Karlsruhe Institute of Technology

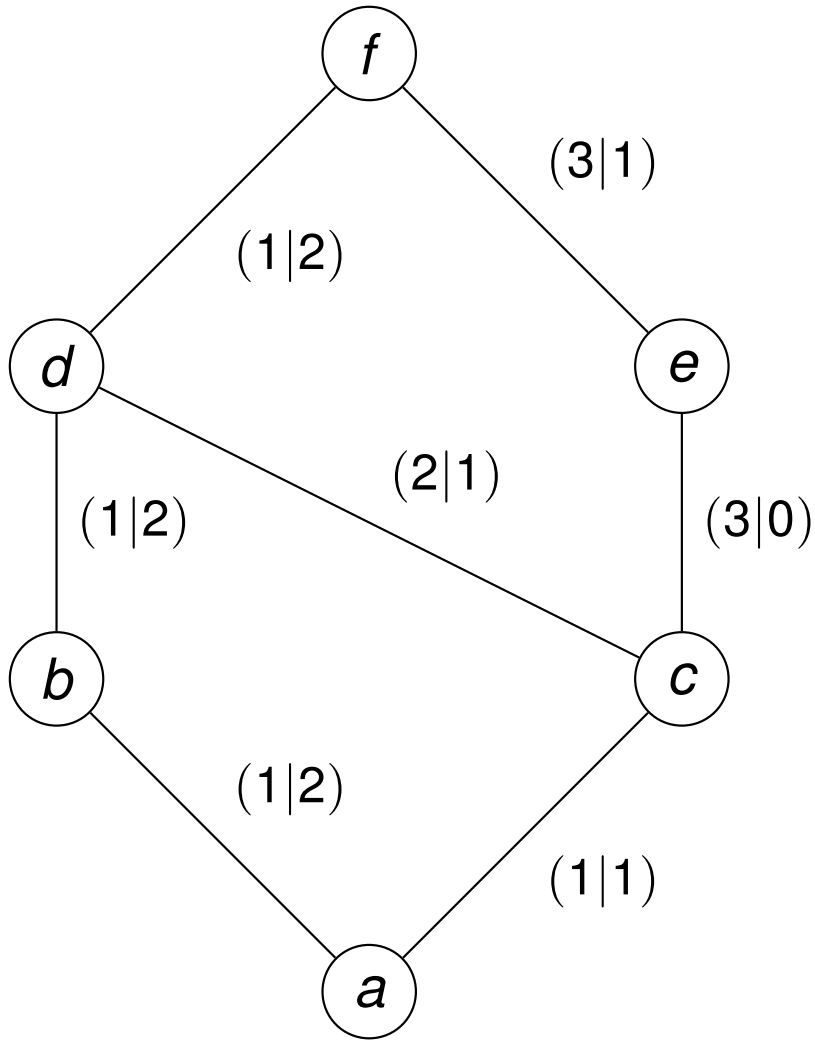
# Flexible Route Planning with Contraction Hierarchies

Robert Geisberger, **Moritz Kobitzsch** and Peter Sanders –
*{geisberger,kobitzsch,sanders}@kit.edu*

Institute for Theoretical Computer Science, Algorithmics II

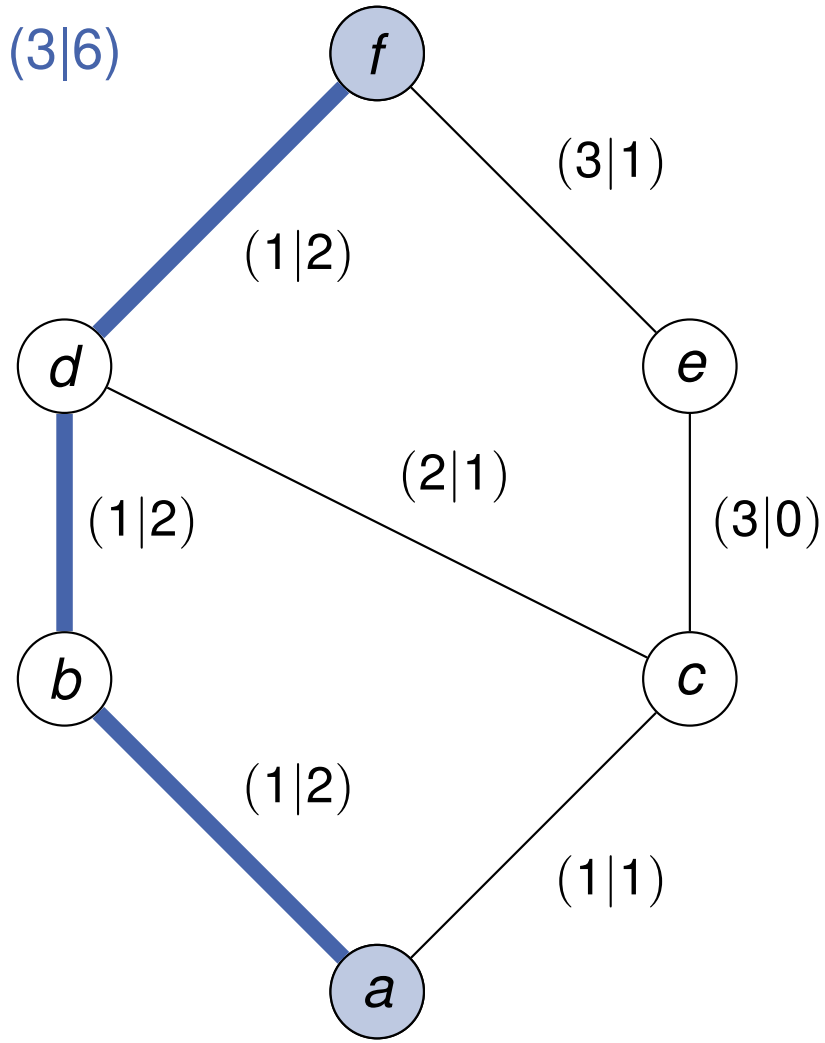
# Motivation

- route planning becomes ubiquitous
- route planning services e. g. Google Maps
- wide spread generates wide variety of interests
- multiple optimization criterias possible (e.g. (time|cost) )



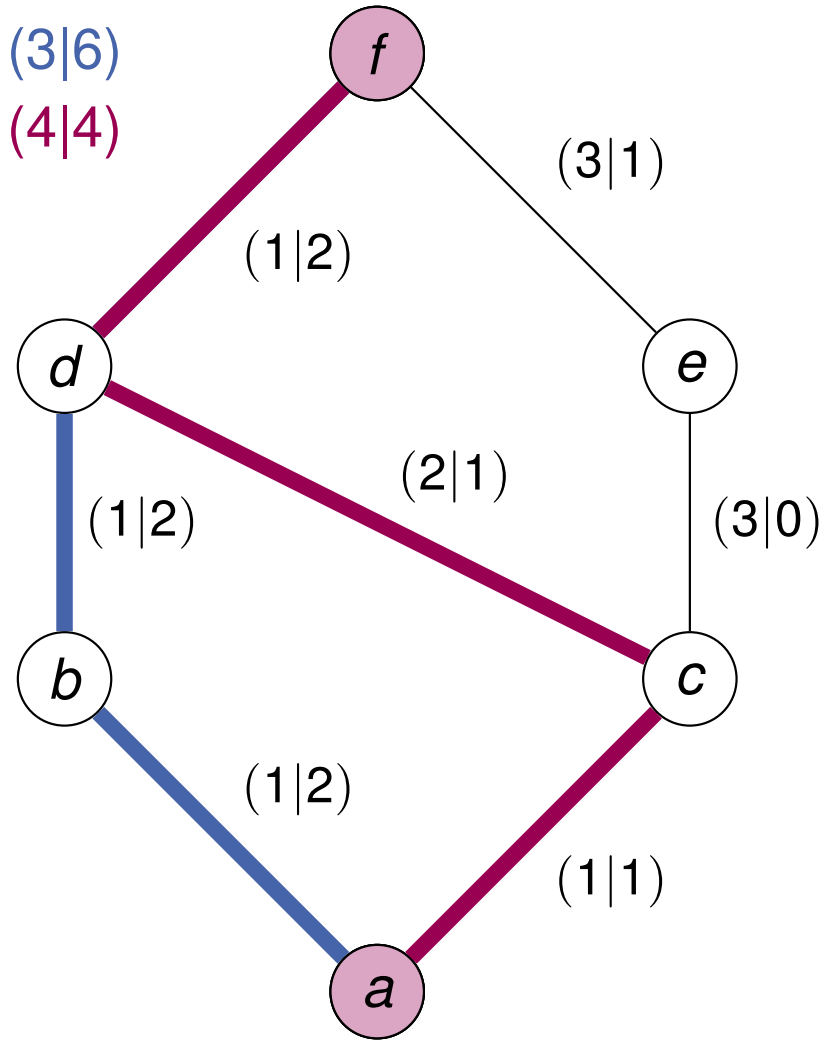
# Motivation

- route planning becomes ubiquitous
- route planning services e. g. Google Maps
- wide spread generates wide variety of interests
- multiple optimization criterias possible (e.g. (time|cost) )



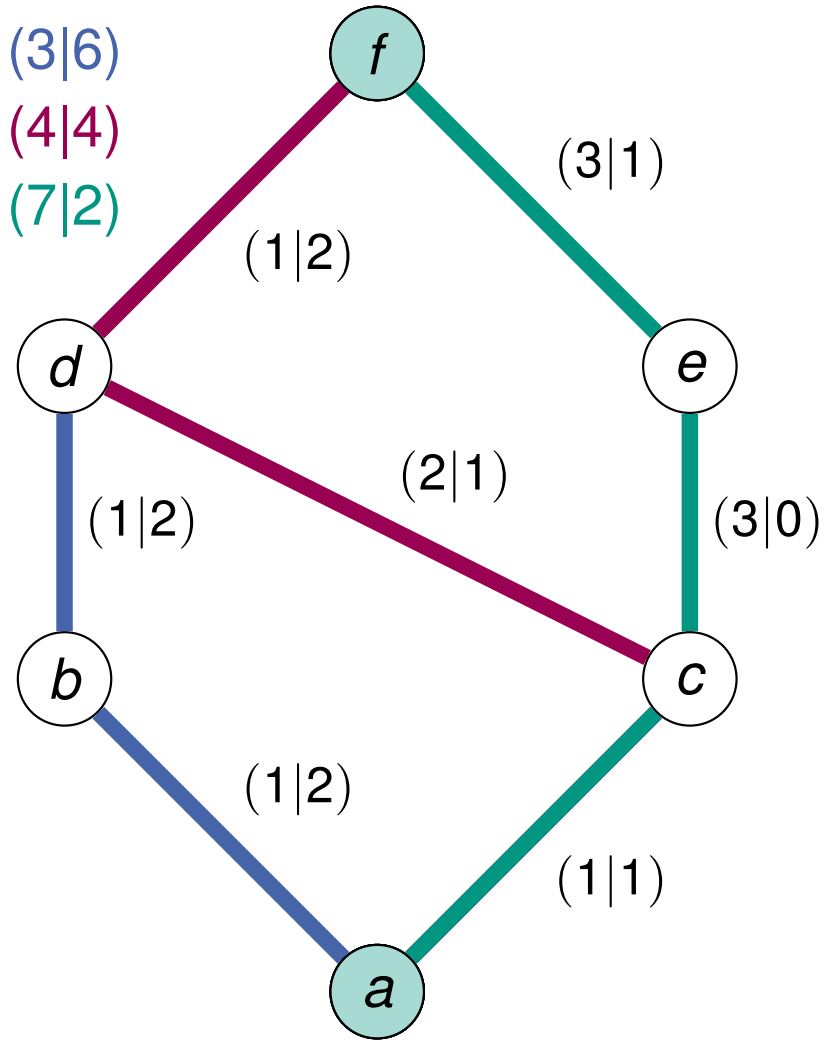
# Motivation

- route planning becomes ubiquitous
- route planning services e. g. Google Maps
- wide spread generates wide variety of interests
- multiple optimization criterias possible (e.g. (time|cost) )



# Motivation

- route planning becomes ubiquitous
- route planning services e. g. Google Maps
- wide spread generates wide variety of interests
- multiple optimization criterias possible (e.g. (time|cost) )



# Just imagine...

Google maps austin airport to hyatt regency Search Maps Show search options

Find businesses, addresses and places of interest.

Get Directions My Maps

austin airport

208 Barton Springs Road, Austin, TX 78704 (Hya

Add Destination - Show options

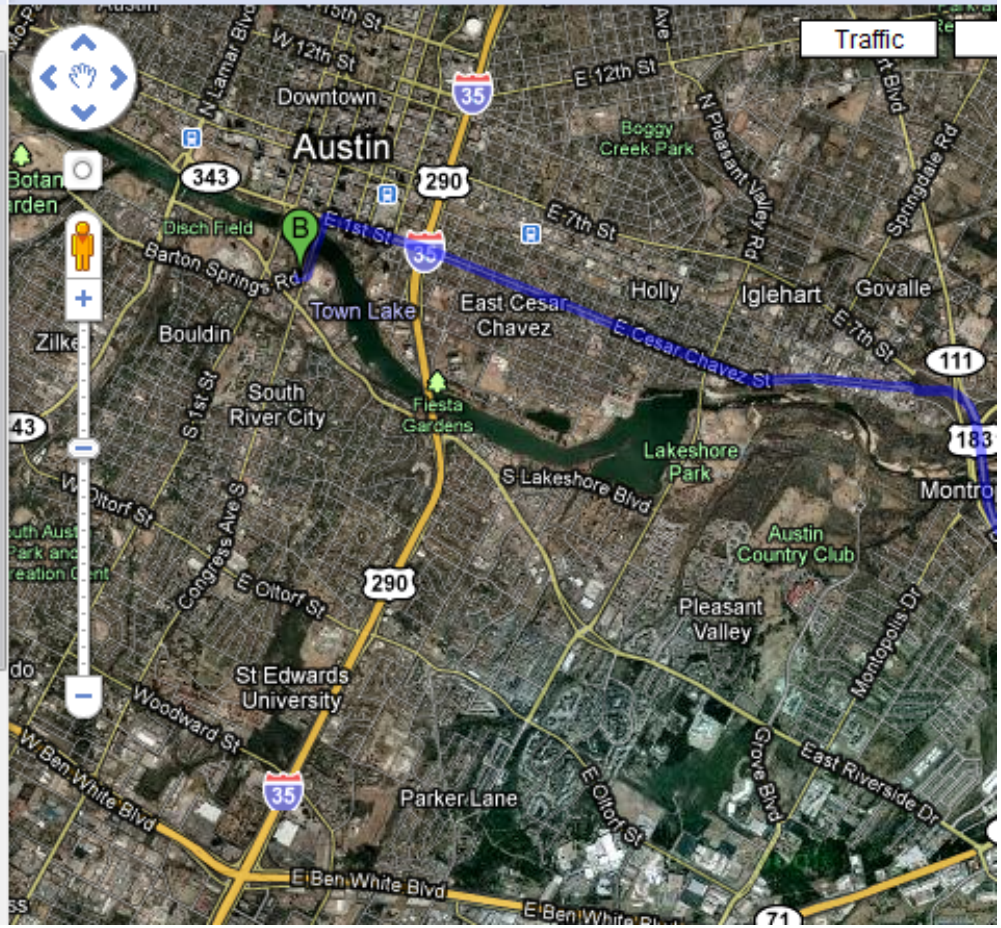
By car Get Directions

Also available: Public Transit Walking

fast cheap

Austin-Bergstrom International Airport
Austin, TX 78719

1. Head **east** on **Presidential Blvd** 0.6 mi
2. Slight **right** to stay on **Presidential Blvd** 0.5 mi
3. Turn **left** at **E State Hwy 71 Service Rd** 0.2 mi
4. Take the ramp on the **left** onto **Bastrop Hwy/TX-71 W** 1.1 mi
5. Take the ramp onto **US-183 N** 1.9 mi



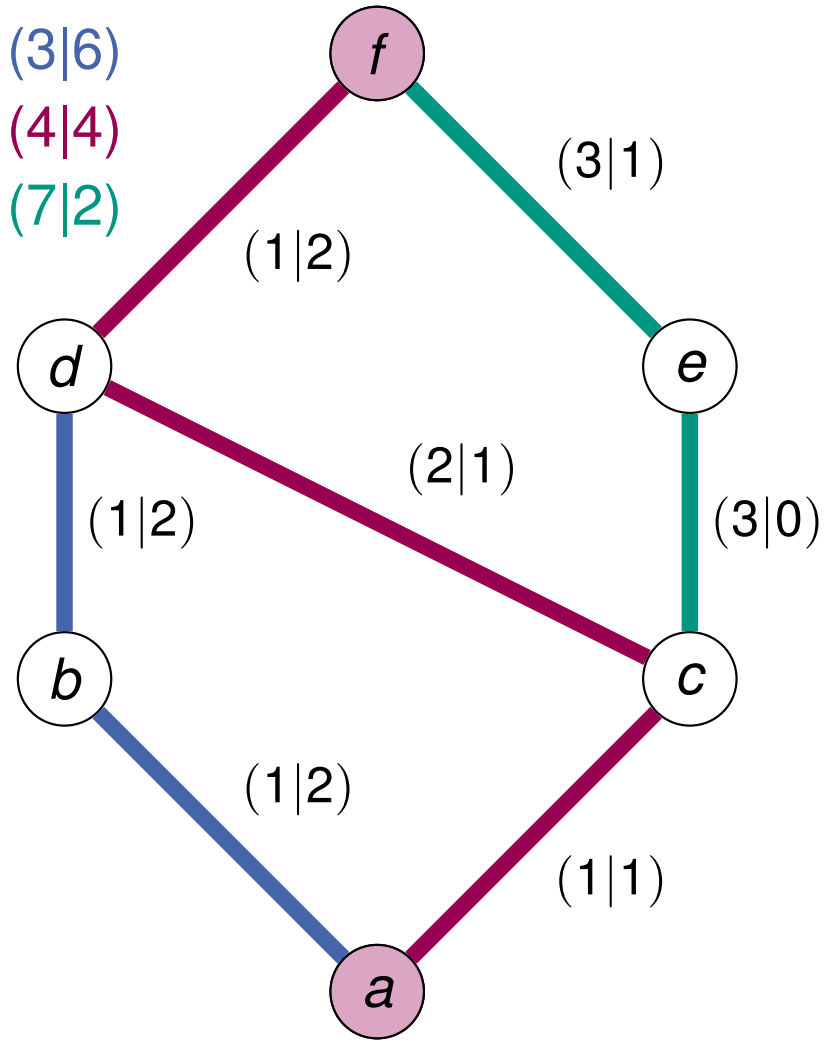


**Robert Geisberger, *Moritz Kobitzsch* and Peter Sanders:**
Flexible Route Planning with Contraction Hierarchies

**Faculty for Computer Science**
Institute for Theoretical Computer Science, Algorithmics II

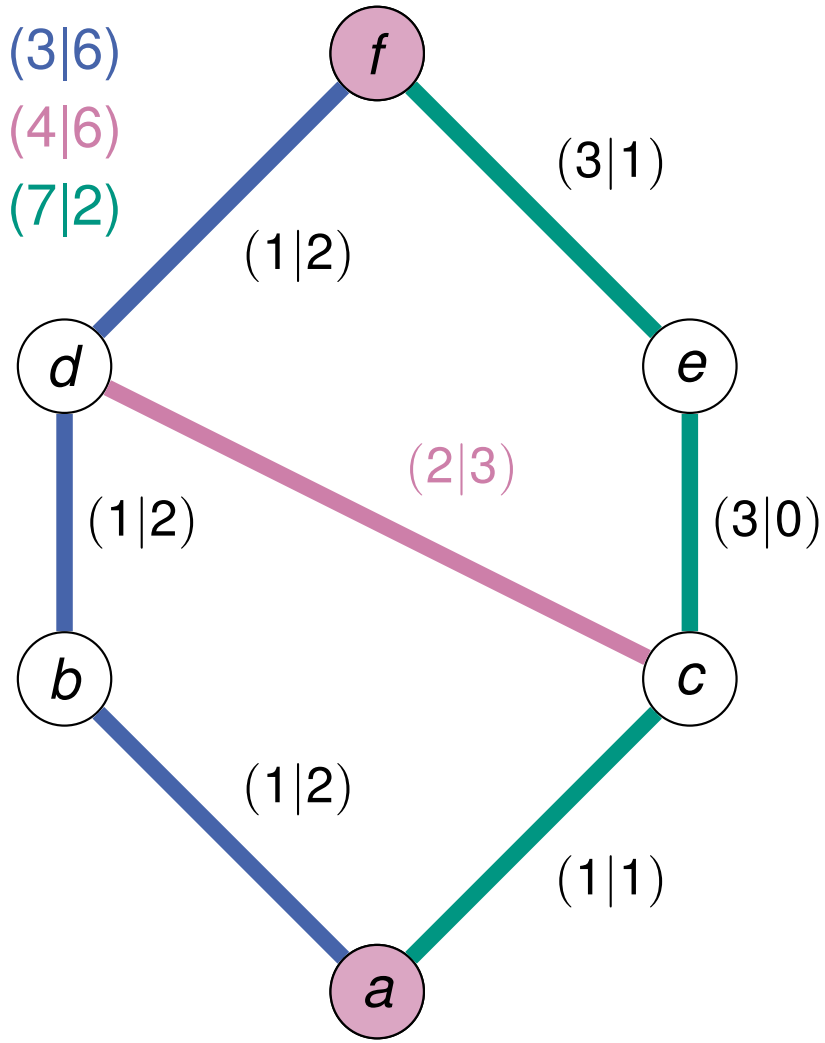
# Pareto Optimality

- find all non dominated routes
- *A* dominates *B* if *A* is better or equal to *B* in every weight term
- may result in an exponential number of routes
- heuristics necessary to gain practical algorithms



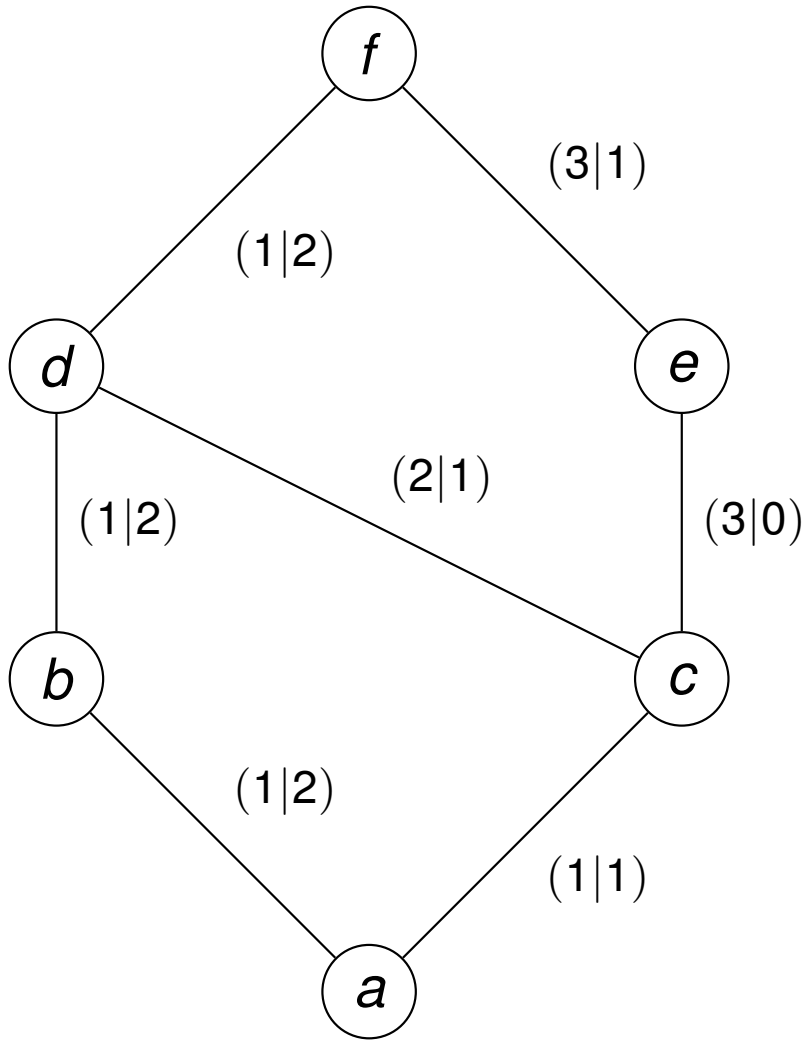
# Pareto Optimality

- find all non dominated routes
- *A* dominates *B* if *A* is better or equal to *B* in every weight term
- may result in an exponential number of routes
- heuristics necessary to gain practical algorithms



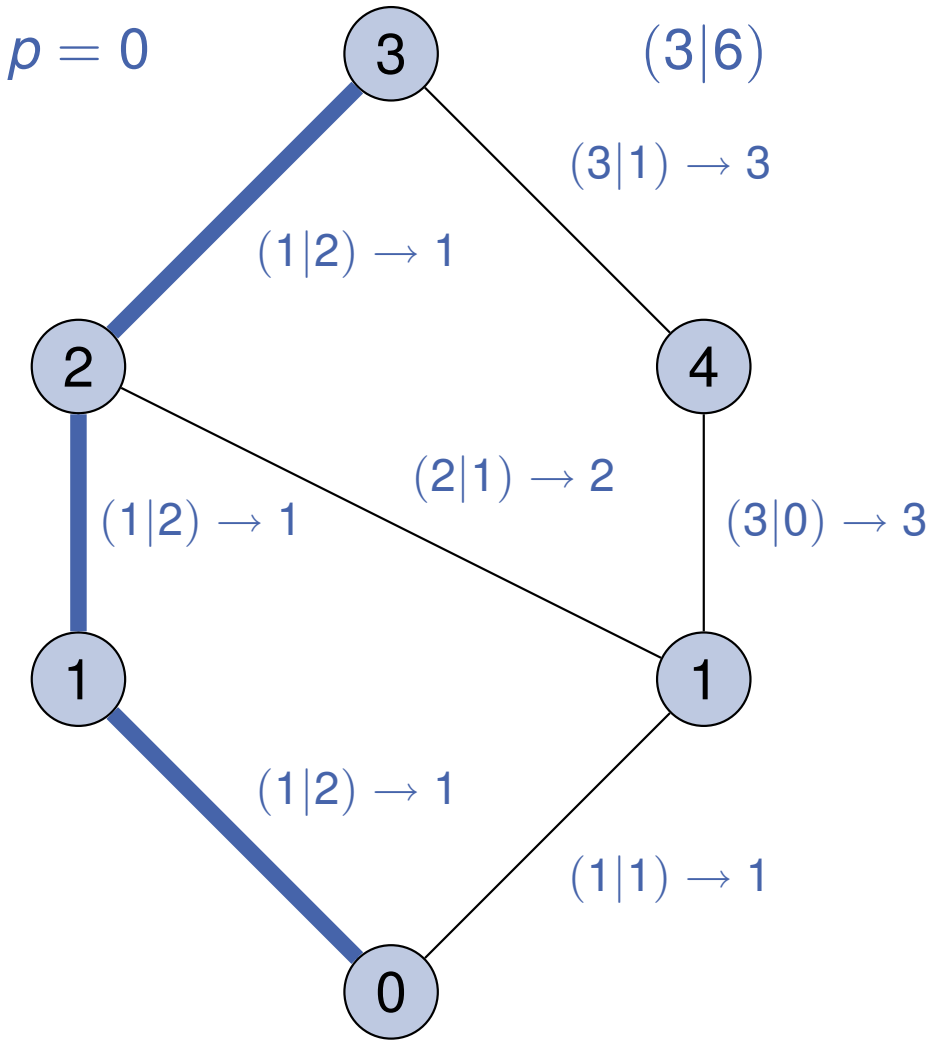
# Parameterized Optimality

- linear combination of weight terms
- $(t|c) \Rightarrow t + p \cdot c$
- preprocessing considers all possible parameter values
- query for fixed parameter $\Rightarrow$ single criteria techniques
- parameter range allows to select maximal impact of weight terms



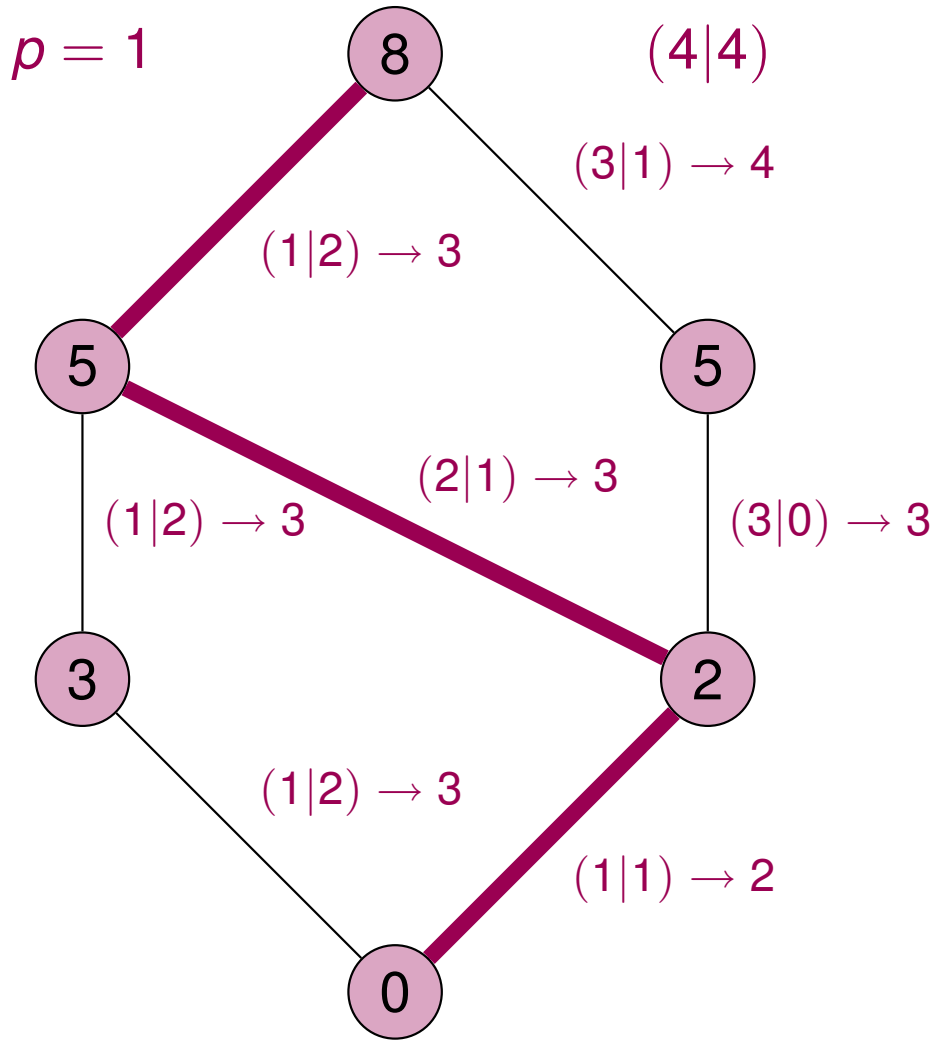
# Parameterized Optimality

- linear combination of weight terms
- $(t|c) \Rightarrow t + p \cdot c$
- preprocessing considers all possible parameter values
- query for fixed parameter $\Rightarrow$ single criteria techniques
- parameter range allows to select maximal impact of weight terms



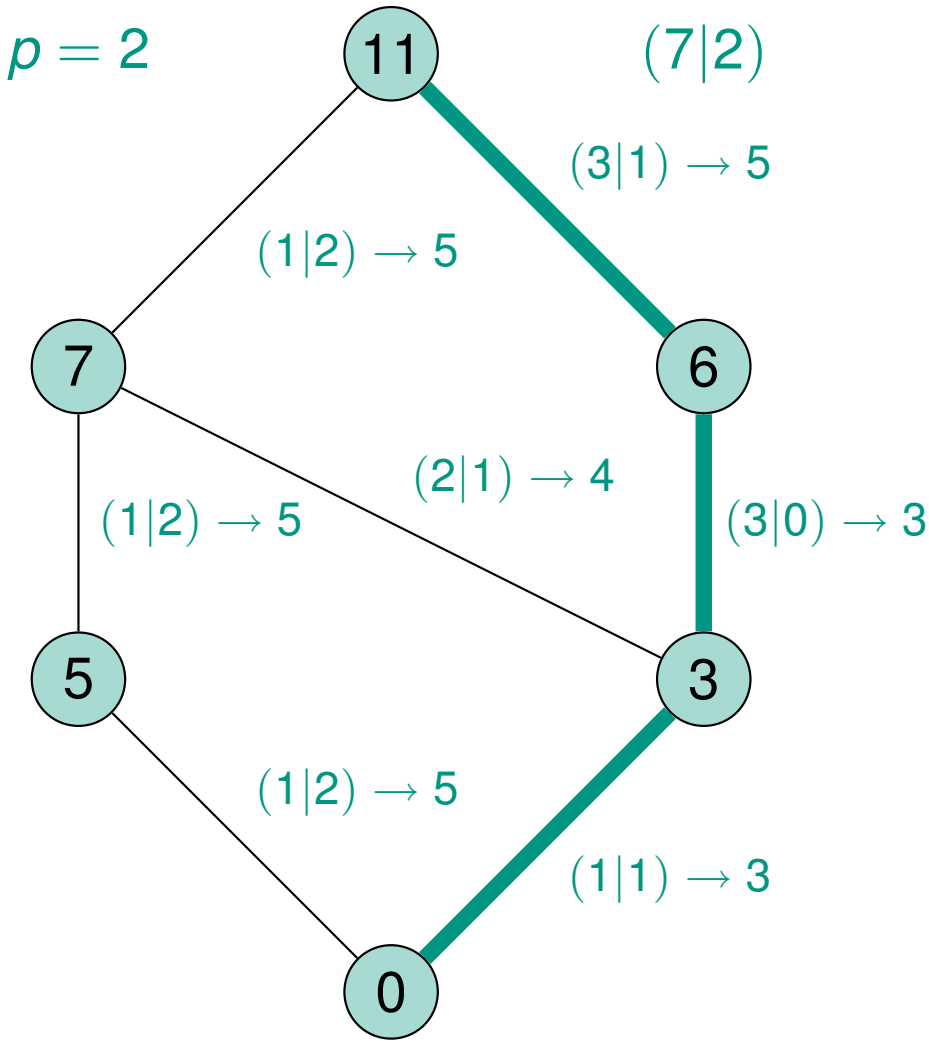
# Parameterized Optimality

- linear combination of weight terms
- $(t|c) \Rightarrow t + p \cdot c$
- preprocessing considers all possible parameter values
- query for fixed parameter $\Rightarrow$ single criteria techniques
- parameter range allows to select maximal impact of weight terms



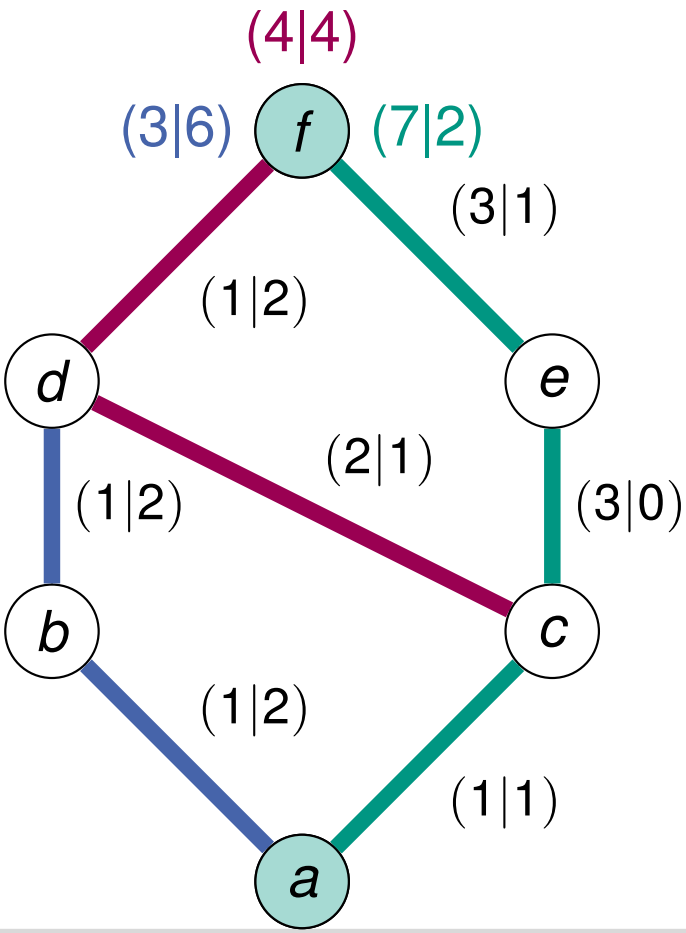
# Parameterized Optimality

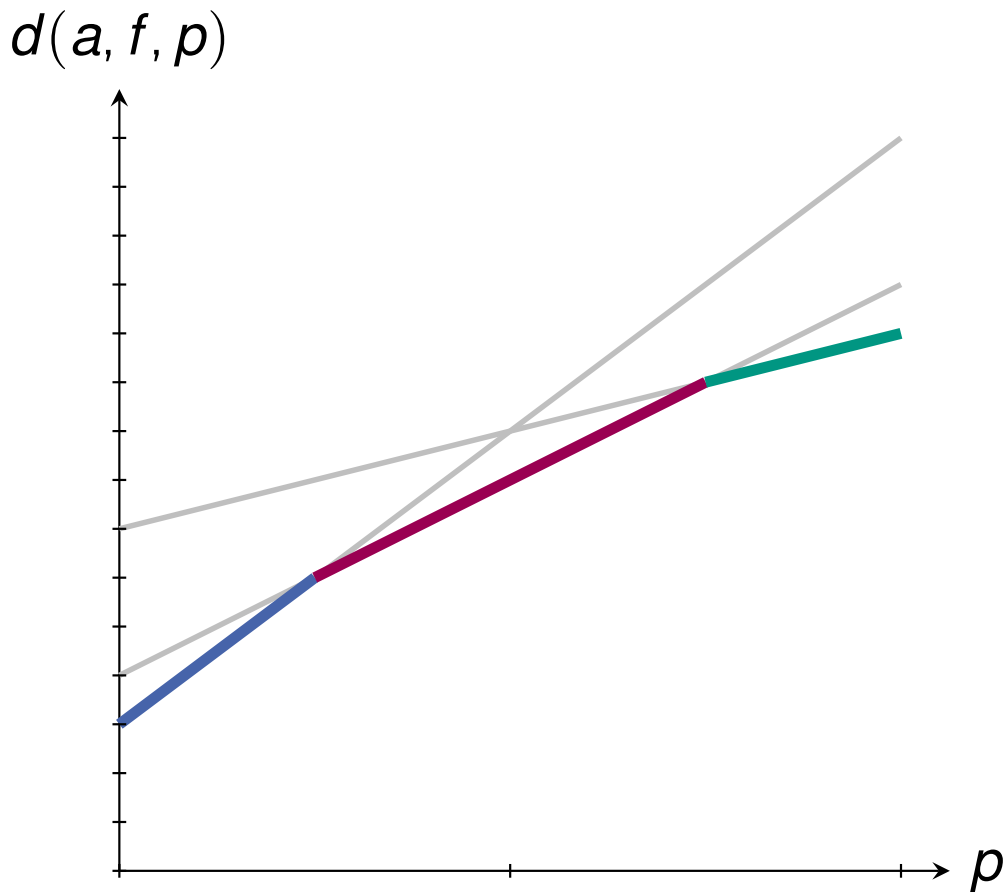
- linear combination of weight terms
- $(t|c) \Rightarrow t + p \cdot c$
- preprocessing considers all possible parameter values
- query for fixed parameter $\Rightarrow$ single criteria techniques
- parameter range allows to select maximal impact of weight terms



# Parameterized Weight Functions

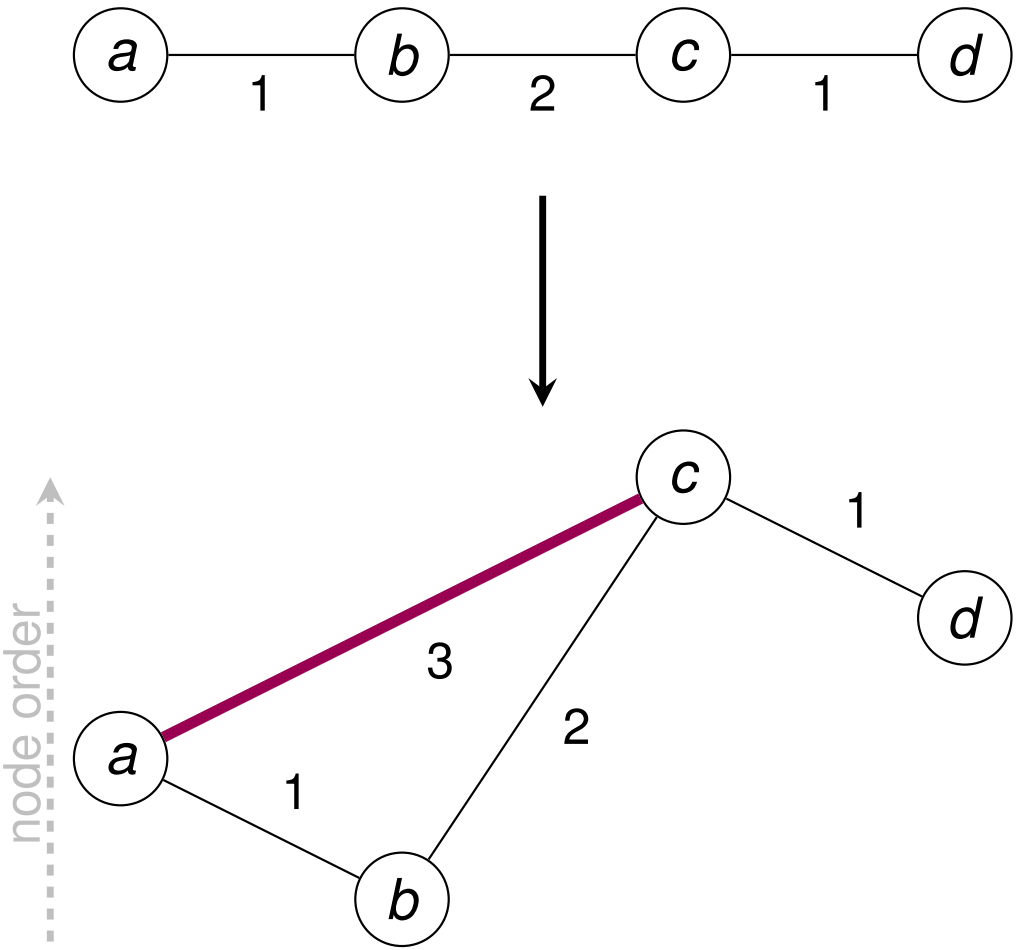
- the distance function $d(a, f)$ for two nodes $a, f$ is concave
- monotone due to positive weight terms



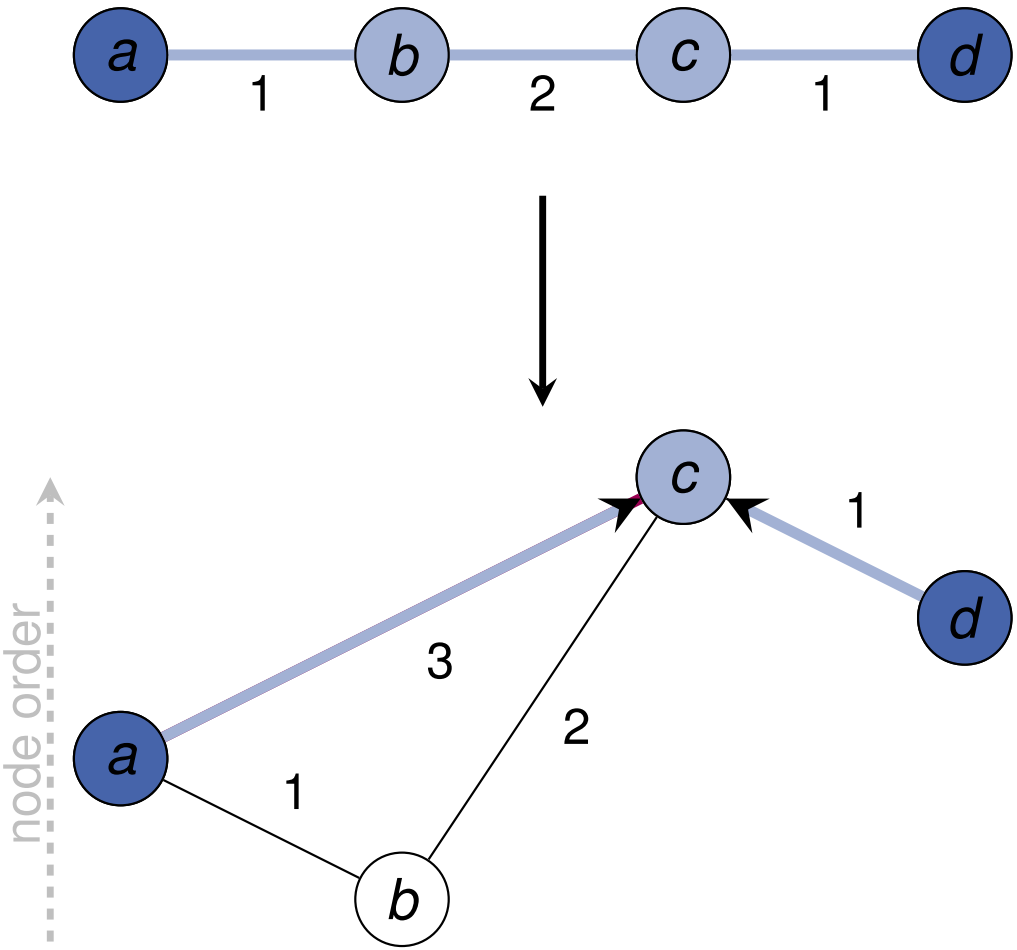
# Contraction Hierarchy Routing

- order nodes by importance $\{v_1, \ldots, v_n\}$
- contract in this order
- preserve original distances by adding shortcuts
- necessity of shortcuts decided by witness paths
- query only relaxes edges to more important nodes



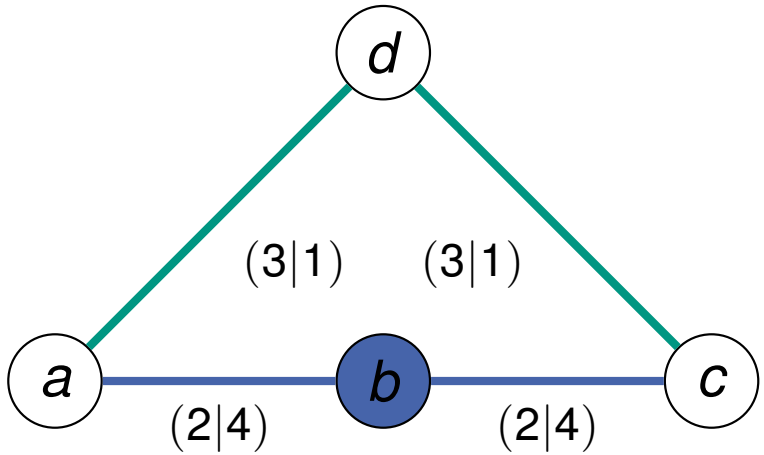
# Contraction Hierarchy Routing

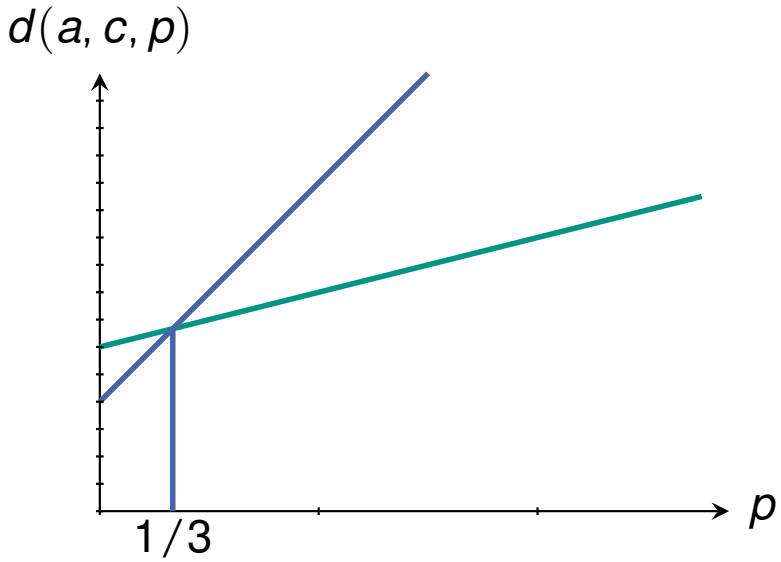
- order nodes by importance $\{v_1, \ldots, v_n\}$
- contract in this order
- preserve original distances by adding shortcuts
- necessity of shortcuts decided by witness paths
- query only relaxes edges to more important nodes



# Flexible Contraction Hierarchies

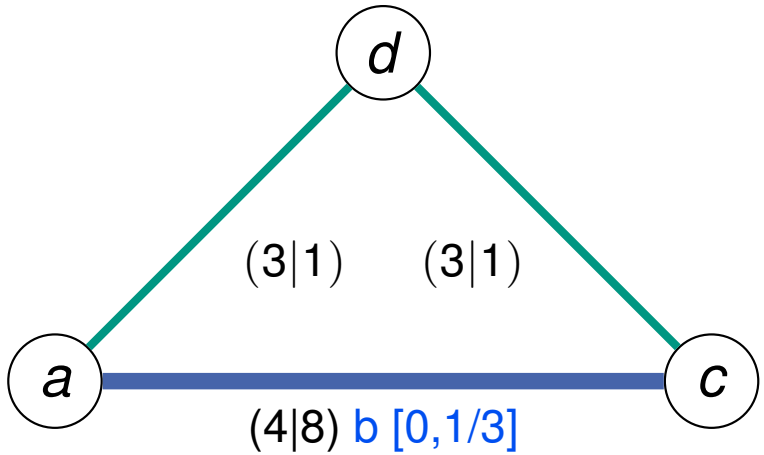
- shortcuts in flexible scenario if $\exists p : (a, b, c)$ is the only shortest path
- necessity of shortcuts only on continuous intervals $\Rightarrow$ store necessity interval in edges
- necessity interval on average deductable from two single criteria Dijkstra queries
- query uses only necessary edges

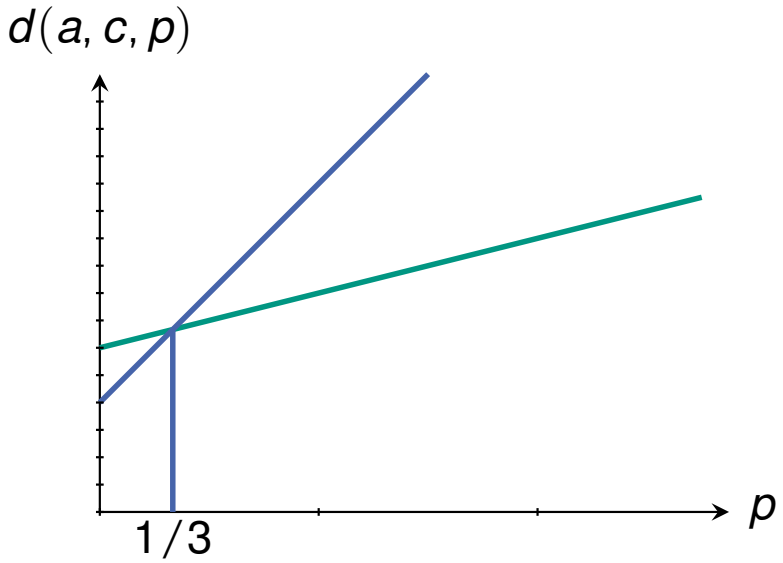


# Flexible Contraction Hierarchies

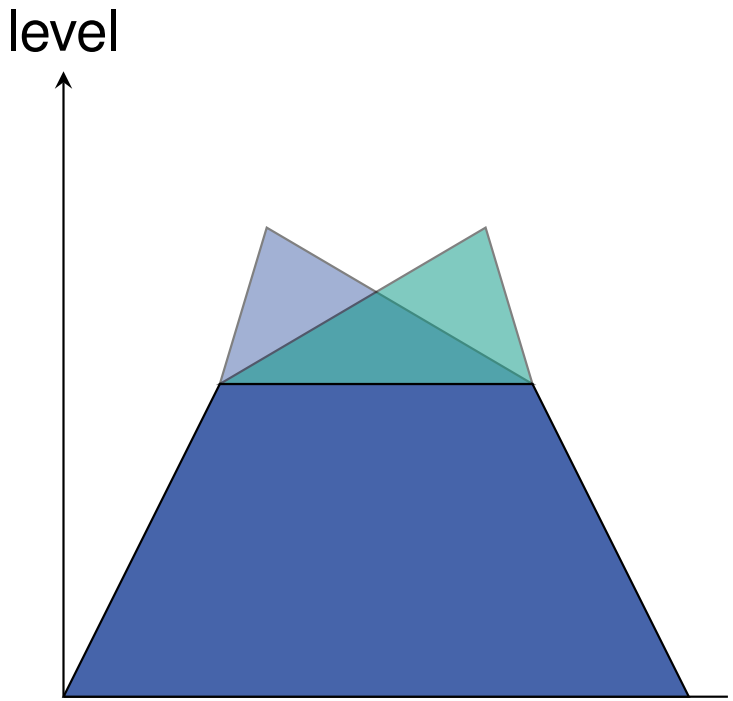
- shortcuts in flexible scenario if $\exists p : (a, b, c)$ is the only shortest path
- necessity of shortcuts only on continuous intervals $\Rightarrow$ store necessity interval in edges
- necessity interval on average deductable from two single criteria Dijkstra queries
- query uses only necessary edges



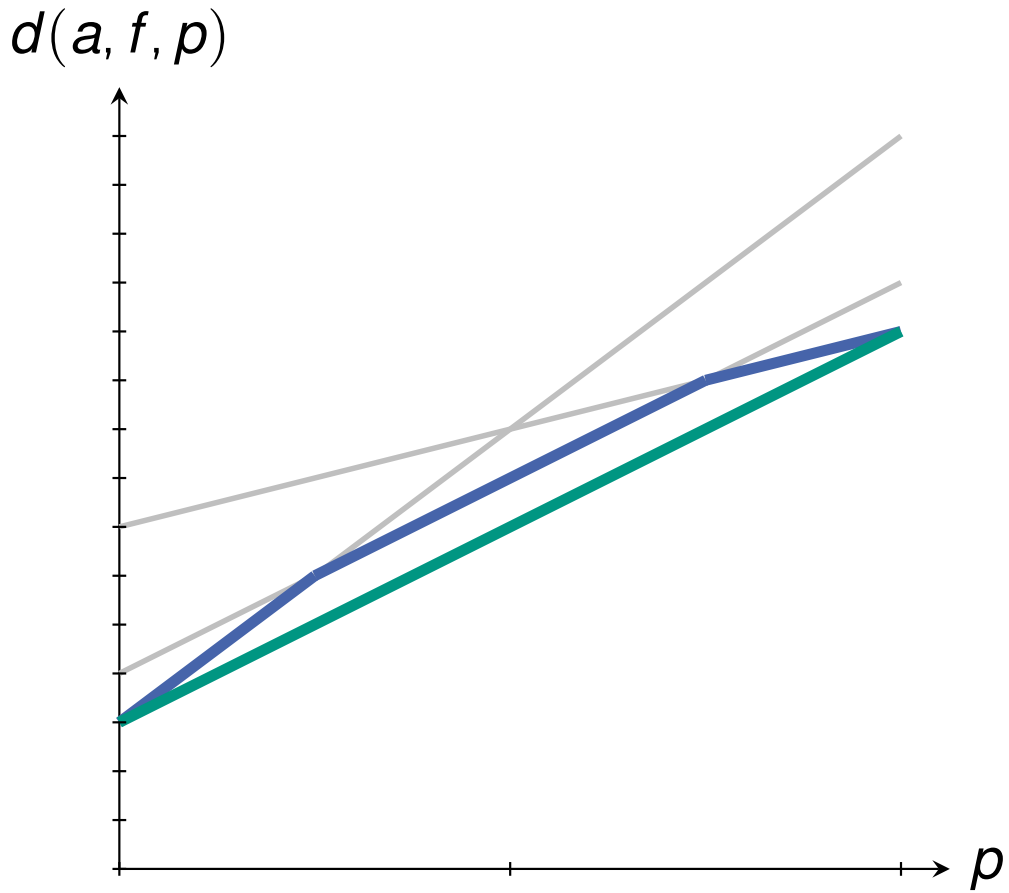
# Flexible Contraction Hierarchies

- single node order not practical for wide parameter range
- a lot of shortcuts for all parameter values
- split parameter interval with heuristics
- repeat as necessary
- buckets to support fast scanning of necessary edges



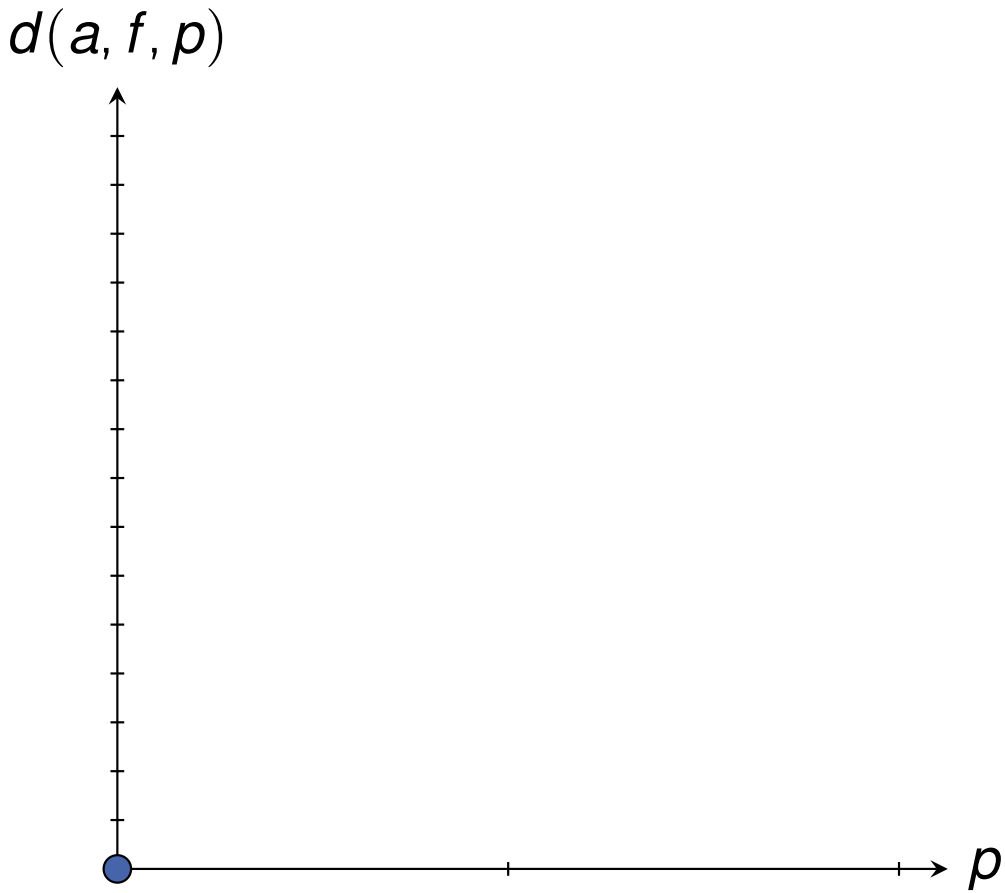
# Core-ALT for flex-CH

- core based approach to
  - speed up preprocessing (uncontracted core)
  - speed up query (contracted core)
- uses ALT algorithm on $|k|$ topmost nodes
- adaptable to flexible scenario with linear interpolation



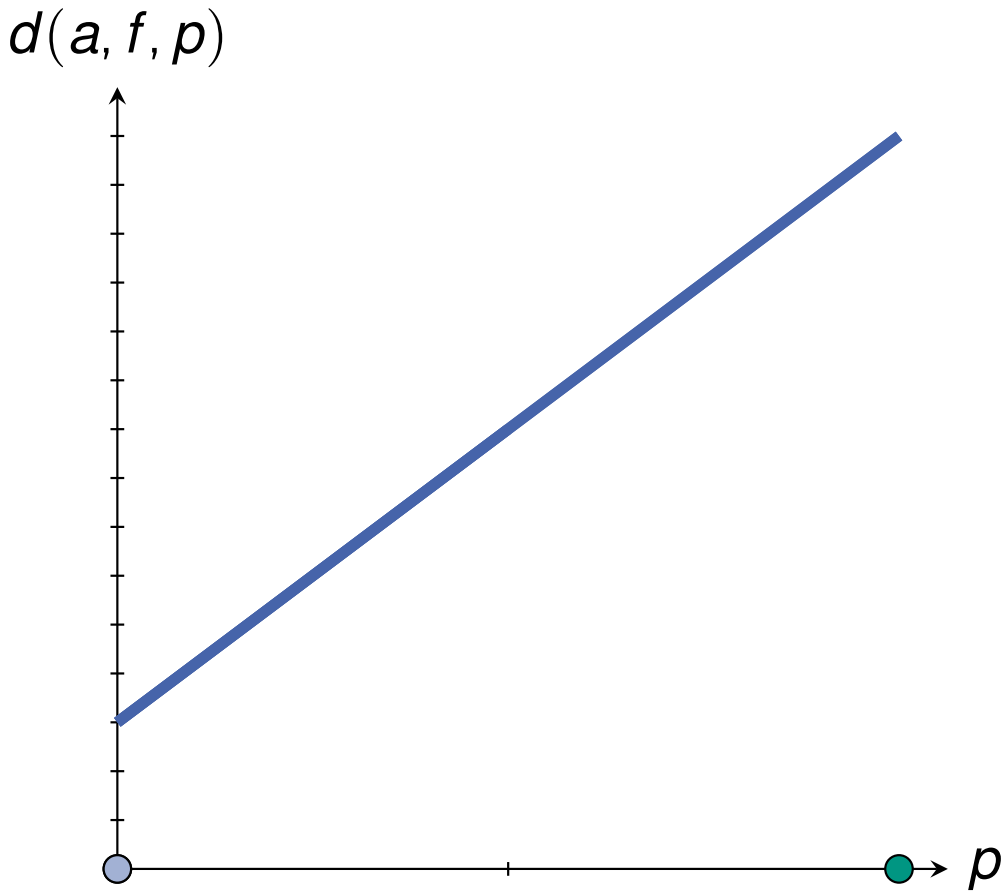
# Profile Queries

- find all possible paths
- profile query with maximal $3 \cdot \#paths - 2$ queries
- approximation: recursion only if improvement larger more than $(1 + \varepsilon)$ possible



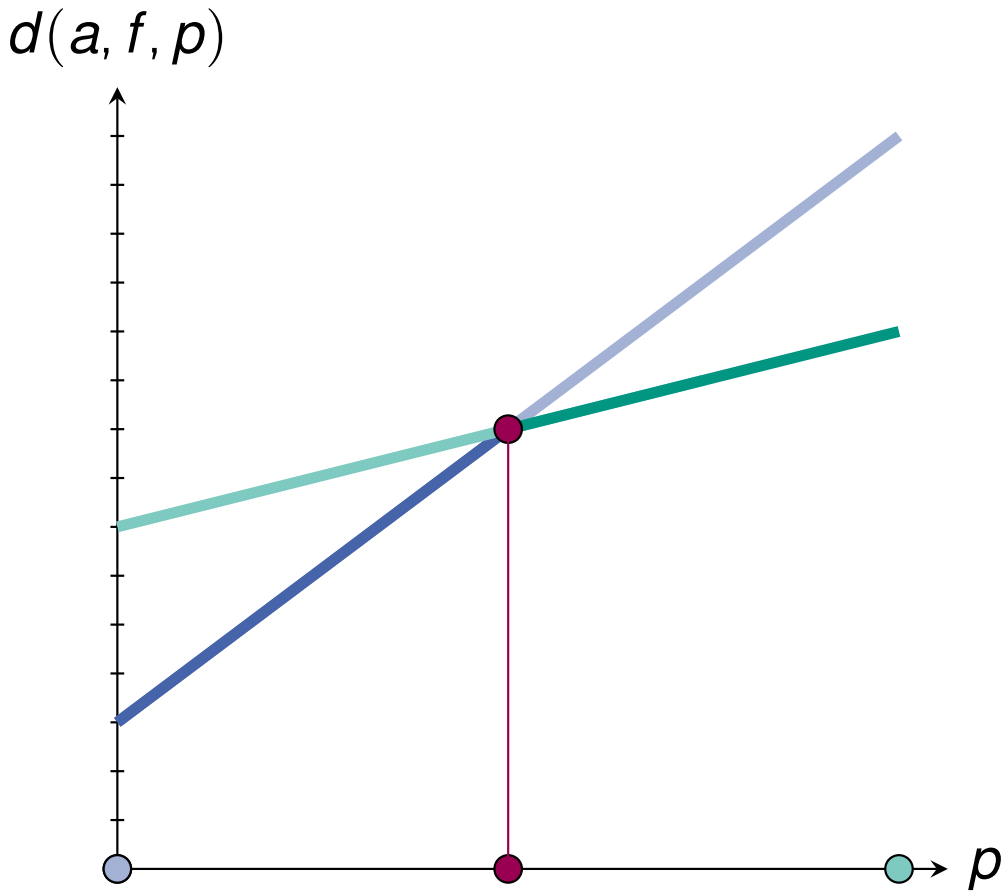
# Profile Queries

- find all possible paths
- profile query with maximal $3 \cdot \#paths - 2$ queries
- approximation: recursion only if improvement larger more than $(1 + \varepsilon)$ possible



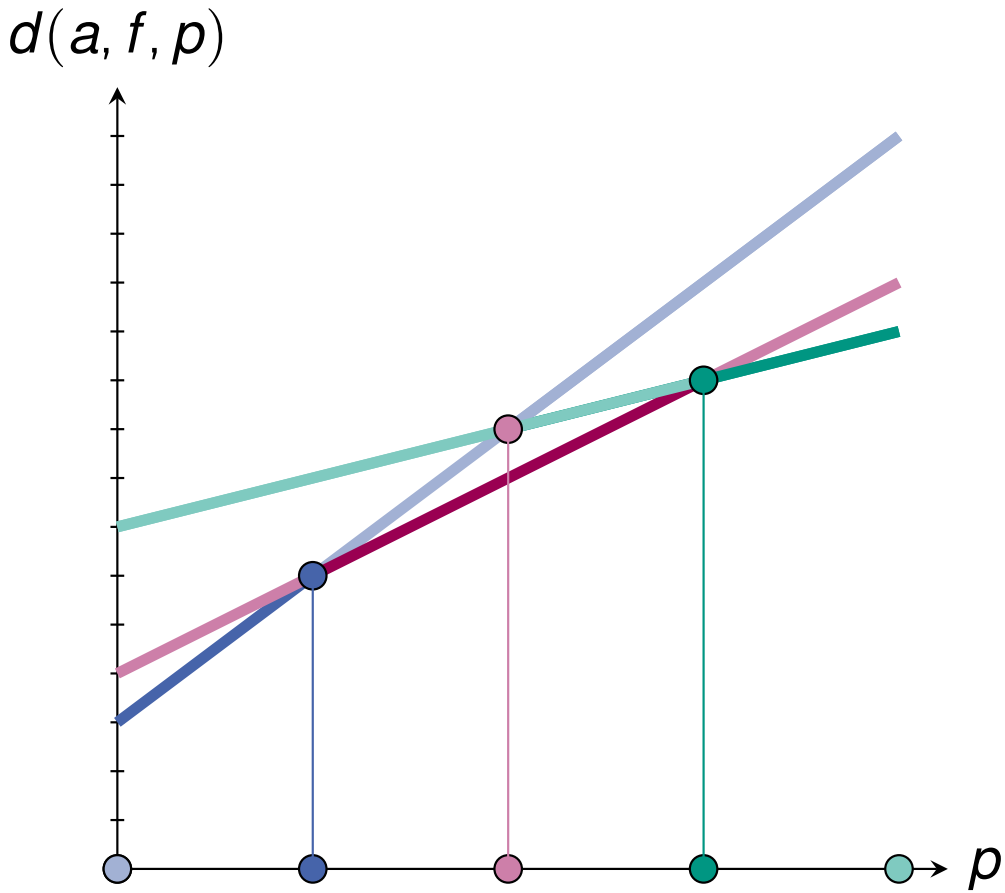
# Profile Queries

- find all possible paths
- profile query with maximal $3 \cdot \#paths - 2$ queries
- approximation: recursion only if improvement larger more than $(1 + \varepsilon)$ possible



# Profile Queries

- find all possible paths
- profile query with maximal $3 \cdot \#paths - 2$ queries
- approximation: recursion only if improvement larger more than $(1 + \varepsilon)$ possible

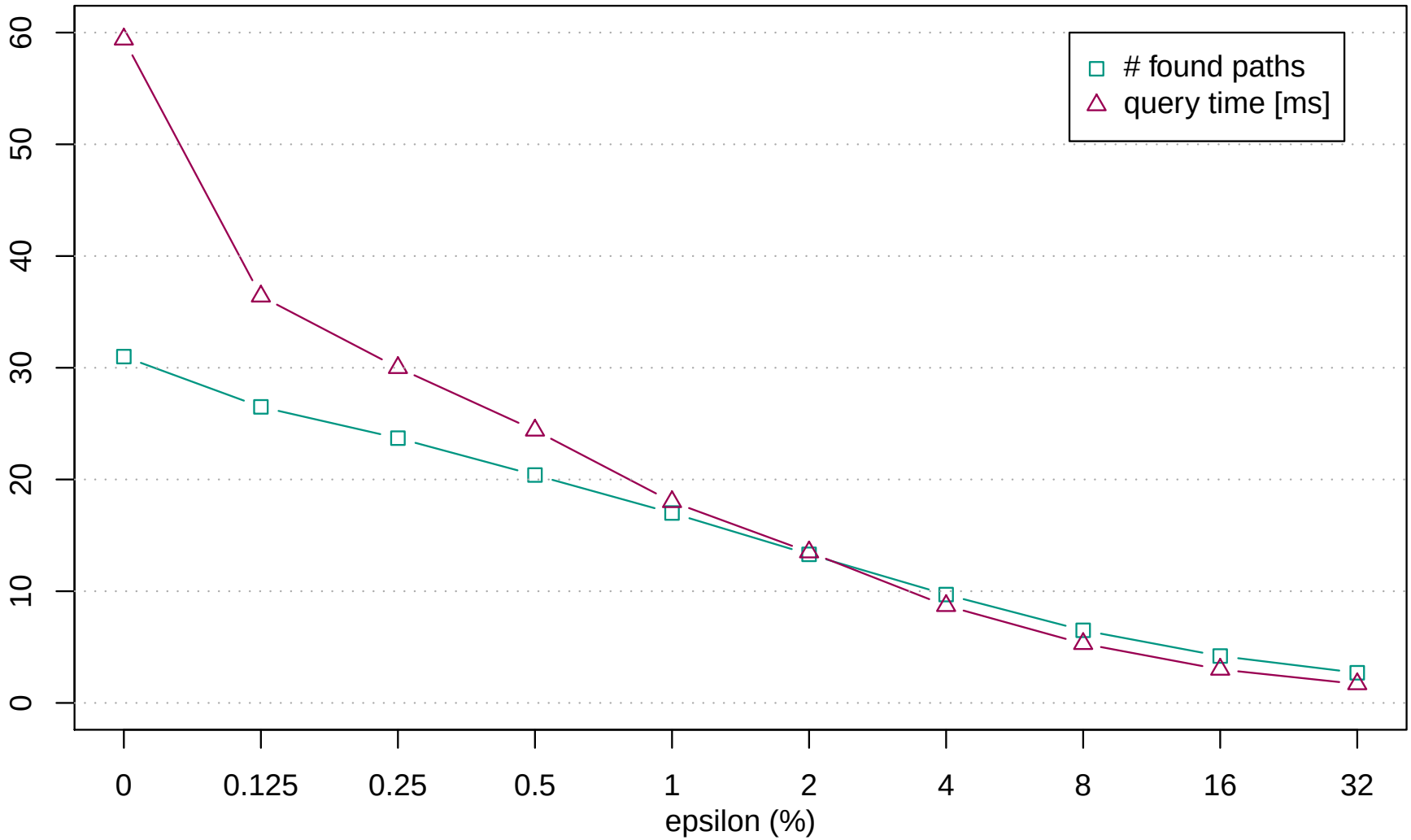


# Experimental results

Table: Preprocessing and query performance for 64 landmarks on the German road network, average over 10 000 queries. (4 692 751 nodes and 10 806 191 directed edges)

| core | preproc [hh:mm] | space [B/node] | query [ms] | speedup |
|---|---|---|---|---|
| - | - | 60 | 2 037.52 | 1 |
| 0 | 1 : 54 | 159 | 2.90 | 698 |
| uncontracted 5 000 | 1 : 08 | 167 | 2.58 | 789 |
| contracted 10 000 | 2 : 07 | 183 | 0.63 | 3 234 |

# Experimental results



# Comparison to SHARC

Table: Approximated profile search on Western Europe

| algorithm | preproc | $\varepsilon$ | # paths | time [ms] |
|---|---|---|---|---|
| flexCH | 5 : 15 | 0.00 | 12.5 | 21.9 |
| flexCH | 5 : 15 | 0.01 | 5.7 | 6.2 |
| Pareto-SHARC | 7 : 12 | (no guarantee) | 5.3 | 35.4 |

# Conclusion

- exact flexible routing for server scenarios
- comparable even to single criteria speedups
- circumvents problems of Pareto-optimality
- scales well through parameter splitting

# Thank You

Thank you for your attention.

# Questions

Questions?

# Future Work

- mobile implementation
- apply parallelization techniques
- more than two weight terms (?)