Efficient Route Planning

SS 2012

Lecture 10, Wednesday July 11th, 2012

(Multi-criteria costs, Multi-label Dijkstra)

Prof. Dr. Hannah Bast
Chair of Algorithms and Data Structures
Department of Computer Science
University of Freiburg

UNI FREIBURG

Overview of this lecture

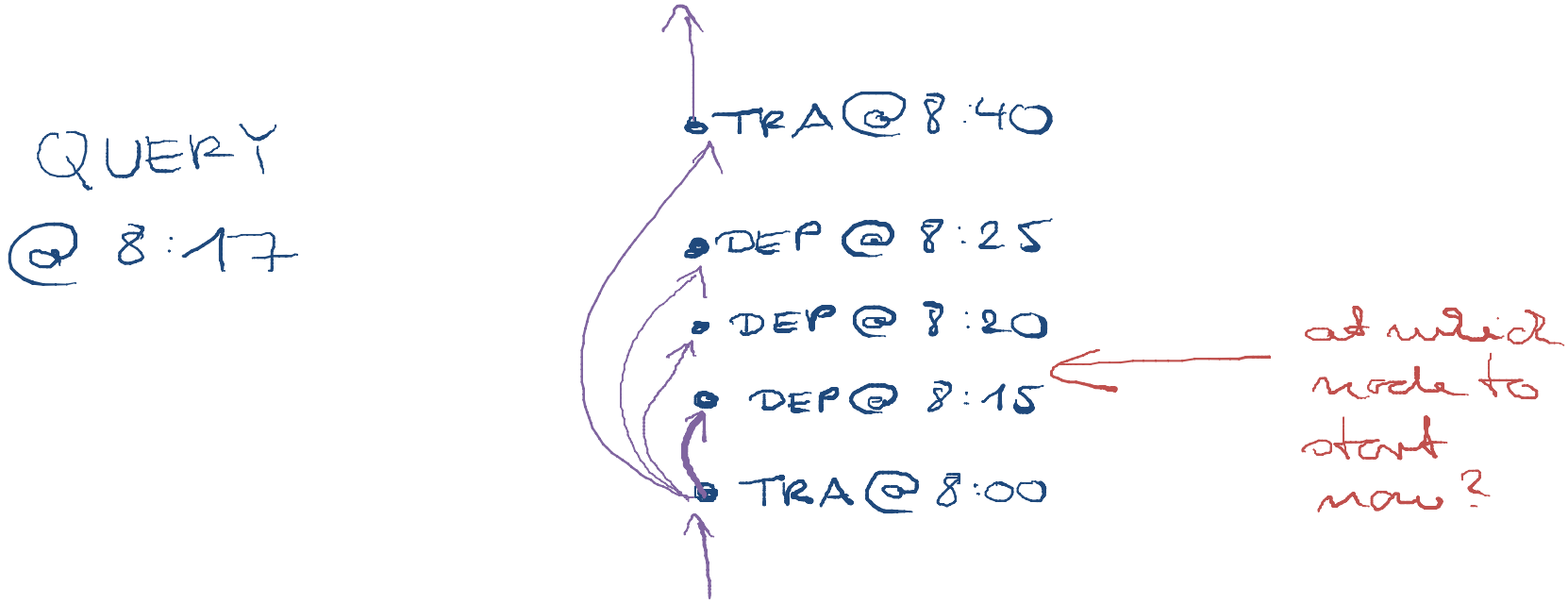
UNI FREIBURG

- Organizational
  - Your results from Ex. Sheet #9 (Transit Networks, GTFS)
  - This is the **third to last** lecture
  - Update on the exam date: **Monday, August 20 … ok?**
- Multi-criteria costs
  - How to model → Pareto sets
  - How to compute shortest paths → Multi-label Dijkstra
  - How do our algorithms so far perform on transit networks?
- Exercise sheet
  - Half theoretical this time
  - Coding part is just a single method this time

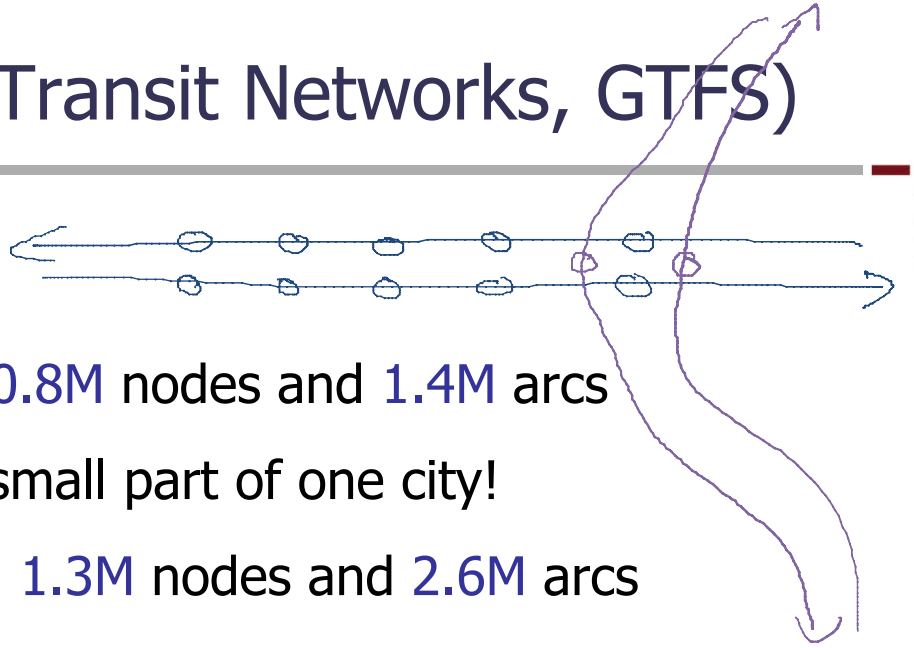
Feedback on ES#9 (Transit Networks, GTFS)

UNI FREIBURG

- Summary / excerpts last checked July 11, 14:24
  - Not always clear how pick node at source station
  - Manhattan dataset had connectivity problems
    - for a random query, prob. of existing SP was < 20%



Results for ES#9 (Transit Networks, GTFS)



UNI FREIBURG

- Summary
  - Manhattan graph has 0.8M nodes and 1.4M arcs
    - And that is only a small part of one city!
    - Compare to BaWü: 1.3M nodes and 2.6M arcs
  - Dijkstra query times around 10 – 100 milliseconds
    - Dijkstra query times for BaWü where 10 times larger
    - But that was only for queries with a connection at all
    - Unfortunately, bad connectivity in Manhattan data
    - No shortest path at all for over 80% of all queries
  - Reason: walking between stations crucial for this dataset

400644,,FT WASHINGTON AV - W 181 ST,,40.850636,-73.938484,,,0,
400631,,FT WASHINGTON AV - W 181 ST,,40.851204,-73.93808,,,0,

Multi-criteria cost functions 1/5

UNI FREIBURG

- So far our costs were always scalar numbers
  - ... namely the travel time
  - But there are many other criteria a user might want to optimize, too:
    - price (both road and transit networks)
    - beauty of the trip (both road and transit networks)
    - minimize walking between stations (transit only)
    - minimize number of transfers (transit only)
  - For the sake of explanation let us look at **two criteria** costs for the rest of the lecture: travel time and penalty

    (the penalty grows with more walking and more transfers)

Multi-criteria cost functions 2/5

UNI FREIBURG

- More than one solution
  - With two (or more) criteria, there is now the possibility of more than one optimal solution

    3 hours with 0 transfers is incomparable to

    2 hours with 1 transfer
  - However, some solutions are strictly better than others:

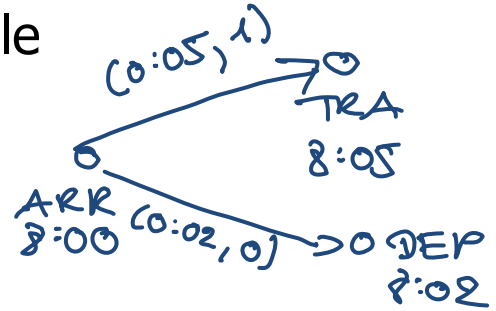
    2 hours with 1 transfer is better than

    3 hours with 2 transfers

Multi-criteria cost functions 3/5

- Formally
  - Costs are pairs (x, y) of scalars
  - We write $(x, y) \leq (x', y')$ if and only if $x \leq x'$ and $y \leq y'$
  - We write $(x, y) = (x', y')$ if and only if $x = x'$ and $y = y'$
  - We write $(x, y) < (x', y')$ iff $(x, y) \leq (x', y')$ and $(x, y) \neq (x', y')$
  - We say that (x, y) and (x', y') are incomparable

    if neither $(x, y) \leq (x', y')$ nor $(x', y') \leq (x, y)$



- Example for such cost pairs
  - If the second component is simply #transfers, an arc from an arrival node at time 8:00 to a transfer node at time 8:05 would have cost (0:05, 1), and all other arcs would have costs (…, 0)

Multi-criteria cost functions 4/5

C
(3,0)
C'
(3,5)
(3,0) (2,1)
(2,1) (4,0)

NOTE:
|C'| can be more than 2

- Lemma 1
  - For each set of costs C there exists a subset C' of C such that
    - for each $c_1, c_2 \in C'$ with $c_1 \neq c_2$, c_1 is incomparable to c_2
    - for each $c \in C$, there exists a $c' \in C'$ with $c' \leq c$
  - Proof: start with C'=C, then... as long as C' contains c_1, c_2 with $c_1 \leq c_2$, remove c_2
- For a given query
  - ... let C be the set of costs of **all** possible paths
  - Then we want to compute a subset C' like above, called the **set of optimal solutions** or the **Pareto set** of C
  - As usual, we discuss only how to obtain the costs, and it will be easy to see in the end how to get paths with these costs

Multi-criteria cost functions 5/5

- For a given C, is this subset C' unique?
  - Let C_1 and C_2 be two subsets of C according to Lemma 1

C_2 Pareto

$c_1 \in C_1 \Rightarrow c_1 \in C \Rightarrow \exists c_2 \in C_2 \quad c_2 \leq c_1$

$c_2 \in C \Rightarrow \exists c_1' \in C_1 \quad c_1' \leq c_2$

C_1 Pareto

$c_1' \leq c_2 \leq c_1 \Rightarrow c_1' \leq c_1$

C_1 Pareto $\Rightarrow c_1' = c_1 \Rightarrow c_2 = c_1 = c_1'$

$\Rightarrow c_1 \in C_2 \Rightarrow C_1 \subseteq C_2$ and $C_2 \subseteq C_1$ analogously

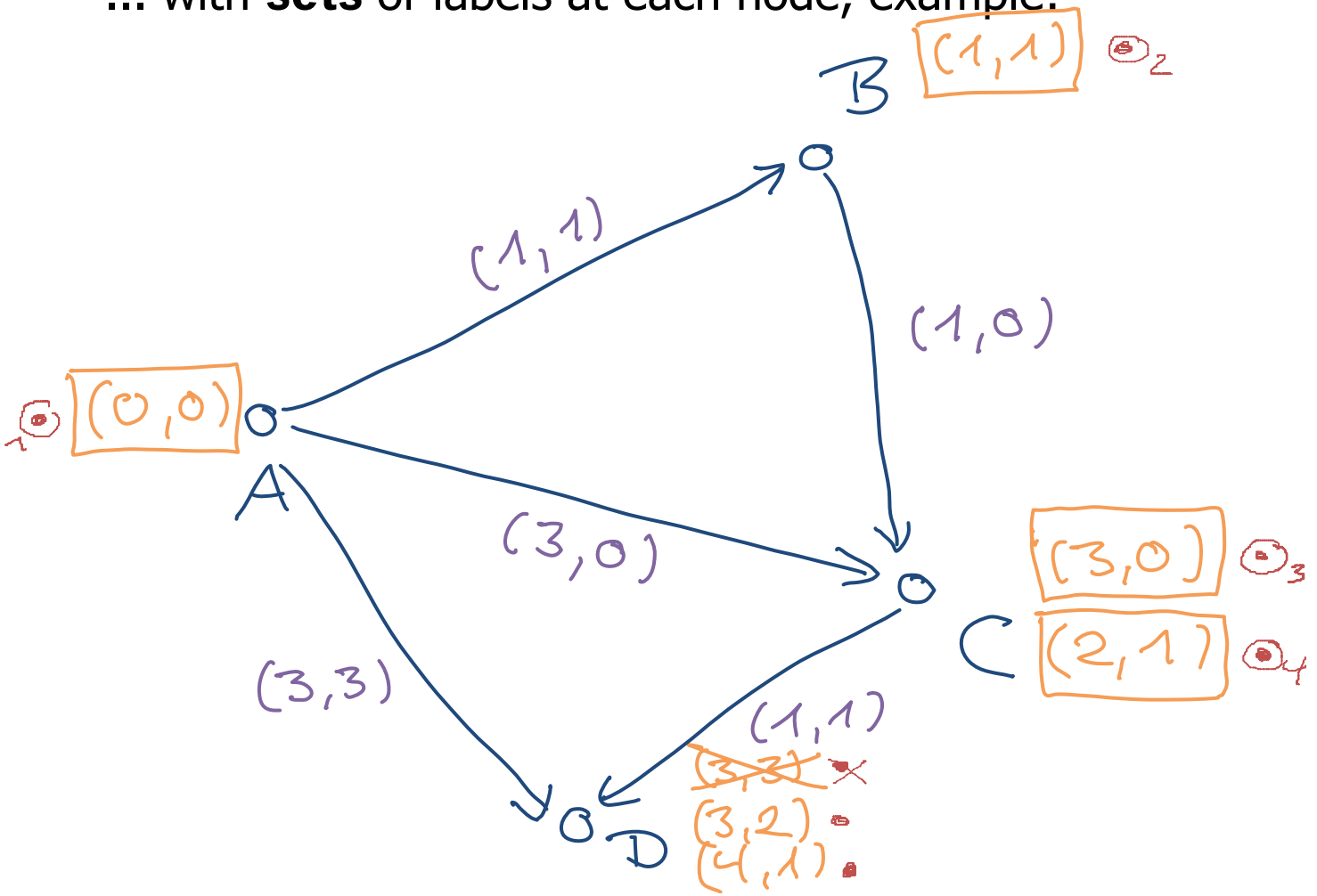
Multi-label Dijkstra 1/5

UNI FREIBURG

- How to compute these sets of solutions
  - Again, a variant of Dijkstra's algorithm does it
  - Consider ordinary Dijkstra, and think of the tentative costs at the nodes as **labels** (contain a single scalar, namely the tentative cost)
  - Initially there is only one label at the source, holding 0
  - All (not yet settled) labels are in a priority queue, according to some order on the set of possible labels
  - When processing the smallest label from the PQ, we settle it, and relax the outgoing arcs of the node to which the label belongs, creating new labels at the adjacent nodes
  - At ech node keep only the best label

Multi-label Dijkstra 2/5

- We can do the exact same thing
  - **...** with **sets** of labels at each node, example:



Multi-label Dijkstra 3/5

UNI FREIBURG

- In which order should we process the labels?
  - Let us denote that order by $<_{PQ}$
    - because it's the order in which we pick nodes from the PQ
  - Then $<_{PQ}$ must be a **refinement** of $<$ that is:

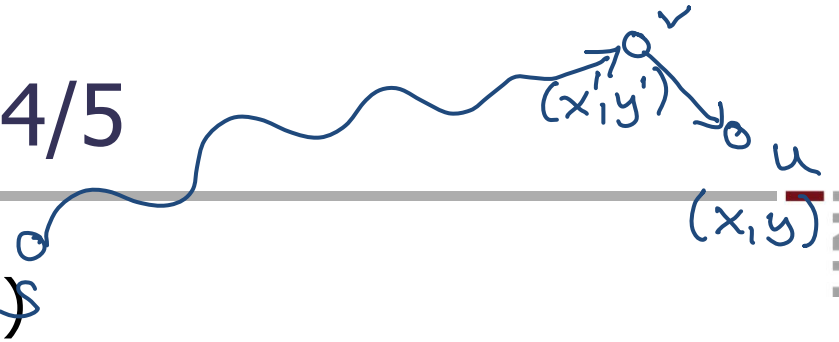
    $(x, y) < (x', y') \Rightarrow (x, y) <_{PQ} (x', y')$
  - Why does that work + why is it required? ... see next slide
  - Examples for $<_{PQ}$ for two-criteria costs:

    $(x, y) <_{PQ} (x', y')$ **iff** $x < x'$ or $(x = x'$ and $y < y')$ [time first]

    $(x, y) <_{PQ} (x', y')$ **iff** $y < y'$ or $(y = y'$ and $x < x')$ [penalty first]

    $(x, y) <_{PQ} (x', y')$ **iff** $x + y < x' + y'$ [sum]

Multi-label Dijkstra 4/5

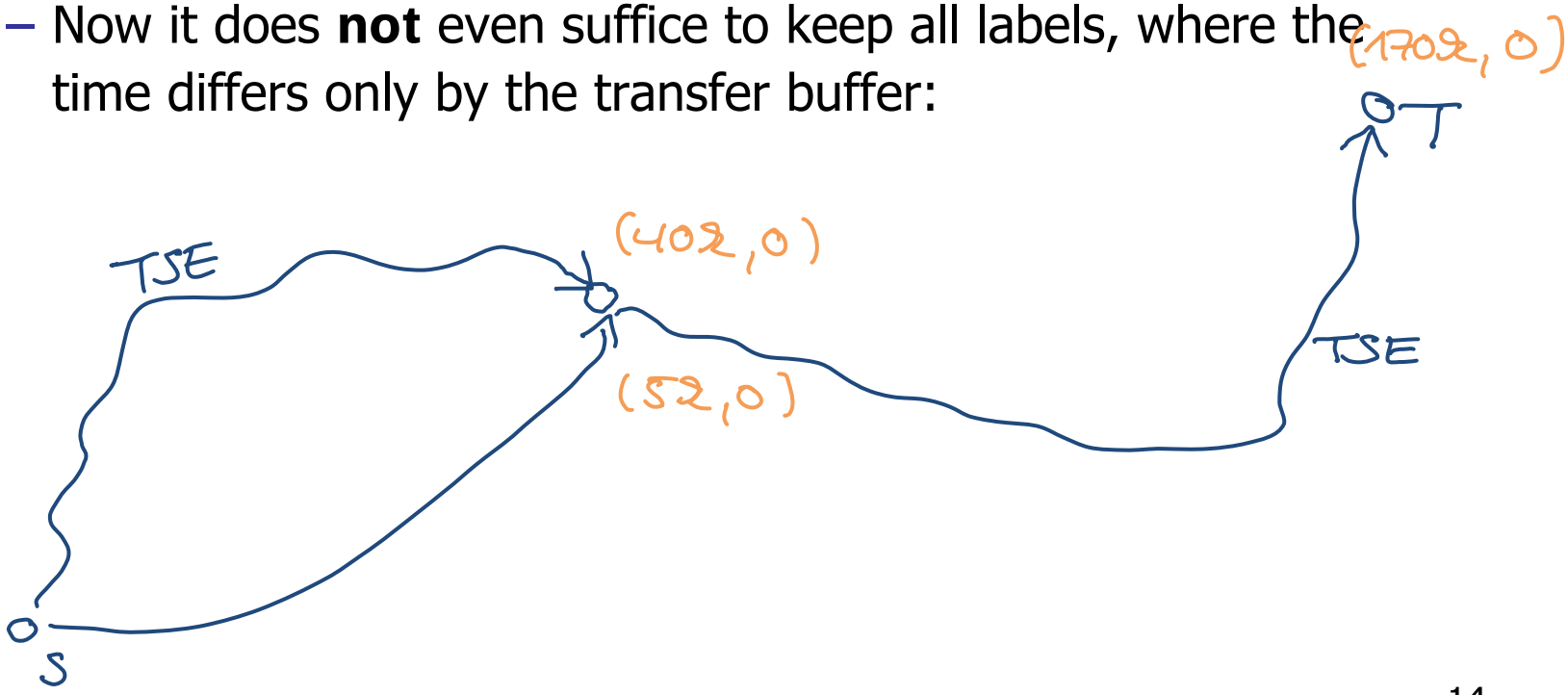


- Correctness proof (sketch)
  - For a given source node s, consider the union C of the sets of optimal costs from s at all nodes
  - As in our correctness proof for ordinary Dijkstra (Lecture 2), consider the **PQ** order on C and assume we have **no equals**:

    $(x_1, y_1) <_{PQ} (x_2, y_2) <_{PQ} (x_3, y_3) <_{PQ} \ldots$
  - Consider an arbitrary cost (x, y) from C at a node u, and let v be the predecessor of u of a shortest path to u with that cost
  - Let (x', y') be the cost of the path until v; note (x', y') < (x, y)
  - If the PQ order is a refinement of the label order, then (x', y') was processed earlier, and by way of induction everything was correct up to this point

Multi-label Dijkstra 5/5

- How about on a time-dependent graph?
  - Then we have a similar problem as with the transfer buffers
  - That is, labels computed along prefixes of shortest paths do not necessarily belong to shortest paths
  - Now it does **not** even suffice to keep all labels, where the time differs only by the transfer buffer:



Road vs. Transit Networks

UNI FREIBURG

- All our routing algorithms so far
  - ... in principle compute shortest paths on arbitrary graphs
  - However, we just verified their efficiency on road networks
- Transit networks in the time-expanded model
  - ... are also just graphs (with static arc costs)
  - How do our algorithms so far perform on these graphs
  - This will be discussed on the remaining slides
    - our discussion will be relatively high-level and informal

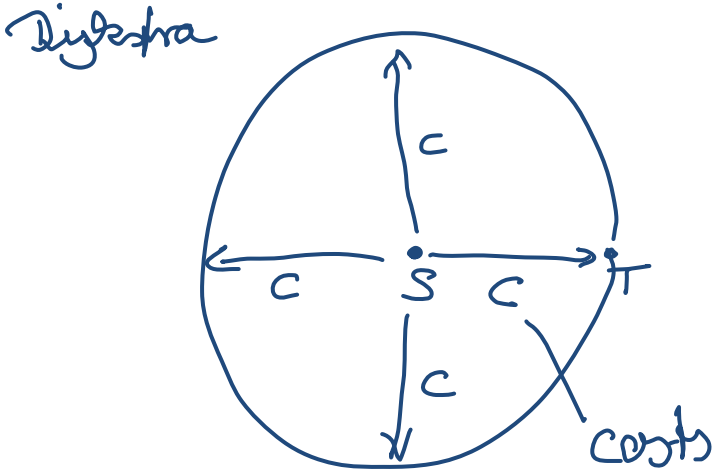
Dijkstra's algorithms

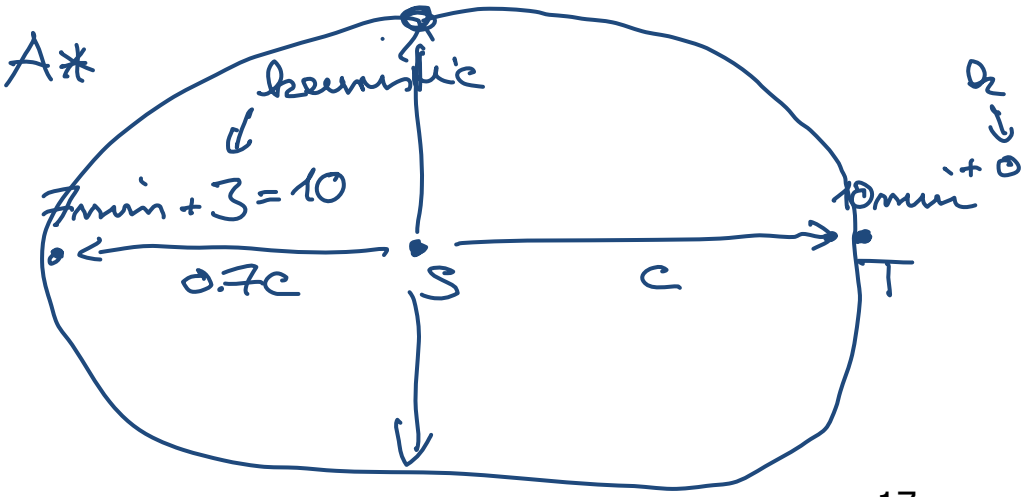
UNI FREIBURG

- Performance on time-expanded transit networks
  - One iteration takes ~ 1 µsec / node, as usual
  - Transit networks are bigger than road networds
    - Recall: Manhattan alone almost as large as BaWü
  - Like for road networks: when travel time from source to target is T, we settle everything within time radius T
  - When T is large or ∞ we search the whole graph ... this happens more often than in road networks, for example:
    - bad connectivity by bus / train
    - overnight connections

A* algorithm with straightline heuristic

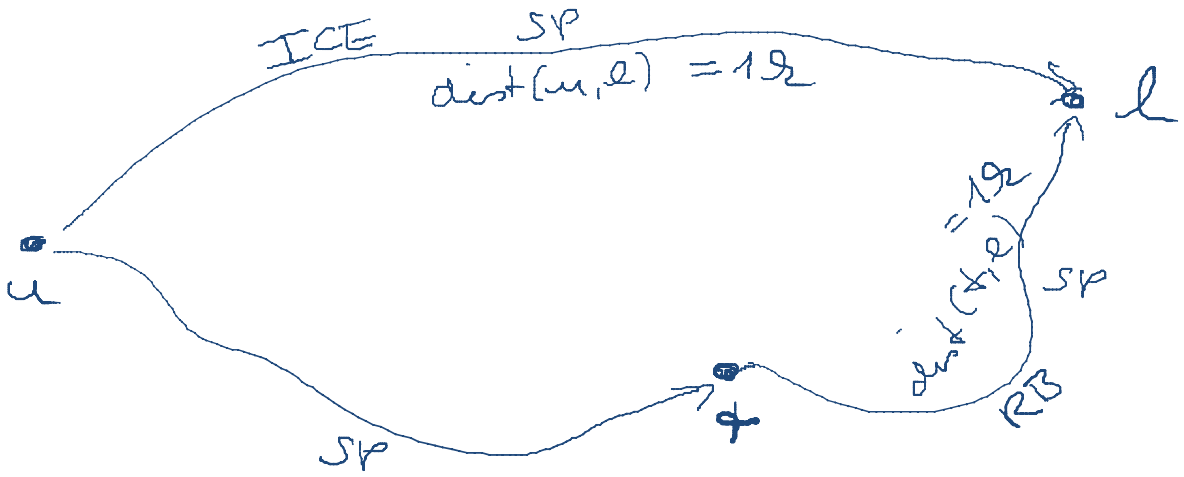
- Performance on time-expanded transit networks
  - Experiments show little improvement over Dijkstra ... why?
  - Consider Bus 10 from Bärenweg to Siegesdenkmal
  - Takes 10 minutes, straightline distance is ≈ 2.5 km
  - Let's say the maximum speed is 100 km/h (trains!)
  - That gives a lower bound of 1.5 minutes





A* algorithm with landmark heuristic

- Performance on time-expanded transit networks
  - Also here, little improvement over Dijkstra ... why?

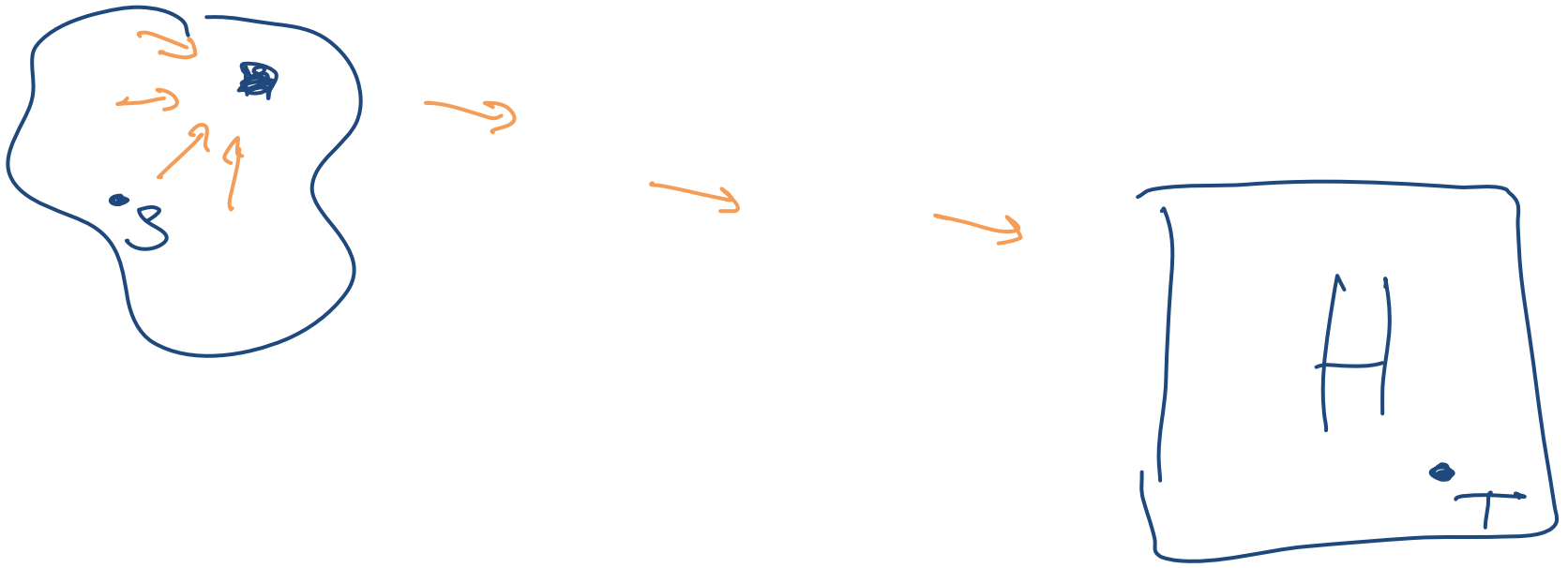


$$\pi(u) = dist(u,l) - dist(t,l)$$
$$= 0$$

PROBLEM: only weak relation between travel time and geometric distance (much weaker than in road networks)

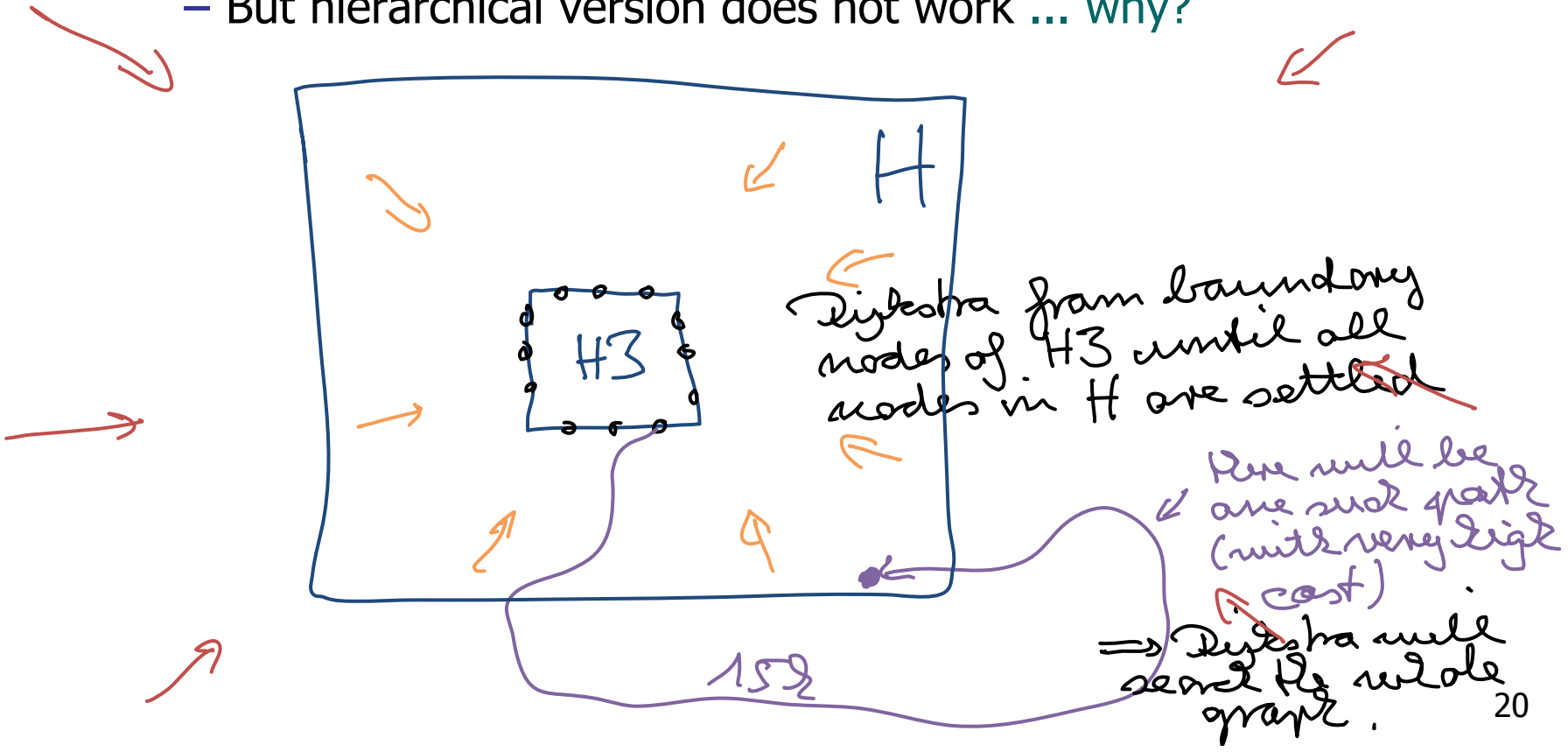
Arc flags 1/2

- Performance on time-expanded transit networks
  - Goal direction works well, good query times ... why?



Arc flags 2/2

- Performance on time-expanded transit networks
  - But precomputation cost (for the much larger transit graph is enormous) → calls for hierarchical version
  - But hierarchical version does not work ... why?



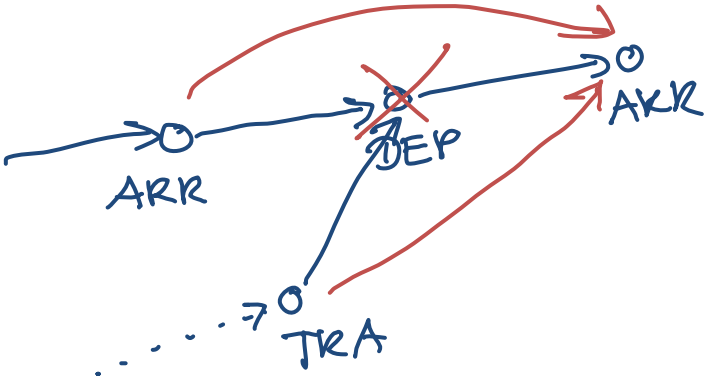
Transit-node routing

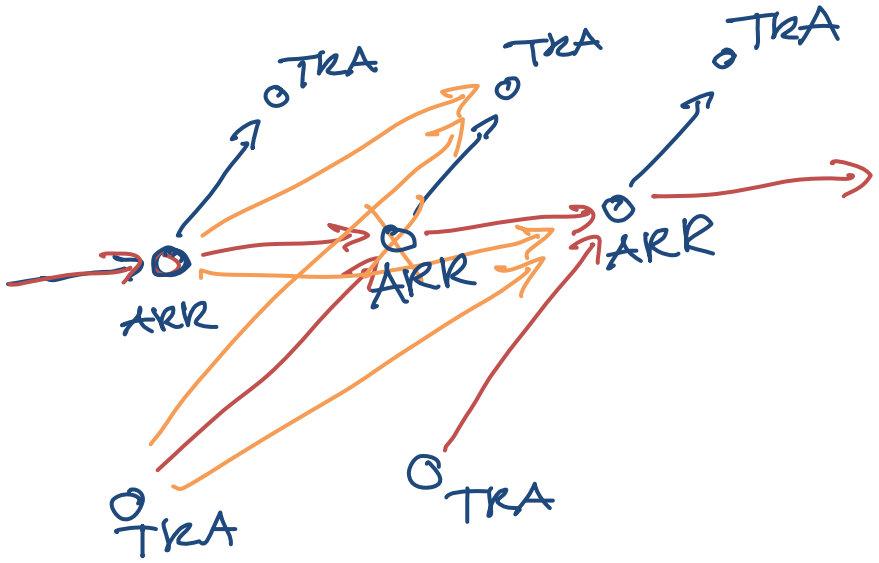
UNI FREIBURG

- Performance on time-expanded transit networks
  - We can also find small sets of transit / access nodes
  - But how do we compute them efficiently?
    - the geometric precomputation is inefficient for the same reasons as just explained for arc flags ... previous slide
    - the CH precomputation is also inefficient ... next slide
  - Also, local queries are not necessarily cheap in transit networks
  - Due to the "15 hours to the next village problem" from the previous slide

Contraction Hierarchies 1/5

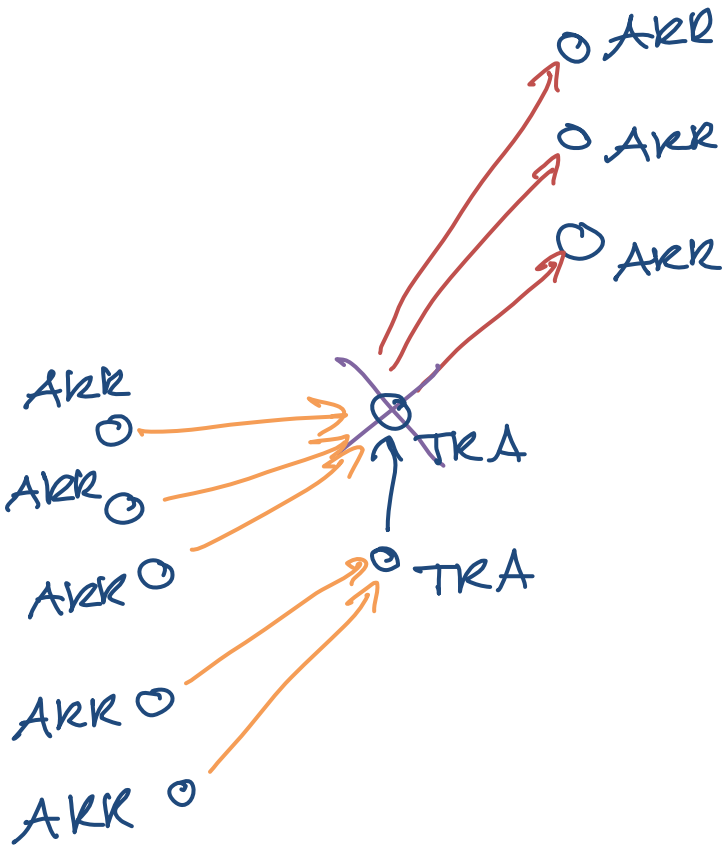
- Performance on time-expanded transit networks
  - Certain nodes can be contracted very well
  - We have already mentioned that the departure nodes can be contracted trivially, and this even saves us arcs
  - It seems like (at least some) arrival nodes can also be contracted without loss





Contraction Hierarchies 2/5

- Performance on time-expanded transit networks
  - But when we start to contract transfer nodes, the degree explodes:



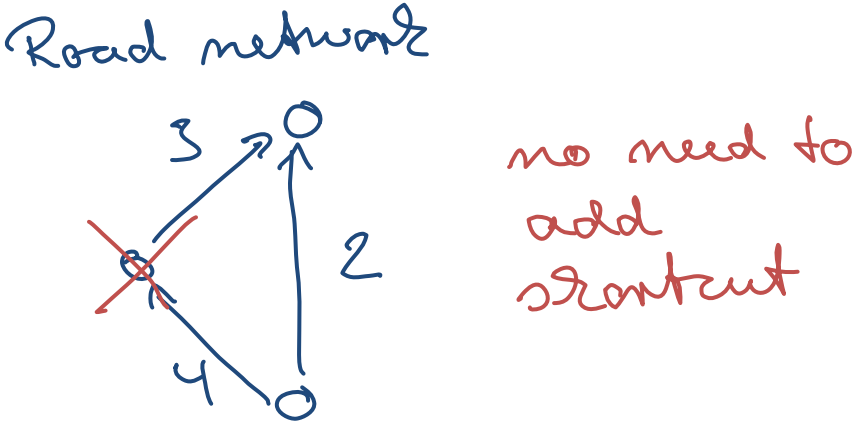
~~3×3 + 2×3 = 15~~

3×3 + 1×3 = 12

Contraction Hierarchies 3/5

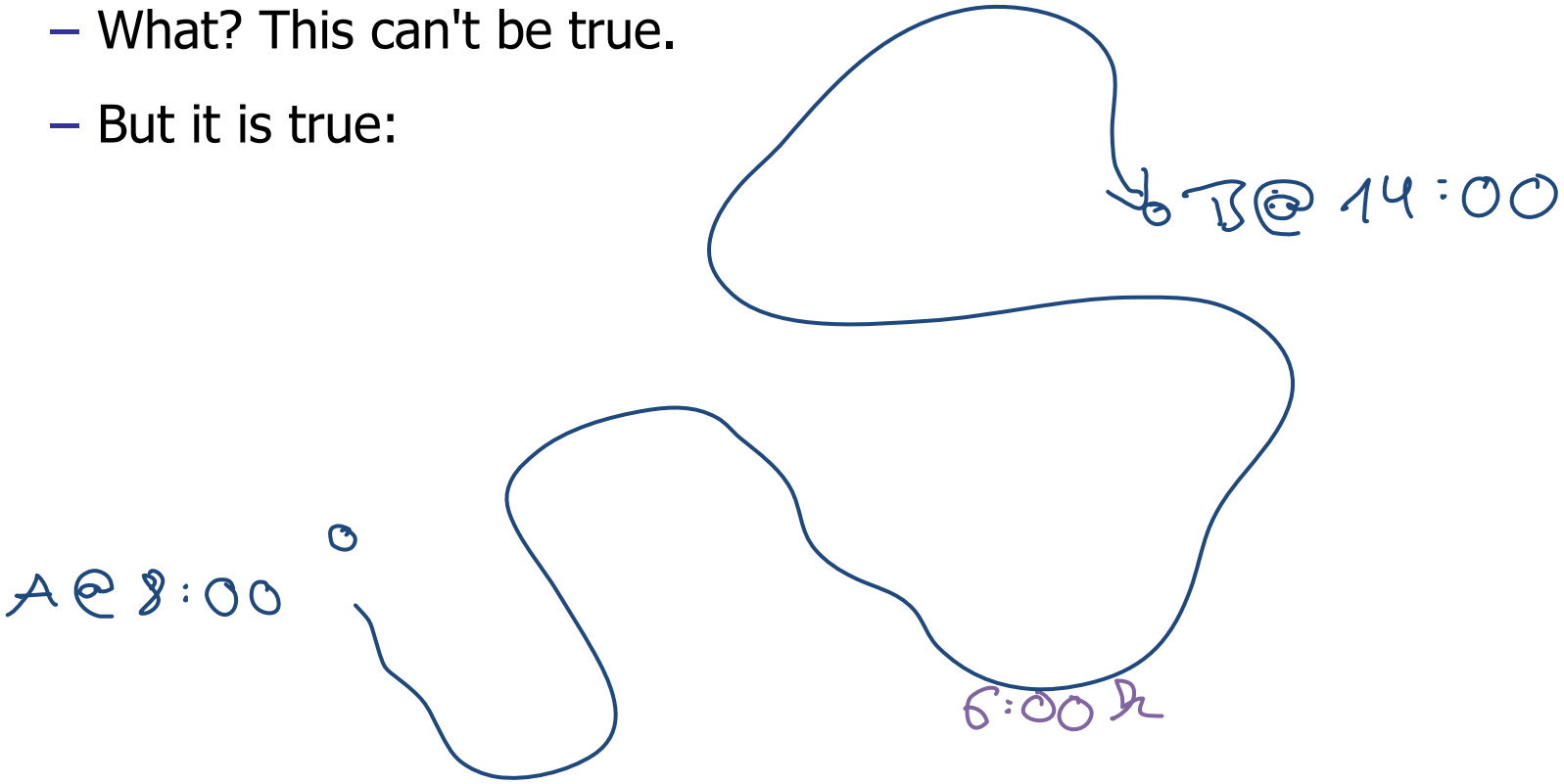
UNI FREIBURG

- Performance on time-expanded transit networks
  - Here is one surprising explanation why we need to add so many shortcuts in transit networks but not in road networks
  - Note that in transit networks (with cost = travel time), whenever we contract a node (with in and out degree > 0), we always need to add **all** the potentially necessary shortcuts
  - In road networks this is not the case, why this difference?



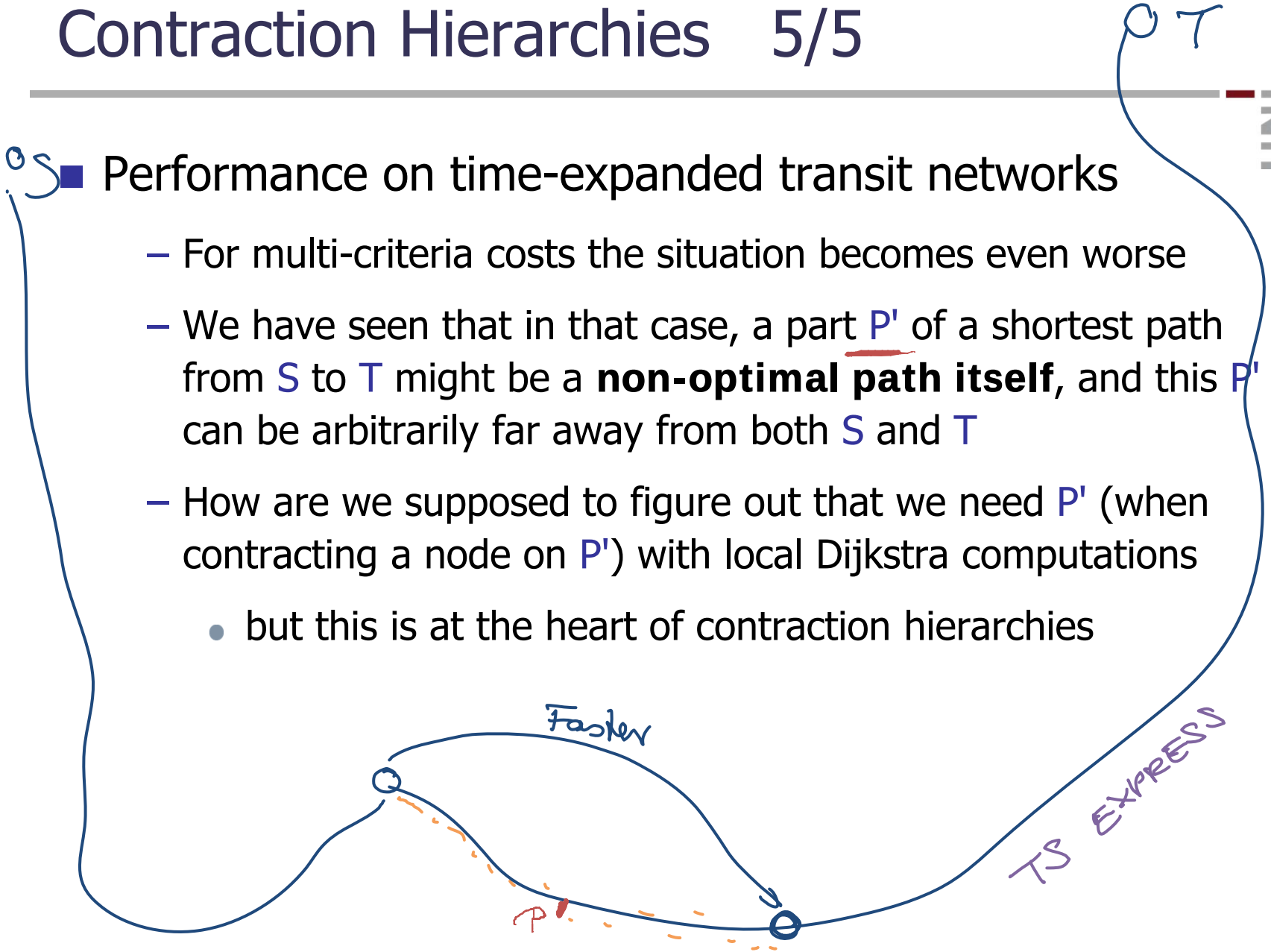
Contraction Hierarchies 4/5

- Performance on time-expanded transit networks
  - The reason is that (when cost = travel time), in a time-expanded transit network **every** path is a shortest path
  - What? This can't be true.
  - But it is true:



Contraction Hierarchies 5/5

- Performance on time-expanded transit networks
  - For multi-criteria costs the situation becomes even worse
  - We have seen that in that case, a part P' of a shortest path from S to T might be a **non-optimal path itself**, and this P' can be arbitrarily far away from both S and T
  - How are we supposed to figure out that we need P' (when contracting a node on P') with local Dijkstra computations
    - but this is at the heart of contraction hierarchies



Summary until here

UNI FREIBURG

- None of our algorithms so far
  - that is: Dijkstra, A*, arc flags, contraction hierarchies, and transit node routing
  - ... is practical for large transit networks
  - And matters seem to become hopeless when realistic features like transfer buffers and multi-criteria cost function come into play
  - And fully-realistic models pose even more challenges:

    service days, vehicle restrictions, finding a suitable source and target station, walking between stations, ...

References

UNI FREIBURG

- Road Networks vs. Transit Networks

  Car or Public Transport — Two Worlds

  Hannah Bast, Efficient Algorithms 2009, LNCS 5760

  http://www.springerlink.com/content/y46257m66372x730/

- Multi-label Dijkstra

  Optimal paths in graphs with [...] multidimensional weights

  Ronald Prescott Loui, CACM 26(9), 1983

  http://portal.acm.org/citation.cfm?doid=358172.358406