Efficient Discovery of Spatial Co-evolving Patterns in Massive Geo-sensory Data

Chao Zhang\textsuperscript{1}, Yu Zheng\textsuperscript{2}, Xiuli Ma\textsuperscript{3}, Jiawei Han\textsuperscript{1}
\textsuperscript{1}UIUC, \textsuperscript{2}Microsoft, \textsuperscript{3}PKU

czhang82@illinois.edu
Big Geo-Sensory Data is Ubiquitous

- Wireless sensor network (WSN): multiple sensors are deployed at different locations to monitor the target condition cooperatively.

- The geo-sensory data is becoming big
  - A modern WSN can contain hundreds of sensors, with each sensor collecting millions of records.

![Map of Beijing with sensor locations and a line graph showing Air Quality Index over time for three sensors: Sensor 1, Sensor 2, and Sensor 3.]}
Spatial Co-evolving Pattern: An Example

- Goal: mining a set of **spatially correlated sensors** that exhibit **frequent co-evolution**.

- Frequent co-evolution for \([s_1, s_2, s_3]\)
  - \(s_1\) decreases in AQI, \(s_2\) and \(s_3\) increase in AQI
  \([[-20/h, -15/h], [+15/h, +20/h], [+15/h, +20/h]]\)
  - Caused by traffic flow in off-work hours
Spatial Co-evolving Pattern Mining

• A spatial co-evolving pattern (SCP) contains
  ▸ a set of spatially connected sensors
  ▸ the frequent co-evolution in their readings
• We aim to find all SCPs from the input geo-sensory data.
Why is it a Challenging Problem?

- The truly interesting evolutions are often flooded by numerous trivial fluctuations.
  - Existing motif discovery methods can only find trivial motifs from such data.
Why is it a Challenging Problem?

- The combinatorial nature of SCP leads to an extremely large search space.
  - An arbitrary number of sensors.
  - The occurring time intervals of an SCP is uncertain.

Spatial combination

Temporal combination
Assembler: A Two-stage Approach

- **Stage I**: find frequent evolutions for individual sensors.
- **Stage II**: assemble individual evolutions into SCPs based on the spatial constraint.
Stage I: Mining Frequent Evolutions for Individual Sensors

- To find interesting evolutions, we must filter trivial fluctuations and identify evolving intervals.
- In geo-sensory data, the changes occur with different rates and durations.
Extract Evolutions using Wavelet Transform

- We capture multi-scale changes using wavelet transform.
  - In the wavelet space, the coefficients of different bases measure the strengths of changes.
  - We preserve large coefficients and discard small coefficients.
Detecting Frequent Evolutions

- After extracting evolving intervals in the time series, we detect frequent evolutions via clustering.

- A segment-and-group approach:
  - Partition each interval into line segments.
  - Use mean shift to cluster the line segments based on slope (change rate).

![Diagram of detecting frequent evolutions](image.png)
Stage II: SCP Generation

- From individual sensors to groups of sensors
  - Pattern assembling via timestamp intersection.

<table>
<thead>
<tr>
<th>Pattern $P_1$</th>
<th>[+20/h, +50/h]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Timestamps</td>
<td>{t_1, t_3, t_4, t_7, t_9, t_{10}, t_{11}, t_{12}, t_{13}, t_{14}}</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Pattern $P_2$</th>
<th>[-30/h, -10/h]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Timestamps</td>
<td>{t_2, t_4, t_5, t_7, t_9, t_{10}, t_{11}, t_{12}, t_{13}, t_{17}}</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Pattern $P_{12}$</th>
<th>{[+20/h, +50/h], [-30/h, -10/h]}</th>
</tr>
</thead>
<tbody>
<tr>
<td>Timestamps</td>
<td>{t_4, t_7, t_9, t_{10}, t_{11}, t_{12}, t_{13}}</td>
</tr>
</tbody>
</table>
Stage II: SCP Generation

- **Anti-monotonicity**: if one set of sensors have SCP, any of its subsets must also have SCP.

- A baseline method based on Apriori: starting with patterns on individual sensors, obtain SCPs in a bottom-up manner.

- The baseline method is not efficient enough
  - It generates numerous candidates and keep them in memory.
  - It performs pair-wise comparison to examine whether two candidates can be joined.
Stage II: SCP Generation

- Can we leverage the spatial constraint to generate SCPs more efficiently?

- The connectivity graph
  - Each set of spatially connected sensors corresponds to a connected component in the connectivity graph.
Parent Relation

- We define the parent relation between two connected components.

**Definition 10 (Parent).** Let $Y$ be a size-$(k+1)$ connected component in a connectivity graph $G$. Given a vertex ordering $\mathcal{V}$, the roll-up operation on $Y$ removes one vertex $s$ from $Y$ such that:
  1. the result set $X = Y - \{s\}$ is still connected;
  2. $s$ is the first possible vertex in $\mathcal{V}$ on the premise of satisfying Condition (1).
We say $X$ is the parent of $Y$, and $Y$ is a child of $X$.

Example:

Suppose $V = 1 \to 2 \to 3 \to 4 \to 5 \to 6$, then the parent relation generates:

$\{245\} \to \{25\} \to \{5\} \to \emptyset$
The SCP Search Tree

- Starting from any connected component, by performing the roll-up operation, we can reach the same node $\phi$.

- A tree structure: each node is a connected component along with the SCPs occurring on it.
Reverse Search of SCPs

- Starting from the root node, we perform depth-first construction of the SCP search tree
  - SCPs are obtained on-the-fly
  - Unqualified branches are pruned with anti-monotonicity.
Experimental Evaluation

• Data sets
  - Air: the AQI data collected by 180 sensors in northern China during 1.5 years.
  - Bike: the bike rental data of 332 docks in New York during one year.
Example SCPs

- On the Air data set:
Example SCPs

- On the Bike data set:
Running Time Comparison

- Efficiency

- Scalability
Summary

• We study the problem of mining spatial co-evolving patterns from massive geo-sensory data.

• We propose the two-stage method Assembler:
  ▶ Stage 1: it obtains frequent evolutions for individual sensors.
  ▶ Stage 2: it assembles single patterns into SCPs.

• The experiment results show that Assembler is effective and efficient.