Karlsruhe Institute of Technology

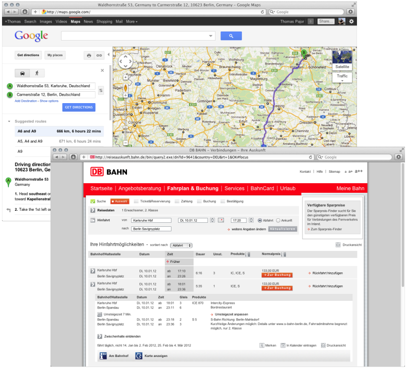
The Impact of Route Planning Algorithms in Practice

SEA 2018, L'Aquila

Dorothea Wagner | June 28, 2018

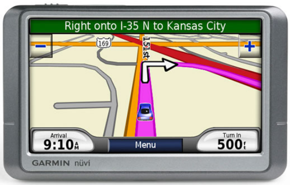
KARLSRUHE INSTITUTE OF TECHNOLOGY – INSTITUTE OF THEORETICAL INFORMATICS – GROUP ALGORITHMICS

Motivation



Important applications, e.g.,

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



Navigation Device for the World

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

Institute for Theoretical Informatics
Chair Algorithmics

Core Problem

Request:

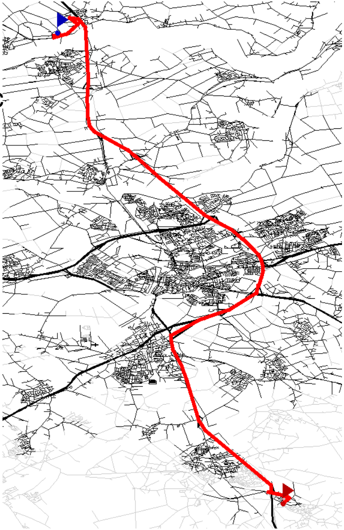
- Find the best connection in a transportation network w.r.t. some metric

Idea:

- Network as graph $G = (V, E)$
- Edge weights are according to metric
- Shortest paths in G equal best connections
- Classic problem (Dijkstra 1959)

Problems:

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



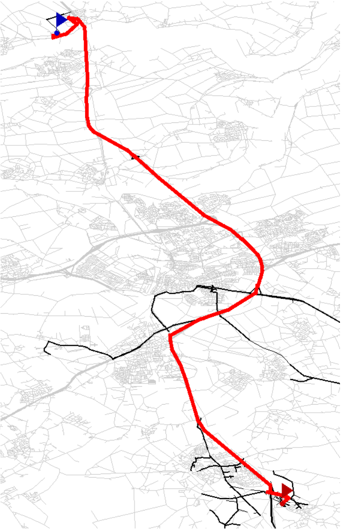
Speed-Up Techniques

Observations:

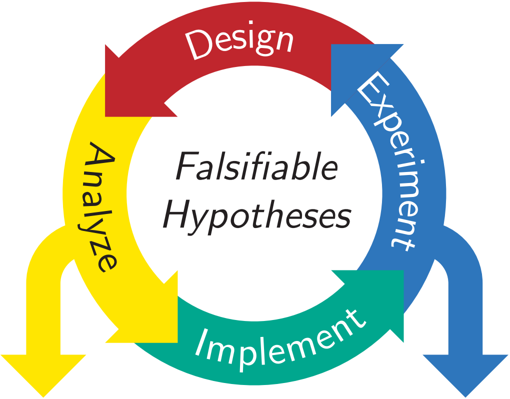
- Dijkstra visits all nodes closer than the target
- Unnecessary computations
- Many requests in a hardly changing network

Idea:

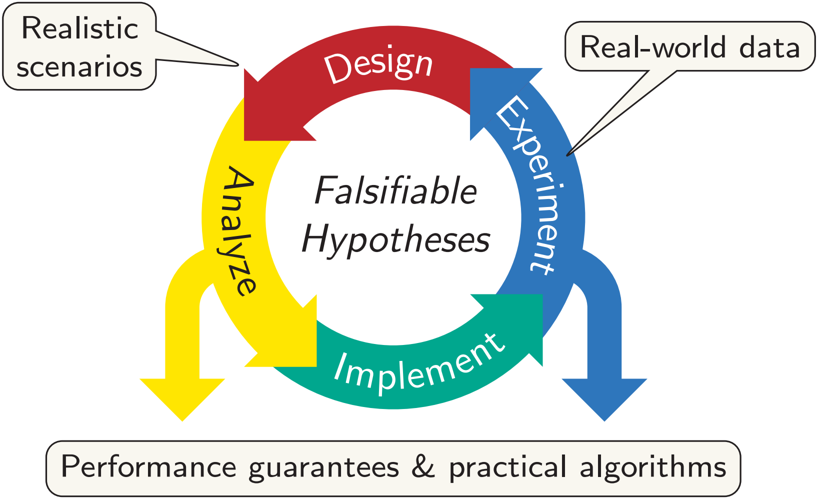
- Two-phase algorithm:
  - Offline: compute additional data during preprocessing
  - Online: speed-up query with this data
- 3 criteria: preprocessing time and space, speed-up over Dijkstra



Showpiece of Algorithm Engineering



Showpiece of Algorithm Engineering



State-of-the-Art

Karlsruhe Institute of Technology

Many techniques tuned for continent-sized road networks:

- Arc-Flags [2004,2006,2009,2013]
- Multi-Level Dijkstra [2000,2008,2009,2011,2016]
- ALT: A*, Landmarks, Triangle Inequality [1968,2005,2012]
- Reach [2004,2007]
- Contraction Hierarchies (CH, CCH) [2008,2013,2014,2016]
- Transit Node Routing (TNR) [2007,2013]
- Hub Labeling (HL) [2003,2011,2013,2014]

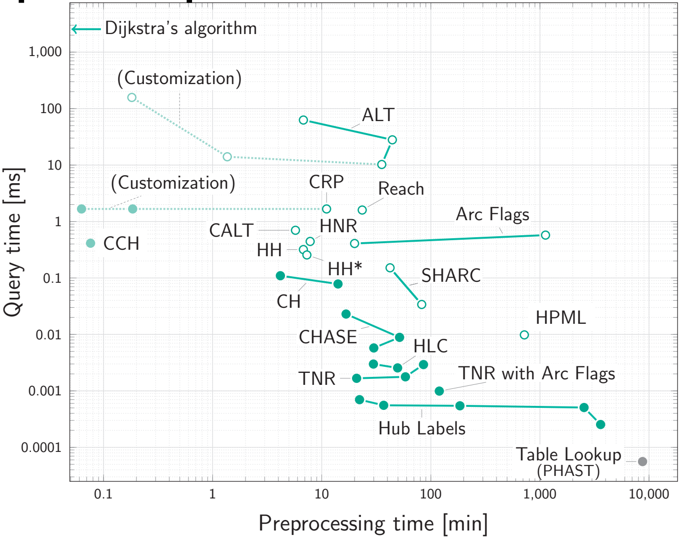
Timetable information:

- Transfer Pattern [2010,2016]
- Raptor [2013]
- Connection Scan [2013,2014,2017]
- Trip-Based Public Transit Routing [2015,2016]

Survey on “Route Planning in Transprotation Networks” [Bast et al.’16]

Institute for Theoretical Informatics
Chair Algorithmics

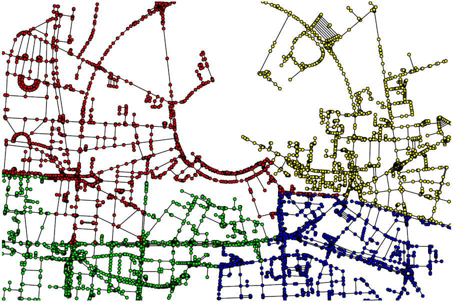
Speedup Techniques [Bast et al.'16]



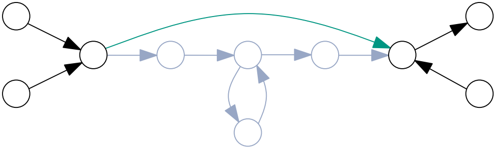
In use at Apple, Bing, Google, TomTom, ...

Basic Techniques

- Partition Network



- Shortcuts



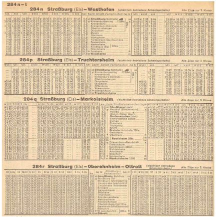
New Challenges

Energy Consumption of Electric Vehicles:

- Restricted battery capacity
- “Range anxiety”

Customizable Metrics and Time-Dependency:

- User preferences
- Traffic congestion
- Historic travel time data

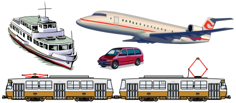


Timetable Information:

- Shortest paths in a timetable graph
- Timetable graphs differ from road graphs

Multimodal Route Planning:

- Incorporate unrestricted walking
- Change mode of transportation during the journey



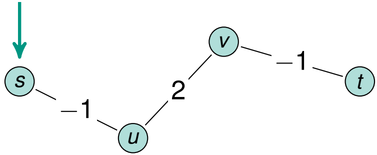
Route Planning for Electric Vehicles

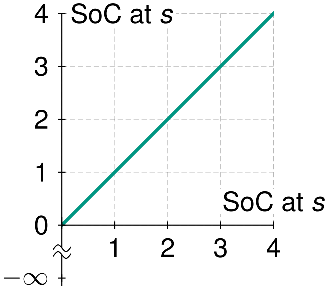
Route Planning for Electric Vehicles

- **Recuperation:** Negative edge costs (no negativen cycles)
- **Battery constraints:** Battery has a limited capacity

SoC function maps SoC (“state of charge”) at source to SoC at target

Example:
min. SoC 0, max. SoC 4



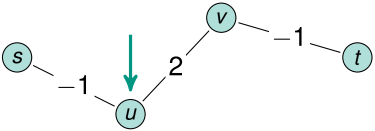


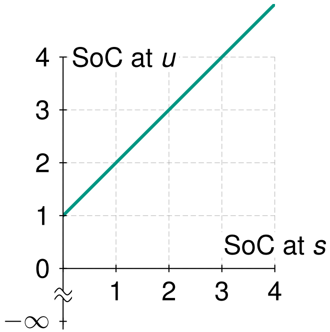
Route Planning for Electric Vehicles

- **Recuperation:** Negative edge costs (no negativen cycles)
- **Battery constraints:** Battery has a limited capacity

SoC function maps SoC (“state of charge”) at source to SoC at target

Example:
min. SoC 0, max. SoC 4



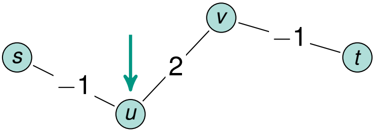


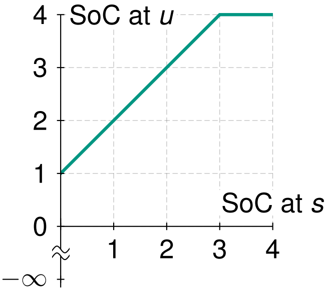
Route Planning for Electric Vehicles

- **Recuperation:** Negative edge costs (no negativen cycles)
- **Battery constraints:** Battery has a limited capacity

SoC function maps SoC (“state of charge”) at source to SoC at target

Example:
min. SoC 0, max. SoC 4



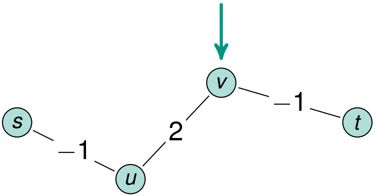


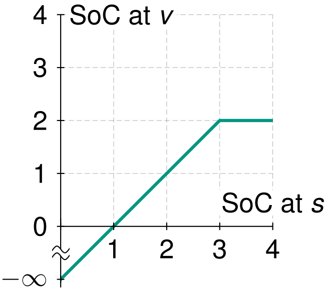
Route Planning for Electric Vehicles

- **Recuperation:** Negative edge costs (no negativen cycles)
- **Battery constraints:** Battery has a limited capacity

SoC function maps SoC (“state of charge”) at source to SoC at target

Example:
min. SoC 0, max. SoC 4



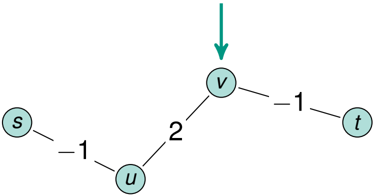


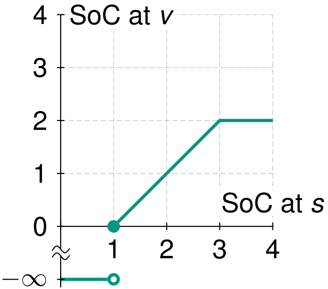
Route Planning for Electric Vehicles

- **Recuperation:** Negative edge costs (no negativen cycles)
- **Battery constraints:** Battery has a limited capacity

SoC function maps SoC (“state of charge”) at source to SoC at target

Example:
min. SoC 0, max. SoC 4



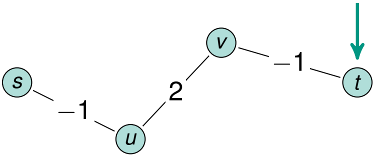


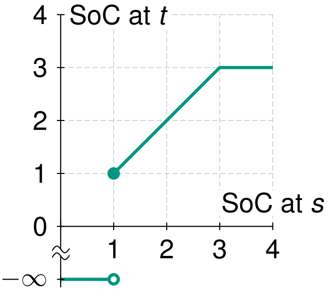
Route Planning for Electric Vehicles

- **Recuperation:** Negative edge costs (no negativen cycles)
- **Battery constraints:** Battery has a limited capacity

SoC function maps SoC (“state of charge”) at source to SoC at target

Example:
min. SoC 0, max. SoC 4



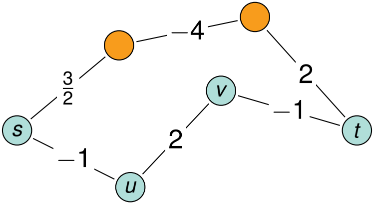


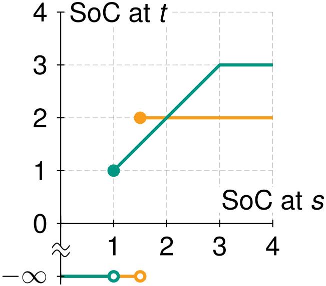
Route Planning for Electric Vehicles

- **Recuperation:** Negative edge costs (no negativen cycles)
- **Battery constraints:** Battery has a limited capacity

SoC function maps SoC (“state of charge”) at source to SoC at target

Example:
min. SoC 0, max. SoC 4



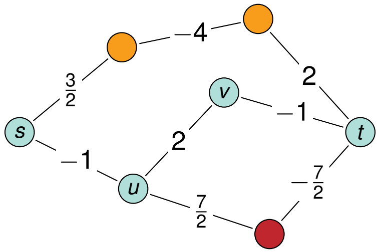


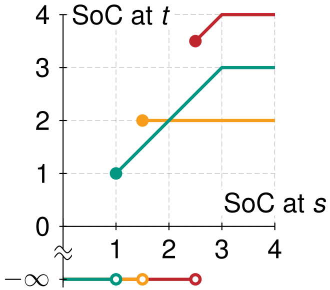
Route Planning for Electric Vehicles

- **Recuperation:** Negative edge costs (no negativen cycles)
- **Battery constraints:** Battery has a limited capacity

SoC function maps SoC (“state of charge”) at source to SoC at target

Example:
min. SoC 0, max. SoC 4



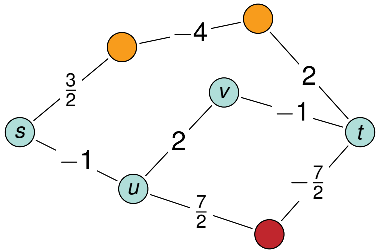


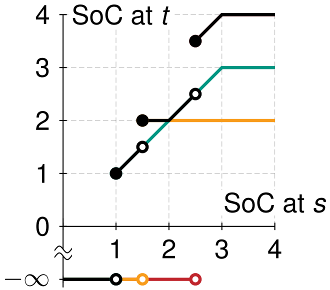
Route Planning for Electric Vehicles

- **Recuperation:** Negative edge costs (no negativen cycles)
- **Battery constraints:** Battery has a limited capacity

SoC function maps SoC (“state of charge”) at source to SoC at target

Example:
min. SoC 0, max. SoC 4



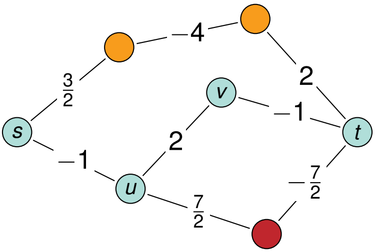


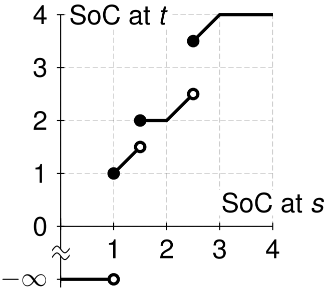
Route Planning for Electric Vehicles

- **Recuperation:** Negative edge costs (no negativen cycles)
- **Battery constraints:** Battery has a limited capacity

SoC function maps SoC (“state of charge”) at source to SoC at target

Example:
min. SoC 0, max. SoC 4





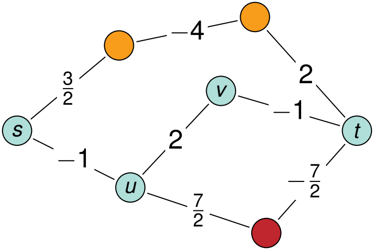
Institute for Theoretical Informatics
Chair Algorithmics

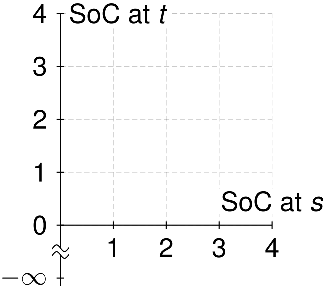
Route Planning for Electric Vehicles

- **Recuperation:** Negative edge costs (no negativen cycles)
- **Battery constraints:** Battery has a limited capacity

SoC function maps SoC (“state of charge”) at source to SoC at target

Example:
min. SoC 0, max. SoC 4

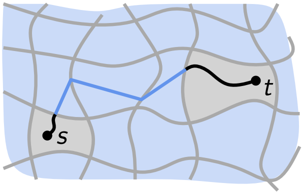


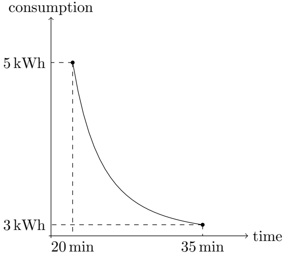


- Speedup techniques have to evaluate functions [Eisner et al.’11]

Energy-Optimal Routes [Baum et al.'13]

- Shortcuts are functions, not scalar values
- Bidirectional search more complicated (unknown state-of-charge at target)
- User-dependent consumption profiles (⇒ custom metrics)





Experiments:

- Fast queries (few milliseconds)
- Fast customization (few seconds)

But: Energy-optimal routes follow slow roads

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

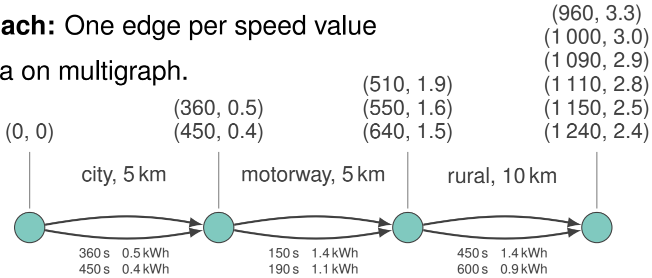
⇒ Consider tradeoff between speed and energy consumption

Find the fastest path such that the battery does not run out: $\mathcal{NP}$**-hard**

Constrained Shortest Paths

- Energy can be saved driving below speed limit
- Additional instructions to the driver
- **Simple approach:** One edge per speed value

⇒ Bicriteria Dijkstra on multigraph.



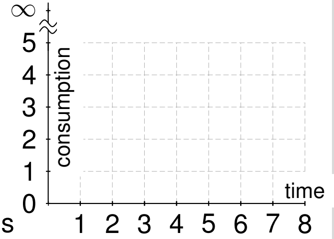
Worst case: n vertices with k parallel edges produce $\Theta(k^n)$ solutions

Simple implementation, but impractical running times

Realistic Model [Baum et al.'17]

Idea: Use continuous tradeoff functions instead of samples

- Times limits $\underline{x}$, $\overline{x}$ (speed limit, traffic flow, ...)
- More accurate model
- Less complex solution space



TFP: Tradeoff Function Propagating Algorithm

- Extends Bicriteria Dijkstra to tradeoff functions

CHAsp = CH & A* & TFP:

- Combines TFP with speedup techniques

Experiments:

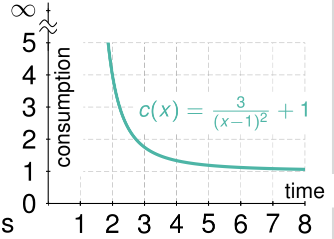
- Moderate preprocessing effort (Europe ∼3 h; Germany ∼30 min)
- Fast exact queries for typical ranges ($<$1 sec)
- Even faster heuristics ($<$100 ms, average error $<$1%)

Institute for Theoretical Informatics
Chair Algorithmics

Realistic Model [Baum et al.'17]

Idea: Use continuous tradeoff functions instead of samples

- Times limits $\underline{x}$, $\overline{x}$ (speed limit, traffic flow, ...)
- More accurate model
- Less complex solution space



TFP: Tradeoff Function Propagating Algorithm

- Extends Bicriteria Dijkstra to tradeoff functions

CHAsp = CH & A* & TFP:

- Combines TFP with speedup techniques

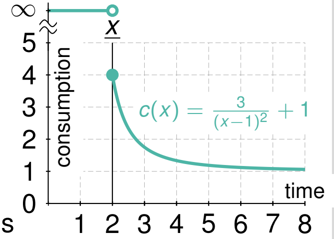
Experiments:

- Moderate preprocessing effort (Europe ~3 h; Germany ~30 min)
- Fast exact queries for typical ranges (<1 sec)
- Even faster heuristics (<100 ms, average error <1%)

Realistic Model [Baum et al.'17]

Idea: Use continuous tradeoff functions instead of samples

- Times limits $\underline{x}$, $\overline{x}$ (speed limit, traffic flow, ...)
- More accurate model
- Less complex solution space



TFP: Tradeoff Function Propagating Algorithm

- Extends Bicriteria Dijkstra to tradeoff functions

CHAsp = CH & A* & TFP:

- Combines TFP with speedup techniques

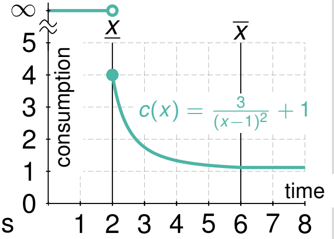
Experiments:

- Moderate preprocessing effort (Europe ~3 h; Germany ~30 min)
- Fast exact queries for typical ranges (<1 sec)
- Even faster heuristics (<100 ms, average error <1%)

Realistic Model [Baum et al.'17]

Idea: Use continuous tradeoff functions instead of samples

- Times limits $\underline{x}$, $\overline{x}$ (speed limit, traffic flow, ...)
- More accurate model
- Less complex solution space



TFP: Tradeoff Function Propagating Algorithm

- Extends Bicriteria Dijkstra to tradeoff functions

CHAsp = CH & A* & TFP:

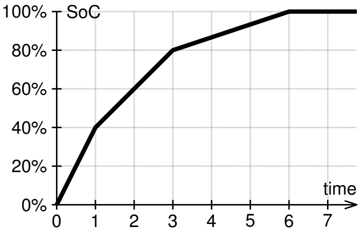
- Combines TFP with speedup techniques

Experiments:

- Moderate preprocessing effort (Europe ∼3 h; Germany ∼30 min)
- Fast exact queries for typical ranges (<1 sec)
- Even faster heuristics (<100 ms, average error <1%)

Including Charging Stops [Baum et al.'15]

- Recharging allowed at some nodes (but requires charging time).
- Realistic models of charging stations:
  - Charging power varies
  - Super chargers
  - Battery swapping stations



Challenges:

1. Recuperation, battery constraints
2. Energy efficient driving vs. time consuming charging stops
   - Detour for reaching a charging station
3. Charging is not uniform
   - Interrupt charging and take another station later

Observations

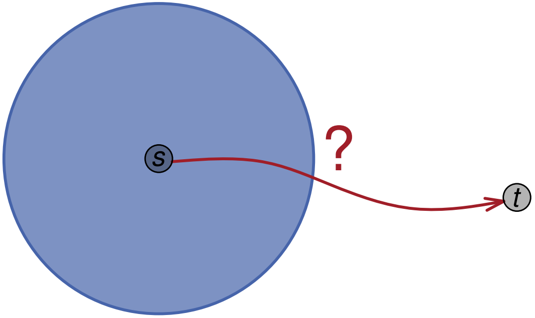
Find the fastest route from *s* to *t*:

Reachable area

Charging station

Observations

Find the fastest route from s to t:

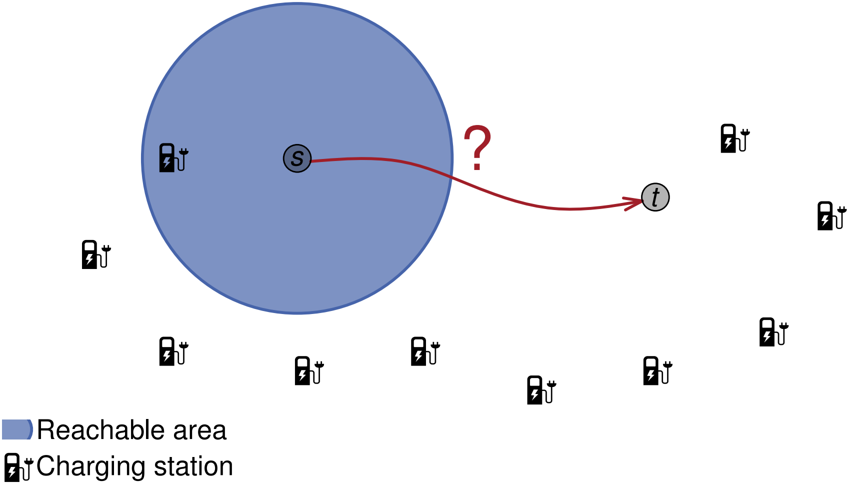


Reachable area

Charging station

Observations

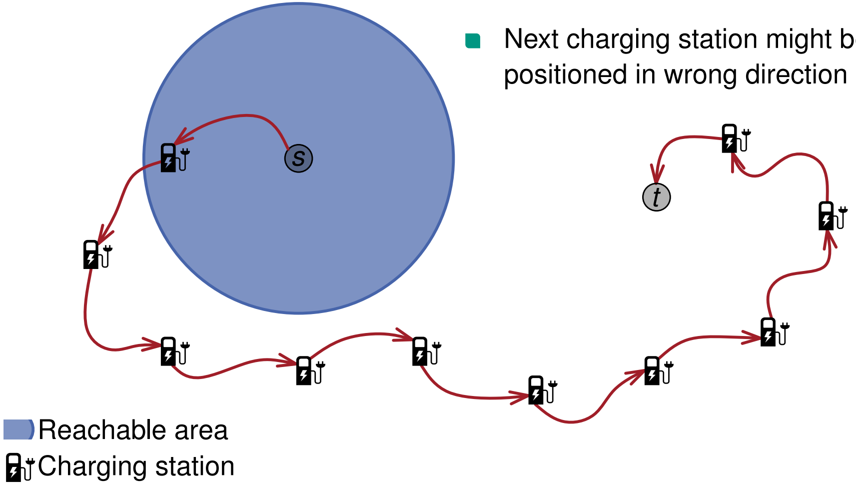
Find the fastest route from s to t:



Observations

KIT Karlsruhe Institute of Technology

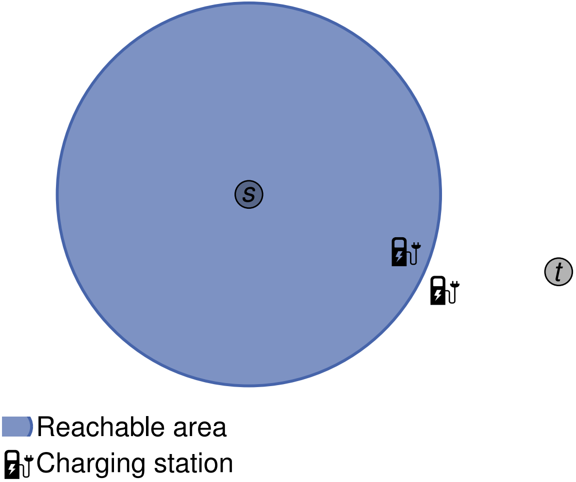
Find the fastest route from *s* to *t*:



- Next charging station might be positioned in wrong direction

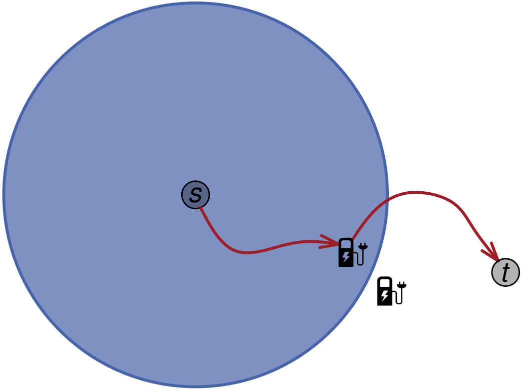
Observations

Find the fastest route from *s* to *t*:



Observations

Find the fastest route from *s* to *t*:

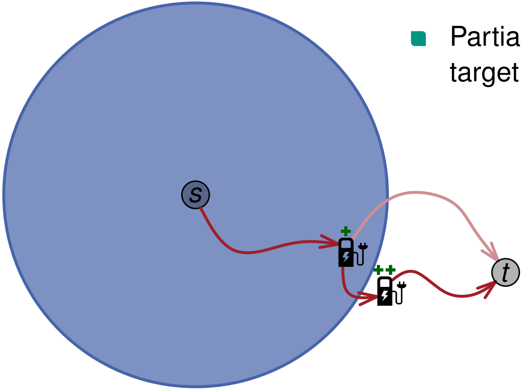


Reachable area

Charging station

Observations

Find the fastest route from *s* to *t*:



- Partial recharging, even if the target is already reachable

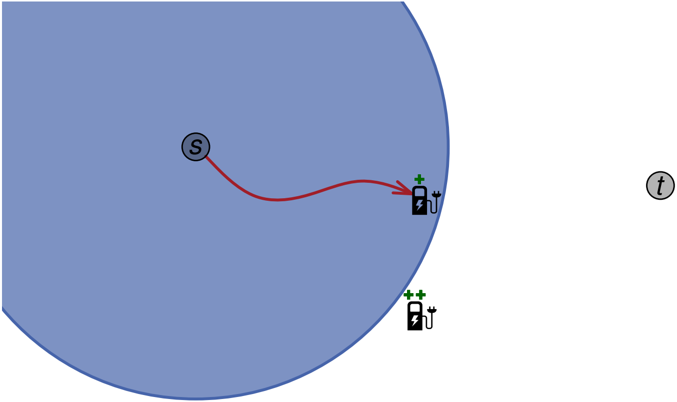
Reachable area

Charging station

Fast charging station / swapping station

Observations

Find the fastest route from *s* to *t*:



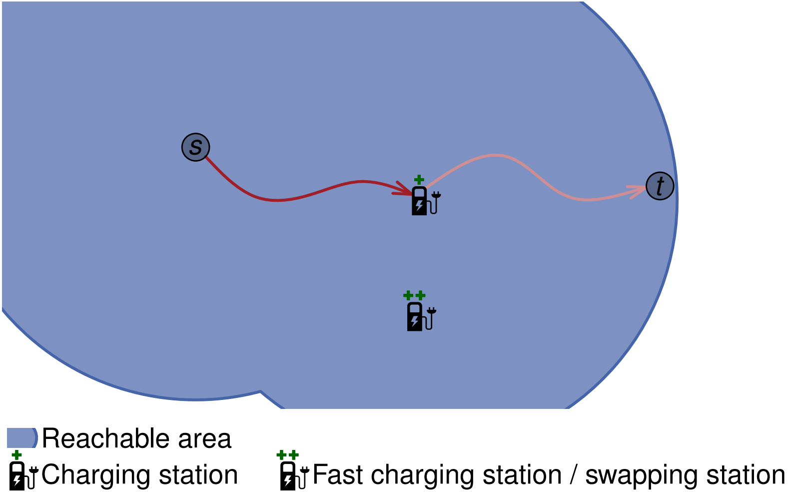
Reachable area

Charging station

Fast charging station / swapping station

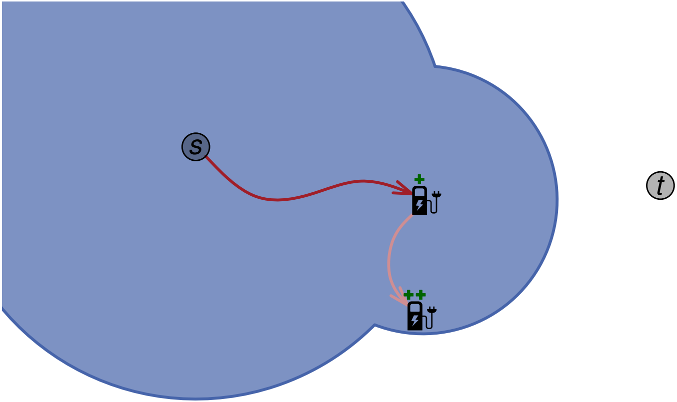
Observations

Find the fastest route from s to t:



Observations

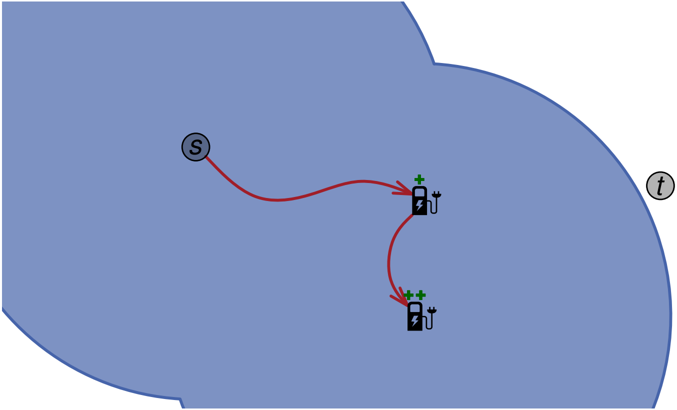
Find the fastest route from *s* to *t*:



Observations

Karlsruhe Institute of Technology

Find the fastest route from s to t:



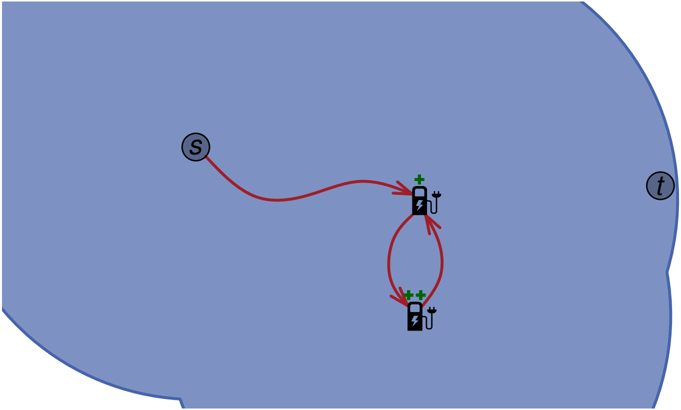
Reachable area

Charging station

Fast charging station / swapping station

Observations

Find the fastest route from *s* to *t*:



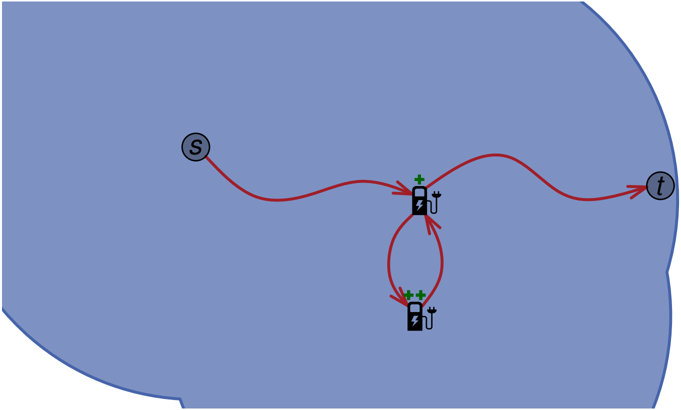
Reachable area

Charging station Fast charging station / swapping station

Observations

Find the fastest route from *s* to *t*:

- Fastest route may contain cycles



Reachable area

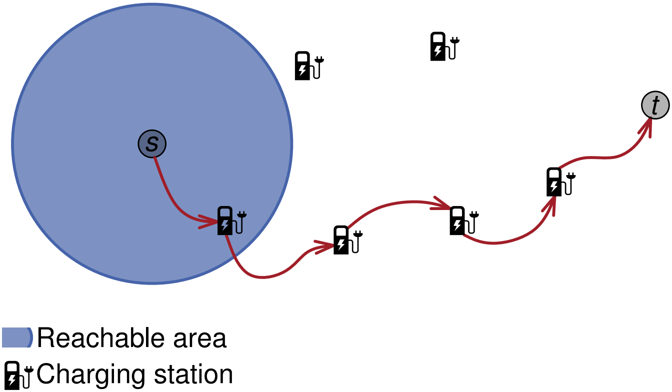
Charging station

Fast charging station / swapping station

Observations

KIT
Karlsruhe Institute of Technology

Find the fastest route from *s* to *t*:

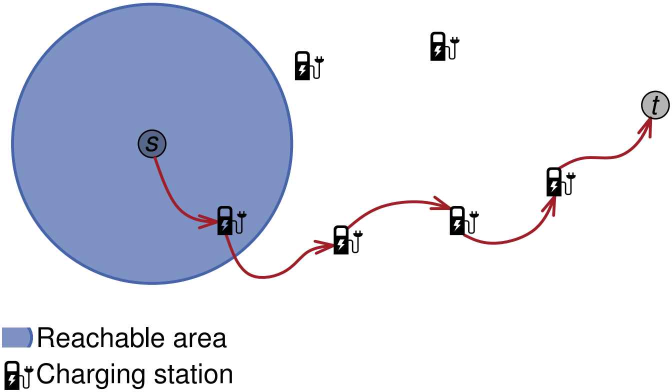


Observations

KIT
Karlsruhe Institute of Technology

Find the fastest route from *s* to *t*:

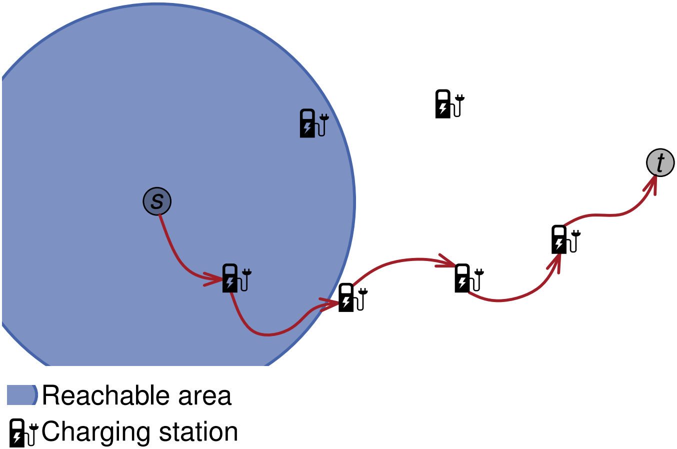
- Larger battery $\Rightarrow$ simpler problem ?



Observations

Find the fastest route from *s* to *t*:

- Larger battery $\Rightarrow$ simpler problem ?

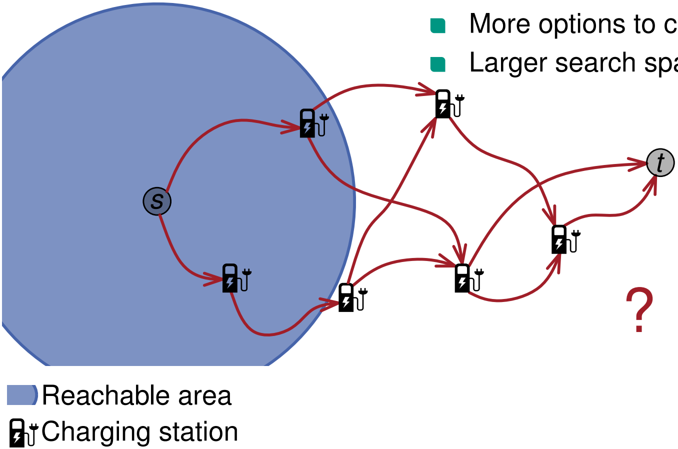


Institute for Theoretical Informatics
Chair Algorithmics

Observations

Find the fastest route from *s* to *t*:

- Larger battery ⇒ simpler problem ?
- More options to consider
- Larger search space



Charging Function Propagation

CFP Algorithm

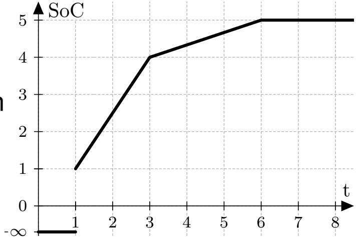
- Based on bicriteria Dijkstra
- If no charging station has been used: label = tuple (travel time, SoC)
- Per vertex: Maintain set of Pareto-optimal labels

Problem: When reaching a charging station: How long to stay?

- Depends on the remaining path to target
- Optimal state-of-charge for departure yet unknown

Solution:

- Delay this decision!
- Keep track of last passed charging station
- Labels represent charging tradeoffs



CHArge

Karlsruhe Institute of Technology

CHArge = CH & A* & CFP:

- Combines CFP with speedup techniques
- Can handle arbitrary charging station types

Experiments:

- Moderate preprocessing times
  Europe $\sim$30 min; Germany $\sim$5 min
- Fast queries on continental-sized networks
  Europe $\sim$1 min; Germany $\sim$1 sec
- Even better results possible, using heuristics
  Europe $\sim$0.1–1 sec; Germany $\sim$20–100 ms
  often optimal solutions, mean error $\sim$1%

Range Visualization

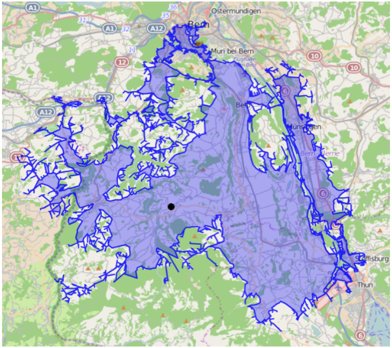
Visualize area reachable by an EV

Goals:

- Exact visualization
- Polygons with few segments
- Fast Computation

Subproblems:

1. Compute reachable subgraph [Baum et al.'15]
2. Compute polygon for visualization [Baum et al.'16]



Experiments: Polygons with $\sim$1 000 segments in $<$100 ms

Range Visualization

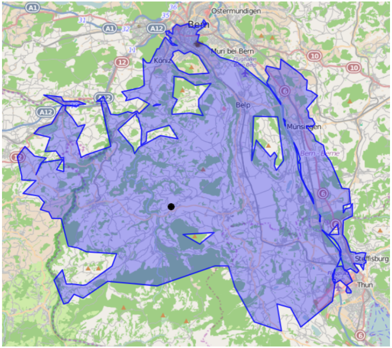
Visualize area reachable by an EV

Goals:

- Exact visualization
- Polygons with few segments
- Fast Computation

Subproblems:

1. Compute reachable subgraph [Baum et al.'15]
2. Compute polygon for visualization [Baum et al.'16]



Experiments: Polygons with $\sim$1 000 segments in $<$100 ms

Time-Dependency

Real-World Weights

Practice:

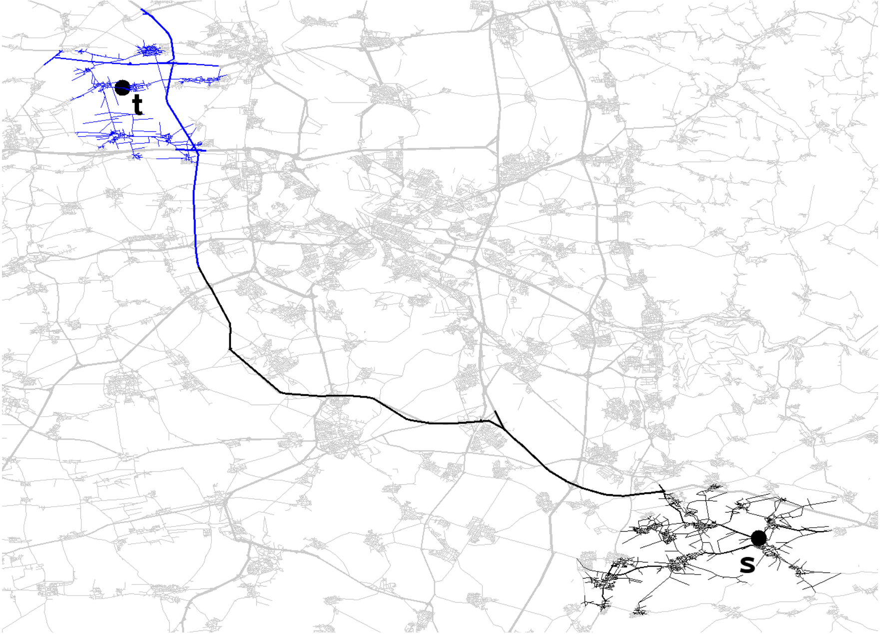
- Traffic congestion
- Accidents
- Road constructions/closures, . . .
- Driver prefers scenic routes, right-turns, . . .
- Driver dislikes highways, a specific road X, . . .
- Historic travel time data

Consequences:

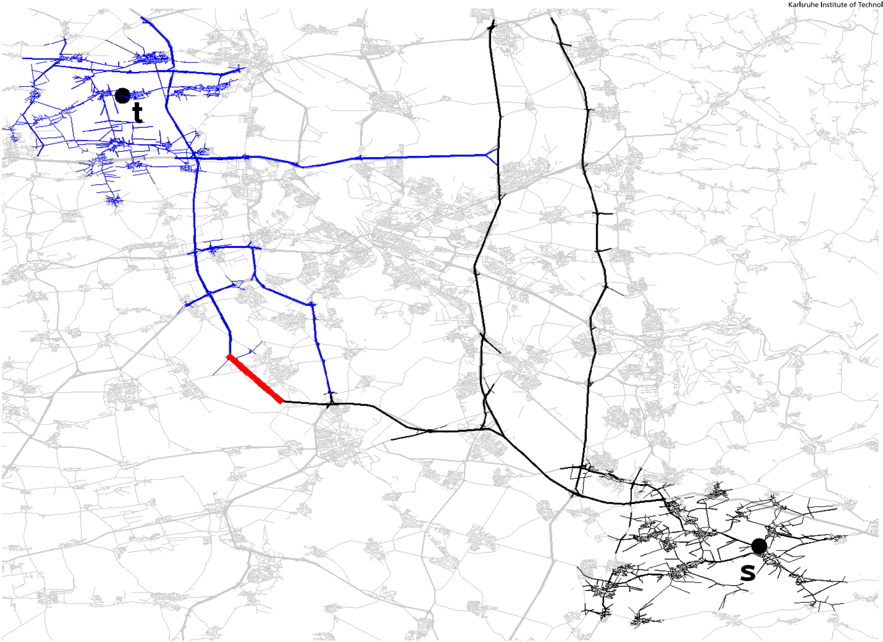
- Weights change unexpectedly and/or depend on the user
- Changes shortest path structure
- Invalidates preprocessed data

Institute for Theoretical Informatics
Chair Algorithmics

Effect of Road Closure



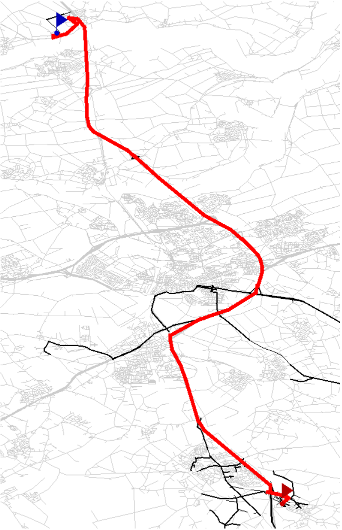
Effect of Road Closure



Shortest Path Computation

Two-phase:

- Preprocessing (slow): compute additional data
- Query (fast): answer *st*-queries using data from preprocessing



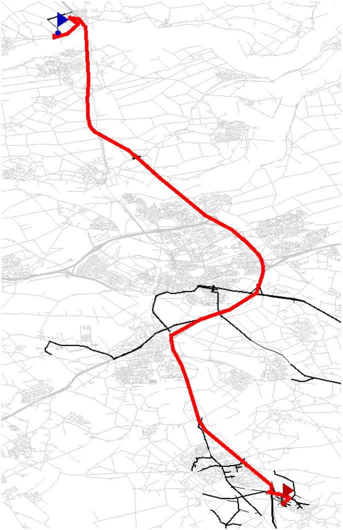
Shortest Path Computation

Two-phase:

- Preprocessing (slow): compute additional data
- Query (fast): answer *st*-queries using data from preprocessing

Three-phase:

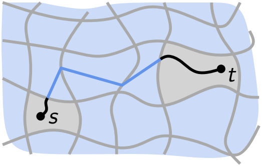
- Preprocessing (slow): compute additional weight-independent data
- Customization (reasonably fast): introduce weights
- Query (fast): answer *st*-queries using data from preprocessing and customization



Customizable Route Planning

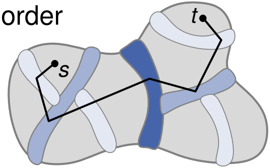
Customizable Multi-level Dijkstra (MLD) [Delling et al. ’11, ’13, ’15]

- Metric independent partition of the graph
- Customization: Compute clique per cell
- Multi-level Dijkstra queries



Customizable Contraction Hierarchies (CCH) [Dibbelt et al. ’14, ’16]

- Uses metric independent nested dissection order
- Customization: Compute shortcut weights
- Elimination-tree query (requires no queue)



Historic Traffic Data

Motivation:

- Traffic around urban centers follows predictable patterns
- In the morning everyone rushes to work → traffic jam
- At noon less traffic
- In the evening everyone goes home → jam in the other direction
- But not on Sunday

Historic Traffic Data

Motivation:

- Traffic around urban centers follows predictable patterns
- In the morning everyone rushes to work $\rightarrow$ traffic jam
- At noon less traffic
- In the evening everyone goes home $\rightarrow$ jam in the other direction
- But not on Sunday

- Exact patterns vary from region to region
- $\rightarrow$ aggregate historic data to make a prediction for each weekday
- Historic traffic aware routing? $\Rightarrow$ Time-dependent shortest paths

Handling Time-Dependency

Schedule-based public transit networks

- Trains depart and arrive at discrete times
- Time-expanded graph:
  - Model discrete events as nodes
  - Connect subsequent events by arcs
    (by same train or at same stop)

Institute for Theoretical Informatics
Chair Algorithmics

Handling Time-Dependency

Schedule-based public transit networks

- Trains depart and arrive at discrete times
- Time-expanded graph:
  - Model discrete events as nodes
  - Connect subsequent events by arcs
    (by same train or at same stop)

Time-dependent road networks

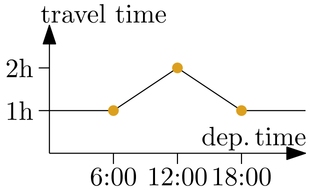
- Can enter an edge at any moment
- Infinite time-expanded graph

$\Rightarrow$ Not what we want

Travel Time Function

Approach: Encode time-dependency in edge weights

Dep.Time	Travel Time
6:00	1h
12:00	2h
18:00	1h

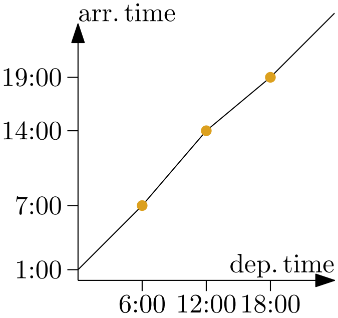


- Linearly interpolate between a finite set of breakpoints

Arrival Time Function

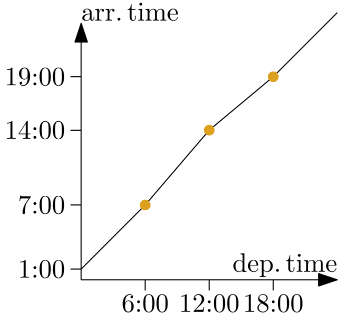
Approach: Encode time-dependency in edge weights

Dep.Time	Arr.Time
6:00	7:00
12:00	14:00
18:00	19:00

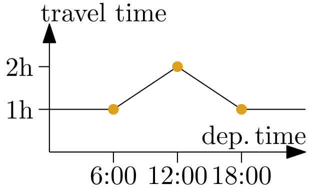


- Linearly interpolate between a finite set of breakpoints

Two Views



$f(x)$



$g(x)$

- Two views on the same information
- $f(x) = g(x) + x$
- This talk: use most convenient view for current task

Properties of TD functions

FIFO: First-In-First-Out

- Per road segment: If we depart later, we arrive later
- Property of most, realistic TD networks
- Formally for arrival time function f:

$$f(x) < f(y) \text{ for } x < y$$

- $\Rightarrow$ f is bijection and f^{-1} exists

Institute for Theoretical Informatics
Chair Algorithmics

Properties of TD functions

Karlsruhe Institute of Technology

FIFO: First-In-First-Out

- Per road segment: If we depart later, we arrive later
- Property of most, realistic TD networks
- Formally for arrival time function f:

$$f(x) < f(y) \text{ for } x < y$$

- $\Rightarrow f$ is bijection and f^{-1} exists

Periodicity

- After period Π (24h in the examples) traffic patterns repeat
- Formally for travel time function g:

$$g(x) = g(x + k\Pi) \text{ for all integers } k$$

Time-dep. Speedup Techniques

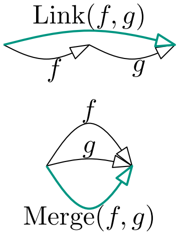
Many speedup techniques have been adapted for time-dependency:

- ArcFlags: Flag arcs if optimal at some point in time
- ALT: Avoid time-dependency during preprocessing, by using lower bounds on functions
- TD-HL would require too much space (time-independent HL already very expensive in space)
- Shortcut/Overlay-based techniques require additional operations on TD functions
- TD-CRP: Requires profile-queries during customization
- Time-Dependent-Sampling uses multiple time-independent CHs (cannot guarantee correctness)

Institute for Theoretical Informatics
Chair Algorithmics

Time-dependent Shortcuts

- Linking: Needed to build shortcuts
- Merging: Needed to reduce multi-edges



Linking : Computing $h(x) = g(f(x))$

Function f	
Dep.Time	Arr.Time
6:00	7:00
12:00	14:00
18:00	19:00

Function g	
Dep.Time	Arr.Time
8:00	10:00
10:00	11:00
15:00	17:00

Institute for Theoretical Informatics
Chair Algorithmics

Linking : Computing $h(x) = g(f(x))$

Function f	
Dep.Time	Arr.Time
6:00	7:00
12:00	14:00
18:00	19:00

Function g	
Dep.Time	Arr.Time
8:00	10:00
10:00	11:00
15:00	17:00

Function h		
Dep.Time	Mid.Time	Arr.Time
6:00	7:00	?
?	8:00	10:00
?	10:00	11:00
12:00	14:00	?
?	15:00	17:00
18:00	19:00	?

Linking : Computing $h(x) = g(f(x))$

Function f

Dep.Time	Arr.Time
6:00	7:00
12:00	14:00
18:00	19:00

Function g

Dep.Time	Arr.Time
8:00	10:00
10:00	11:00
15:00	17:00

Function h

Dep.Time	Mid.Time	Arr.Time
6:00	7:00	$g(7{:}00)$
$f^{-1}(8{:}00)$	8:00	10:00
$f^{-1}(10{:}00)$	10:00	11:00
12:00	14:00	$g(14{:}00)$
$f^{-1}(15{:}00)$	15:00	17:00
18:00	19:00	$g(19{:}00)$

Linking : Computing $h(x) = g(f(x))$

Function f

Dep.Time	Arr.Time
6:00	7:00
12:00	14:00
18:00	19:00

Function g

Dep.Time	Arr.Time
8:00	10:00
10:00	11:00
15:00	17:00

Function h

Dep.Time	Mid.Time	Arr.Time
6:00	7:00	9:00
≈6:51	8:00	10:00
≈8:34	10:00	11:00
12:00	14:00	15:48
13:12	15:00	17:00
18:00	19:00	21:00

Linking : Computing $h(x) = g(f(x))$

Function f

Dep.Time	Arr.Time
6:00	7:00
12:00	14:00
18:00	19:00

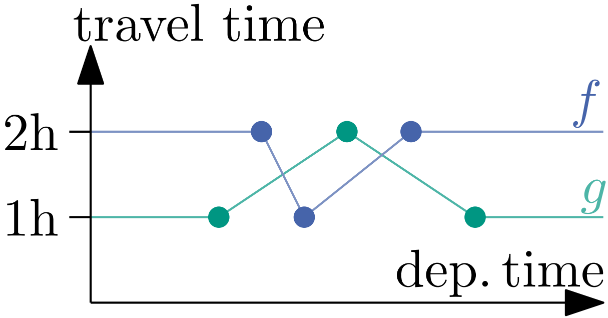
Function g

Dep.Time	Arr.Time
8:00	10:00
10:00	11:00
15:00	17:00

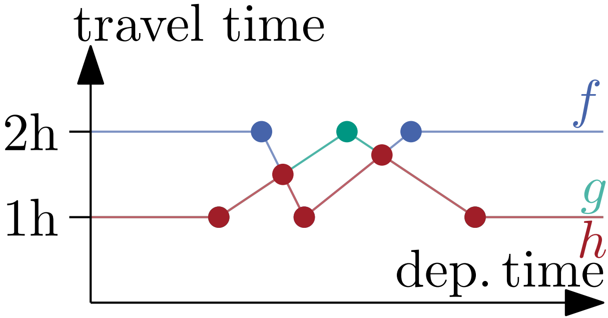
Function h

Dep.Time	Arr.Time
6:00	9:00
$\approx$6:51	10:00
$\approx$8:34	11:00
12:00	15:48
13:12	17:00
18:00	21:00

Merging : $h(x) = \min\{f(x), g(x)\}$

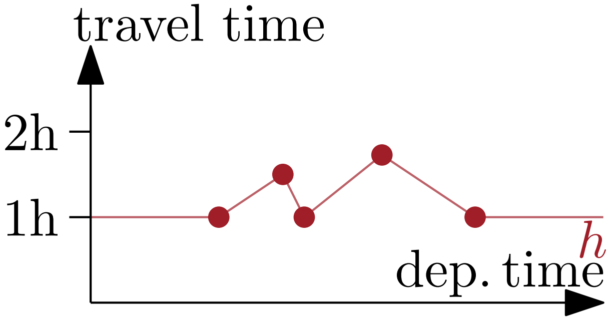


Merging : $h(x) = \min\{f(x), g(x)\}$



Institute for Theoretical Informatics
Chair Algorithmics

Merging : $h(x) = \min\{f(x), g(x)\}$

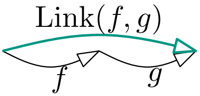


Complexity

$|f| = \#$ breakpoints of function f

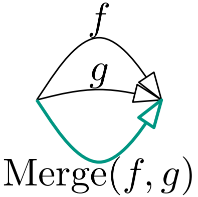
Linking:

- Space: $|h| \leq |f| + |g|$
- Time: $O(|f| + |g|)$
  - Coordinated sweep of both functions

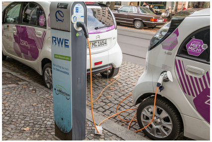


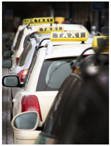
Merging:

- Space: $|h| \leq 2 \cdot (|f| + |g|)$
- Time: $O(|f| + |g|)$
  - Also coordinated sweep



Multimodal Route Planning

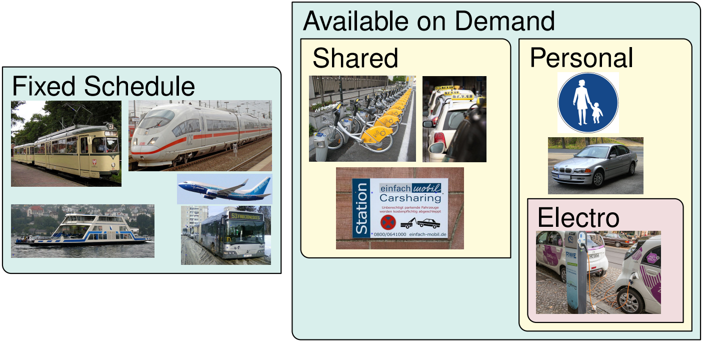




- Many modes of transportation

Institute for Theoretical Informatics
Chair Algorithmics

Multimodal Route Planning



- Many modes of transportation
- Many different set of rules
- and many more modes and variations exist

Unrestricted Walking

Common Algorithms & Walking Restrictions:

Algorithm	Footpaths
RAPTOR [Delling et al. '12/'14]	Transitively closed
CSA [Dibbelt et al. '13/'14]	Transitively closed
Trip-Based Routing [Witt '15]	Transitively closed
Transfer Patterns [Bast et al. '10/'16]	Max. 400 meters
Frequency-Based [Bast, Storandt '14]	Max. 15 minutes
Public Transit Labeling [Delling et al. '15]	As specified by the timetable

Unrestricted Walking

Common Algorithms & Walking Restrictions:

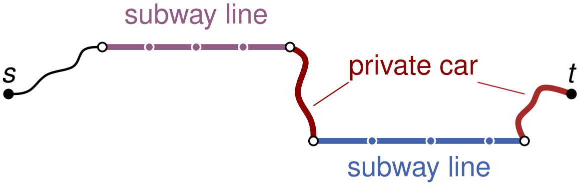
Algorithm	Footpaths
RAPTOR [Delling et al. '12/'14]	Transitively closed
CSA [Dibbelt et al. '13/'14]	Transitively closed
Trip-Based Routing [Witt '15]	Transitively closed
Transfer Patterns [Bast et al. '10/'16]	Max. 400 meters
Frequency-Based [Bast, Storandt '14]	Max. 15 minutes
Public Transit Labeling [Delling et al. '15]	As specified by the timetable

Problems:

- Transitively close graph $\Rightarrow$ limited walking
  (e.g. walking $\leq$ 15 min $\Rightarrow$ avg. degree $>$ 100)
- Unrestricted walking reduces travel times significantly [Wagner & Zündorf '17]
- Open problem: Efficient algorithms

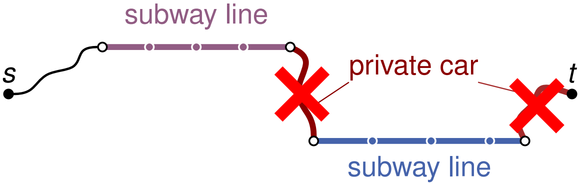
Multiple Transportation Modes

Problem: Unrestricted routes allow arbitrary transfers



Multiple Transportation Modes

Problem: Unrestricted routes allow arbitrary transfers

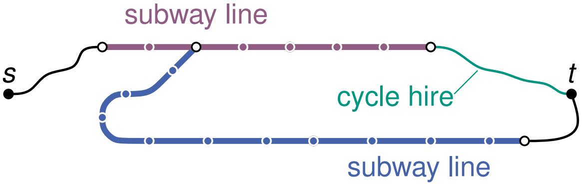


- Not all sequences of transportation modes are reasonable

Multiple Transportation Modes

[Delling et al.'09, Dibbelt et al.'12]

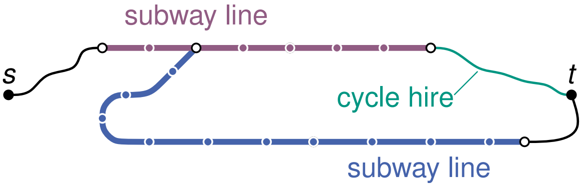
Problem: Unrestricted routes allow arbitrary transfers



- Not all sequences of transportation modes are reasonable
- Label constrained shortest paths
- Dijkstra's algorithm on product of network and finite-state automaton
- Adopt speed-up techniques

Multiple Transportation Modes

Shortcoming



Multiple Transportation Modes

Shortcoming

Multiple Transportation Modes

Shortcoming

s ? t

- Restrictions must be known in advance
- User might not know them
- Only one route is computed (no alternatives)

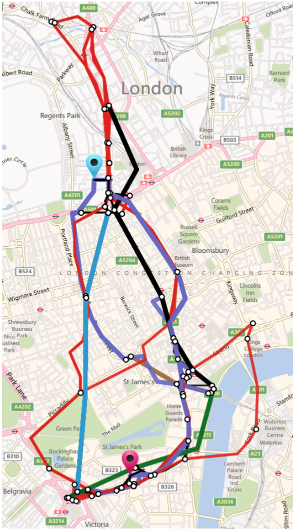
Goal: compute a *useful set* of multimodal journeys

Multiple Transportation Modes

[Delling et al.'13]

Karlsruhe Institute of Technology

- Train, Bus, Tube, Taxi, Walking, Cycling
- Optimize w.r.t. multiple criteria:
  travel time, costs, emissions,
  # of mode changes, walking duration . . .
- Pareto solution set too large

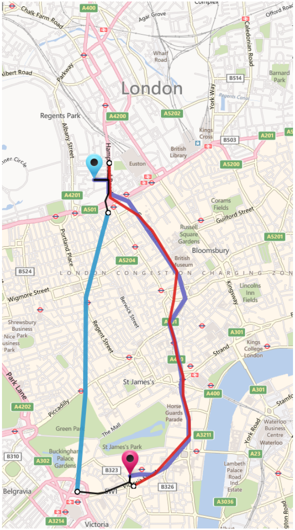


Multiple Transportation Modes

[Delling et al.'13]

- Train, Bus, Tube, Taxi, Walking, Cycling
- Optimize w.r.t. multiple criteria:
  travel time, costs, emissions,
  # of mode changes, walking duration . . .
- Pareto solution set too large

⇒ Reduce to most relevant journeys



Institute for Theoretical Informatics
Chair Algorithmics

Multiple Transportation Modes

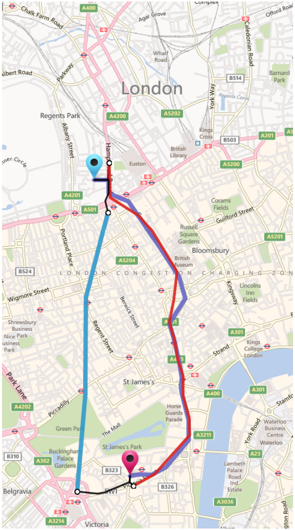
[Delling et al.'13]

- Train, Bus, Tube, Taxi, Walking, Cycling
- Optimize w.r.t. multiple criteria: travel time, costs, emissions, # of mode changes, walking duration ...
- Pareto solution set too large

⇒ Reduce to most relevant journeys

Preliminary results

- Grade by relevance
- Fuzzy filter



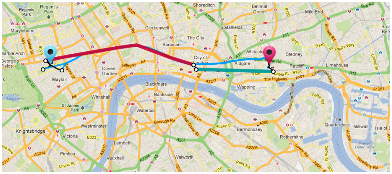
Conclusion & Outlook

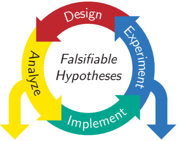
Success story for algorithm engineering

- Fast route planning on road and timetable networks
- Metric matters
- Multimodal route planning expensive

Many new challenges

- Scalability and quality in multimodal route planning
- Incorporating alternative mobility concepts
- Robustness, adjustable to unforeseen traffic situations
- Personalized route planning
- Eco-friendliness
- Autonomous driving
- Traffic control
- . . .





Thanks for your attention!