PRESS: A Novel Framework of Trajectory Compression in Road Networks

Renchu Song, Weiwei Sun, Baihua Zheng, Yu Zheng
Background

• An increasingly huge volume of trajectory data are being collected today.
  › E.g., vehicle tracking

• The trajectory data can support a number of important applications.
  › Taxi scheduling
  › Travel route recommendation
  › Transportation design
Background

- A trajectory is a sequence of time-ordered spatial points.

\[(x_1, y_1, t_1), (x_2, y_2, t_2), (x_3, y_3, t_3) \ldots\]

- The “huge volume” problem of trajectory data:
  - The number of trajectories in the database can be large.
  - Each trajectory can consist of many redundant spatial points.

- Huge volume \(\rightarrow\) expensive data storage, high communication cost, etc.
Background

• How can we effectively compress the trajectory data in a road network?

• Goals
  ‣ Effectively reduce the size of trajectory data without incurring too much information loss.
  ‣ Support the efficient processing of common spatial queries.
PRESS: An Overview

- Map matcher
- Trajectory re-formatter
- Spatial compressor
- Temporal compressor
- Query processor
PRESS: Key Features

- A novel representation to separate the spatial and temporal information
- Lossless spatial compression and error-bounded temporal compression.
- Support common spatiotemporal queries without fully decompressing the data.
Trajectory Representation

• A road network is a directed graph $G=(V, E)$.

• A separate representation of trajectory
  ▶ Spatial side: a sequence of consecutive edges in the graph.
  ▶ Temporal side: a sequence of timestamps.
Trajectory Representation

- Spatial side: a sequence of consecutive edges in the graph.

\[(e_1, e_2, e_3, ..., e_n)\]

- Temporal side: a sequence of \((d_i, t_i)\) tuples.
  - \(d_i\) is the network distance the object has traveled at time \(t_i\) since the start of the trajectory.
Spatial Compression

• How do we compress a sequence of consecutive edges \((e_1, e_2, e_3, ..., e_n)\)?

• Two key techniques
  ‣ Shortest path compression
  ‣ Frequent sub-trajectory compression
Shortest Path Based Compression

- Intuition: objects tend to take the shortest path between two spatial locations.
- Given two edges $e_i$ and $e_j$, if the sub-trajectory $(e_i, e_{i+1}, ..., e_j)$ matches the shortest path between $e_i$ and $e_j$, we only record $e_i$ and $e_j$ for that sub-trajectory.
Frequent Path Based Compression

• Given a large number of trajectories, many of them may share certain frequent sub-trajectories.

• Suppose we extract the frequent sub-trajectories, we can use certain coding scheme to compress the trajectory data.

• Key steps:
  ‣ Frequent Sub-Trajectory (FST) extraction
  ‣ Huffman-based coding to compress the trajectory data
FST Extraction

- Consider each spatial path (a sequence of edges) as a string.
- With the input trajectories, build a trie to count the frequency of different sub-trajectories.
FST Extraction

- Set a threshold to limit the maximal length for sub-trajectory extraction.

\[ TD = \{ T_{s1}=\langle e_1, e_5, e_8, e_6, e_3 \rangle, \\
T_{s2}=\langle e_1, e_5, e_2, e_1, e_4, e_8 \rangle, \\
T_{s3}=\langle e_2, e_1, e_4, e_6 \rangle \} \]

**Sub-trajectories =**

\[ \{ \langle e_1, e_5, e_8 \rangle, \langle e_5, e_8, e_6 \rangle, \langle e_8, e_6, e_3 \rangle, \langle e_6, e_3 \rangle, \langle e_3 \rangle, \langle e_1, e_5, e_2 \rangle, \\
\langle e_5, e_2, e_1 \rangle, \langle e_2, e_1, e_4 \rangle, \langle e_1, e_4, e_8 \rangle, \langle e_4, e_8 \rangle, \langle e_8 \rangle, \langle e_2, e_1, e_4 \rangle, \\
\langle e_1, e_4, e_6 \rangle, \langle e_4, e_6 \rangle, \langle e_6 \rangle \} \]
Frequent Path Based Compression

- Once the sub-trajectories are extracted and counted, next steps are:
  - Use the Aho-Corasick string matching algorithm to decompose a trajectory into non-overlapping sub-trajectories.
  - Performs Huffman encoding on the sub-trajectories.

✓ The more frequent a sub-trajectory is, the shorter the code is expected to be.
Frequent Path Based Compression

- An example:

<table>
<thead>
<tr>
<th>input $T'$</th>
<th>$\langle e_1, e_4, e_7, e_5, e_8, e_6, e_3, e_1, e_5, e_2, e_{10} \rangle$</th>
</tr>
</thead>
<tbody>
<tr>
<td>decomposition</td>
<td>$\langle e_1, e_4 \rangle, \langle e_7 \rangle, \langle e_5 \rangle, \langle e_8, e_6, e_3 \rangle, \langle e_1, e_5, e_2 \rangle, \langle e_{10} \rangle$</td>
</tr>
<tr>
<td>Trie nodes</td>
<td>16, 22, 4, 9, 10, 24</td>
</tr>
<tr>
<td>Huffman code</td>
<td>0111, 01010000, 1111, 01001, 00110, 0101001</td>
</tr>
<tr>
<td>Result</td>
<td>011101010000111101001001100101001</td>
</tr>
</tbody>
</table>
Temporal Compression

• Recall the temporal representation of a trajectory:

  \[(d_1, t_1), (d_2, t_2), (d_3, t_3) \ldots\]

• We want to compress the sequence without introducing too large errors.
Two Error Metrics

Definition 1  (Time Syn. Network Dis. (TSND)).
Given a trajectory $T$ and its compressed one $T'$, TSND measures the maximum difference between the distance object travels via trajectory $T$ and that via trajectory $T'$ at any time slot with $TSND(T, T') = \max_{t_x} (|\text{Dis}(T, t_x) - \text{Dis}(T', t_x)|)$.

Definition 2  (Network Syn. Time Diff. (NSTD)).
NSTD defines the maximum time difference between a trajectory $T$ and its compressed form $T'$ while traveling any same distance with $NSTD(T, T') = \max_{d_x} (|\text{Tim}(T, d_x) - \text{Tim}(T', d_x)|)$. 

![TSND Diagram](a)

![NSTD Diagram](b)
Error-Bounded Temporal Compression

• Use the Before Opening Window (BOPW) algorithm \(^{[1]}\) to compress the sequence \(T\).
  
  ‣ Scan the tuples in \(T\) sequentially, try to skip \((d_{i+1}, t_{i+1})\) by linking \((d_i, t_i)\) and \((d_{i+2}, t_{i+2})\) if the introduced errors (TSND and NSTD) are within the tolerance threshold.

Supporting Common Queries

• Three common spatiotemporal queries are supported by PRESS without decompressing the data:

  ‣ Where: given a timestamp $t$, return the location in the trajectory at timestamp $t$.
  
  ‣ When: given a location $(x, y)$ in the trajectory, return the timestamp when the object is at $(x, y)$.
  
  ‣ Range: given a spatial region $R$ and a temporal range $[t_1, t_2]$, check whether the trajectory pass $R$ during the interval $[t_1, t_2]$. 

Thursday, September 18, 14
Experiments

• Data set:
  
  ‣ A real taxi trajectory data set from one of the largest taxi companies in Singapore.
  
  ‣ ~465,000 trajectories generated by ~15,000 taxis.
  
  ‣ Original storage cost: 13.2GB.
Compression Effectiveness

- Effects of SP compression and FST compression
Compression Effectiveness

- Effects of temporal compression
  - BTC: considers only the effect of temporal compression
  - PRESS: considers both spatial and temporal compression
Comparison with Alternative Methods

- MMTC [1]: map-matched trajectory compression
- Nonmaterial [2]


Query Performance Comparison

**Figure 14:** Performance of \textit{where}_{at} query

**Figure 15:** Performance of \textit{when}_{at} query

**Figure 16:** Performance of \textit{range} query
Summary

• PRESS is a novel framework for compressing trajectory data in road networks.
  ‣ Lossless spatial compression
  ‣ Error-bounded temporal compression
  ‣ Fully support common spatiotemporal queries without decompression

• The experiments on real data demonstrate the effectiveness and efficiency of PRESS.
Thanks!