brazilian Computer Society.

Braz, F., Orlando, S., Orsini, R., Raffaetà, A., Roncato, A., and Silvestri, C. (2007). Approximate aggregations in trajectory data warehouse. In *Proc. of ICDE Workshop STDM*, pages 536–545.

Dunham, M. H. (2003). *Data mining introductory and advanced topics*. Prentice Hall.

Gray, J., Chaudhuri, S., Bosworth, A., Layman, A., Reichart, D., Venkatrao, M., Pellow, F., and Pirahesh, H. (1997). Data cube: A relational aggregation operator generalizing group-by, cross-tab and sub-totals. *Data Mining and Knowledge Discovery*, 1:29–54.

Han, J., Stefanovic, N., and Kopersky, K. (1998). Selective materialization: An efficient method for spatial data cube construction. *PAKDD'98*, pages 144–158.

Kimball, R. (1996). *Data Warehouse Toolkit*. John Wiley.

Marchant, P., Briseboi, A., Bedard, Y., and Edwards, G. (2004). Implementation and evaluation of a hypercube-based method for spatiotemporal exploration and analysis. *ISPRS Journal of Photogrammetry and Remote Sensing*, 59:6–20.

Rakesh Agrawal, T. I. and Swami, A. N. (1993). Mining association rules between sets of items in large databases. *Proceedings of the ACM International Conference on Management of Data*, pages 207–216.

Rivest, S., Bedard, Y., and Marchand., P. (2001). Towards better support for spatial decision making: Defining the characteristics of spatial on-line analytical processing(solap). *Geomatica*, 55(4):539–555.

Shekhar, S., Lu, C., Tan, X., Chawla, S., and Vatsavai, R. (2001). *Map Cube: Avisualization Tool for Spatial Data Warehouse*, chapter Geographic Data Mining and Knowledge Discovery. Taylor and Francis.

T. Uno, Y. Uchida, T. A. and Arimura, H. (2003). Lcm: An efficient algorithm for enumerating frequent closed item sets. In *FIMI'03: Workshop on Frequent Itemset Mining Implementations*, Florida, USA. IEEE Computer Society.