

Clustering Spatio-Temporal Patterns using Levelwise Search

Abhishek Sharma, Raj Bhatnagar
University of Cincinnati
Cincinnati, OH, 45221
sharmaak,rbhatnag@ececs.uc.edu

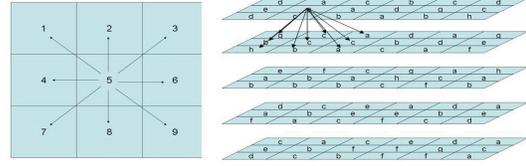


Figure 1: Spatial Grids at Successive Time Points

Abstract

Spatio-Temporal patterns are those which change in space with respect to time. Mining of such patterns enable us to understand the behavior of a phenomenon and predict it's responses under a given set of conditions. Here we propose a general algorithm for mining of such patterns which can be applied to many applications. The algorithm we propose is intuitive in nature. We will compare our algorithm with a few other methods and show that it can extract a wide range of patterns which other methods cannot.

1 Motivation

Discovery of spatio-temporal patterns in databases is of significant importance for many application domains. Spatial and temporal constraints introduce a very high level of structure in the datasets causing most of the standard data mining algorithms unsuitable for discovering patterns from such data. Much work has been done for discovering spatial patterns or temporal patterns in data but very little has been explored in the context of spatio-temporal datasets.

The main problem in any data mining task is to control the explosion of possible hypotheses, especially in the early phases of the search for the interesting hypotheses. The motivation for our algorithm presented in this paper is to conquer this explosion of hypotheses in a systematic manner such that computational cost is minimized and none of the interesting hypotheses are lost in the process.

2 Data Context

Our algorithm is developed in the context of grid-based datasets from domains having spatio-temporal characteristics. Each grid is assumed to represent the

spatial image of observations taken at some point in time.

Figure 1, in the right half, shows a number of spatial grids stacked over each other, in order of time instants at which each grid is observed. For an example, let us consider each grid to be covering a city and each cell of the grid representing a number of blocks in the city. The data stored in each cell represents the number of crime incidents that occurred in a month in that cell of the city. Successive grids represent the spatial crime distribution for successive months.

Temporal Patterns: A string of cell-values taken from, say, cell (1,1), would represent the temporal characteristic of the crime incidents in cell (1,1) of the city. If our grid is of size 10X10 then there are 100 temporal profiles, one for each cell in the grid. With these 100 temporal profiles we can use an algorithm for mining temporal patterns to discover frequently occurring temporal substring or subsequence patterns in the dataset.

Spatio-Temporal patterns: Spatio-temporal patterns are characterized by substrings or subsequences that occur close together, and or frequently, across the three dimensions of space-time. We characterize this by defining a more general set of strings in the context of the grid example shown in Figure 1 above. Starting from a cell (x,y) we connect to any one of the nine adjacent cells in the next grid. So, there are nine two-number strings and 81 three-number strings starting from any single cell of a grid. The number of such possible strings explodes very quickly but it is this set of strings that contains the spatio-temporal patterns. It is not possible to make explicit this humongous set of strings and it must remain only implicitly specified for any mining algorithm that must be scalable and efficient. An example of a pattern discovered by such algorithms is of a crime spree that moves from one neighborhood to others as weather changes from summer to winter and then back to the older neighborhoods in the next summer. The methodol-

ogy is applicable to situations where the grids may represent various types of pollution, weather, or social data.

3 Related Work

Much work has been done on mining and clustering of mobile objects in the spatio-temporal context. These objects make trajectories in time in a 2-dimensional plane. The goal of such research is to cluster those objects or trajectories which follow the same or similar path according some metric of similarity. Vlachos et. al. in [2], use the LCSS (Longest Common Subsequence) measure of similarity for clustering similar trajectories. They use projections of subsequences of two trajectories to find the similarity between them. Buzan et. al. in [3] also use the LCSS metric to cluster motion trajectories in video. Their method can cluster similar trajectories of varying lengths by reducing the length of the longer trajectories and matching only those parts which are similar. Mamoulis et. al. in [4], use the *Apriori* algorithm [1] for clustering trajectories obtained from historical data. They consider corresponding points of various trajectories and cluster them depending on whether they all belong to a small spatial neighborhood or not.

In all the above cases the database comes in the form of known or observed trajectories. Not much work has been done in the area of finding patterns in a dataset of natural phenomena like weather etc. which has not been reduced to a subset of interesting trajectories but consists of an implicit set of a very large number of possible trajectories. Some work has been done along these lines but that focuses on some very specific situational cases [5, 6]. The algorithm we present works on a whole area rather than a few specific points in the area. No data about the interesting trajectories is required.

4 Our Methodology

We assume that each cell (x,y,t) stores a value given as $v(x,y,t) = \{1, 2, \dots, k\}$ where $1 \leq x \leq X$, $1 \leq y \leq Y$ and $1 \leq t \leq T$. The number k is the maximum number of possible integer values (possibly quantized real values) from which a cell may contain one of the values.

We represent the i^{th} n-sequence across the grids in Figure 1 as $S_t(i)$ which starts at time instant *grid t* and continues for some time in the following two forms:

1. $\langle x_{it}, y_{it}, t_{it}, q_{it} \rangle, \langle d_{it+1}, q_{it+1} \rangle, \dots, \langle d_{it+n-1}, q_{it+n-1} \rangle$
2. $\langle x_{it}, y_{it}, t_{it}, q_{it} \rangle, \langle x_{it+1}, y_{it+1}, t_{it+1}, q_{it+1} \rangle, \dots, \langle x_{it+n-1}, y_{it+n-1}, t_{it+n-1}, q_{it+n-1} \rangle$

where d_i is the direction in which the sequence moves from time point $i-1$ to time point i and $q_i \in k$. The first form starts by giving an (x,y) location, a time instant, and a value in that cell, followed by a direction of cell in the next level and a value in that cell and so on. In the second form the profile is a sequence of points containing complete (x,y), t, and q value for each point of the profile.

An *interesting cluster C* is a set of sequences $\langle S_1, S_2, \dots, S_m \rangle$ such that they all share a *sub-sequence* $\langle x_i, y_i, t_i, q_i \rangle, \langle d_{i+1}, q_{i+1} \rangle, \dots, \langle d_{i+j-1}, q_{i+j-1} \rangle$.

We start our process by considering only three consecutive grids at a time, for all possible such sets of grids. For each set of three grids (three time points) we consider all sequences $S_i = \langle x_1, y_1, t_1, q_1 \rangle, \langle d_2, q_2 \rangle, \langle d_3, q_3 \rangle$ and identify all those clusters of sequences C_i which meet the clustering criterion. For example we may say that a C_i is a cluster if:

1. all sequences in it are 3-sequences
2. all of them have same values for $t_1, q_1, \langle d_2, q_2 \rangle, \dots, \langle d_3, q_3 \rangle$
3. each cluster has at least some minimum number (p) of sequences

The algorithm for finding such clusters is as follows:
procedure *subsequencesets*(t)

- 1: $CS_t^3 \leftarrow \emptyset$
- 2: *generatesubsequence*(t)
- 3: **for** $i = 1$ to *sizeof*(S_t) **do**
- 4: *check*(i) $\leftarrow 0$
- 5: **end for**
- 6: **for** $i = 1$ to *sizeof*(S_t) **do**
- 7: $tempCS_t^3 \leftarrow \emptyset$
- 8: **if** *check*(i) = 1 **then**
- 9: continue
- 10: **end if**
- 11: *check*(i) $\leftarrow 1$
- 12: *count* $\leftarrow 0$
- 13: **for** $j = 1$ to *sizeof*(S_t) **do**
- 14: **if** *check*(j) = 1 **then**
- 15: continue
- 16: **end if**
- 17: *check*(i) $\leftarrow 1$
- 18: **if** $q_{i1:i3} = q_{j1:j3}$ and $d_{i2:i3} = d_{j2:j3}$ **then**
- 19: *count* \leftarrow *count* + 1
- 20: $tempCS_t^3 \leftarrow tempCS_t^3 \cup S_t(j)$
- 21: **end if**
- 22: **if** *count* $\geq p$ **then**
- 23: $CS_t^3 \leftarrow CS_t^3 \cup tempCS_t^3$
- 24: **end if**
- 25: **end for**
- 26: **end for**

Figure 3 shows two clusters of 3-sequences in 3 grids and if $p=5$ then only set *awill* be retained in CS_1^3 and set *b* will be dropped.

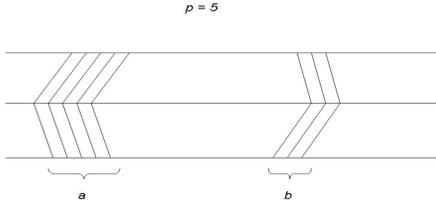


Figure 2: Figure 1

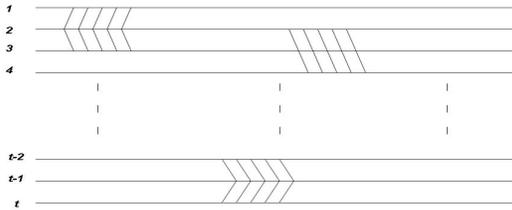


Figure 3: Clusters of 3-sequences in 3 grids

Let CS_1^3 be the set of clusters $\langle C_1, C_2, \dots, C_n \rangle$ found from the 3 grids beginning at time instant 1. In each of the i^{th} iteration we consider grids $i, i+1, i+2$ and compute CS_i^3 . That is, we compute $CS_1^3, CS_2^3, \dots, CS_{t-2}^3$. We use form 1 of the string description for the above algorithm because these are short and contain all the information.

Now, using CS_1^3 and CS_2^3 we can compute CS_1^4 using form 2 of the string descriptions as follows:

1. combine all those sequences from CS_1^3 and CS_2^3 which overlap over the grids 2 and 3 and include them in CS_1^4 . That is, where $x_{i2:i3}=x_{j2:j3}$ and $y_{i2:i3}=y_{j2:j3}$ for $i \in CS_1^3$ and $j \in CS_2^3$.
2. each CS_1^4 has at least some minimum number (p) of length 4 sequences

Algorithm for obtain CS_1^4 is as follows (substituting $t=4$):

procedure *combinesequencesets(t)*

- 1: $CS_1^t \leftarrow \emptyset$
- 2: **for** $i = 1$ to $\text{sizeof}(CS_1^{t-1})$ **do**
- 3: $\text{temp}1CS_1^t \leftarrow \emptyset$
- 4: **for** $j = 1$ to $\text{sizeof}(CS_{t-2}^3)$ **do**
- 5: **if** $x_{(it-2 : it-1)} = x_{(jt-2 : jt-1)}$ and $y_{(it-2 : it-1)} = y_{(jt-2 : jt-1)}$ **then**

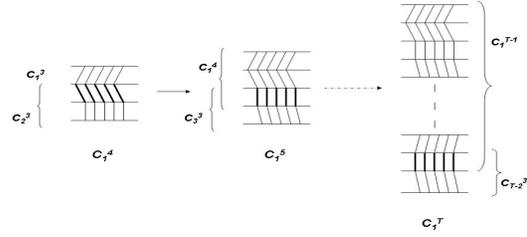


Figure 4: Combination of length-3 clusters to get length-4 clusters

- 6: $\text{temp}1CS_1^t \leftarrow \text{temp}1CS_1^t \cup (CS_1^{t-1}(i) \cup CS_{t-2}^3(j))$
- 7: **end if**
- 8: **end for**
- 9: **for** $j = 1$ to $\text{sizeof}(\text{temp}1CS_1^t)$ **do**
- 10: $\text{check}(j) \leftarrow 0$
- 11: **end for**
- 12: **for** $j = 1$ to $\text{sizeof}(\text{temp}1CS_1^t)$ **do**
- 13: $\text{temp}2CS_1^t \leftarrow \emptyset$
- 14: **if** $\text{check}(j) = 1$ **then**
- 15: continue
- 16: **end if**
- 17: $\text{check}(j) \leftarrow 1$
- 18: $\text{count} \leftarrow 0$
- 19: **for** $k = 1$ to $\text{sizeof}(\text{temp}1CS_1^t)$ **do**
- 20: **if** $\text{check}(k) = 1$ **then**
- 21: continue
- 22: **end if**
- 23: **if** $\text{temp}1CS_1^t(j) = \text{temp}1CS_1^t(k)$ **then**
- 24: $\text{temp}2CS_1^t \leftarrow \text{temp}2CS_1^t \cup \text{temp}1CS_1^t(k)$
- 25: $\text{count} \leftarrow \text{count} + 1$
- 26: $\text{check}(k) \leftarrow 1$
- 27: **end if**
- 28: **end for**
- 29: **if** $\text{count} \geq p$ **then**
- 30: $CS_1^t \leftarrow CS_1^t \cup \text{temp}2CS_1^t$
- 31: **end if**
- 32: **end for**
- 33: **end for**

The working of the above algorithm is illustrated in Figure 4.

Similarly, each CS_1^t , where $3 \leq t$, can be computed using CS_1^{t-1} and CS_{t-2}^3 as follows:

1. combine all those sequences from CS_1^{t-1} and CS_{t-2}^3 which overlap over the grids $t-2$ and $t-1$ and include them in CS_1^t . That is, where $x_{it-2:it-1}=x_{jt-2:jt-1}$ and $y_{it-2:it-1}=y_{jt-2:jt-1}$ for $i \in CS_1^{t-1}$ and $j \in CS_{t-2}^3$.
2. each CS_1^t has at least some minimum number (p)

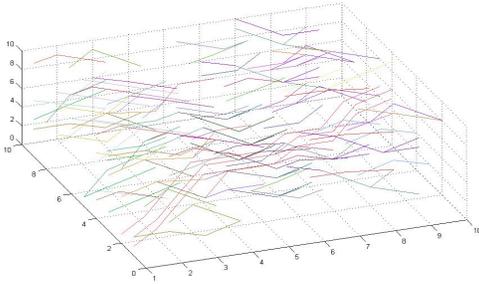


Figure 5: All strings of length 3 or more in data

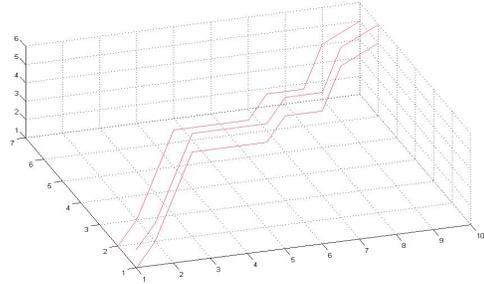


Figure 6: Cluster of length 10 strings

) of sequences

General Methodology: We can view all the clusters length 3 strings as belonging to one level of a lattice. The next level of the lattice consists of all those clusters of length 4 strings that can be created by combining some nodes from the lattice below. The lattice thus grows upwards to clusters of longer strings. The criteria for selecting the clusters at each level are upto the users and determine, to a great effect, the natures of clusters that get to populate the lattice. It is this levelwise search approach that gives the efficiency, scalability and power to our algorithm. When we are building the nodes at higher levels of the lattice, we can proceed by making explicit only those clusters that preserve some desirable monotonic or anti-monotonic property of the resulting clusters. This property can then be used to prune away the undesirable candidates from the levelwise search.

5 Results

The results shown here are obtained from a synthetic dataset. The data consists of 10 Grids, and each Grid consists of 10 X 10 cells. The first dataset was composed of random integers between 1 and 10. Then a few patterns covering all the 10 grids was introduced into the data. Figure 5 shows all the patterns found by the algorithm for length 3 or more. Figure 6 shows a cluster of length 10 strings where smaller clusters were combined based on their spatial proximity. The algorithm was able to extract the exact cluster that was introduced in the data.

The next example in Figure 7 depicts a special kind of pattern. There are two spatio-temporal patterns whose clusters converge at some point in time. These can also be identified by our algorithm.

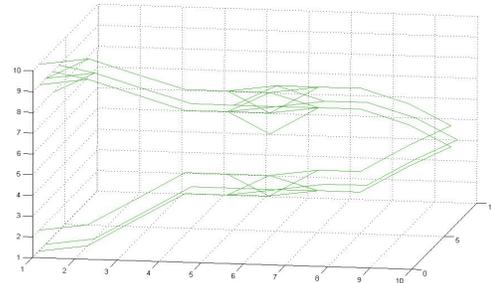


Figure 7: Two clusters that meet at a time point

6 Conclusion

We have demonstrated essential aspects of a methodology for discovering spatio-temporal patterns in raw data of a spatio-temporal dataset in which interesting trajectories have not been identified. Results demonstrate its efficacy and the approach can be extended to clusters and patterns with various types of properties and in various types of applications.

References

- [1] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In Proc. of Very Large Data Bases, pages 487-499, 1994.
- [2] M. Vlachos, G. Kollios, D. Gunopulos. Discovering Similar Multidimensional Trajectories, ICDE, 673-684, 2002.
- [3] Buzan D., Sclaroff S., Kollios G. Extraction and clustering of motion trajectories in video. In Proc. International Conference on Pattern Recognition, 2004.
- [4] N. Mamoulis, H. Cao, G. Kollios, M. Hadjieleftheriou, Y. Tao, and D. Cheung. Mining, indexing, and querying historical spatiotemporal data.

In Proc. of Intl. Conf. on Knowledge Discovery and Data Mining, pages . 236-245, 2004.

- [5] Stern, Spatio-temporal patterns of subjectively reported congestion in Tel Aviv metropolitan area, *Journal of Transport Geography*, March 2004, vol. 12, iss. 1, pp. 63-71(9) Elsevier Science
- [6] McGregor, Glenn R. Identification of air quality affinity areas in Birmingham, UK *Applied Geography*, 16 (1996) pp. 109-122