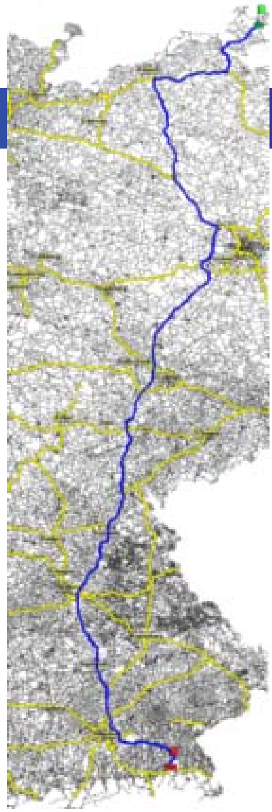
Efficient Computation of Bypass Areas

Jörg Roth
Computer Science Department
Nuremberg Institute of Technology
Germany

Motivation



- Route planning on road networks is a well understood problem.
- Computation of the *one* route that minimize costs from position to position.

Alternative requirements:

- the driver may not like the optimal route (e.g. personal knowledge),
- the driver wants to reach a stopover (e.g. an arbitrary fuel station)

… but does not want to exceed the costs too much

Motivation

Basic question:

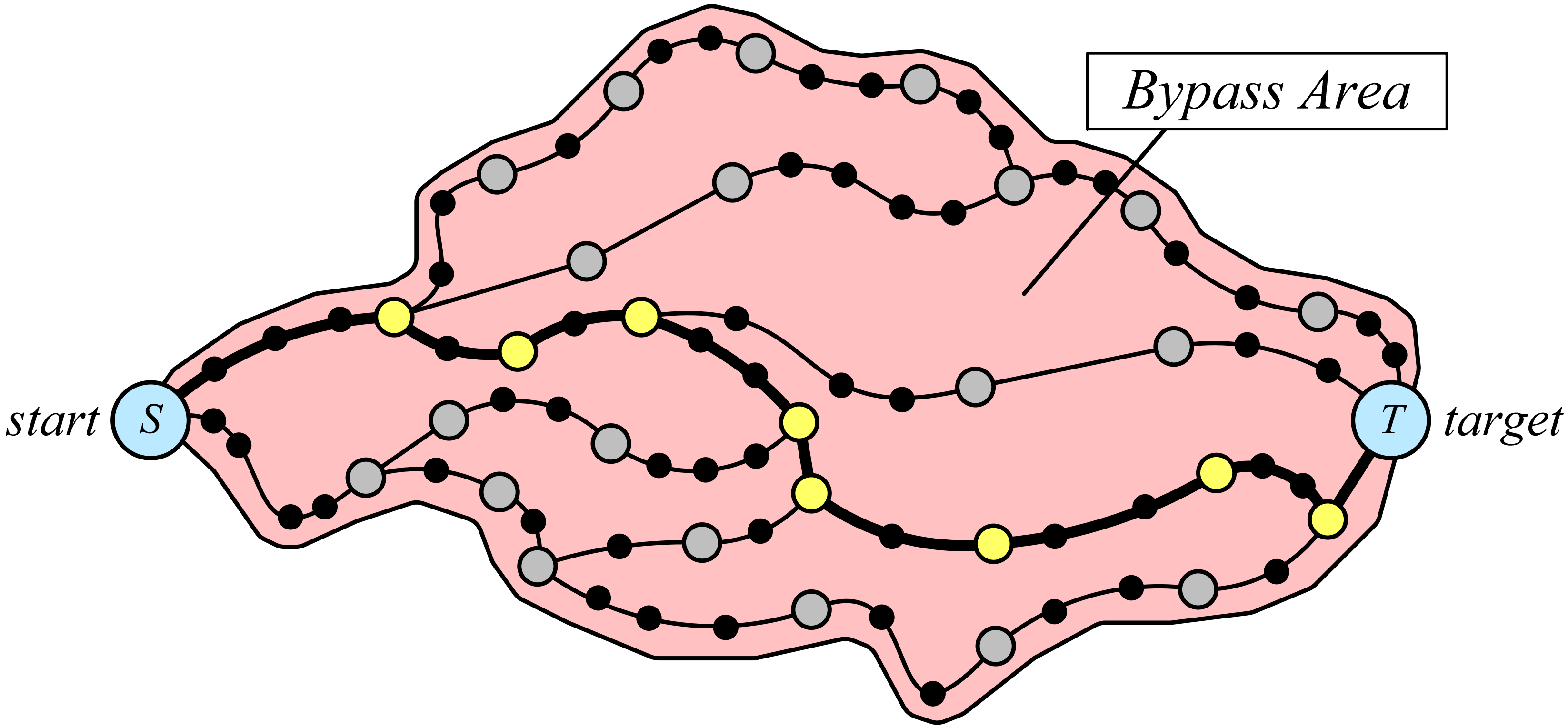
What are *all* bypasses that do not exceed the minimal costs from start to target by a given factor?

Useful for several services:

- I want to drive to *xy*, but have to refuel. My driving time should not increase more than 5%. Tell me *all* fuel stations.
- Driving to downtown, I want to pass a letter box.

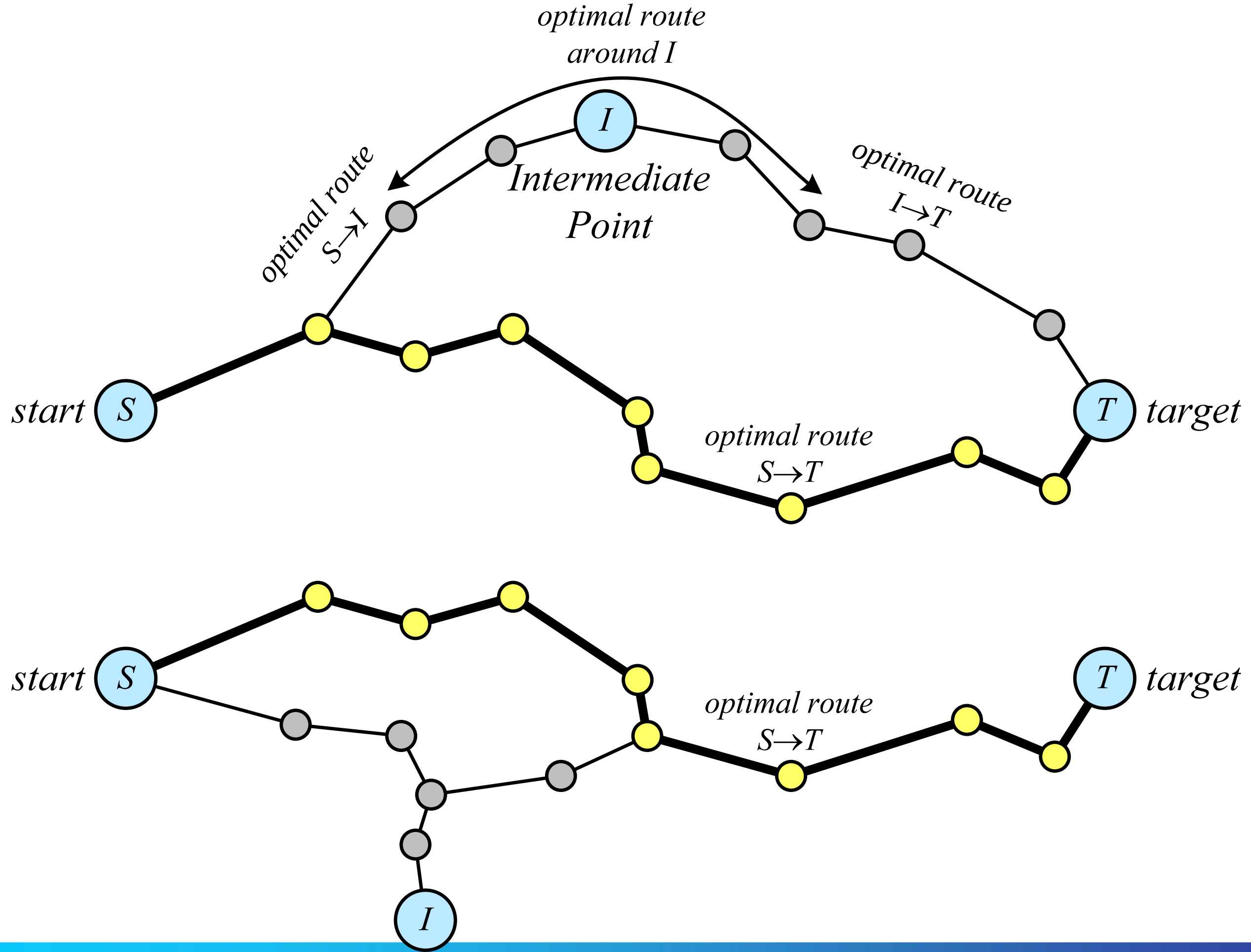
Bypasses

The Problem: we want to get *all* bypasses (and intermediate points) up to a certain cost limit



Two Types of Bypasses

Bypasses can by *locally optimal* or not

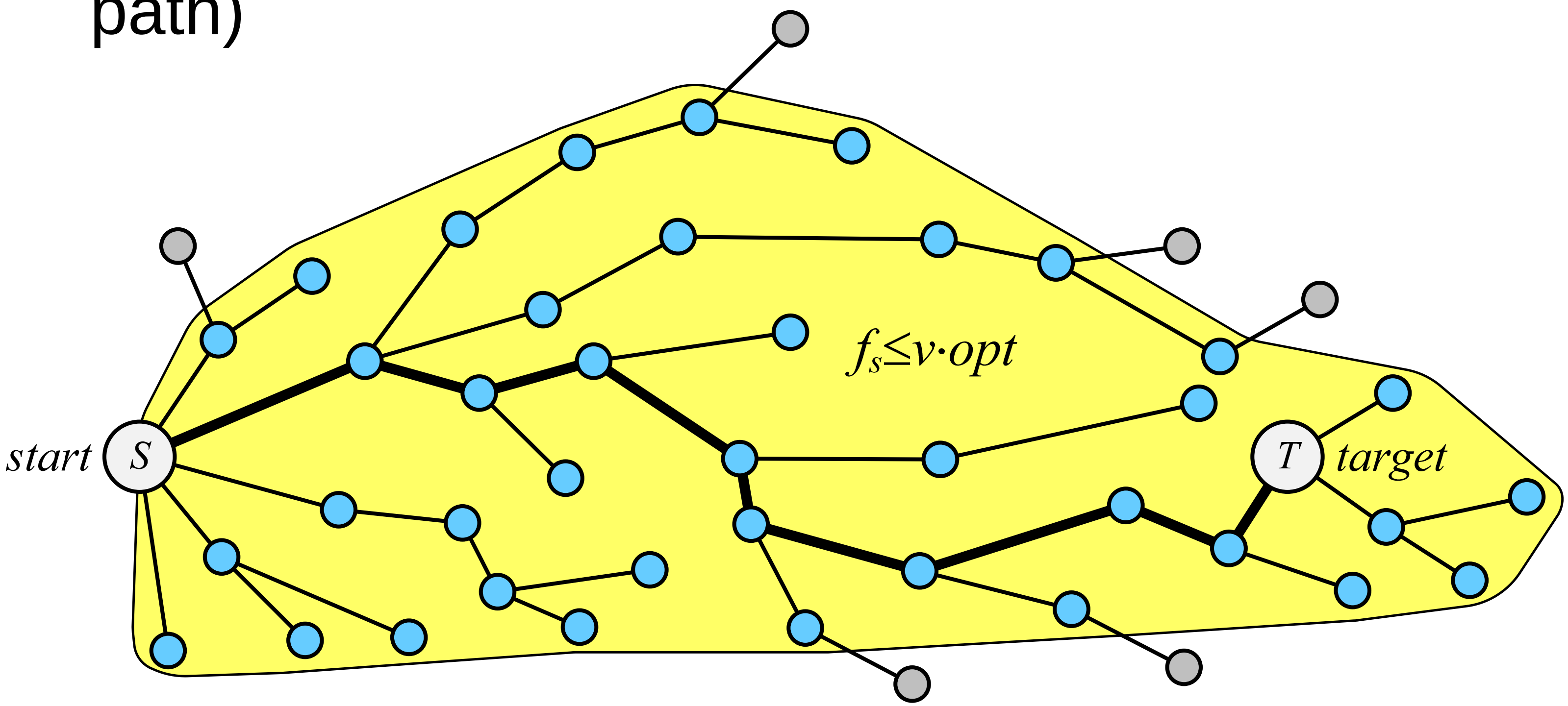


The Approach

- One approach for both types of bypasses – the caller decides:
  - Fuel station: not locally optimal
  - Bypassing a region, avoid a jam: locally optimal
- Problem: We cannot plan paths to *all* possible intermediate points
  - many millions candidates
  - for each: two route planning calls
- Our approach is based on *field generation*
  - computation time of bypass areas takes only approx. 2 times longer compared to optimal route

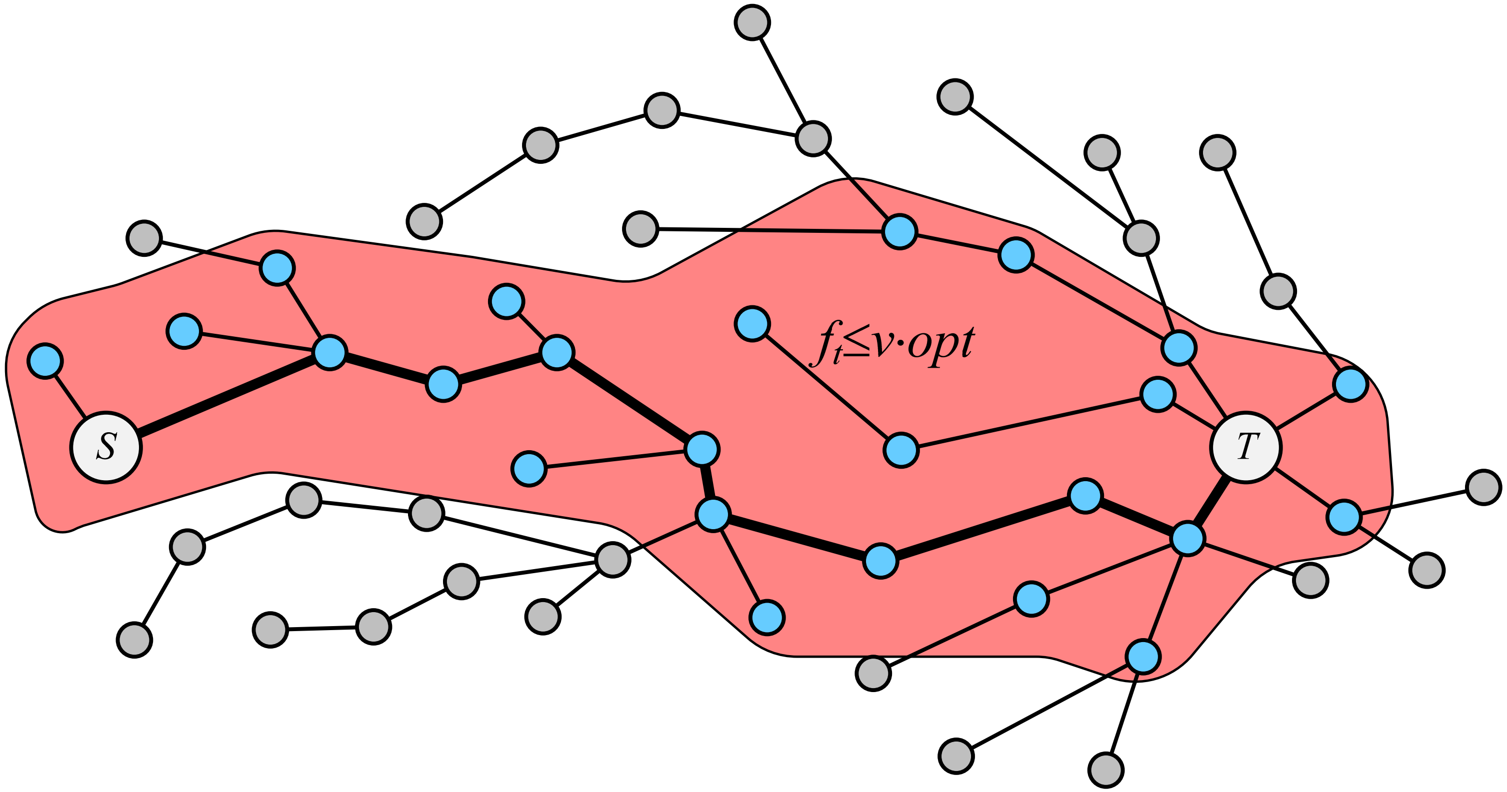
Start Field Generation

- A*: generates the field from S until T is reached
- Here: we generate the field from S until the cost limit is reached (on-the fly computes the optimal path)



Target Field Generation

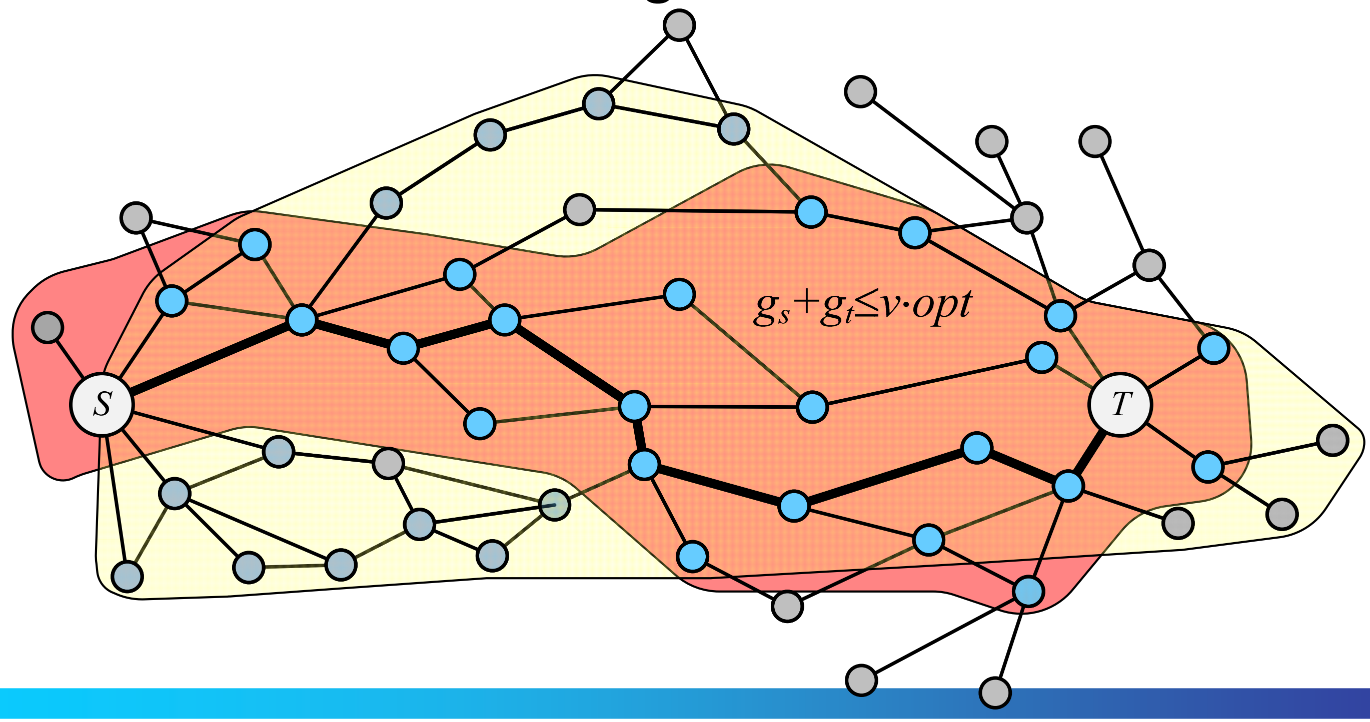
- The same going from T
- Benefit: now ideal A*-estimation available



Intersection of Both

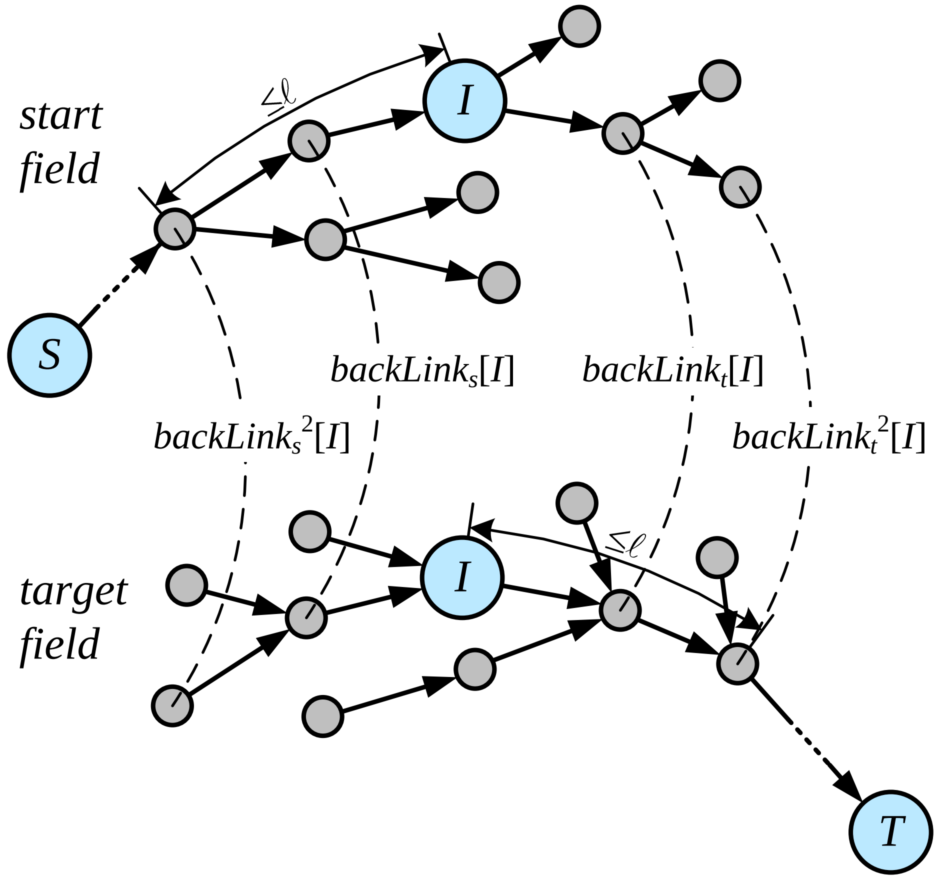
Intersecting both fields considering both costs:

- Costs $S \rightarrow any$ and to $any \rightarrow T$ now exactly known
- Easy to get those crossings where the cost sum does not exceed the given limit

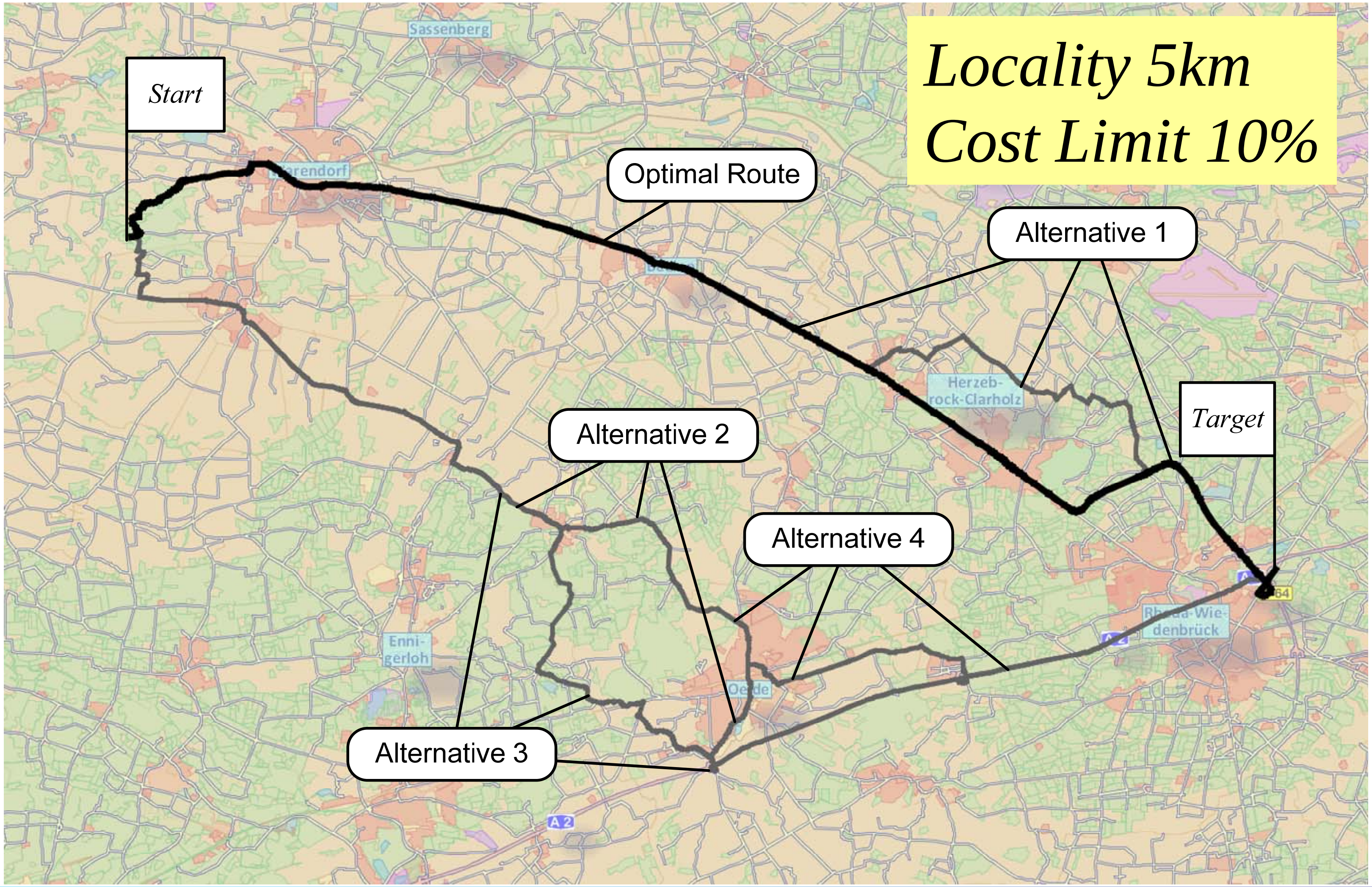


Local Optimality

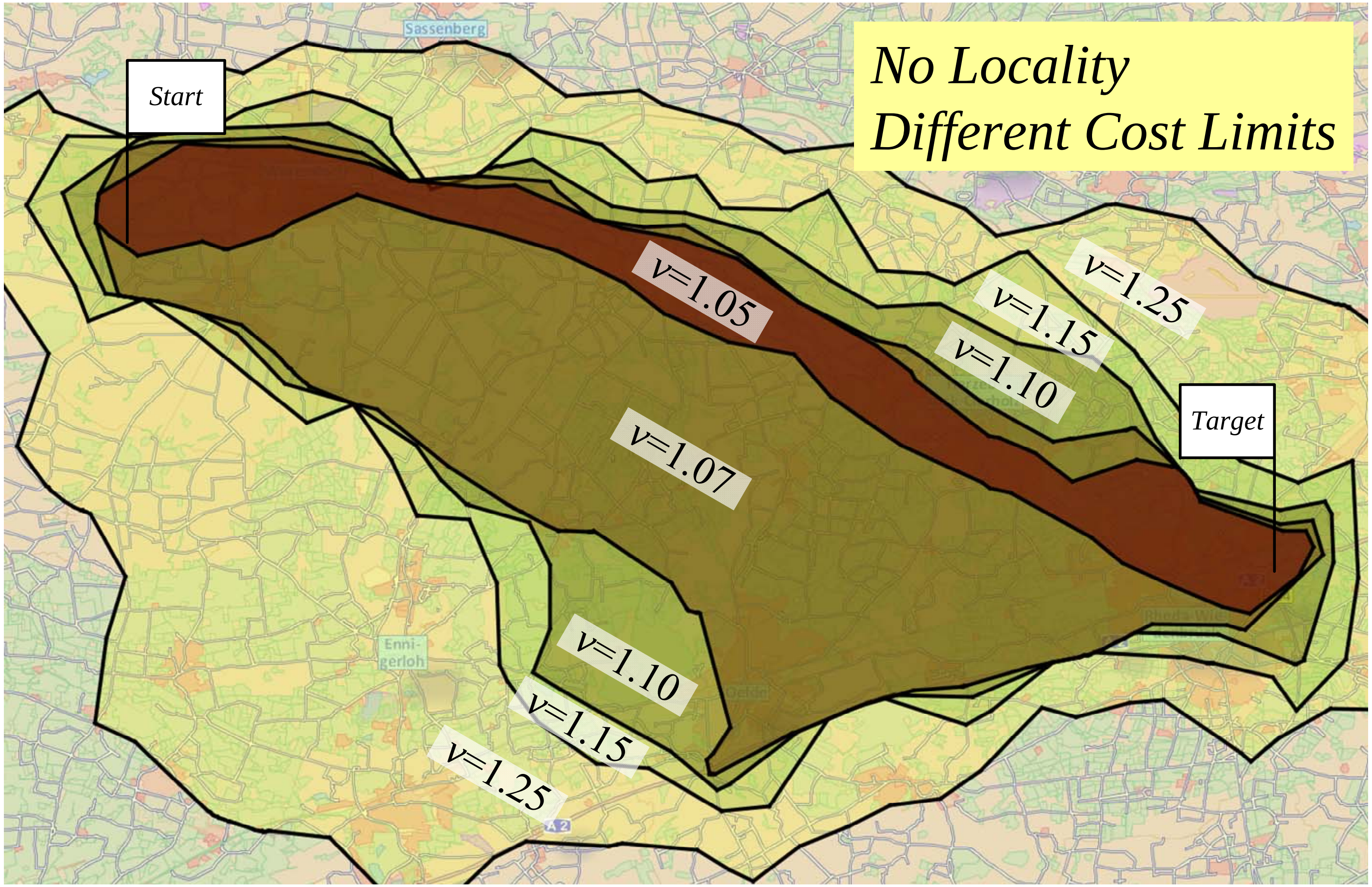
- If desired, remove those crossings that do not produce locally optimal routes
- Fortunately: a quick test is available that runs in *constant* time for each crossing
- only approx. 3% of total execution time



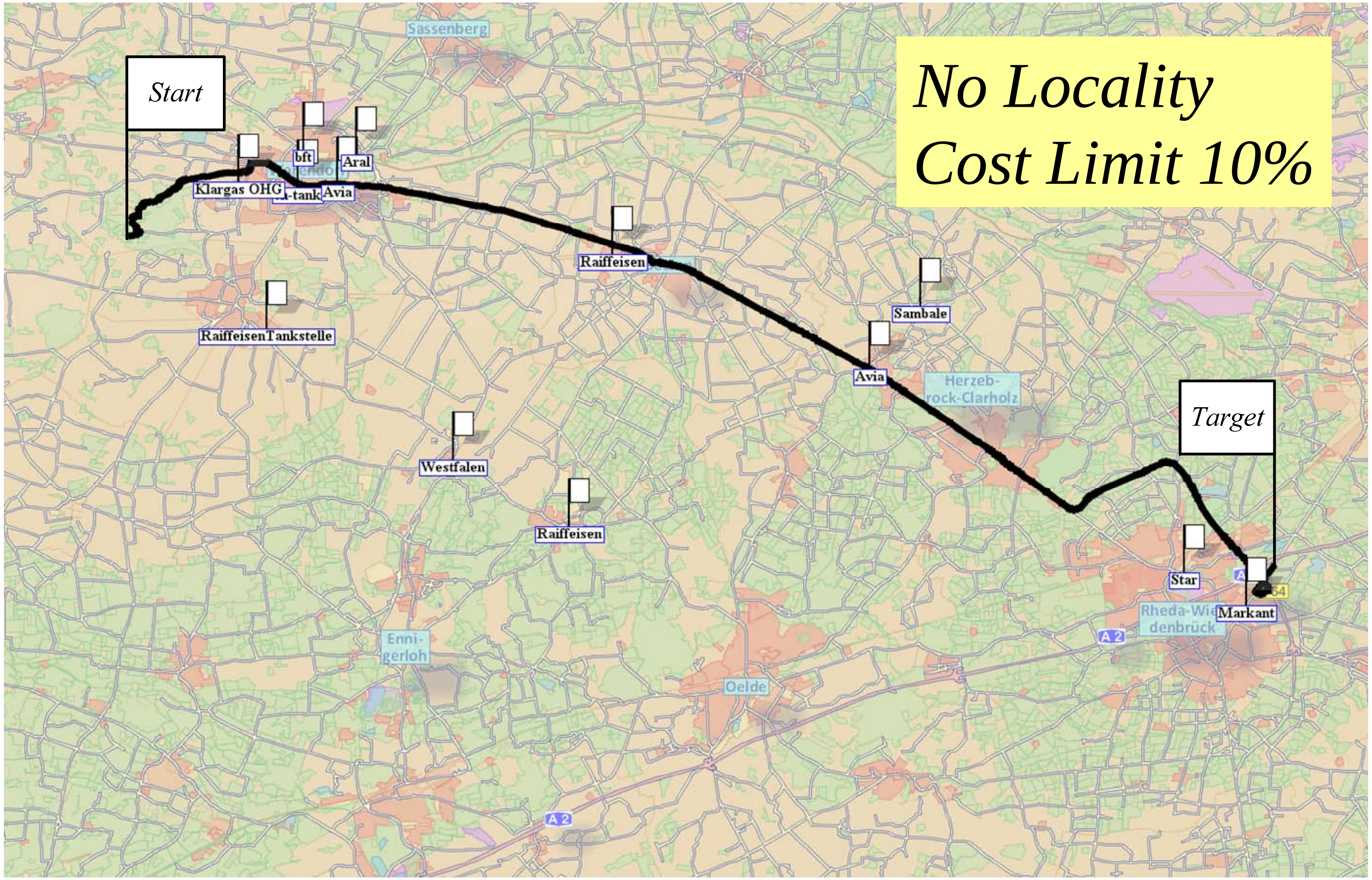
Example: All Locally Optimal Alternatives



Example: Bypass Areas (Concave Hulls)

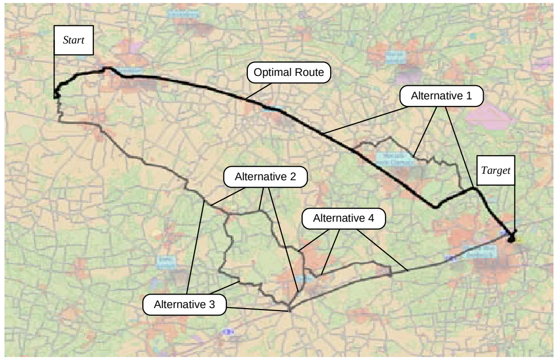


Example: All Fuel Stations in Cost Limit



Conclusions

- Computation of bypasses and bypass areas is a useful function for several services
- We presented an approach that computes this function in acceptable runtime
  - Re-using A* structures
  - Not iterating through all potential intermediate crossings
  - Efficient! ($\approx 2 \cdot \mathit{Path\text{-}Planning}$)



- Approach works for locally optimal or not locally optimal bypasses – depending on the usage scenario and application

Contact

Jörg Roth

Nuremberg Institute of Technology

Joerg.Roth@th-nuernberg.de

http://www.wireless-earth.org

HomeRun

Carbot

Zonezz