Significant Cascading Spatiotemporal Pattern Discovery

Group 4, Yiqun Xie and Yan Li

1. Motivation

Cascading spatiotemporal pattern (CSTP) is a reflection of the underlying mechanism of natural or social processes. Identifying CSTPs provides meaningful evidence for domain experts and government agencies to forecast potential consequences of an event and plan ahead. In a natural disaster such as a hurricane, it normally results in heavy rainfall and strong winds, which will further cause flooding and wind damage and lead to societal issues such as power outage. Early awareness of such consequences is critical to government agencies in order to make a timely reaction to reduce the damage. In criminology, bar closing may lead to drunk driving and assaults in the local areas (Figure 1) [11]. Discovering these types of patterns which have not been uncovered offers important and supportive knowledge in decision making.

In order to develop a more robust schema for CSTP mining, statistical significance needs to be incorporated into the mining process. Without significance test, patterns are mined mainly based on their frequency (or variations of frequency). This leaves space for chance patterns which happens by randomness and should not be preserved in the results. These chance patterns may provide misleading guidance or hints to domain scientists or decision makers and lead to inefficient or incorrect use of resources and poor plans. In this case, adding statistical significance test and filtering out CSTPs that happen by random chance is a critical step to boost the quality and confidence of the results.

![Figure 1 Example of cascading pattern in a spatiotemporal framework](image)

2. Problem definition
Given a set of features \( \{ f_1, f_2, f_3, \ldots f_n \} \) and set of point instances for each feature, the task is to find statistically significant cascading spatiotemporal patterns. A CSTP is represented as a connected graph, with each node representing a feature type and edge representing a cascading relationship between different feature types. A complete list of input, outputs, objective and constraints is shown as follows:

- **Input**
  - A set of instances of \( N \) features \( f_1, f_2, f_3, \ldots f_n \) in a spatiotemporal framework
  - An interaction distance threshold \( d \)
  - An interaction time-interval threshold \( t \)
  - An interest measure and a threshold \( r \)
  - A Null hypothesis \( H_0 \) for chance pattern
  - A significance level threshold \( p \)

- **Output**
  - Statistically significant cascading spatiotemporal patterns

- **Objective**
  - Computational efficiency

- **Constraint**
  - Correctness and completeness

In the list of inputs, besides the set of features and their point instances, two thresholds immediately follow up – one defined on the distance and other on the time interval between two instances. These two thresholds on spatial and temporal domain are required to determine if two instances can interact with each other. Together with the set of feature points, a directed acyclic spatiotemporal graph can be built by these three inputs, and two points become neighbor nodes in the graph if they are within the range of interaction. To evaluate a given pattern, an interest measure is needed to generate a score. Thus, a third threshold \( r \) is needed to remove patterns with low interest score.

Since Null hypothesis is sensitive to observed dataset, it is added to the input so it can vary in different application domains based on the underlying point process (e.g., complete spatial randomness). The last threshold \( p \) indicates the significance level imposed, which is normally chosen to be 0.01 or 0.05. In this work, the interest measure used is cascading participation index (CPI) [6], which is an spatiotemporal extension of participation index [11]. Cascading participation ratio (CPR) is needed for CPI computation. Denote CSTP as a sample cascading spatiotemporal pattern which has multiple participating event types, its CPR is computed as follows:
CPI is the minimum of CPRs of all its participating event types:

\[
CPI = \min\{CPR(CSTP, M)\}
\]

3. Related work

Frequent-set pattern mining was initiated by association rule mining in transaction processing [1-4]. A set of items is considered associated if they frequently appear on the same transaction. Moving forward to spatial domain, colocation pattern is defined to identify geo-located features that have a high frequency of neighbor-relationship [6]. Spatial colocation pattern takes a stationary view of a geographic phenomenon and works with a snapshot of a certain state of point distribution. For stationary spatial objects (e.g., branches of KFC, McDonalds), assumptions made by colocation are appropriate to find the patterns. However, for dynamic spatiotemporal processes such as chain consequences of a natural disaster, colocation fails to offer meaningful information due to the missing time domain. This leads to the development of cascading pattern detection [11,12]. Figure 2 shows a taxonomy tree of related work.

Most of the existing approaches in literature do not consider statistical significance [6-9, 11, 12] and use a fixed threshold to filter out patterns with low support. There has been efforts to incorporate significance test with spatial colocation pattern detection [5]. However, the test with temporal domain has not been explored. In addition, the work on significant colocation pattern [5] detection resulted in an approximation approach due to the expensive cost of computation. This shows the hardness of the problem even for pure spatial domain. Comparing to the relationship defined by colocation, cascading pattern has a richer context with the additional
domain of time. The added time domain differs from spatial domain since direction is implicitly embedded into the graph given the timestamps of events. Based on this property, complex combinations of cascading relationships can be generated and many of them are hard to examine. This leads to an increase in run-time comparing to spatial colocation pattern discovery.

In the colocation domain, many approaches have been proposed to accelerate the mining process [6-8]. For example, graph partition, apriori and grid-based sampling have been applied to colocation discovery [6]. These methods need to be reconsidered before being applied onto spatiotemporal graph [3,4].

In addition, cascading is also discovered in some research work in different fashions [9, 10]. In some work, the goal of cascading detection is to alarm abnormal cascading node [1]. For example, at a given timestamp, the number of cascaded events rises above a preset threshold [9]. Another type of cascading pattern discovery is performed on top of blog dataset [10]. Each post in the network can be referenced by another. In that case, the patterns identified do not include event type information but just the topological relationship between general nodes.

4. Challenges

Comparing to significance test on colocation pattern, the test on cascading pattern discovery is more concerned with the computational cost. Cascading by definition reflects a multi-level propagation process, which greatly increases the enumeration space of candidate patterns. To describe the relationship between colocation and cascading patterns, an analogy could be the relationship between combination and permutation. Combination directly reflects the nature of pattern enumeration space for colocation problem. For cascading problem, the enumeration space is in fact much greater than permutation since the candidate pattern can have a directed graph structure instead of a pure single line transitive sequence. For simplicity, a set of 3 features can yield at most 4 candidate colocation patterns. For cascading pattern, the number rises up to 24.

Despite the increased enumeration space, the temporal propagation process also leads to an enlarged search space. For colocation pattern, the size of search space is limited by the interaction distance threshold. In the case of cascading patterns, the distance threshold can hardly be used to narrow down the search space for patterns with more than two participating features. The reason is that the distance threshold only limits the interaction distance between two neighboring nodes in the graph. If two nodes in a pattern are not directly connected by one edge, their spatial distance could be more than the distance threshold. In such case, any filtering strategy based on the distance threshold cannot be applied (e.g., the grid based approach in [5]). Evaluation of interest measure on a given pattern is identified as the major computational bottleneck in [12].
Unfortunately, there is no known probabilistic model to compute the p-value. In this case, Monte Carlo simulation is used to test the significance of cascading pattern, which requires the mining algorithm to be repetitively run for a large number of trials (e.g., 1000). Due to the increased computational cost of cascading pattern discovery, stronger filters are necessary to reduce the cost of mining algorithm in order to perform the expensive significance test.

5. Contribution

Our contribution focuses on two parts: (1) a more robust problem definition; and (2) a more efficient mining algorithm.

For problem definition, a Null hypothesis is introduced to filter out chance patterns which may occur under a random point distribution. For the algorithm part we separated the design for the general algorithm and the method for Monte Carlo simulation (MCS). A general algorithm refers to a method usable on both observed dataset and random datasets generated in MCS trials. In contrast, the method for MCS is not applicable to observed dataset. The reason is in MCS, we are only interested in the relationship between simulated cascading participation index (CPI) and observed-CPI. Since it is a binary relationship, the results from observed dataset can be used to create more strict filters for CPI evaluation.

5.1 Significance test

A p-value test is used to check the significance of pattern results. Our Null hypothesis $H_0$ is defined by the following constraints:

- Instances of each feature type distribute independently from the instances of other features in the spatiotemporal framework;
- Instances of each feature type follow a homogeneous Poisson distribution in the spatiotemporal domain (random in three dimensions). In case of spatial-autocorrelation or inhibition, other point processes (e.g., Matern cluster process, Poisson hardcore process) will be chosen accordingly to better approximate the observed point distribution.

A pattern identified in the observed dataset is considered as a chance pattern if the probability of getting a higher CPI of the same pattern in a random point distribution is above a pre-specified significance level $p_0$ (e.g., 0.01, 0.05). Since there is no known distribution of CPI under any point processes, MCS is used to generate simulation trials and approximate the exact CPI distribution. Monte Carlo simulation will generate $M$ simulation trials (e.g., 1,000, 10,000) to estimate CPI distribution. Each trial is executed with the following steps:

- Generate a spatiotemporal point distribution under $H_0$
- Evaluate CPI for each pattern
- For each pattern, if its simulated CPI is higher than observed CPI, increase corresponding count $C_{\text{high}}$ of this pattern by 1.
After the last trial of MCS, a pattern is considered as a chance pattern if its $C_{\text{high}}$ is greater than $M \cdot p_0$.

5.2 General algorithm for CPI evaluation

The general algorithm is used for CPI evaluation in both observed dataset and simulated datasets. For this part, a participant expansion (PE) algorithm is proposed to speed up the evaluation process.

The PE algorithm is used within the framework of Apriori algorithm. Apriori algorithm requires anti-monotonicity property on interest measure. That is, given two CSTPs, namely CSTP$_1$ and CSTP$_2$, $\text{CPI}(\text{CSTP}_1) \geq \text{CPI}(\text{CSTP}_2)$ if CSTP$_1 \subseteq$ CSTP$_2$.

Apriori algorithm starts pattern enumeration with unit-size patterns and gradually expands out to large-size patterns by combining smaller size patterns. If a pattern cannot pass the CPI threshold, it will be pruned from the pattern tree since any pattern combined from it must also fail the CPI threshold according to the anti-monotonicity property.

In Apriori algorithm, a new pattern C is generated with two smaller size patterns A and B (Figure 3). The size of a pattern is determined by the number of edges included in the pattern. A size $k$ pattern is generated with two size $k-1$ patterns in the Apriori tree. To evaluate the CPI of a newly generated pattern, previous method on CPI evaluation [11] uses a spatiotemporal join to find which instances of A and B can be combined into the new pattern. The spatiotemporal join is performed with a nested loop approach to iterate through all possible pairs of A and B. Since majority of the instances cannot be joined, a large overhead is involved in the spatiotemporal join.

Instead of using a nested loop based join, the Participant Expansion (PE) algorithm uses a localized pattern matching approach to search for valid join-patterns so that the join overhead is minimized. To apply PE algorithm, pattern merge in the Apriori tree is divided into two types as shown in Figure 4:

Type-I merge: The set of nodes from two parent patterns are different.

Type-II merge: The set of nodes from two parent patterns are the same.
Lemma-1: Type I merge at least introduces a new node to one of the join-patterns and no new loop is created; Type II merge introduces a new undirected loop to each parent pattern.

For Type-I merge, PE algorithm can evaluate exact CPI for a given pattern. For Type-II merge, it serves as an upper bound to the actual CPI.

In PE algorithm, a point instance is defined as a participant of a pattern if it is involved in an instance of that pattern. Furthermore, for each point instance, a pattern ID (PID) generated in Apriori tree will be stored with it if the point instance is a participant of that pattern.

5.2.1 Type-I merge

For Type-I merge, an intersection node is defined as the node in a parent pattern that connects an old node and a new node. For example, in Figure 4(a), node A in the left parent pattern (A→B,C) connects old nodes B,C to the new node D from another pattern.

Lemma-2: For type I join pattern, if an intersecting node instance contains pattern IDs (PIDs) from both parent patterns, then it must be a participant of the newly merged pattern.

For each instance of intersecting node, a localized CPI evaluation will be initiated by checking the PID condition in Lemma-2. In the next phase, Lemma-2 can be generalized to a participant rule and a participant expansion lemma:

Participant rule: A node instance either (1) contains both PIDs from parent patterns; or (2) is a new node and contains one PID (from its parent).

Participant expansion lemma: For type I join pattern, if an intersecting node instance satisfies participant rule, then its neighbors that satisfy the rule must also be a participant of the newly merged pattern. More generally, if a node instance satisfies participant rule, then its neighbors that satisfy the rule must also be a participant of the newly merged pattern.

By using participant rule, the PE algorithm can expand from the validated intersecting node (Lemma-2) to all participating feature instances. The count of participating feature instances is calculated during the process so that CPI can be directly computed by the end of expansion. The PE algorithm has an additional level of fast filtering before expansion come into play. By using Lemma-2, it will first perform a
quick linear scan on instances of intersecting node only, which can yield CPR of intersecting node. Since CPR is an upper bound of CPI, PE algorithm can terminate without moving into expansion phase if CPR is already smaller than our pre-defined threshold.

5.2.1 Type-II merge

For Type-II merge, the lemmas defined for Type-I merge cannot be strictly applied. The lemmas initially depend on the correctness of intersecting node instances. For Type-I merge, the correctness can be guaranteed by checking PIDs. However, checking PIDs does not consider undirected loops created in Type-II pattern so the connectivity of the undirected loop cannot be guaranteed. Although PE algorithm cannot compute exact CPIs for Type-II merge pattern, the non-exact CPIs computed using the same procedure are proven to be upper bounds of the exact CPIs. In this case, PE algorithm works as a filtering step for pattern evaluation in Type-II merge.

![Figure 5 Undirected loop unchecked for Type-II merge](image)

### 5.3 MCS acceleration

A dynamic updating framework is proposed to filter out unnecessary CPI evaluations in MCS trials. According to the anti-monotonicity property of CPI (Lemma-3), a pattern cannot pass a CPI value if its subset fails to pass. In this case, a pattern is not evaluated during a MCS trial if any of its subset yields a non-passing CPI value.

**Lemma-3**: If $\text{CSTP}_1 \subseteq \text{CSTP}_2$, then $\text{CPI}(\text{CSTP}_1) \geq \text{CPI}(\text{CSTP}_2)$.

![Figure 6 Dynamic updating framework for MCS acceleration](image)

In a simulated trial within Monte Carlo simulation, we are only interested in the relationship between simulated CPIs and observed CPIs. Thus, in a simulated trial, if the lowest CPI of pattern parents, which is an upper bound of CPI of child pattern, is already lower than the observed CPI, the exact CPI must also be lower. In this case, the relationship between observed and simulated CPI is already determined and there
is no need to further evaluate the exact CPI. The upper bounds or exact CPIs of patterns with size-k are passed onto patterns with size k+1 in the Apriori tree so the pruning information is dynamically updated to the next pattern level.

6. Validation

The algorithm is implemented in Java and all the experiments are done on a CORE i-7 laptop (ultra-low volt) with 8GBs of RAM. The baseline approach (spatiotemporal join in [12]) is still under implementation for run-time comparison. The current run-time comparison is done using the reported run-time in [12] as a rough estimation.

6.1 Experiments with synthetic dataset

The synthetic dataset is generated with three types of features, namely A, B and C. Each feature contains the same number of point instances, which vary from 100 to 100,000 in the controlled experiments. There are three artificial patterns in the synthetic dataset: A → B, B → C, and A → B → C. One third of point instances from each feature type are participants of the three artificial pattern while the other two thirds follow a homogeneous Poisson point process in the spatiotemporal framework. CPI threshold is set to 0.1. The number of MCS trials used is 1,000 and p-value threshold is 0.01.

The criteria for correctness and completeness is as follows:

**Correctness criterial:** Patterns that occur purely due to randomness should be removed from the result. In this experiment, only A → B, B → C, and A → B → C are valid patterns and all the others, if occurred, are chance patterns caused by the random point process.

**Completeness criterial:** All the artificially created patterns should be preserved in the results since they do not happen by randomness. In this experiment, A → B, B → C, and A → B → C must be kept as valid patterns through significance test.

<table>
<thead>
<tr>
<th>All results:</th>
<th>Significant patterns:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pattern1 (cpi-0.43): A-B</td>
<td>Pattern1 (cpi-0.43, p-0.0): A-B</td>
</tr>
<tr>
<td>Pattern2 (cpi-0.12): A-C</td>
<td>Pattern4 (cpi-0.42, p-0.0): B-C</td>
</tr>
<tr>
<td>Pattern3 (cpi-0.14): B-A</td>
<td>Pattern7 (cpi-0.36, p-0.0): A-B-C</td>
</tr>
<tr>
<td>Pattern4 (cpi-0.42): B-C</td>
<td>Chancy patterns:</td>
</tr>
<tr>
<td>Pattern5 (cpi-0.11): C-A</td>
<td>Pattern2 (cpi-0.12, p-0.874): A-C</td>
</tr>
<tr>
<td>Pattern6 (cpi-0.13): C-B</td>
<td>Pattern3 (cpi-0.14, p-0.117): B-A</td>
</tr>
<tr>
<td>Pattern7 (cpi-0.36): A-B-C</td>
<td>Pattern5 (cpi-0.11, p-0.929): C-A</td>
</tr>
<tr>
<td></td>
<td>Pattern5 (cpi-0.13, p-0.394): C-B</td>
</tr>
</tbody>
</table>

Figure 7 Significant patterns in synthetic dataset

Figure 7 shows the significant patterns identified in the synthetic dataset. Originally 7 patterns are discovered using the CPI threshold. Three artificial patterns are
successfully detected as non-random patterns while other four patterns occurred purely by chance are filtered out by significance test although they have CPIs higher than the threshold.

Figure 8 shows the run-time of proposed approach. The number of points is the total number of points combined from instances of the three feature types. Since the baseline approach [11] has not been implemented, a roughly estimated comparison can be made with the reported run-time in the paper. As reported, the run-time for 30,000 points is about 5,000 seconds (with two identified patterns). The run-time for the same number of points is about 0.253 seconds for the new method. Again, this is not a scientific comparison since the implementation platform is different and the data is different. In the next step, the baseline algorithm will be implemented in Java and run with the same synthetic dataset in the same environment in order to make a valid comparison.

Since in Monte Carlo simulation, the neighbor graph needs to be generated on the fly. The graph building time is also compared between a naïve brute force approach and a filter-and-refine approach using spatiotemporal partitioning. **For graph building time, the sharp increase from 4.762 to 556.213 seconds (30000 to 300000 points) is caused by simple filtering steps in the original naïve approach.** In the naïve approach, instead of checking all conditions before determining whether an edge should be added, the conditions are checked in a cost-wise low-to-high order. For example, instead of computing the spatial distance, the temporal distance is firstly checked, followed by checks on X-distance and Y-distance before all the square and square root operations. As the total number of points increases, the same space of study area is increasingly filled up, leading to a denser graph. This indicates the number of point pairs passing the early simple filters also gets larger and larger, which further increases the time cost.

<table>
<thead>
<tr>
<th>Graph building time (input: feature instances)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of points</td>
</tr>
<tr>
<td>300</td>
</tr>
<tr>
<td>3000</td>
</tr>
<tr>
<td>30000</td>
</tr>
<tr>
<td>300000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Mining time (input: ST graph)</th>
<th>MCS time (1000 trials)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of points</td>
<td>Mining time (s)</td>
</tr>
<tr>
<td>300</td>
<td>0.013</td>
</tr>
<tr>
<td>3000</td>
<td>0.049</td>
</tr>
<tr>
<td>30000</td>
<td>0.253</td>
</tr>
<tr>
<td>300000</td>
<td>1.153</td>
</tr>
</tbody>
</table>

Figure 8 Significant patterns in synthetic dataset

Through the time comparison we can see the improved efficiency allows a complete run of Monte Carlo simulation within feasible time on a single machine. A full comparison to the baseline approach will be made after its implementation in Java.
6.2 Experiments with real dataset

A good real dataset for case study is still under search. The goal is to find an interesting use-case in the domain of Food, Energy and Water Nexus. In addition, we are also working on extending the problem definition with tele-coupling and teleconnection examples to break the limit of a constrained spatiotemporal interaction window. In that case, more real datasets may be available for experiments.

7. Future work

For future work, we plan to extend this work with two parts:

1) Problem definition

Firstly, so far all cascading work only considers relationship between nearby instances in the spatiotemporal framework. In real world phenomenon, cascading relationships can be observed on a much larger scale due to hidden natural process such as El Nino or tele-communication such as Internet messages. Thus, we would like to extend the range of cascading from ST-neighbors to tele-connections.

Secondly, the inputs are only point dataset so far. However, the real world phenomenon are not bounded by point features. For example, snow and floods are all regional events not represented by point dataset. Thus, we would like to extend input types from ST point instances to ST-polygons (e.g., snow, flood) and ST-rasters (e.g., land use, land cover).

Thirdly, we would like to extend feature types by generating events using locations and time-interval. In this case, we not only can find relationships between actual events but also relationships between specific locations and time intervals.

2) Case study

We would like to collaborate with domain experts in FEW Nexus to find use-cases of this approach so we can output some interesting patterns which can assist development of domain scientific research.

References