Karlsruhe Institute of Technology

Algorithmic Aspects of Route Planning

escar – Embedded Security in Cars

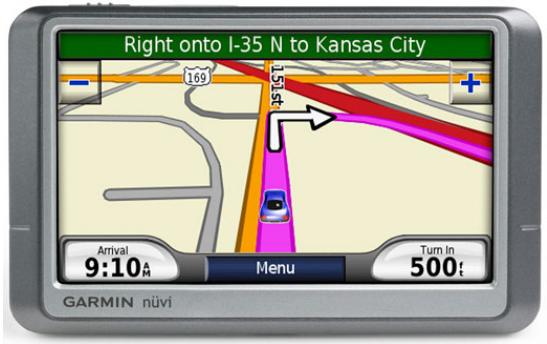
Dorothea Wagner | Hamburg, November 19, 2014

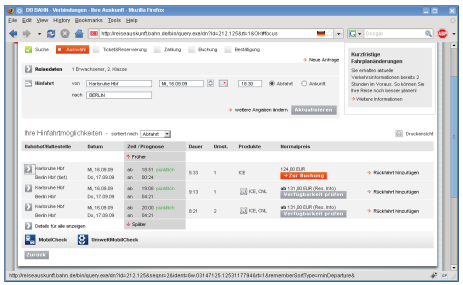
FACULTY FOR INFORMATICS · INSTITUTE FOR THEORETICAL INFORMATICS · CHAIR ALGORITHMICS

Motivation

Important application, e. g.,

- Navigation systems for cars
- Google Maps, Bing Maps, . . .
- Timetable information





Many commercial systems

- Use heuristic methods
- Consider “reasonable” part of the network
- Have no quality guarantees

Find methods for route planning in transportation networks with provably optimal solutions regarding the quality of the routes.

Problem

Request:

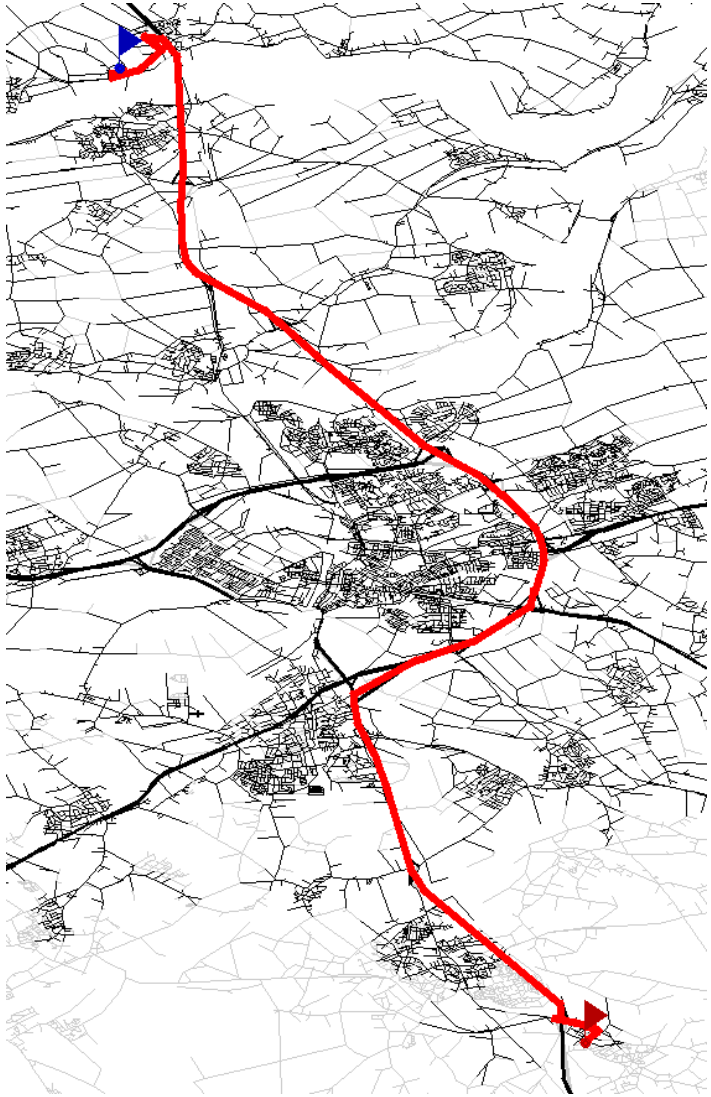
- Find the best connection in a transportation network

Idea:

- Network as graph $G = (V, E)$
- Edge weights are travel times
- Shortest paths in G equal quickest connections
- Classic problem (Dijkstra)

Problems:

- Transport networks are huge
- Dijkstra too slow ($>$ 1 second)



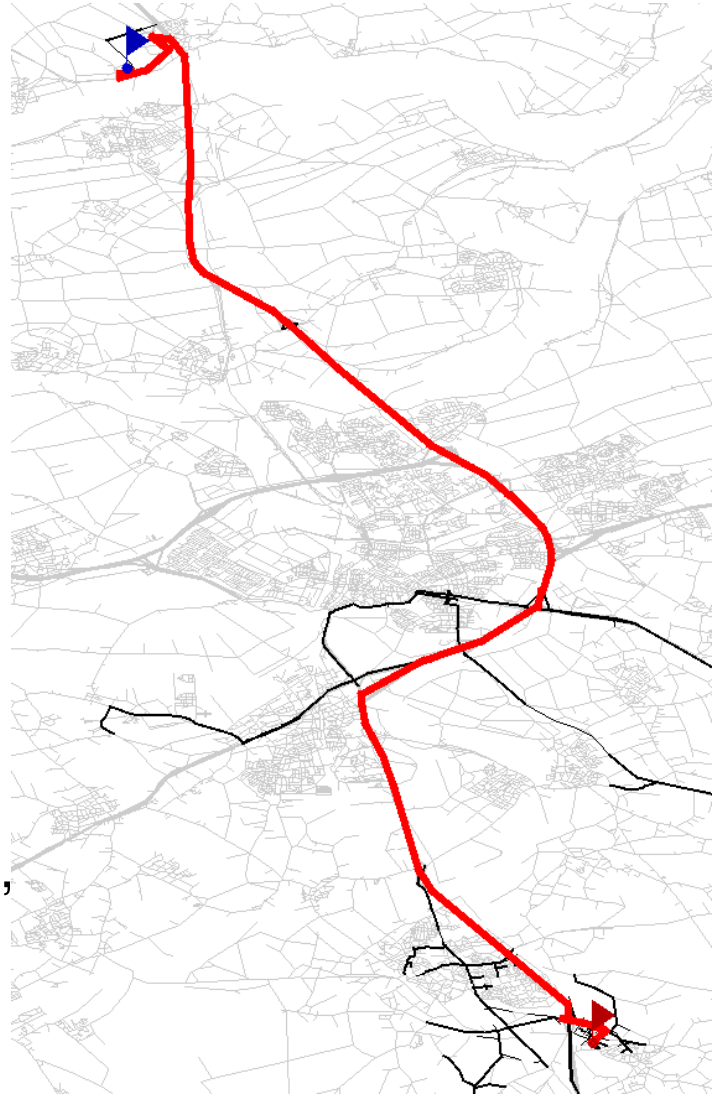
Speed-Up Techniques

Observations:

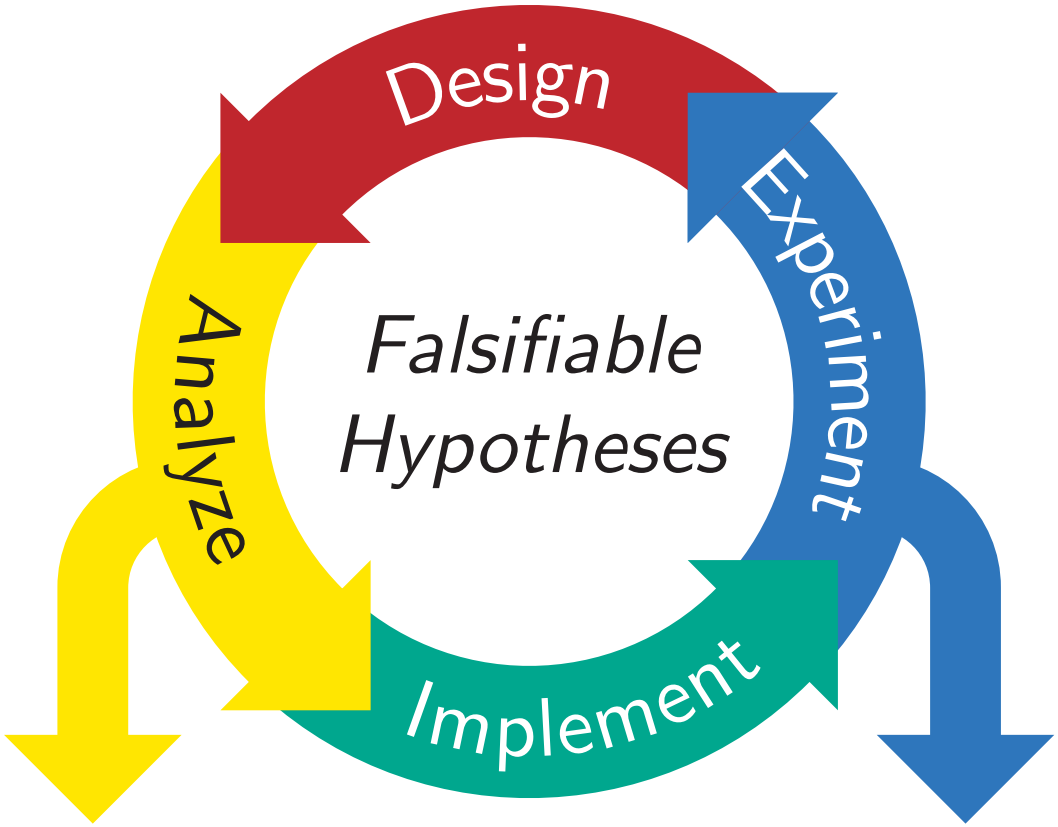
- Dijkstra visits all nodes closer than the target
- Unnecessary computations
- Many requests in a hardly changing network

Idea:

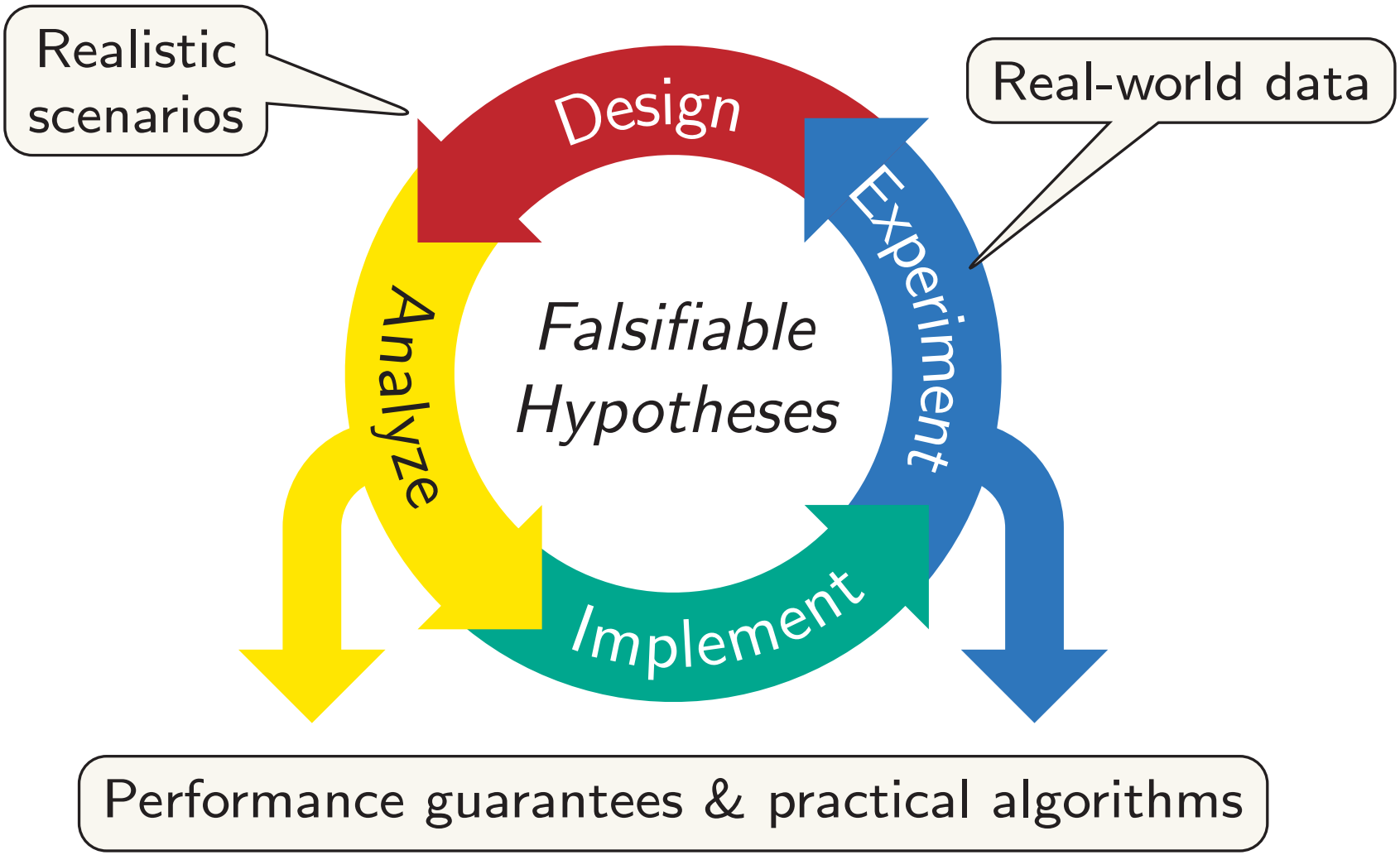
- Two-phase algorithm:
  - Offline: compute additional data during preprocessing
  - Online: speed-up query with this data
- 3 criteria: preprocessing time and space, speed-up over Dijkstra



Showpiece of Algorithm Engineering



Showpiece of Algorithm Engineering



History I

Phase I: Theory (1959 - 1999):

- Improve theoretical worst-case running time
- By introduction of better data structures
- Bidirectional search, A*-search (goal-directed)

Phase II: Speed-up techniques (1999 - 2005):

- Two approaches: goal-directed and hierarchical approach
- Improvement on this for several inputs

Phase III: Road networks (2005 - 2008):

- Focus on continent-sized road networks
- DIMACS challenge in 2006
- Speed-up factors in range of several millions over Dijkstra

History II

Phase IV: Towards more realistic scenarios (2008-2012):

- Time-dependency, multicriteria, alternative routes, ...
- Timetable information
- Back to theory: why do things work?

Now: New challenges (since 2012):

- Other metrics, e. g., energy consumption
- Customizability (supporting user-centric route planning)
- Multimodal

Exploiting Shortcuts

Observation:

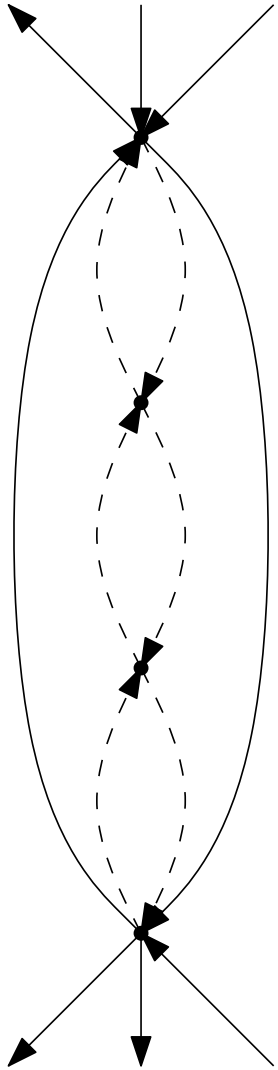
- Nodes with low degree are not important

Contract graph

- Iteratively remove such nodes
- Add shortcuts to preserve distances between non-removed nodes

Query:

- Bidirectional
- Prune edges heading less important nodes

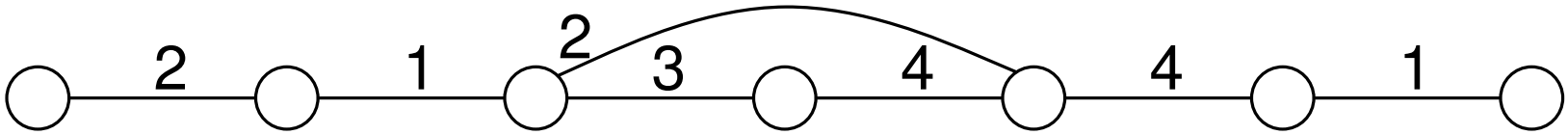


Contraction Hierarchies

Idea: Solely use contraction

Approach: Heuristically order nodes by “importance”

- Contract nodes in that order

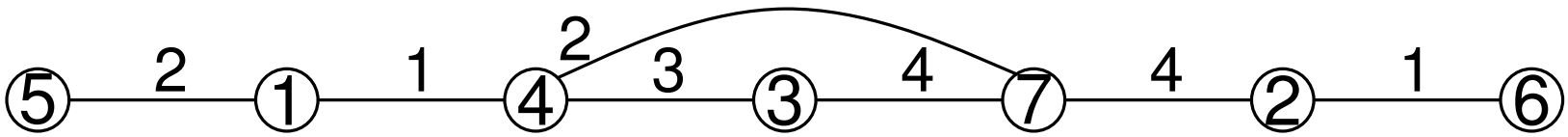


Contraction Hierarchies

Idea: Solely use contraction

Approach: Heuristically order nodes by “importance”

- Contract nodes in that order

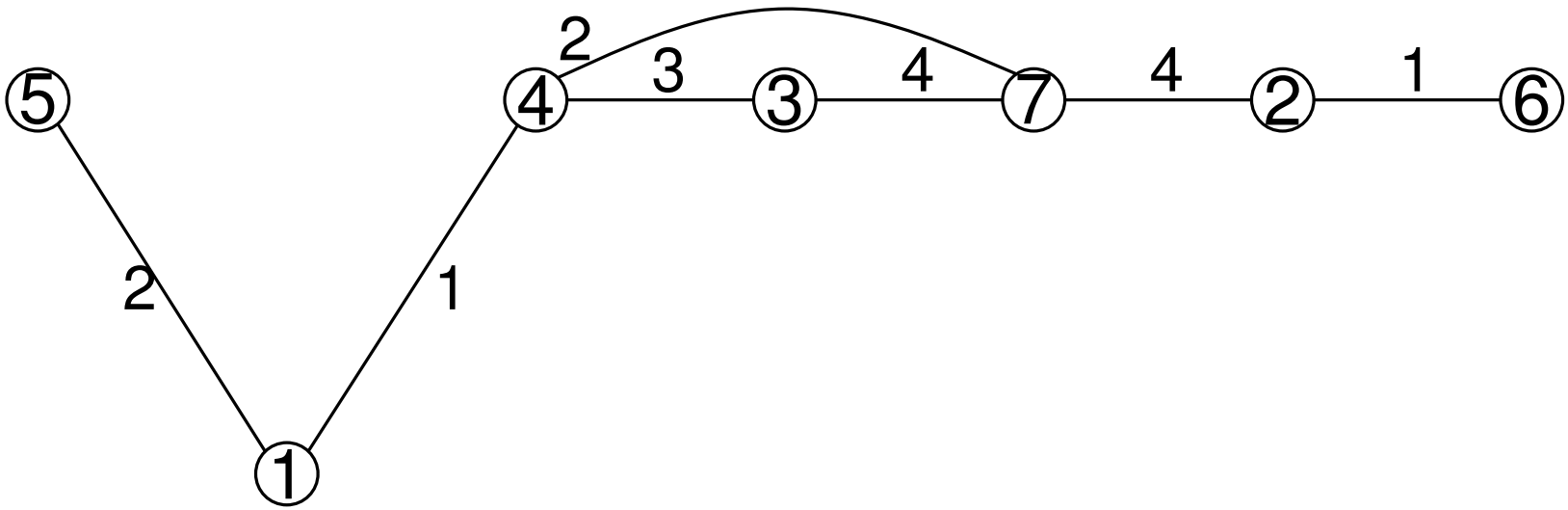


Contraction Hierarchies

Idea: Solely use contraction

Approach: Heuristically order nodes by “importance”

- Contract nodes in that order

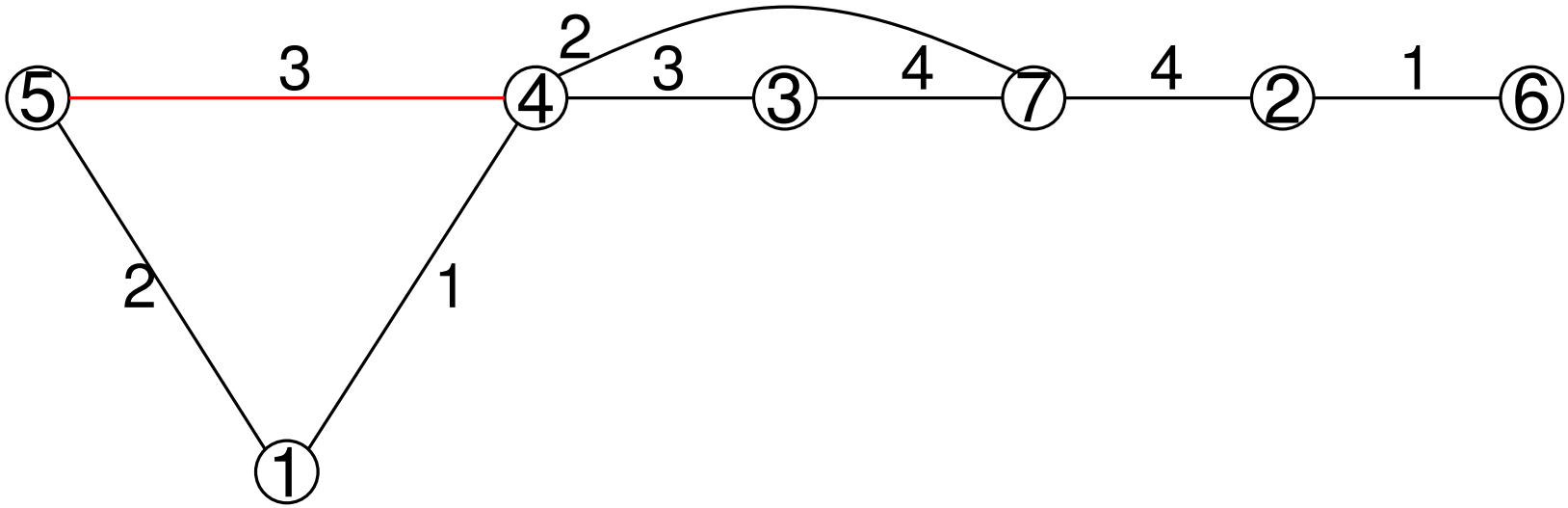


Contraction Hierarchies

Idea: Solely use contraction

Approach: Heuristically order nodes by “importance”

- Contract nodes in that order

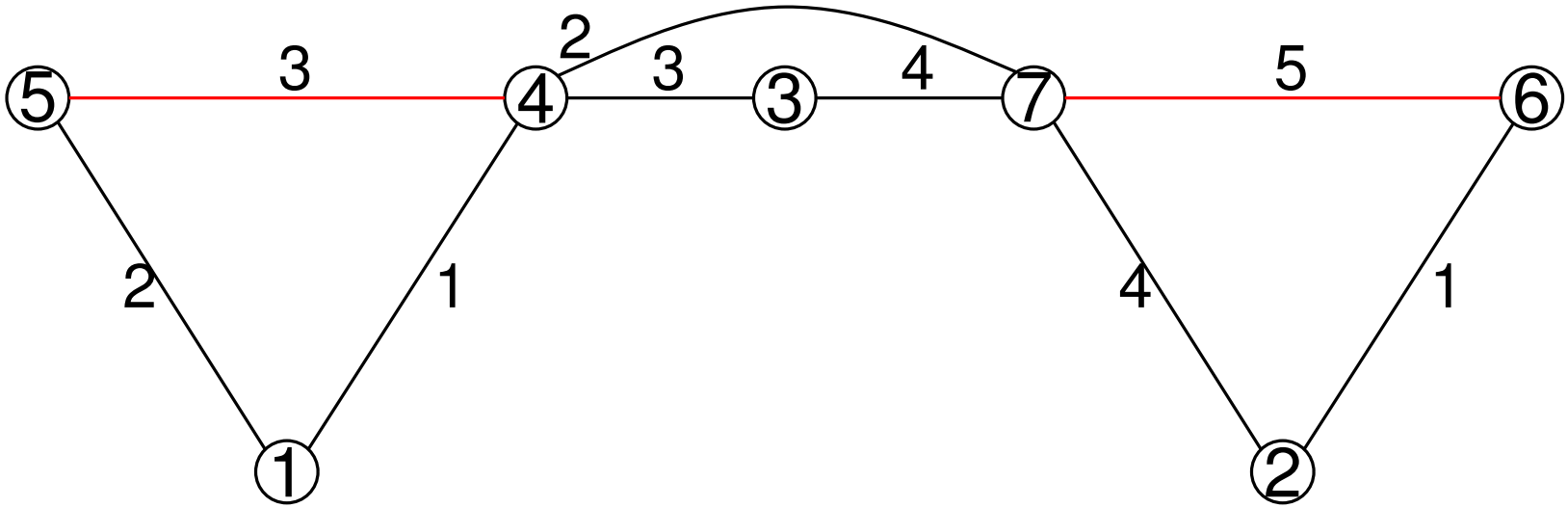


Contraction Hierarchies

Idea: Solely use contraction

Approach: Heuristically order nodes by “importance”

- Contract nodes in that order

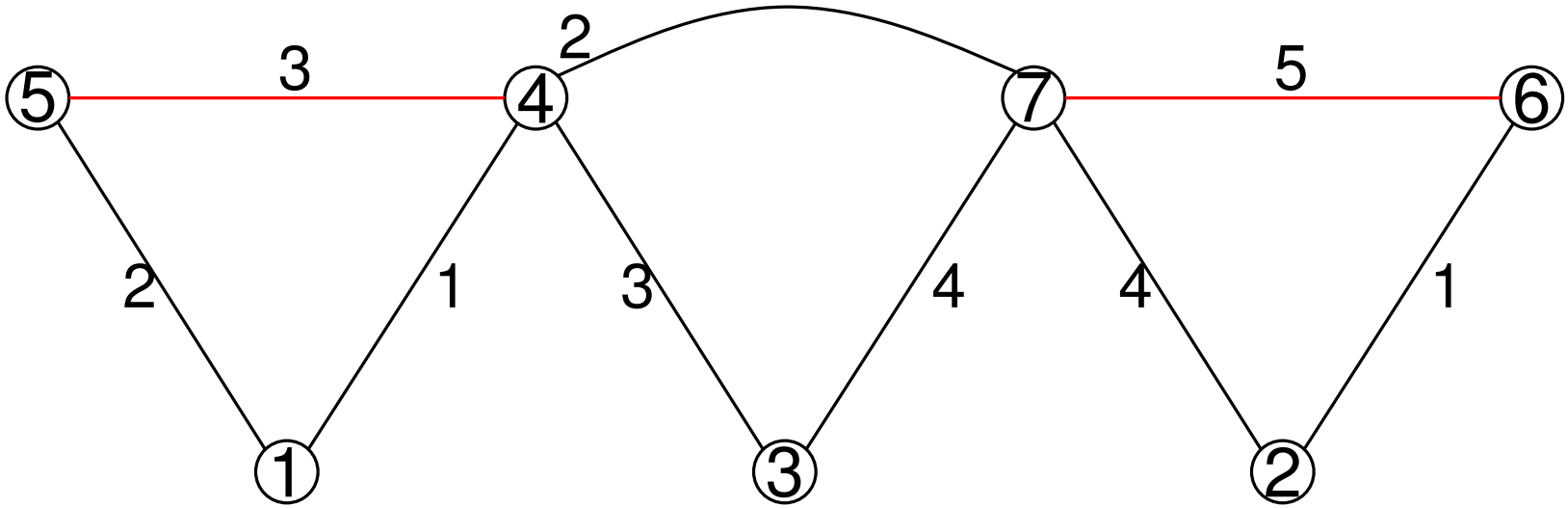


Contraction Hierarchies

Idea: Solely use contraction

Approach: Heuristically order nodes by “importance”

- Contract nodes in that order

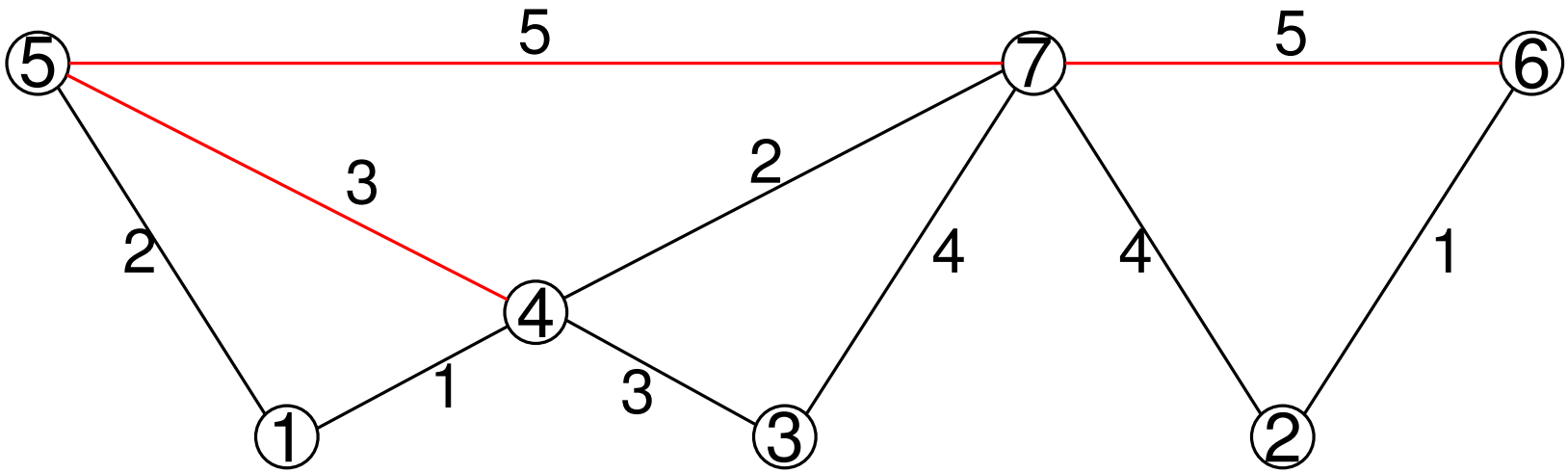


Contraction Hierarchies

Idea: Solely use contraction

Approach: Heuristically order nodes by “importance”

- Contract nodes in that order

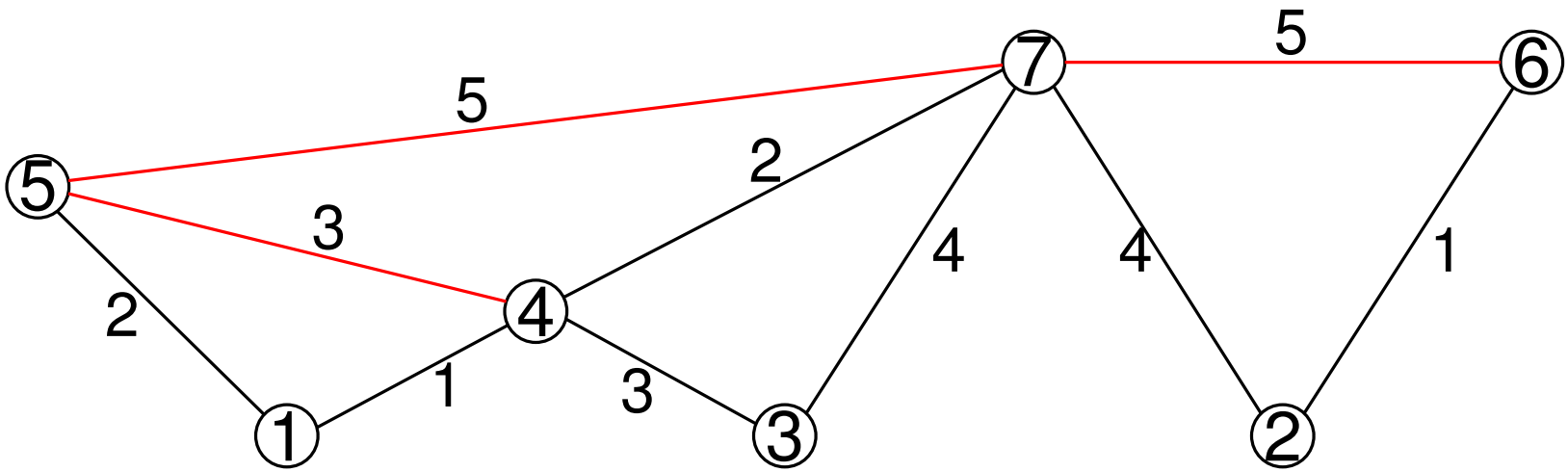


Contraction Hierarchies

Idea: Solely use contraction

Approach: Heuristically order nodes by “importance”

- Contract nodes in that order

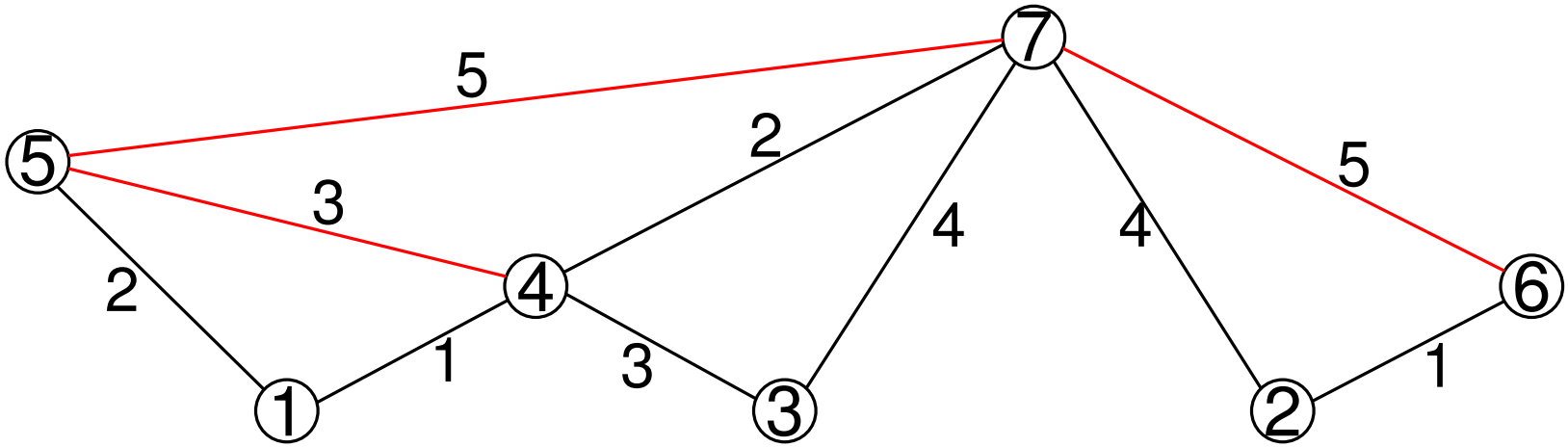


Contraction Hierarchies

Idea: Solely use contraction

Approach: Heuristically order nodes by “importance”

- Contract nodes in that order

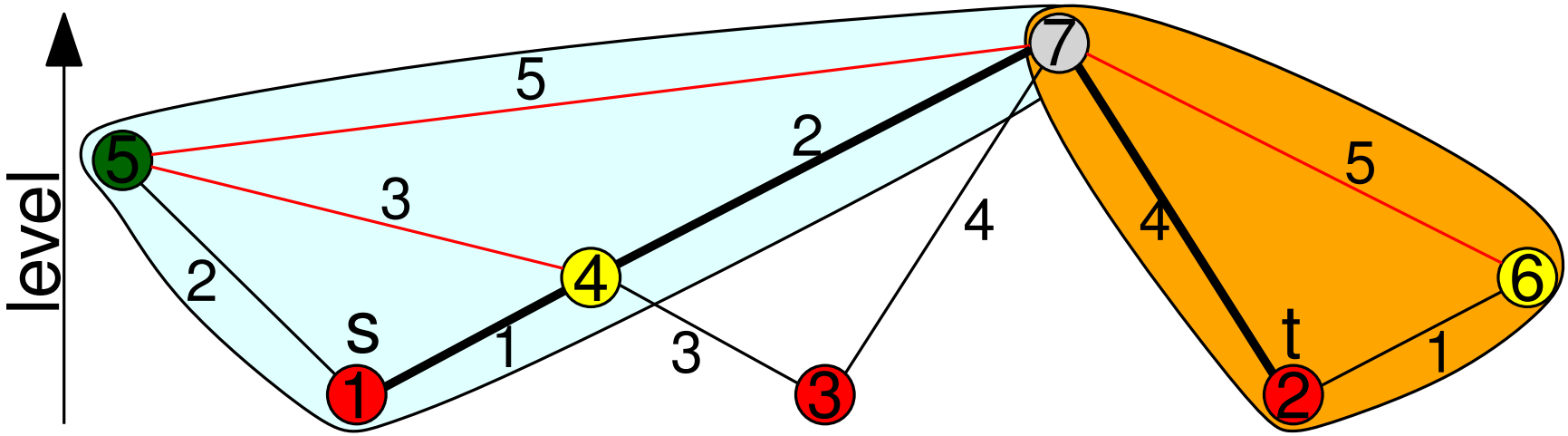


Contraction Hierarchies

Idea: Solely use contraction

Approach: Heuristically order nodes by “importance”

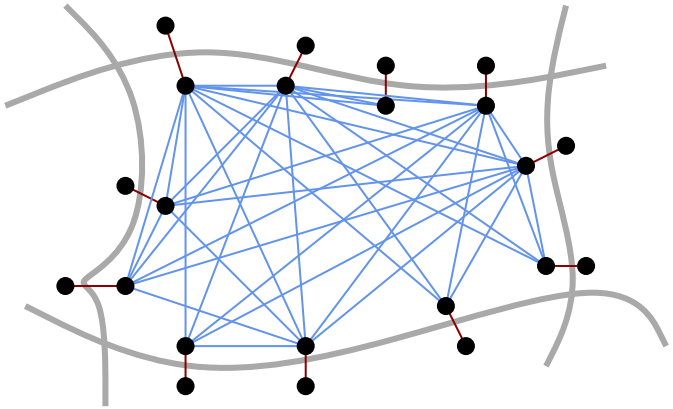
- Contract nodes in that order



- query only looks at edges to more important nodes

(Multi-Level) Overlays

Observation: Many (long-distance) paths share large subpaths
Idea: Precompute partial solutions

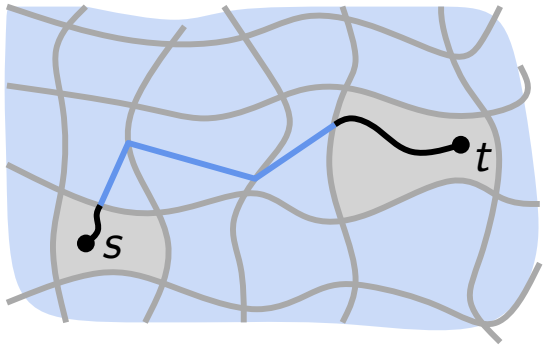


Overlay graph:

- Partition graph
- Compute shortcuts between boundary nodes:
  - Conserve distances

Queries:

- Multi-level Dijkstra variant
- Ignore edges towards less important nodes

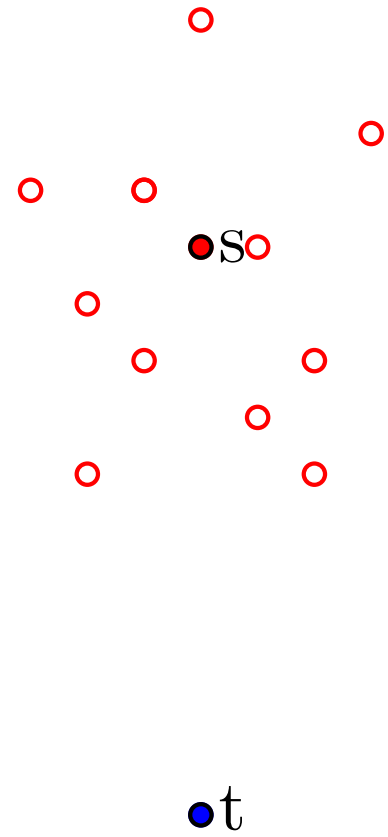


Analogous: Hierarchies with several levels of increasing node importance

Hub Labeling

Preprocessing:

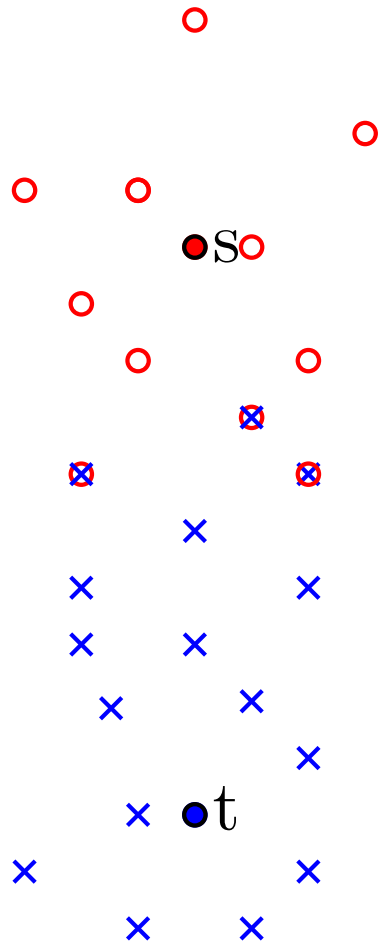
- For each node u, compute label $L(u)$
  - A set of hub nodes v and their distance $\text{dist}(u, v)$ to u



Hub Labeling

Preprocessing:

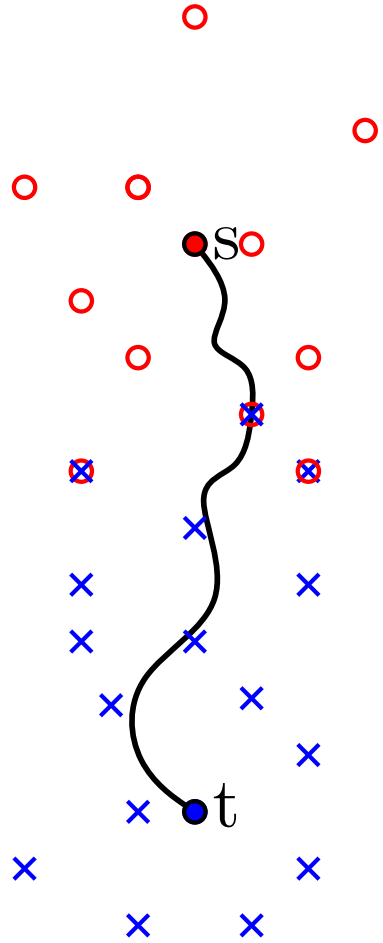
- For each node u, compute label $L(u)$
  - A set of hub nodes v and their distance $\text{dist}(u, v)$ to u
- Labels must fulfill cover property:
  for every s, t-pair, the shortest path goes through the intersection of $L(s) \cap L(t)$



Hub Labeling

Preprocessing:

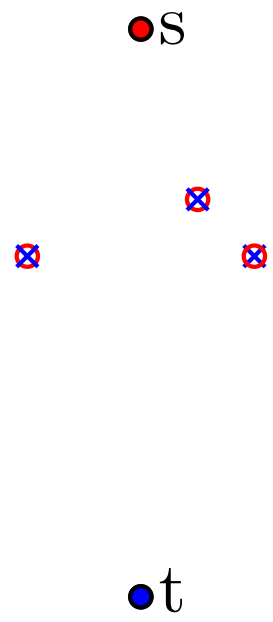
- For each node u, compute label $L(u)$
  - A set of hub nodes v and their distance $\text{dist}(u, v)$ to u
- Labels must fulfill cover property:
  for every s, t-pair, the shortest path goes through the intersection of $L(s) \cap L(t)$



Hub Labeling

Preprocessing:

- For each node u, compute label $L(u)$
  - A set of hub nodes v and their distance $\text{dist}(u, v)$ to u
- Labels must fulfill cover property:
  for every s, t-pair, the shortest path goes through the intersection of $L(s) \cap L(t)$



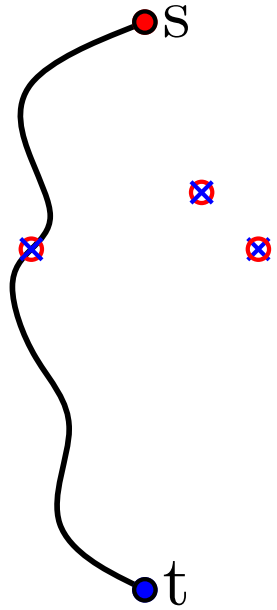
***s*–*t* query:**

- Find node $v \in L(s) \cap L(t)$ ...

Hub Labeling

Preprocessing:

- For each node u, compute label $L(u)$
  - A set of hub nodes v and their distance $\text{dist}(u, v)$ to u
- Labels must fulfill cover property:
  for every s, t-pair, the shortest path goes through the intersection of $L(s) \cap L(t)$



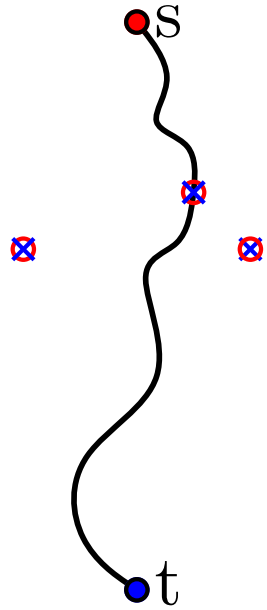
s–t query:

- Find node $v \in L(s) \cap L(t) \ldots$
- $\ldots$ that minimizes $\text{dist}(s, v) + \text{dist}(v, t)$

Hub Labeling

Preprocessing:

- For each node u, compute label $L(u)$
  - A set of hub nodes v and their distance $\text{dist}(u, v)$ to u
- Labels must fulfill cover property:
  for every s, t-pair, the shortest path goes through the intersection of $L(s) \cap L(t)$



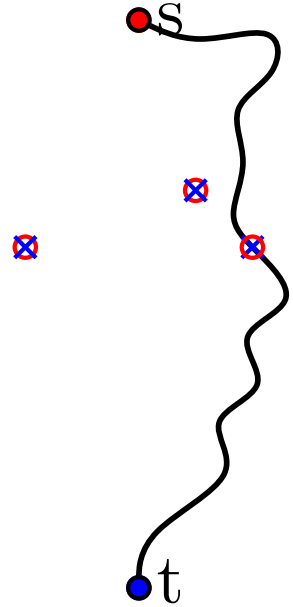
s–t query:

- Find node $v \in L(s) \cap L(t) \ldots$
- $\ldots$ that minimizes $\text{dist}(s, v) + \text{dist}(v, t)$

Hub Labeling

Preprocessing:

- For each node u, compute label $L(u)$
  - A set of hub nodes v and their distance $\text{dist}(u, v)$ to u
- Labels must fulfill cover property:
  for every s, t-pair, the shortest path goes through the intersection of $L(s) \cap L(t)$



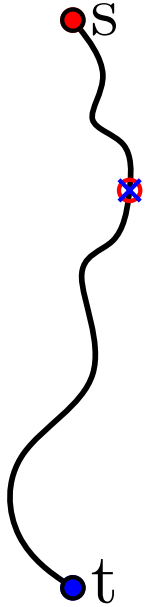
s–t query:

- Find node $v \in L(s) \cap L(t) \ldots$
- $\ldots$ that minimizes $\text{dist}(s, v) + \text{dist}(v, t)$

Hub Labeling

Preprocessing:

- For each node u, compute label $L(u)$
  - A set of hub nodes v and their distance $\text{dist}(u, v)$ to u
- Labels must fulfill cover property:
  for every s, t-pair, the shortest path goes through the intersection of $L(s) \cap L(t)$



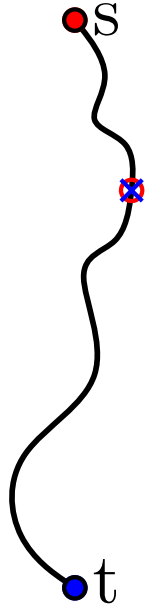
s–t query:

- Find node $v \in L(s) \cap L(t) \ldots$
- $\ldots$ that minimizes $\text{dist}(s, v) + \text{dist}(v, t)$

Hub Labeling

Preprocessing:

- For each node u, compute label $L(u)$
  - A set of hub nodes v and their distance $\text{dist}(u, v)$ to u
- Labels must fulfill cover property:
  for every s, t-pair, the shortest path goes through the intersection of $L(s) \cap L(t)$



s–t query:

- Find node $v \in L(s) \cap L(t)$ ...
- ... that minimizes $\text{dist}(s, v) + \text{dist}(v, t)$

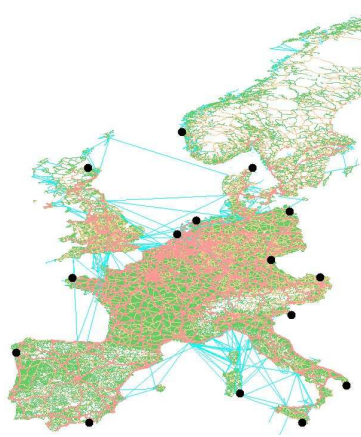
Observations:

- Very simple query (can even be implemented in SQL)
- Query performance depends only on label sizes
- The “magic” lies in computing a small labeling efficiently

Experimental Evaluation

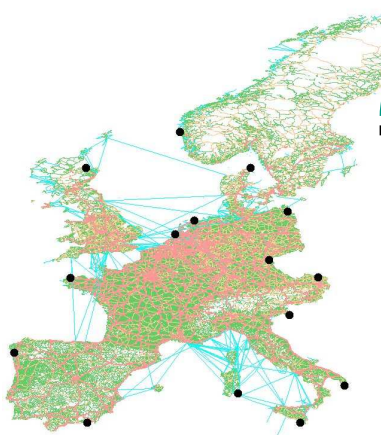
Input: Road network of Europe

- Approx. 18M nodes
- Approx. 42M edges



	Preprocessing		Query	
Algorithm	Time [h:m]	Space [GiB]	Time [μs]	Speedup
Dijkstra [Dij59]	—	—	2 550 000	—
ALT [GH05, GW05]	0:42	2.2	24 521	104
CRP [DGPW11]	1:00	0.5	1 650	1 545
Arc-Flags [Lau04]	0:20	0.3	408	6 250
CH [GSSD08]	0:05	0.2	110	23 181
TNR [ALS13]	0:20	2.1	1.25	2 040 000
HL [ADGW12]	0:37	18.8	0.56	4 553 571

Experimental Evaluation



Input: Road network of Europe

- Approx. 18M nodes
- Approx. 42M edges

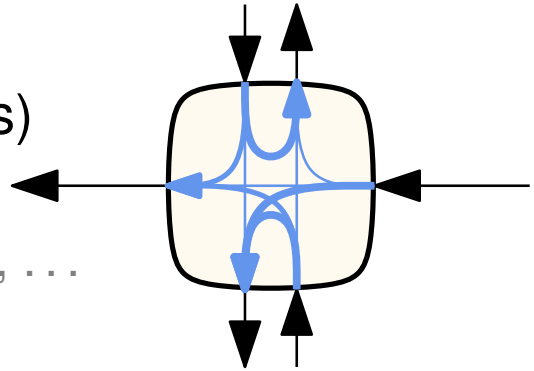
	Preprocessing		Query	
Algorithm	Time [h:m]	Space [GiB]	Time [μs]	Speedup
Dijkstra [Dij59]	—	—	2 550 000	—
ALT [GH05, GW05]	0:42	2.2	24 521	104
CRP [DGPW11]	1:00	0.5	1 650	1 545
Arc-Flags [Lau04]	0:20	0.3	408	6 250
CH [GSSD08]	0:05	0.2	110	23 181
TNR [ALS13]	0:20	2.1	1.25	2 040 000
HL [ADGW12]	0:37	18.8	0.56	4 553 571

In use at Bing, Google, Tomtom, . . .

New Challenges

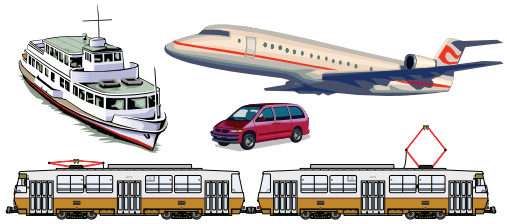
More realistic metrics:

- Turn costs, electro mobility
- Points of interests (nearest POIs, shortest via-POIs)
- User customizable metrics
  e.g., height restrictions, avoid freeways, eco-friendliness, . . .
- Fast customization time per metric
- Very small space overhead



Multimodal networks:

- Change the type of transportation during the journey
- Allow only “reasonable” transfers
- Several constraints to the shortest path
- Multicriteria



Route Planning for Electric Vehicles

Electric vehicles:

- Future means of transportation
- Run on regenerative energy sources

But:

- Restricted battery capacity
- Long recharging times
- “Range anxiety”

⇒ Consider energy consumption in route planning applications

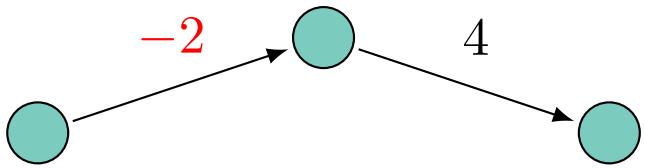
Task: Given start and destination in a road network, find the route that minimizes energy consumption.

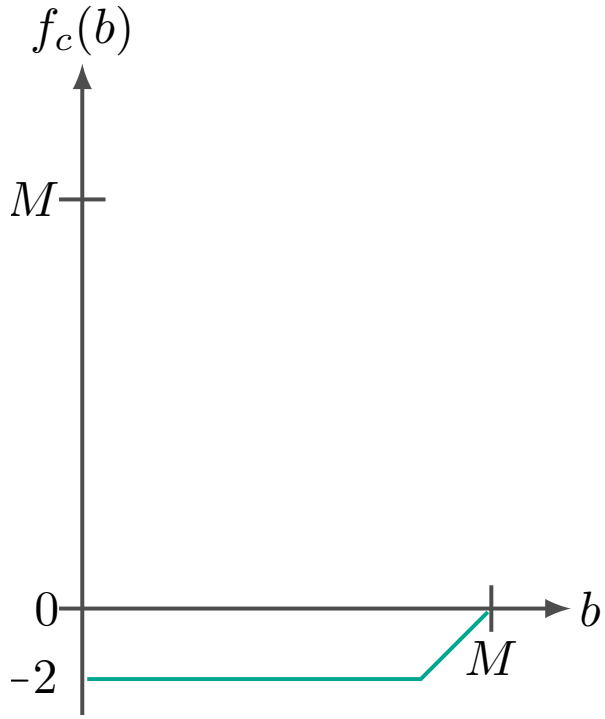
Energy-Optimal Routes

Challenges:

- Negative edge weights (recuperation)
- Battery constraints (no over-, undercharging)

Energy consumption depends on battery state-of-charge (at the start):



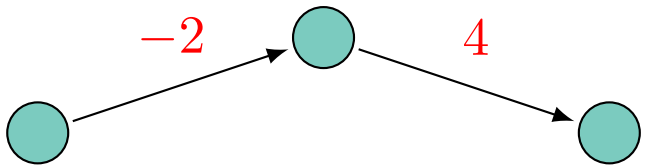


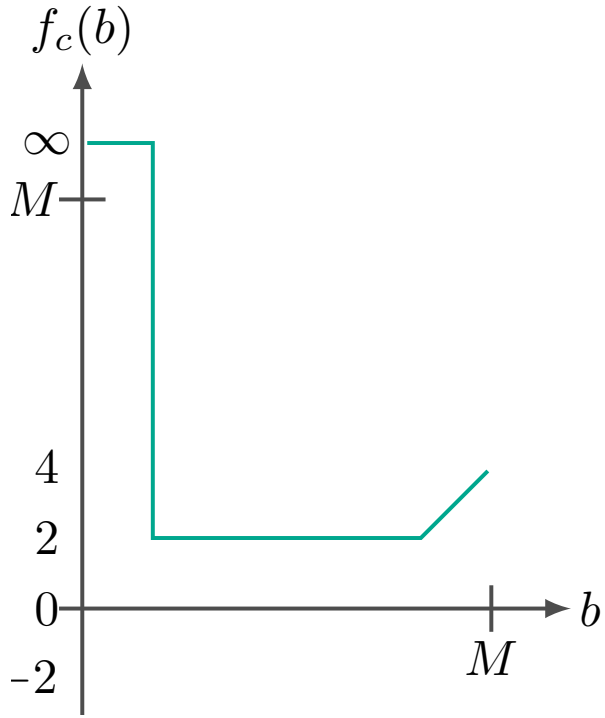
Energy-Optimal Routes

Challenges:

- Negative edge weights (recuperation)
- Battery constraints (no over-, undercharging)

Energy consumption depends on battery state-of-charge (at the start):



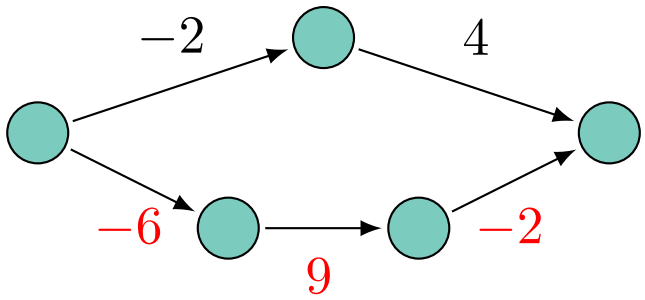


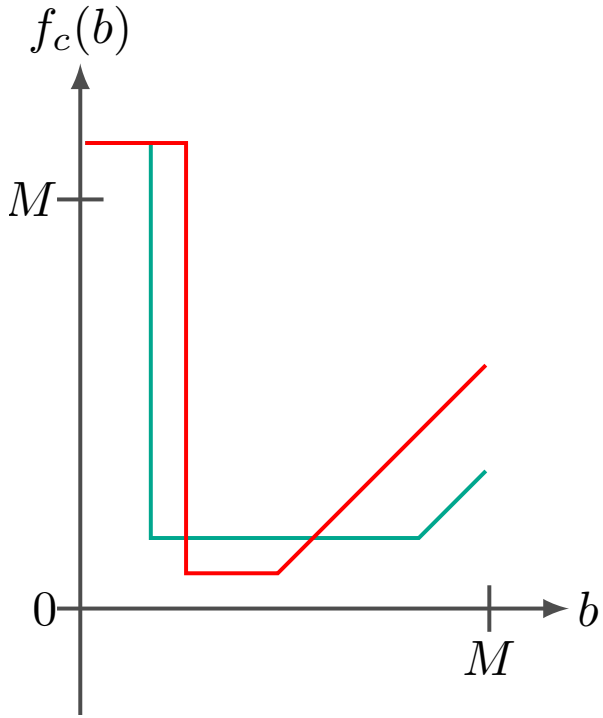
Energy-Optimal Routes

Challenges:

- Negative edge weights (recuperation)
- Battery constraints (no over-, undercharging)

Energy consumption depends on battery state-of-charge (at the start):



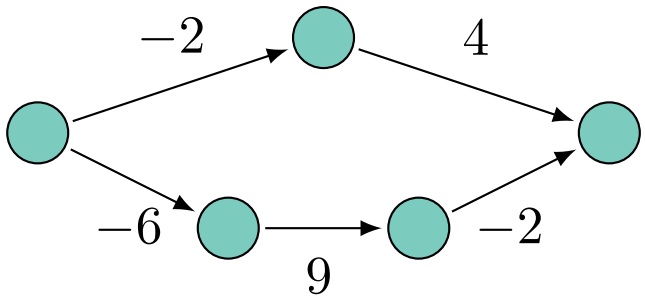


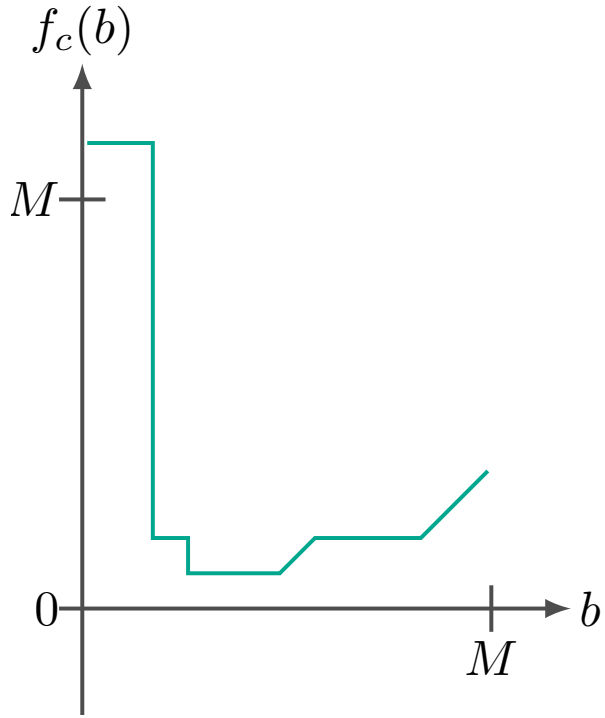
Energy-Optimal Routes

Challenges:

- Negative edge weights (recuperation)
- Battery constraints (no over-, undercharging)

Energy consumption depends on battery state-of-charge (at the start):





Energy-Optimal Routes

Requirements for speedup techniques:

- Shortcuts are functions, not scalar values
- User-dependent consumption profiles ($\Rightarrow$ custom metrics)

Experiments:

- Energy-optimal paths: 63 % extra time
- Fastest paths: 62 % extra energy

$\Rightarrow$ Energy-optimal routes: follow slow roads

Current and future work:

- Consider trade-off between travel time and energy consumption
  - Energy can be saved driving below speed limit
- Integration of charging stations into route planning
  - Different types of charging stations (super chargers, switching stations)

Custom Metrics

Problem

- Preprocessing is metric-dependent
- State-of-the-art algorithms tailored to travel time
  heavily exploit ‘hierarchy’ of road categories

Naive solution

- Compute preprocessing for each metric, e. g.
  - Distance
  - Pedestrian
  - Travel time, but don’t use toll roads
  - Travel time, avoid left turns, height restrictions, avoid tolls, . . .
- Preprocessing and query time increase significantly
- Higher space overhead

Customizable Contraction Hierachies

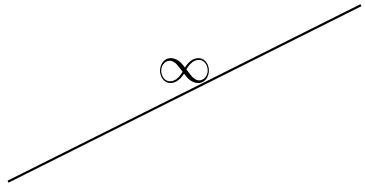
Idea:

- CH topology is the same regardless of metric
- Quickly introduce new metric

Customizable Contraction Hierachies

Idea:

- CH topology is the same regardless of metric
- Quickly introduce new metric

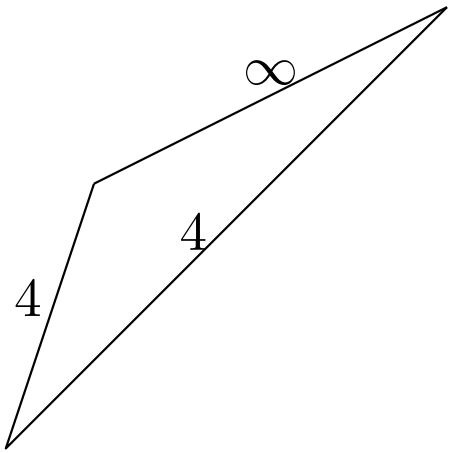


an edge in the CH

Customizable Contraction Hierachies

Idea:

- CH topology is the same regardless of metric
- Quickly introduce new metric

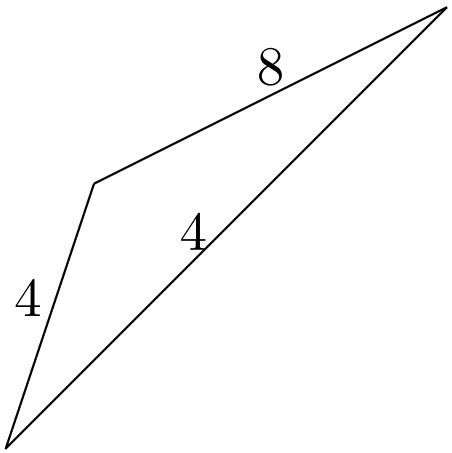


establish lower triangle inequality

Customizable Contraction Hierachies

Idea:

- CH topology is the same regardless of metric
- Quickly introduce new metric

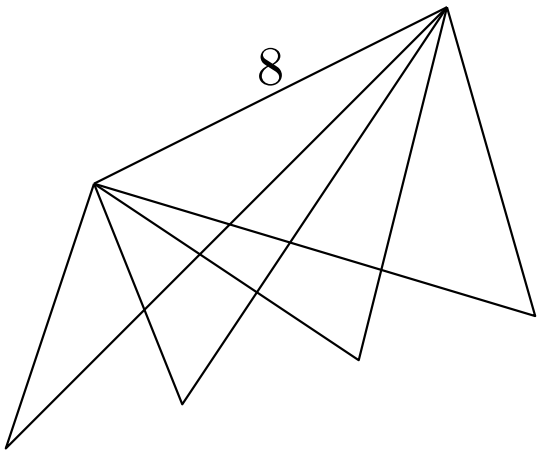


establish lower triangle inequality

Customizable Contraction Hierachies

Idea:

- CH topology is the same regardless of metric
- Quickly introduce new metric



do this for all lower triangles

Vision: Navi for the World

Worldwide network composed of car, rail, flight, . . .

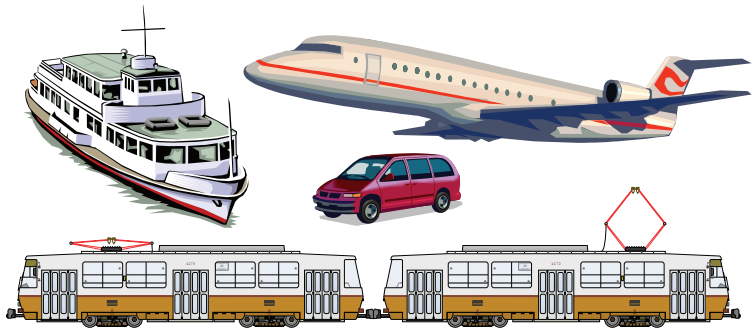
Multimodal Routing

Up to now, research mostly on uni-modal routing

- Restricted to one transportation network
- Time-independent and time-dependent (separately)

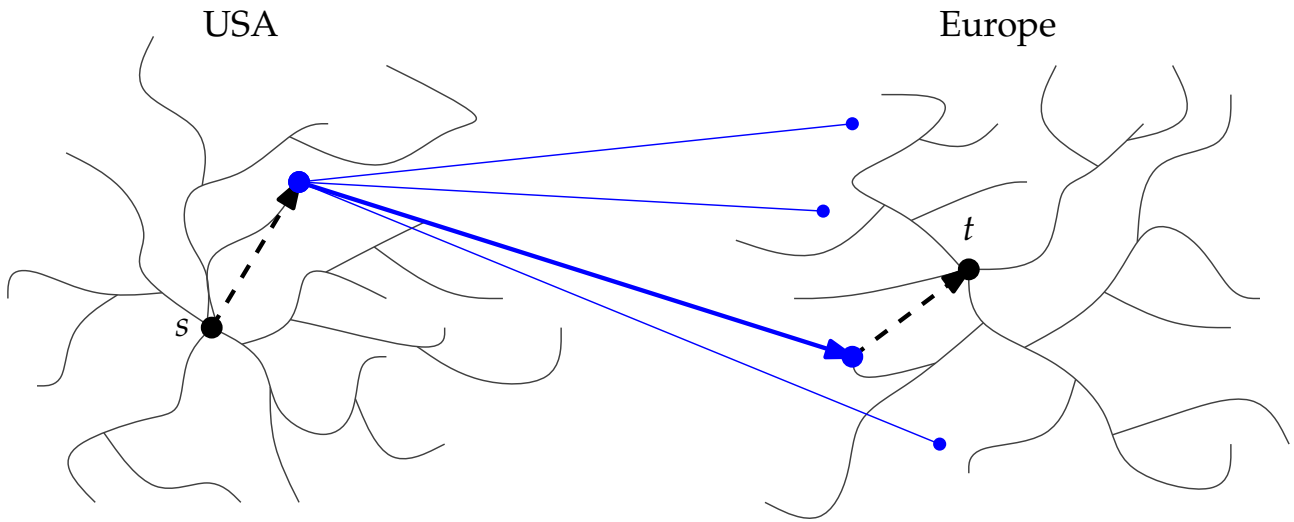
What we really want is planning a journey by

- Choosing source and destination
- Desired means of transportation (car, train, flight, . . .)
- . . . in a mixed network



Access-Node Routing: Idea

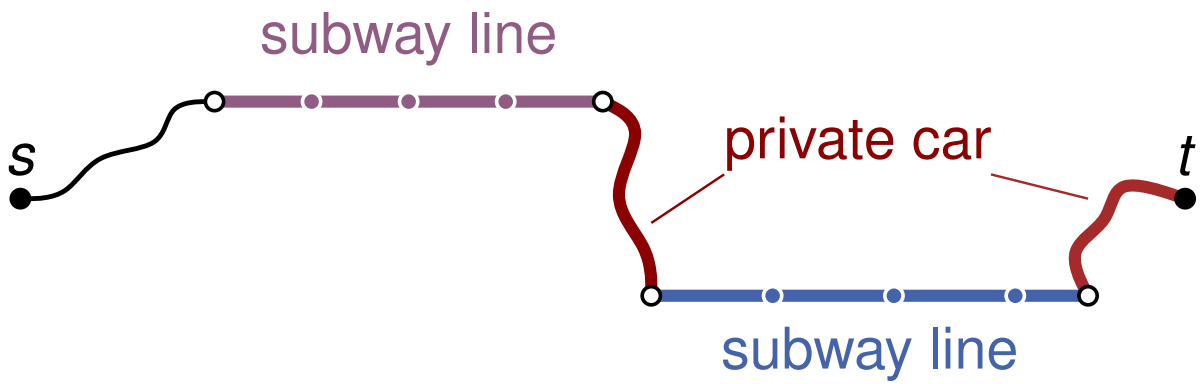
Assumption: Road network only used in the beginning and end
Observation: Number of “relevant” entry points in the public transportation network is small



Idea: Compute for each road node its access-nodes and their distances

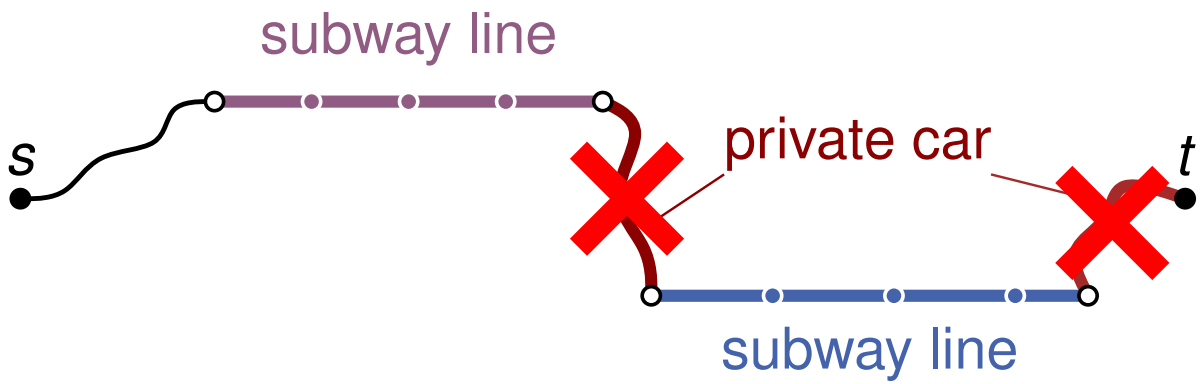
Multiple Transportation Modes

Problem: Unrestricted journeys allow arbitrary transfers



Multiple Transportation Modes

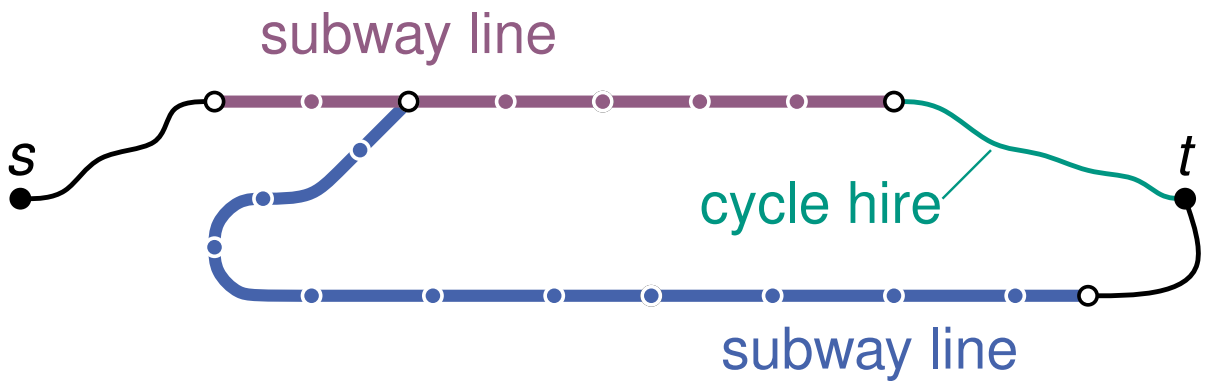
Problem: Unrestricted journeys allow arbitrary transfers



- Not all sequences of transportation modes are reasonable

Multiple Transportation Modes

Problem: Unrestricted journeys allow arbitrary transfers

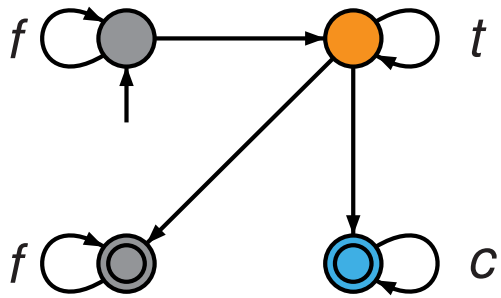


- Not all sequences of transportation modes are reasonable
- Preferred mode of transport varies between users

Solution

„Label Constrained Shortest Path Problem“ (LCSPP)

- Define alphabet of transportation mode
- Finite-state automaton describes sequences of verhicles
- Every path must fulfill the requirements imposed by the automaton



Algorithms for LCSPP

- Dijkstra on the product graph with the automaton works but is slow
- Can be combined with speed-up techniques
- Automaton as input for the query: user-constrained shortest paths

Solution?

Problems of LCSPP

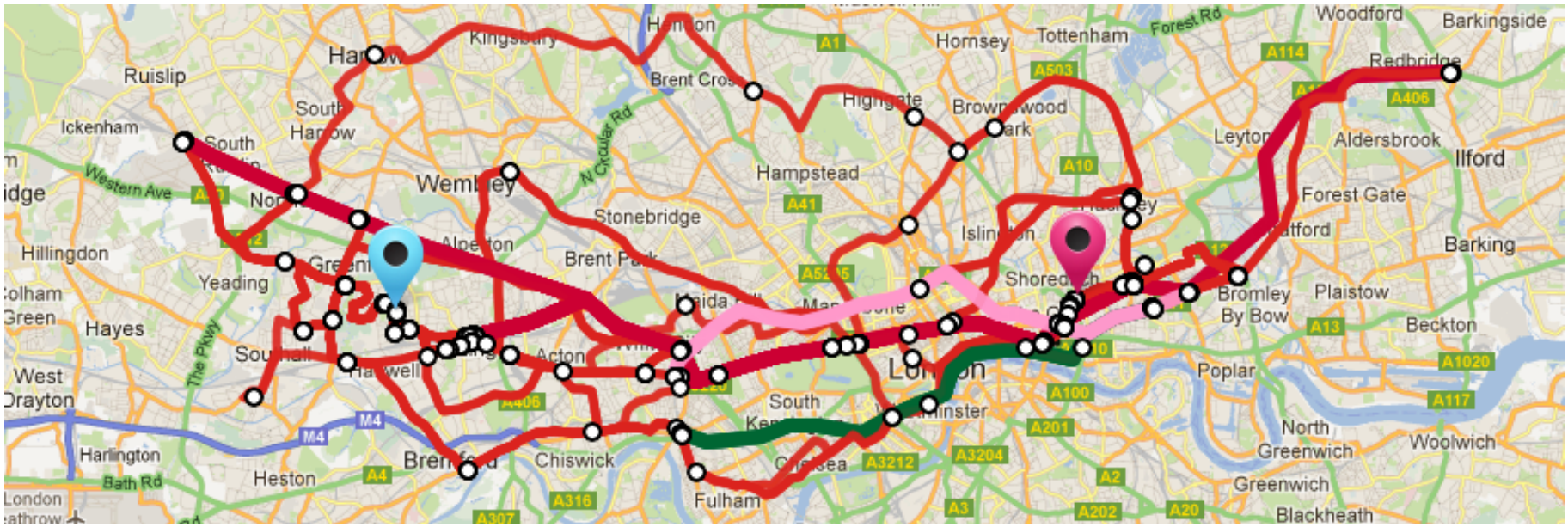
- Restrictions must be known in advance
- User might not know them
- Only a single (best?) journey is computed (no alternatives)

Goal: Compute a *useful set* of multimodal journeys

Multicriteria Multimodal Routing

Idea: Compute multicriteria, multimodal Pareto sets

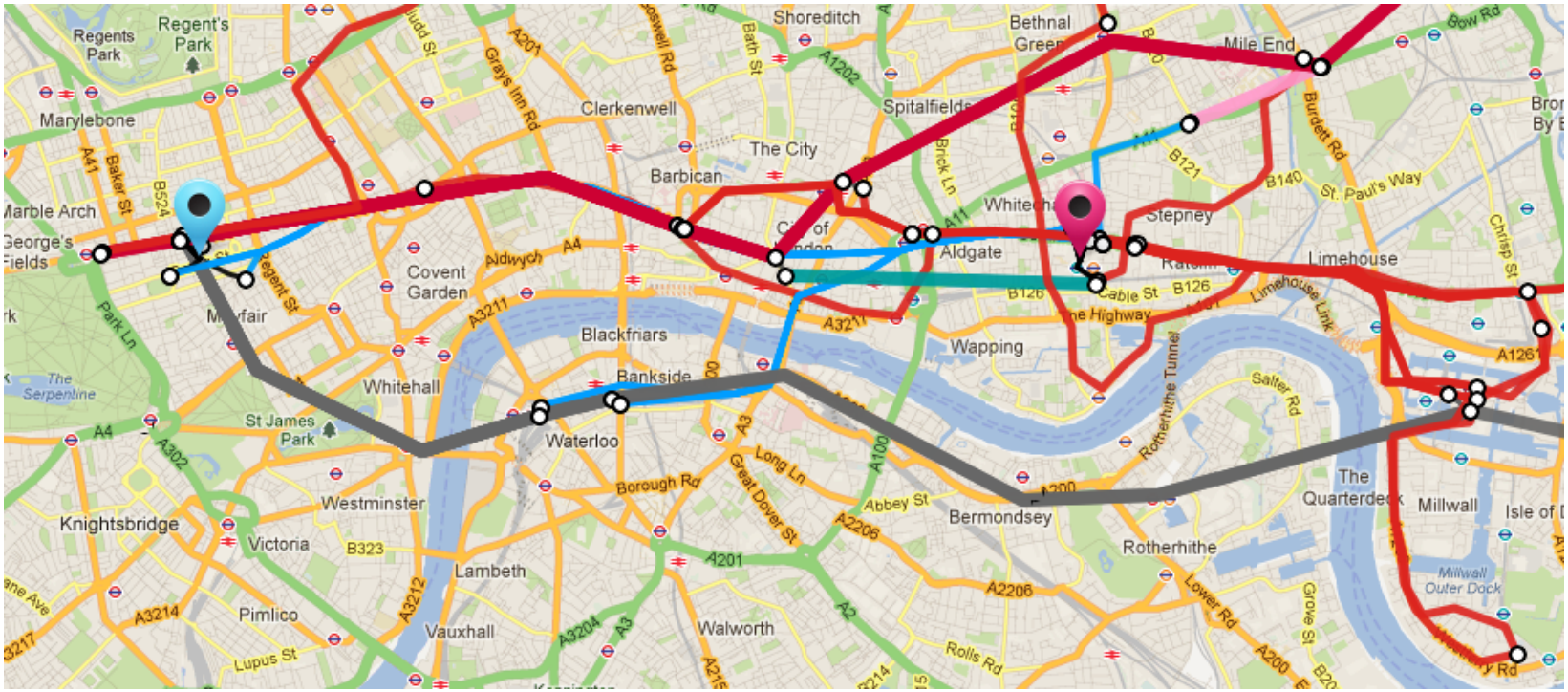
- Optimize arrival time plus
- Various (per mode of transport) „convenience criteria“
  for example # transfers (trains), walking time, taxi costs, etc.



criteria: arrival time, # transfers, walking time **69 journeys**.

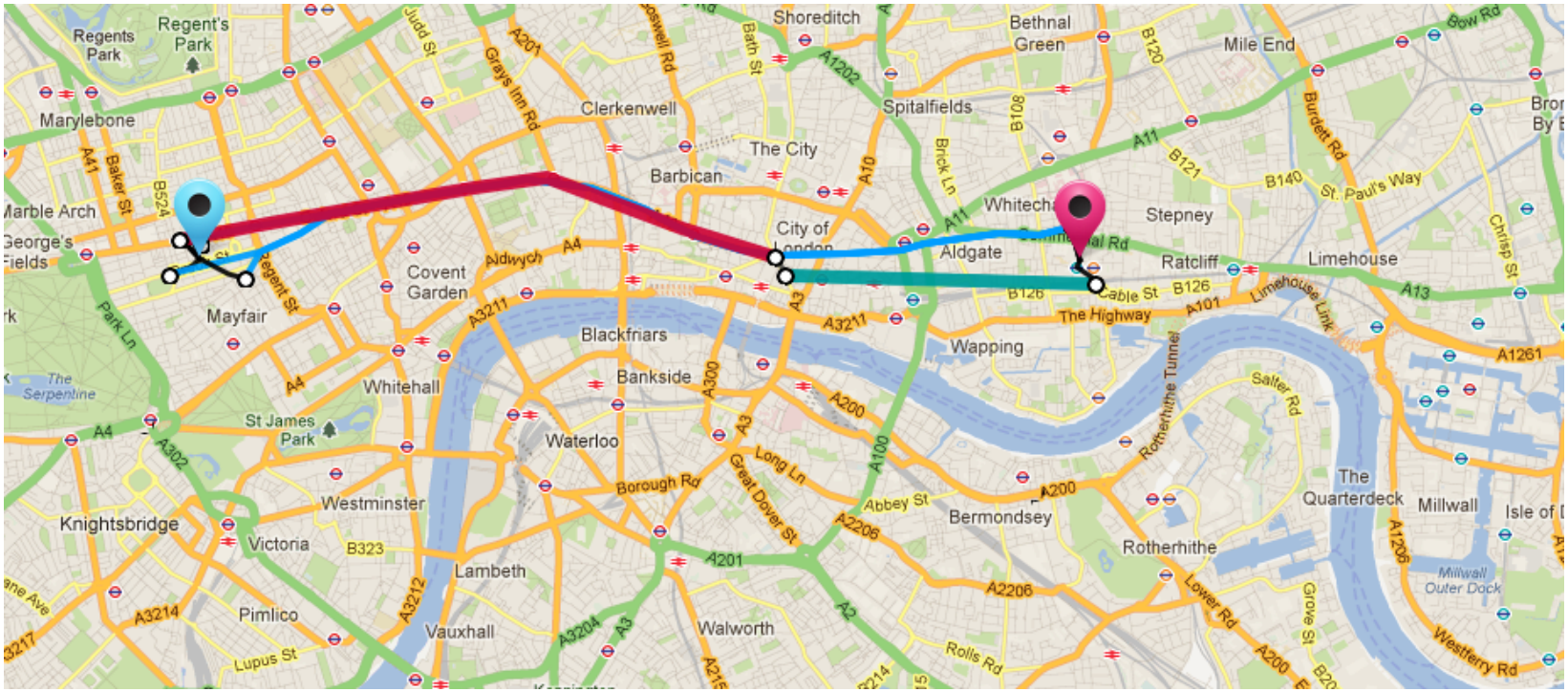
Known problem: Pareto set sizes explode in the number of criteria

Relevant Journeys



- 10 min of walking to arrival 10 sec earlier?
- 1 hour of bus drive to walk 10 sec less?
- Rate the journeys using fuzzy logic
- Journeys with a higher rating are more relevant

Relevant Journeys



the three top-rated journeys.

- 10 min of walking to arrival 10 sec earlier?
- 1 hour of bus drive to walk 10 sec less?
- Rate the journeys using fuzzy logic
- Journeys with a higher rating are more relevant

Reducing the Amount of Work

Problem: Queries are slow (> 1 s)

Many irrelevant journeys $\Rightarrow$ can we avoid computing them?

Filter already during the algorithm

- MCR-hf: fuzzy filter
- MCR-hb: Pareto filter, but discrete criteria

Restricted walking (arbitrary heuristic)

- MCR-t*x*-r*y*: max *x* minutes of walking between vehicles and max. *y* at source/target

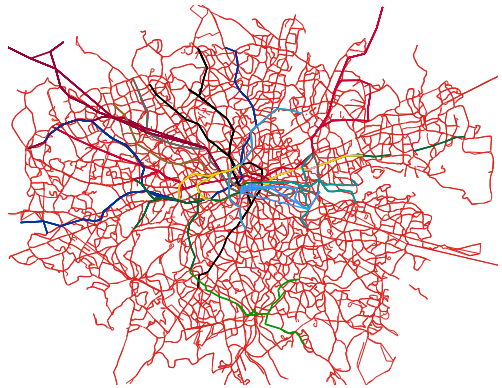
Reduce the dimension/number of criteria

- MR-*x*: increase for every *x* minutes of walking the #transfers by +1

Experimental Evaluation

London, multimodal:

- Roads: 260 k nodes, 1.4 M edges
- Subway, bus, tram, . . .
  21 k stops, 5 M connections
- 564 cycle hire station



Criteria: arrival time, # transfers, walking time

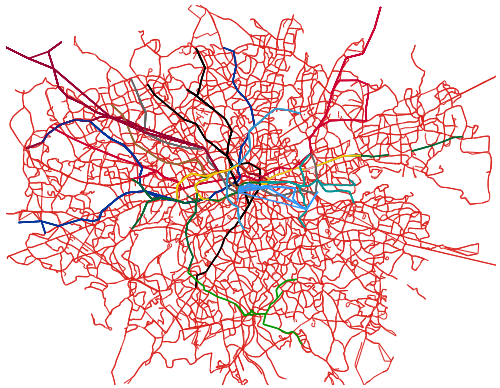
Algorithm	# Sol.	Time [ms]	Quality
MCR	29.1	1 438.7	100 %
MCR-hf	10.9	699.4	89 %
MCR-hb	9.0	456.7	91 %
MCR-t10-r15	13.2	885.0	30 %
MR-10	4.3	39.4	45 %

Intel Xeon E5-2670, 2.6 GHz, 64 GiB DDR3-1600 RAM, 20 MiB L2 cache

Experimental Evaluation

London, multimodal:

- Roads: 260 k nodes, 1.4 M edges
- Subway, bus, tram, . . .
  21 k stops, 5 M connections
- 564 cycle hire station



Criteria: arrival time, # transfers, walking time

Algorithm	# Sol.	Time [ms]	Quality
MCR	29.1	1 438.7	100 %
MCR-hf	10.9	699.4	89 %
MCR-hb	9.0	456.7	91 %
MCR-t10-r15	13.2	885.0	30 %
MR-10	4.3	39.4	45 %

Intel Xeon E5-2670, 2.6 GHz, 64 GiB DDR3-1600 RAM, 20 MiB L2 cache

Conclusion

Summary

- Algorithm Engineering: combination of theory and practice
- (Very) fast route planning on road networks
- Other metrics besides travel time
- Customizable
- Multimodal routeplanning is more expensive
  - Fast methods when only optimizing travel time
  - Network offers many interesting trade-offs between criteria
  - Multicriteria optimization useful, to allow the user to chose his journey
  - Fuzzy filtering is a practical method to rate the journey relevance
- For an overview of algorithmic techniques, see [BDG+14]

Bast et al., *Route planning in transportation networks.* MSR-TR-2014-4

- . . . many security aspects

Thank you for your attention!

Bibliographie I

Ittai Abraham, Daniel Delling, Andrew V. Goldberg, and Renato F. Werneck.

Hierarchical hub labelings for shortest paths.
In *Proceedings of the 20th Annual European Symposium on Algorithms (ESA'12)*, volume 7501 of *Lecture Notes in Computer Science*, pages 24–35. Springer, 2012.

Julian Arz, Dennis Luxen, and Peter Sanders.

Transit node routing reconsidered.
In *Proceedings of the 12th International Symposium on Experimental Algorithms (SEA'13)*, volume 7933 of *Lecture Notes in Computer Science*, pages 55–66. Springer, 2013.

Hannah Bast, Daniel Delling, Andrew V. Goldberg, Matthias Müller–Hannemann, Thomas Pajor, Peter Sanders, Dorothea Wagner, and Renato F. Werneck.
Route planning in transportation networks.
Technical Report MSR-TR-2014-4, Microsoft Research, 2014.

Chris Barrett, Riko Jacob, and Madhav V. Marathe.

Formal-language-constrained path problems.
SIAM Journal on Computing, 30(3):809–837, 2000.

Daniel Delling, Julian Dibbelt, Thomas Pajor, Dorothea Wagner, and Renato F. Werneck.

Computing multimodal journeys in practice.
In *Proceedings of the 12th International Symposium on Experimental Algorithms (SEA'13)*, volume 7933 of *Lecture Notes in Computer Science*, pages 260–271. Springer, 2013.

Daniel Delling, Andrew V. Goldberg, Thomas Pajor, and Renato F. Werneck.

Customizable route planning.
In *Proceedings of the 10th International Symposium on Experimental Algorithms (SEA'11)*, volume 6630 of *Lecture Notes in Computer Science*, pages 376–387. Springer, 2011.

Bibliographie II

Edsger W. Dijkstra.

A note on two problems in connexion with graphs.
Numerische Mathematik, 1:269–271, 1959.

Daniel Delling, Thomas Pajor, and Dorothea Wagner.

Accelerating multi-modal route planning by access-nodes.
In *Proceedings of the 17th Annual European Symposium on Algorithms (ESA'09)*, volume 5757 of *Lecture Notes in Computer Science*, pages 587–598. Springer, September 2009.

Julian Dibbelt, Ben Strasser, and Dorothea Wagner.

Customizable contraction hierarchies.
In *Proceedings of the 13th International Symposium on Experimental Algorithms (SEA'14)*, volume 8504 of *Lecture Notes in Computer Science*, pages 271–282. Springer, 2014.

Marco Farina and Paolo Amato.

A fuzzy definition of "optimality" for many-criteria optimization problems.
IEEE Transactions on Systems, Man, and Cybernetics, Part A, 34(3):315–326, 2004.

Andrew V. Goldberg and Chris Harrelson.

Computing the shortest path: A* search meets graph theory.
In *Proceedings of the 16th Annual ACM–SIAM Symposium on Discrete Algorithms (SODA'05)*, pages 156–165. SIAM, 2005.

Robert Geisberger, Peter Sanders, Dominik Schultes, and Daniel Delling.

Contraction hierarchies: Faster and simpler hierarchical routing in road networks.
In *Proceedings of the 7th Workshop on Experimental Algorithms (WEA'08)*, volume 5038 of *Lecture Notes in Computer Science*, pages 319–333. Springer, June 2008.

Bibliographie III

Andrew V. Goldberg and Renato F. Werneck.

Computing point-to-point shortest paths from external memory.
In *Proceedings of the 7th Workshop on Algorithm Engineering and Experiments (ALENEX'05)*, pages 26–40. SIAM, 2005.

Martin Holzer, Frank Schulz, and Dorothea Wagner.

Engineering multilevel overlay graphs for shortest-path queries.
ACM Journal of Experimental Algorithmics, 13(2.5):1–26, December 2008.

Dominik Kirchler, Leo Liberti, Thomas Pajor, and Roberto Wolfler Calvo.

UniALT for regular language constraint shortest paths on a multi-modal transportation network.
In *Proceedings of the 11th Workshop on Algorithmic Approaches for Transportation Modeling, Optimization, and Systems (ATMOS'11)*, volume 20 of *OpenAccess Series in Informatics (OASIcs)*, pages 64–75, 2011.

Ulrich Lauther.

An extremely fast, exact algorithm for finding shortest paths in static networks with geographical background.
In *Geoinformation und Mobilität - von der Forschung zur praktischen Anwendung*, volume 22, pages 219–230. IfGI prints, 2004.

Frank Schulz, Dorothea Wagner, and Karsten Weihe.

Dijkstra's algorithm on-line: An empirical case study from public railroad transport.
ACM Journal of Experimental Algorithmics, 5(12):1–23, 2000.