Karlsruhe Institute of Technology

Efficient Traffic Assignment for Public Transit Networks

Lars Briem, Sebastian Buck, Holger Ebhart, Nicolai Mallig, Ben Strasser, Peter Vortisch, Dorothea Wagner, and Tobias Zündorf

SEA · July 22nd, 2017

INSTITUTE OF THEORETICAL INFORMATICS · ALGORITHMICS GROUP

Overview

Introduction:

- Public Transit Network
- Demand
- Problem Statement

Our Algorithm:

- Perceived Arrival Time
- Assignment
- Decision Model

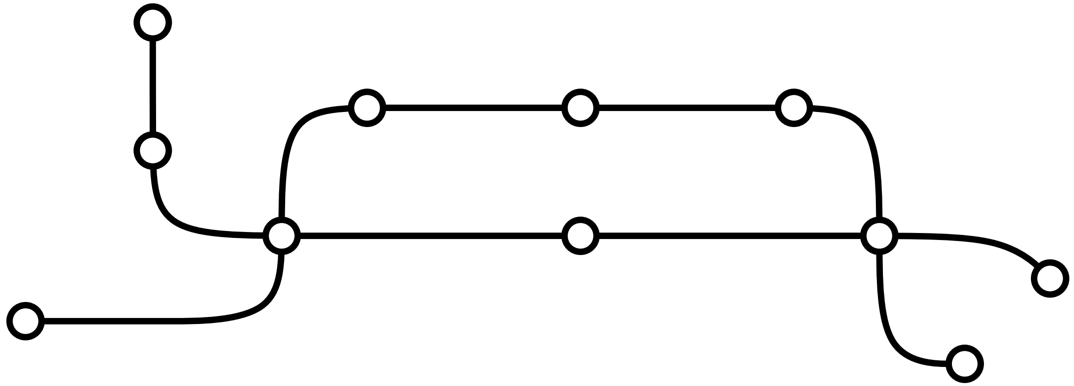
Evaluation:

- Performance
- Result Quality

Public Transit Network

Timetable components:

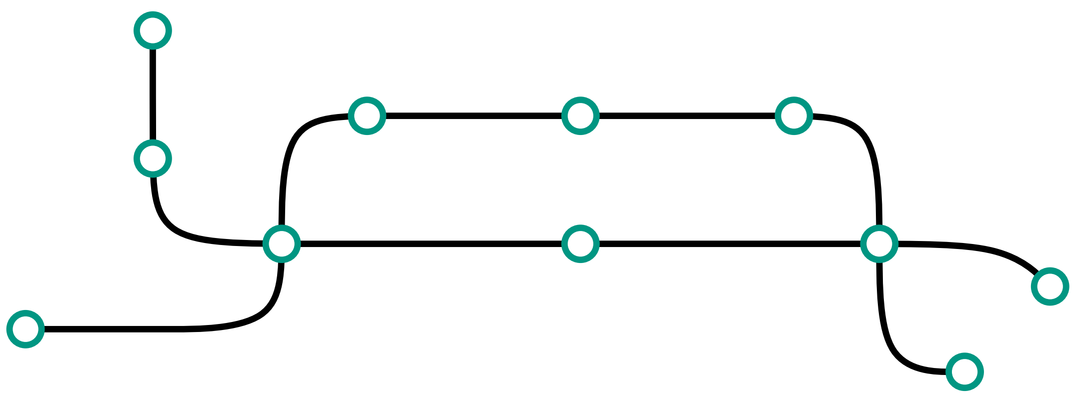
- A set of stops $\mathcal{S}$ (stops, platforms)
- A set of connections $\mathcal{C}$
- A set of trips $\mathcal{T}$ (minimum change times, walking)
- A set of transfer edges $\mathcal{E} \subseteq \mathcal{S} \times \mathcal{S}$ (vehicles)



Public Transit Network

Timetable components:

- A set of stops $\mathcal{S}$ (stops, platforms)
- A set of connections $\mathcal{C}$
- A set of trips $\mathcal{T}$ (minimum change times, walking)
- A set of transfer edges $\mathcal{E} \subseteq \mathcal{S} \times \mathcal{S}$ (vehicles)



Institute of Theoretical Informatics
Algorithmics Group

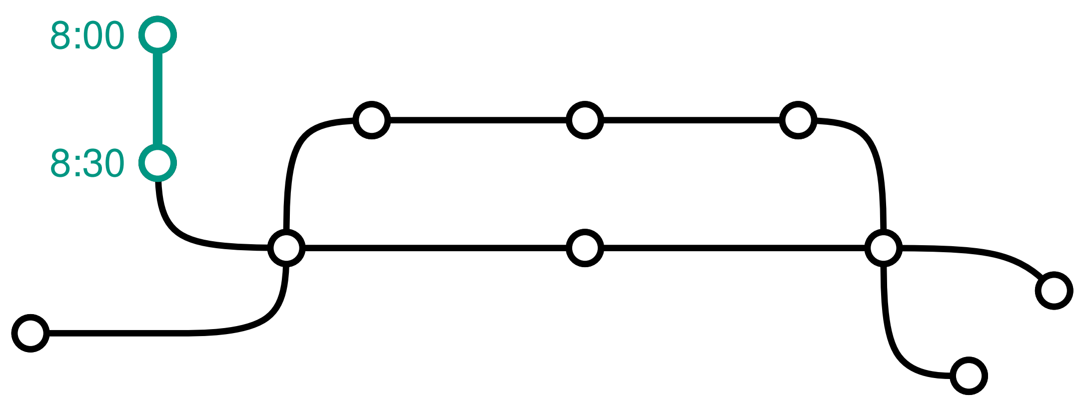
Public Transit Network

Timetable components:

- A set of stops $\mathcal{S}$ (stops, platforms)
- A set of connections $\mathcal{C}$
- A set of trips $\mathcal{T}$ (minimum change times, walking)
- A set of transfer edges $\mathcal{E} \subseteq \mathcal{S} \times \mathcal{S}$ (vehicles)

Connection:

- departure stop $s_{\text{dep}} \in \mathcal{S}$
- arrival stop $s_{\text{arr}} \in \mathcal{S}$
- trip $t \in \mathcal{T}$
- departure time $\tau_{\text{dep}} \in \mathbb{R}$
- arrival time $\tau_{\text{arr}} \in \mathbb{R}$



Institute of Theoretical Informatics
Algorithmics Group

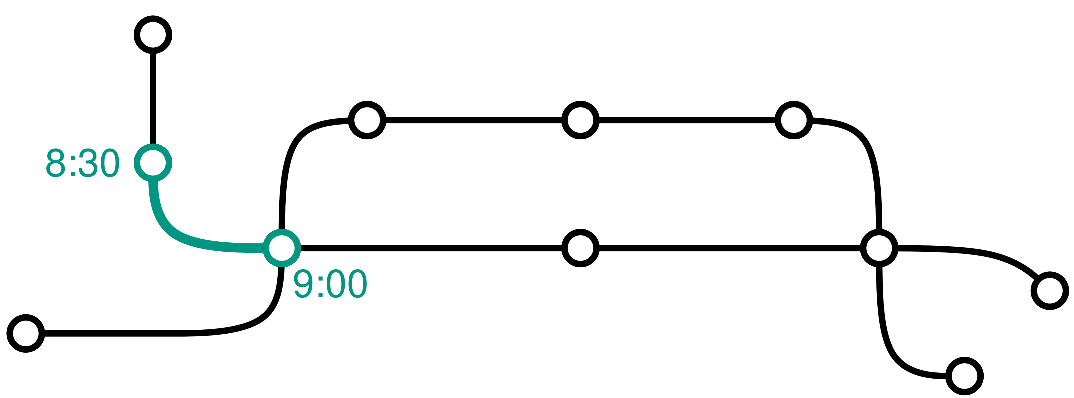
Public Transit Network

Timetable components:

- A set of stops $\mathcal{S}$ (stops, platforms)
- A set of connections $\mathcal{C}$
- A set of trips $\mathcal{T}$ (minimum change times, walking)
- A set of transfer edges $\mathcal{E} \subseteq \mathcal{S} \times \mathcal{S}$ (vehicles)

Connection:

- departure stop $s_{\text{dep}} \in \mathcal{S}$
- arrival stop $s_{\text{arr}} \in \mathcal{S}$
- trip $t \in \mathcal{T}$
- departure time $\tau_{\text{dep}} \in \mathbb{R}$
- arrival time $\tau_{\text{arr}} \in \mathbb{R}$



Institute of Theoretical Informatics
Algorithmics Group

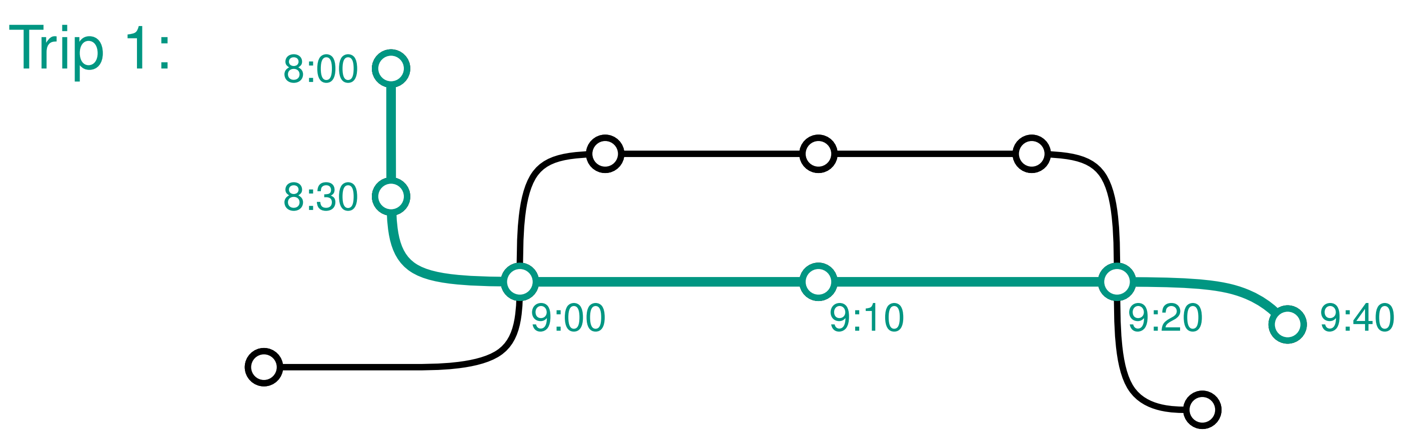
Public Transit Network

Timetable components:

- A set of stops $\mathcal{S}$ (stops, platforms)
- A set of connections $\mathcal{C}$
- A set of trips $\mathcal{T}$ (minimum change times, walking)
- A set of transfer edges $\mathcal{E} \subseteq \mathcal{S} \times \mathcal{S}$ (vehicles)

Trip:

- Subsequent connections served by the same vehicle



Institute of Theoretical Informatics
Algorithmics Group

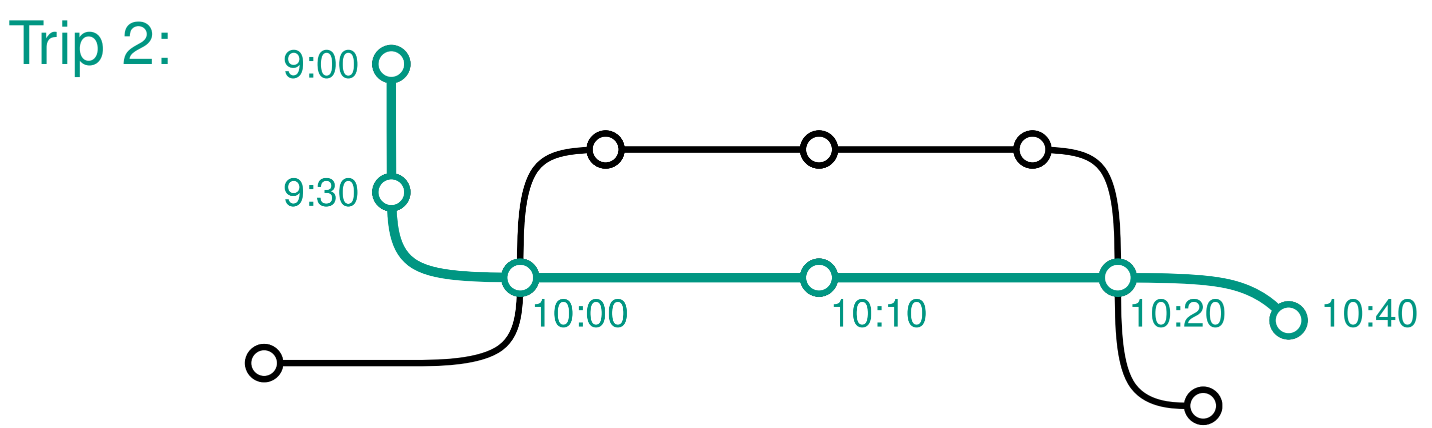
Public Transit Network

Timetable components:

- A set of stops $\mathcal{S}$ (stops, platforms)
- A set of connections $\mathcal{C}$
- A set of trips $\mathcal{T}$ (minimum change times, walking)
- A set of transfer edges $\mathcal{E} \subseteq \mathcal{S} \times \mathcal{S}$ (vehicles)

Trip:

- Subsequent connections served by the same vehicle



Institute of Theoretical Informatics
Algorithmics Group

Public Transit Network

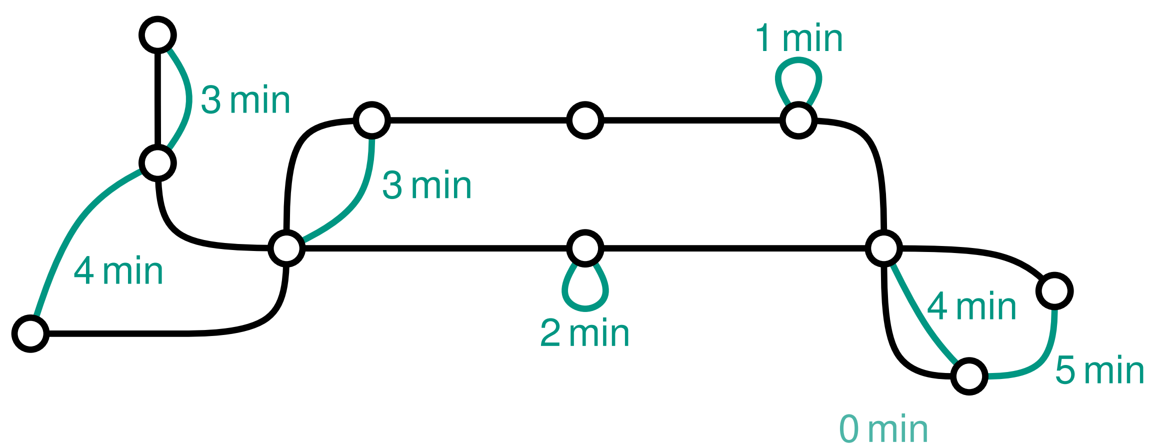
Karlsruhe Institute of Technology

Timetable components:

- A set of stops $\mathcal{S}$ (stops, platforms)
- A set of connections $\mathcal{C}$
- A set of trips $\mathcal{T}$ (minimum change times, walking)
- A set of transfer edges $\mathcal{E} \subseteq \mathcal{S} \times \mathcal{S}$ (vehicles)

Transfer graph:

- Describes possible transfers between stops



Demand

Karlsruhe Institute of Technology

Definition:

- Demand is a list of passengers, each with:
  - An origin stop
  - A destination stop
  - A desired departure time

Example:

Origin	Destination	Departure Time
Paddington	King's Cross	8:00 am
King's Cross	Temple	9:00 am
Paddington	Embankment	9:30 am
Piccadilly Circus	Westminster	9:30 am
. . .		

Institute of Theoretical Informatics
Algorithmics Group

Problem Statement: Assignment

Given:

- A public transit network (timetable & transfer graph)
- Demand

Problem:

- Compute the utilization of every vehicle, at every given time
- Assign all passengers to journeys

Institute of Theoretical Informatics
Algorithmics Group

Problem Statement: Assignment

Karlsruhe Institute of Technology

Given:

- A public transit network (timetable & transfer graph)
- Demand

Problem:

- Compute the utilization of every vehicle, at every given time
- Assign all passengers to journeys

Note:

- A passenger may be assigned proportionally to multiple journeys
- Assigned journeys are not necessarily optimal

Perceived Arrival Time (PAT)

Purpose:

- Associated with a connection *c* and a specific destination *d*
- Measures how useful *c* is for reaching *d*
- Depends on four parameters:
  - Cost for changing between vehicles λ_{trans}
  - Cost factor for waiting λ_{wait}
  - Cost factor for walking λ_{walk}
  - The maximum delay of a connection $\Delta_{\tau}^{\text{max}}$

Assumption:

- Passengers try to optimize their PAT

Institute of Theoretical Informatics
Algorithmics Group

Perceived Arrival Time (PAT)

Karlsruhe Institute of Technology

Formal definition:

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

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

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

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

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

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

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

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

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

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

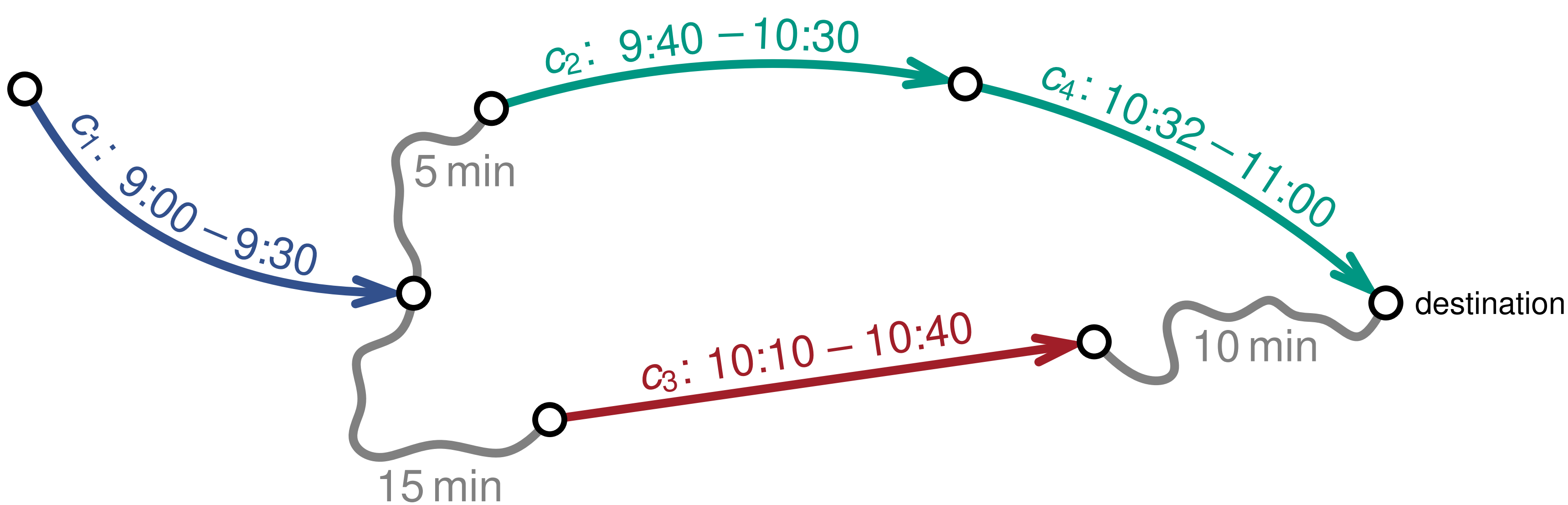
Institute of Theoretical Informatics
Algorithmics Group

Perceived Arrival Time (PAT)

Example:

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

Connection	PAT
c_4	
c_3	
c_2	
c_1	



Institute of Theoretical Informatics
Algorithmics Group

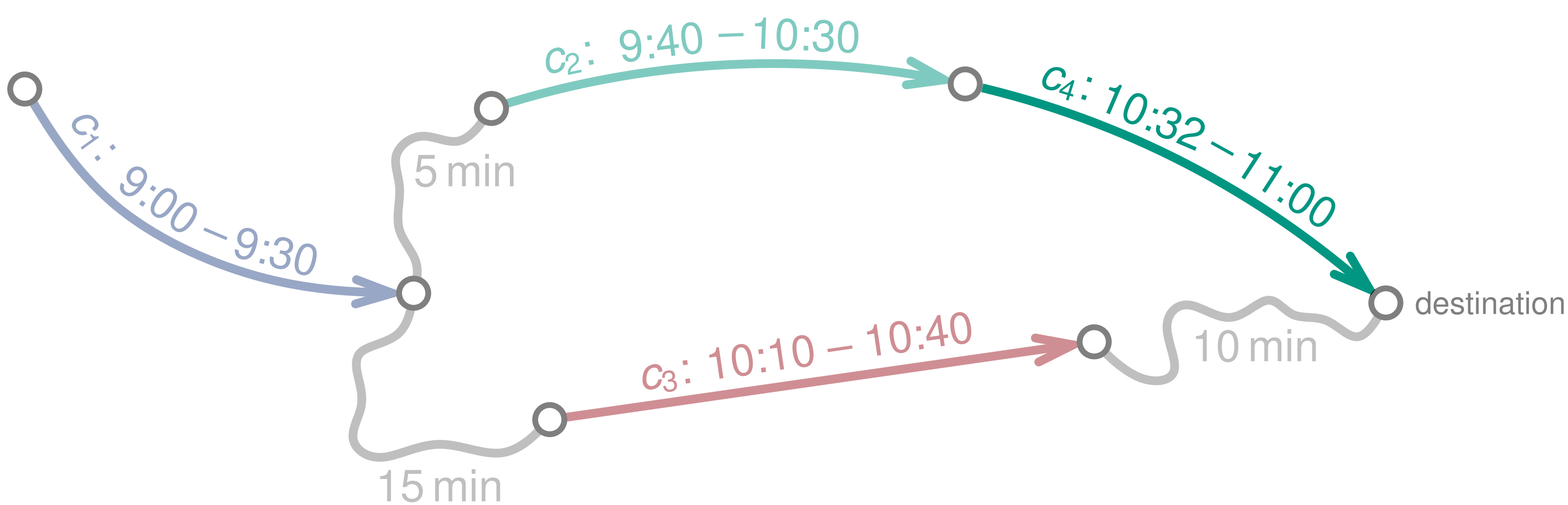
Perceived Arrival Time (PAT)

Example:

- $\lambda_{\text{walk}} = 3, \quad \lambda_{\text{wait}} = 2, \quad \lambda_{\text{trans}} = 5\,\text{min}$
- **Case 1:** Connection c reaches destination

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

Connection	PAT
c_4	11:00
c_3	
c_2	
c_1	



Institute of Theoretical Informatics
Algorithmics Group

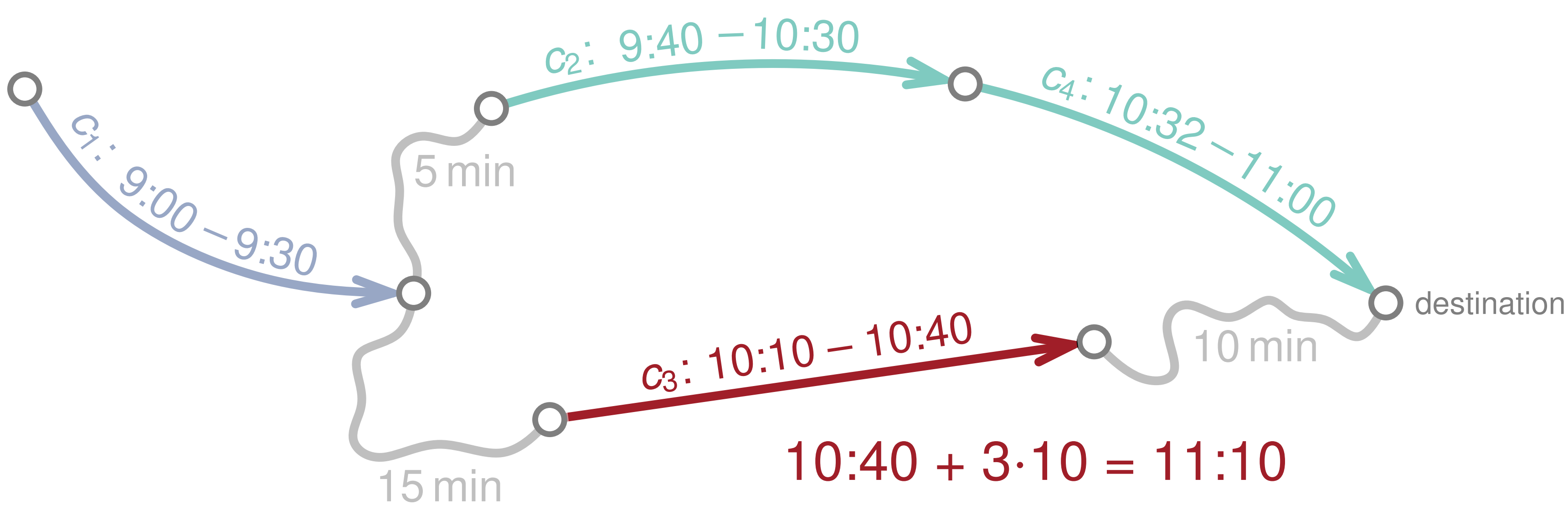
Perceived Arrival Time (PAT)

Example:

- $\lambda_{\text{walk}} = 3, \quad \lambda_{\text{wait}} = 2, \quad \lambda_{\text{trans}} = 5\,\text{min}$
- **Case 2:** Walk from connection c to destination

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

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



Institute of Theoretical Informatics
Algorithmics Group

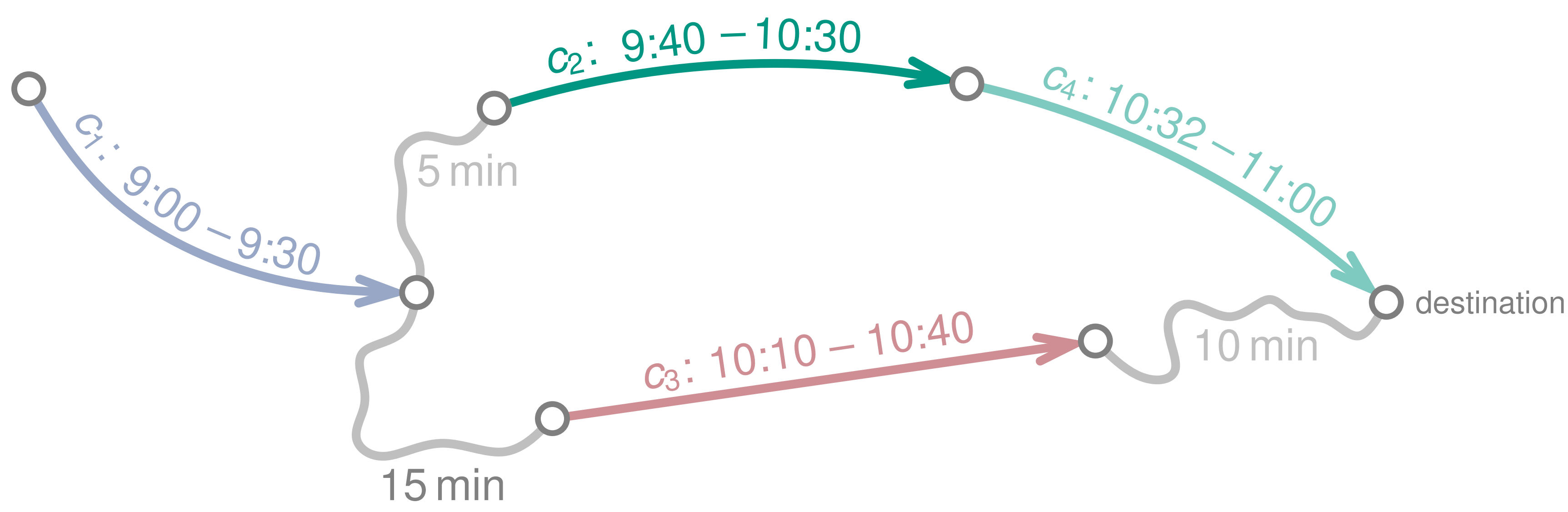
Perceived Arrival Time (PAT)

Example:

- $\lambda_{walk} = 3, \quad \lambda_{wait} = 2, \quad \lambda_{trans} = 5\,\text{min}$
- **Case 3:** Continue with con. c' of same trip

  $\Rightarrow$ PAT = PAT c'

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



Institute of Theoretical Informatics
Algorithmics Group

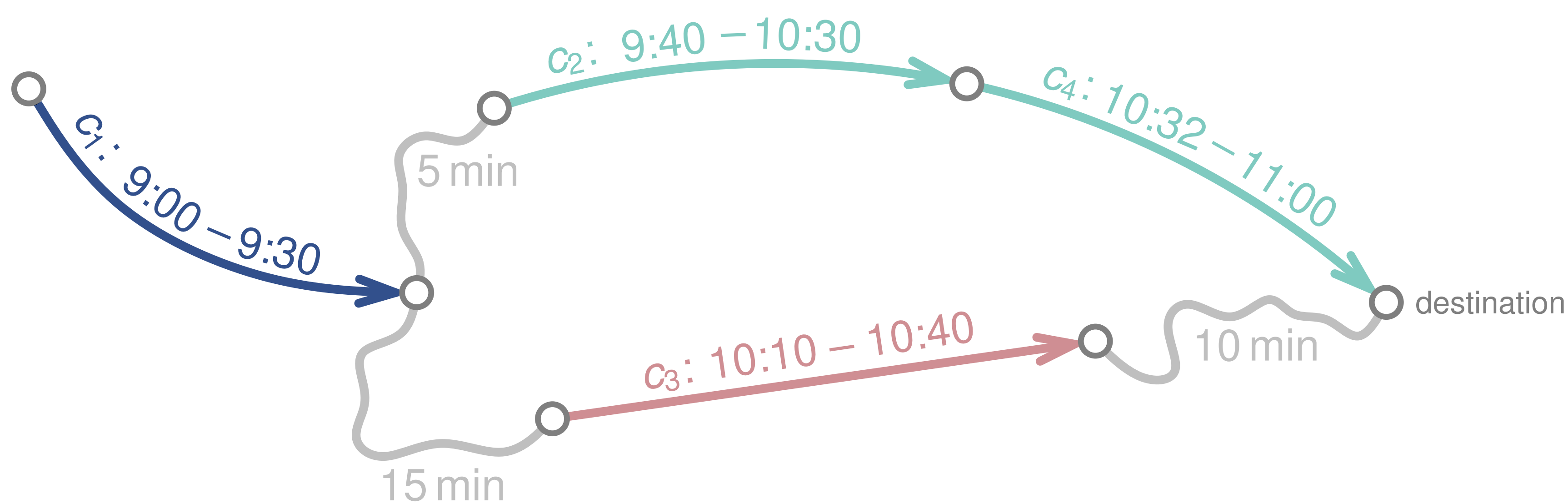
Perceived Arrival Time (PAT)

Karlsruhe Institute of Technology

Example:

- $\lambda_{\text{walk}} = 3, \quad \lambda_{\text{wait}} = 2, \quad \lambda_{\text{trans}} = 5\,\text{min}$
- **Case 4:** Continue with con. c' of different trip

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



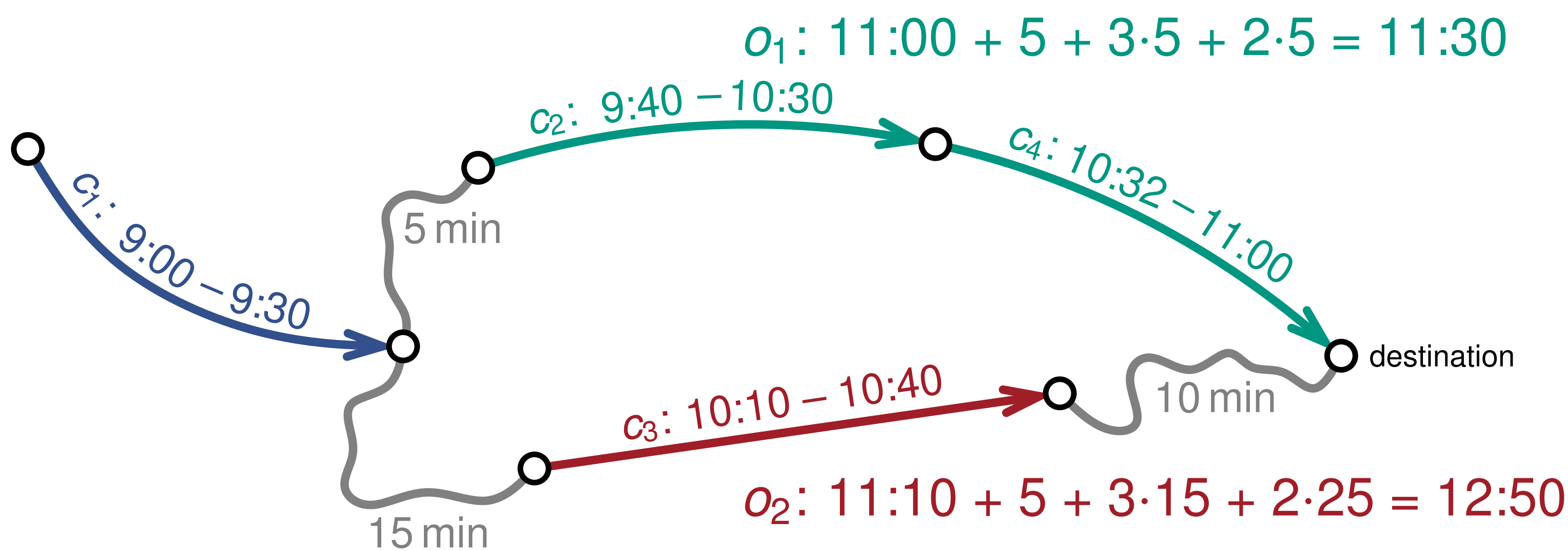
Institute of Theoretical Informatics
Algorithmics Group

Perceived Arrival Time (PAT)

Example:

- $\lambda_{walk} = 3, \quad \lambda_{wait} = 2, \quad \lambda_{trans} = 5\,\text{min}$
- **Case 4:** Continue with con. c' of different trip

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



Institute of Theoretical Informatics
Algorithmics Group

Perceived Arrival Time (PAT)

Example:

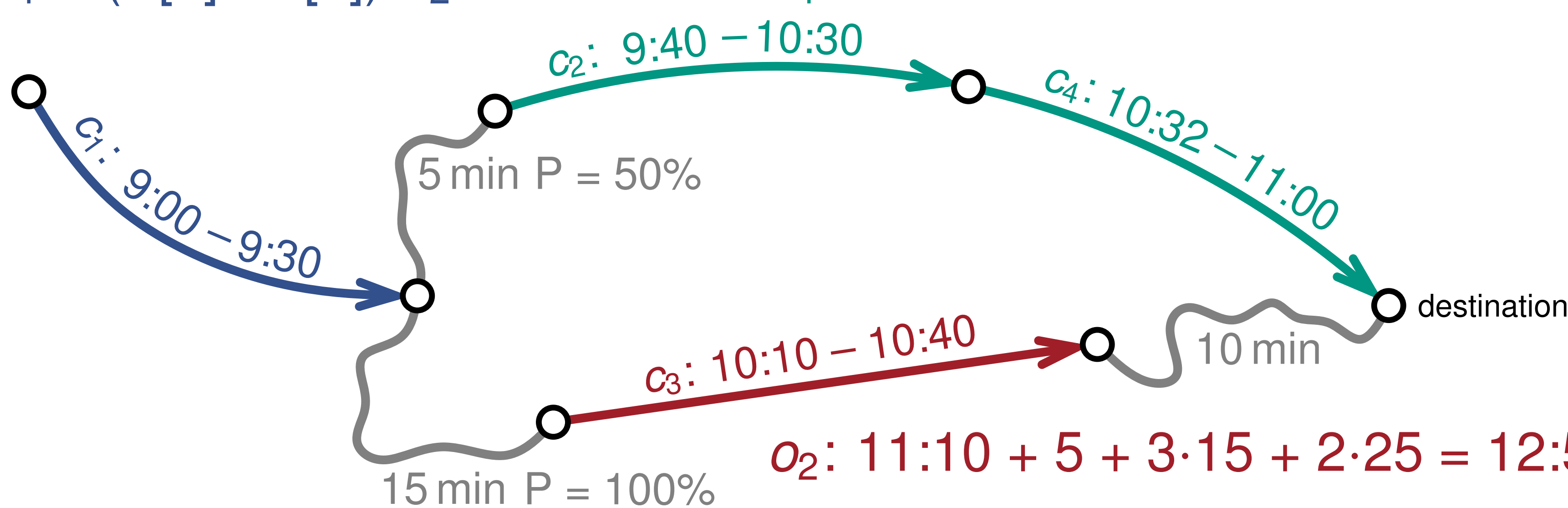
- $\lambda_{\text{walk}} = 3, \quad \lambda_{\text{wait}} = 2, \quad \lambda_{\text{trans}} = 5\,\text{min}$
- **Case 4:** Continue with some option o_i

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

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

$P[1] \cdot o_1 + (P[2] - P[1]) \cdot o_2$

o_1: 11:00 + 5 + 3·5 + 2·5 = 11:30



o_2: 11:10 + 5 + 3·15 + 2·25 = 12:50

Institute of Theoretical Informatics
Algorithmics Group

Perceived Arrival Time (PAT)

Karlsruhe Institute of Technology

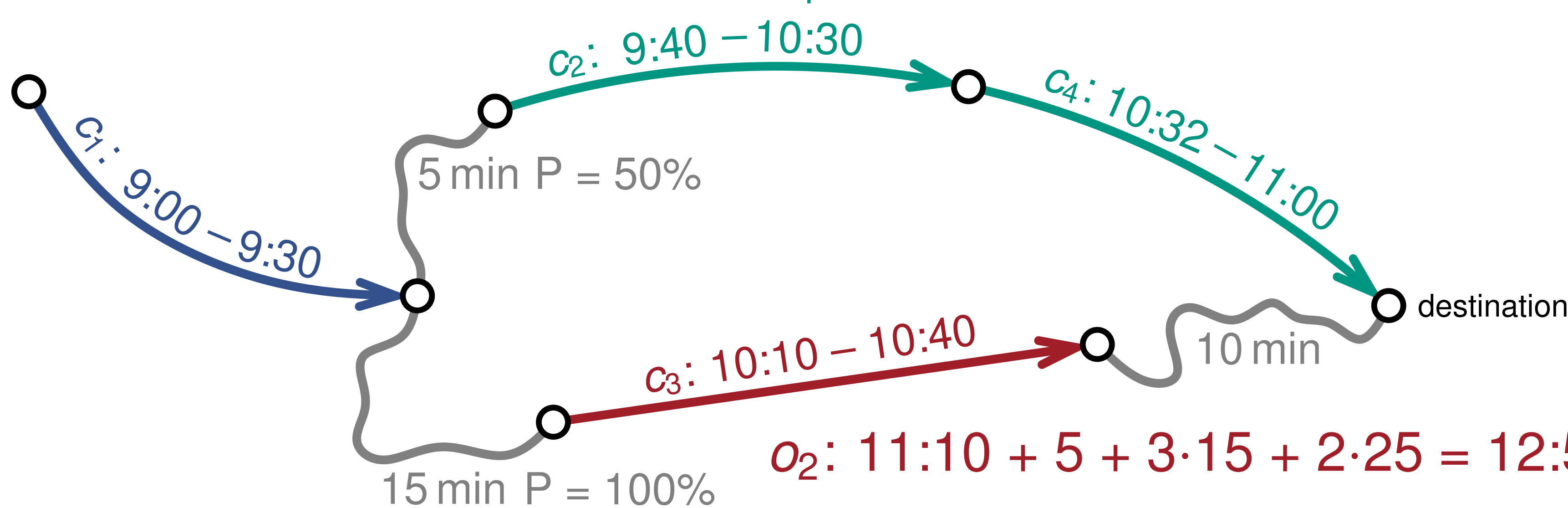
Example:

- $\lambda_{walk} = 3, \quad \lambda_{wait} = 2, \quad \lambda_{trans} = 5\,\text{min}$
- **Case 4:** Continue with some option o_i

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

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

$0.5 \cdot 11{:}30 + 0.5 \cdot 12{:}50 = 12{:}10$ $\quad o_1: 11{:}00 + 5 + 3{\cdot}5 + 2{\cdot}5 = 11{:}30$



$o_2: 11{:}10 + 5 + 3{\cdot}15 + 2{\cdot}25 = 12{:}50$

Institute of Theoretical Informatics
Algorithmics Group

Our Algorithm

Concept:

- Simulate passengers movement through the network
- Decide per connection c, which passengers use c
- Passengers with same destination meet
  $\Rightarrow$ Have to make the same decisions
  $\Rightarrow$ Algorithm can benefit from synergy effects

Institute of Theoretical Informatics
Algorithmics Group

Our Algorithm

Concept:

- Simulate passengers movement through the network
- Decide per connection c, which passengers use c
- Passengers with same destination meet
  $\Rightarrow$ Have to make the same decisions
  $\Rightarrow$ Algorithm can benefit from synergy effects

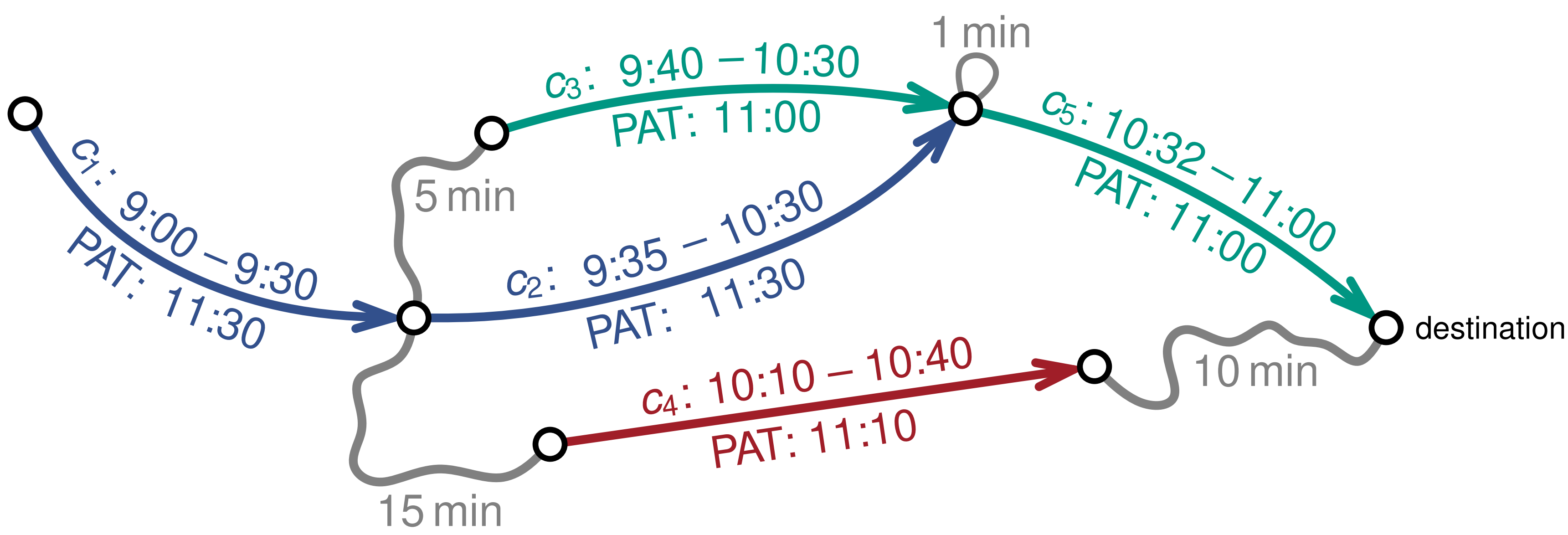
Overview:

- Sort passengers by destination
- Compute assignment for each destination in 3 steps:
  - Compute PATs for every connection
  - Simulate passenger movement based on PATs
  - Remove unwanted cycles from journeys (optional)

Assignment Computation – Example

- Process connections in ascending order by departure time
- Decide whether passengers use a connection or not

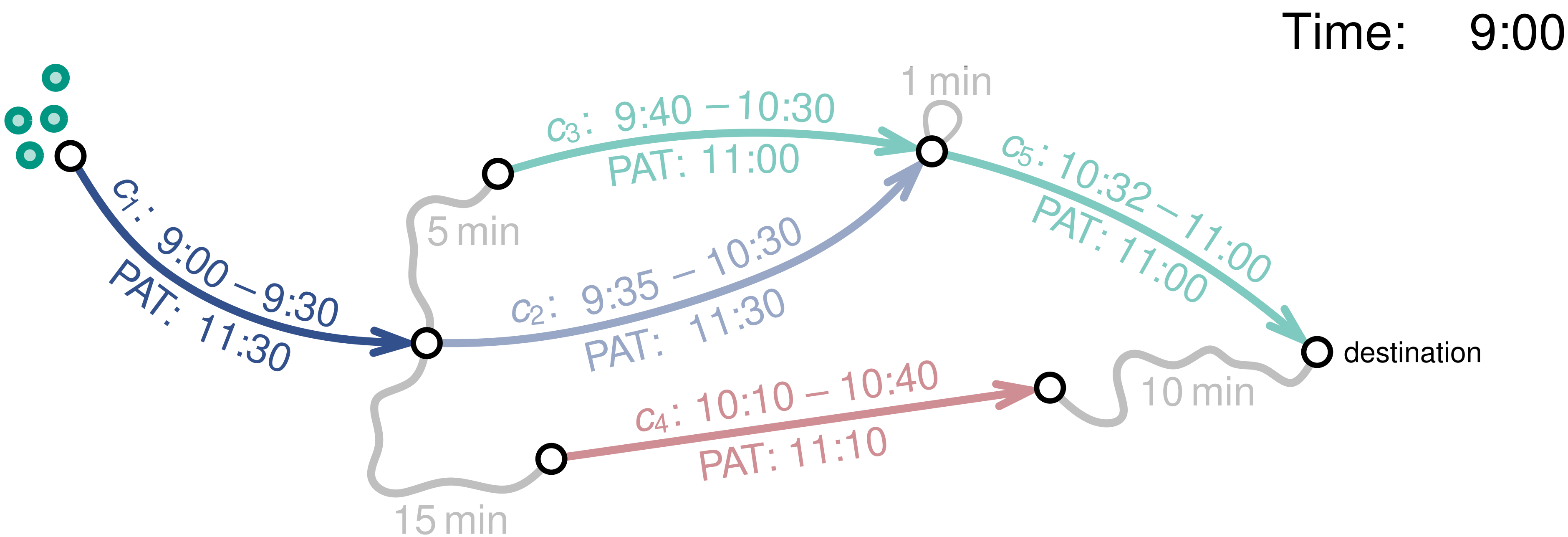
Time: 0:00



Assignment Computation – Example

- Process connections in ascending order by departure time
- Decide whether passengers use a connection or not

1. Generate passengers from demand



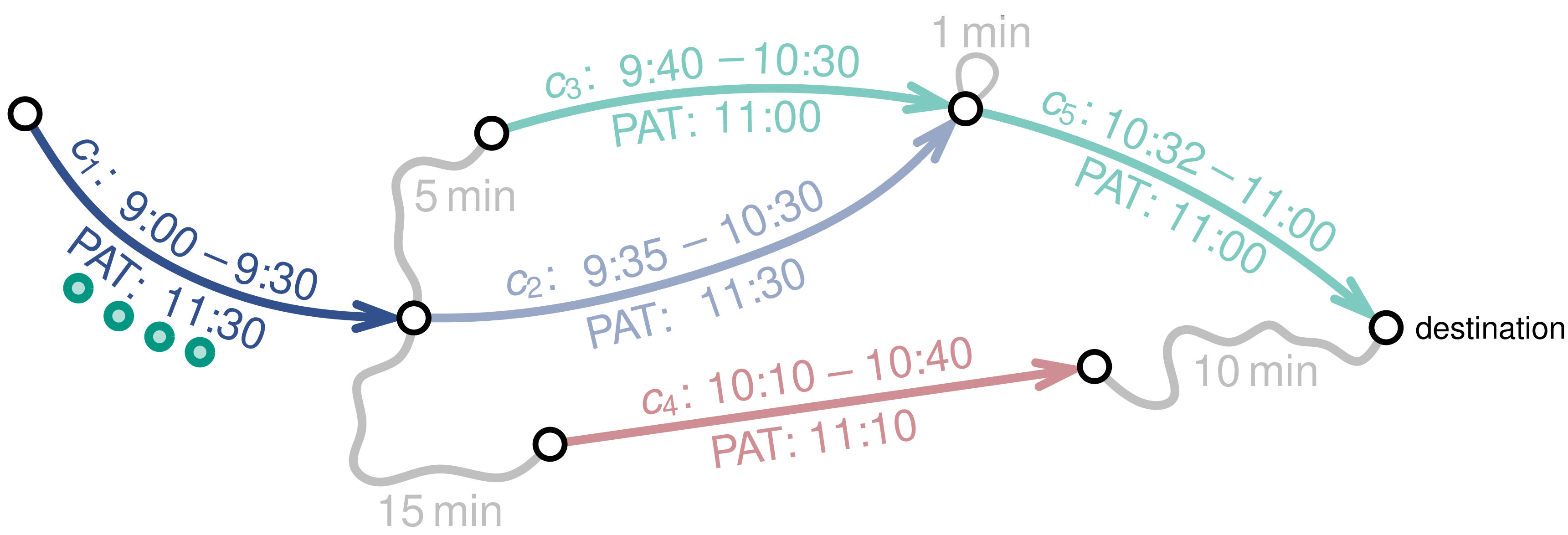
Institute of Theoretical Informatics
Algorithmics Group

Assignment Computation – Example

- Process connections in ascending order by departure time
- Decide whether passengers use a connection or not

2. Decide which passengers enter the connection

Time: 9:00



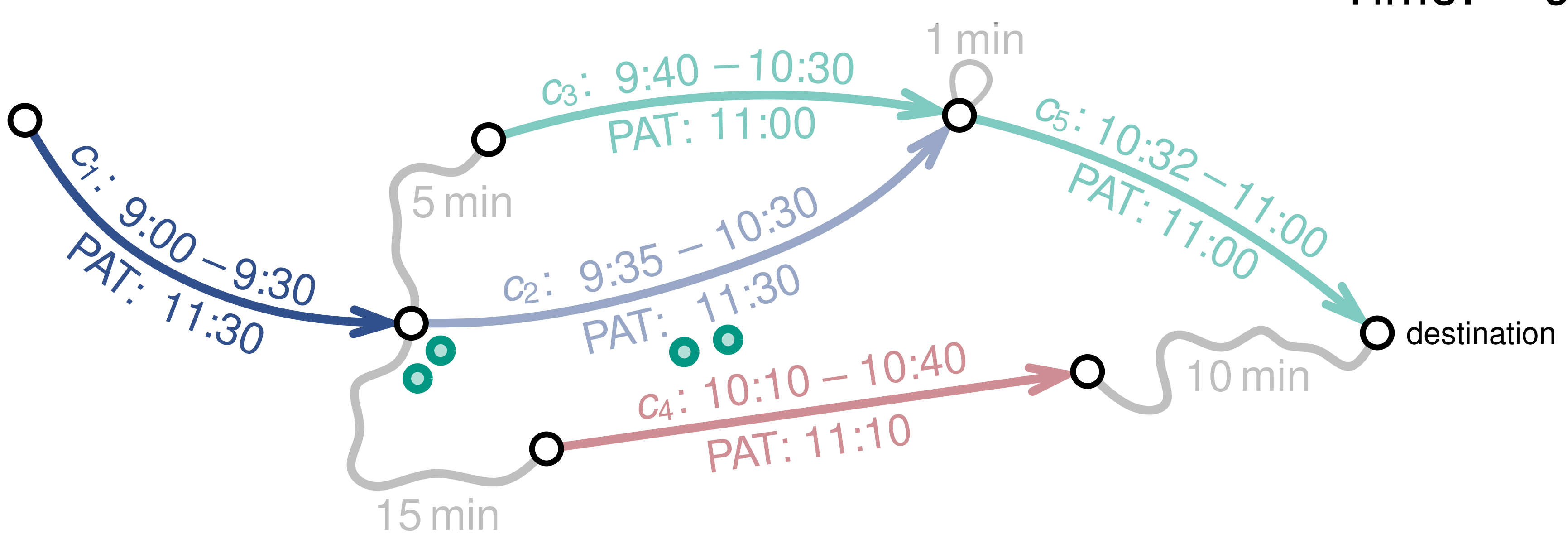
Institute of Theoretical Informatics
Algorithmics Group

Assignment Computation – Example

- Process connections in ascending order by departure time
- Decide whether passengers use a connection or not

3. Decide which passengers leave the trip

Time: 9:00



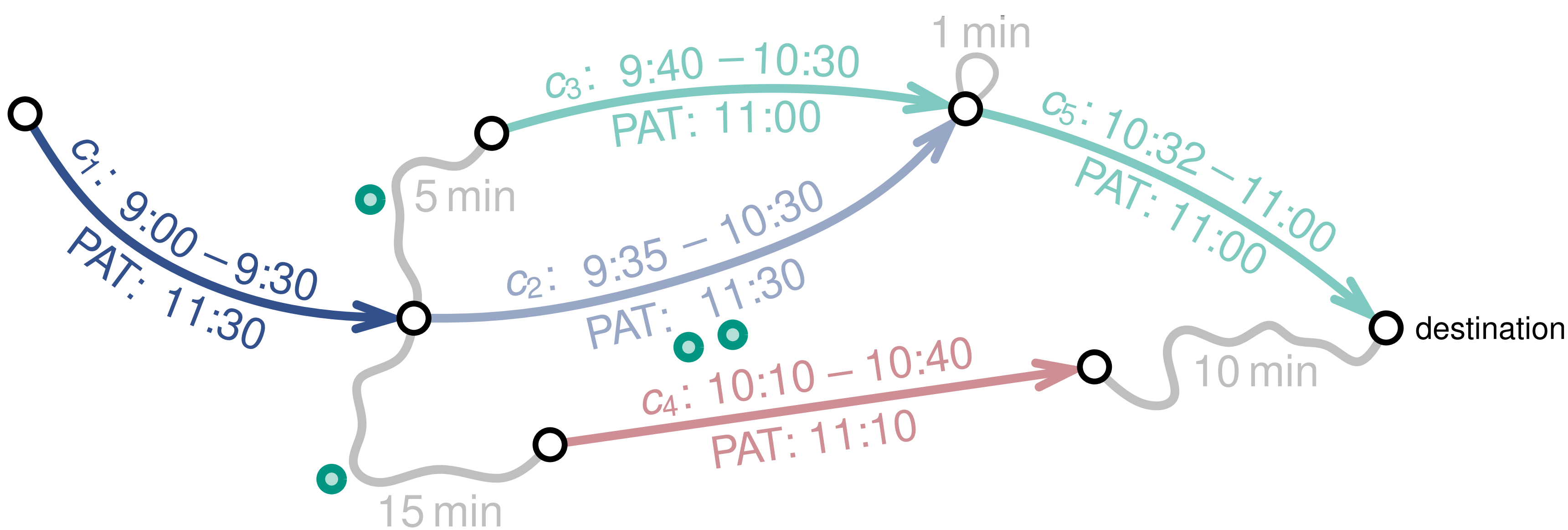
Institute of Theoretical Informatics
Algorithmics Group

Assignment Computation – Example

- Process connections in ascending order by departure time
- Decide whether passengers use a connection or not

4. Move disembarking passengers to their next stop

Time: 9:00



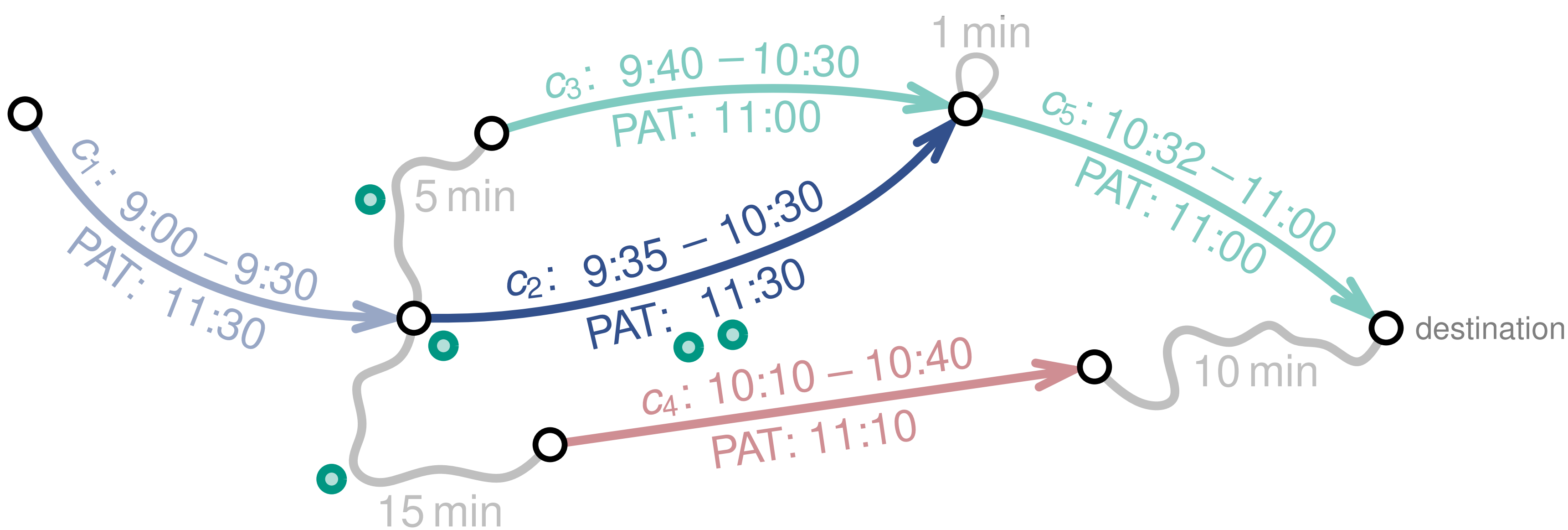
Institute of Theoretical Informatics
Algorithmics Group

Assignment Computation – Example

- Process connections in ascending order by departure time
- Decide whether passengers use a connection or not

1. Generate passengers from demand

Time: 9:35

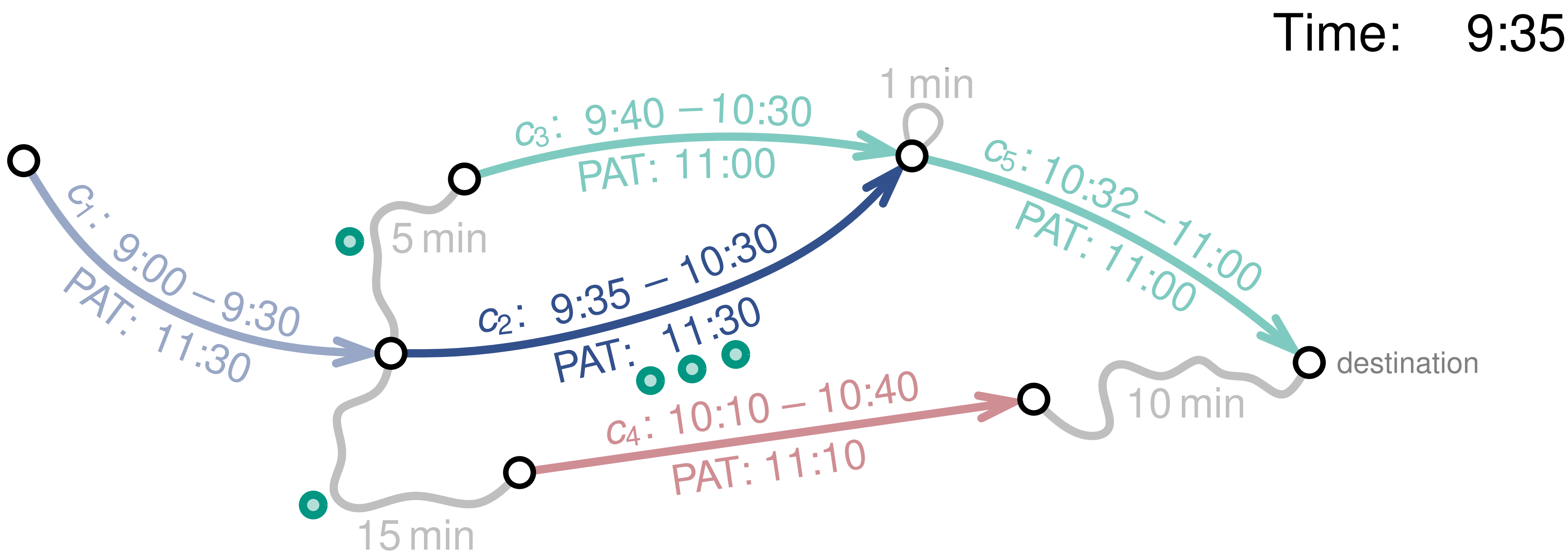


Institute of Theoretical Informatics
Algorithmics Group

Assignment Computation – Example

- Process connections in ascending order by departure time
- Decide whether passengers use a connection or not

2. Decide which passengers enter the connection

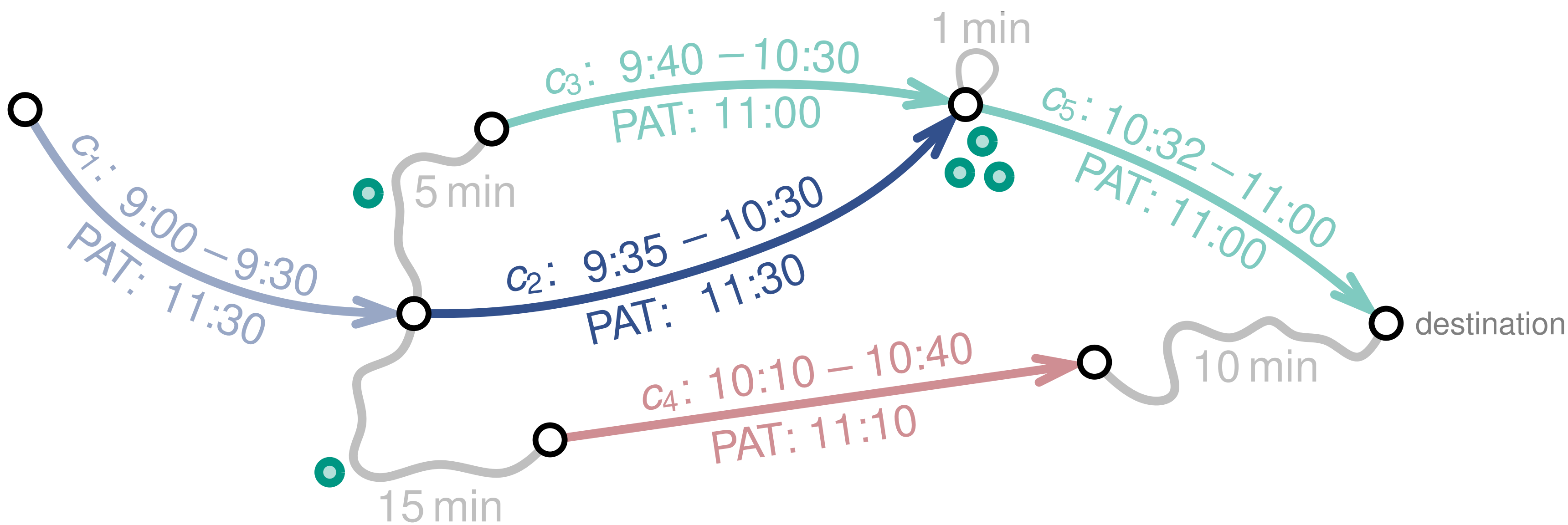


Assignment Computation – Example

- Process connections in ascending order by departure time
- Decide whether passengers use a connection or not

3. Decide which passengers leave the trip

Time: 9:35



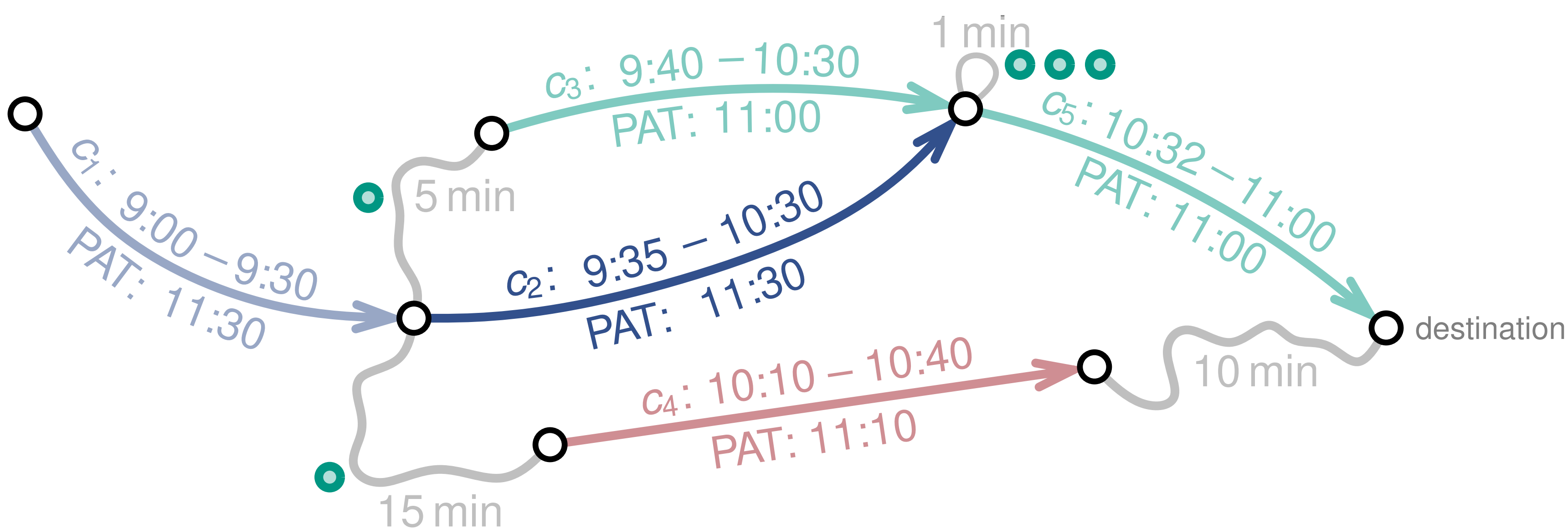
Institute of Theoretical Informatics
Algorithmics Group

Assignment Computation – Example

- Process connections in ascending order by departure time
- Decide whether passengers use a connection or not

4. Move disembarking passengers to their next stop

Time: 9:35



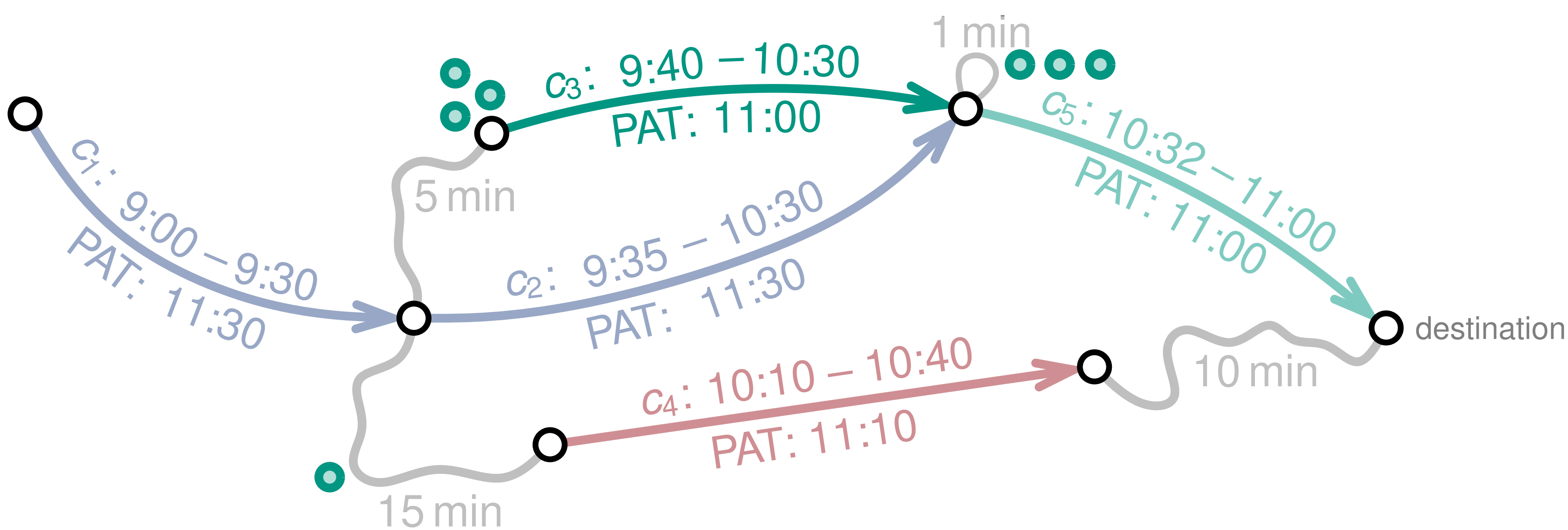
Institute of Theoretical Informatics
Algorithmics Group

Assignment Computation – Example

- Process connections in ascending order by departure time
- Decide whether passengers use a connection or not

1. Generate passengers from demand

Time: 9:40

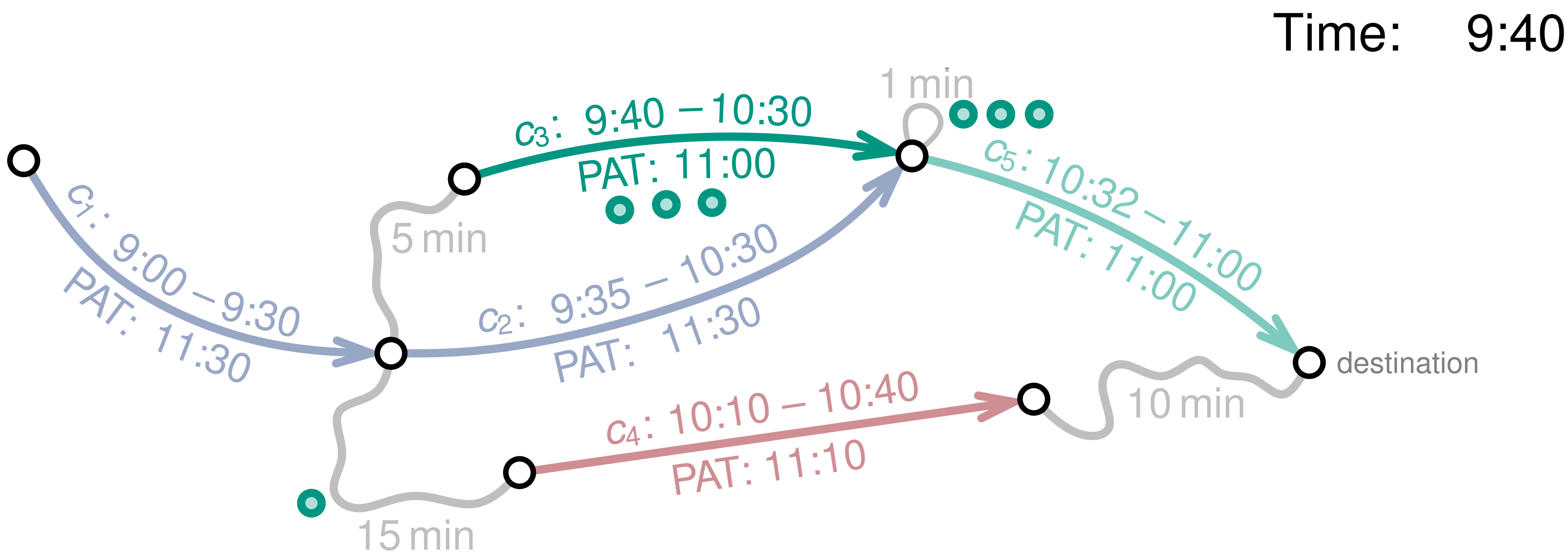


Institute of Theoretical Informatics
Algorithmics Group

Assignment Computation – Example

- Process connections in ascending order by departure time
- Decide whether passengers use a connection or not

2. Decide which passengers enter the connection



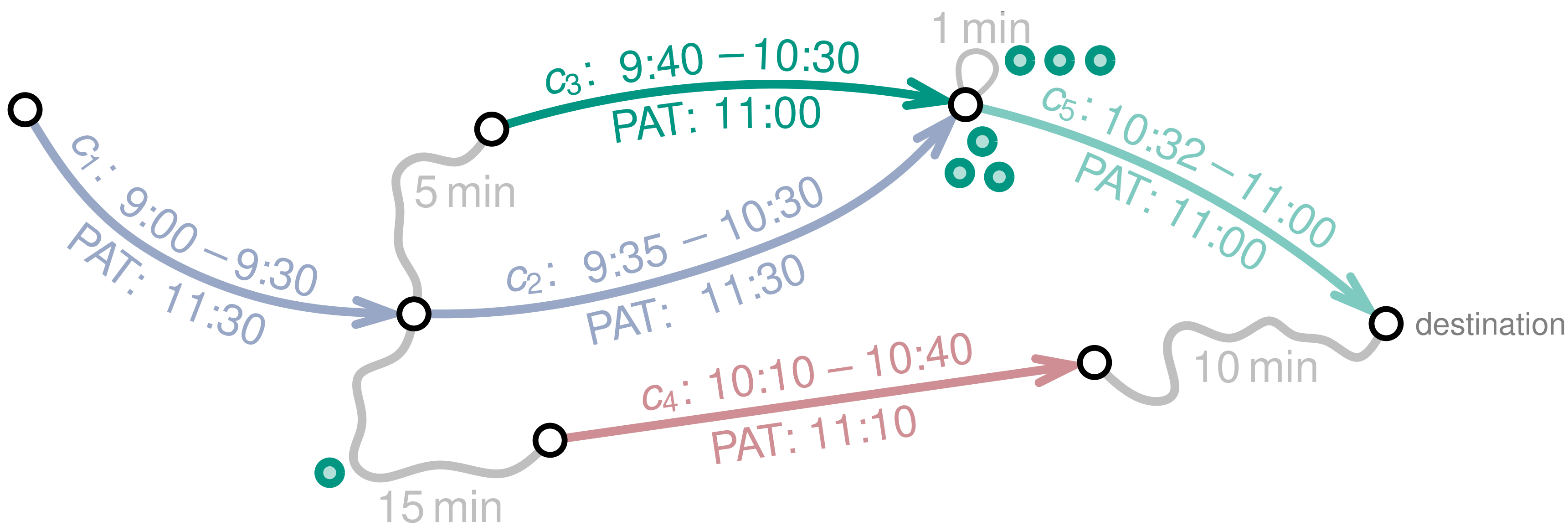
Institute of Theoretical Informatics
Algorithmics Group

Assignment Computation – Example

- Process connections in ascending order by departure time
- Decide whether passengers use a connection or not

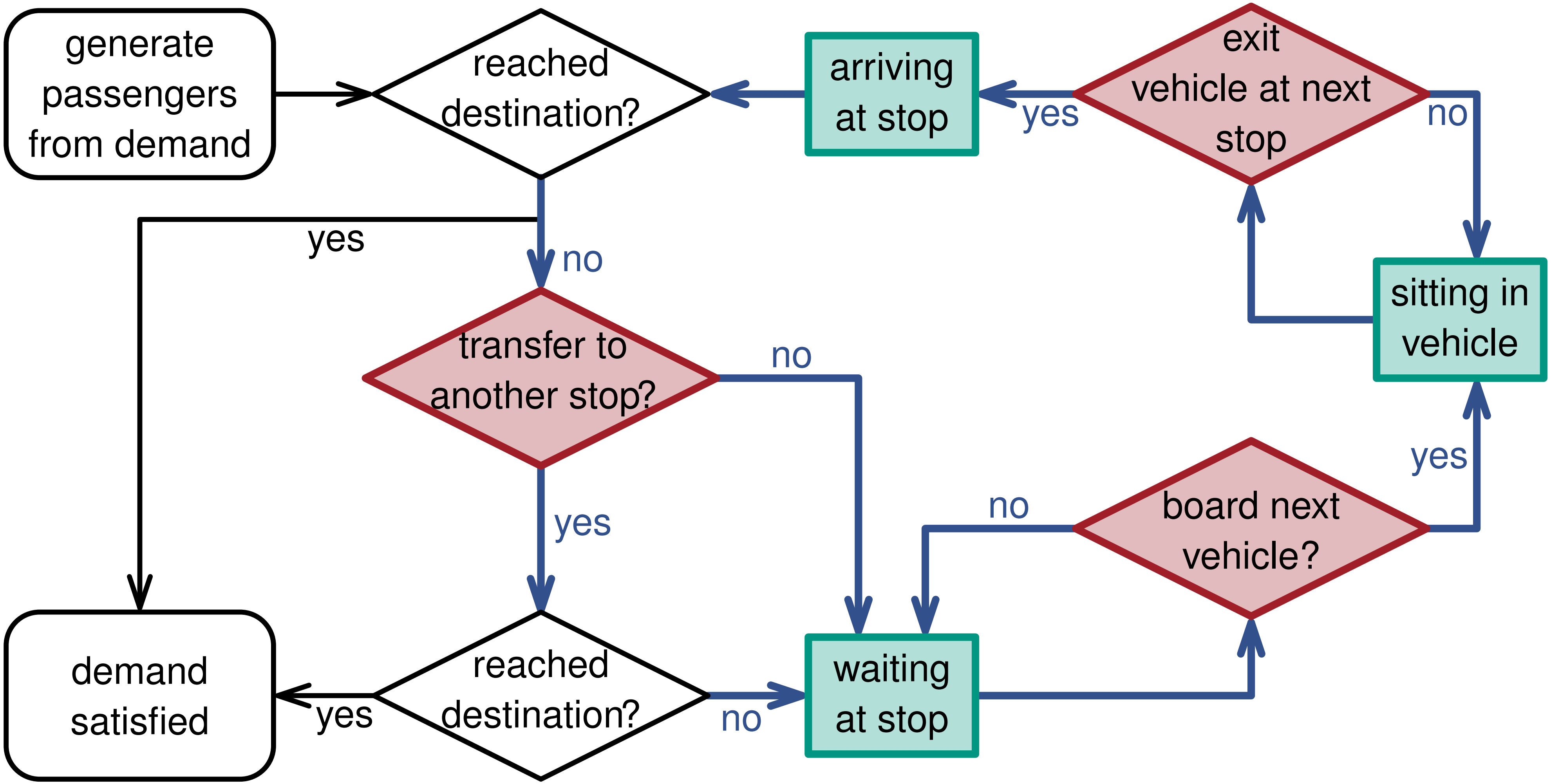
3. ...

Time: 9:40



Institute of Theoretical Informatics
Algorithmics Group

Assignment Computation – Decision Graph



Institute of Theoretical Informatics
Algorithmics Group

Decision Model

Karlsruhe Institute of Technology

Purpose:

- Determines which connections a passenger takes
- Depends on the passenger's delay tolerance $\lambda_{\Delta\text{max}}$

Institute of Theoretical Informatics
Algorithmics Group

Decision Model

Karlsruhe Institute of Technology

Purpose:

- Determines which connections a passenger takes
- Depends on the passenger's delay tolerance $\lambda_{\Delta\max}$

Definition:

- Given the options $o_1, \ldots, o_k$ and their PATs
- Assign a gain $g(i)$ to every option:

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

- The probability $P[i]$ that a passenger chooses option i is:

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

Institute of Theoretical Informatics
Algorithmics Group

Cycles

Karlsruhe Institute of Technology

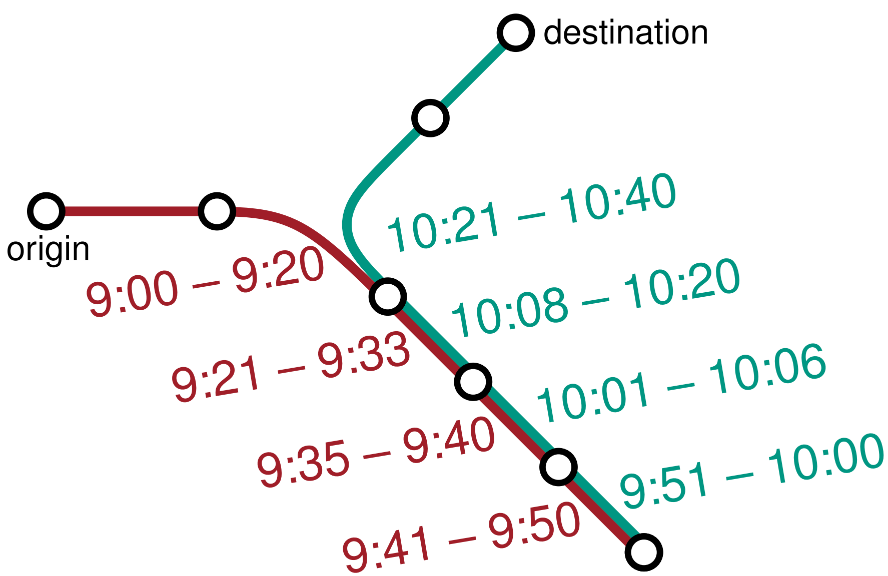
Cycle definition:

- Visiting a stop more than once
- Assigning cycles might be undesirable
- Journey with cycle can have minimum PAT
- High waiting cost leads to cycles

Karlsruhe Institute of Technology

Cycle definition:

- Visiting a stop more than once
- Assigning cycles might be undesirable
- Journey with cycle can have minimum PAT
- High waiting cost leads to cycles

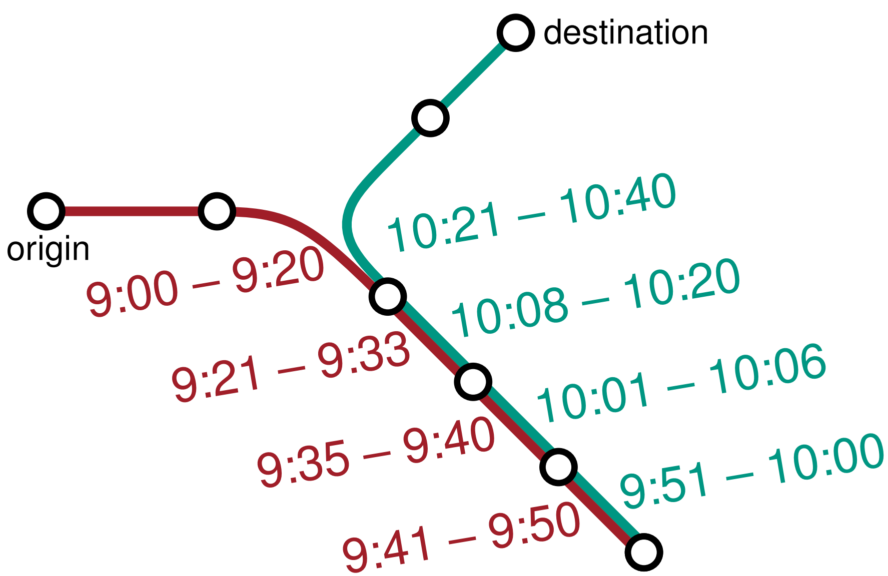


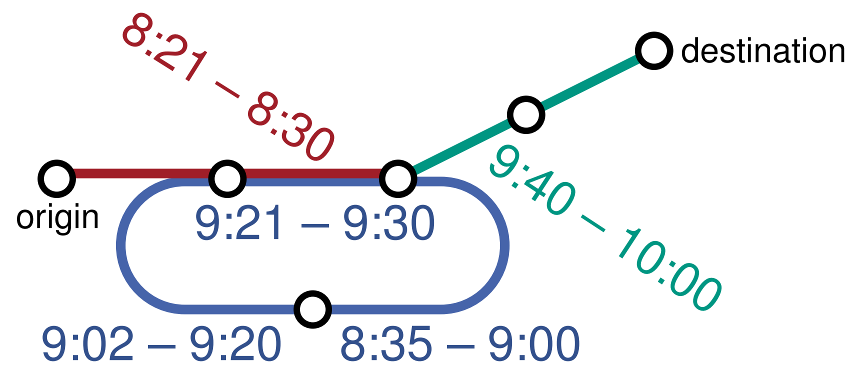
Institute of Theoretical Informatics
Algorithmics Group

Cycles

Cycle definition:

- Visiting a stop more than once
- Assigning cycles might be undesirable
- Journey with cycle can have minimum PAT
- High waiting cost leads to cycles



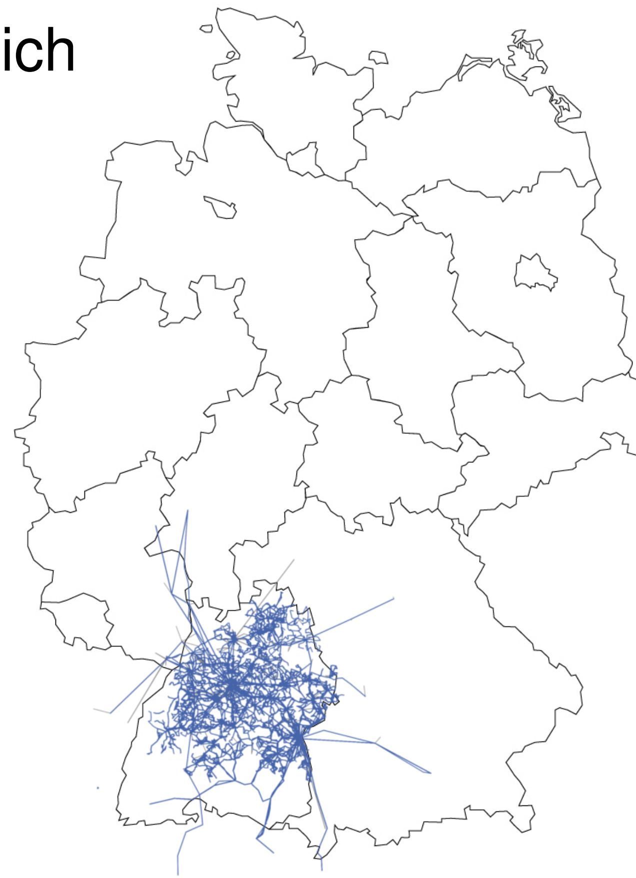


Evaluation

Instance:

- Greater region of Stuttgart
- Reaching as far as Frankfurt, Basel or Munich
- Comprises the traffic of one day

Number of vertices	15 115
Number of stops	13 941
Number of edges	33 890
Number of edges without loops	18 775
Number of connections	780 042
Number of trips	47 844
Number of passenger	1 249 910



Evaluation – Running Time

Karlsruhe Institute of Technology

Used parameters:

- Walking cost factor $\lambda_{walk} = 2$
- Waiting cost factor $\lambda_{wait} = 0.5$
- Transfer cost $\lambda_{trans} = 5\,\text{min}$
- Delay tolerance $\lambda_{\Delta\text{max}} = 5\,\text{min}$
- Max delay $\Delta_{\tau}^{\text{max}} = 1\,\text{min}$

Running time comparison:

- VISUM running time $\approx$ 30 min (using 8 threads)
- Our algorithm: (passenger multiplier = 10)

Number of threads	1	2	4
Running time [sec]	108.92	65.57	38.41

Institute of Theoretical Informatics
Algorithmics Group

Evaluation – Running Time

Karlsruhe Institute of Technology

Used parameters:

- Walking cost factor $\lambda_{walk} = 2$
- Waiting cost factor $\lambda_{wait} = 0.5$
- Transfer cost $\lambda_{trans} = 5\,\text{min}$
- Delay tolerance $\lambda_{\Delta max} = 5\,\text{min}$

} No measurable influence on the running time

- Max delay $\Delta_{\tau}^{max} = 1\,\text{min}$

} Influence the running time

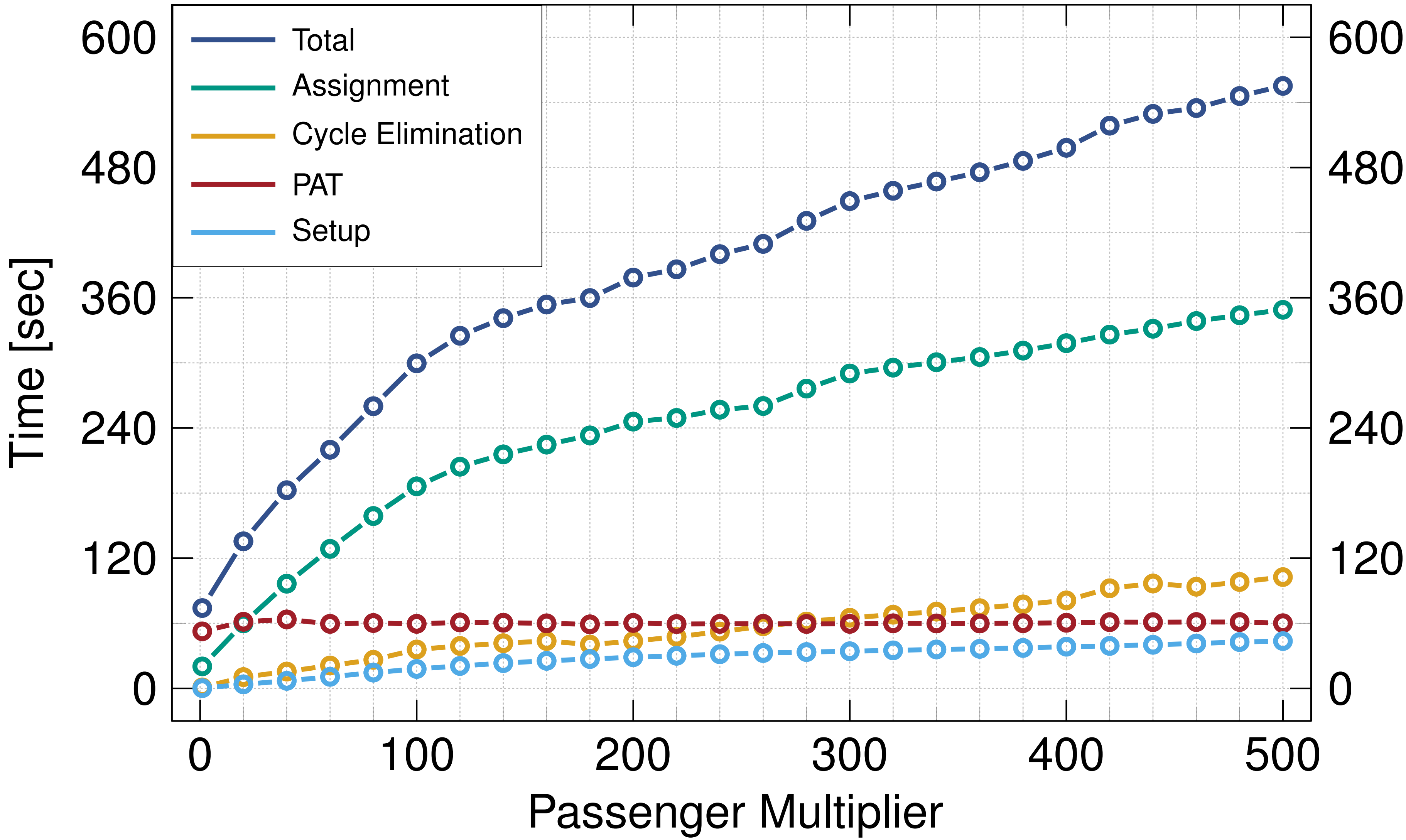
Running time comparison:

- VISUM running time $\approx 30\,\text{min}$ (using 8 threads)
- Our algorithm: (passenger multiplier = 10)

Number of threads	1	2	4
Running time [sec]	108.92	65.57	38.41

Institute of Theoretical Informatics
Algorithmics Group

Evaluation – Running Time



Institute of Theoretical Informatics
Algorithmics Group

Evaluation – Assignment Quality

- Both assignments look similar
- VISUM produces a slightly lower travel time
- Our algorithm produces a slightly lower number of trips

	VISUM			Our Algorithm		
Quantity	min	mean	max	min	mean	max
Total travel time [min]	2.98	46.885	429.00	2.98	47.199	429.00
Time spent in vehicle [min]	0.02	21.059	380.00	0.02	21.231	323.97
Time spent walking [min]	2.00	22.394	149.00	2.00	22.476	149.00
Time spent waiting [min]	0.00	3.432	217.02	0.00	3.492	217.02
Trips per passenger	1.00	1.771	6.00	1.00	1.746	8.00
Connections per passenger	1.00	9.396	109.00	1.00	9.474	97.00
Passengers per connection	0.00	12.740	1 290.10	0.00	12.847	1 233.60

Evaluation – Assignment Quality

- Both assignments look similar
- VISUM produces a slightly lower travel time
- Our algorithm produces a slightly lower number of trips

	VISUM			Our Algorithm		
Quantity	min	mean	max	min	mean	max
Total travel time [min]	2.98	**46.885**	429.00	2.98	**47.199**	429.00
Time spent in vehicle [min]	0.02	21.059	380.00	0.02	21.231	323.97
Time spent walking [min]	2.00	22.394	149.00	2.00	22.476	149.00
Time spent waiting [min]	0.00	3.432	217.02	0.00	3.492	217.02
Trips per passenger	1.00	1.771	6.00	1.00	1.746	8.00
Connections per passenger	1.00	9.396	109.00	1.00	9.474	97.00
Passengers per connection	0.00	12.740	1 290.10	0.00	12.847	1 233.60

Evaluation – Assignment Quality

- Both assignments look similar
- VISUM produces a slightly lower travel time
- Our algorithm produces a slightly lower number of trips

	VISUM			Our Algorithm		
Quantity	min	mean	max	min	mean	max
Total travel time [min]	2.98	46.885	429.00	2.98	47.199	429.00
Time spent in vehicle [min]	0.02	21.059	380.00	0.02	21.231	323.97
Time spent walking [min]	2.00	22.394	149.00	2.00	22.476	149.00
Time spent waiting [min]	0.00	3.432	217.02	0.00	3.492	217.02
Trips per passenger	1.00	**1.771**	6.00	1.00	**1.746**	8.00
Connections per passenger	1.00	9.396	109.00	1.00	9.474	97.00
Passengers per connection	0.00	12.740	1 290.10	0.00	12.847	1 233.60

Institute of Theoretical Informatics
Algorithmics Group

Evaluation – Assignment Quality

- Both assignments look similar
- VISUM produces a slightly lower travel time
- Our algorithm produces a slightly lower number of trips

	VISUM			Our Algorithm		
Quantity	min	mean	max	min	mean	max
Total travel time [min]	2.98	46.885	429.00	2.98	47.199	429.00
Time spent in vehicle [min]	0.02	21.059	380.00	0.02	21.231	323.97
Time spent walking [min]	2.00	22.394	149.00	2.00	22.476	149.00
Time spent waiting [min]	0.00	3.432	217.02	0.00	3.492	217.02
Trips per passenger	1.00	1.771	6.00	1.00	1.746	8.00
Connections per passenger	1.00	9.396	109.00	1.00	9.474	97.00
Passengers per connection	0.00	12.740	**1 290.10**	0.00	12.847	**1 233.60**

Institute of Theoretical Informatics
Algorithmics Group

Thank you for your attention!