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
{tchond,gamper}@inf.unibz.it

[2]Department of Computer Science
Aarhus University, Denmark
pbour@cs.au.dk

[3]Department of Computer Science
Hunboldt-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:

$$Sim(p,p') = \frac{\sum_{(n_x,n_y)\in p\cap p'} w_{xy}}{\ell(p')}$$

Alternative Path

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

k-Shortest Paths with Limited Overlap

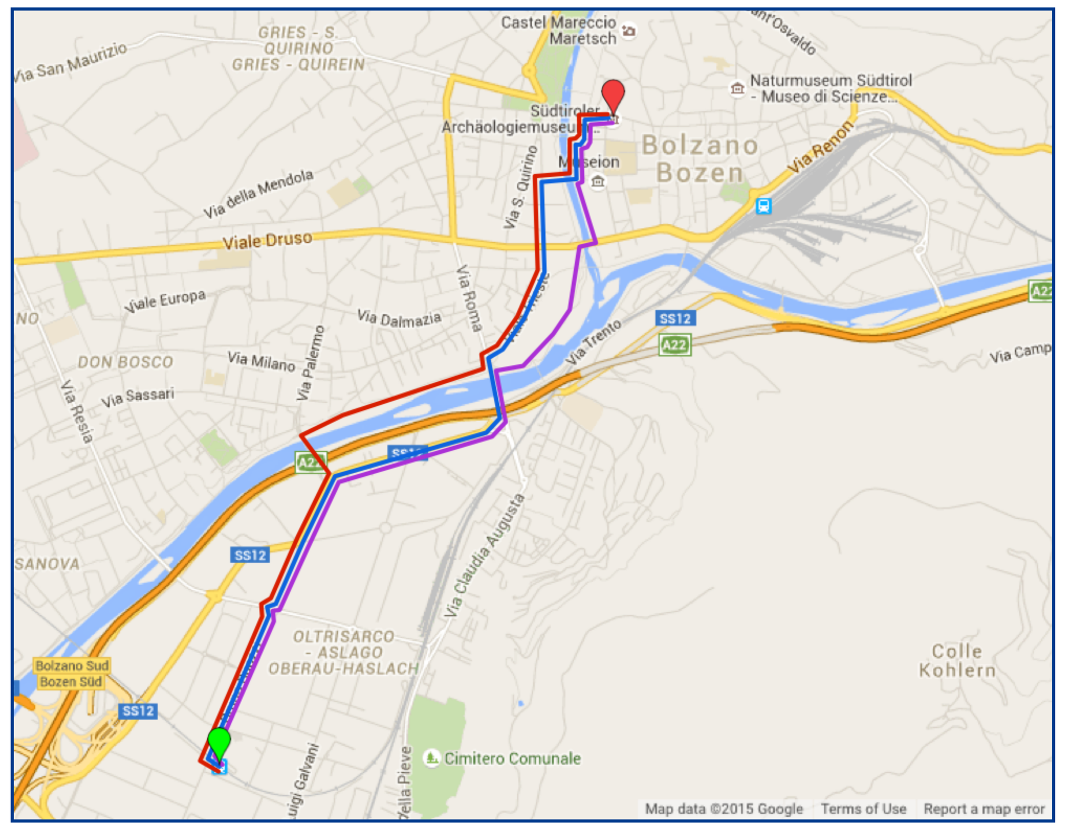
Given a source *s* and a target *t*, the **k-SPwLO** is a set of *k* paths from *s* to *t*, sorted by length in increasing order, such that:

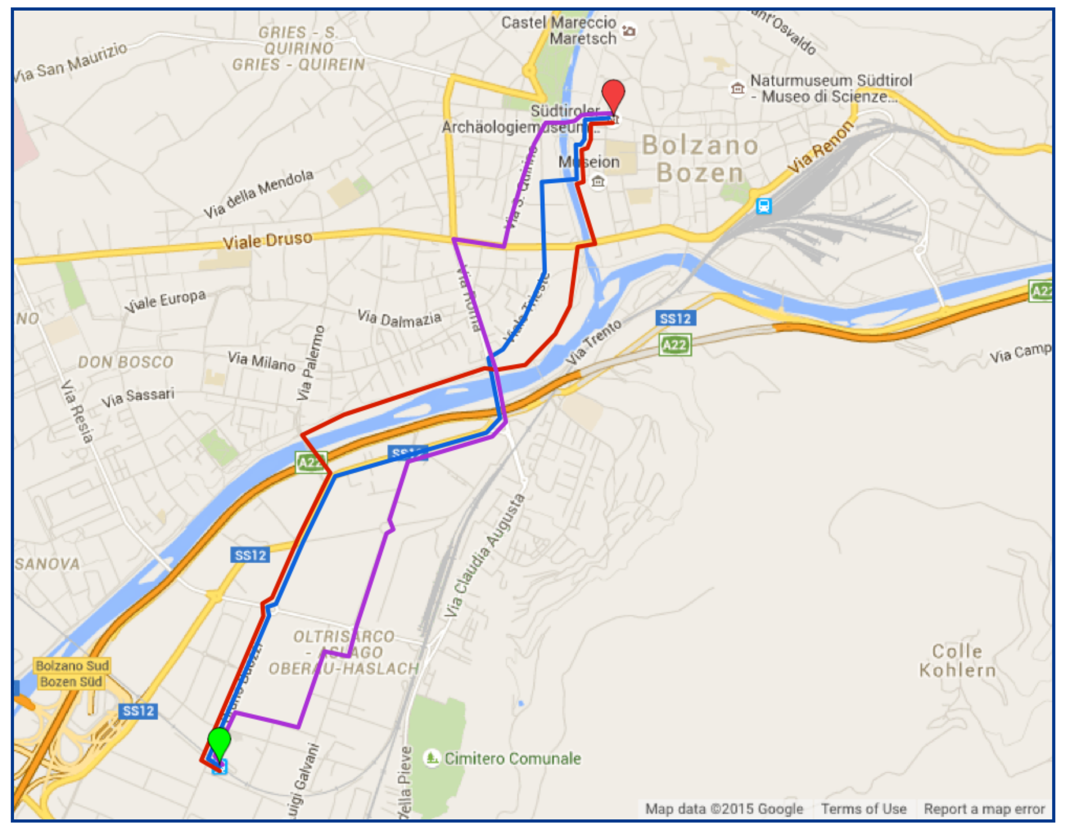
(a) the set includes the shortest path $p_0(s \rightarrow t)$,

(b) every path is dissimilar to its predecessors w.r.t. a similarity threshold *θ*,

(c) all *k* paths are as short as possible.

Example





Result Set 1 (No constraint)			
p		ℓ	V_{Sim}
Shortest path	blue	4.0 km	∅
1st alternative	red	4.1 km	{75%}
2nd alternative	purple	4.1 km	{70%, 42%}

Result Set 2 (*θ* = 50%)			
p		ℓ	V_{Sim}
Shortest path	blue	4.0 km	∅
1st alternative	red	4.3 km	{48%}
2nd alternative	purple	4.5 km	{25%, 9%}

A Baseline Algorithm

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

Experiments

Datasets	Nodes	Edges
Oldenburg	6,105	14,058
San Jose	18,263	47,594

Response time varying *k* and *θ*: