Karlsruher Institut für Technologie

Algorithmen für Routenplanung

21. Vorlesung, Sommersemester 2018

Tobias Zündorf | 18. Juli 2018

Institut für Theoretische Informatik · Algorithmik · Prof. Dr. Dorothea Wagner

Inhalt heute

Unbeschränktes Laufen:

- Limitierungen von transitiv-abgeschlossenen Graphen
- Auswirkung von vollständigen Fußweggraphen

Verkehrsumlegung:

- Fahrzeugauslastungen bei statistischem Reiseaufkommen

Unbeschränktes Laufen

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Fußwegegraphen

Bisher:

- Effiziente Algorithmen
- Eingeschränkte Modelle für Laufen

Einschränkungen:

RAPTOR	Transitiv abgeschlossen
CSA	Transitiv abgeschlossen
Trip-Based Routing	Transitiv abgeschlossen
Transfer Patterns	Max. 400 m
PTL	Specified by PT-network
MEAT	Kein laufen möglich

Fußwegegraphen

Argumente für Einschränkung:

- Laufen lohnt sich nie
- Laufen vom Nutzer unerwünscht
- Laufen in der Praxis nicht relevant

Aber:

- Relevanz nie bewiesen/widerlegt
- Algorithmus sollte über Relevanz entscheiden

Transitiver Abschluss

Frage: Bis zu welcher Länge ist Laufen praktikabel?

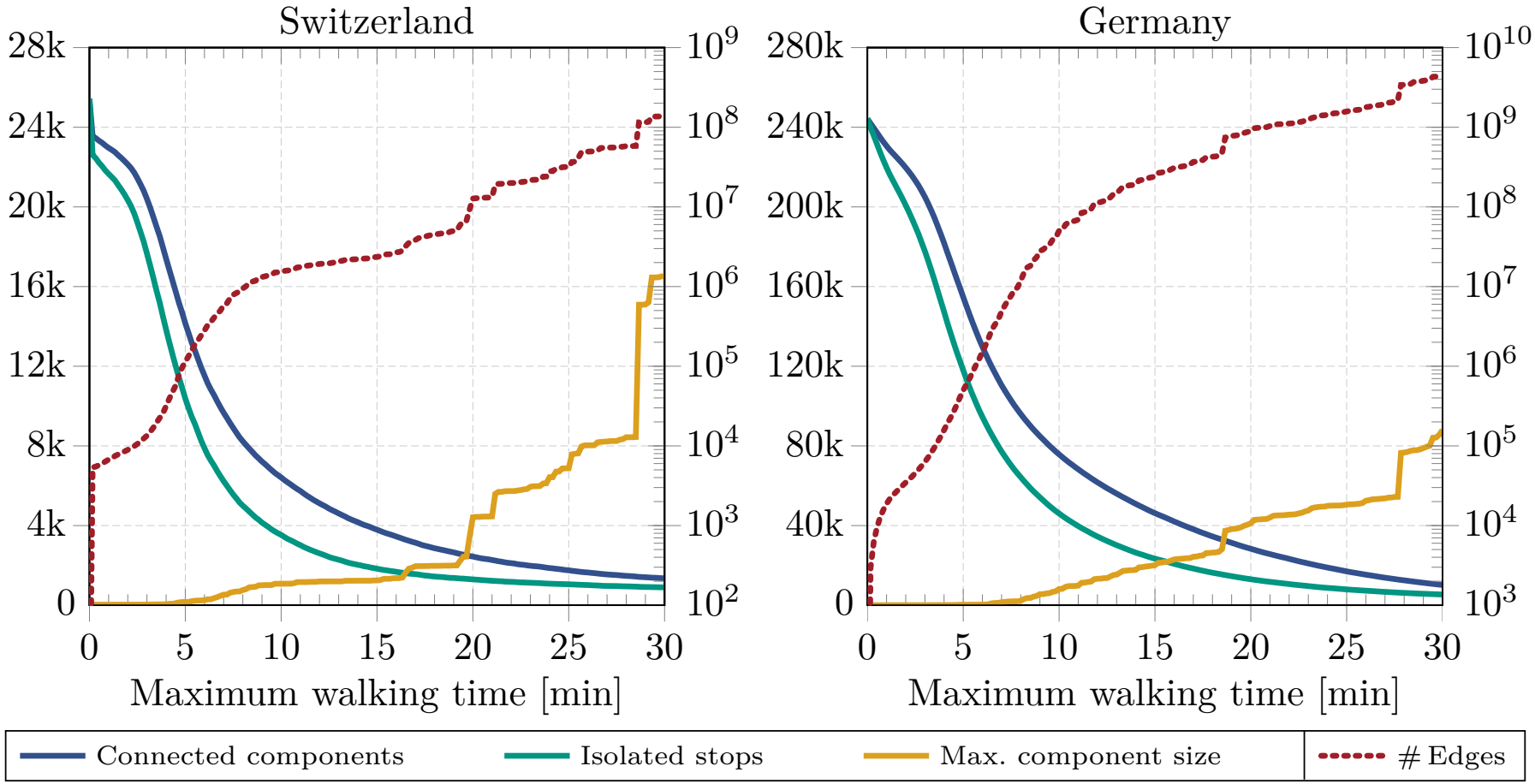
Ansatz:

- Starte mit vollständigem Fußweggraphen
- Wähle maximale Laufzeit τ
- Verbinde Stops $\Leftrightarrow$ Laufdistanz $\leq \tau$
- Bilde transitiven Abschluss des resultierenden Graphen

- Wie groß wird dieser Graph?

Transitiver Abschluss

Größe des transitiven Abschlusses:



Fazit

Beobachtung:

- Transitiver Abschluss nur für kurze Laufwege praktikabel
- Somit bestehende Algorithmen nicht anwendbar

Fazit

Beobachtung:

- Transitiver Abschluss nur für kurze Laufwege praktikabel
- Somit bestehende Algorithmen nicht anwendbar

Frage: Machen längere Laufwege in der Praxis einen unterschied?

Probleme:

- Hängt von der Tageszeit ab (Tag vs. Nacht)
- Hängt von der Distanz ab (lang vs. kurz)
- Hängt von Start und Ziel ab (ländlich vs. städtisch)

Fazit

Beobachtung:

- Transitiver Abschluss nur für kurze Laufwege praktikabel
- Somit bestehende Algorithmen nicht anwendbar

Frage: Machen längere Laufwege in der Praxis einen unterschied?

Probleme:

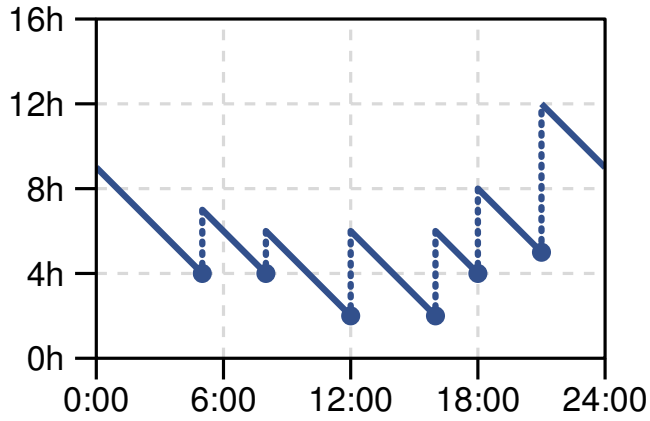
- Hängt von der Tageszeit ab (Tag vs. Nacht)
- Hängt von der Distanz ab (lang vs. kurz)
- Hängt von Start und Ziel ab (ländlich vs. städtisch)

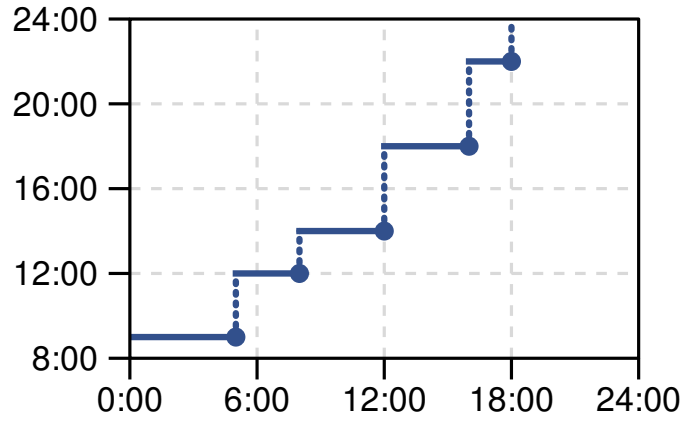
$\Rightarrow$ Auswertung von Reisezeitprofilen
Auf Instanzen mit unterschiedlich vielen Fußwegen
Für viele s-t-Paare

Profile & Profil-Algorithmen

Wiederholung Profil:

- Funktion die Abfahrtszeit auf Reisezeit oder Ankunftszeit abbildet





Profil-Algorithmen für unbeschränktes Laufen:

- Die meisten Public Transit Algorithmen sind nicht geeignet
- Ausnahme: Multimodal Multicriteria RAPTOR (MCR)
  - Kann mit beliebigem Laufen umgehen
  - Unterstützt aber nur Earliest-Arrival-Queries

Profile – RAPTOR & MCR

MCR Profile Algorithmus?:

- MCR basiert auf dem RAPTOR Algorithmus
- RAPTOR kann für Profilberechnung angepasst werden (rRAPTOR)
- Warum ist dies nicht auf MCR übertragbar?

Profile – RAPTOR & MCR

MCR Profile Algorithmus?:

- MCR basiert auf dem RAPTOR Algorithmus
- RAPTOR kann für Profilberechnung angepasst werden (rRAPTOR)
- Warum ist dies nicht auf MCR übertragbar?

rRAPTOR Profil Algorithmus: (Wiederholung)

- Profileinträge sind durch Abfahrten am Start begrenzt
- Sammel alle Abfahrtszeiten am Start
- Führe RAPTOR für jede Abfahrtszeit einmal aus

Profile – RAPTOR & MCR

MCR Profile Algorithmus?:

- MCR basiert auf dem RAPTOR Algorithmus
- RAPTOR kann für Profilberechnung angepasst werden (rRAPTOR)
- Warum ist dies nicht auf MCR übertragbar?

rRAPTOR Profil Algorithmus: (Wiederholung)

- Profileinträge sind durch Abfahrten am Start begrenzt
- Sammel alle Abfahrtszeiten am Start
- Führe RAPTOR für jede Abfahrtszeit einmal aus

Probleme mit unbeschränktem laufen / MCR:

- Beliebiges Laufen vor der ersten Fahrt möglich
- Jede Abfahrt im Netzwerk kann die erste genutzte sein

$\Rightarrow$ Zu viele RAPTOR Anfragen

Neuer Algorithmus – fbRAPTOR

Ziel:

- Nutze Earliest-Arrival-Algorithmus als Black Box (MCR)
- Earliest-Arrival-Algorithmus berechnet einzelne Profileinträge
- Anzahl Ausführungen $\approx$ Anzahl Profileinträge

Neuer Algorithmus – fbRAPTOR

Ziel:

- Nutze Earliest-Arrival-Algorithmus als Black Box (MCR)
- Earliest-Arrival-Algorithmus berechnet einzelne Profileinträge
- Anzahl Ausführungen $\approx$ Anzahl Profileinträge

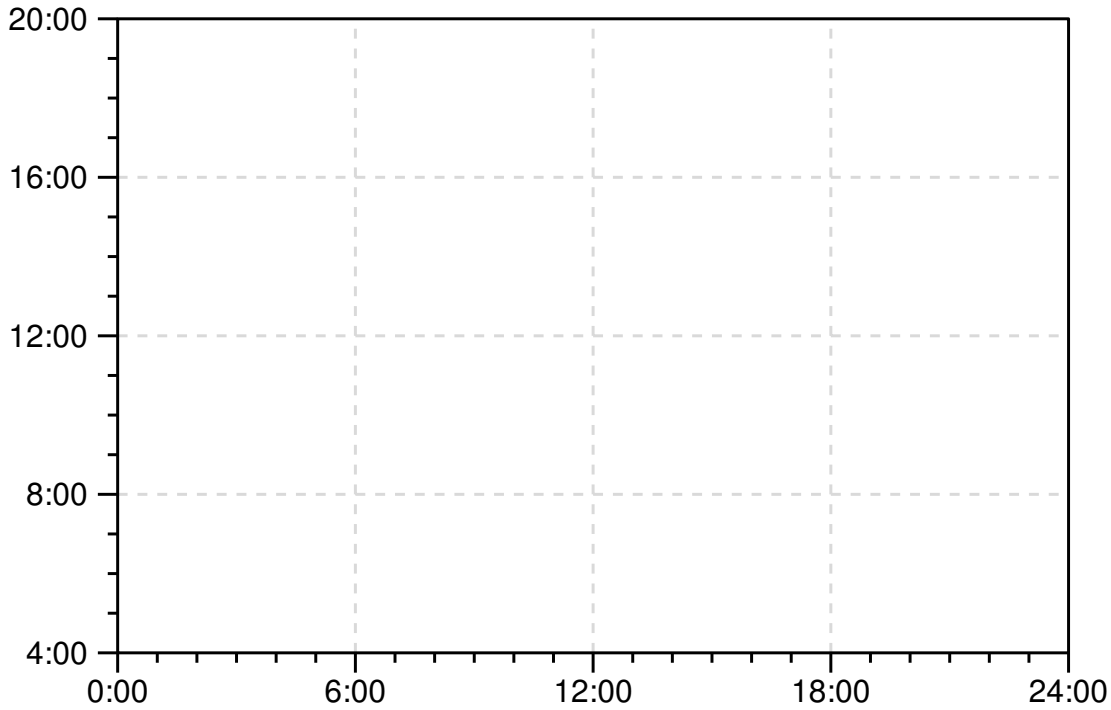
Vorgehen:

- Baue das Profil Eintrag um Eintrag auf
- Pro Profileintrag werden zwei Black Box Aufrufe genutzt:
  - Starte mit frühest möglicher Abfahrtszeit
  - Vorwärts Suche $\Rightarrow$ Frühest mögliche Ankunftszeit
  - Rückwärts Suche $\Rightarrow$ Spätest mögliche Abfahrtszeit
  - Fahre mit der spätest möglichen Abfahrtszeit + ε fort

fbRAPTOR Beispiel

Ziel:

- Berechne ein Profil für das Intervall [0:00, 24:00]

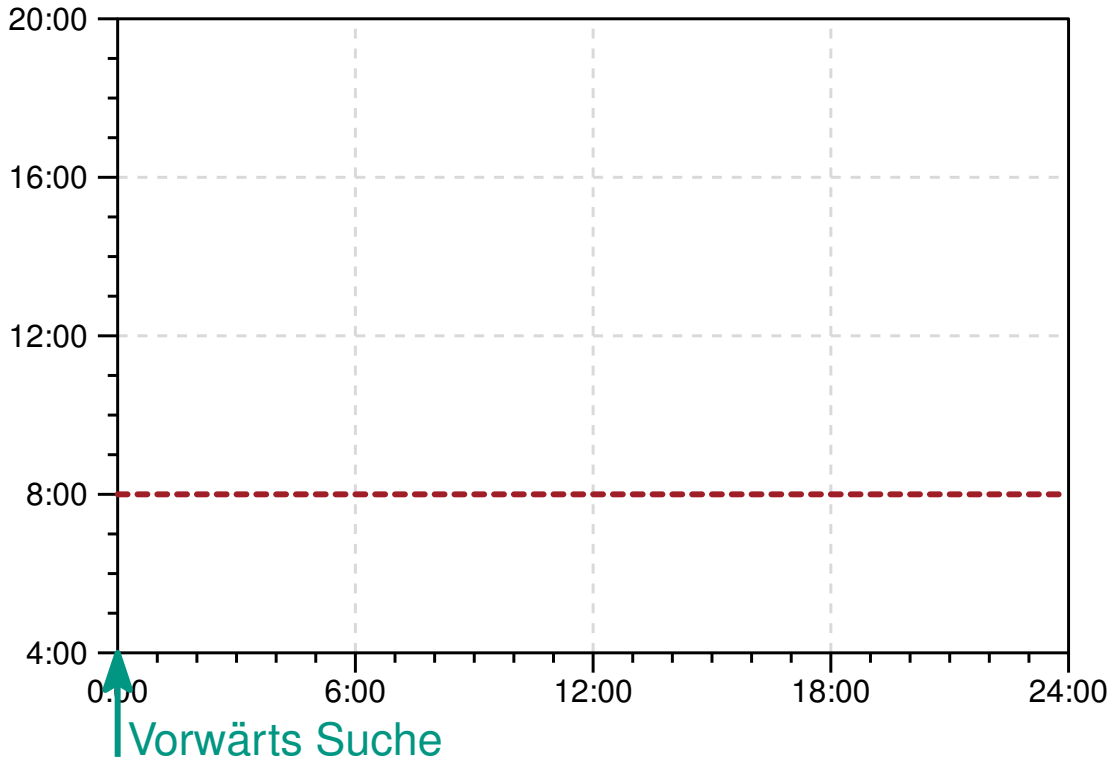


fbRAPTOR Beispiel

Ziel:

- Berechne ein Profil für das Intervall [0:00, 24:00]

Schritt 1: Vorwärts Suche für frühest mögliche Abfahrtszeit (0:00)

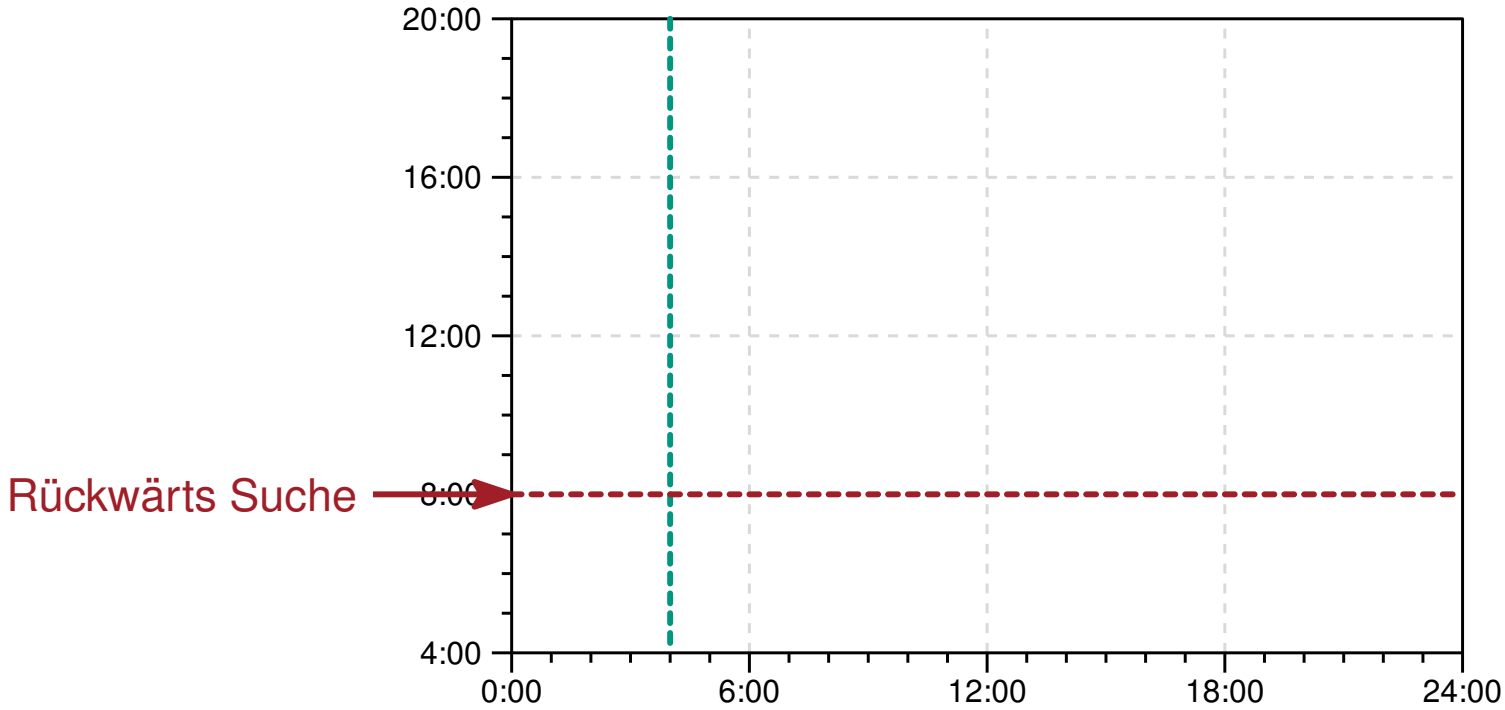


fbRAPTOR Beispiel

Ziel:

- Berechne ein Profil für das Intervall [0:00, 24:00]

Schritt 2: Rückwärts Suche für die resultierende Ankunftszeit (8:00)

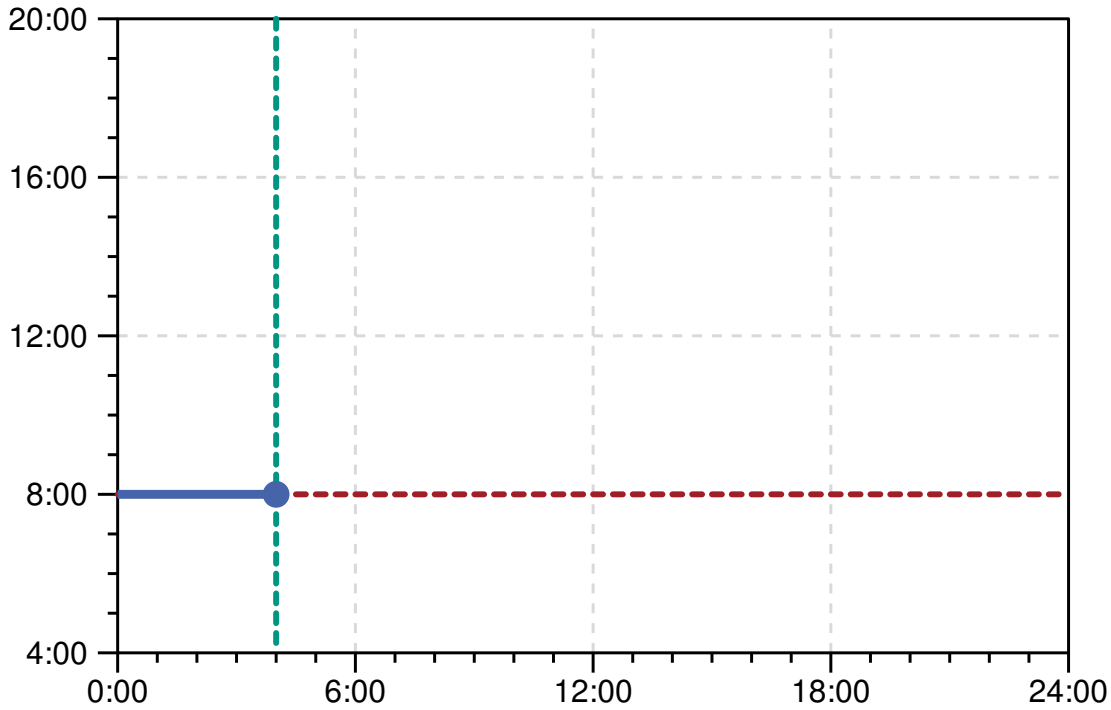


fbRAPTOR Beispiel

Ziel:

- Berechne ein Profil für das Intervall [0:00, 24:00]

Schritt 3: Beide Ergebnisse zusammen bilden einen Profileintrag

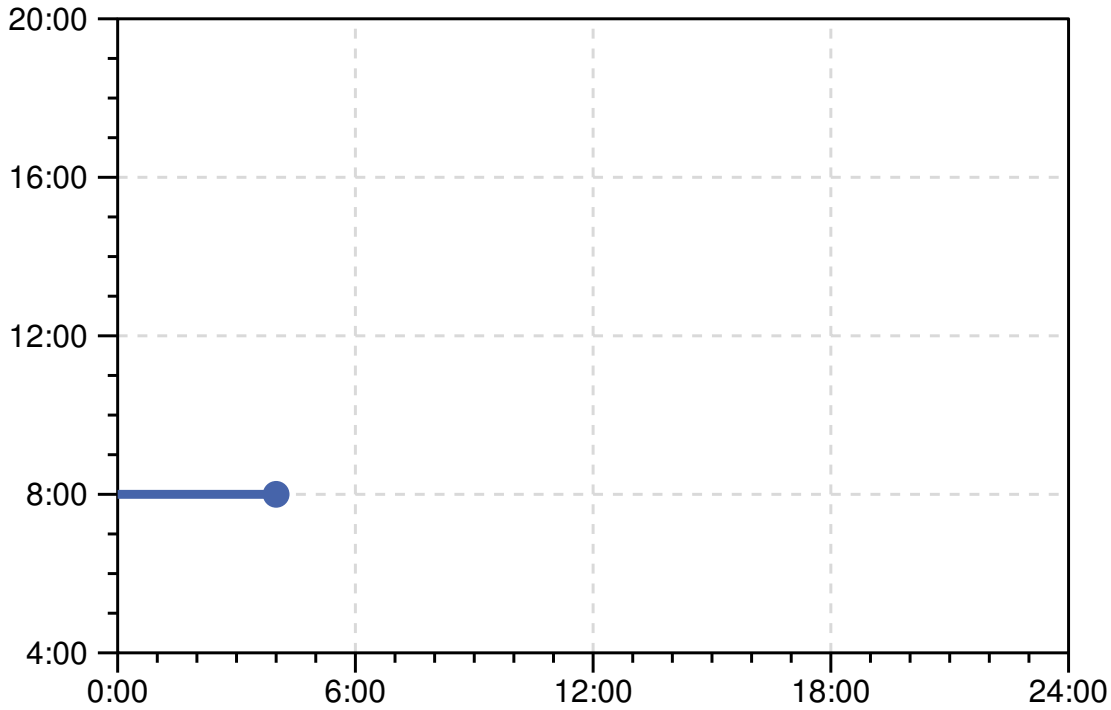


fbRAPTOR Beispiel

Ziel:

- Berechne ein Profil für das Intervall [0:00, 24:00]

Schritt 4: Das Profil is schon korrekt für das Intervall [0:00, 4:00]

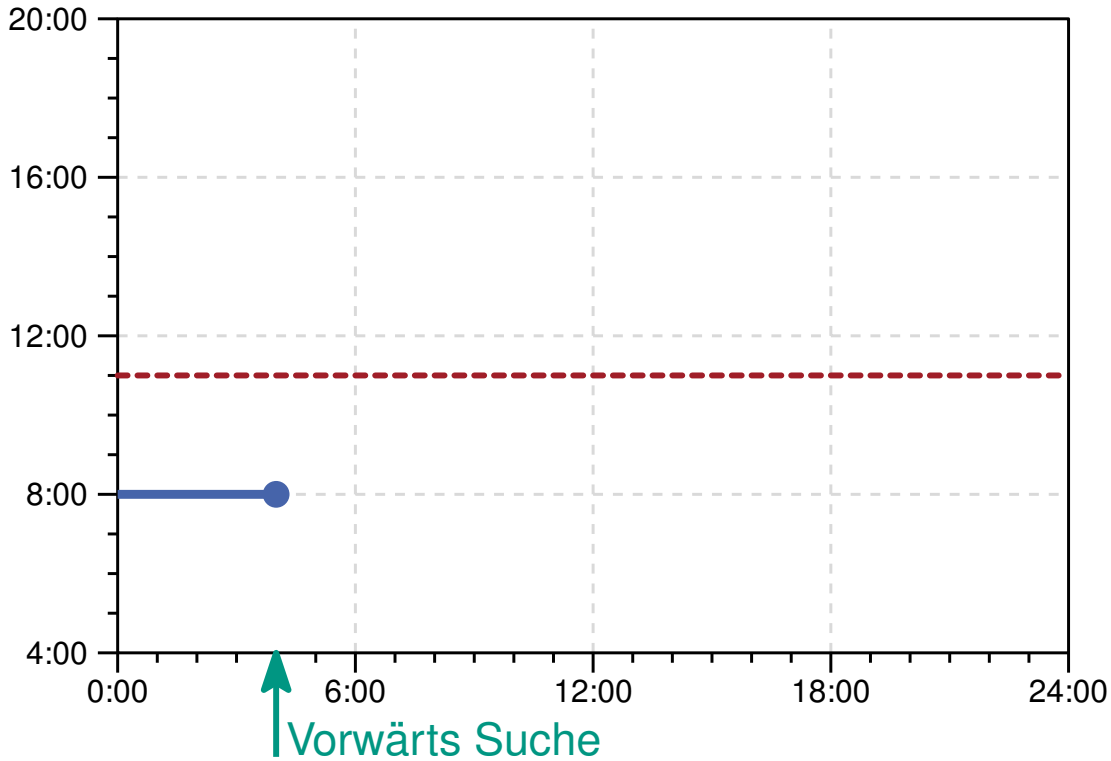


fbRAPTOR Beispiel

Ziel:

- Berechne ein Profil für das Intervall (4:00, 24:00]

Schritt 1: Vorwärts Suche für frühest mögliche Abfahrtszeit ($4{:}00 + \varepsilon$)

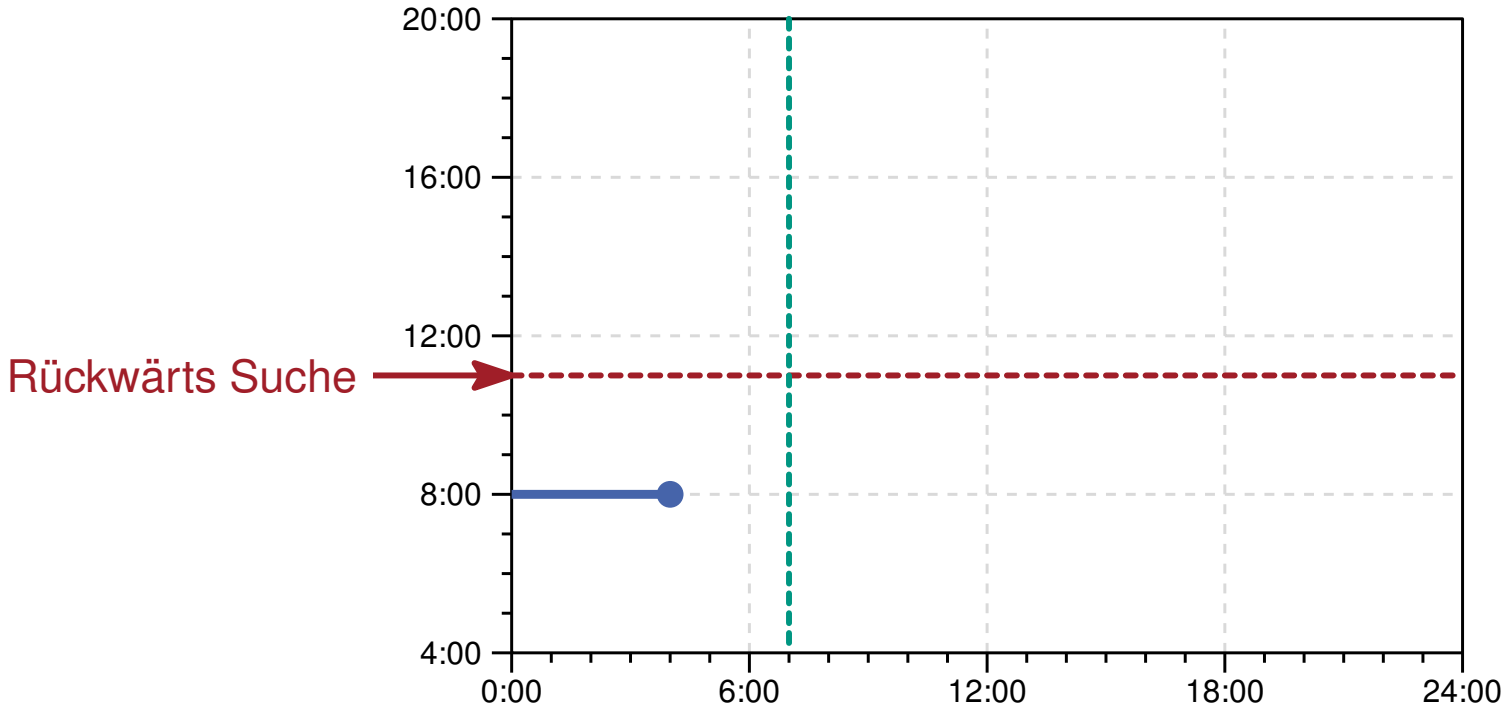


fbRAPTOR Beispiel

Ziel:

- Berechne ein Profil für das Intervall (4:00, 24:00]

Schritt 2: Rückwärts Suche für die resultierende Ankunftszeit (11:00)

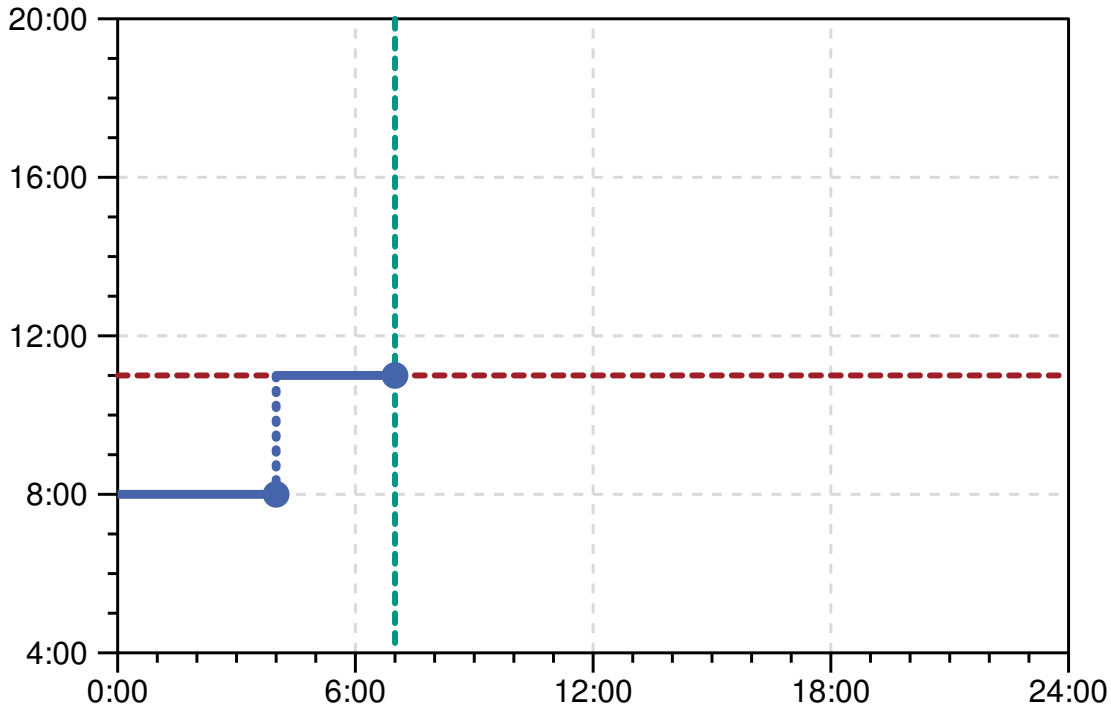


fbRAPTOR Beispiel

Ziel:

- Berechne ein Profil für das Intervall (4:00, 24:00]

Schritt 3: Beide Ergebnisse zusammen bilden einen Profileintrag

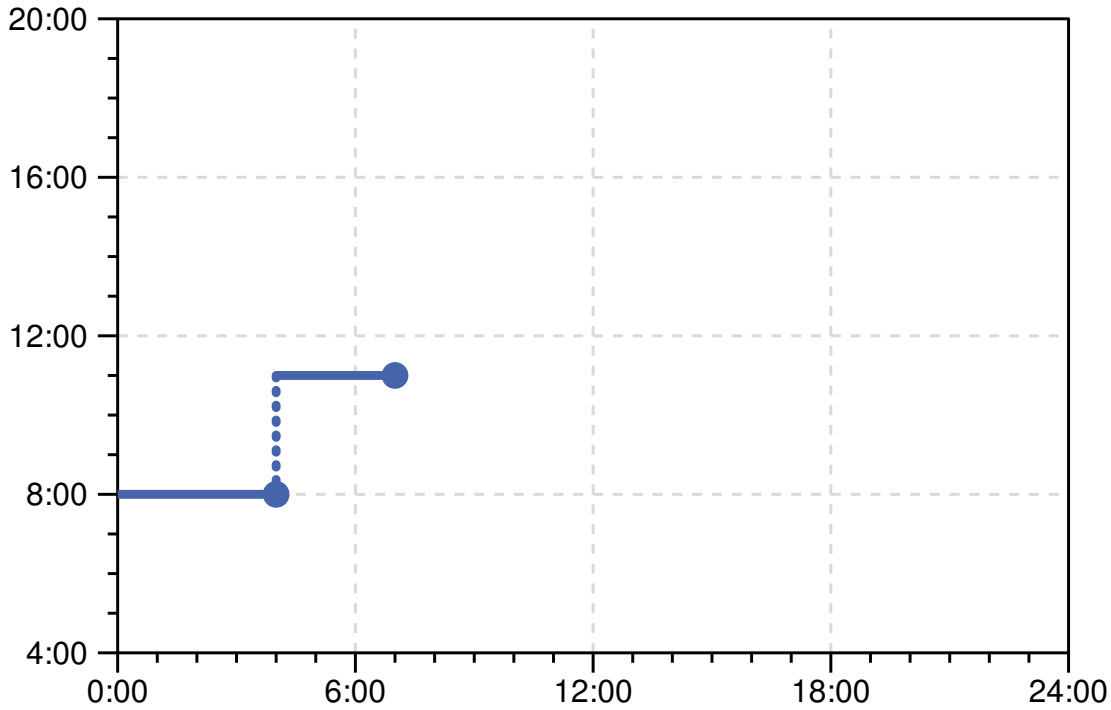


fbRAPTOR Beispiel

Ziel:

- Berechne ein Profil für das Intervall (4:00, 24:00]

Schritt 4: Das Profil is schon korrekt für das Intervall [0:00, 7:00]

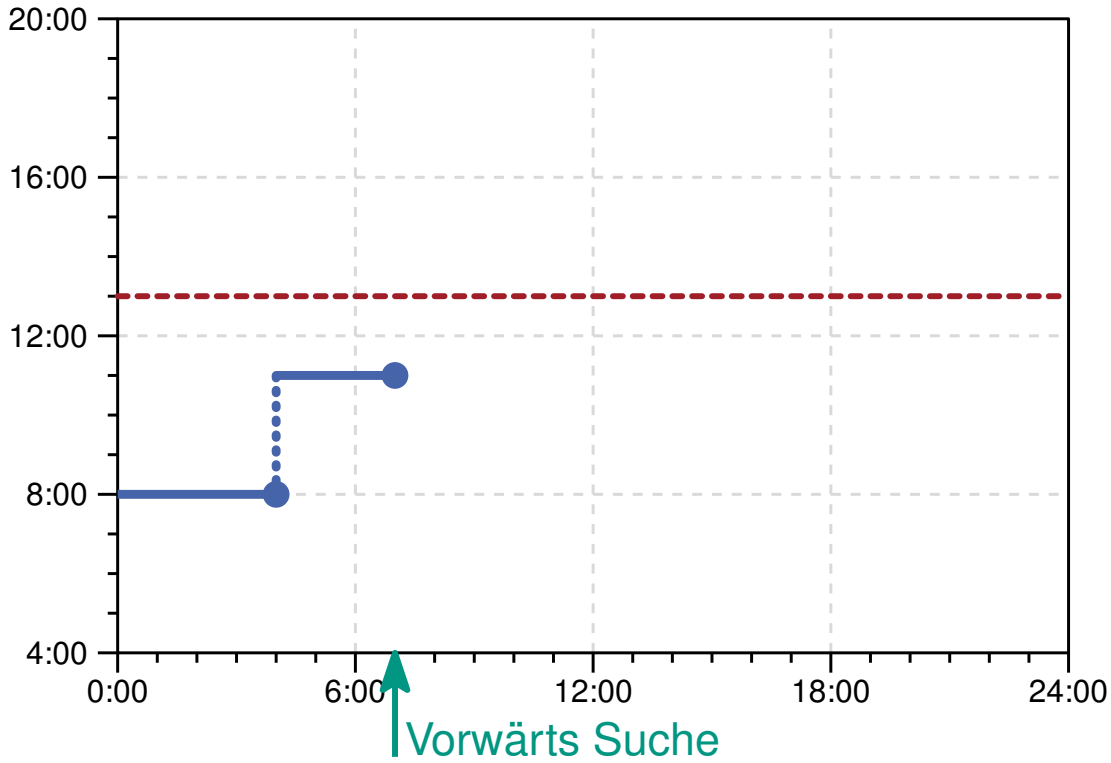


fbRAPTOR Beispiel

Ziel:

- Berechne ein Profil für das Intervall (7:00, 24:00]

Schritt 1: Vorwärts Suche für frühest mögliche Abfahrtszeit (7:00 + ε)

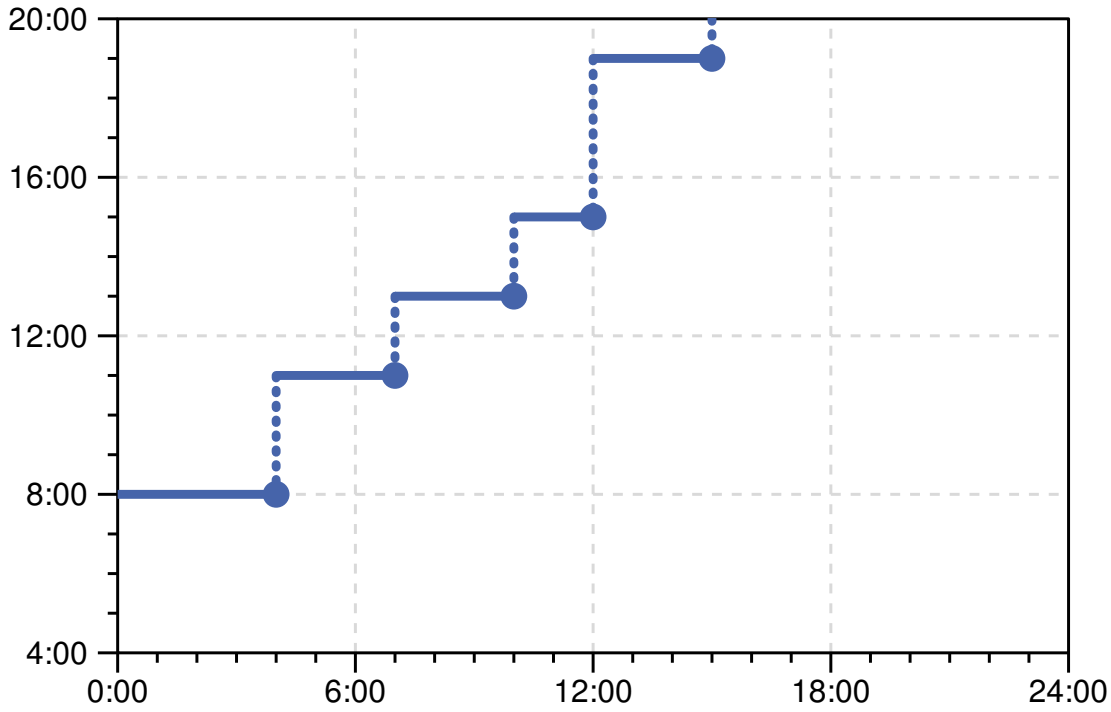


fbRAPTOR Beispiel

Ziel:

- Berechne ein Profil für das Intervall (7:00, 24:00]

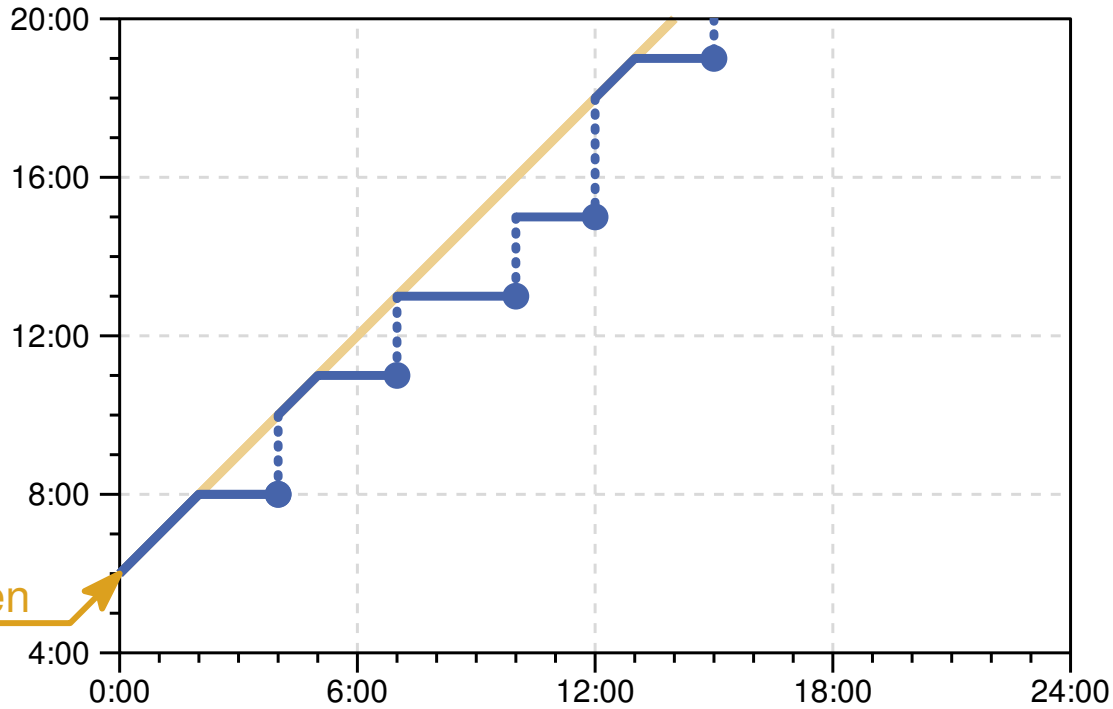
Ende: Der Algorithmus endet, wenn das gesamte Profil berechnet wurde



fbRAPTOR – Unbeschränktes Laufen

Problem:

- Unbeschränktes Laufen verhindert Fortschritt



Beispiel:

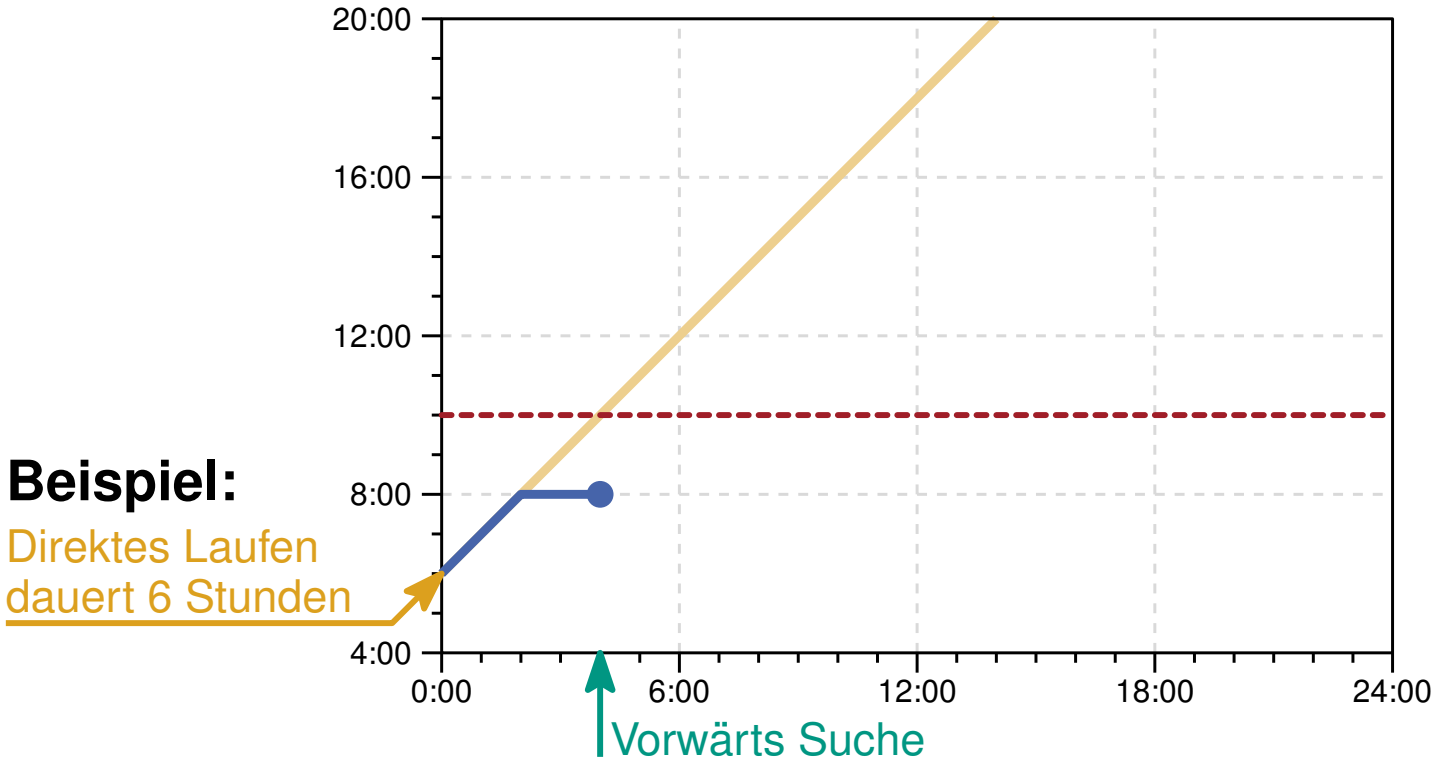
Direktes Laufen dauert 6 Stunden

fbRAPTOR – Unbeschränktes Laufen

Ziel:

- Berechne ein Profil für das Intervall (4:00, 24:00]

Schritt 1: Vorwärts Suche für frühest mögliche Abfahrtszeit (4:00 + ε)

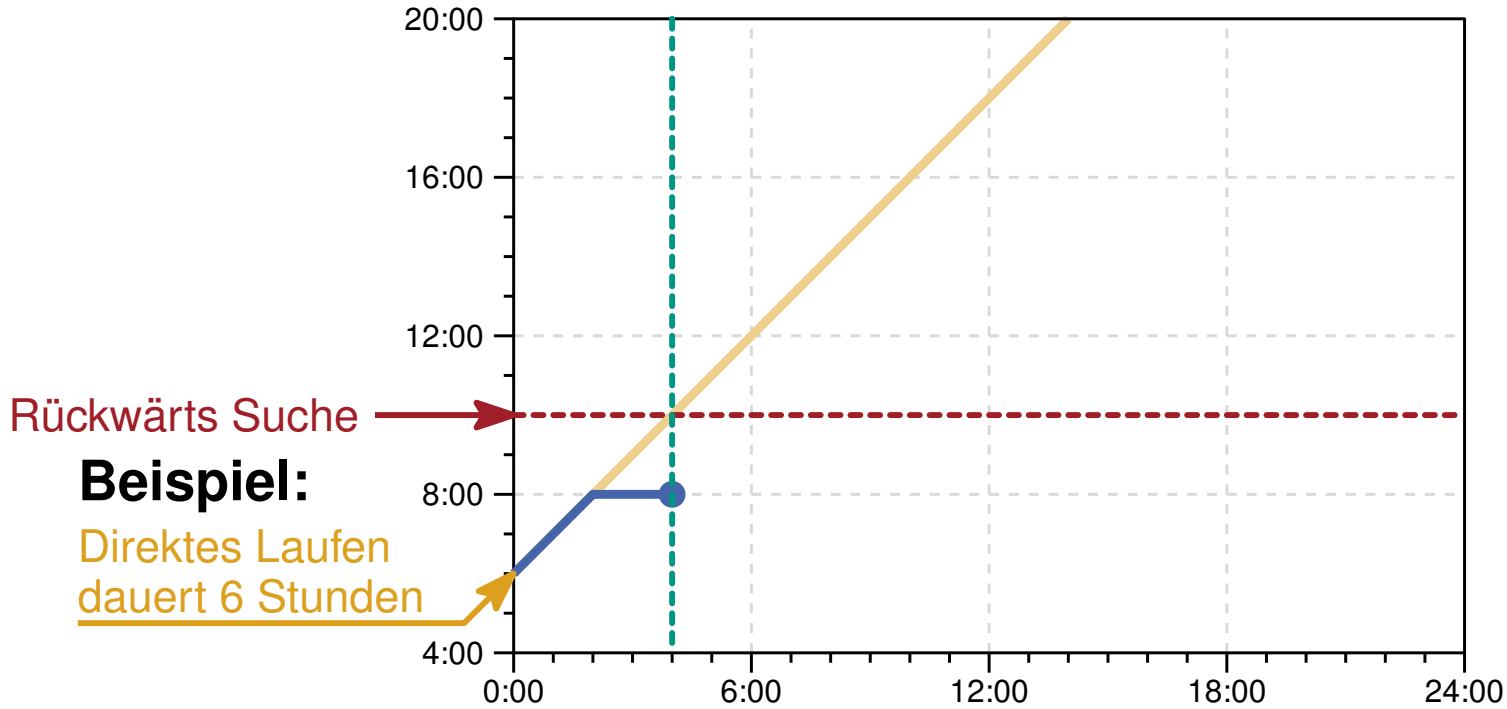


fbRAPTOR – Unbeschränktes Laufen

Ziel:

- Berechne ein Profil für das Intervall (4:00, 24:00]

Schritt 2: Rückwärts Suche für die resultierende Ankunftszeit (10:00)

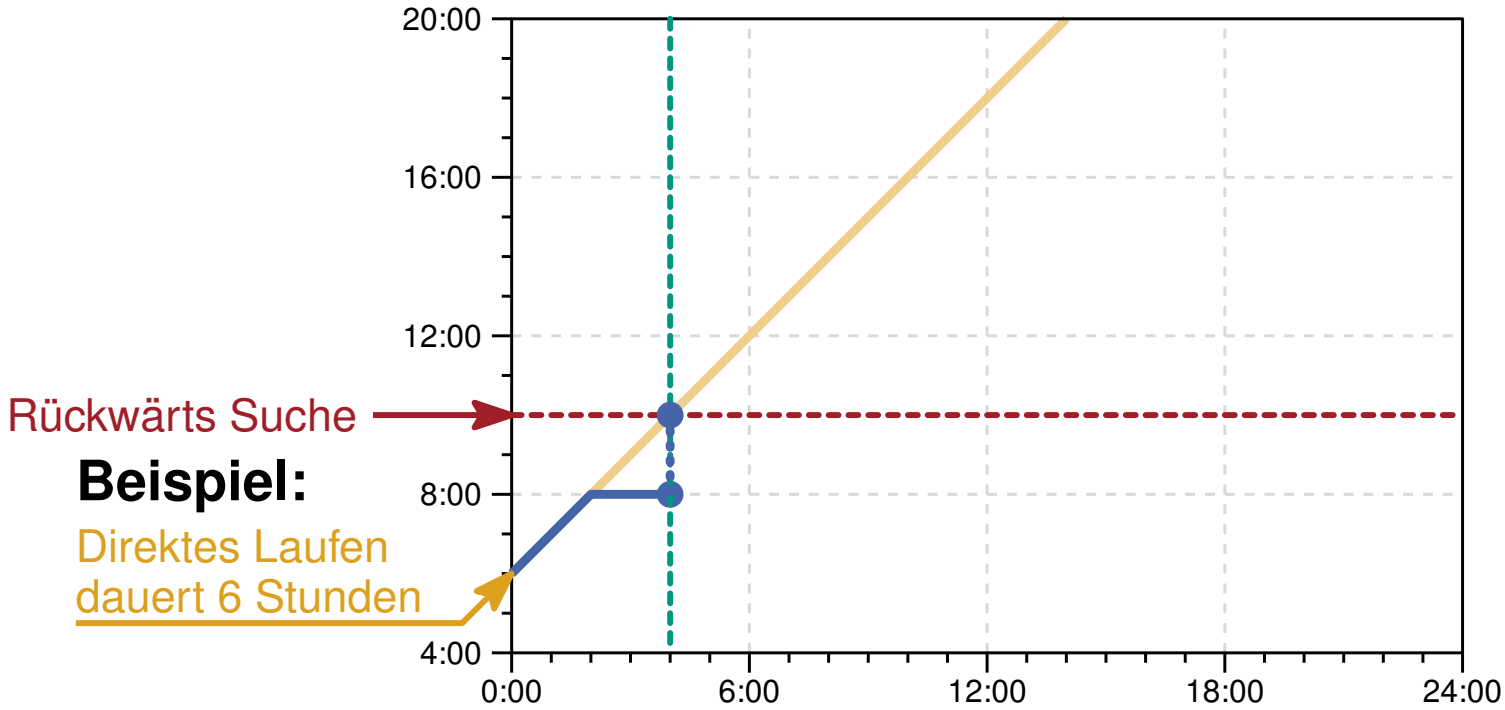


fbRAPTOR – Unbeschränktes Laufen

Ziel:

- Berechne ein Profil für das Intervall (4:00, 24:00]

Schritt 3: Beide Ergebnisse zusammen bilden einen Profileintrag

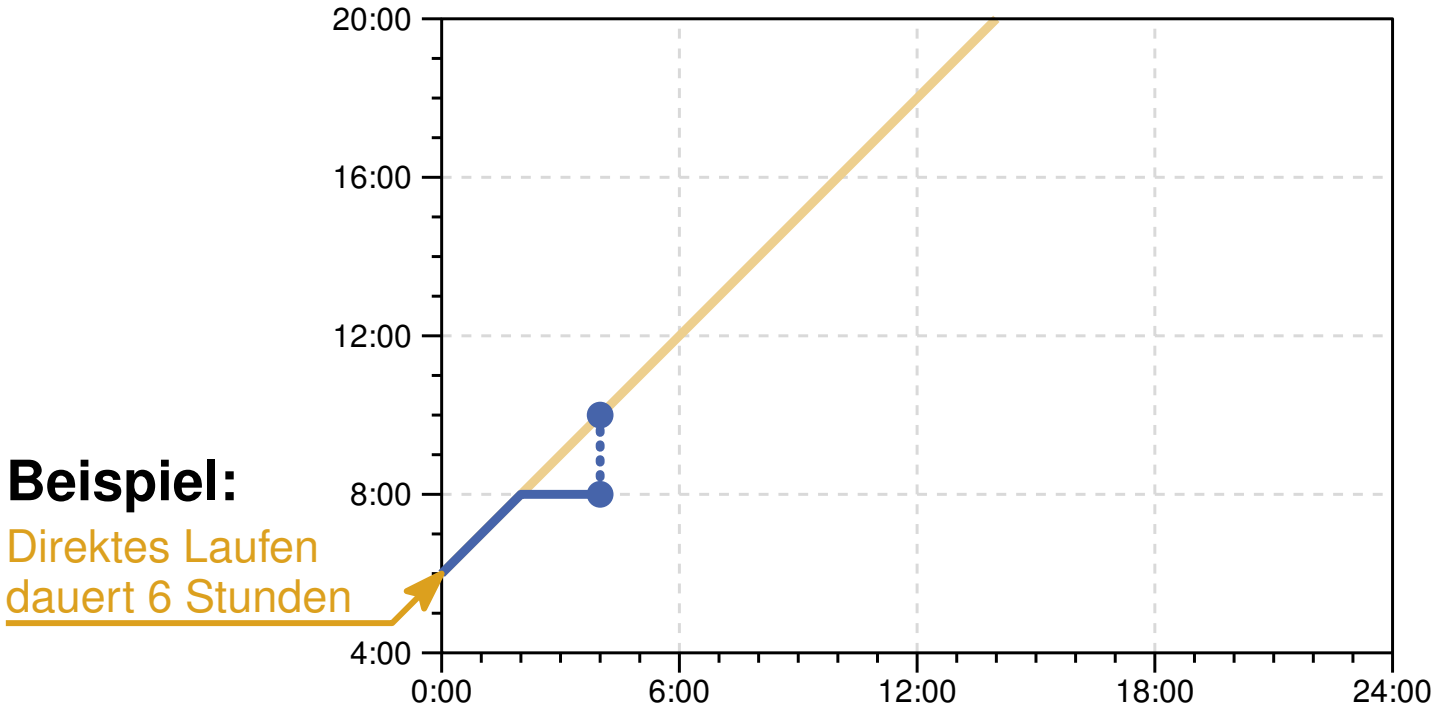


fbRAPTOR – Unbeschränktes Laufen

Ziel:

- Berechne ein Profil für das Intervall (4:00, 24:00]

Schritt 4: Das Profil is immer noch korrekt für das Intervall [0:00, 4:00]

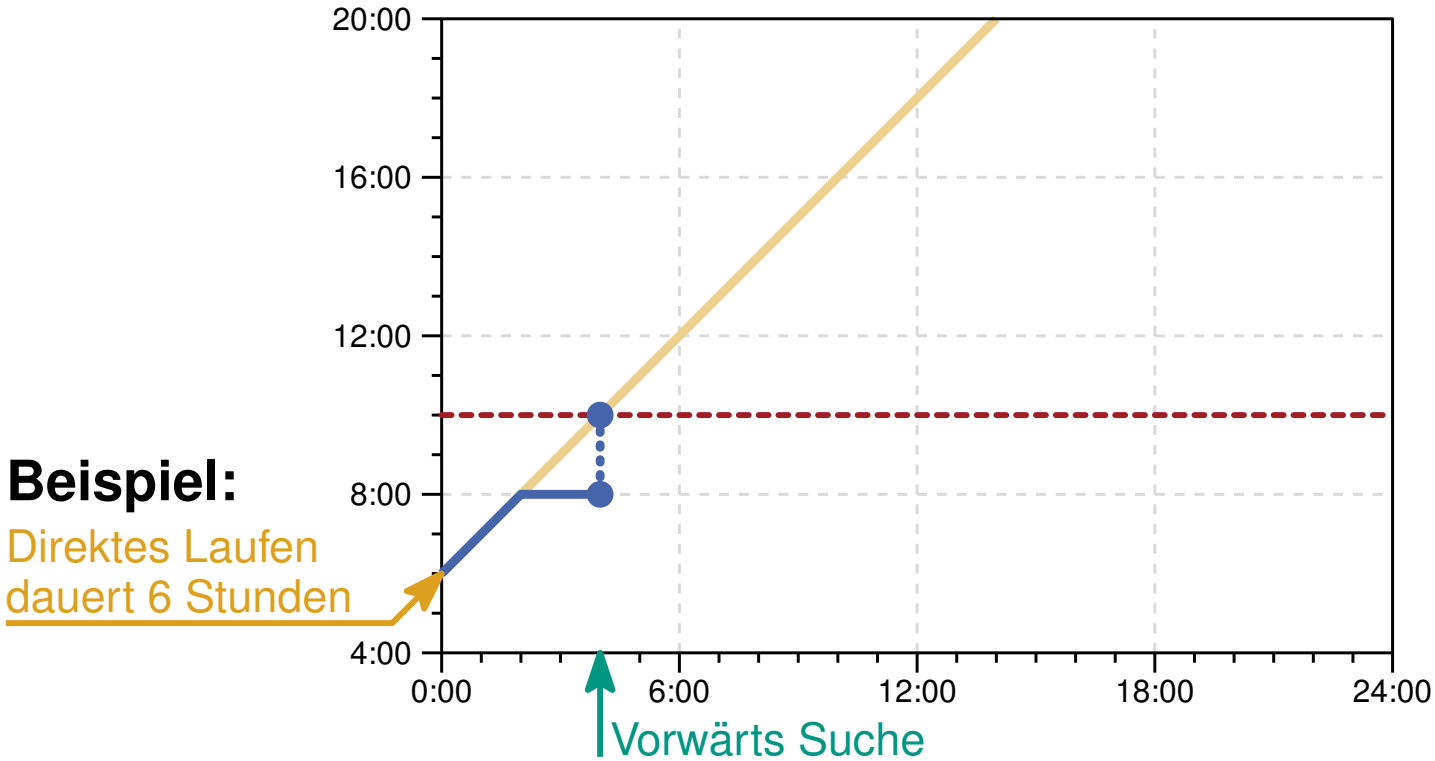


fbRAPTOR – Unbeschränktes Laufen

Ziel:

- Berechne ein Profil für das Intervall (4:00, 24:00]

Schritt 1: Vorwärts Suche für frühest mögliche Abfahrtszeit (4:00 + ε)



fbRAPTOR – Unbeschränktes Laufen

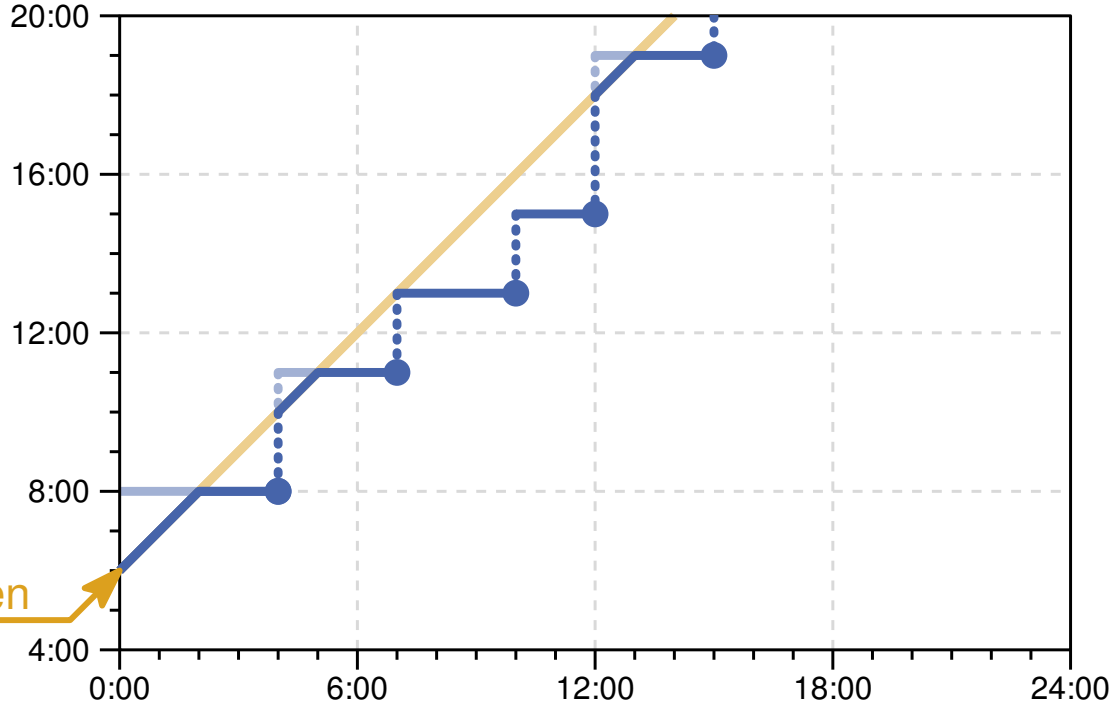
Ziel:

- Berechne ein Profil für das Intervall (4:00, 24:00]

Lösung: Berechne Profil ohne direktes Laufen
& Füge den direkten Laufweg danach hinzu

Beispiel:

Direktes Laufen
dauert 6 Stunden



Auswirkung des Fußwegegraphen

Frage: Machen längere Laufwege in der Praxis einen unterschied?

Ansatz:

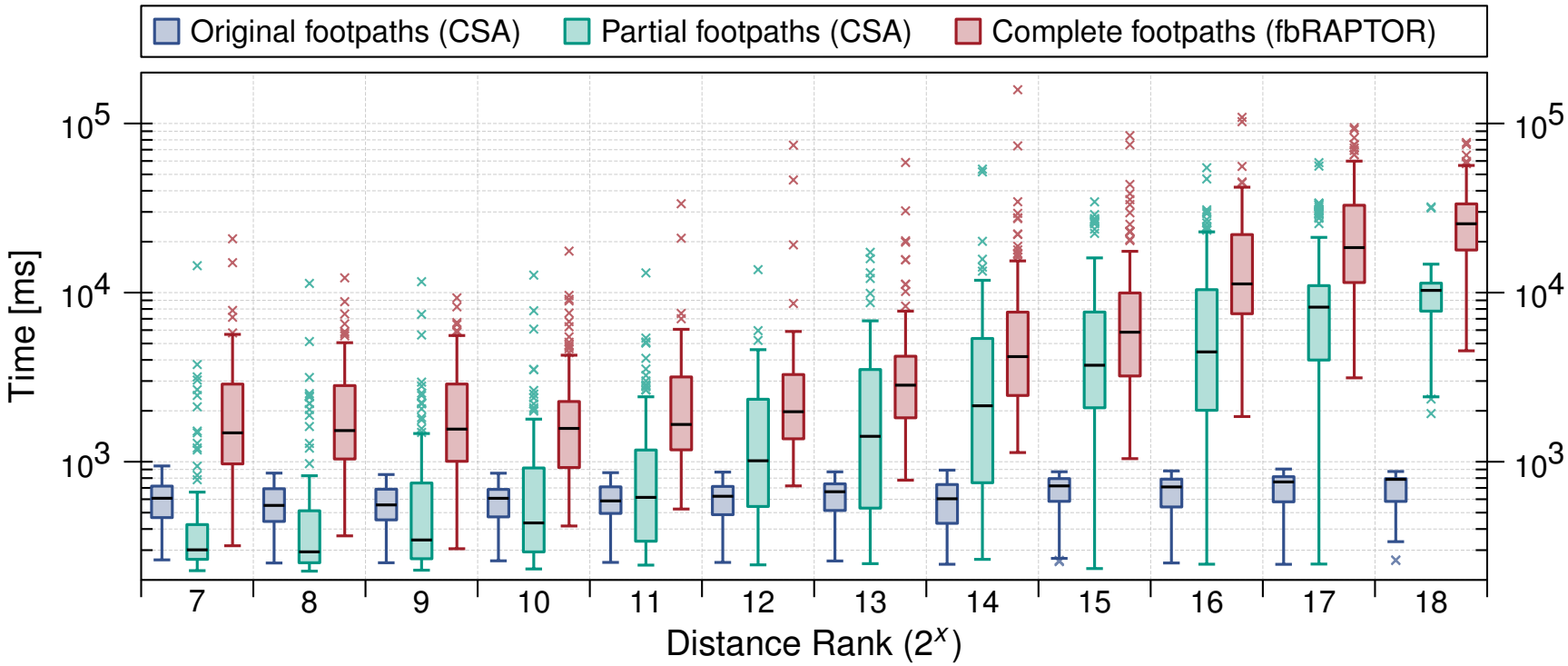
- Vergleiche Reisezeiten in typischen Netzwerken
  - Nur Fußwege die im Public Transit Network spezifiziert sind
  - Transitiv abgeschlossener Fußwegegraph
  - Vollständiger Fußwegegraph

PT network	Footpaths	Stops	Vertices	Edges	Connections	Max. deg.
Switzerland	original	25 427	25 427	5 604	4 373 268	25
	partial	25 427	25 427	3 104 974	4 373 268	1 246
	complete	25 427	604 230	1 844 286	4 373 268	25
Germany	original	244 245	244 245	95 036	46 119 896	18
	partial	244 245	244 245	26 193 136	46 119 896	2 622
	complete	244 245	6 876 758	21 382 408	46 119 896	21

Experimente

Vergleich der Anfragezeiten:

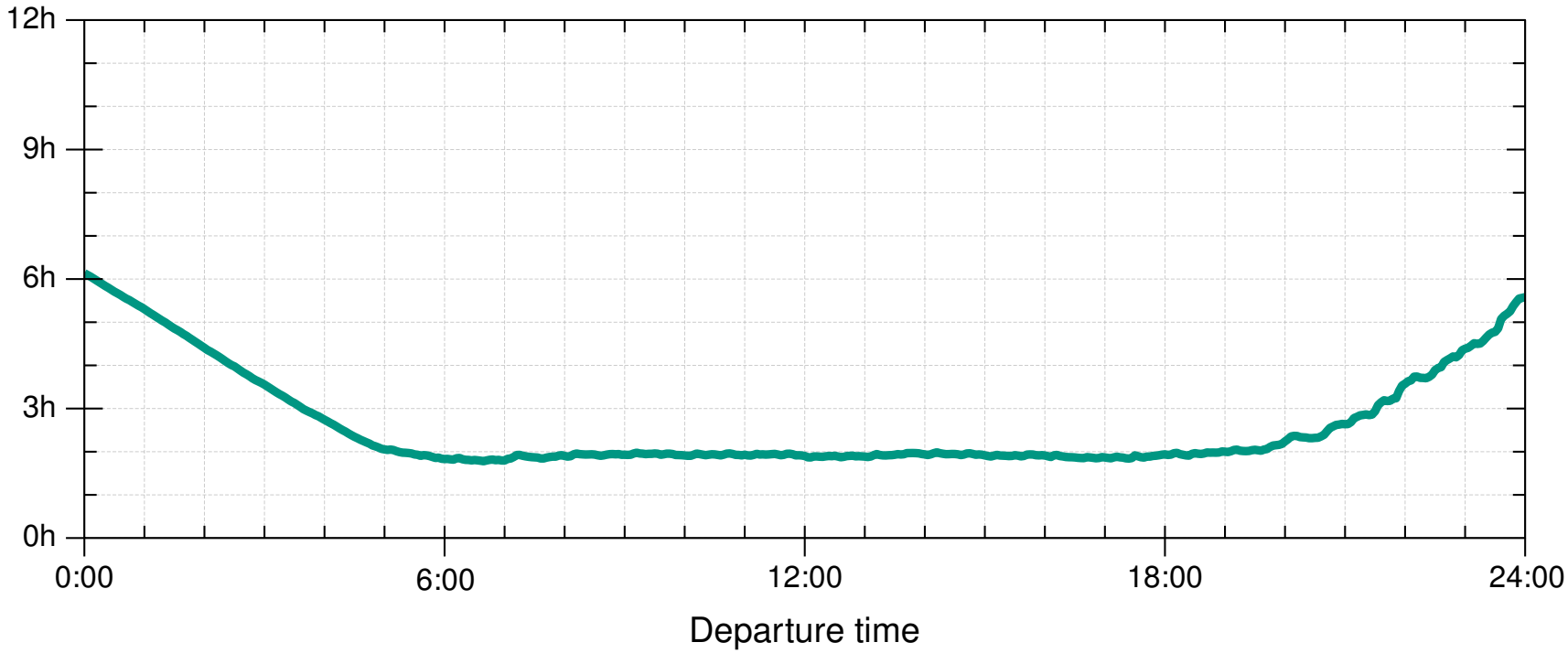
- Auf dem Schweizer Netzwerk
- Für Pareto-Profile (Reisezeit, Anzahl Umstiege)
- CSA berücksichtigt nur partial Fußwege
- fbRAPTOR berücksichtigt complete Fußwege



Experimente

Vergleich der Reisezeiten:

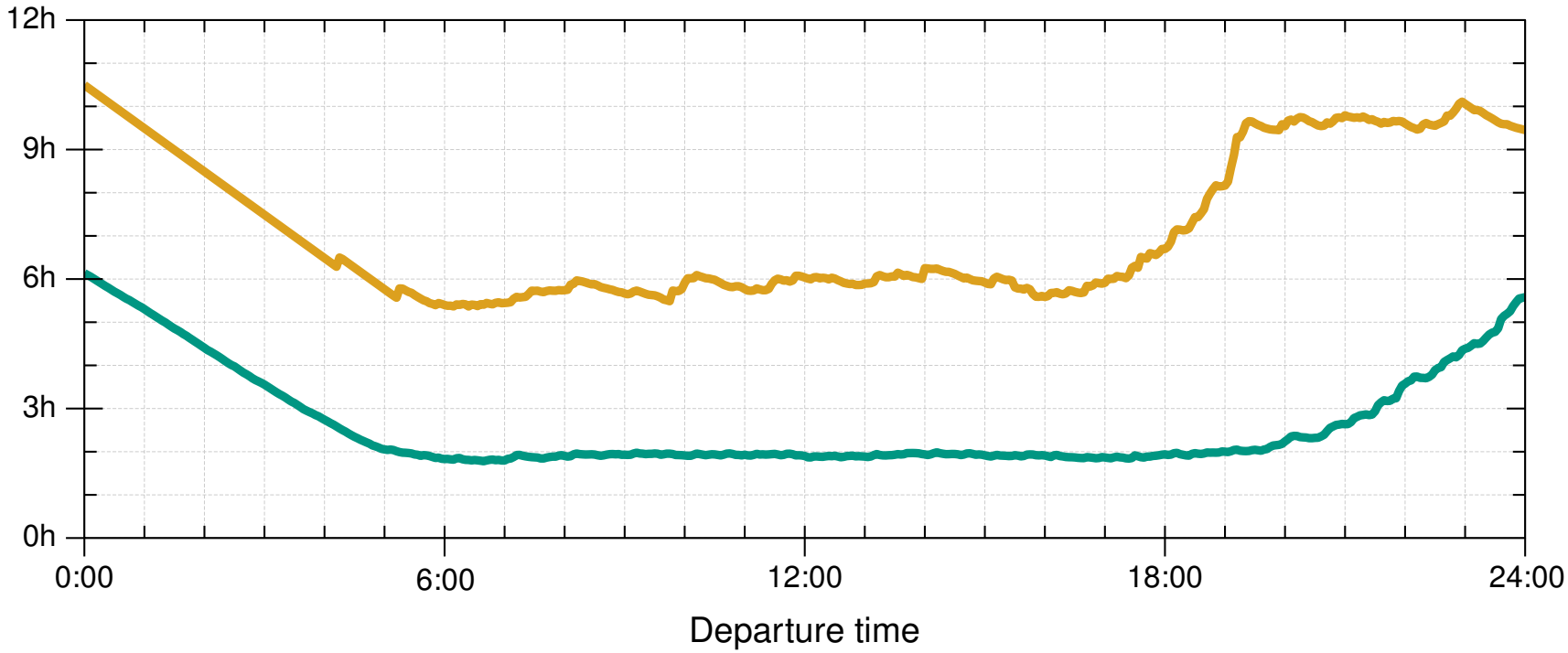
- Switzerland complete vs. Switzerland original (Distance rank 16)



Experimente

Vergleich der Reisezeiten:

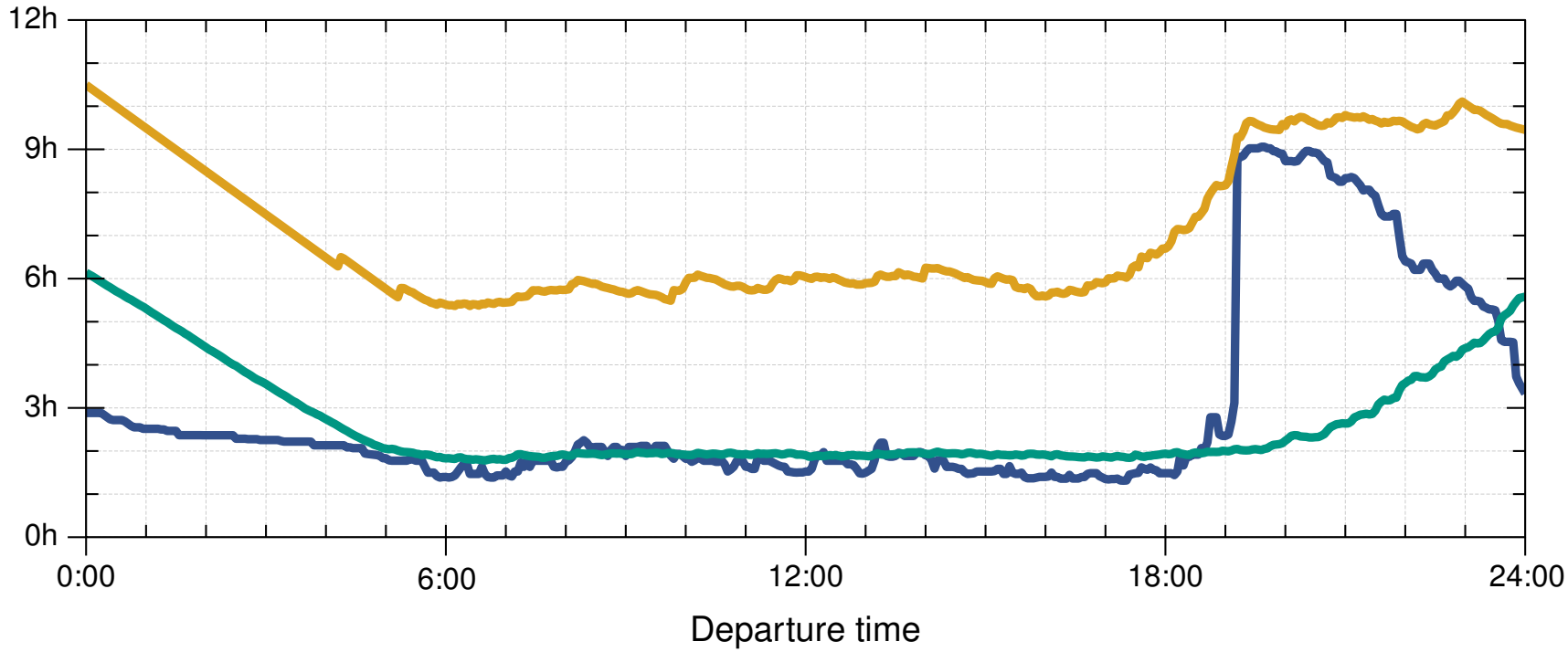
- Switzerland complete vs. Switzerland original (Distance rank 16)



Experimente

Vergleich der Reisezeiten:

- Switzerland complete vs. Switzerland original (Distance rank 16)

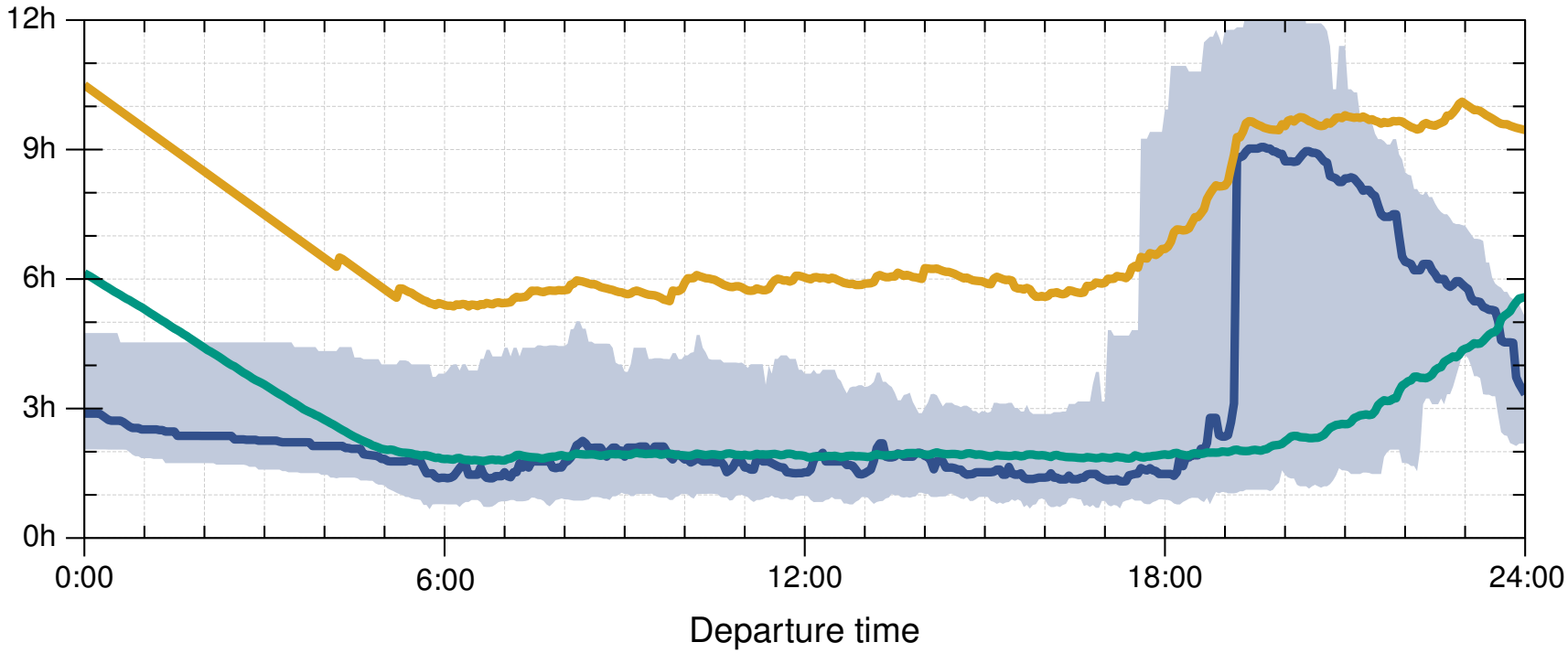


Experimente

Vergleich der Reisezeiten:

- Switzerland complete vs. Switzerland original (Distance rank 16)

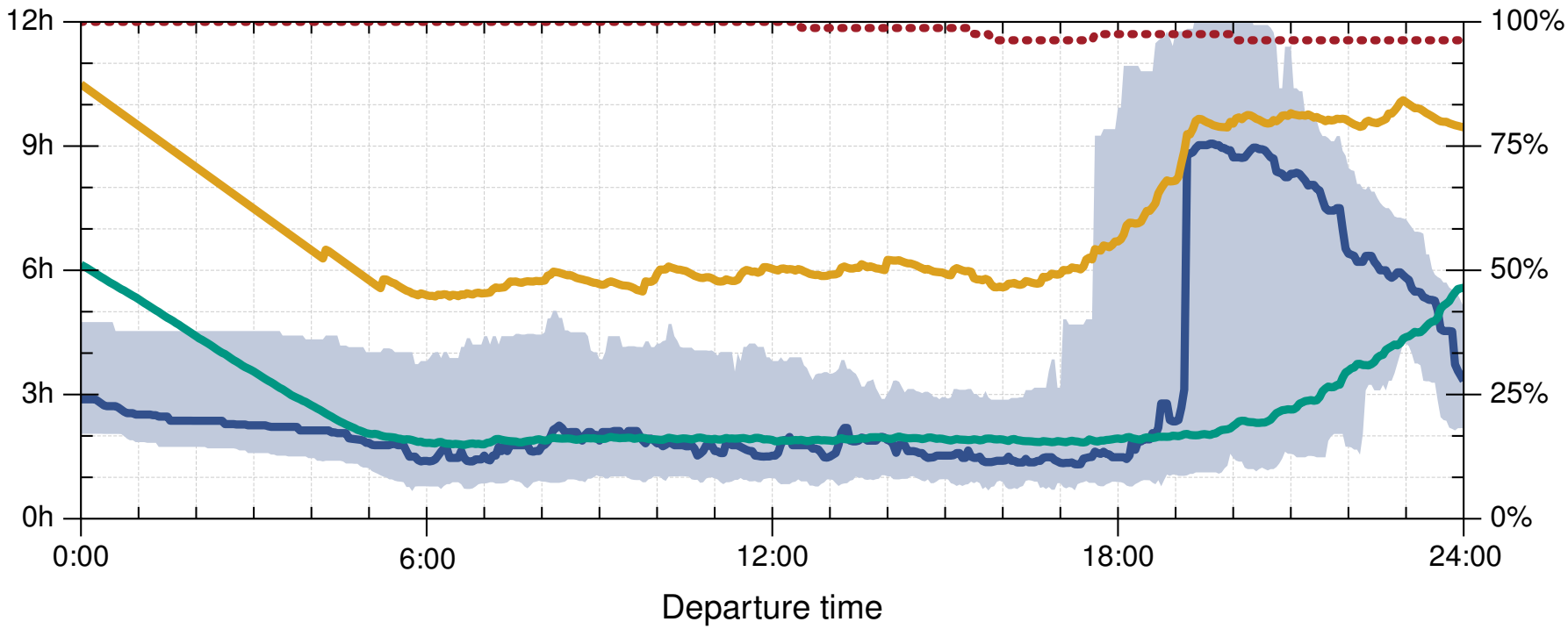
Average travel time (complete)
Average travel time (original)
Median of travel time difference
Interquartile range of travel time diff.



Experimente

Vergleich der Reisezeiten:

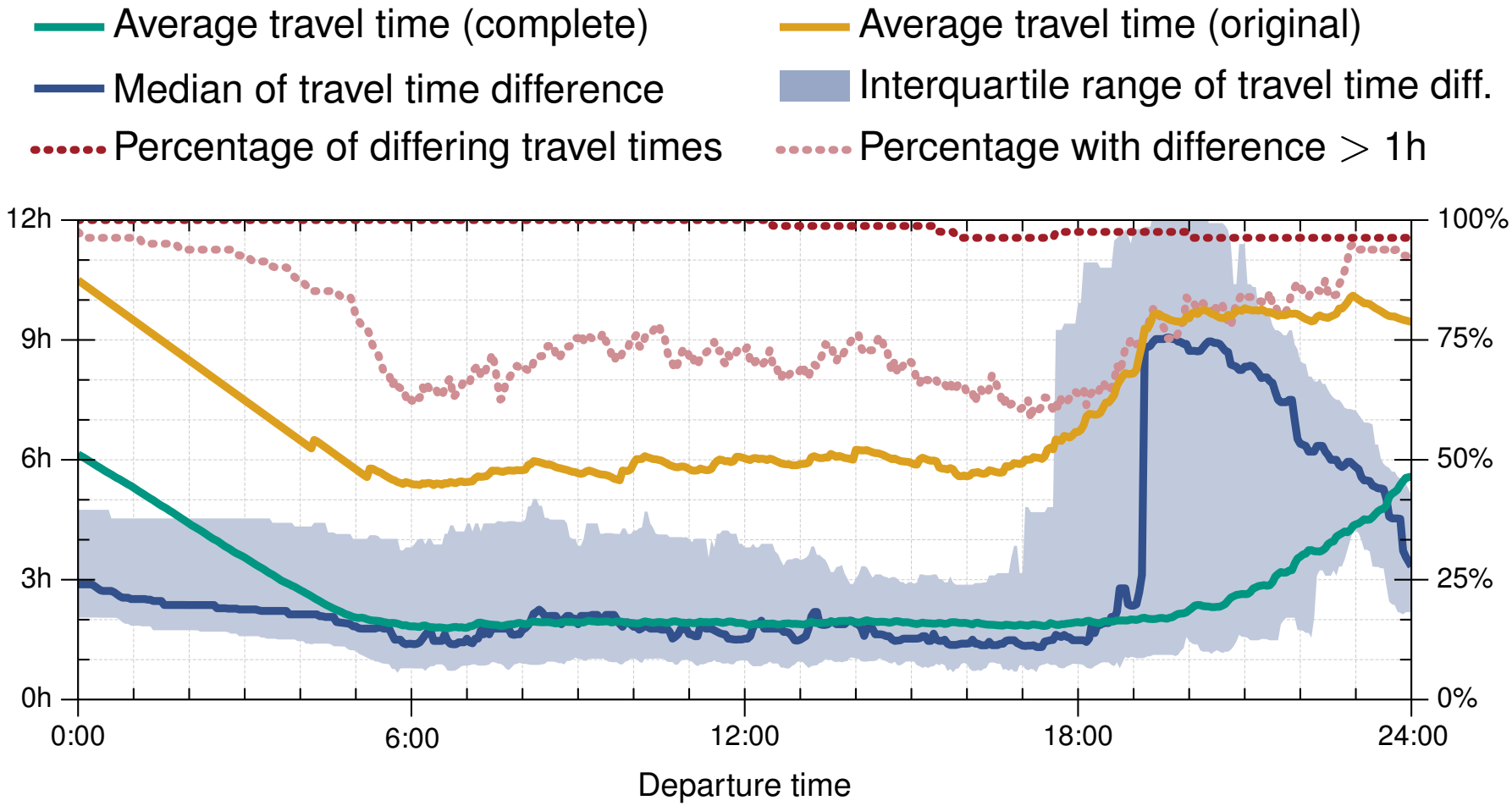
- Switzerland complete vs. Switzerland original (Distance rank 16)



Experimente

Vergleich der Reisezeiten:

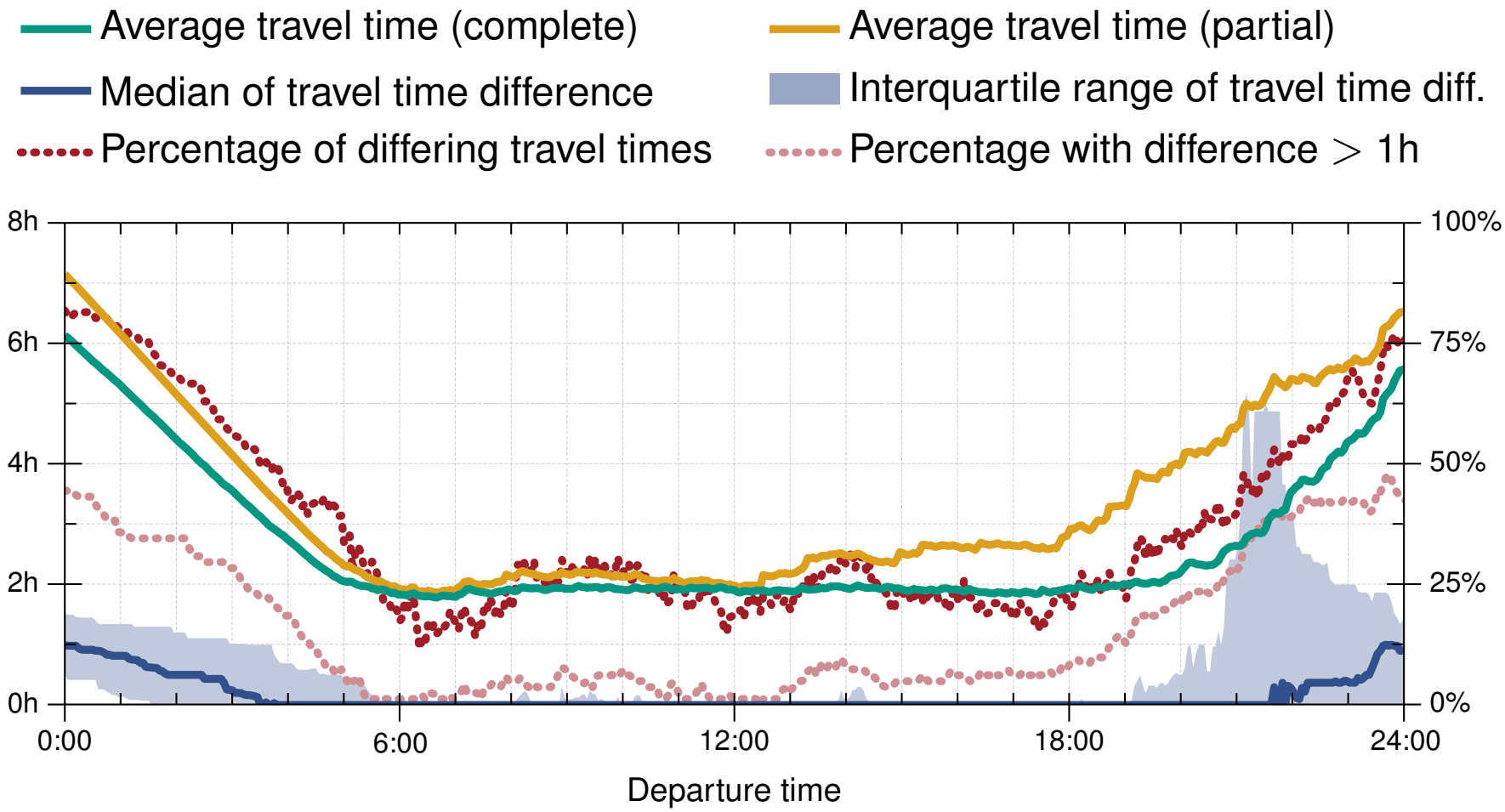
- Switzerland complete vs. Switzerland original (Distance rank 16)



Experimente

Vergleich der Reisezeiten:

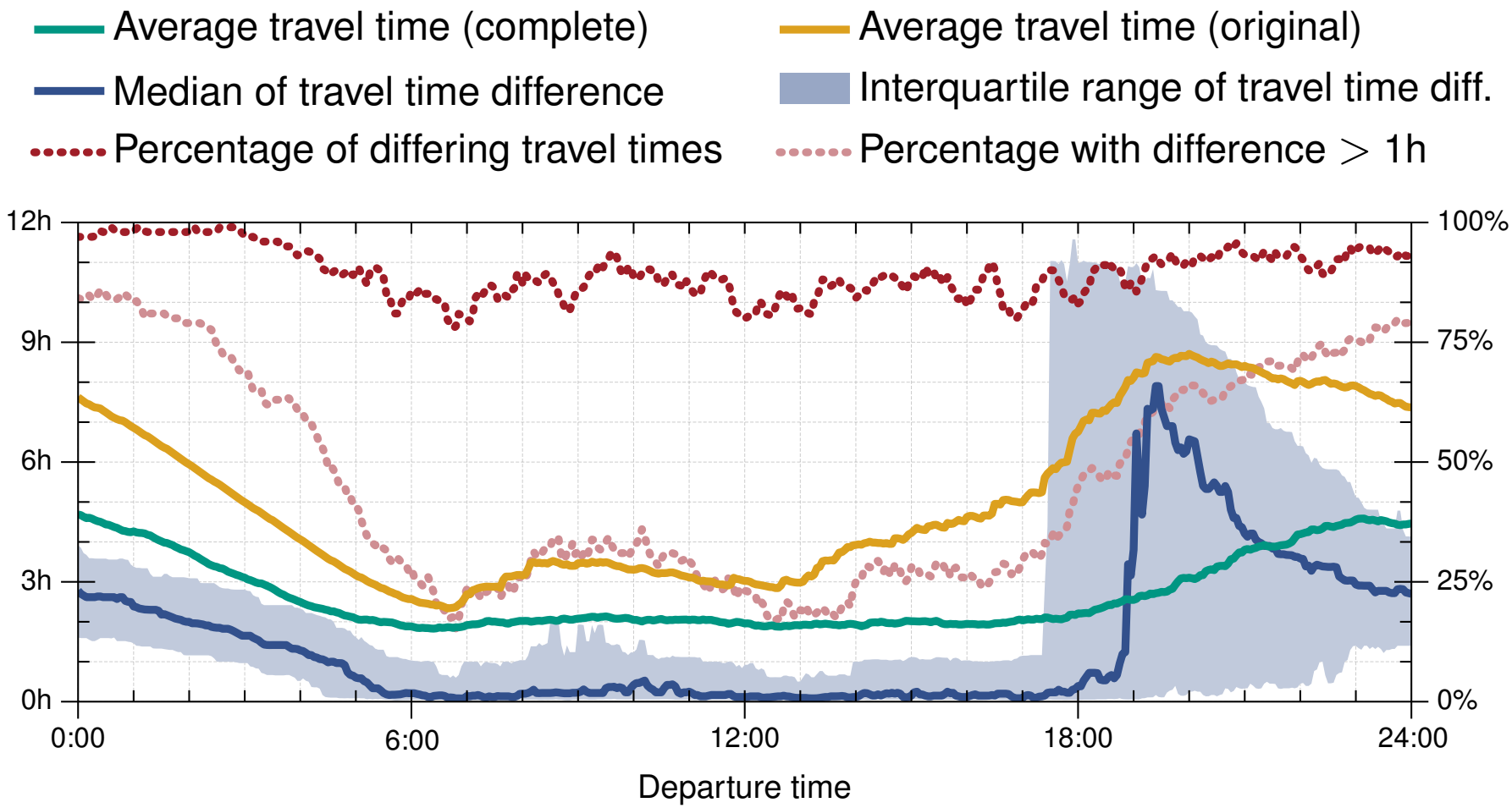
- Switzerland complete vs. Switzerland partial (Distance rank 16)



Experimente

Vergleich der Reisezeiten:

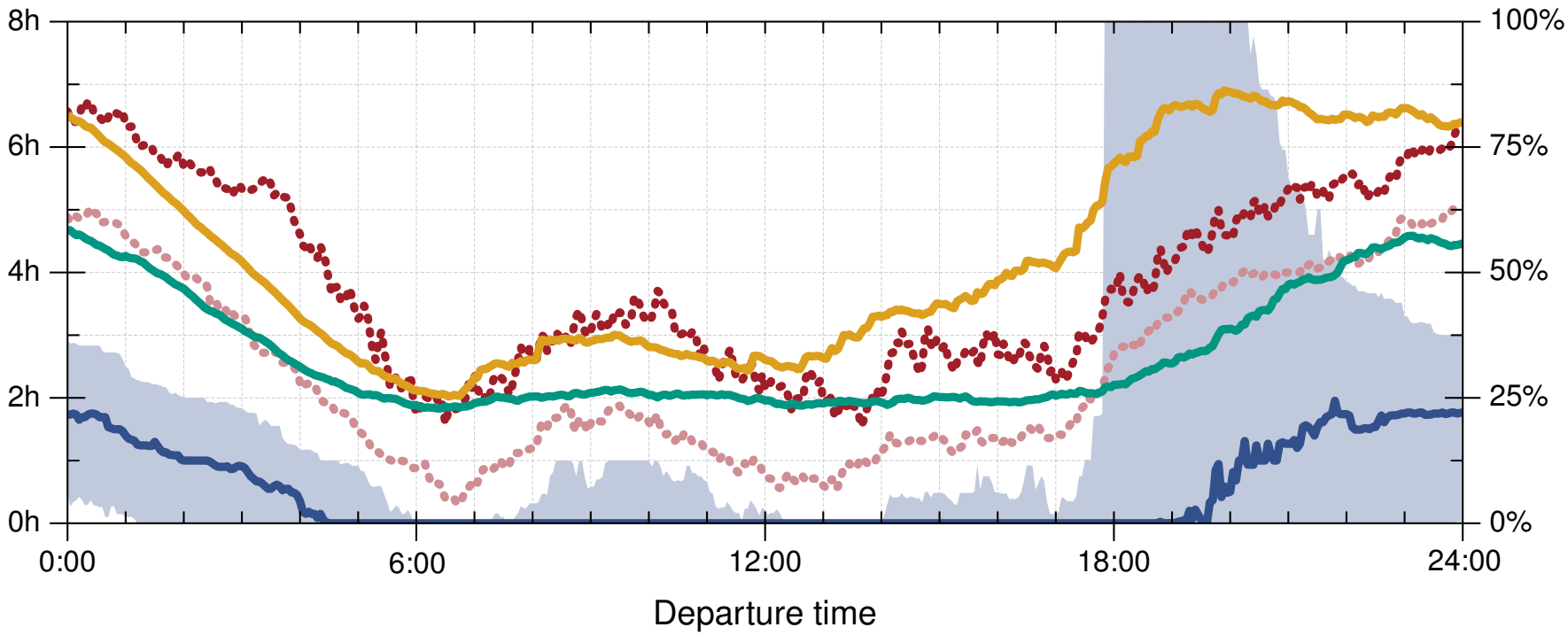
- Germany complete vs. Germany original (Distance rank 16)



Experimente

Vergleich der Reisezeiten:

- Germany complete vs. Germany partial (Distance rank 16)



Verkehrsumlegung

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Verkehrsumlegung

Gegeben:

- Öffentliches Verkehrsnetz
- Liste mit erwarteter Nachfrage
  (Tupel aus: Startknoten, Zielknoten, Abfahrtszeit)

Gesucht:

- Auslastung der Fahrzeuge

Verkehrsumlegung

Gegeben:

- Öffentliches Verkehrsnetz
- Liste mit erwarteter Nachfrage
  (Tupel aus: Startknoten, Zielknoten, Abfahrtszeit)

Gesucht:

- Auslastung der Fahrzeuge

Anwendung:

- Planung von neuen Linien
- Optimierung von Umleitungen

Verkehrsumlegung

Ansatz:

- Weise jedem Passagier (aus Nachfrage) eine Journey zu
- Algorithmus basiert auf CSA

Verkehrsumlegung

Ansatz:

- Weise jedem Passagier (aus Nachfrage) eine Journey zu
- Algorithmus basiert auf CSA

Aber:

- Verhalten der Passagiere nicht immer eindeutig
- Erlaube suboptimale Journeys
- Erlaube mehrere (Anteilig) Journeys pro Passagier

Gefühlte Ankunftszeit (PAT)

Idee:

- PAT für eine Connection c und Zielstop d
- Misst wie nützlich c ist um d zu erreichen
- Berücksichtigt vier Parameter:
  - Umstiegskosten λ_{trans}
  - Wartekosten λ_{wait}
  - Laufkosten λ_{walk}
  - Maximale erwartete Verspätung $\Delta_{\tau}^{\max}$

Annahme:

- Passagiere versuchen die PAT zu minimieren

Gefühlte Ankunftszeit (PAT)

Formale Definition:

$$\tau^{\mathrm{p}}_{\mathrm{arr}}(c, c', d) := \tau^{\mathrm{p}}_{\mathrm{trans}}(c, c') + \tau^{\mathrm{p}}_{\mathrm{wait}}(c, c') + \tau^{\mathrm{p}}_{\mathrm{arr}}(c', d)$$

$$\tau^{\mathrm{p}}_{\mathrm{arr}}(c, d \mid \mathrm{walk}) := \begin{cases} \tau_{\mathrm{arr}}(c) & \text{if } v_{\mathrm{arr}}(c) = d \\ \tau_{\mathrm{arr}}(c) + \lambda_{\mathrm{walk}} \cdot \tau_{\mathrm{trans}}(v_{\mathrm{arr}}(c), d) & \text{otherwise} \end{cases}$$

$$\mathcal{T}(c) := \{c' \in \mathcal{C} \mid \mathrm{trip}(c') = \mathrm{trip}(c) \wedge \tau_{\mathrm{dep}}(c') \geq \tau_{\mathrm{arr}}(c)\}$$

$$\tau^{\mathrm{p}}_{\mathrm{arr}}(c, d \mid \mathrm{trip}) := \begin{cases} \min\{\tau^{\mathrm{p}}_{\mathrm{arr}}(c', d) \mid c' \in \mathcal{T}(c)\} & \text{if } \mathcal{T}(c) \neq \emptyset \\ \infty & \text{otherwise} \end{cases}$$

$$\tau^{\mathrm{p}}_{\mathrm{arr}}(c, c', d) := \tau^{\mathrm{p}}_{\mathrm{trans}}(c, c') + \tau^{\mathrm{p}}_{\mathrm{wait}}(c, c') + \tau^{\mathrm{p}}_{\mathrm{arr}}(c', d)$$

$$\mathcal{R}(c) := \{c' \in \mathcal{C} \mid \tau_{\mathrm{wait}}(c, c') \geq 0\}$$

$$\mathcal{R}_{\mathrm{opt}}(c) := \{c' \in \mathcal{R}(c) \mid \forall \bar{c} \in \mathcal{R}(c) : \tau_{\mathrm{wait}}(c, \bar{c}) \geq \tau_{\mathrm{wait}}(c, c') \Rightarrow \tau^{\mathrm{p}}_{\mathrm{arr}}(c, \bar{c}, d) \geq \tau^{\mathrm{p}}_{\mathrm{arr}}(c, c', d)\}$$

$$\langle c_1, \ldots, c_k \rangle \text{ with } \forall i \in [1, k] : c_i \in \mathcal{R}_{\mathrm{opt}}(c) \wedge \forall i \in [2, k] : \tau_{\mathrm{wait}}(c, c_i) \geq \tau_{\mathrm{wait}}(c, c_{i-1})$$

$$\tau^{c}_{\mathrm{wait}}(i) := \begin{cases} \tau_{\mathrm{wait}}(c, c_i) & \text{if } i \in [1, k] \\ -\infty & \text{otherwise} \end{cases}$$

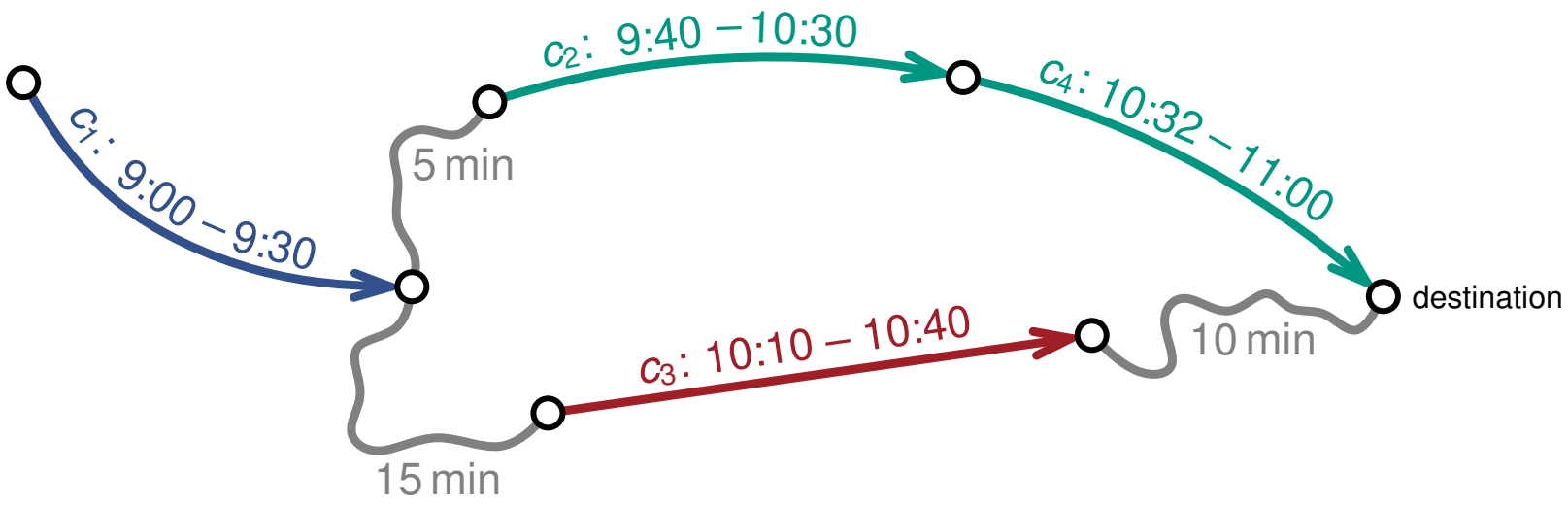
$$\tau^{\mathrm{p}}_{\mathrm{arr}}(c, d \mid \mathrm{trans}) := \begin{cases} \sum_{i=1}^{k} \left(\frac{P[\tau^{c}_{\mathrm{wait}}(i-1) < \Delta^{c}_{\tau} \leq \tau^{c}_{\mathrm{wait}}(i)]}{P[\Delta^{c}_{\tau} \leq \tau^{c}_{\mathrm{wait}}(k)]} \cdot \tau^{\mathrm{p}}_{\mathrm{arr}}(c, c_i, d) \right) & \text{if } k > 0 \\ \infty & \text{otherwise} \end{cases}$$

Beispiel PAT Berechnung

Beispiel:

- $\lambda_{walk} = 3, \quad \lambda_{wait} = 2, \quad \lambda_{trans} = 5\,\text{min}$

Connection	PAT
c_4	
c_3	
c_2	
c_1	



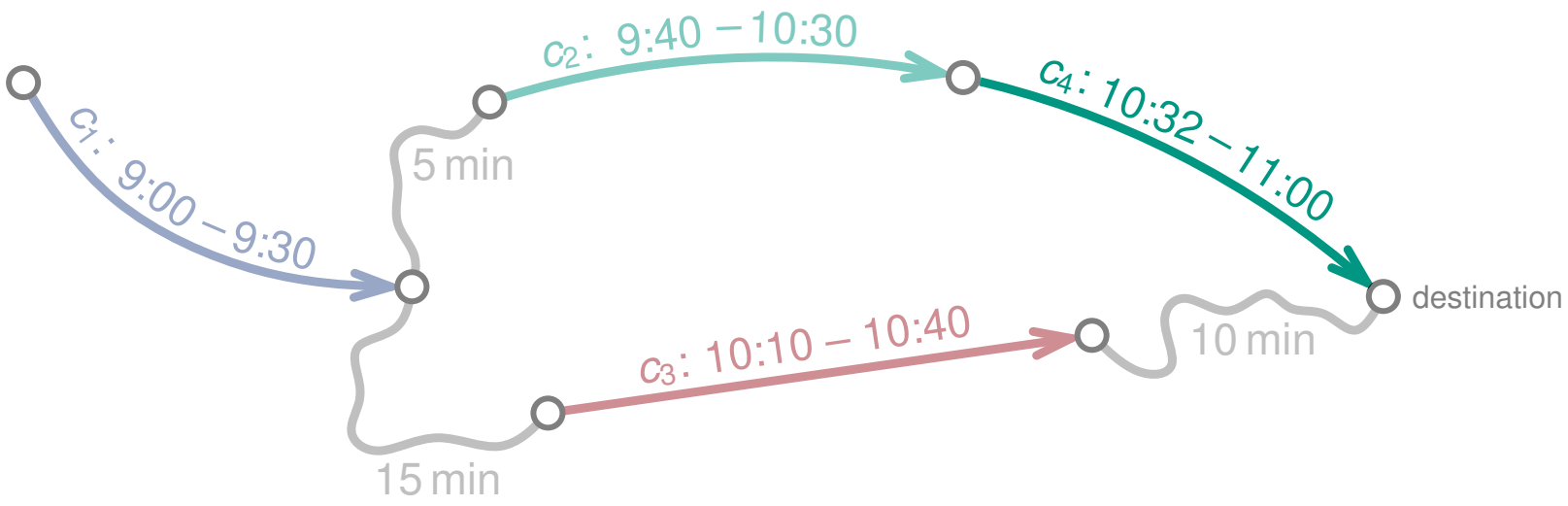
Beispiel PAT Berechnung

Beispiel:

- $\lambda_{\text{walk}} = 3, \quad \lambda_{\text{wait}} = 2, \quad \lambda_{\text{trans}} = 5\,\text{min}$
- **Fall 1:** Connection c erreicht Ziel

  $\Rightarrow$ PAT = arrival time $\tau_{\text{arr}}(c)$

Connection	PAT
c_4	11:00
c_3	
c_2	
c_1	



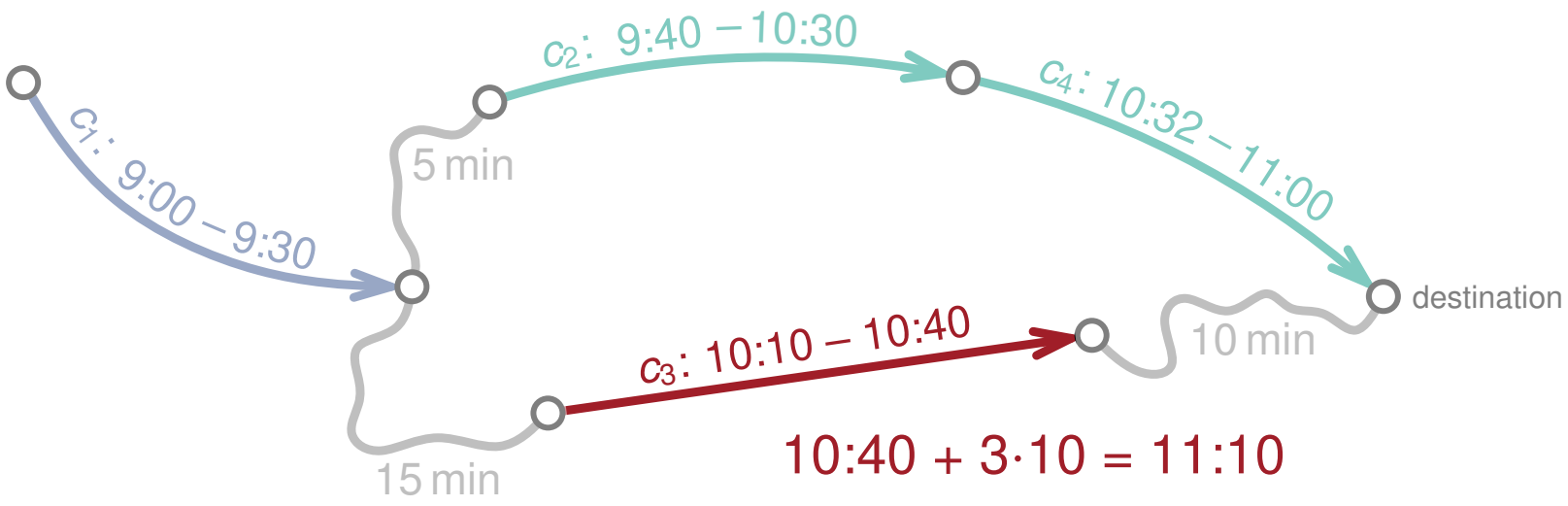
Beispiel PAT Berechnung

Beispiel:

- $\lambda_{walk} = 3, \quad \lambda_{wait} = 2, \quad \lambda_{trans} = 5\,\text{min}$
- **Fall 2:** Lauf von Connection c zum Ziel

  $\Rightarrow \text{PAT} = \tau_{arr}(c) + (\lambda_{walk} \cdot \tau_{walking})$

Connection	PAT
c_4	11:00
c_3	11:10
c_2	
c_1	



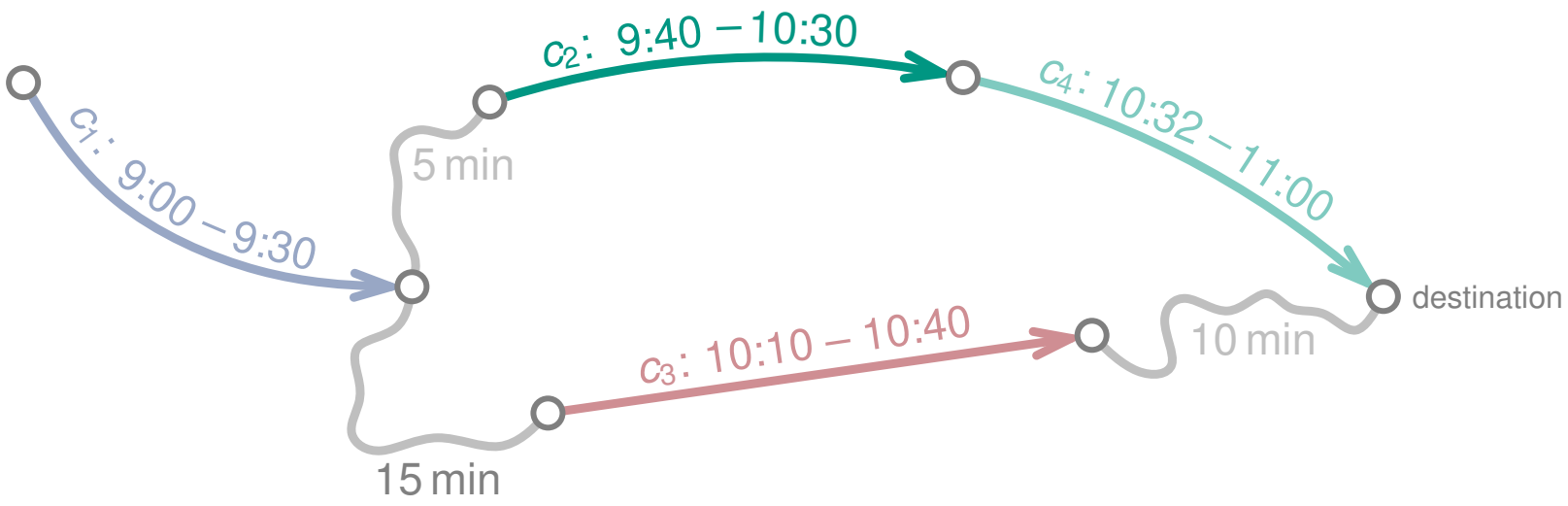
Beispiel PAT Berechnung

Beispiel:

- $\lambda_{\text{walk}} = 3, \quad \lambda_{\text{wait}} = 2, \quad \lambda_{\text{trans}} = 5\,\text{min}$
- **Fall 3:** Weiterfahren mit Con. c' (gleicher Trip)

  $\Rightarrow$ PAT = PAT c'

Connection	PAT
c_4	11:00
c_3	11:10
c_2	11:00
c_1	

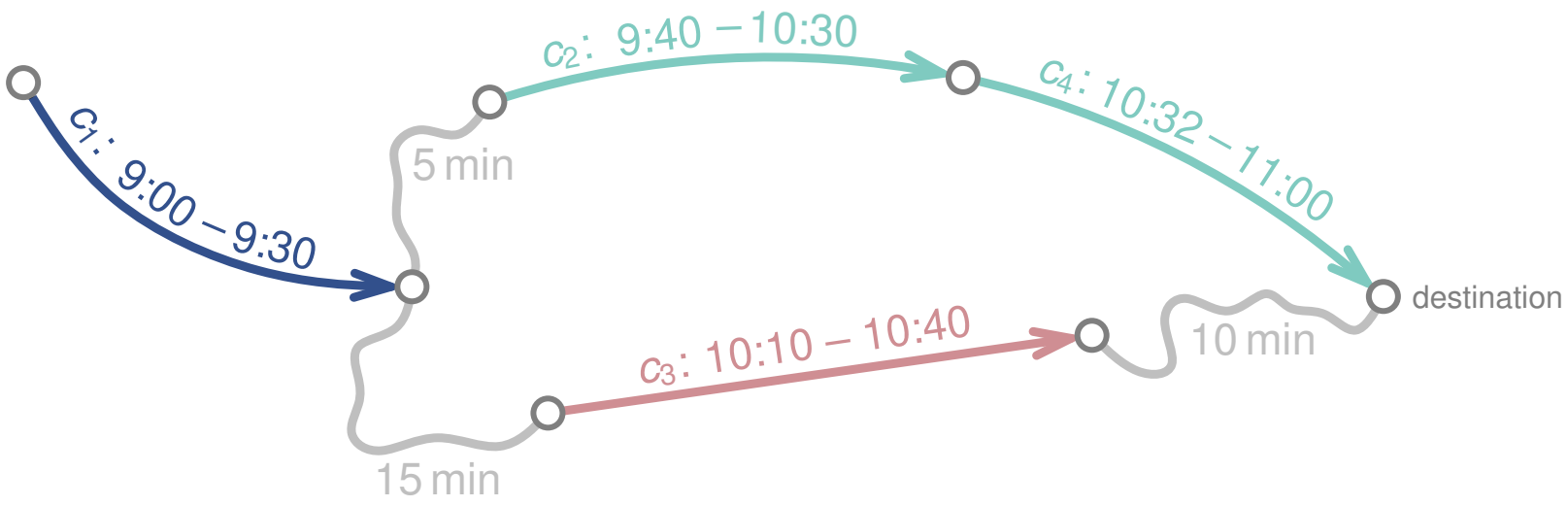


Beispiel PAT Berechnung

Beispiel:

- $\lambda_{\text{walk}} = 3, \quad \lambda_{\text{wait}} = 2, \quad \lambda_{\text{trans}} = 5\,\text{min}$
- **Fall 4:** Weiterfahren mit Con. c' (anderer Trip)

Connection	PAT
c_4	11:00
c_3	11:10
c_2	11:00
c_1	

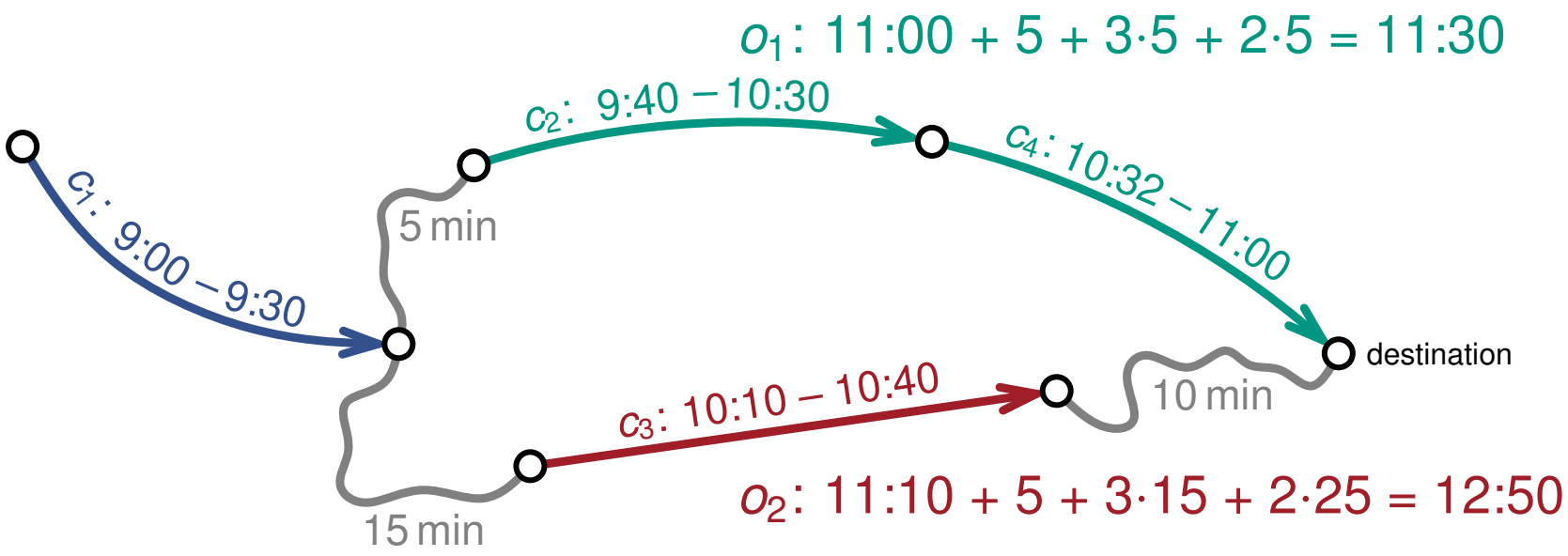


Beispiel PAT Berechnung

Beispiel:

- $\lambda_{walk} = 3, \quad \lambda_{wait} = 2, \quad \lambda_{trans} = 5\,\text{min}$
- **Fall 4:** Weiterfahren mit Con. c' (anderer Trip)

Connection	PAT
c_4	11:00
c_3	11:10
c_2	11:00
c_1	



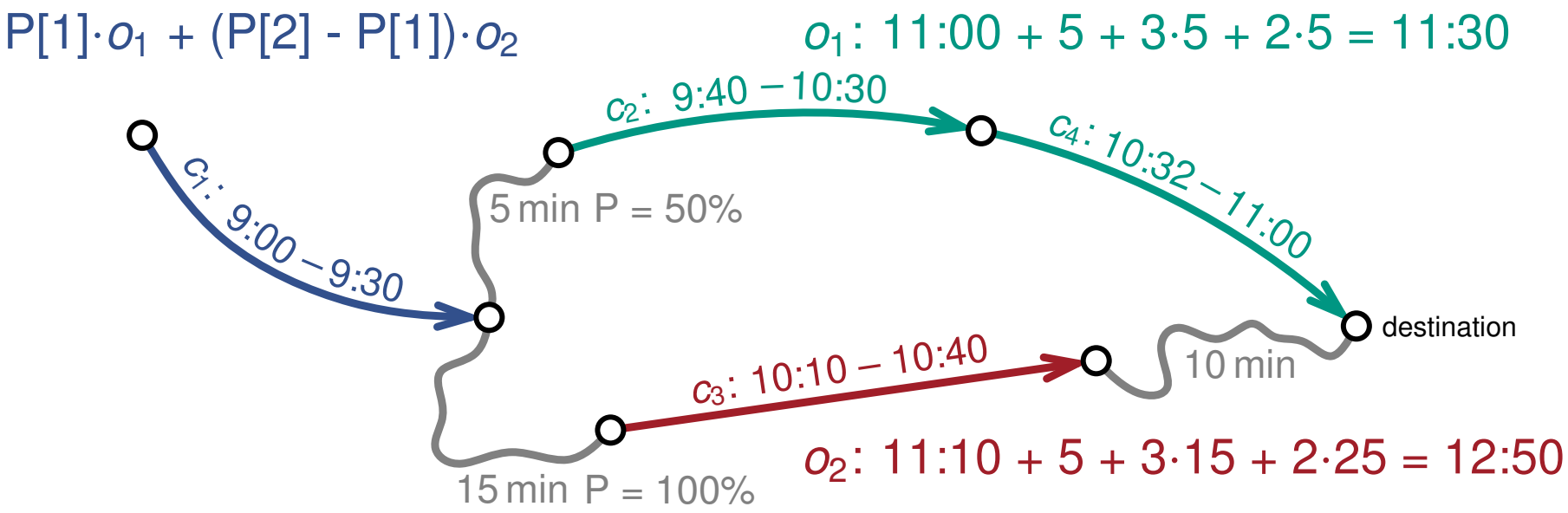
Beispiel PAT Berechnung

Beispiel:

- $\lambda_{walk} = 3, \quad \lambda_{wait} = 2, \quad \lambda_{trans} = 5\,\text{min}$
- **Fall 4:** Weiterfahren mit einer Option o_i

  $\Rightarrow \text{PAT} = \sum_i \left(\text{transfer probability}(o_i) \cdot o_i\right)$

Connection	PAT
c_4	11:00
c_3	11:10
c_2	11:00
c_1	



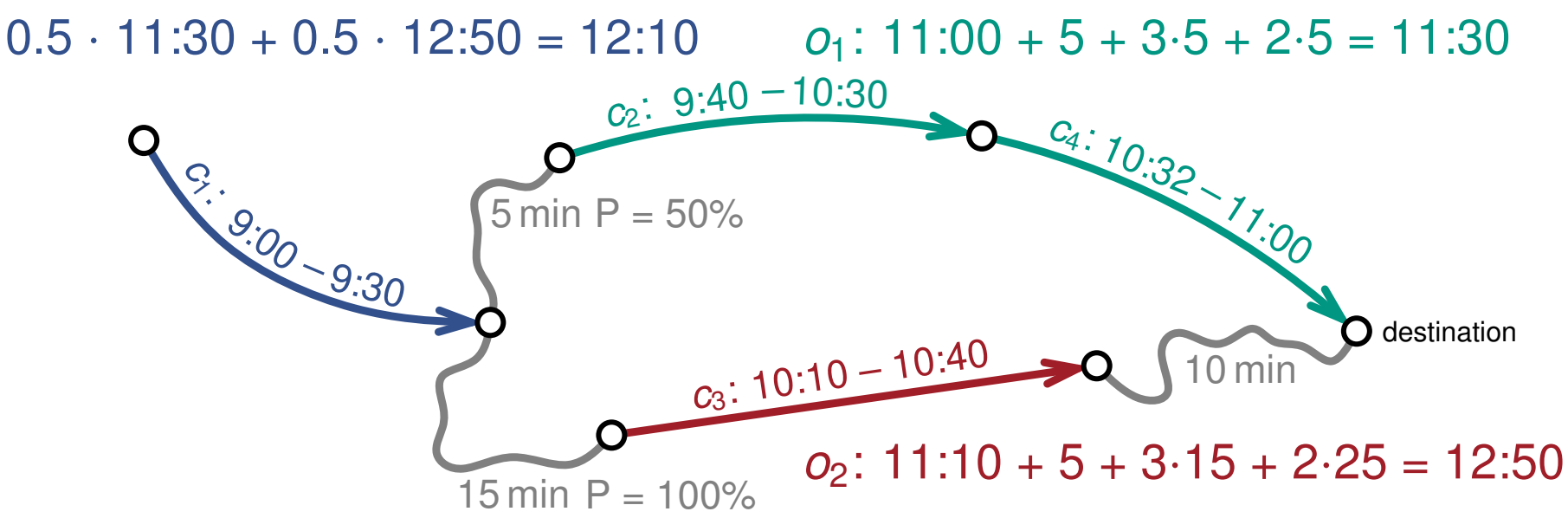
Beispiel PAT Berechnung

Beispiel:

- $\lambda_{\text{walk}} = 3, \quad \lambda_{\text{wait}} = 2, \quad \lambda_{\text{trans}} = 5\,\text{min}$
- **Fall 4:** Weiterfahren mit einer Option o_i

  $\Rightarrow \text{PAT} = \sum_i \left(\text{transfer probability}(o_i) \cdot o_i\right)$

Connection	PAT
c_4	11:00
c_3	11:10
c_2	11:00
c_1	12:10



Umlegungsalgorithmus

Ansatz:

- Simuliere Bewegung der Passagiere im Netzwerk
- Entscheide pro Connection c, wer c benutzt
- Passagiere mit selbem Ziel werden sich treffen
  $\Rightarrow$ Müssen die selben Entscheidungen treffen
  $\Rightarrow$ Algorithmus profitiert von Synergieeffekten

Umlegungsalgorithmus

Ansatz:

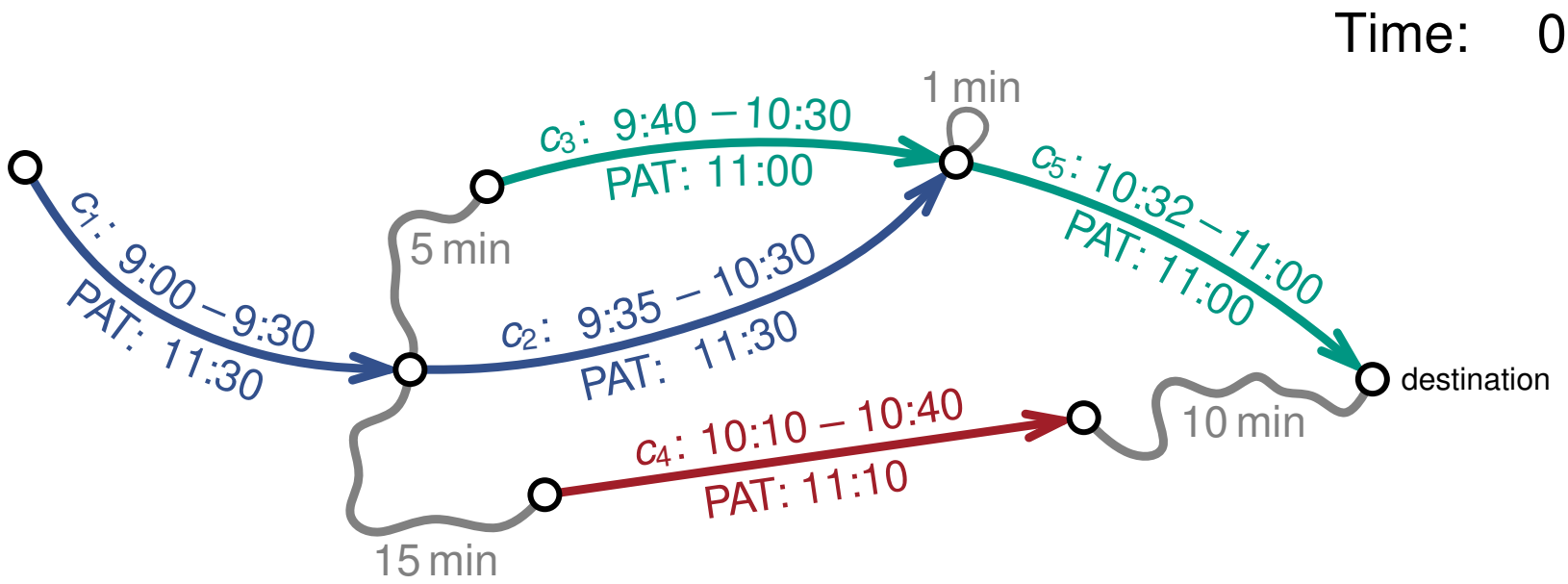
- Simuliere Bewegung der Passagiere im Netzwerk
- Entscheide pro Connection c, wer c benutzt
- Passagiere mit selbem Ziel werden sich treffen
  $\Rightarrow$ Müssen die selben Entscheidungen treffen
  $\Rightarrow$ Algorithmus profitiert von Synergieeffekten

Überblick:

- Sortiere Passagiere nach Zielstop
- Berechne Umlegung pro Zielstop in 3 Schritten:
  - Berechne PATs für jede Connection
  - Simuliere Bewegung der Passagiere basierend auf PATs
  - Entferne überflüssige Kreise aus Journeys (optional)

Beispiel Umlegungsberechnung

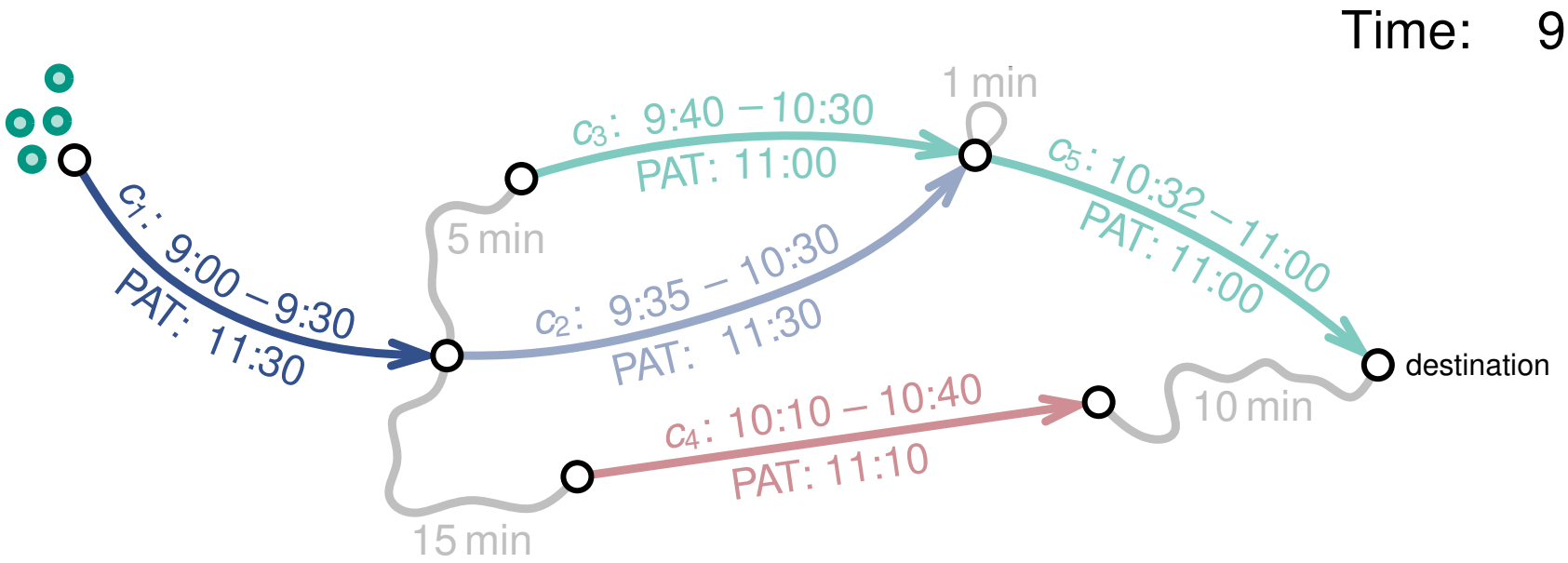
- Bearbeite Connections chronologisch (nach Abfahrtszeit)
- Entscheide welche Passagiere die Connection benutzen



Beispiel Umlegungsberechnung

- Bearbeite Connections chronologisch (nach Abfahrtszeit)
- Entscheide welche Passagiere die Connection benutzen

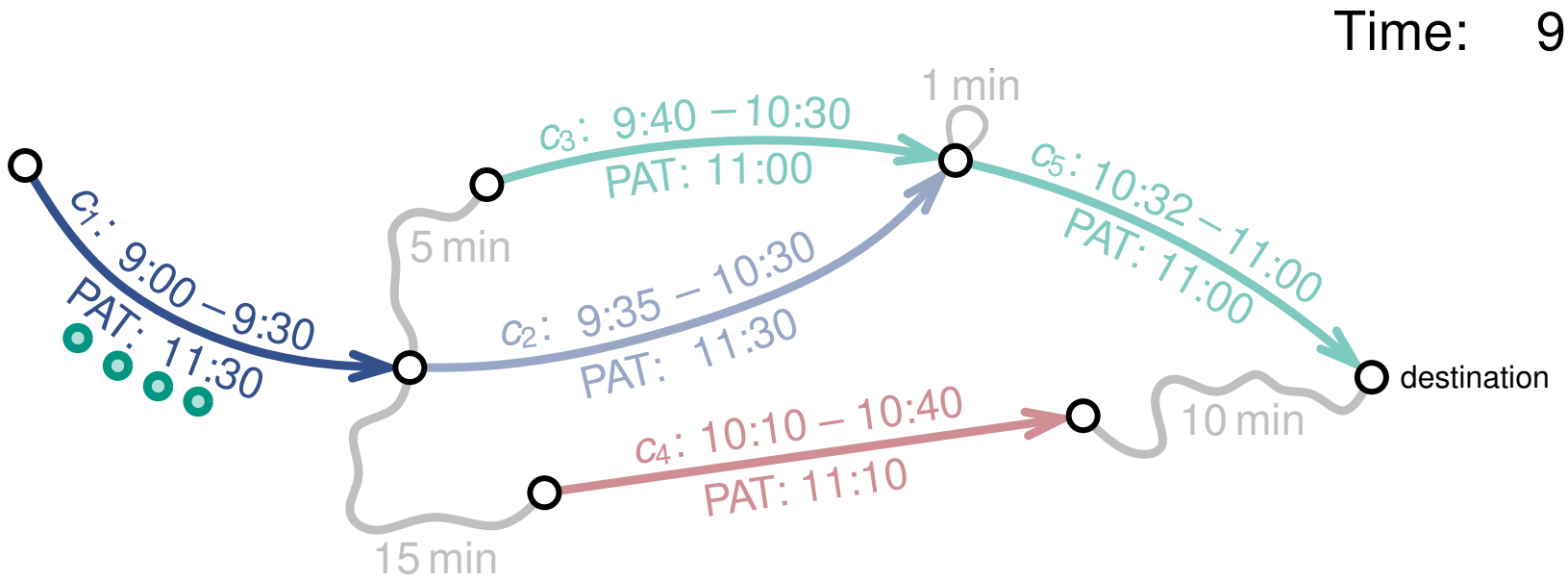
1. Erzeuge Passagiere entsprechend der Nachfrage



Beispiel Umlegungsberechnung

- Bearbeite Connections chronologisch (nach Abfahrtszeit)
- Entscheide welche Passagiere die Connection benutzen

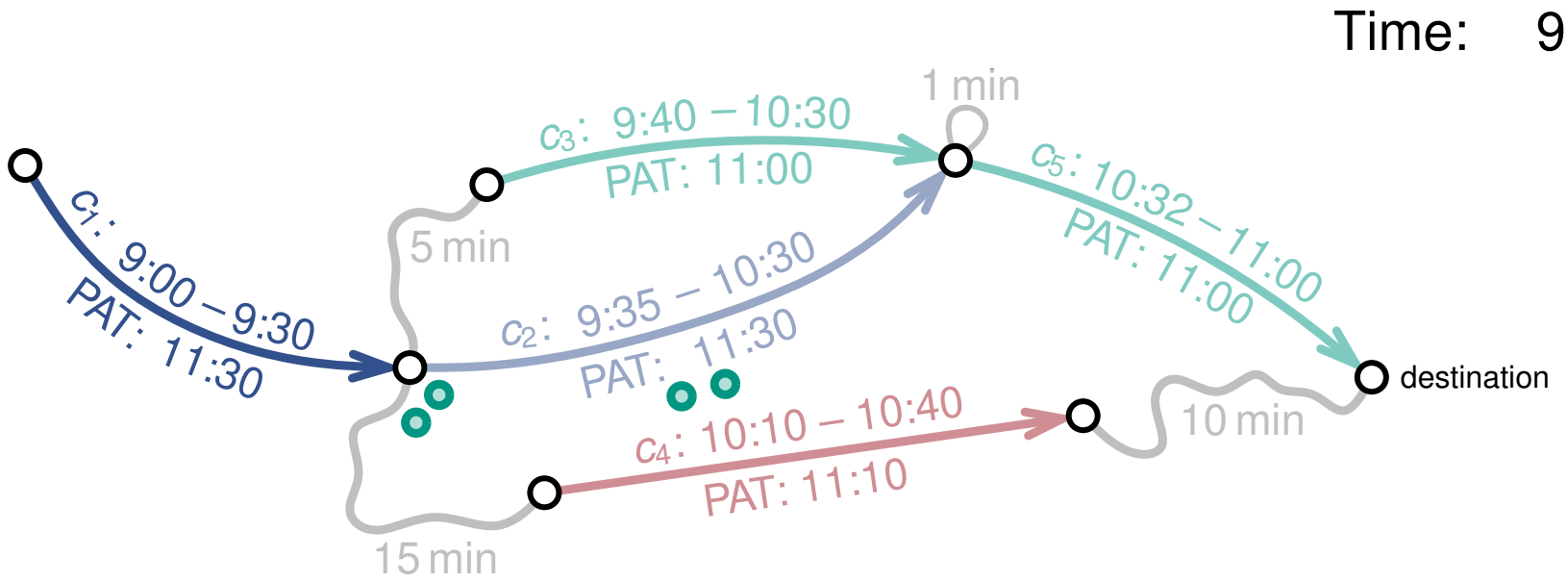
2. Entscheide welche Passagiere einsteigen



Beispiel Umlegungsberechnung

- Bearbeite Connections chronologisch (nach Abfahrtszeit)
- Entscheide welche Passagiere die Connection benutzen

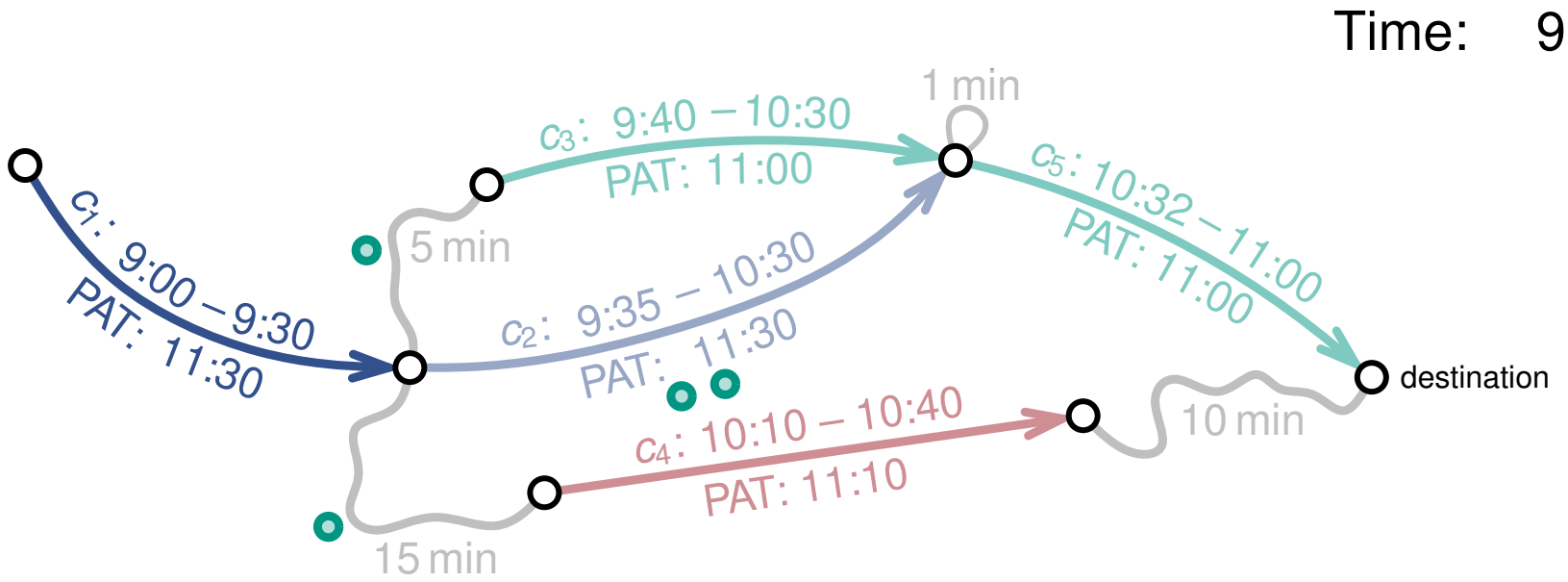
3. Entscheide welche Passagiere aussteigen



Beispiel Umlegungsberechnung

- Bearbeite Connections chronologisch (nach Abfahrtszeit)
- Entscheide welche Passagiere die Connection benutzen

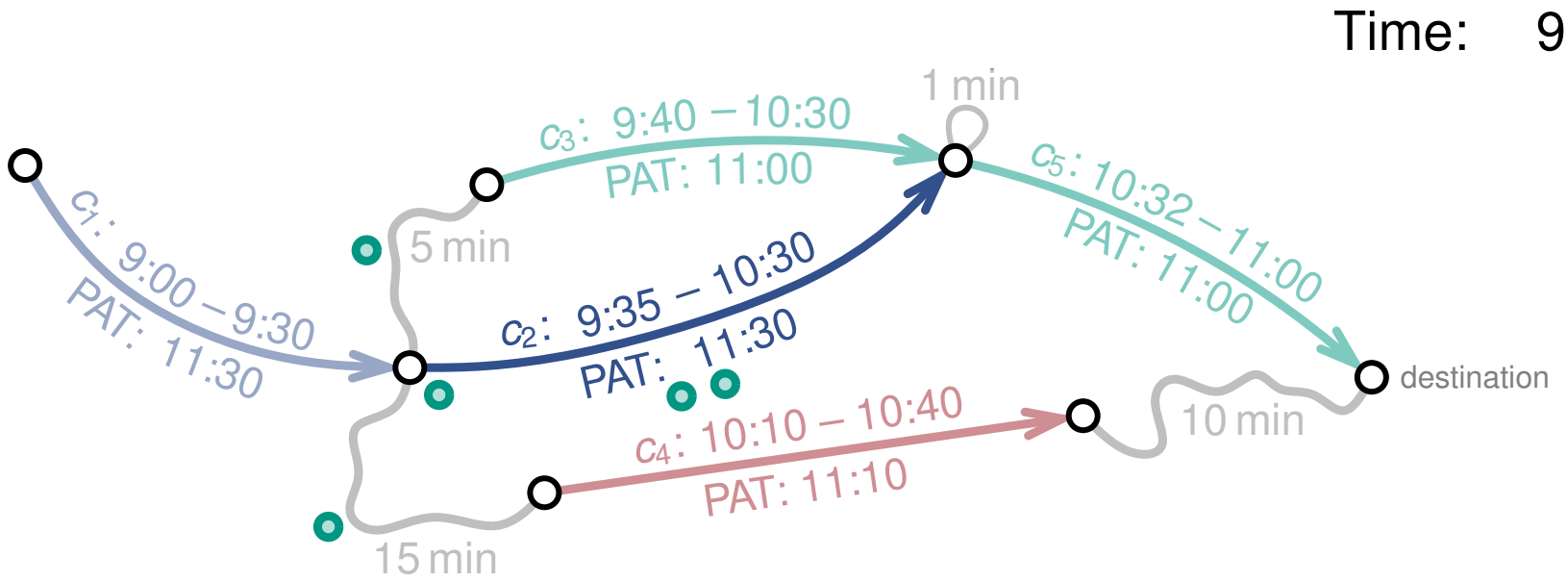
4. Verschiebe ausgestiegene Passagiere zum nächsten Stop



Beispiel Umlegungsberechnung

- Bearbeite Connections chronologisch (nach Abfahrtszeit)
- Entscheide welche Passagiere die Connection benutzen

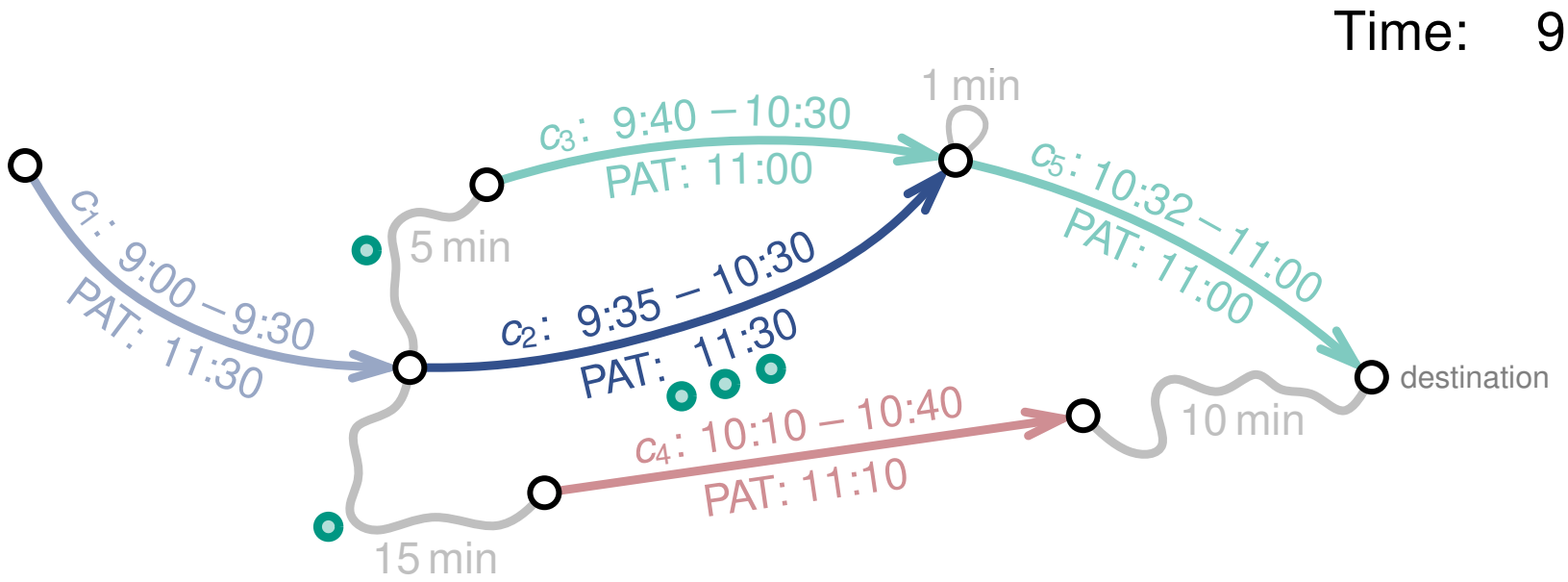
1. Erzeuge Passagiere entsprechend der Nachfrage



Beispiel Umlegungsberechnung

- Bearbeite Connections chronologisch (nach Abfahrtszeit)
- Entscheide welche Passagiere die Connection benutzen

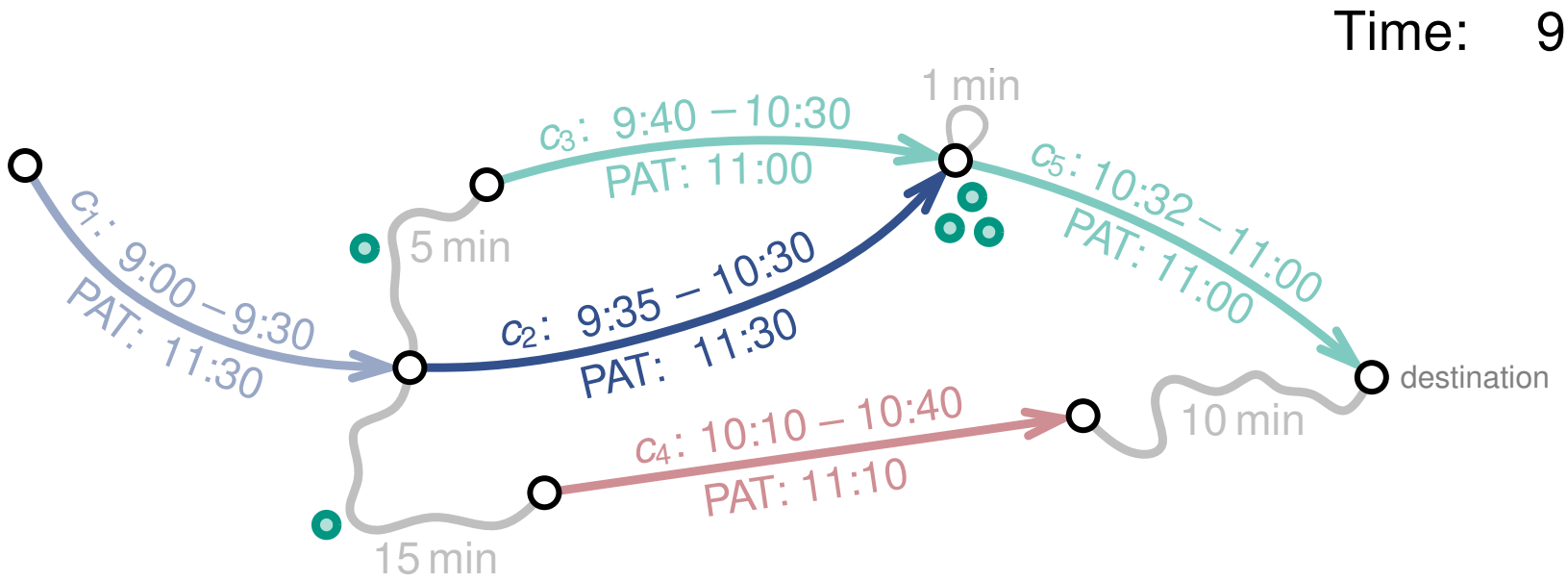
2. Entscheide welche Passagiere einsteigen



Beispiel Umlegungsberechnung

- Bearbeite Connections chronologisch (nach Abfahrtszeit)
- Entscheide welche Passagiere die Connection benutzen

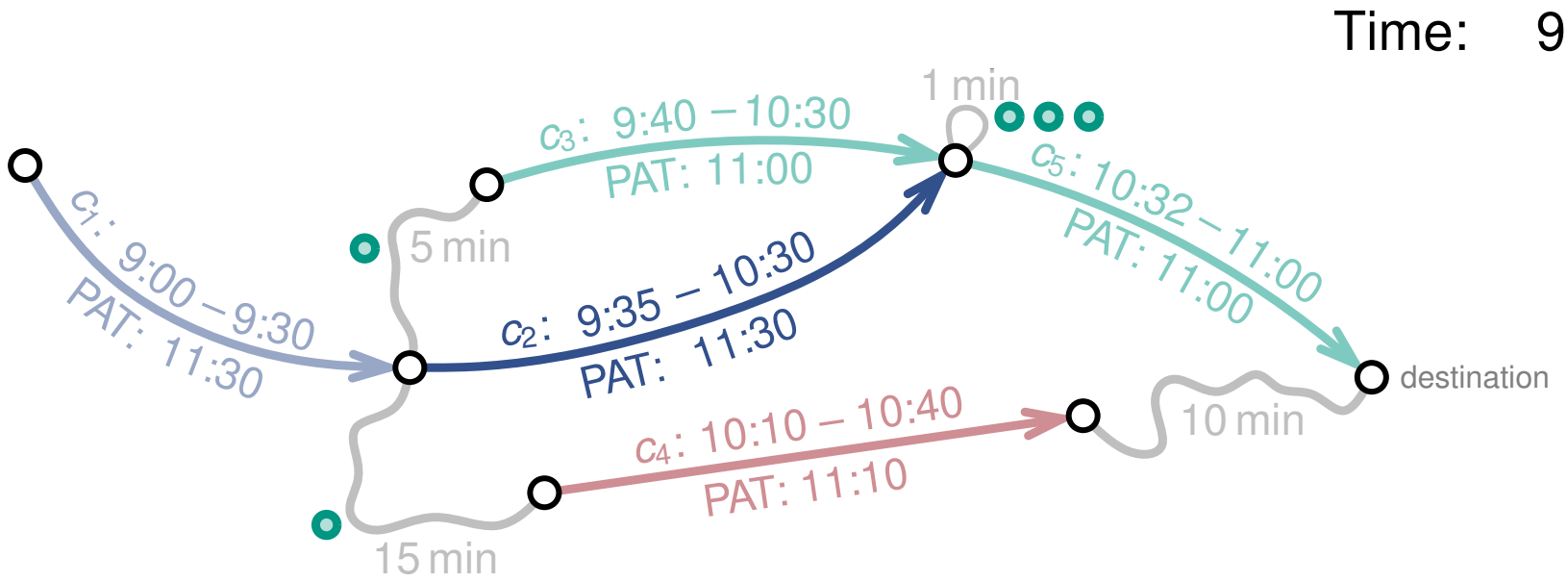
3. Entscheide welche Passagiere aussteigen



Beispiel Umlegungsberechnung

- Bearbeite Connections chronologisch (nach Abfahrtszeit)
- Entscheide welche Passagiere die Connection benutzen

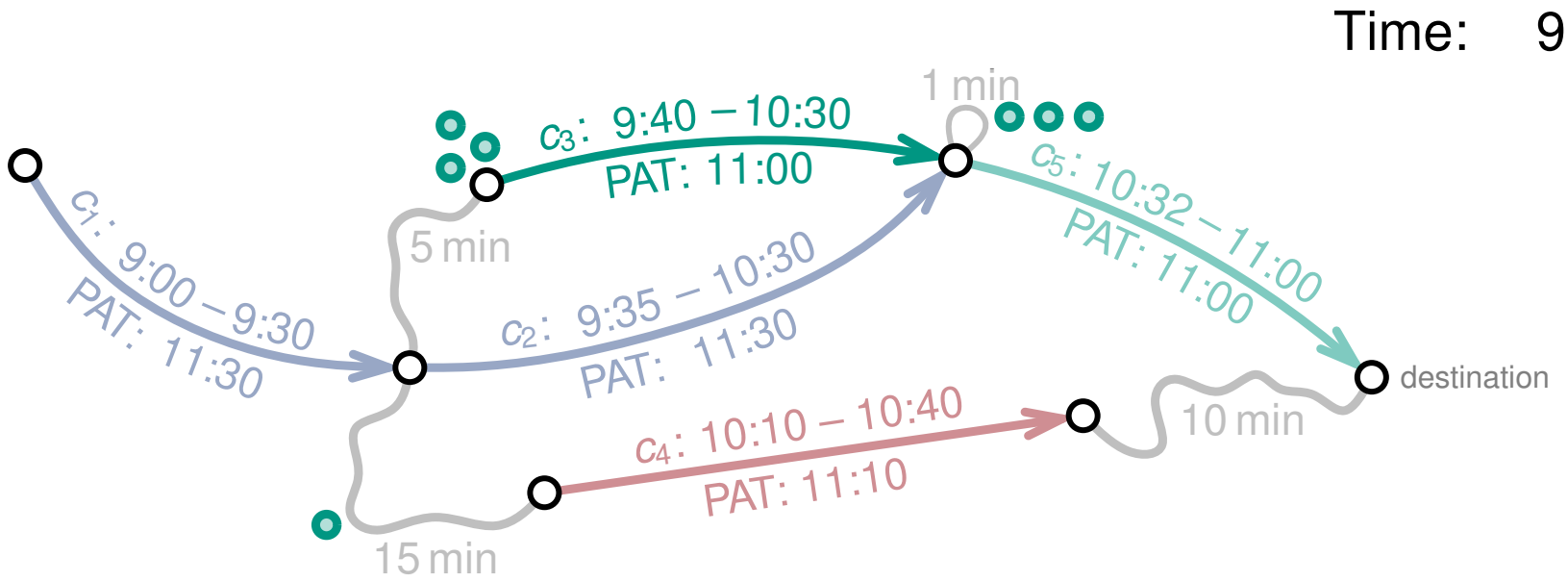
4. Verschiebe ausgestiegene Passagiere zum nächsten Stop



Beispiel Umlegungsberechnung

- Bearbeite Connections chronologisch (nach Abfahrtszeit)
- Entscheide welche Passagiere die Connection benutzen

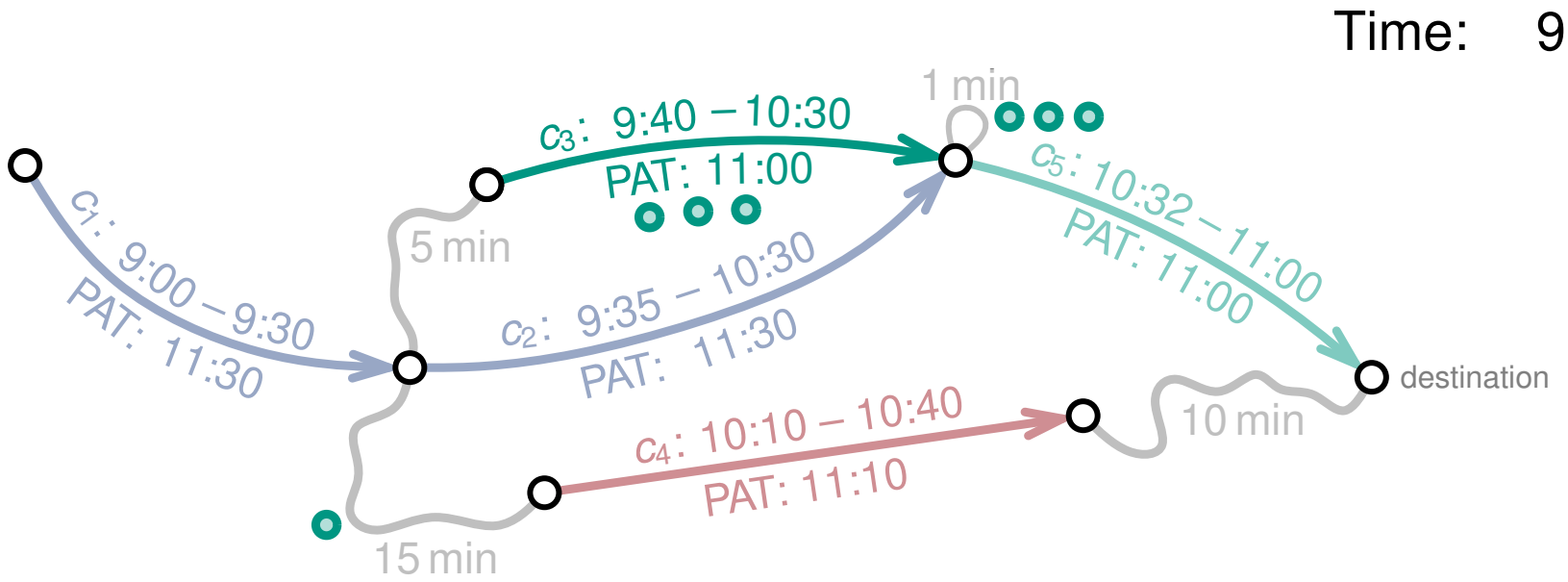
1. Erzeuge Passagiere entsprechend der Nachfrage



Beispiel Umlegungsberechnung

- Bearbeite Connections chronologisch (nach Abfahrtszeit)
- Entscheide welche Passagiere die Connection benutzen

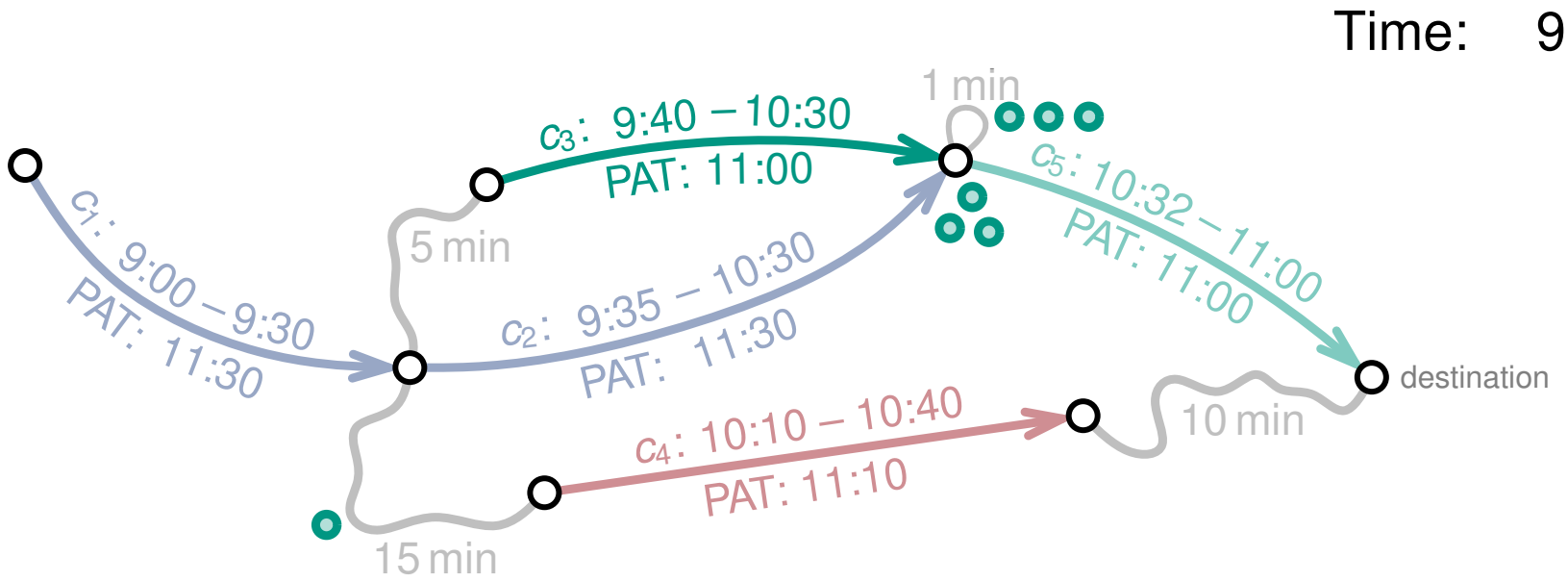
2. Entscheide welche Passagiere einsteigen



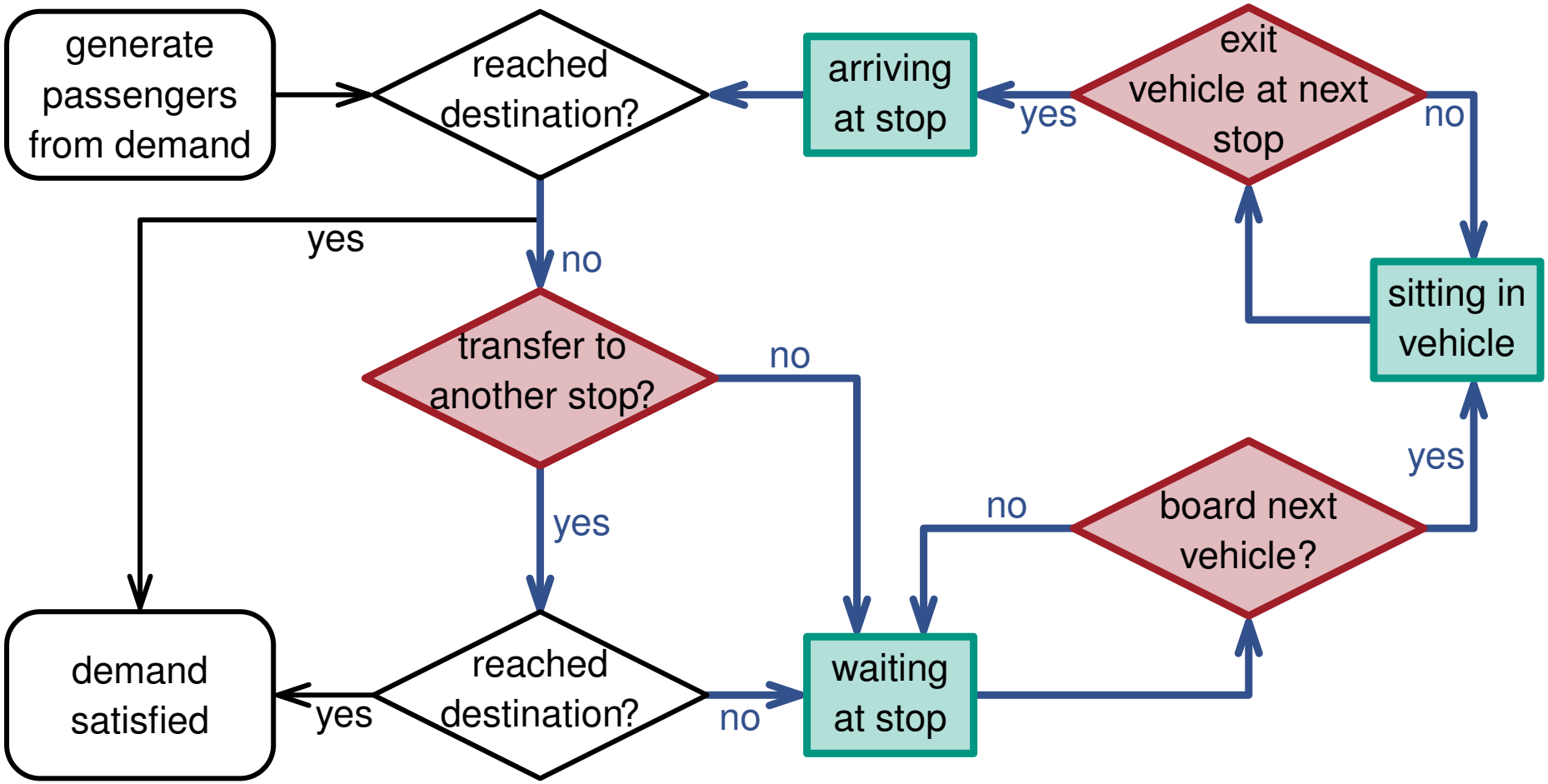
Beispiel Umlegungsberechnung

- Bearbeite Connections chronologisch (nach Abfahrtszeit)
- Entscheide welche Passagiere die Connection benutzen

3. ...



Umlegungsberechnung Übersicht



Entscheidungsmodell

Ziel:

- Entscheidet welche Connection ein Passagier nimmt
- Hängt von der Verspätungstoleranz $\lambda_{\Delta max}$ des Passagieres ab

Entscheidungsmodell

Ziel:

- Entscheidet welche Connection ein Passagier nimmt
- Hängt von der Verspätungstoleranz $\lambda_{\Delta\max}$ des Passagieres ab

Definition:

- Gegeben sind die Optionen $o_1, \ldots, o_k$ und ihre PATs
- Bestimme den Nutzen $g(i)$ jeder Option:

$$g(i) := \max\left(0, \min_{j \neq i}(PAT(o_j)) - PAT(o_i) + \lambda_{\Delta\max}\right)$$

- Die Wahrscheinlichkeit $P[i]$, dass ein Passagier Option i wählt, ist:

$$P[i] := \frac{g(i)}{\sum_{j=1}^{k} g(j)}$$

Kreise

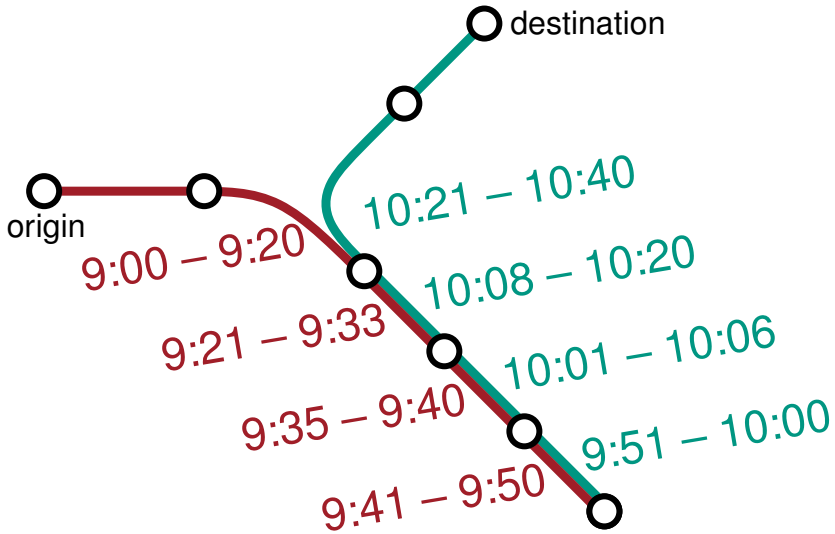
Kreis Definition:

- Den selben Stop mehrfach besuchen
- Umlegungen mit Kreisen können unerwünscht sein
- Eine Journey mit Kreisen kann optimal bezüglich PAT sein
- Hohe Wartekosten können zu Kreisen führen

Kreise

Kreis Definition:

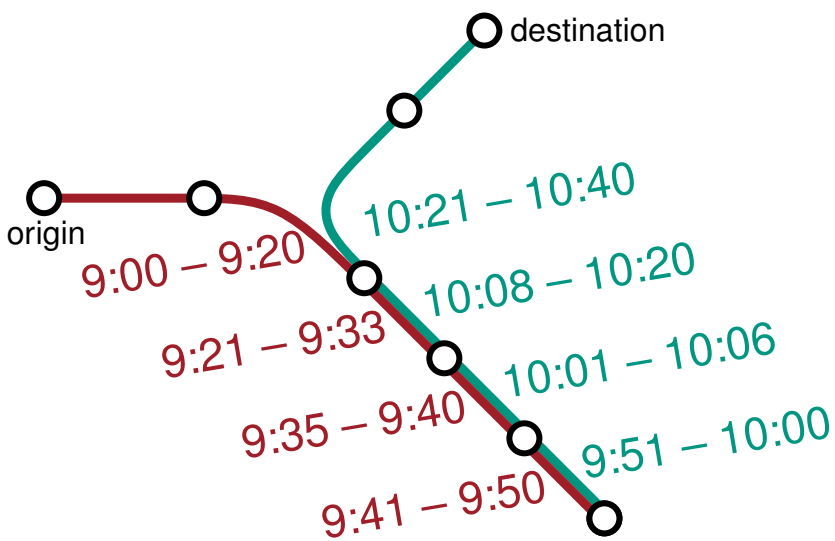
- Den selben Stop mehrfach besuchen
- Umlegungen mit Kreisen können unerwünscht sein
- Eine Journey mit Kreisen kann optimal bezüglich PAT sein
- Hohe Wartekosten können zu Kreisen führen

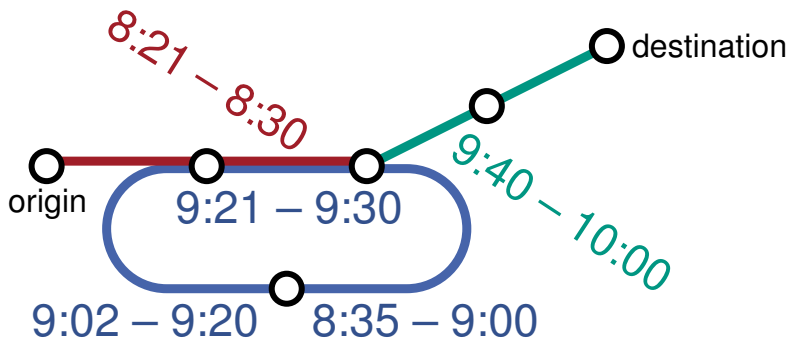


Kreise

Kreis Definition:

- Den selben Stop mehrfach besuchen
- Umlegungen mit Kreisen können unerwünscht sein
- Eine Journey mit Kreisen kann optimal bezüglich PAT sein
- Hohe Wartekosten können zu Kreisen führen



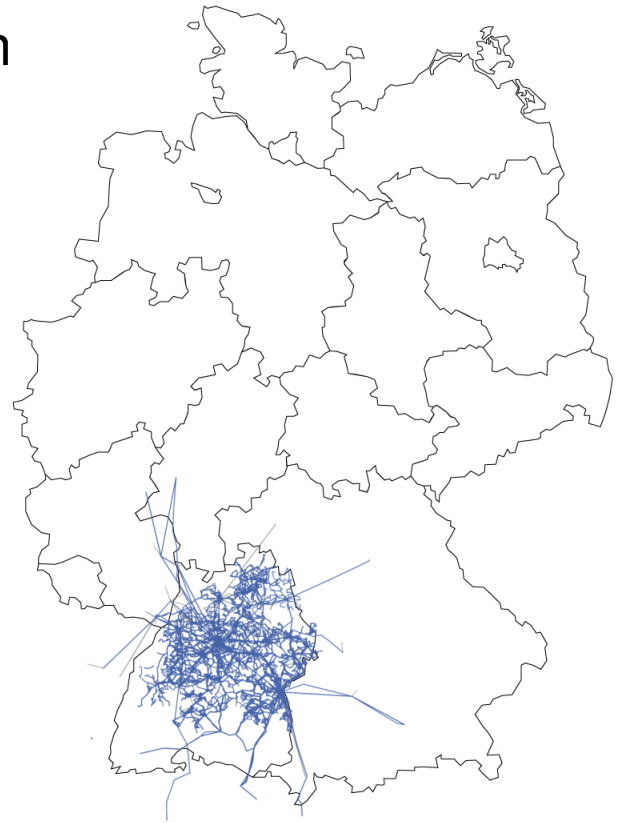


Auswertung

Instanzen:

- Großraum Stuttgart
- Enthält auch Frankfurt, Basel und München
- Beschreibt den Verkehr eines Tages

Anzahl Knoten	15 115
Anzahl Stops	13 941
Anzahl Kanten	33 890
Anzahl Kanten ohne Schlaufen	18 775
Anzahl Connections	780 042
Anzahl Trips	47 844
Anzahl Passagiere	1 249 910



Auswertung – Laufzeiten

Benutzte Parameter:

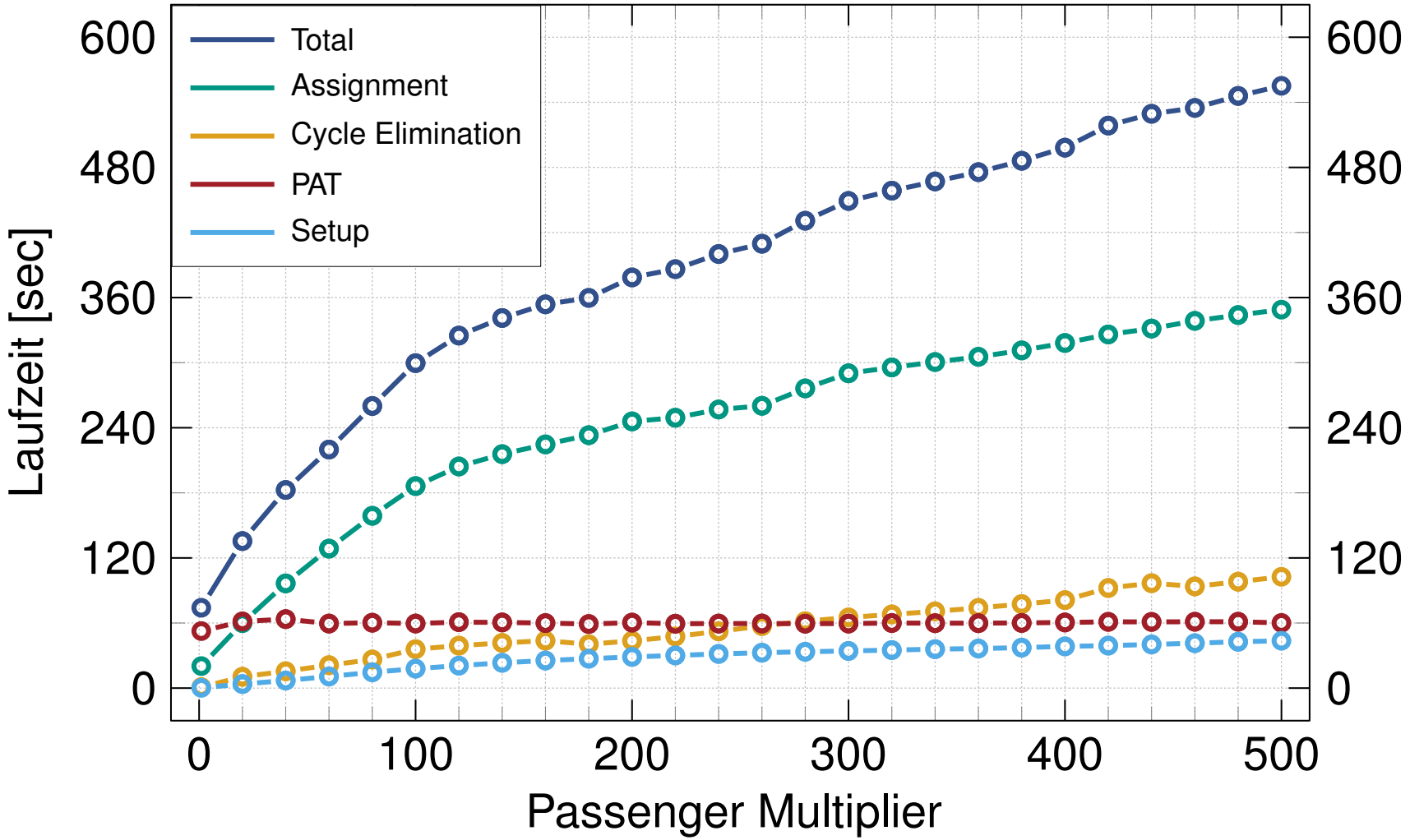
- Laufkosten $\lambda_{\text{walk}} = 2$
- Wartekosten $\lambda_{\text{wait}} = 0.5$
- Umstiegskosten $\lambda_{\text{trans}} = 5\,\text{min}$
- Verspätungstoleranz $\lambda_{\Delta\text{max}} = 5\,\text{min}$
- Maximale erwartete Verspätung $\Delta_{\tau}^{\text{max}} = 1\,\text{min}$

Laufzeitvergleich:

- Laufzeit VISUM $\approx 30\,\text{min}$ (mit 8 Threads)
- PAT basierte Umlegung: (mit passenger multiplier = 10)

Anzahl Threads	1	2	4
Laufzeit [sec]	108.92	65.57	38.41

Auswertung – Laufzeiten



Auswertung – Umlegungsqualität

- Beide Umlegungen sind sehr ähnlich
- VISUM berechnet etwas kürzere Fahrzeiten
- PAT basierter Algorithmus berechnet Journeys mit weniger Umstiegen

	VISUM			PAT basierter Algorithmus		
Eingenschaft	min	mean	max	min	mean	max
Reisezeit [min]	2.98	46.885	429.00	2.98	47.199	429.00
Zeit im Fahrzeug [min]	0.02	21.059	380.00	0.02	21.231	323.97
Laufzeit [min]	2.00	22.394	149.00	2.00	22.476	149.00
Wartezeit [min]	0.00	3.432	217.02	0.00	3.492	217.02
Züge pro Passagier	1.00	1.771	6.00	1.00	1.746	8.00
Connections pro Passagier	1.00	9.396	109.00	1.00	9.474	97.00
Passagiere pro Connection	0.00	12.740	1 290.10	0.00	12.847	1 233.60