Efficient Trajectory Pattern Mining for both Sparse and Dense Dataset

Ajaya Kumar Akasapu
Research Scholar, K L University
Department of CSE
Rungta College of Engineering and Technology, Bhilai (CG)

Lokesh Kumar Sharma
Department of IT and MCA
Rungta College of Engineering and Technology, Bhilai (CG)

G. Ramakrishna
K L University,
Vijayawada (AP)

ABSTRACT
The comprehension of phenomena related to movement – not only of people and vehicles but also of animals and other moving objects – has always been a key issue in many areas of scientific investigation or social analysis. Many applications track the movement of mobile objects, using location- acquisition technologies such as Global Positioning System (GPS), Global System for Mobile Communications (GSM) etc., and it can be represented as sequences of time stamped locations. In this paper, we analyze the trajectories of moving vehicles and we develop an algorithm for mining the frequent patterns of Trajectory data. We use the extensions of sequential pattern mining to spatiotemporal annotated sequential patterns. The description of frequent behaviors in terms of both space (i.e., the regions of space visited during movements) and time (ie, the duration of movements). In this paper an efficient trajectory pattern mining is proposed by incorporating three key techniques. In this paper we have examined ways of partitioning data for trajectory pattern discovery. Our aim has been to identify methods that will enable efficient counting of frequent sets in cases where the data is much too large to be contained in primary memory, and also where the density of the data means that the number of candidates to be considered becomes very large. Our starting point was a method which makes use of an initial preprocessing of the data into a tree structure (the P-tree) which incorporates a partial counting of support totals

General Terms
Trajectory Data Mining, Mobility Data Mining.

Keywords
Trajectory Pattern Mining, Trajectory Data.

1. INTRODUCTION
Our everyday actions, the way people live and move, leave digital traces in the information systems of the organizations that provide services through the wireless networks for mobile communication. The location technologies, such as Global System for Mobile Communications (GSM) and UMTS, currently used by wireless phone operators are capable of providing an increasingly better estimate of a user's location, while the integration of various positioning technologies proceeds: Global Positioning System (GPS)-equipped mobile devices can transmit their trajectories to some service provider, Wi-Fi and Bluetooth devices may be a source of data for indoor positioning, Wi-Max can become an alternative for outdoor positioning, and so on. These devices produce a huge amount of trajectory data which is described as geometry changes over time continuously. Since a large amount of trajectories can be accumulated for a short period of time, many applications need to summarize the data or extract valuable knowledge from it. As a part of the trend, discovery of trajectory patterns has been paid great attention due to many applications. For the pattern discovery of spatiotemporal data, many techniques in the literature have partitioned data space into disjoint cells (e.g., fixed grid) [7]. The reasons are mainly two-folded. First, space-decomposition techniques bring efficiency of discovery process. Obviously, dealing with symbols identifying each cell is much simpler than handling real coordinates which should have bigger data size and give lower intuitions for data processing. Second, spatiotemporal data has a distinct characteristic from general data for mining studies (e.g., basket data). Assume 'Arnav' arrives at his work at 9 a.m. every weekday. Therefore, there is an opportunity and a challenge to discover automatically, from these trajectories, spatiotemporal patterns that convey useful knowledge. Trajectory pattern discovery has many important, real-world applications driven by the real need such as Homeland security (e.g., border monitoring), Law enforcement (e.g., video surveillance), Location-based service etc.

The remainder of the paper is organized as follows: In Section 2, the trajectory pattern mining problem is defined. In Section 3, related work on sequential pattern mining and trajectory pattern mining are reported. In Section 4, we presented proposed trajectory pattern by pattern growth. Our experimental results and performance analysis are reported in Section 5 and our study is concluded in Section 6.

2. PROBLEM DEFINITION
The basic object of this work is the trajectory that describes the movement of an object. Trajectory data are normally obtained from location-aware devices that capture the position of an object at a specific time interval. The collection of these kinds of data is becoming more common, and as a result large amounts of trajectory data are available in the format of sample points. In many application domains, such as transportation management, animal migration, and tourism, useful knowledge about moving behavior or moving patterns can only be extracted from trajectories, if the background geographic information where trajectories are located is considered. Therefore, there is a necessity for a special processing on trajectory data before applying data mining techniques. The location, like a GSM cell or latitude - longitude pair, is abstracted using ordinary Cartesian coordinates, as formally stated by the following (Definitions are adopted from [5][14]):

Definition 1. (ST-sequence) A spatio-temporal sequence (ST-sequence) or trajectory is a sequence of triples $S = \{(x_0, y_0, t_0) \ldots (x_k, y_k, t_k)\}$, where t_i ($i = 0..k$) is a timestamp, $\forall 0 \leq i < k, t_i < t_i + 1$ and (x_i, y_i) are points in $\Re^2$.

The key task in sequence mining consists in counting the occurrences of a pattern, i.e., those segments of the input data that

match a candidate pattern. Matching the elements of a sequence in standard sequential patterns requires simple equality tests between symbols; instead, in our context it requires matching spatial locations, on the base of some notion of approximated match and error tolerance. That can be formally expressed in a simple and general way by means of a neighborhood function N: $\Re^2 \rightarrow P(\Re^2)$, which assigns to each pair (x, y) a set N(x, y) of neighboring points.

Definition 2. (Spatial containment,≤ N) Given a sequence of spatial points S = {(x_0, y_0). . . (x_k, y_k,)},, a spatio-temporal sequence T = {(x_0, y_0, t_0), . . . , (x_n, y_n , t_n)} and a neighborhood function N: $\Re^2 \rightarrow P(\Re^2)$, we say that S is contained in T (S ≤ N T, or simply S ≤ T, when N is clear from context) if and only if there exists a sequence of integers $0 \leq i_0 < \cdots < i_k \leq n$ such that: $\forall 0 \leq i < k$. (x_j, y_j) ∈ N(x_{ij} , y_{ij}).

The inclusion of temporal information in a sequential pattern can be obtained by making the patterns include temporal constraints between consecutive elements of the sequence, following the same spirit of temporally annotated sequences (TAS) [3][6]:

Definition 3. (T-pattern) A Trajectory pattern, called T-pattern, is a pair (S, A), where S {(x_0, y_0), ..., (x_k, y_k)} is a sequence of points in $\Re^2$, and A = {α_1, …, α_k} ∈ $\Re_+^k$ is the (temporal) annotation of the sequence. T-patterns will also be represented as (S, A) = (x_0, y_0) $\overset{\alpha_1}{\rightarrow}$ (x_1, y_1) $\overset{\alpha_2}{\rightarrow} \ldots \overset{\alpha_k}{\rightarrow}$ (x_k, y_k).

Problem Statement: Given a trajectory database and the min_support threshold, trajectory pattern mining is to find the complete set of trajectory patterns in the database. Definition 4 is represented trajectory pattern mining.

Definition 4. (Trajectory pattern mining) Given a trajectory database TD, a time tolerance τ , a neighborhood function N() and a minimum support threshold s_{min} , the trajectory pattern mining problem consists of finding all frequent T-patterns, i.e., all T-patterns (S,A) such that support D, τ,N(S,A) ≥ s_{min} ; where the support D, τ,N of a T-pattern (S,A) is the number of input trajectories T ∈ TD such that (S,A) ≤ N, τ T.

3. RELATED WORK

Agrawal and Srikant[1] first introduced sequential pattern mining problem. Given a set of sequences, where each sequence consists of a list of elements and each element consists of a set of items, and given a user-specified min-support threshold, sequential pattern mining is to find all frequent subsequences. Agrawal and Srikant [1] generalized their definition of sequential patterns in [5] to include time constraints, sliding time window, and user-defined taxonomy, and presented an apriori-based, improved algorithm GSP (i.e., generalized sequential patterns). Mannila et al. [12] presented a problem of mining frequent episodes in a sequence of events, where episodes are essentially acyclic graphs of events whose edges specify the temporal precedent-subsequent relationship without restriction on interval. Lu et al. [11] proposed inter-transaction association rules that are implication rules whose two sides are totally-ordered episodes with timing-interval restrictions. Han et al. [8] developed a frequent pattern mining method for mining partial periodicity patterns that are frequent maximal patterns where each pattern appears in a fixed period with a fixed set of offsets and with sufficient support.
Almost all of the above proposed methods for mining sequential patterns and other time-related frequent patterns are a priori-like, i.e., based on the Apriori principle, which states the fact that any super-pattern of an infrequent pattern cannot be frequent, and based on a candidate generation and test paradigm proposed in association mining [7]. A typical Apriori like sequential pattern mining method, such as trajectory pattern mining [5], adopts a multiple-pass, candidate generation-and-test approach. Fosca Giannotti [5] proposed spatiotemporal pattern, which formalizes the aggregate movement behaviour. The new pattern, that is called trajectory pattern, represents a set of individual trajectories that share the property of visiting the same sequence of places with similar travel times. Therefore, two notions are central: (i) the regions of interest in the given space, and (ii) the typical travel time of moving objects from region to region. In fact, in their approach a trajectory pattern is a sequence of spatial regions that, on the basis of the source trajectory data, emerge as frequently visited in the order specified by the sequence; in addition, the transition between two consecutive regions in such a sequence is annotated with a typical travel time that, again, emerges from the input trajectories. Also they incorporated pattern-growth methods for mining trajectory patterns efficiently. The general idea is that trajectory databases are recursively projected into a set of smaller projected databases and trajectory patterns are grown in each projected databases by exploring only local frequent fragments.
Above reported works are mainly based on Apriori or FP-Growth framework. The Apriori like algorithms suffer problem such as a huge set of candidate sequences could be generated in a large sequence database; Multiple scans of databases in mining; generates a combinatorial explosive number of candidates when mining long sequential patterns. FP-Growth algorithms are good when data set is dense. But in case of sparse data set FP-Growth utilizes more space and it take same computation time as Apriori algorithm [4].
In this paper we use Apriori-TFP structure an algorithm, which completes the summation of the final support counts, storing the results in a second set-enumeration tree (the T-tree, of Total support counts), ordered in the opposite way to the P-tree. The T-tree finally contains all frequent sets with their complete support-counts. The algorithm used, essentially a form of Apriori that makes use of the partial counting that has already been done, is described in [13].

4. PROPOSED METHOD

Step I. Preprocess the GPS data and identify the trajectory. It is accomplished by the execution of a spatial query. All the task relevant objects are collected into one database.

Step II. Arrange the trajectory points ascending order on basis of time and identify stopping point with respect to input threshold.

Step III. In this step, we use the concept of partial support counting using the "P-tree" (Partial support tree). The idea is to copy the input data (in one pass) into a data structure, which maintains all the relevant aspects of the input, and then mine this structure. A P-tree is a set enumeration tree structure in which to store partial counts for item sets. The top, single attribute, level comprises an array of references to structures of the form shown to the right, one for each column. Each of these top-level structures is then the root of a sub-tree of the overall P-tree.

The advantages offered by the P-tree table are [4]:

1. Reduced storage requirements (particularly where the data set contained duplicate rows).

2. Faster run times because the desired total support counts had already been partially calculated.

Step IV. In this step the P-tree is examined and creates T-tree [4]. The T-tree is generated in an Apriori manner. There are a number of features of the P-tree Table that enhance the efficiency of this process:

1. The first pass of the P-tree will be to calculate supports for singletons and thus the entire P-tree must be traversed. However, on the second pass when calculating the support for "doubles" we can ignore the top level in the P tree, i.e. we can start processing from index 2. Further, at the end of the previous pass we can delete the top level (cardinality = 1) part of the table. Consequently as the T-tree grows in size the P-tree table shrinks.

2. To prevent double counting, on the first pass of the P-tree, we update only those elements in the top-level array of the T-tree that correspond to the column numbers in node codes (not parent codes). On the second pass, for each P-tree table record found, we consider only those branches in the T-tree that emanate from a top level element corresponding to a column number represented by the node code (not the parent code). Once the appropriate branch has been located we proceed down to level 2 and update those elements that correspond to the column numbers in the union of the parent and node codes. We then repeat this process for all subsequent levels until there are no more levels in the T-tree to consider.

5. EXPERIMENT AND RESULT

We used a dataset of moving vehicles in London City in our experiment. Data consists of positions of the vehicles which has been GPS-tracked are stored in a relational database. The data have been recorded only while the vehicles moved. Each record includes the vehicle-id, date and time, the latitude, longitude, and altitude of the position. To facilitate analysis of movement data, initial pre-processing in the database is performed, which enriches the data with additional fields: the time of the next position in the sequence, the time interval and the distance in space to the next position, speed, direction, acceleration (change of the speed), and turn (change of the direction) [2]. The enriched pre-processed data set is used to derive starting point and ending point of vehicle and Trajectories are separated. Figure 1 shows the performance of algorithm.

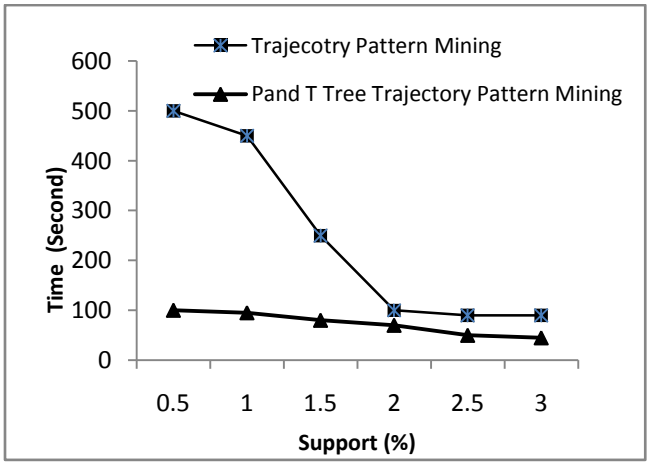

Figure 1. Graph showing performance of two algorithms.

6. CONCLUSIONS

The world becomes more and more mobile therefore mobility data attracts data mining community to find known and interesting pattern form mobility database. In this paper, we have performed a systematic study on mining of trajectory patterns in large trajectory databases and developed a tree approach for efficient and scalable mining of trajectory patterns. Instead of refinement of the apriori-like, candidate generation-and-test approach a P-tree structure is proposed. The experimental results we have reported here show that the Tree Partitioning method described is extremely effective in limiting the maximal memory requirements of the algorithm, while its execution time scales only slowly and linearly with increasing data dimensions. Its overall performance, both in execution time and especially in memory requirements, is significantly better than that obtained from either simple data segmentation or from other methods considered. The advantage increases with increasing density of data and with reduced thresholds of support – i.e. for the cases that are in general most challenging for association rule mining. Furthermore, a relatively high proportion of the time required by the method is taken up in the preprocessing stage during which the P-trees are constructed. Because this stage is independent of the later stages, in many applications it could be accepted as a one-off data preparation cost. In this case, the gain over other methods becomes even more marked. Note also that the P-tree construction and the partitioning thereof, are essentially generic: it leads to no loss of relevant information, and so could be used as the first stage of other quite different algorithms for completing the support-counts. A further advantage, not examined here, is that the independent processing of sub-trees can be carried out in parallel.

7. REFERENCES

[1] Agrawal R. and Srikant, R. 1995. Mining Sequential Patterns. Proc. 1995 Int'l Conf. Data Eng. (ICDE '95), 3-14.

[2] Andrienko , G. Andrienko , N and Wrobel W. 2007. Visual Analytics Tools for Analysis of Movement Data. ACM SIGKDD. 38-46.

[3] Andrienko, G. Malerba, D. May, M and Teisseire M. 2006. Mining spatio-temporal data. Journal of Intelligent Information Systems 27(3). 187–190.

[4] Coenen, F. Leng, P. and Ahmed, S. 2004. Data Structure for Association Rule Mining: T-Trees and P-Trees. IEEE Transactions on Knowledge And Data Engineering, Vol. 16(6). 774-778.

[5] Giannotti F., M. Nanni, Pedreschi, D. and Pinelli, F. 2007. Trajectory Pattern Mining, KDD'07, August 12-15, San Jose, California, USA. 330-339

[6] Giannotti, F. Nanni, M. Pinelli, F. and Pedreschi, D. 2006. T-Patterns: Temporally Annotated Sequential Patterns over Trajectories. Technical report, ISTI-CNR.

[7] Giannotti, F. and Pedreschi, D. 2008. Mobility, Data Mining and Privacy: Geographic Knowledge Discovery. Springer Verlag.

[8] Han J. 2001. Prefixspan: Mining sequential patterns by prefix-projected growth. In ICDE, 215–225.

[9] Kuijpers, G. and Othman W. 2007. Trajectory Databases: Data Models, Uncertainty and Complete Query Languages. ICDT Springer Verlag, LNCS 4353: 224-238

[10] Lee, J. T. Chen, Y. A. and Ip, W. C. 2009. Mining Frequent Trajectory Patterns in Spatial-Temporal Databases. Information Sciences: an International Journal. Vol. 179(13). 2218-2231.

[11] Lu, H. Han, J. and Feng, L. 1998. Stock Movement and n-Dimensional Intertransaction Association Rules. Proc. SIGMOD Workshop Research Issues on Data Mining and Knowledge Discovery. pp. 1-7.

[12] Mannila, H. and Toivonen, H., 1996. Discovering generalized episodes using minimal occurrences. In: Proc. of the 2nd Int. Conf. on Knowledge Discovery and Data Mining .146-151.

[13] Pei J. and Han J. 2004. Mining Sequential Patterns by Pattern-Growth: The PrefixSpan Approach. IEEE Transactions on Knowledge and Data Engineering, vol. 16(10).1424-1440.

[14] Sharma, L. K., Vyas O. P., Scheider S. and Akasapu A. 2010. Nearest Neibhour Classification for Trajectory Data. ITC 2010, Springer LNCS CCIS 101, pp. 180–185.