The Multi-Rule Partial Sequenced Route Query

Wei-Shinn Ku

Presentation Outline

- Introduction
- Related Work
- The MRPSR Query
- System Design
- Experimental Validation
- Conclusion
Introduction

- In GIS systems, much work has focused on efficiently answering spatial queries.
  - How to efficiently answer fundamental spatial query types?
  - Nearest neighbor query, range query…
- More complex spatial query types must be considered.

Auburn University
Computer Science and Software Engineering

Introduction

- A novel query type, MRPSR.
  - Objective: assist users to plan trips that involve several POIs of different categories based on a number of traveling rules.
  - Outcome: a route with the shortest distance.
- Traveling rules: constraints expressed as sub-sequence of POI categories.
  - May only involve a subset of the categories.
Introduction

- An example
  - Alice’s trip: a bank, a restaurant, a gas station and a movie theater
- Rules:
  - Visit a bank to withdraw money before having lunch at a restaurant.
  - Fill up gas before going to watch a movie.
- How to find a least-cost route?
Presentation Outline

- Introduction
- Related Work
- The MRPSR Query
- System Design
- Experimental Validation
- Conclusion

Related Work

- The Traveling Salesman Problem (TSP)
  - Difference:
    - With TSP, each POI must be visited exactly once, no category.
    - With MRPSR, each POI is associated with a category and one may select any element of that category.
Related Work

- The Trip Planning Query (TPQ)
  - Based on POI categories
  - Users ask for an optimal route through exactly one POI in each category.
  - Proven to be NP-hard
  - Comparison with MRPSR:
    - With TPQ, user specifies no order on the POI categories to be visited.
    - With MRPSR, users can specify some sequences on the POI categories to be visited.

Related Work

- The Optimal Sequenced Route (OSR) Query
  - Users ask for an optimal route through exactly one POI in each category in a particular order imposed on all the categories.
  - Comparison with MRPSR:
    - With OSR, users impose a complete order on all the POI categories to be visited.
    - With MRPSR, users can impose some partial orders on the POI categories to be visited.
Related Work

- The Multi-Type Nearest Neighbor (MTNN) Query
  - An extended solution of OSR with the assumption that the POI type number is small.
- The Sequential Ordering Problem (SOP)
  - Given a graph with vertices and weighted edges, find a minimal cost Hamiltonian path from the start vertex to the terminal vertex which also observes precedence constraints.
  - Comparison with MRPSR
    - With SOP, no POI categories.
    - With MRPSR, no Hamiltonian path is sought.

Presentation Outline

- Introduction
- Related Work
- The MRPSR Query
- System Design
- Experimental Validation
- Conclusion
The MRPSR query

- Problem Formulation
  - The MRPSR query is to ask for a route that satisfies the following three requirements:
    - Traverse through exactly one POI in each category.
    - The traveling distance is minimized.
    - Conforms with the traveling rules.
  - A partial sequenced rule
    - defined as an ordered subset of POI categories.
    - e.g., $C_{ATM} \rightarrow C_{Supermarket} \rightarrow C_{ATM} \rightarrow C_{Restaurant}$

Properties of the MRPSR Query

- The Trip Planning Query (TPQ) and the Optimal Sequenced Route (OSR) Query are special cases of the MRPSR query.
  - When the set of partial sequence rules is empty
  - When the set of partial sequence rules only contains one complete sequence involving all the POI categories
- The problem of the MRPSR Query is NP-hard.
- The set of the partial sequence rules
  - Compatible: there exists a total order of categories that satisfies the sequence specified in each of the rules in the set.
  - Counter-example: $\{C_1 \rightarrow C_2, C_2 \rightarrow C_3, C_3 \rightarrow C_1\}$
The MRPSR query

- Properties of The MRPSR Query
  - Solvability
    - If a MRPSR query is solvable, then the corresponding set of the partial sequence rules must be compatible.
    - Otherwise, no matter how POI's are selected, it will be impossible to order them so that the ordered sequence meets all of the constraints.

Presentation Outline

- Introduction
- Related Work
- The MRPSR Query
  - System Design
- Experimental Validation
- Conclusion
System Design

- Activity On Vertex (AOV) network
  - Before finding a near-optimal route to fulfill all the partial sequence rules, we need a solution to verify if they are compatible.
  - Represents POI categories as vertices and prerequisites as edges.
  - The rule set is compatible if and only if the corresponding AOV network is a directed acyclic graph.
  - Otherwise, the trip is infeasible, i.e., the AOV has a cycle.
  - Whenever the count of a vertex drops to zero (in-degree = 0), we place the vertex in a list ($L_{\text{zero}}$).

---

System Design (Cont.)

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Name</th>
<th>Prerequisites</th>
</tr>
</thead>
<tbody>
<tr>
<td>C1</td>
<td>Bank</td>
<td>None</td>
</tr>
<tr>
<td>C2</td>
<td>Bookstore</td>
<td>None</td>
</tr>
<tr>
<td>C3</td>
<td>Restaurant</td>
<td>C1, C2</td>
</tr>
<tr>
<td>C4</td>
<td>Gas Station</td>
<td>None</td>
</tr>
<tr>
<td>C5</td>
<td>Hospital</td>
<td>C4</td>
</tr>
<tr>
<td>C6</td>
<td>Shopping Center</td>
<td>C5</td>
</tr>
<tr>
<td>C7</td>
<td>Church</td>
<td>C3, C6</td>
</tr>
<tr>
<td>C8</td>
<td>Coffee Shop</td>
<td>C3</td>
</tr>
<tr>
<td>C9</td>
<td>Gift Shop</td>
<td>C7, C8</td>
</tr>
<tr>
<td>C10</td>
<td>Park</td>
<td>C7</td>
</tr>
</tbody>
</table>

![Diagram of AOV network]
System Design – NNPSR

- The Nearest Neighbor-based Partial Sequence Route (NNPSR) algorithm
  - By utilizing both the $L_{\text{zero}}$ list and any well-known nearest neighbor query algorithm to generate an efficient route.
  - Search for the nearest POI whose category is included in $L_{\text{zero}}$ from the last point.
  - At each step, update the adjacency list and $L_{\text{zero}}$ list.

System Design – NNPSR-LORD

- Nearest Neighbor-based Partial Sequence Route with the Light Optimal Route Discoverer (NNPSR-LORD) Algorithm
  - Obtaining a complete POI sequence by the NNPSR algorithm, we can further shorten the route distance by combining NNPSR with LORD.
  - Return the optimal route following the sequence discovered by NNPSR
The Advanced A* Search-based Partial Sequenced Route (AASPSR) Algorithm

- A* Search
  - Considering the location of the destination
  - Retrieving POI's with the minimum sum of distances to the starting point and destination (major axis of the ellipse)
- Advanced A* Search
  - A* Search + NNPSR to avoid roundabout ways
System Design – AASPSR (Cont.)

Presentation Outline

- Introduction
- Related Work
- The MRPSR Query
- System Design
- Experimental Validation
- Conclusion
Experimental Validation

- MRPSR problem is *NP-hard*
- LORD-based brute-force solution
  - Get the optimal route distance and the corresponding response time.

The Setup

- The percentage of the constrained categories (PCC), the average category cardinality (ACC), and the number of total categories (NTC)
- Real California dataset
  - 63 categories
- Synthetic dataset
  - POI's generated randomly within the region of California (uniform distribution)

---

Experimental Validation – California Dataset

<table>
<thead>
<tr>
<th>Category</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Building</td>
<td>4110</td>
</tr>
<tr>
<td>Church</td>
<td>7680</td>
</tr>
<tr>
<td>Hospital</td>
<td>835</td>
</tr>
<tr>
<td>Locale</td>
<td>13481</td>
</tr>
<tr>
<td>Park</td>
<td>6728</td>
</tr>
<tr>
<td>School</td>
<td>11173</td>
</tr>
<tr>
<td>Populated place</td>
<td>6900</td>
</tr>
<tr>
<td>Summit</td>
<td>5594</td>
</tr>
<tr>
<td>Valley</td>
<td>7596</td>
</tr>
</tbody>
</table>
Experimental Validation

- Effect of the Percentage of the Constrained Categories
  - Route distance (6 categories, cardinality 6000)

Fig. 5(a) California dataset
Fig. 5(b) Synthetic dataset

Experimental Validation

- Effect of the Percentage of the Constrained Categories
  - Response time (6 categories)

Fig. 8(a) NNPSR, AASPSR, and LORD-based brute-force (real)
Fig. 8(b) NNPSR-LORD and LORD-based brute-force (real)
Experimental Validation

- Effect of the Percentage of the Constrained Categories
  - Response time (6 categories, cardinality 6000)

- Effect of the Average Category Cardinality
  - Route distance (6 categories)

Fig. 6(c) NNPSR, AASPSR, and LORD-based brute-force (synthetic)

Fig. 6(d) NNPSR-LORD and LORD-based brute-force (synthetic)

Fig. 6(a) Percentage of constrained categories = 25%

Fig. 6(b) Percentage of constrained categories = 60%
Experimental Validation

- Effect of the Average Category Cardinality
  - Response time (6 categories, 33% PCC)

![Graphs showing response time vs. average category cardinality for different systems with 6 categories, 33% PCC.](Figures: graphs showing response time vs. average category cardinality for different systems with 6 categories, 33% PCC.)
Experimental Validation

- Effect of the Number of Total Categories
  - Route distance (66% PCC, California dataset)

![Route distance graph]

Experimental Validation

- Effect of the Number of Total Categories
  - Response time (66% PCC, California dataset)

![Response time graph]
Experimental Validation

- Remarkable Interesting Discoveries
  - AASPSR is only suitable for trips with low PCC, for example, TPQ (PCC equals zero). With a higher PCC, the route distance of AASPSR increases dramatically.
  - AASPSR usually shows the shortest response time because the computation cost comes mainly from the selection of the POIs along the smallest ellipse.
  - AASPSR outperforms NNPSR in terms of both the route distance and response time only with very low NTC.
    - 3 Categories: OK.
    - 6 and 9 categories: a quicker response time but a longer route distance

Presentation Outline

- Introduction
- Related Work
- The MRPSR Query
- System Design
- Experimental Validation
- Conclusion
Conclusion

- Formally define the partial sequenced route planning query and prove it to be an NP-hard problem.
- Propose a Nearest Neighbor-based Partial Sequence Route (NNPSR) query algorithm, which utilizes topological sort for combining multiple traveling rules.
- Integrate NNPSR with the LORD algorithm to further reduces the trip distance.
- Design an Advanced A* Search-based Partial Sequence Route (AASPSR) query algorithm, which employs distance heuristic functions to generate efficient trips.
- Compare the performance of the above proposed algorithms with the LORD-based brute-force solution by simulations.

Questions & suggestions