Alternative Routing: k-Shortest Paths with Limited Overlap

Theodoros Chondrogiannis\(^1\)  
Panagiotis Bouros\(^{2,3}\)  
Johann Gamper\(^1\)  
Ulf Leser\(^3\)

\(^1\)Faculty of Computer Science  
Free University of Bozen-Bolzano, Italy  
(tcpond.gamper)@inf.unibz.it

\(^2\)Department of Computer Science  
Aarhus University, Denmark  
pbour@cs.au.dk

\(^3\)Department of Computer Science  
Humboldt-Universität zu Berlin, Germany  
leser@informatik.hu-berlin.de

Motivation
Finding multiple short yet different routes between two locations in a road network is a problem with various real-world applications:
- Commercial Route Planners
- Evacuation planning
- Humanitarian aid

Related Work
- Finding \(k\) dissimilar alternative paths
- Candidate sets
- Alternative graphs
- Edge penalties
- Multi-criteria optimization

Alternative Paths
Path Similarity
The similarity of a path \(p\) to another path \(p'\) is determined by their overlap ratio:

\[
Sim(p, p') = \sum_{(x, y) \in p \cap p'} w_{xy} / |p'|
\]

Alternative Path
Given a set of paths \(P\) from \(s\) to \(t\) and a similarity threshold \(\theta\), a path \(p(s \rightarrow t)\) is alternative to \(P\) if \(\forall p \in P: Sim(p, p) \leq \theta\).

k-Shortest Paths with Limited Overlap
Given a source \(s\) and a target \(t\), the k-SPwLO is a set of \(k\) paths from \(s\) to \(t\), sorted by length in increasing order, such that:
(a) the set includes the shortest path \(p_0(s \rightarrow t)\),
(b) every path is dissimilar to its predecessors w.r.t. a similarity threshold \(\theta\),
(c) all \(k\) paths are as short as possible.

Example

Result Set 1 (No constraint)

<table>
<thead>
<tr>
<th>(p)</th>
<th>(l)</th>
<th>(\theta)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shortest</td>
<td>4.0 km</td>
<td>0</td>
</tr>
<tr>
<td>1st alternative</td>
<td>4.1 km</td>
<td>75%</td>
</tr>
<tr>
<td>2nd alternative</td>
<td>4.1 km</td>
<td>70%, 42%</td>
</tr>
</tbody>
</table>

Result Set 2 (\(\theta = 50\%\))

<table>
<thead>
<tr>
<th>(p)</th>
<th>(l)</th>
<th>(\theta)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shortest</td>
<td>4.0 km</td>
<td></td>
</tr>
<tr>
<td>1st alternative</td>
<td>4.3 km</td>
<td>48%</td>
</tr>
<tr>
<td>2nd alternative</td>
<td>4.5 km</td>
<td>25%, 9%</td>
</tr>
</tbody>
</table>

A Baseline Algorithm

ALGORITHM BSL (\(G, s, t, k, \theta\))
initialize empty set \(P_{LO}\)
while \(|P_{LO}| < k \text{ and } p_t\) is not null do
\(p_t = \text{NextSP}(G, s, t)\)
compute \(V_{Sim}\) for \(p_t\)
if \(V_{Sim} \leq \theta\) then add \(p_t\) to \(P_{LO}\)
return \(P_{LO}\)

- Employs Yen’s algorithm to create new paths
- Applies no pruning; all possible paths have to be considered

OnePass Algorithm

OBSERVATION: If \(p\) is an alternative to \(P_{LO}\) i.e. \(V_{Sim}(p) \leq \theta\) then \(V_{Sim} \leq \theta\) holds for every subpath of \(p\).

ALGORITHM OnePass (\(G, s, t, k, \theta\))
initialize \(P_{LO}\) and priority queue \(Q\) with \(p_0(s, t)\)
while \(|P_{LO}| < k \text{ and } Q\) is not empty do
\([p_v, V_p] \leftarrow \text{extract label with min} \ p\) from \(Q\);
if \(\text{End\_node}(p_v) = t\) then
add \(p_v\) to \(P_{LO}\)
update \(V_{Sim}\) for all labels in \(Q\)
else
expand \(p_v\) and create new paths
compute \(V_{Sim}\) for the new paths
enqueue every new path where \(V_{Sim} \leq \theta\)
return \(P_{LO}\)

Experiments

<table>
<thead>
<tr>
<th>Datasets</th>
<th>Nodes</th>
<th>Edges</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oldenburg</td>
<td>6,105</td>
<td>14,058</td>
</tr>
<tr>
<td>San Joaquin</td>
<td>18,263</td>
<td>47,594</td>
</tr>
</tbody>
</table>

Response time varying \(k\) and \(\theta\):