

# Algorithmen für Routenplanung

11. Vorlesung, Sommersemester 2019

Tobias Zündorf | 4. Juni 2019

INSTITUT FÜR THEORETISCHE INFORMATIK · ALGORITHMIK



**Alternative  
Route**

→

flickr.com/photos/duncan/

# Wiederholung Punkt-zu-Punkt

## Anfrage:

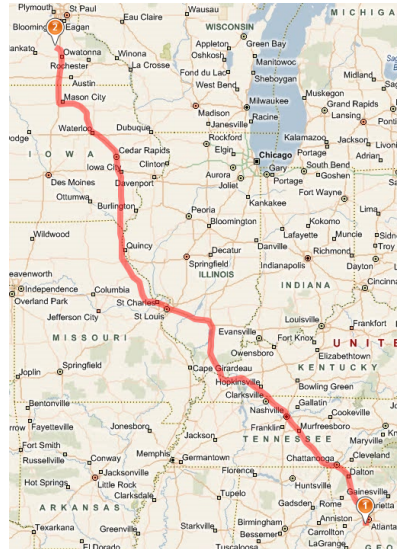
- finde die **beste** Route in einem Transportnetz

## Idee:

- Netzwerk als Graph  $G = (V, E)$
- Kantengewichte sind **Reisezeiten**
- **kürzester** Weg in  $G$  entspricht **schnellster** Verbindung

## Ergebnisse:

- schnelle Algorithmen existieren



# Alternativ-Routen

## Anfrage:

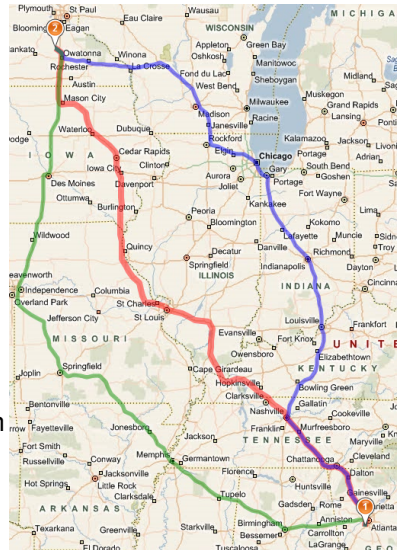
- finde **gute Alternativen** in einem Transportnetz

## Problem:

- der kürzeste Weg ist wohldefiniert
- Was aber ist eine gute Alternative?
- Problem erscheint rein heuristischer Natur

## Ziele:

- lieber keine als schlechte Routen zeigen
- sollte nicht deutlich langsamer als Punkt-zu-Punkt sein

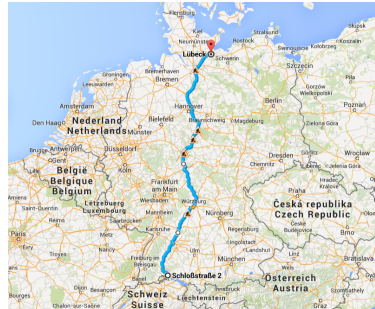
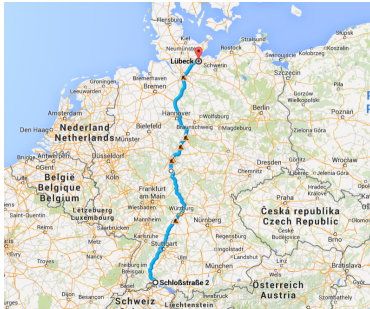


## Berechne top- $k$ kürzeste Wege

- + Wege sind möglichst kurz
- oft erst für hohes  $k$  relevant

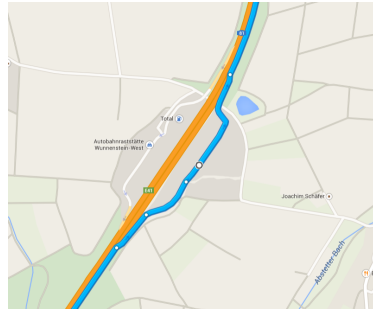
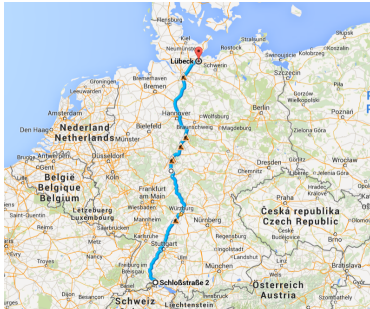
## Berechne top- $k$ kürzeste Wege

- + Wege sind möglichst kurz
- oft erst für hohes  $k$  relevant



## Berechne top- $k$ kürzeste Wege

- + Wege sind möglichst kurz
- oft erst für hohes  $k$  relevant

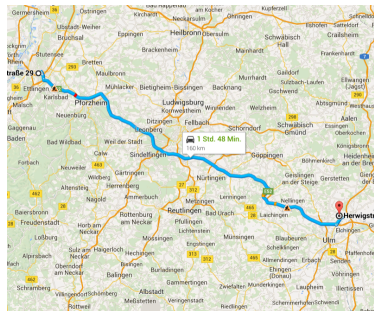
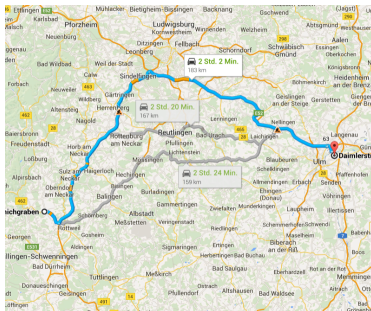


## Optimiere verschiedene Metriken

- z.B. schnellster und kürzester Weg
- + bedient verschiedene Vorlieben
- verpasst interessante Alternativen

## Optimiere verschiedene Metriken

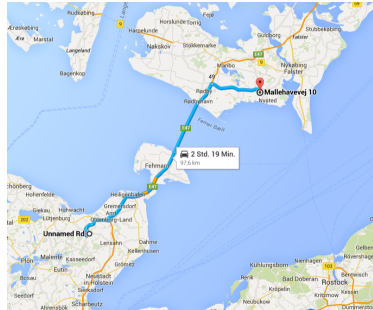
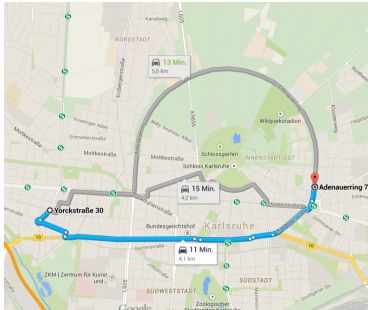
- z.B. schnellster und kürzester Weg
- + bedient verschiedene Vorlieben
- verpasst interessante Alternativen





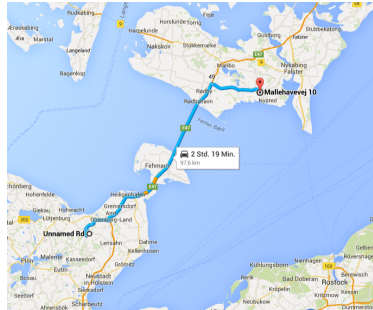
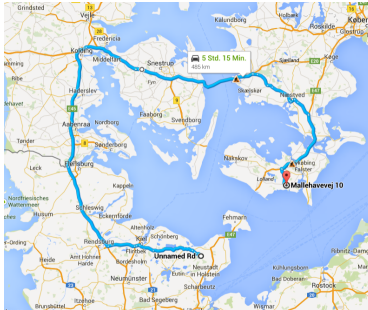
## Disjunkte Pfade

- + stark unterschiedliche Pfade
- verpasst interessante Alternativen



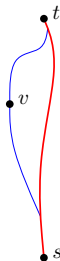
## Disjunkte Pfade

- + stark unterschiedliche Pfade
- verpasst interessante Alternativen



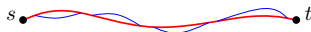
## Via-Knoten

- nutze dritten Knoten
- berechne zusammengesetzten Pfad
- Problem: **Woher** kommt dieser Knoten?



## Simulierter Stau

- **verlangsamen** einzelne Segmente künstlich
- Gefahr vieler kleiner Umleitungen



## Und wie?

- was macht eine gute Alternative aus
- wie berechnen wir sie schnell

# Intuition

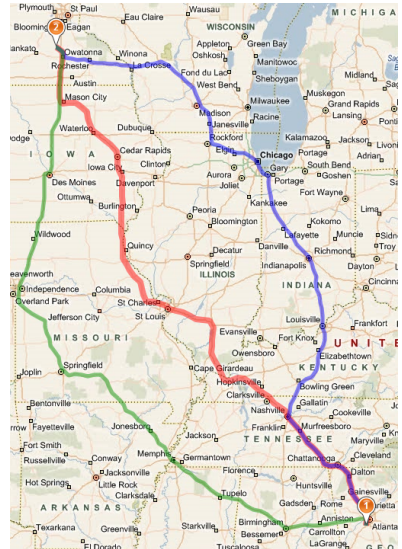
## Alternativen sollten erfüllen:

- nicht viel länger als der kürzeste Weg
- signifikant verschieden

## Erste Idee:

- finde einen Pfad, der Länge und **Gemeinsamkeit minimiert**
  - maximal  $x\%$  länger
  - teilt maximal  $y\%$

## Ist das genug?



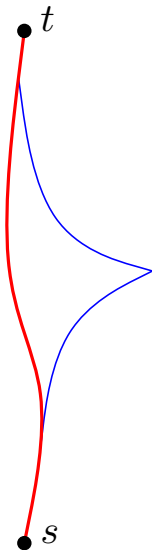
# Das dritte Kriterium

## Problem:

- Routen können seltsam aussehen
- sogar wenn sie verschieden und kurz sind
- lokale Umwege
- “Diese Strecke macht doch keinen Sinn. Das kann ich besser!”

## Idee:

- kurze Subpfade müssen kürzeste Pfade sein
  - beliebige Paare von Knoten auf dem Pfad sollen kleinen stretch haben
- ⇒ keine unnötigen lokalen Umwege



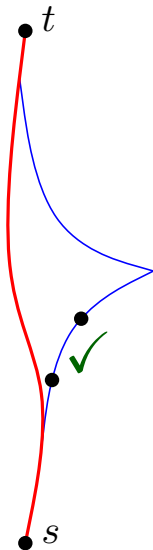
# Das dritte Kriterium

## Problem:

- Routen können seltsam aussehen
- sogar wenn sie verschieden und kurz sind
- lokale Umwege
- “Diese Strecke macht doch keinen Sinn. Das kann ich besser!”

## Idee:

- kurze Subpfade müssen kürzeste Pfade sein
  - beliebige Paare von Knoten auf dem Pfad sollen kleinen stretch haben
- ⇒ keine unnötigen lokalen Umwege



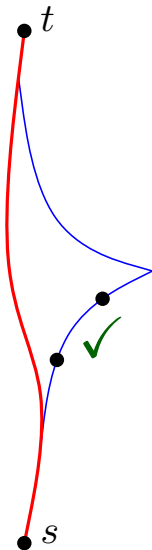
# Das dritte Kriterium

## Problem:

- Routen können seltsam aussehen
- sogar wenn sie verschieden und kurz sind
- lokale Umwege
- “Diese Strecke macht doch keinen Sinn. Das kann ich besser!”

## Idee:

- kurze Subpfade müssen kürzeste Pfade sein
  - beliebige Paare von Knoten auf dem Pfad sollen kleinen stretch haben
- ⇒ keine unnötigen lokalen Umwege



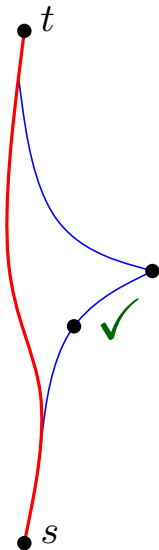
# Das dritte Kriterium

## Problem:

- Routen können seltsam aussehen
- sogar wenn sie verschieden und kurz sind
- lokale Umwege
- “Diese Strecke macht doch keinen Sinn. Das kann ich besser!”

## Idee:

- kurze Subpfade müssen kürzeste Pfade sein
  - beliebige Paare von Knoten auf dem Pfad sollen kleinen stretch haben
- ⇒ keine unnötigen lokalen Umwege





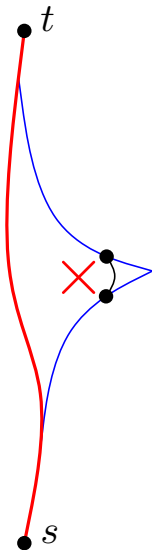
# Das dritte Kriterium

## Problem:

- Routen können seltsam aussehen
- sogar wenn sie verschieden und kurz sind
- lokale Umwege
- “Diese Strecke macht doch keinen Sinn. Das kann ich besser!”

## Idee:

- kurze Subpfade müssen kürzeste Pfade sein
  - beliebige Paare von Knoten auf dem Pfad sollen kleinen stretch haben
- ⇒ keine unnötigen lokalen Umwege



## Idee

Alternativ-Pfade sollen hinreichend verschieden sein.

- $P$  ist die Kantenmenge des kürzesten  $st$ -Pfads
- $P'$  ist die Kantenmenge des alternativen  $st$ -Pfads
- $\ell(Q)$  ist die Summe über alle Kanten-Längen
- Eine Alternative ist  $\gamma$ -verschieden, wenn

$$\ell(P \cap P') \leq \gamma \ell(P)$$

## Idee

Alternativ-Pfade sollen kein große Umwege sein

- Sei  $\text{dist}(s, t)$  die Länge des kürzesten  $st$ -Pfads
- Sei  $P'$  der alternative  $st$ -Pfad und  $\ell(P')$  seine Länge
- $P'$  hat einen Stretch von höchstens  $\epsilon$ , wenn

$$\ell(P') \leq (1 + \epsilon) \text{dist}(s, t)$$

Kriterium für die Vorlesung vereinfacht, siehe [ADGW13] für Details.

## Idee

Alternativ-Pfade sollen lokal sinnvoll sein.

- Sei  $P'$  der alternative Pfad und  $P'_{xy}$  der Subpfad von  $x$  bis  $y$
- $P'$  besteht den  $T$ -Test, wenn für alle  $P'_{xy}$  mit  $\ell(P'_{xy}) \leq T$  gilt, dass  $P'_{xy}$  ein kürzester  $xy$ -Pfad ist
- $P'$  ist  $\alpha$ -lokal-optimal, wenn er den Test für  $T = \alpha \text{dist}(s, t)$  besteht

Kriterium für die Vorlesung vereinfacht, siehe [ADGW13] für Details.

- **Ziel:** Finde Pfad, der  $\gamma$ -verschieden ist, einen Stretch  $\epsilon$  hat und  $\alpha$ -lokal-optimal ist.

- **Ziel:** Finde Pfad, der  $\gamma$ -verschieden ist, einen Stretch  $\epsilon$  hat und  $\alpha$ -lokal-optimal ist.
- $\gamma$ ,  $\epsilon$ , und  $\alpha$  sind Tuningparameter des Problems
- Typische Werte sind:  $\alpha = 25\%$ ,  $\epsilon = 25\%$ ,  $\gamma = 80\%$

- **Ziel:** Finde Pfad, der  $\gamma$ -verschieden ist, einen Stretch  $\epsilon$  hat und  $\alpha$ -lokal-optimal ist.
- $\gamma$ ,  $\epsilon$ , und  $\alpha$  sind Tuningparameter des Problems
- Typische Werte sind:  $\alpha = 25\%$ ,  $\epsilon = 25\%$ ,  $\gamma = 80\%$
- Bei mehreren Alternativen:  
Neue Alternative muss  $\gamma$ -verschieden von der Vereinigung aller bisher gefunden Pfade sein.

# Weiteres Vorgehen

- Bisher: Wir haben gültige Alternativen formalisiert
- Nun: Wie finden wir gute Alternativen?

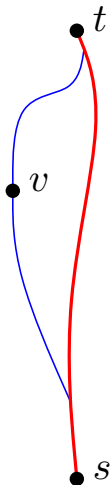


- Bisher: Wir haben gültige Alternativen formalisiert
- Nun: Wie finden wir gute Alternativen?
- **Vorgehen:**
  - Finde Kandidaten-Pfad nach einem heuristischen Verfahren
  - Prüfe, ob der Pfad eine gute Alternative ist
- Findet nicht immer eine Alternative

- Bisher: Wir haben gültige Alternativen formalisiert
- Nun: Wie finden wir gute Alternativen?
- **Vorgehen:**
  - Finde Kandidaten-Pfad nach einem heuristischen Verfahren
  - Prüfe, ob der Pfad eine gute Alternative ist
- Findet nicht immer eine Alternative
- **Problem:** Wie effizient testen?
- Nicht trivial
- Der Test braucht eine nicht vernachlässigbare Menge von Rechenzeit
- Details nicht in der Vorlesung

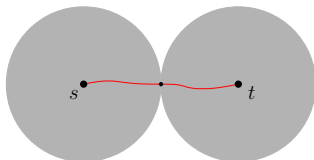
## Single via paths:

- **Konkatenation** von zwei kürzesten Pfaden:  $s-v$  und  $v-t$
- Eigenschaften:
  - **linear** viele Pfade
  - einzelner Pfad ist definiert durch via Knoten  $v$  (Notation:  $P_v$ )
  - lokal optimal von  $s$  nach  $v$  und  $v$  nach  $t$
  - Verletzungen nur um  $v$  herum
  - Alternativen können effizient berechnet werden sobald  $v$  bekannt ist



## Relaxiertes Stop-Kriterium

- suche bidirektional
- stoppe sobald Radien größer als  $(1 + \epsilon)\ell(Opt)$



## Finde Alternative:

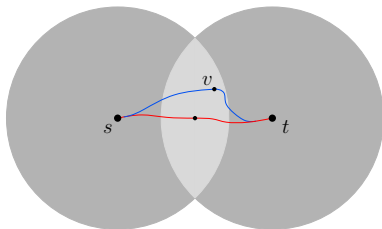
- für alle  $v$ , die von beiden Suchen gescannt worden sind:
- Teste, ob  $P_v$  den Anforderungen genügt
- finde **besten**  $P_v$  anhand einer Optimierungsfunktion

## Problem:

- Anzahl der Kandidaten ist **sehr hoch**
- Interaktion mit Beschleunigungstechniken?

## Relaxiertes Stop-Kriterium

- suche bidirektional
- stoppe sobald Radien größer als  $(1 + \epsilon)\ell(Opt)$



## Finde Alternative:

- für alle  $v$ , die von beiden Suchen gescannt worden sind:
- Teste, ob  $P_v$  den Anforderungen genügt
- finde **besten**  $P_v$  anhand einer Optimierungsfunktion

## Problem:

- Anzahl der Kandidaten ist **sehr hoch**
- Interaktion mit Beschleunigungstechniken?

## Idee

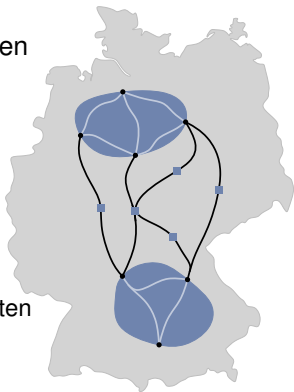
- Lasse Vorwärts- und Rückwärtssuche laufen bis Stretch erreicht ist
- Entpacke Pfad für jeden Knoten im gemeinsamen Suchraum
- Teste Pfad

## Probleme

- gegenläufige Ziele
  - CH will Suchraum klein haben
  - Alternativrouten brauchen genug Kandidaten
- Tests und Pfadenpackungen kosten Zeit

## “Lösung”

- Für gute Alternativen muss man vereinzelt auch Kanten folgen, die den Suchraum verlassen
- Je weiter man den Suchraum verlässt, desto besser werden die gefunden Pfade, aber desto schlechter werden die Laufzeiten
- Siehe [ADGW13] für Details



## Idee

- Analog zu Dijkstra
- Lasse Vorwärts- und Rückwärtssuche laufen bis Stretch erreicht ist
- Teste Pfad

## Probleme

- Reach dünnt Suchraum schlechter aus als CH
- Mehr Kandidaten
- Bessere Alternativen
- Aber langsamer

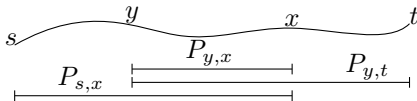
## Problem mit Via-Knoten:

- lokale Optimalität am Via-Knoten verletzt



## Problem mit Via-Knoten:

- lokale Optimalität am Via-Knoten verletzt



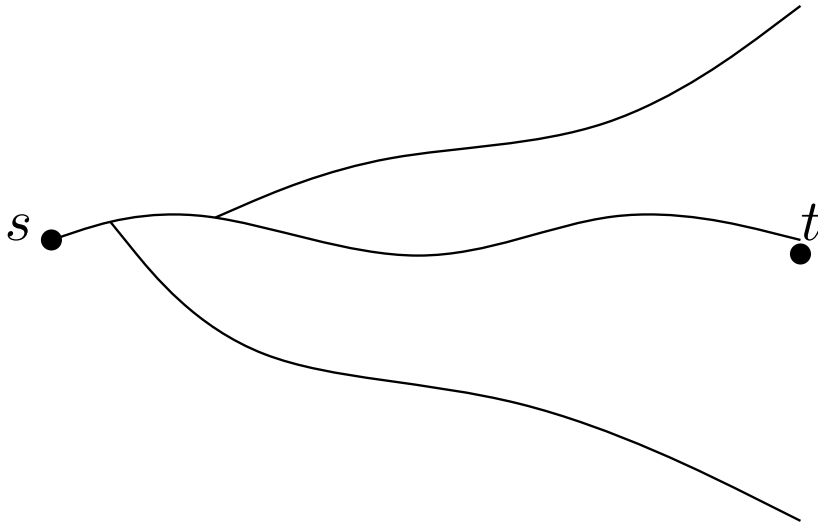
## Idee:

- Suche zwei kürzeste Pfade  $P_{s,y}$  und  $P_{x,t}$
- mit  $P_{x,y} \subseteq P_{s,y}$  und  $P_{x,y} \subseteq P_{x,t}$
- $P_{x,y}$  heißt Plateau
- Alternative besteht  $T$ -Test für  $T \leq \ell(P_{x,y})$
- Langes Plateau  $\rightarrow$  gute Alternative
- Randfall Via-Knoten:  $x = y$

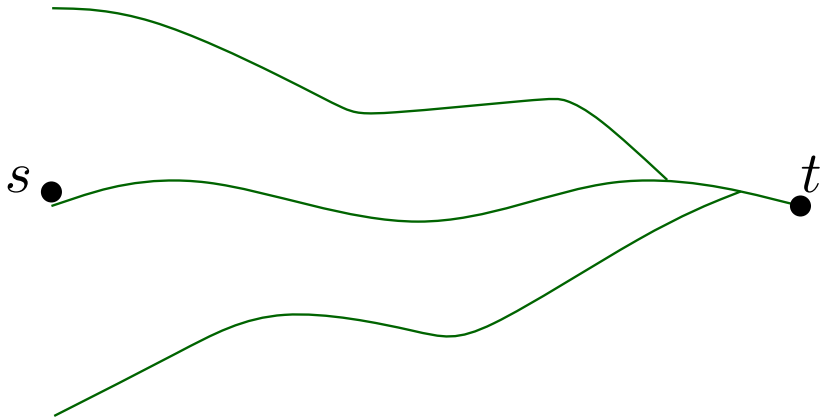
$s$  ●

$t$   
●

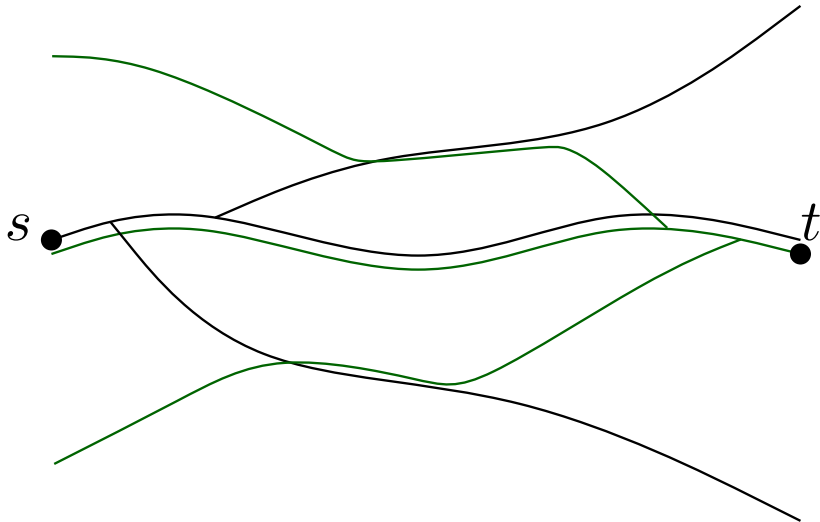
# Plateau



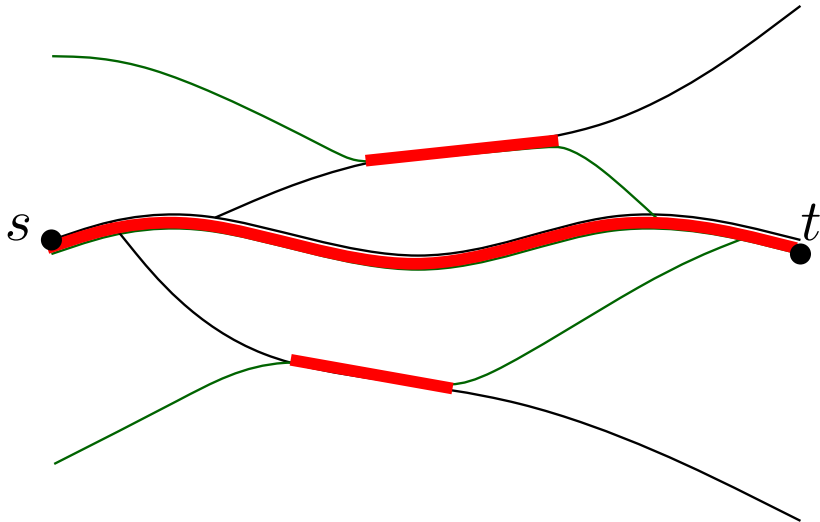
# Plateau



# Plateau

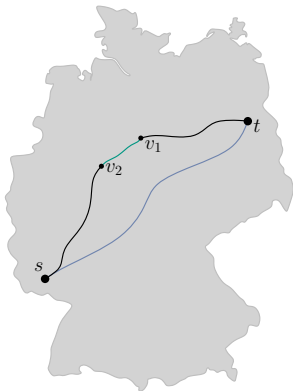


# Plateau



- benötigt optimale Suchbäume
- einfach mit Dijkstra
- CH: komplex, aber möglich

(Grundlage: vollständiges Entpacken des Suchraums: [Kob13])



- Via-Knoten einfacher und schneller zu berechnen
  - Aber: Mehr Via-Knoten als Plateaus
  - d.h. mehr Zulässigkeitstests mit Via-Knoten
- 
- Jede Plateau-Alternative ist auch eine Via-Alternative
  - **Beweis:**
    - Sei  $P_{s,t} = s \rightarrow y \rightarrow x \rightarrow t$  die Plateau-Alternative
    - Für jeden Knoten  $v \in P_{y,x}$  ist  $P_{s,t}$  eine Via-Alternative mit Via-Knoten  $v$ , da  $P_{s,v}$  und  $P_{v,t}$  kürzeste Pfade sind



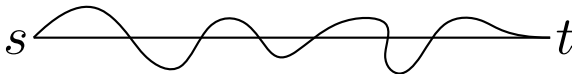
## Idee:

- Suche kürzesten Pfad  $P_1$
- Simuliere Stau auf  $P_1$
- Suche neuen kürzesten Pfad  $P_2$  der Stau in Betracht zieht
- Simuliere Stau auf  $P_1$  und  $P_2$
- Suche neuen kürzesten Pfad  $P_3$  der Stau in Betracht zieht
- ...
- Wiederhole bis Alternativen zu lang werden

## Idee:

- Suche kürzesten Pfad  $P_1$
  - Simuliere Stau auf  $P_1$
  - Suche neuen kürzesten Pfad  $P_2$  der Stau in Betracht zieht
  - Simuliere Stau auf  $P_1$  und  $P_2$
  - Suche neuen kürzesten Pfad  $P_3$  der Stau in Betracht zieht
  - ...
  - Wiederhole bis Alternativen zu lang werden
- 
- Englischer Fachbegriff: Penalty Method
  - Quelle: [BDGS11]

Ist das eine gute Alternative?



- Nicht lokal optimal
- Bestrafe nicht nur die Kanten auf dem Weg, sondern auch die inzidenten Kanten.

## Strafterme

- Einfluss von additiven Straftermen hängt vom Graphen ab  
mehr Grad-2 Knoten  $\rightarrow$  größerer Einfluss
- **Lösung:** Multiplikative Strafterme
- Anders formuliert: Setzt die Geschwindigkeit gleichmäßig runter  
anstatt jede Kante um  $x$  Sekunden zu verlängern

## Wie implementieren?

- Benutze 3-Phasen Vorberechnungstechnik
- Nur Teile der Metrik ändern sich → kann ausgenutzt werden
- [KRS13] baut auf MLD/CRP auf

algorithm	first		second		third	
	time [ms]	success rate [%]	time [ms]	success rate [%]	time [ms]	success rate [%]
X-BDV	11 451.5	94.5	12 225.9	80.6	13 330.9	59.6
CRP- $\pi$	130.0	96.3	130.0	84.0	130.0	62.9
X-REV	20.4	91.3	33.6	70.3	42.6	43.0
HiDAR	18.2	91.5	18.2	75.7	18.2	55.9
X-CHV-v1	16.9	90.7	20.3	70.1	22.1	42.3
X-CHV-v2	3.1	58.2	3.6	28.6	3.9	10.9

- X-BDV: Bidirektionaler Dijkstra
- X-REV: Reach
- X-CHV-v1 und X-CHV-v2: Zwei CH-Varianten, v1 sucht mehr Kanten außerhalb des Suchraums ab
- HiDAR: Plateau-CH
- CRP- $\pi$ : Simulierter Stau

- Eine gute Alternative ist:
  - Nicht viel länger
  - Lokal-optimal
  - Hinreichend verschieden
- Alternativen findet man durch
  - Kandidaten heuristisch aufzählen
  - Schlechte Kandidaten herausfiltern
- Kandidaten findet man mit
  - Via-Knoten
  - Plateaus
  - Simuliertem Stau



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

Alternative routes in road networks.

*ACM Journal of Experimental Algorithmics*, 18(1):1–17, 2013.



Roland Bader, Jonathan Dees, Robert Geisberger, and Peter Sanders.

Alternative route graphs in road networks.

In *Proceedings of the 1st International ICST Conference on Theory and Practice of Algorithms in (Computer) Systems (TAPAS'11)*, volume 6595 of *Lecture Notes in Computer Science*, pages 21–32. Springer, 2011.



Moritz Kobitzsch.

HiDAR: An alternative approach to alternative routes.

In *Proceedings of the 21st Annual European Symposium on Algorithms (ESA'13)*, volume 8125 of *Lecture Notes in Computer Science*, pages 613–624. Springer, 2013.



Moritz Kobitzsch, Marcel Radermacher, and Dennis Schieferdecker.

Evolution and evaluation of the penalty method for alternative graphs.

In *Proceedings of the 13th Workshop on Algorithmic Approaches for Transportation Modeling, Optimization, and Systems (ATMOS'13)*, OpenAccess Series in Informatics (OASISs), pages 94–107, 2013.