Route Planning Algorithms in Transportation Networks

7th International Network Optimization Conference

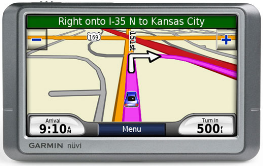
Dorothea Wagner | May 18, 2015, Warsaw, Poland

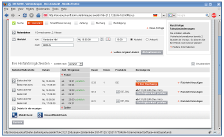
FACULTY FOR INFORMATICS · INSTITUTE FOR THEORETICAL INFORMATICS · CHAIR ALGORITHMICS

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:

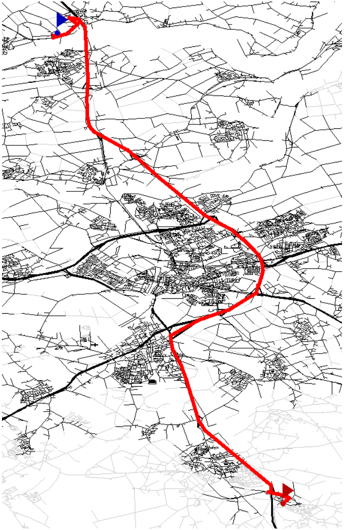
- Find the best connection in a transportation network

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)



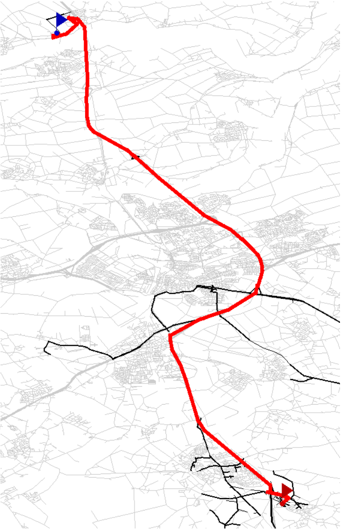
Speed-Up Techniques

Observations:

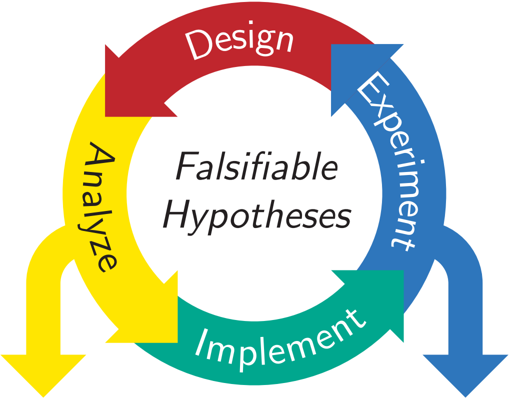
- Dijkstra visits all nodes closer than the target
- Unnecessary computations
- Many requests in a hardly changing network

Idea:

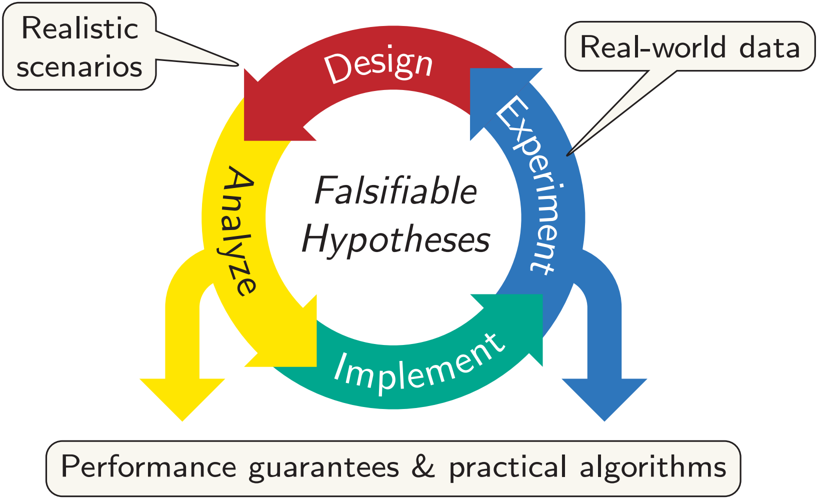
- Two-phase algorithm:
  - Offline: compute additional data during preprocessing
  - Online: speed-up query with this data
- 3 criteria: preprocessing time and space, speed-up over Dijkstra



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

Institute for Theoretical Informatics
Chair Algorithmics

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

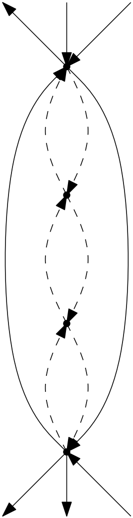
Speed-Up Techniques

Many techniques:

- Arc-Flags [Lau04]
- Multi-Level Dijkstra [SWW00, HSW08]
  - Customizable Route Planning (CRP) [DGPW11]
- ALT: A*, Landmarks, Triangle Inequality [GH05, GW05]
- Reach [GKW07]
- Contraction Hierarchies (CH) [GSSD08]
- Transit Node Routing (TNR) [ALS13]
- Hub Labeling (HL) [ADGW12]
- ...

Shortcuts

[SWW99, SS05, GSSD08]



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 to less important nodes

Institute for Theoretical Informatics
Chair Algorithmics

Contraction Hierarchies [GSSD08]

Idea: solely use contraction

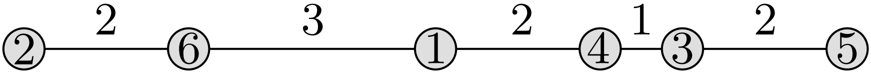
Approach:

- Heuristically order nodes by “importance”
- Contract nodes in that order
- Node v contracted by

forall the *edges* (u, v) *and* (v, w) **do**
  if (u, v, w) *unique shortest path* **then**
    add shortcut (u, w) with weight $\text{len}(u, v) + \text{len}(v, w)$;

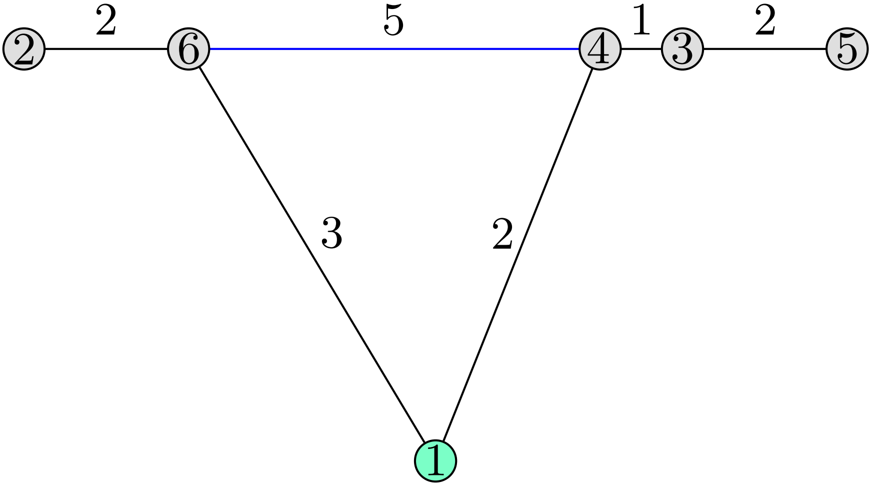
- Query only looks at edges to more important nodes

Example: CH Preprocessing

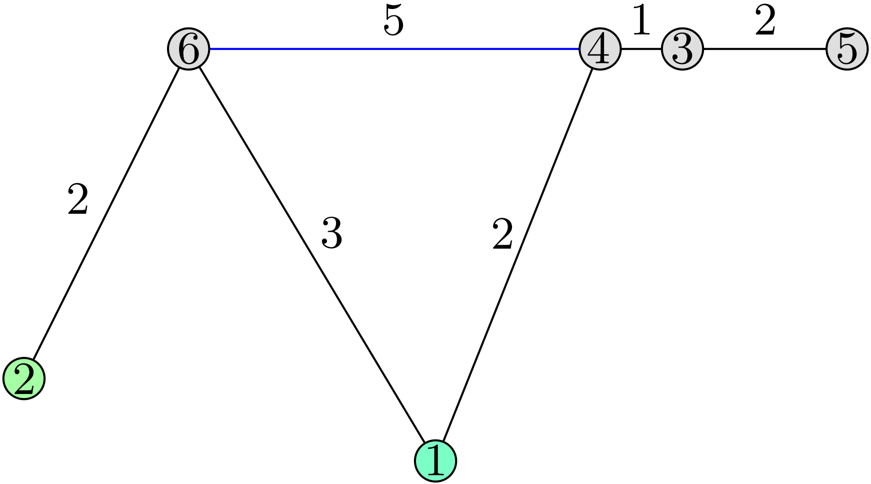


Institute for Theoretical Informatics
Chair Algorithmics

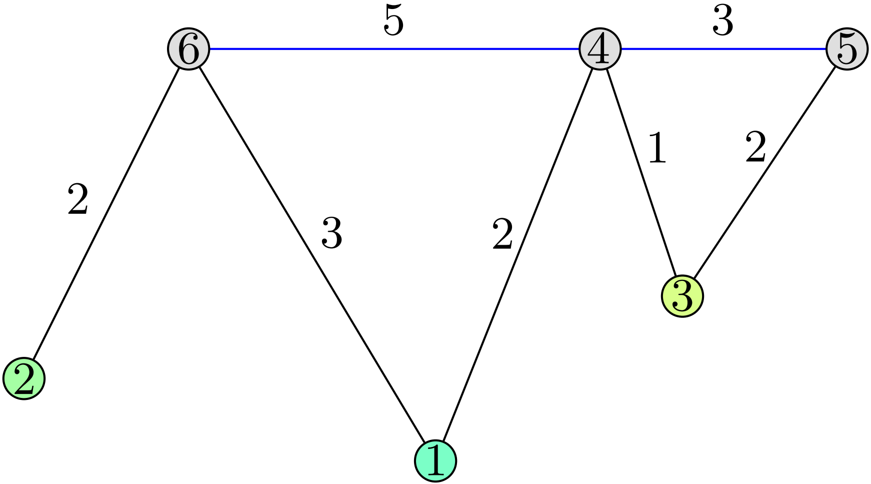
Example: CH Preprocessing



Example: CH Preprocessing

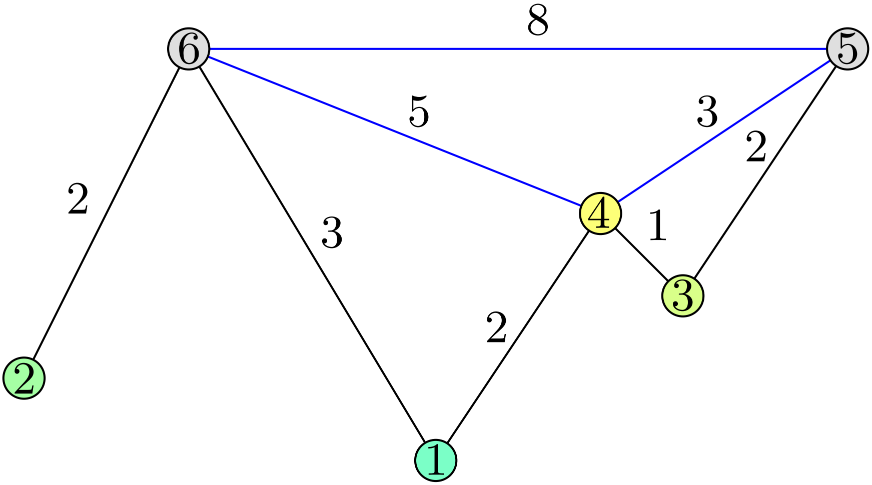


Example: CH Preprocessing

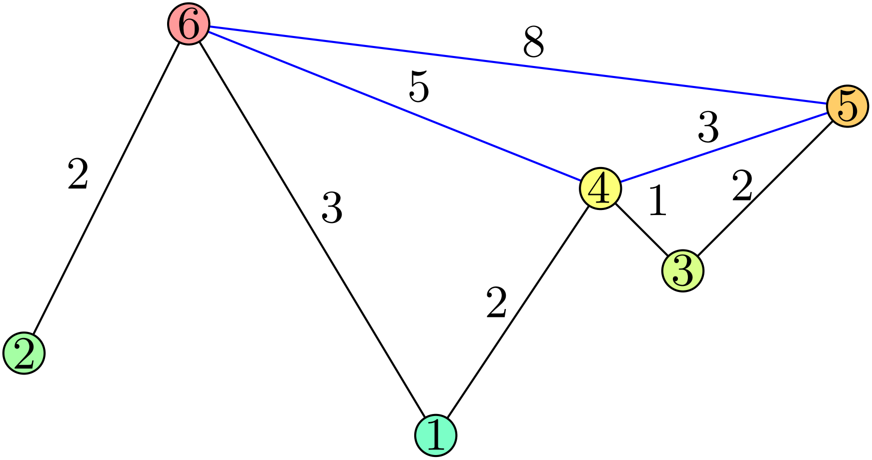


Institute for Theoretical Informatics
Chair Algorithmics

Example: CH Preprocessing

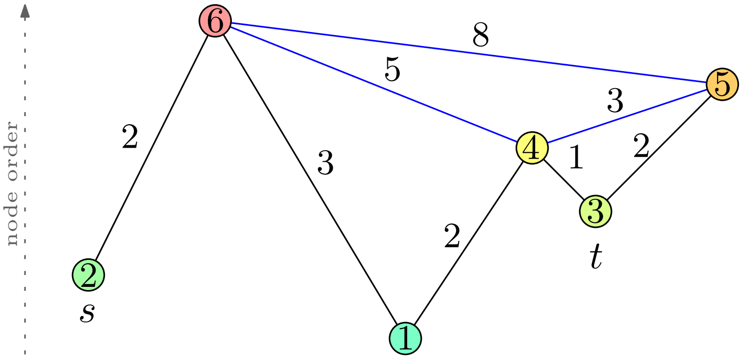


Example: CH Preprocessing



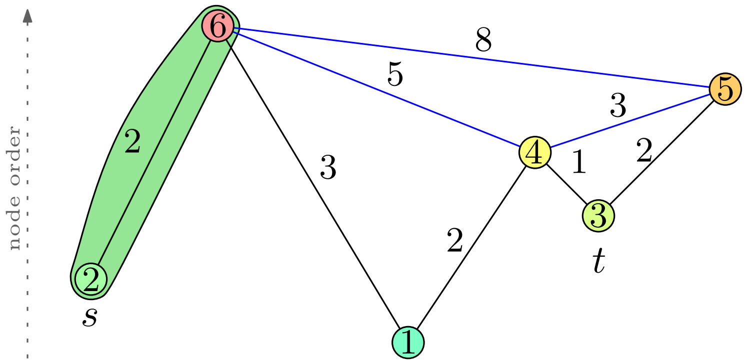
CH Query

- Modified bidirectional Dijkstra
- Upward graph $G_\uparrow := (V, E_\uparrow)$ with $E_\uparrow := \{(u, v) \in E : u < v\}$
  downward graph $G_\downarrow := (V, E_\downarrow)$ with $E_\downarrow := \{(u, v) \in E : u > v\}$
- Forward search in $G_\uparrow$ and backward search in $G_\downarrow$



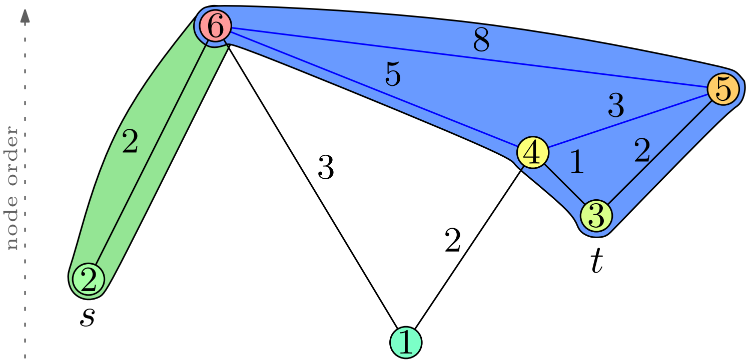
CH Query

- Modified bidirectional Dijkstra
- Upward graph $G_\uparrow := (V, E_\uparrow)$ with $E_\uparrow := \{(u,v) \in E : u < v\}$
  downward graph $G_\downarrow := (V, E_\downarrow)$ with $E_\downarrow := \{(u,v) \in E : u > v\}$
- Forward search in $G_\uparrow$ and backward search in $G_\downarrow$



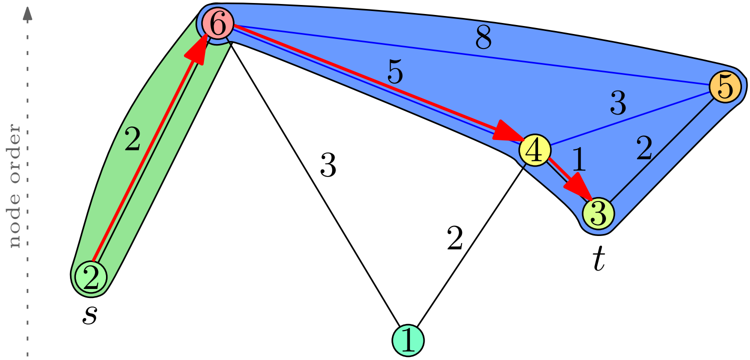
CH Query

- Modified bidirectional Dijkstra
- Upward graph $G_\uparrow := (V, E_\uparrow)$ with $E_\uparrow := \{(u,v) \in E : u < v\}$
  downward graph $G_\downarrow := (V, E_\downarrow)$ with $E_\downarrow := \{(u,v) \in E : u > v\}$
- Forward search in $G_\uparrow$ and backward search in $G_\downarrow$



CH Query

- Modified bidirectional Dijkstra
- Upward graph $G_\uparrow := (V, E_\uparrow)$ with $E_\uparrow := \{(u, v) \in E : u < v\}$
  downward graph $G_\downarrow := (V, E_\downarrow)$ with $E_\downarrow := \{(u, v) \in E : u > v\}$
- Forward search in $G_\uparrow$ and backward search in $G_\downarrow$



From Practice to Theory: WeakCH

Question: What is a good contraction order?

- No guarantees on search space [GSSD08]

Institute for Theoretical Informatics
Chair Algorithmics

From Practice to Theory: WeakCH

Question: What is a good contraction order?

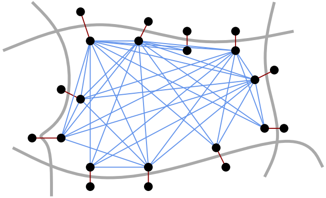
- No guarantees on search space [GSSD08]

WeakCH [BCRW13]

- Balanced separator nodes are important
  $\rightarrow$ resulting CH is called *weak*
- $O(n^\alpha)$ separators $\rightarrow$ $O(n^\alpha)$ nodes in the search space
- Order is independent of metric

(Multi-Level) Overlays [SWW00, HSW08]

Observation: many (long-distance) paths share large subpaths
Idea: precompute partial solutions

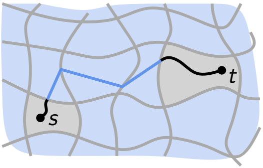


Overlay graph:

- Select important nodes (separators, path coverage, heuristic)
- Compute shortcut-edges:
  - Skip unimportant nodes
  - Conserve distances to important nodes

Queries:

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

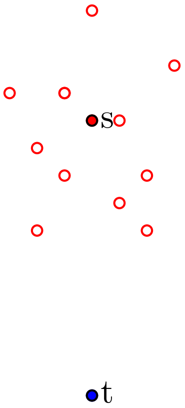


analogous: hierarchies with several levels of nodes of varying importances

Hub Labeling

Preprocessing:

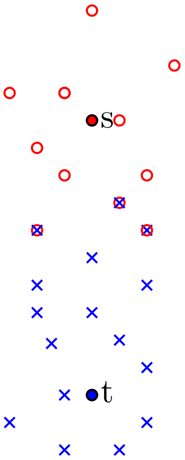
- For each node u, compute label $L(u)$
  - A set of hub nodes v and their distance $\mathrm{dist}(u, v)$ to u



Hub Labeling

Preprocessing:

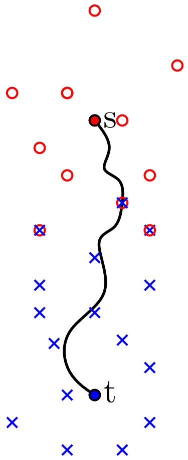
- For each node u, compute label $L(u)$
  - A set of hub nodes v and their distance $\mathrm{dist}(u, v)$ to u
- Labels must fulfill cover property:
  for every s, t-pair, the shortest path goes through the intersection of $L(s) \cap L(t)$



Hub Labeling

Preprocessing:

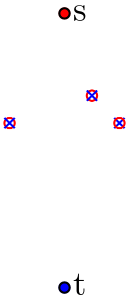
- For each node u, compute label $L(u)$
  - A set of hub nodes v and their distance $\mathrm{dist}(u, v)$ to u
- Labels must fulfill cover property:
  for every s, t-pair, the shortest path goes through the intersection of $L(s) \cap L(t)$



Hub Labeling

Preprocessing:

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



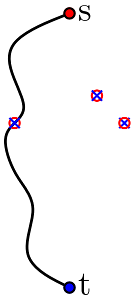
***s*–*t* query:**

- 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 $\mathrm{dist}(u, v)$ to u
- Labels must fulfill cover property:
  for every s, t-pair, the shortest path goes through
  the intersection of $L(s) \cap L(t)$



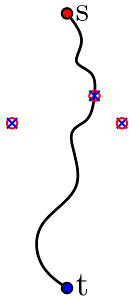
***s*–*t* query:**

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

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:**

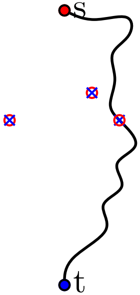
- Find node $v \in L(s) \cap L(t) \ldots$
- $\ldots$ that minimizes $\text{dist}(s, v) + \text{dist}(v, t)$

Institute for Theoretical Informatics
Chair Algorithmics

Hub Labeling

Preprocessing:

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



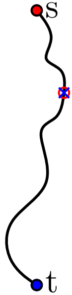
***s*–*t* query:**

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

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:**

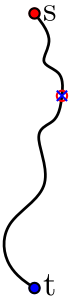
- Find node $v \in L(s) \cap L(t) \ldots$
- $\ldots$ that minimizes $\text{dist}(s, v) + \text{dist}(v, t)$

Institute for Theoretical Informatics
Chair Algorithmics

Hub Labeling

Preprocessing:

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



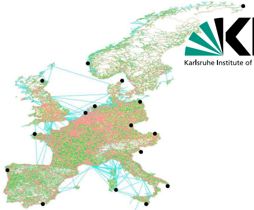
***s*–*t* query:**

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

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

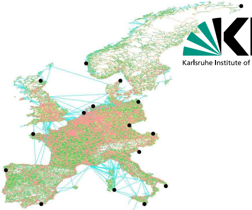


Input: Road network of Europe

- 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”

$\Rightarrow$ Consider energy consumption in route planning applications

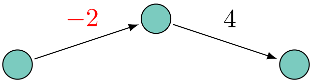
Task: Given start and destination in a road network,
find the route that minimizes energy consumption.

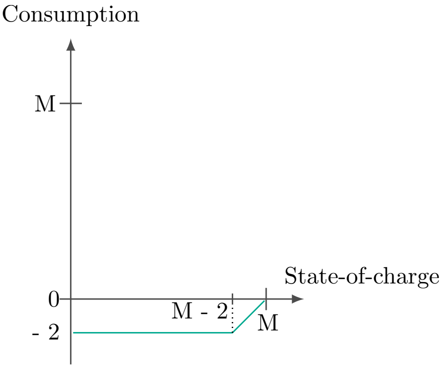
Energy-Optimal Routes

Challenges:

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

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



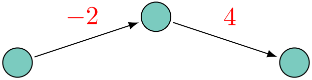


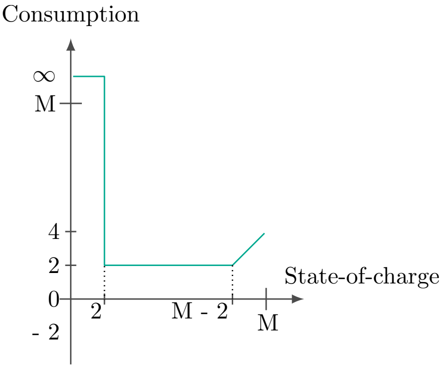
Energy-Optimal Routes

Challenges:

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

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



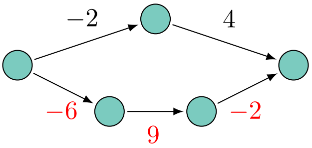


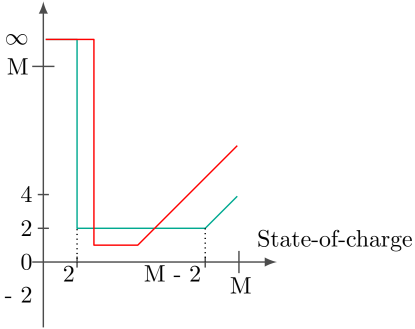
Energy-Optimal Routes

Challenges:

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

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





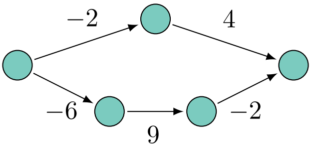
Institute for Theoretical Informatics
Chair Algorithmics

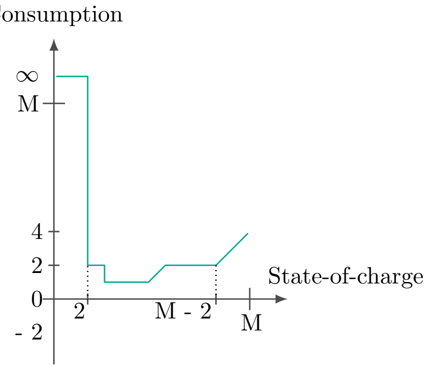
Energy-Optimal Routes

Challenges:

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

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





Energy-Optimal Routes [BDPW13, BDHS+14]

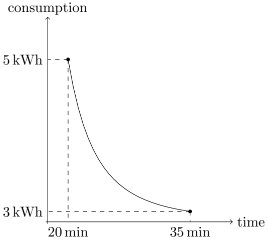
Requirements for speedup techniques:

- 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



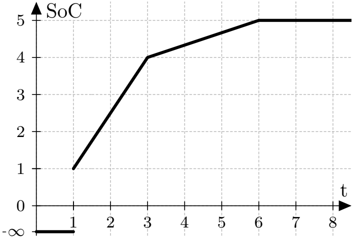
Trading travel time for energy consumption:

- Consider constrained paths
  - E.g., find the fastest path such that the battery does not run out
  - $\mathcal{NP}$-hard
- Energy can be saved driving below speed limit
- Additional instructions to the driver

Including Charging Stops

Task: Find the fastest path such that the battery does not run out.

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



Approach:

- Extension of bicriteria search
- Propagates charging functions
- CHArge: Combination with CH and A*
  - Optimal routes in seconds / minutes
- Heuristic approaches (based on CHArge)
  - Near-optimal solutions in well below a second

Custom Metrics

Karlsruhe Institute of Technology

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

From Theory to Practice: Customizable Contraction Hierarchies [DSW14]

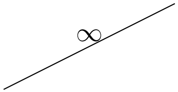
Idea:

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

From Theory to Practice: Customizable Contraction Hierarchies [DSW14]

Idea:

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

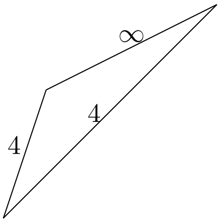


an edge in the CH

From Theory to Practice: Customizable Contraction Hierarchies [DSW14]

Idea:

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

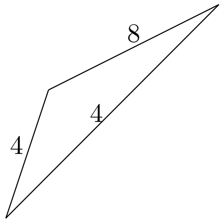


establish lower triangle inequality

From Theory to Practice: Customizable Contraction Hierarchies [DSW14]

Idea:

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



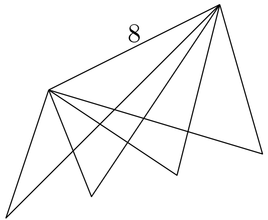
establish lower triangle inequality

From Theory to Practice: Customizable Contraction Hierarchies [DSW14]

Karlsruhe Institute of Technology

Idea:

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



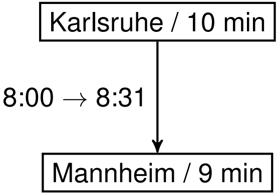
do this for all lower triangles

What is a Timetable?

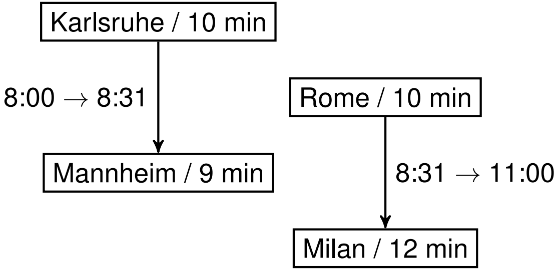
Karlsruhe / 10 min

Institute for Theoretical Informatics
Chair Algorithmics

What is a Timetable?

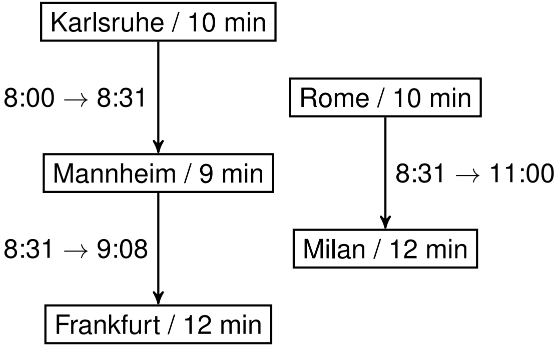


What is a Timetable?



Institute for Theoretical Informatics
Chair Algorithmics

What is a Timetable?



Existing Approaches

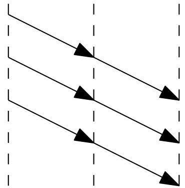
list of connections and stops

query

Institute for Theoretical Informatics
Chair Algorithmics

Existing Approaches

list of connections and stops



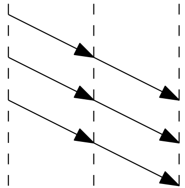
Time Expanded
[PSWZ08]

query

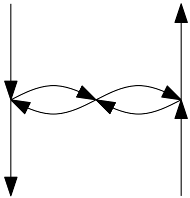
Institute for Theoretical Informatics
Chair Algorithmics

Existing Approaches

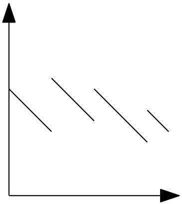
list of connections and stops



Time Expanded
[PSWZ08]



Time Dependent
[PSWZ08]



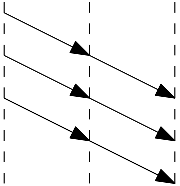
complex arc weights

query

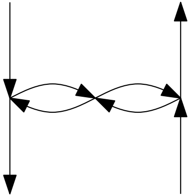
Institute for Theoretical Informatics
Chair Algorithmics

Existing Approaches

list of connections and stops



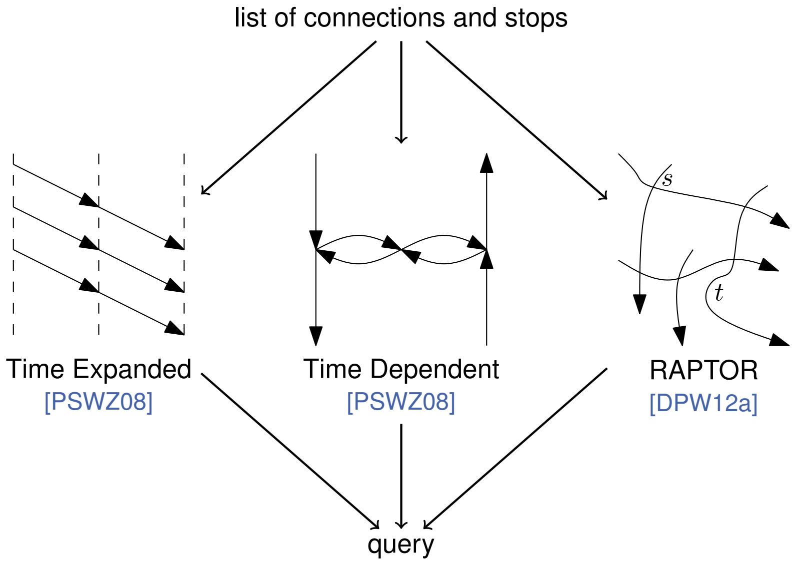
Time Expanded
[PSWZ08]



Time Dependent
[PSWZ08]

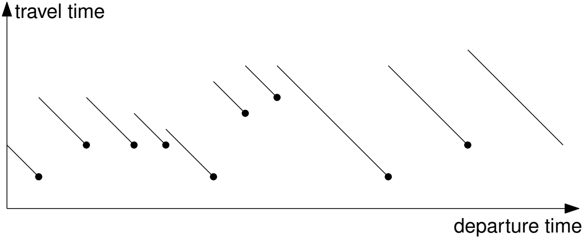
query

Existing Approaches



Timetable Queries

- Inherently time-dependent: discrete departure times
- More query scenarios:
  - Depart now: earliest arrival time?
  - Depart later: shortest travel time?
  - Profile queries: set of journeys with varying departure times
  - Multicriteria: number of transfers, price, ...
- Different network structure: less hierarchical, less well-separated, very different schedules at night, ...



Connection Scan (CSA) [DPSW13]

Output: earliest arrival time
Input: timetable, source stop, source time, target stop

stop ID		0	1	2	3	4	
earliest arrival time	$\cdots$	$+\infty$	$+\infty$	$+\infty$	$+\infty$	$+\infty$	$\cdots$

elementary connections ordered by departure time	$\cdots$	dep.stop	arr.stop	dep.time	arr.time		dep.stop	arr.stop	dep.time	arr.time		dep.stop	arr.stop	dep.time	arr.time		$\cdots$

*missing in the example: footpaths and minimum change times

Institute for Theoretical Informatics
Chair Algorithmics

Connection Scan (CSA) [DPSW13]

Output: earliest arrival time
Input: timetable, source stop, source time, target stop

stop ID		0	1	2	3	4	
earliest arrival time	…	$+\infty$	$+\infty$	$+\infty$	$+\infty$	$+\infty$	…

elementary connections ordered by departure time				
…				
dep: 1	arr: 3	9:00	9:25	
dep: 3	arr: 4	9:15	9:45	
dep: 3	arr: 4	9:40	9:55	
…				

*missing in the example: footpaths and minimum change times

Institute for Theoretical Informatics
Chair Algorithmics

Connection Scan (CSA) [DPSW13]

Output: earliest arrival time
Input: timetable, source stop, source time, target stop

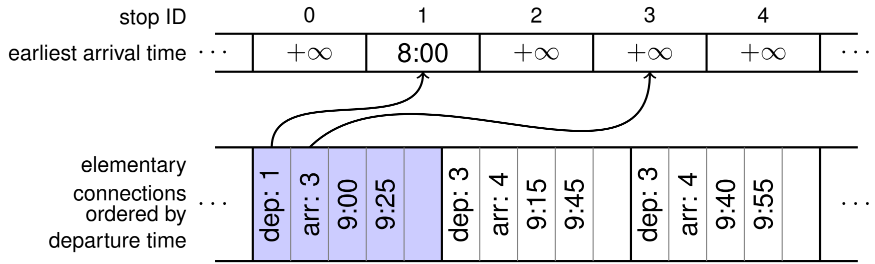
stop ID		0	1	2	3	4	
earliest arrival time	…	$+\infty$	8:00	$+\infty$	$+\infty$	$+\infty$	…

elementary connections ordered by departure time				
…				
dep: 1	arr: 3	9:00	9:25	
dep: 3	arr: 4	9:15	9:45	
dep: 3	arr: 4	9:40	9:55	
…				

*missing in the example: footpaths and minimum change times

Connection Scan (CSA) [DPSW13]

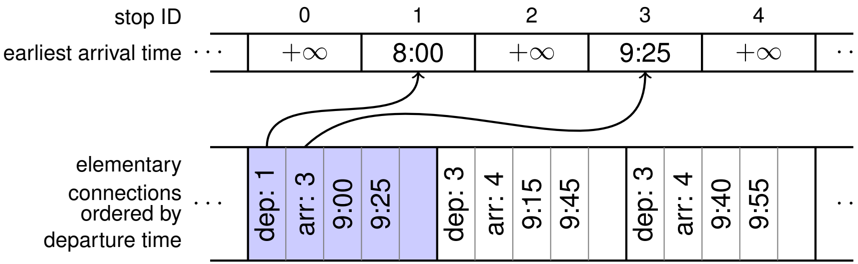
Output: earliest arrival time
Input: timetable, source stop, source time, target stop



*missing in the example: footpaths and minimum change times

Connection Scan (CSA) [DPSW13]

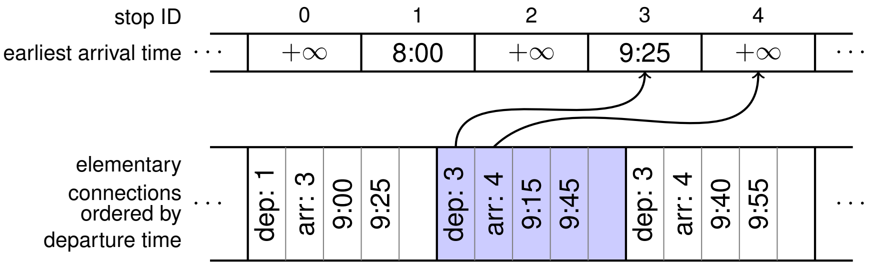
Output: earliest arrival time
Input: timetable, source stop, source time, target stop



*missing in the example: footpaths and minimum change times

Connection Scan (CSA) [DPSW13]

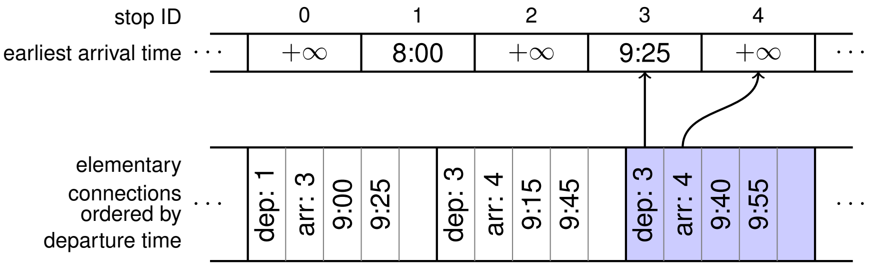
Output: earliest arrival time
Input: timetable, source stop, source time, target stop



*missing in the example: footpaths and minimum change times

Connection Scan (CSA) [DPSW13]

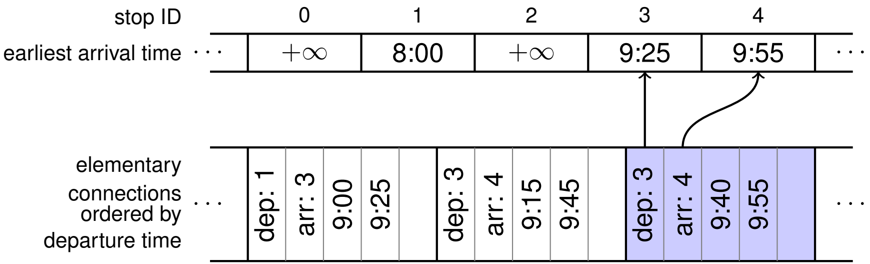
Output: earliest arrival time
Input: timetable, source stop, source time, target stop



*missing in the example: footpaths and minimum change times

Connection Scan (CSA) [DPSW13]

Output: earliest arrival time
Input: timetable, source stop, source time, target stop



*missing in the example: footpaths and minimum change times

Connection Scan (CSA) [DPSW13]

Output: earliest arrival time
Input: timetable, source stop, source time, target stop

stop ID		0	1	2	3	4	
earliest arrival time	...	$+\infty$	8:00	$+\infty$	9:25	9:55	...

elementary connections ordered by departure time																
...	dep: 1	arr: 3	9:00	9:25		dep: 3	arr: 4	9:15	9:45		dep: 3	arr: 4	9:40	9:55		...

*missing in the example: footpaths and minimum change times

time table graph is a DAG
faster than Dijkstra, better use of modern processor architectures

Experimental Evaluation

Karlsruhe Institute of Technology

Input: timetable

- London: 5 M connections, 21 k stops
- Germany: 46 M connections, 252 k stops

	Algorithm	Time [ms]	speed-up.
London	TE Dijkstra [PSWZ08]	44.8	—
	TD Dijkstra [PSWZ08]	10.9	4.1
	CSA [DPSW13]	1.8	24.9
DE	TE Dijkstra [PSWZ08]	2960.2	—
	CSA [DPSW13]	298.6	9.9
	CSAccel [SW14]	8.7*	340.2

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

*preprocessing: 30 min, 256.4 MiB

Navigation Device for the World

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

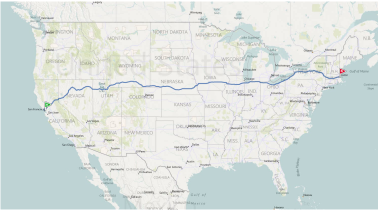
Multimodal Routing

Up to now:

- 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



Adapting Speed-Up Techniques

[Paj09], [KLPC11]

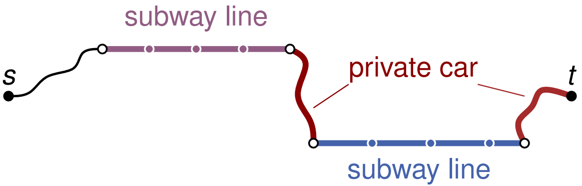
- **Bidirectional search**
  easily adaptable (time-dependency is hard)
- **Goal-directed search**
  ALT adaptable but low speed-ups,
  Arc-Flags turns out difficult
- **Contraction**
  adaptable with some restrictions
  - Contracted graph is called the **Core**

two promising approaches:

- **Access-node routing (ANR)**
  adapting ideas from transit-node routing (table lookups)
- **User-constrained CH (UCCH)**
  augmenting contraction hierarchies

Multiple Transportation Modes

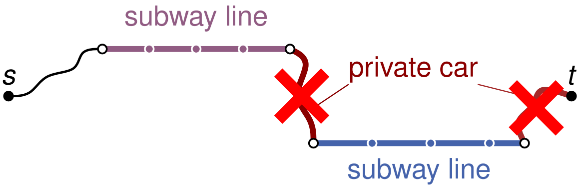
Problem: Unrestricted journeys allow arbitrary transfers



Institute for Theoretical Informatics
Chair Algorithmics

Multiple Transportation Modes

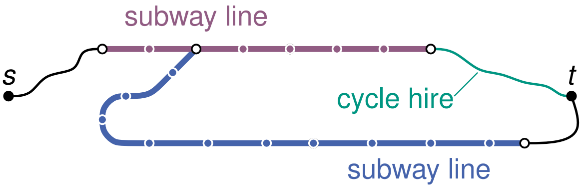
Problem: Unrestricted journeys allow arbitrary transfers



- Not all sequences of transportation modes are reasonable

Multiple Transportation Modes

Problem: Unrestricted journeys allow arbitrary transfers

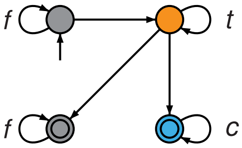


- Not all sequences of transportation modes are reasonable
- Preferred mode of transport varies between users

Solution

"Label Constrained Shortest Path Problem" (LCSPP)

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

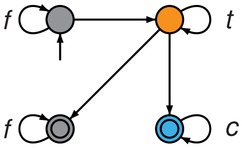


Institute for Theoretical Informatics
Chair Algorithmics

Solution

"Label Constrained Shortest Path Problem" (LCSPP)

- Define alphabet of transportation mode
- Finite-state automaton describes sequences of vehicles
- 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 [BJM00]
- Speed-up techniques: ANR [DPW09], SDALT [KLPC11]
- Automaton as input during the query: UCCH [DPW12b]

User-constrained CH (UCCH) [DPW12b]

Multimodal CH:

- Contraction introduces shortcuts with label sequences
- Witness search depends on constraints
  requires a-priori knowledge of the constraint automata

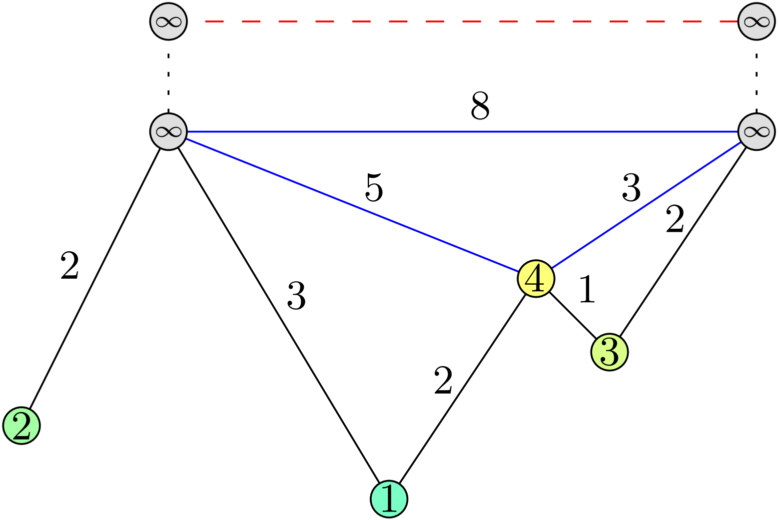
Idea: do not contract nodes with incident link-edges.

- Contraction and witness search are limited to each modality

$\Rightarrow$ Preprocessing independent of mode sequence constraints

Institute for Theoretical Informatics
Chair Algorithmics

Example: UCCH Preprocessing



UCCH

Karlsruhe Institute of Technology

Preprocessing

- Linked nodes are not contracted thus contained in the core
- Shortcuts between core nodes preserve distances
  allows using the road network between rail stations

Query

- CH search on the component
- Label constrained search on the core
- Engineering yields further improvement

Experimental Evaluation

Networks:
road: europe & north america (50 M nodes, 125 M edges)
train: europe (31 k stops, 1.6 M connections)
flight: Star Alliance (1 172 airports, 28 k connections)

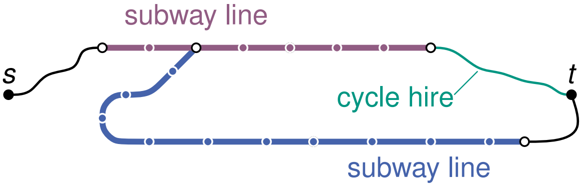
		Preprocessing		Query	
	Algorithm	Time [h:m]	Space [MiB]	Time [ms]	Speedup
road & flight	Dijkstra	—	—	33 862	1
	ANR [DPW09]	3:04	14 050	1.07	31 551
	UCCH [DPW12b]	1:18	542	0.67	50 540
all three	Dijkstra	—	—	35 261	1
	ANR [DPW09]	—	—	—	—
	UCCH [DPW12b]	1:27	558	70.52	500

Intel Xeon E5430, 2.66 GHz, 32 GiB RAM, 12 MiB L2 cache

Institute for Theoretical Informatics
Chair Algorithmics

Solution?

Problems of LCSPP



Solution?

Problems of LCSPP

Solution?

Problems of LCSPP

s ? t

- 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

Institute for Theoretical Informatics
Chair Algorithmics

Multicriteria Multimodal Routing [DDP+13]

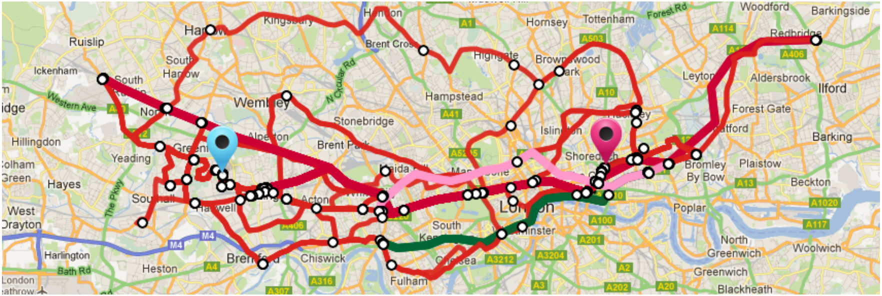
Idea: compute multicriteria, multimodal Pareto sets

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

Multicriteria Multimodal Routing [DDP+13]

Idea: compute multicriteria, multimodal Pareto sets

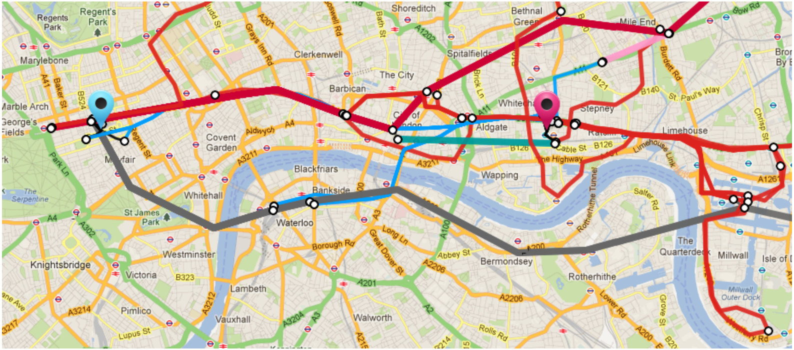
- Optimize arrival time plus
- Various (per mode of transport) „convenience criteria“
  for example # transfers (trains), walking time, taxi costs, etc.



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

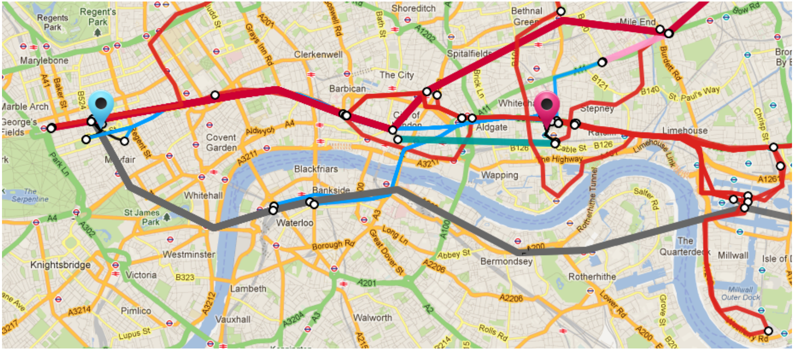
Known problem: Pareto set sizes explode in the number of criteria

Relevant Journeys



- 10 min of walking to arrive 10 sec earlier?
- 1 hour of bus drive to walk 10 sec less?

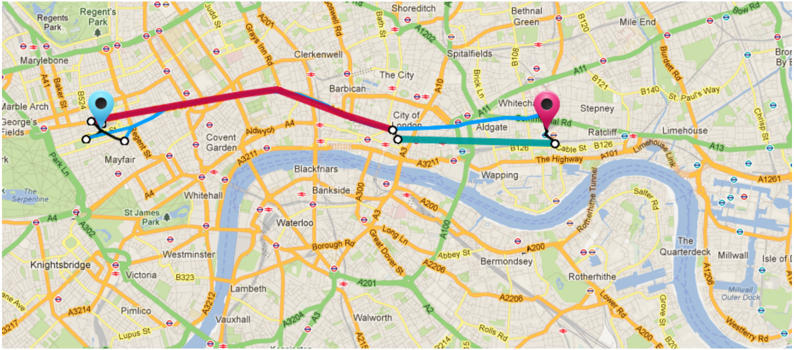
Relevant Journeys



- 10 min of walking to arrive 10 sec earlier?
- 1 hour of bus drive to walk 10 sec less?
- Rate the journeys using fuzzy logic [FA04]
- Journeys with a higher rating are more relevant

Relevant Journeys

Karlsruhe Institute of Technology



the three top-rated journeys.

- 10 min of walking to arrive 10 sec earlier?
- 1 hour of bus drive to walk 10 sec less?
- Rate the journeys using fuzzy logic [FA04]
- 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-tx-ry: max x minutes of walking between vehicles and max. y at source/target

Reduce the dimension/number of criteria

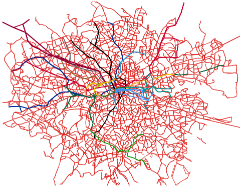
- MR-x: increase for every x minutes of walking the #transfers by +1

Institute for Theoretical Informatics
Chair Algorithmics

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

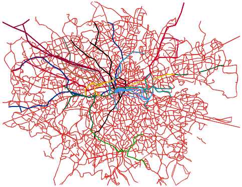
Algorithm	# Sol.	Time [ms]	Quality-6 Avg.	Quality-6 Sd.
MCR	29.1	1 438.7	100 %	0 %
MCR-hf	10.9	699.4	89 %	11 %
MCR-hb	9.0	456.7	91 %	10 %
MCR-t10-r15	13.2	885.0	30 %	31 %
MR-10	4.3	39.4	45 %	29 %

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

			Quality-6	
Algorithm	# Sol.	Time [ms]	Avg.	Sd.
MCR	29.1	1 438.7	100 %	0 %
MCR-hf	10.9	699.4	89 %	11 %
MCR-hb	9.0	456.7	91 %	10 %
MCR-t10-r15	13.2	885.0	30 %	31 %
MR-10	4.3	39.4	45 %	29 %

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

Conclusion

Karlsruhe Institute of Technology

Summary

- Algorithm Engineering: combination of theory and practice
- (Very) fast route planning on road and timetable networks
- Considered metric matters
- Multimodal route planning is more expensive
  - Network offers many interesting trade-offs between criteria
  - Multicriteria optimization useful, to allow the user to chose his journey

Outlook

- Formalization of quality for multimodal journeys done?
- Scalability: multimodal multicriteria for worldwide routing?
- Additional questions: delay-robustness, park & ride, . . . ?

Thank you for your attention!

Institute for Theoretical Informatics
Chair Algorithmics

Bibliography 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.

Reinhard Bauer, Tobias Columbus, Ignaz Rutter, and Dorothea Wagner.
Search-space size in contraction hierarchies.
In *Proceedings of the 40th International Colloquium on Automata, Languages, and Programming (ICALP'13)*, volume 7965 of *Lecture Notes in Computer Science*, pages 93–104. 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.

Moritz Baum, Julian Dibbelt, Lorenz Hübschle-Schneider, Thomas Pajor, and Dorothea Wagner.
Speed-consumption tradeoff for electric vehicle route planning.
In *Proceedings of the 14th Workshop on Algorithmic Approaches for Transportation Modeling, Optimization, and Systems (ATMOS'14)*, OpenAccess Series in Informatics (OASIcs), pages 138–151, 2014.

Moritz Baum, Julian Dibbelt, Thomas Pajor, and Dorothea Wagner.
Energy-optimal routes for electric vehicles.
In *Proceedings of the 21st ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 54–63. ACM Press, 2013.

Bibliography II

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.

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

Julian Dibbelt, Thomas Pajor, Ben Strasser, and Dorothea Wagner.
Intriguingly simple and fast transit routing.
In *Proceedings of the 12th International Symposium on Experimental Algorithms (SEA'13)*, volume 7933 of *Lecture Notes in Computer Science*, pages 43–54. Springer, 2013.

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.

Bibliography III

Daniel Delling, Thomas Pajor, and Renato F. Werneck.
Round-based public transit routing.
In *Proceedings of the 14th Meeting on Algorithm Engineering and Experiments (ALENEX'12)*, pages 130–140. SIAM, 2012.

Julian Dibbelt, Thomas Pajor, and Dorothea Wagner.
User-constrained multi-modal route planning.
In *Proceedings of the 14th Meeting on Algorithm Engineering and Experiments (ALENEX'12)*, pages 118–129. SIAM, 2012.

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.

Andrew V. Goldberg, Haim Kaplan, and Renato F. Werneck.
Better landmarks within reach.
In *Proceedings of the 6th Workshop on Experimental Algorithms (WEA'07)*, volume 4525 of *Lecture Notes in Computer Science*, pages 38–51. Springer, June 2007.

Bibliography IV

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.

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.

Thomas Pajor.
Multi-modal route planning.
Master's thesis, Universität Karlsruhe (TH), March 2009.

Evangelia Pyrga, Frank Schulz, Dorothea Wagner, and Christos Zaroliagis.
Efficient models for timetable information in public transportation systems.
ACM Journal of Experimental Algorithmics, 12(2.4):1–39, 2008.

Bibliography V

Peter Sanders and Dominik Schultes.
Highway hierarchies hasten exact shortest path queries.
In *Proceedings of the 13th Annual European Symposium on Algorithms (ESA'05)*, volume 3669 of *Lecture Notes in Computer Science*, pages 568–579. Springer, 2005.

Ben Strasser and Dorothea Wagner.
Connection scan accelerated.
In *Proceedings of the 16th Meeting on Algorithm Engineering and Experiments (ALENEX'14)*, pages 125–137. SIAM, 2014.

Frank Schulz, Dorothea Wagner, and Karsten Weihe.
Dijkstra's algorithm on-line: An empirical case study from public railroad transport.
In *Proceedings of the 3rd International Workshop on Algorithm Engineering (WAE'99)*, volume 1668 of *Lecture Notes in Computer Science*, pages 110–123. Springer, 1999.

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.