Most Diverse Near-Shortest Paths

Christian Häcker

Panagiotis Bouros

Ernst Althaus

Theodoros Chondrogiannis

JOHANNES GUTENBERG UNIVERSITÄT MAINZ

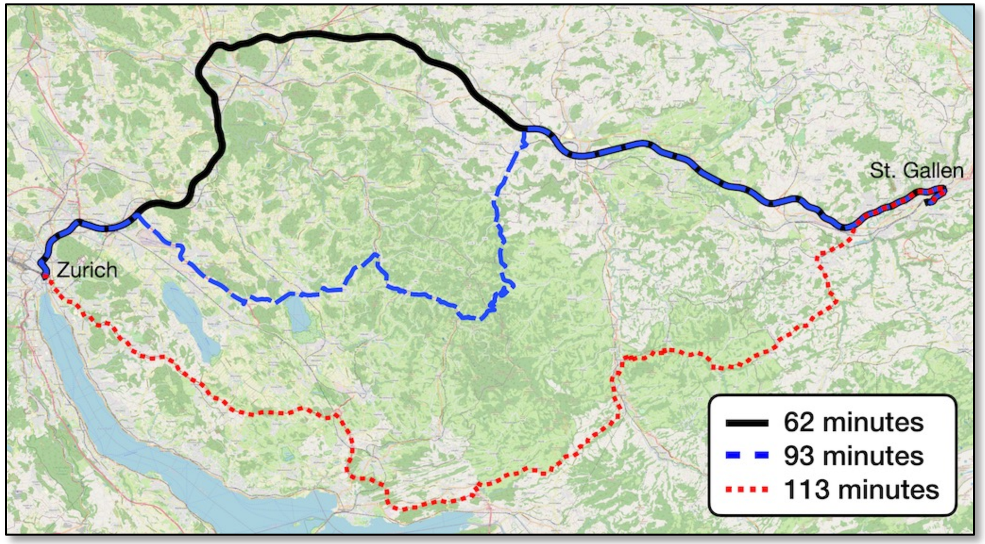
Universität Konstanz

Outline

- Motivation
- Previous work
- Problem definition
- Evaluation methods
  - An exact approach
  - Heuristic-based approaches
- Experimental analysis
- Conclusions

Motivation

- Alternative routing
  - Need to recommend a set of diverse paths
- Existing works mainly consider
  - Path length as an optimization criterion
  - Path dissimilarity as a constraint
- But,
  - Recommended paths might be too long
  - Setting dissimilarity thresholds/constraints is counterintuitive



- Our proposal, to consider
  - Path length as a constraint => near-shortest paths
  - Dissimilarity as an optimization criterion => most diverse paths

Previous work

Previous work

k-Shortest Paths with Limited Overlap
[Chondrogiannis et al., VLDBJ'20, EDBT'17, SIGSPATIAL'18]

Length as optimization criterion
Similarity as constraint

k-Dissimilar Paths with Minimum Collective Length
[Chondrogiannis et al, SIGSPATIAL'18]
[Liu et al. TKDE'17]

Previous work

Length as optimization criterion
Similarity as constraint

k-Shortest Paths with Limited Overlap
[Chondrogiannis et al., VLDBJ'20, EDBT'17, SIGSPATIAL'18]

k-Dissimilar Paths with Minimum Collective Length
[Chondrogiannis et al, SIGSPATIAL'18]
[Liu et al. TKDE'17]

k-Diversified Shortest Paths
[Cheng et al. IWCTS@SIGSPATIAL'19]

Length and similarity as optimization criteria

Previous work

Length as optimization criterion
Similarity as constraint

k-Shortest Paths with Limited Overlap
[Chondrogiannis et al., VLDBJ'20, EDBT'17, SIGSPATIAL'18]

k-Dissimilar Paths with Minimum Collective Length
[Chondrogiannis et al, SIGSPATIAL'18]
[Liu et al. TKDE'17]

Plateau method
[Li et al., PVLDB'20]

Alternative graph
[Bader et al., ICST TAPAS'11]

Single-via Paths
[Abraham et al., JEA'13]

Pareto optimal paths
[Deling and Wagner, SEA09]
[Kriegel et al., ICDE10],
[Mouratidis ae al., ICDE10]

Large set of candidates
Post processing

k-Diversified Shortest Paths
[Cheng et al. IWCTS@SIGSPATIAL'19]

Length and similarity as optimization criteria

Previous work

Iterative approaches

Penalty-based method
[Akgün et al., EOR'00]
[Johnson et al., OSTI'93]
[Rouphail et al., AATTE'95]

Direct method
[Jeong et al., KSCE'09]
[Peng et al., VLDBJ'21]

Length as optimization criterion
Similarity as constraint

***k*-Shortest Paths with Limited Overlap**
[Chondrogiannis et al., VLDBJ'20, EDBT'17, SIGSPATIAL'18]

***k*-Dissimilar Paths with Minimum Collective Length**
[Chondrogiannis et al, SIGSPATIAL'18]
[Liu et al. TKDE'17]

Large set of candidates
Post processing

Plateau method
[Li et al., PVLDB'20]

Alternative graph
[Bader et al., ICST TAPAS'11]

Single-via Paths
[Abraham et al., JEA'13]

Pareto optimal paths
[Deling and Wagner, SEA09]
[Kriegel et al., ICDE10],
[Mouratidis ae al., ICDE10]

Length and similarity as optimization criteria

***k*-Diversified Shortest Paths**
[Cheng et al. IWCTS@SIGSPATIAL'19]

Previous work

Iterative approaches

Penalty-based method ✓
[Akgün et al., EOR'00]
[Johnson et al., OSTI'93]
[Rouphail et al., AATTE'95]

Direct method ✓
[Jeong et al., KSCE'09]
[Peng et al., VLDBJ'21]

Length as optimization criterion
Similarity as constraint

k-Shortest Paths with Limited Overlap
[Chondrogiannis et al., VLDBJ'20, EDBT'17, SIGSPATIAL'18]

k-Dissimilar Paths with Minimum Collective Length
[Chondrogiannis et al, SIGSPATIAL'18]
[Liu et al. TKDE'17]

Large set of candidates
Post processing

Plateau method
[Li et al., PVLDB'20]

Alternative graph
[Bader et al., ICST TAPAS'11]

Single-via Paths ✓
[Abraham et al., JEA'13]

Pareto optimal paths
[Deling and Wagner, SEA09]
[Kriegel et al., ICDE10],
[Mouratidis ae al., ICDE10]

Length and similarity as optimization criteria

k-Diversified Shortest Paths
[Cheng et al. IWCTS@SIGSPATIAL'19]

Problem definition

Road network G
Source s
Target t
Requested paths k
Length constraint threshold $\varepsilon \geq 0$

Problem definition

Road network *G*
Source *s*
Target *t*
Requested paths *k*
Length constraint threshold $\varepsilon \geq 0$

*k*MDNSP(*G*,*s*,*t*,*k*,*ε*)

Most Diverse Near-Shortest Paths

Set $P_{k\text{MDNSP}}$:

A. of *k* near-shortest paths

$$\forall p \in P_{k\text{MDNSP}} : \ell(p) \leq (1+\epsilon) \cdot \ell(p_s)$$

B. with the highest diversity among all path sets P_A that satisfy Condition A

$$P_{k\text{MDNSP}} = \arg\max_{\forall P \subseteq P_A} \{Div(P)\}$$

Problem definition

Road network G
Source s
Target t
Requested paths k
Length constraint threshold $\varepsilon \geq 0$

kMDNSP(G,s,t,k,ε)

Most Diverse Near-Shortest Paths

Set $P_{k\text{MDNSP}}$:

A. of k near-shortest paths

$$\forall p \in P_{k\mathrm{MDNSP}} : \ell(p) \leq (1+\epsilon) \cdot \ell(p_s)$$

B. with the highest diversity among all path sets P_A that satisfy Condition A

$$P_{k\mathrm{MDNSP}} = \arg\max_{\forall P \subseteq P_A} \{Div(P)\}$$

Path set diversity

$$Div(P) = \min_{\forall p, p' \in P} Dis(p, p')$$

Path dissimilarity

$$Dis(p, p') = 1 - \frac{\sum_{\forall (n_i, n_j) \in p \cap p'} w(n_i, n_j)}{\sum_{\forall (n_i, n_j) \in p \cup p'} w(n_i, n_j)}$$

Problem definition

Road network G
Source s
Target t
Requested paths k
Length constraint threshold $\varepsilon \geq 0$

kMDNSP(G,s,t,k,ε)

Most Diverse Near-Shortest Paths

Set $P_{k\text{MDNSP}}$:

A. of k near-shortest paths

$$\forall p \in P_{k\text{MDNSP}} : \ell(p) \leq (1+\epsilon) \cdot \ell(p_s)$$

B. with the highest diversity among all path sets P_A that satisfy Condition A

$$P_{k\text{MDNSP}} = \arg\max_{\forall P \subseteq P_A} \{Div(P)\}$$

NP-hard

Path set diversity

$$Div(P) = \min_{\forall p,p' \in P} Dis(p,p')$$

Path dissimilarity

$$Dis(p,p') = 1 - \frac{\sum_{\forall (n_i,n_j) \in p \cap p'} w(n_i,n_j)}{\sum_{\forall (n_i,n_j) \in p \cup p'} w(n_i,n_j)}$$

Exact approach

Compute all Near-Shortest Paths

Generate Candidate k-Subsets

Exact approach

Compute all Near-Shortest Paths

- Path enumeration
  - [Carlyle and Wood, Networks'05]
  - Traverse the network in depth-first fashion
  - Filter out subpaths that violate the near-shortest path constraint w.r.t. ε
- Optimization
  - Estimate lower bounds for path extensions
  - Compute shortest path tree $T_{N \rightsquigarrow t}$

Generate Candidate k-Subsets

Exact approach

Compute all Near-Shortest Paths

- Path enumeration
  - [Carlyle and Wood, Networks'05]
  - Traverse the network in depth-first fashion
  - Filter out subpaths that violate the near-shortest path constraint w.r.t. ε
- Optimization
  - Estimate lower bounds for path extensions
  - Compute shortest path tree $T_{N \rightsquigarrow t}$

Generate Candidate k-Subsets

- Dynamic programming
  - "Filling a rucksack" algorithm [Knuth'05]
  - Incrementally build a binominal tree of height k
  - Represent all subsets with up to k near-shortest paths
- Optimization
  - Upon adding a new path, diversity of subsets can only drop
  - Prune unpromising subsets

Exact approach – critique

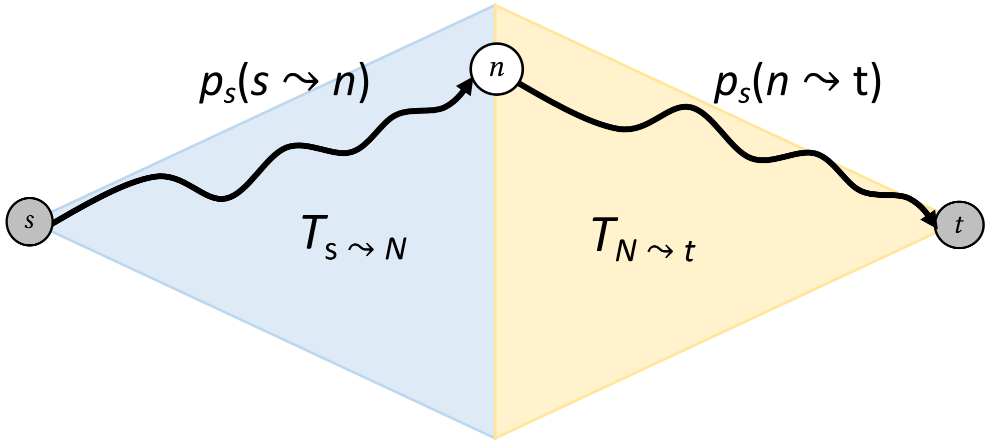
- Pros
  - Simple, straightforward

- Cons
  - Large number of near-shortest paths
  - Exponential cost
  - Impractical for real-world networks

- Solution
  - Heuristic-based methods
  - Reduce number of computed paths
  - Trade quality of the result for performance

(Simple) single-via paths

- Single-via paths (SVP)
  - [Abraham et al., JEA'13]
  - Reverse shortest path tree $T_{s \rightsquigarrow N}$
  - Shortest path tree $T_{N \rightsquigarrow t}$
  - $p_{sv}(n) = p_s(s \rightsquigarrow n) \circ p_s(n \rightsquigarrow t)$



- Simple single-via paths (SSVP)
  - $$p_{ssv}(n) = \begin{cases} p_{sv}(n), \text{ if simple} \\ p_s(s \rightsquigarrow n) \circ \text{Dijkstra}(G', n, t) \text{ or } \text{Dijkstra}(G'', s, n) \circ p_s(n \rightsquigarrow t), \text{ otherwise} \end{cases}$$

SSVP-based approach

~~Compute all Near-Shortest Paths~~

Generate Candidate k-Subsets

- Dynamic programming
  - “Filling a rucksack” algorithm [Knuth’05]
  - Incrementally build a binominal tree of height k
  - Represent all subsets with up to k near-shortest paths
- Optimization
  - Upon adding a new path, diversity of subsets can only drop
  - Prune unpromising subsets

SSVP-based approach

Compute all Near-Shortest SSVPs

- Path enumeration
  - Compute shortest path tree $T_{s \rightsquigarrow N}$
  - Compute reverse shortest path tree $T_{N \rightsquigarrow t}$
  - Construct $p_{ssv}(n)$ for each node $n \in N$

Generate Candidate k-Subsets

- Dynamic programming
  - "Filling a rucksack" algorithm [Knuth'05]
  - Incrementally build a binominal tree of height k
  - Represent all subsets with up to k near-shortest paths
- Optimization
  - Upon adding a new path, diversity of subsets can only drop
  - Prune unpromising subsets

Penalty-based approach

Built upon [Johnson et al., OSTI'93], [Rouphail et al., AATTE'95]

~~Compute all Near-Shortest Paths~~

~~Compute all Near-Shortest SSVPs~~

Generate Candidate k-Subsets

- Dynamic programming
  - "Filling a rucksack" algorithm [Knuth'05]
  - Incrementally build a binominal tree of height k
  - Represent all subsets with up to k near-shortest paths
- Optimization
  - Upon adding a new path, diversity of subsets can only drop
  - Prune unpromising subsets

Penalty-based approach

Built upon [Johnson et al., OSTI’93], [Rouphail et al., AATTE’95]

Compute Near-Shortest Paths set

- Path computation
  - Iterative approach
    - Compute a near-shortest path *p*
    - Penalize all edges on *p*
    - Repeat
- Optimization
  - Dynamically adjusted penalties

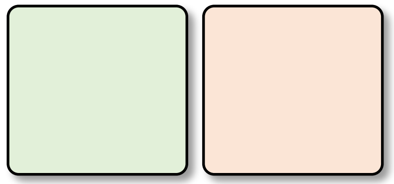
Generate Candidate k-Subsets

- Dynamic programming
  - “Filling a rucksack” algorithm [Knuth’05]
  - Incrementally build a binominal tree of height *k*
  - Represent all subsets with up to *k* near-shortest paths
- Optimization
  - Upon adding a new path, diversity of subsets can only drop
  - Prune unpromising subsets

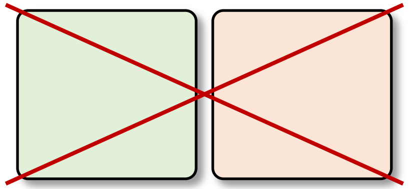
Direct approach

Inspired by [Jeong et al., KSCE'09]

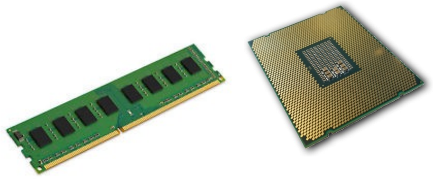
- Both SSVP- and penalty-based approaches
  - Operate in two phases, similar to exact
  - Reduce search space
  - But, still need to generate candidate *k*-subsets



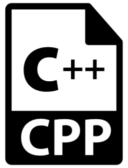
- Direct approach
  - Incrementally build $P_{k\text{MDNSP}}$ in *k-1* rounds
  - Initially, $P_{k\text{MDNSP}} = \{p_s\}$
  - In each round
    - Consider last recommended path *p*
    - Alter *p* to construct the near-shortest *p'* with the highest dissimilarity to all paths in $P_{k\text{MDNSP}}$
    - Add *p'* to $P_{k\text{MDNSP}}$



Experimental analysis

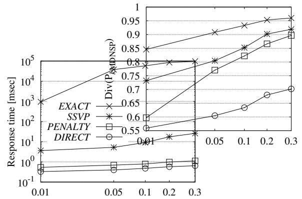


- Setup
  - 2x AMD EPYC 7351 16-Core Processors, 512 GiB 2666Mhz DDR4 RAM
  - GNU/Linux 5.4.0-66



- Methods
  - Implemented in C++, compiled with GNU G++ 9
  - *EXACT* and *SSVP*, *PENTALTY*, *DIRECT*

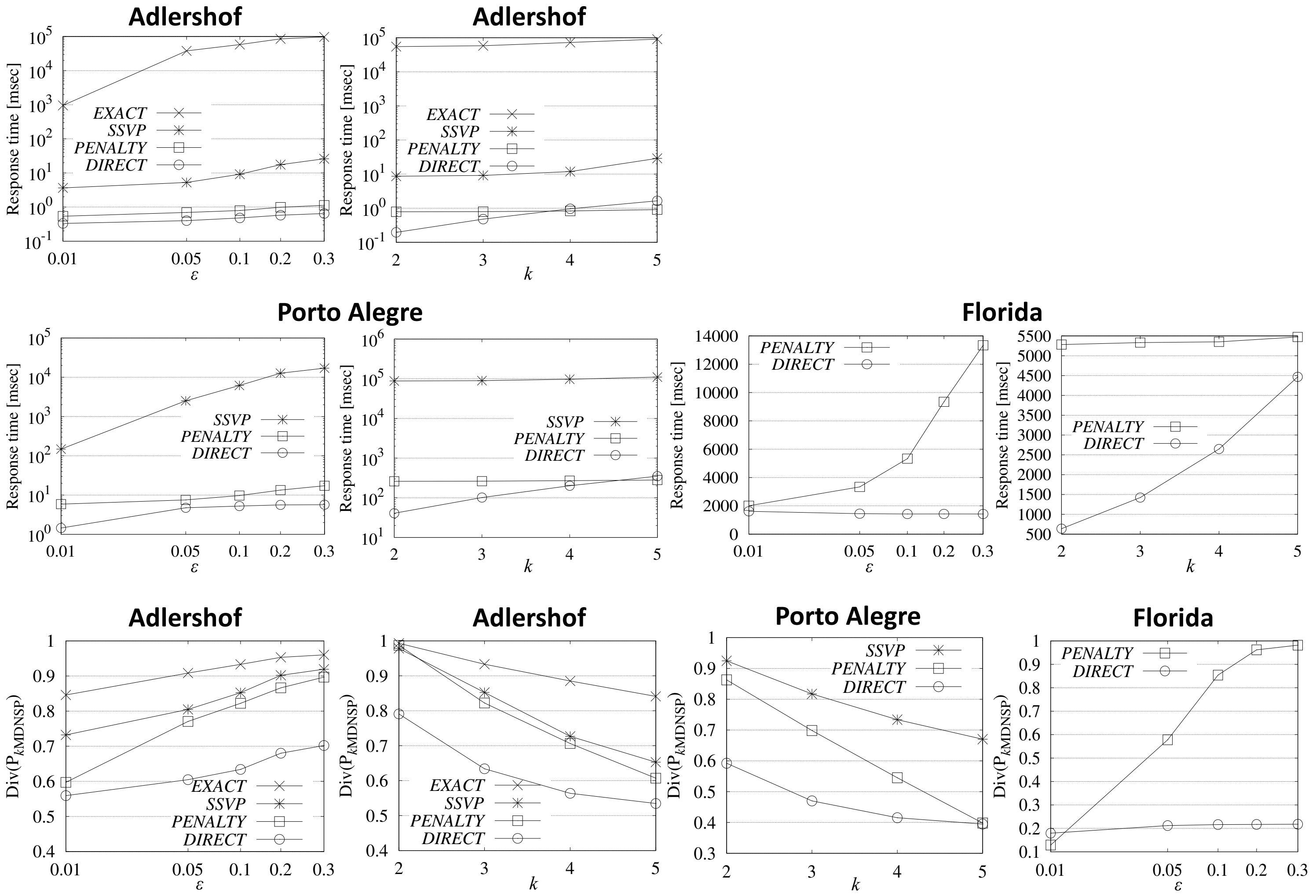
- Datasets & experiments
  - 5 real-world road networks
    - Different sizes and topologies: city-center, grid-based, ring-based, state-wide
    - Adlershof, Oldenburg, Porto Alegre, Milan, Chicago, Florida
  - Varied number of recommended paths *k* and near-shortest path factor *ε*
  - Measured response time and result diversity, counted computed near-shortest paths



- Key questions
  - Is computation of *k*MDNSP with *EXACT* practical?
  - How good can heuristic-based methods scale?
  - How is result quality affected?

Findings

- Is *EXACT* practical?
  - Only for toy-networks with up to some 100’s of nodes
  - Already orders of magnitude slower
- How good heuristic-based methods scale?
  - *SSVP* can handle networks with less then 100,000 of nodes
  - *PENALTY* and *DIRECT* can handle even state-wide networks
- How is result quality affected?
  - Best heuristic *SSVP*, followed by *PENALTY*



To sum up…

- Contributions
  - A novel instance of alternative routing problem
  - Recommend *k* near-shortest paths with the highest diversity
  - Exact and heuristic-based solutions

- Future work
  - Other evaluation approaches, e.g., flow algorithms
  - Alternative definitions of path diversity
  - Visualize and compare results

Thank you

Questions

?