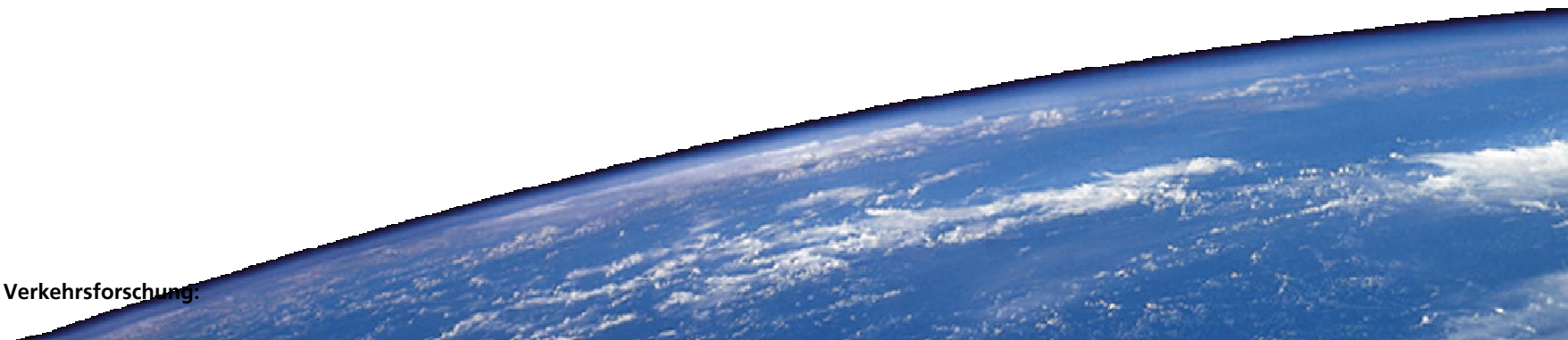
Preparation of Digital Maps for Traffic Simulation; Part1: Approach and Algorithms

Daniel Krajzewicz, Georg Hertkorn, Julia Ringel, Peter Wagner
Institute of Transportation at the German Aerospace Centre

Presented at the ISC 2005 in Berlin



Problem Description

What we do: microscopic simulation of large road networks.

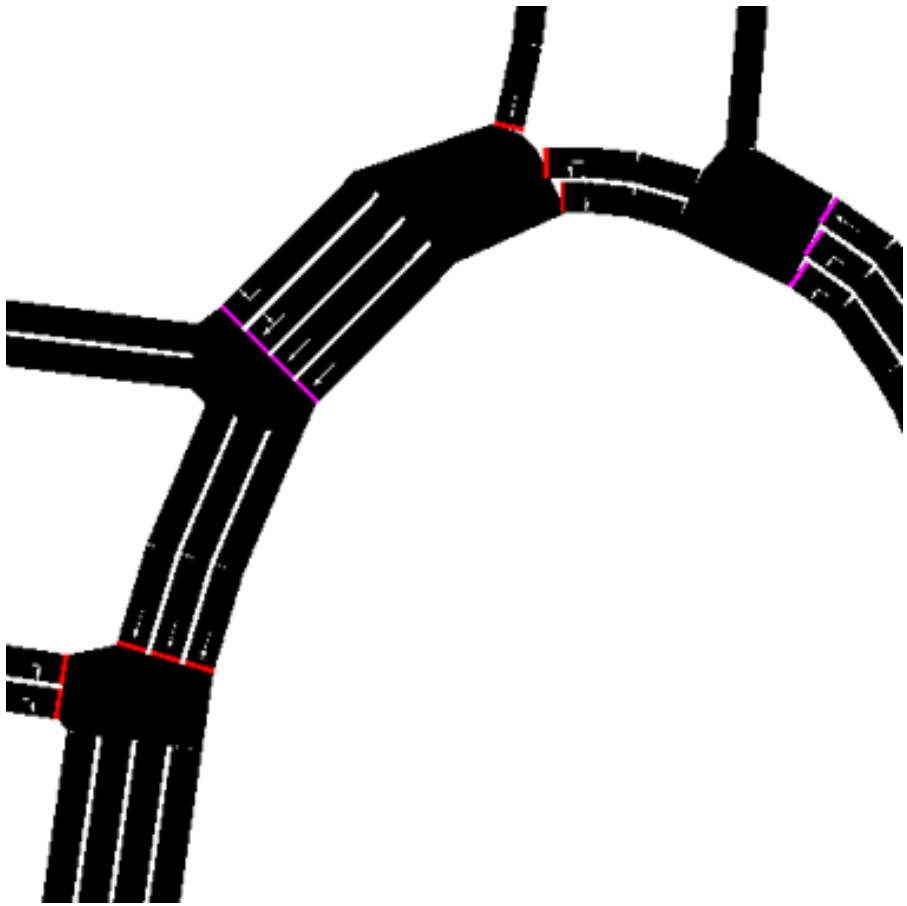
Mainly needed for this:
a fast simulation (SUMO) and road networks

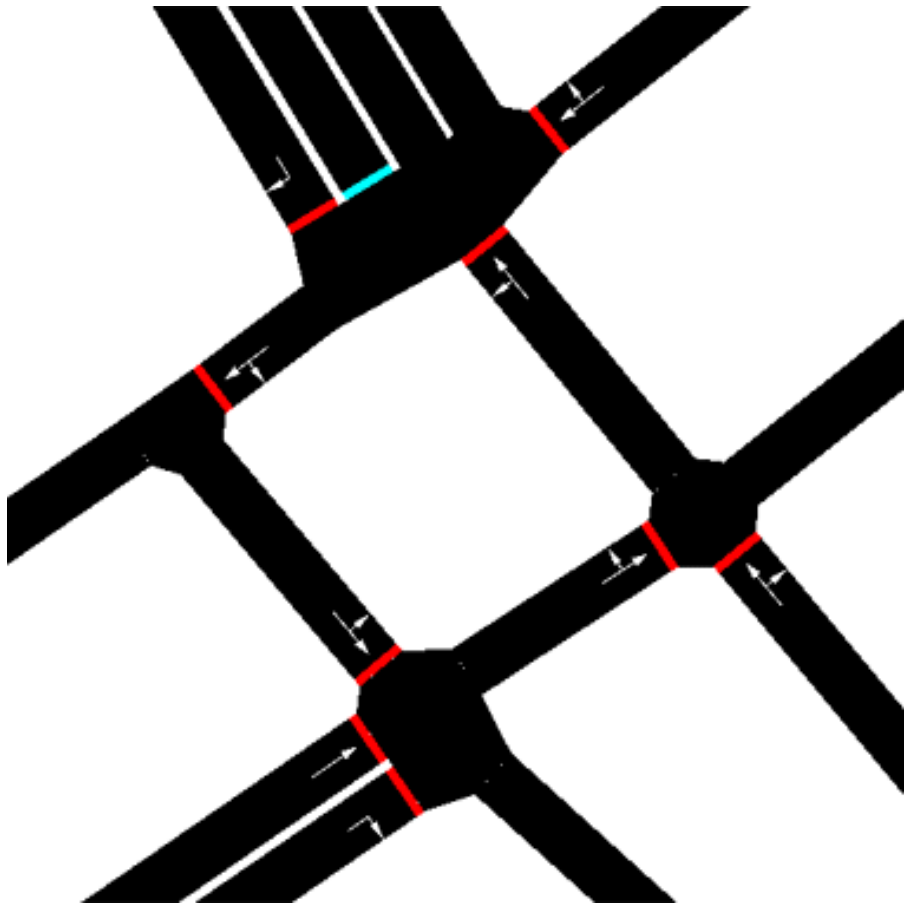
For microscopic modelling, following information is necessary for every junction:

- **The lane-to-lane connections**
  Which lanes may be reached from which lane?
- **The right-of-way rules**
  Which flow has to wait for another flow?

(Not regarded herein: junction's geometry)

Problem Description



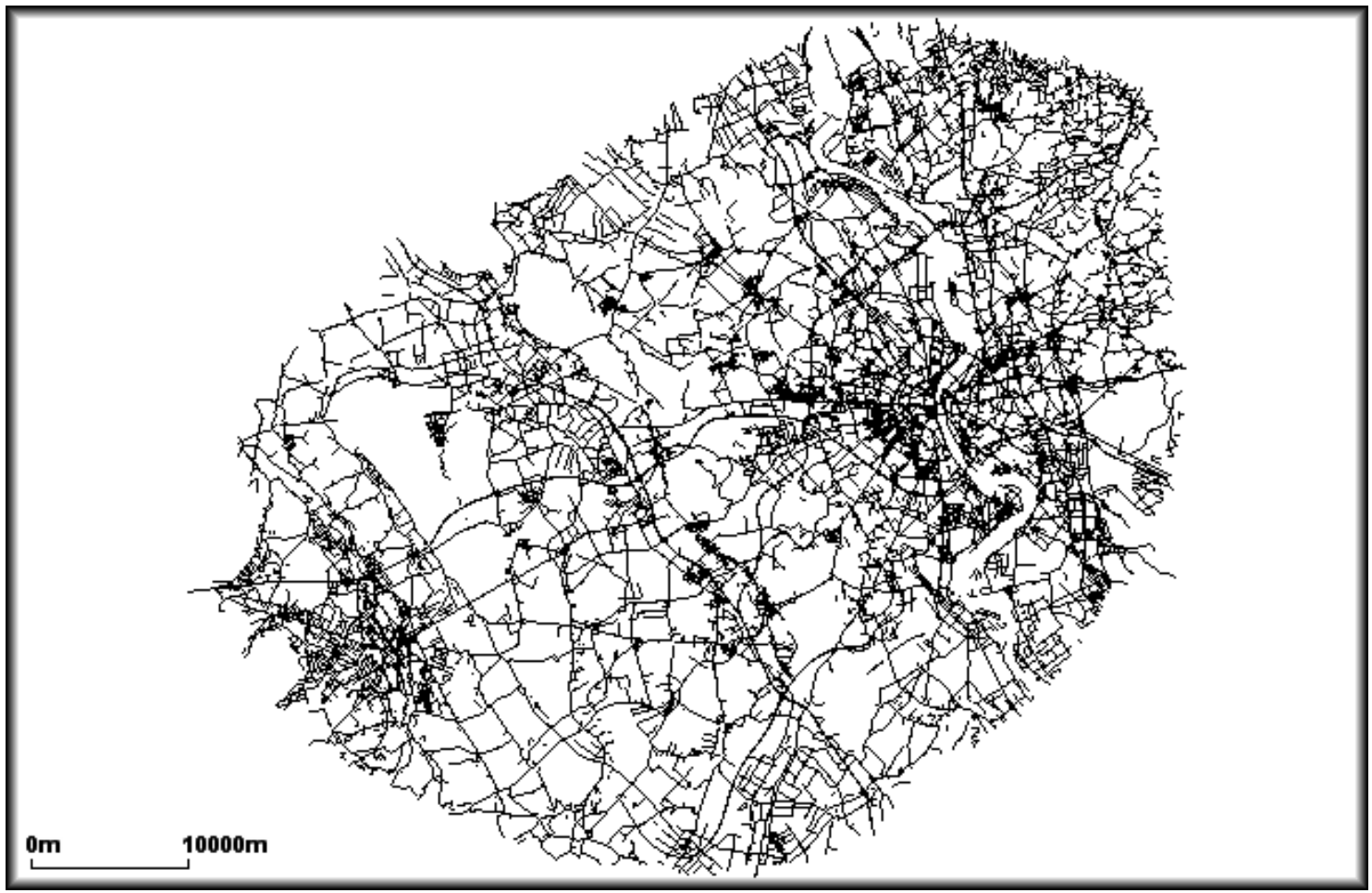


Problem Description

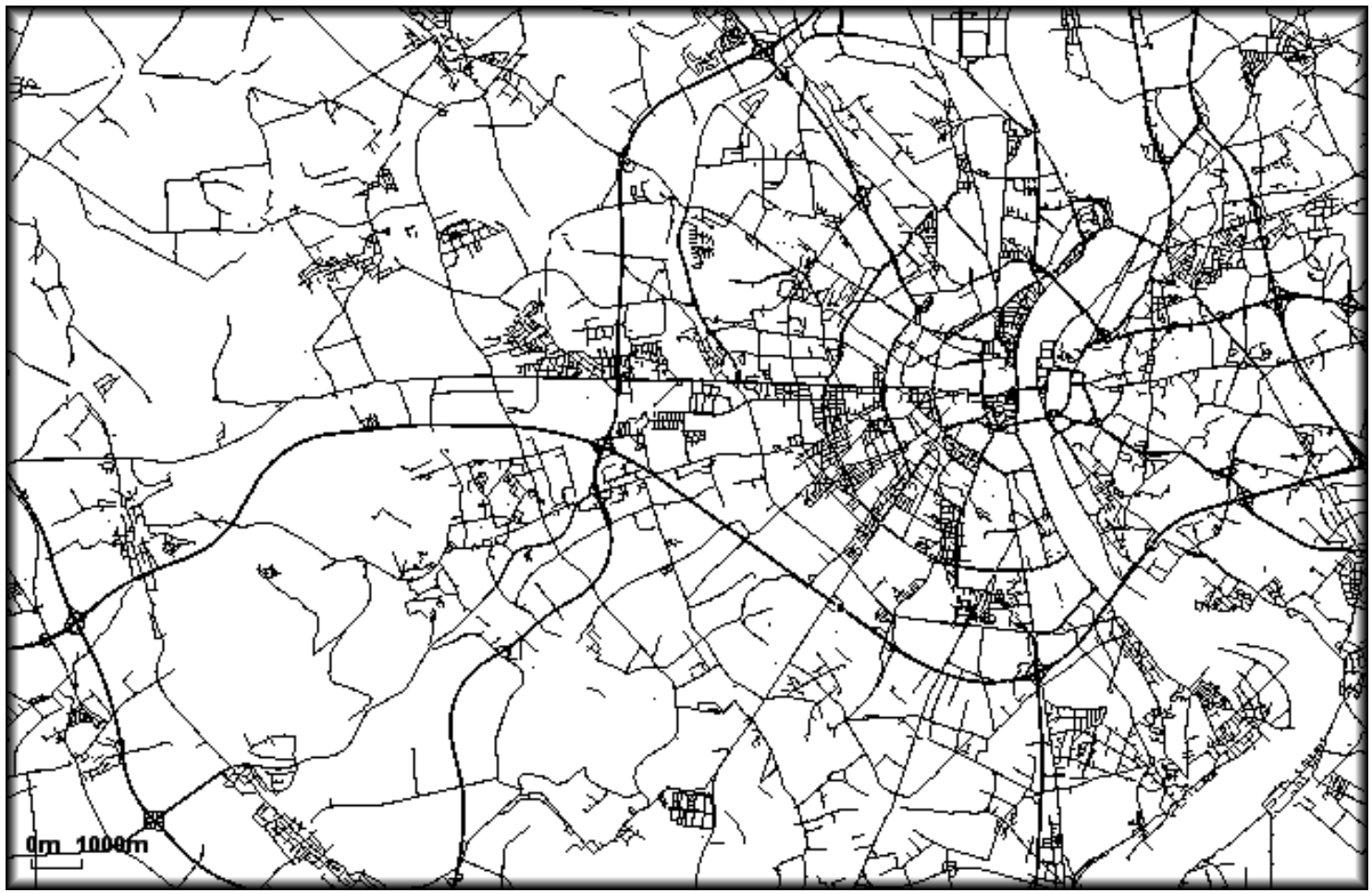
Mostly used approach: Edit the network by hand

...but: we deal with REALLY LARGE networks

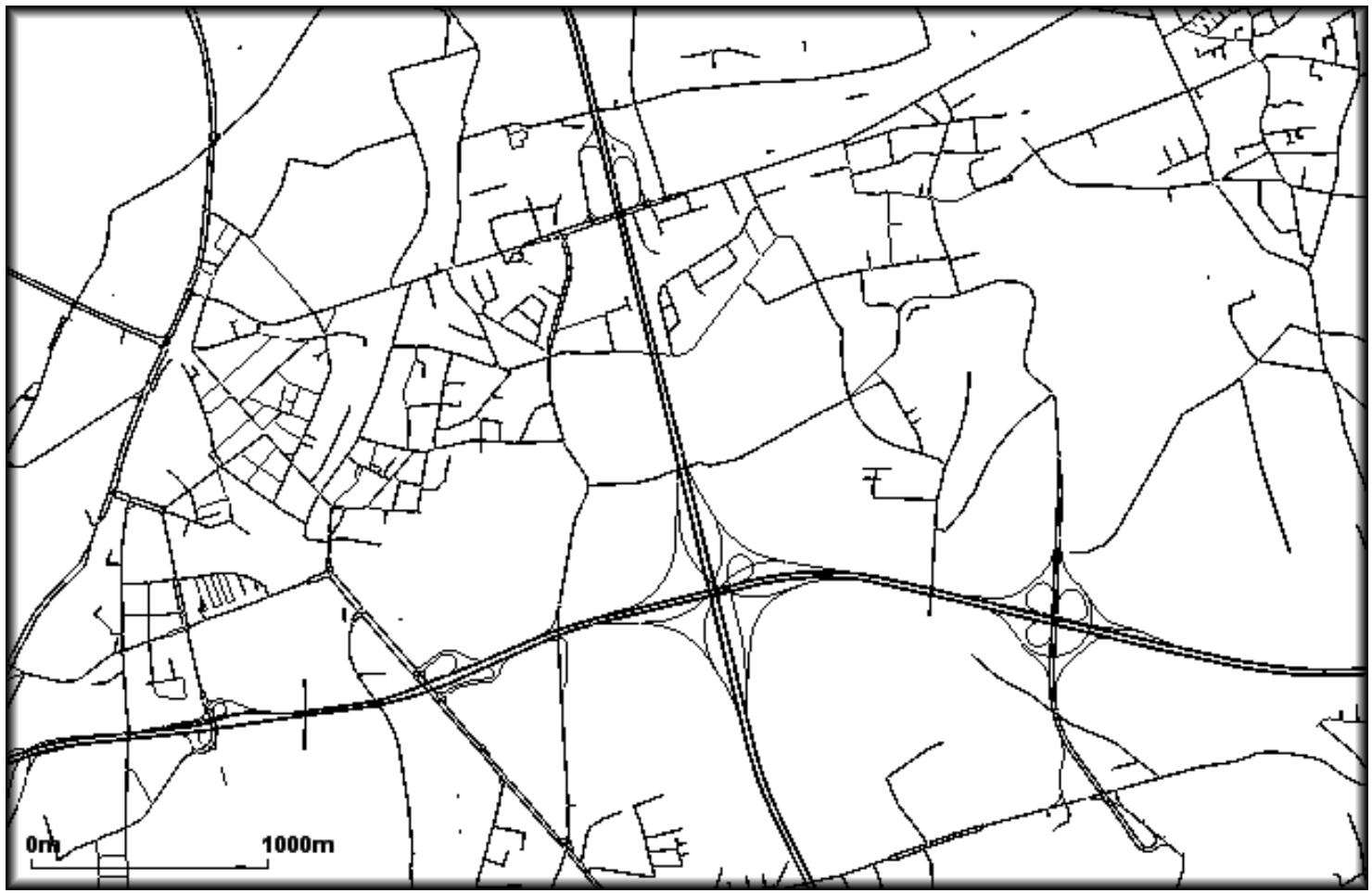
Problem Description



Problem Description



Problem Description



Problem Description

Assume you need 1min for each junction (if you have the data): 10.000 junctions → 4 weeks of hard (error-proned) work

The only solution: use algorithms that do the job

Computing lane-to-lane connections - Overview

Steps:

1. for each edge: compute turnaround edges
2. for each node: sort each node's edges
3. for each node: compute each node's type
4. for each node: set edge priorities
5. for each edge: compute edge-to-edge connections
6. for each edge: compute lanes-to-edge connections
7. for each node: compute lane-to-lane connections
8. for each edge: recheck lanes
9. for each edge: append turnarounds

... quite many; we will not present them all herein.
A complete description may be found in the publication and the source.

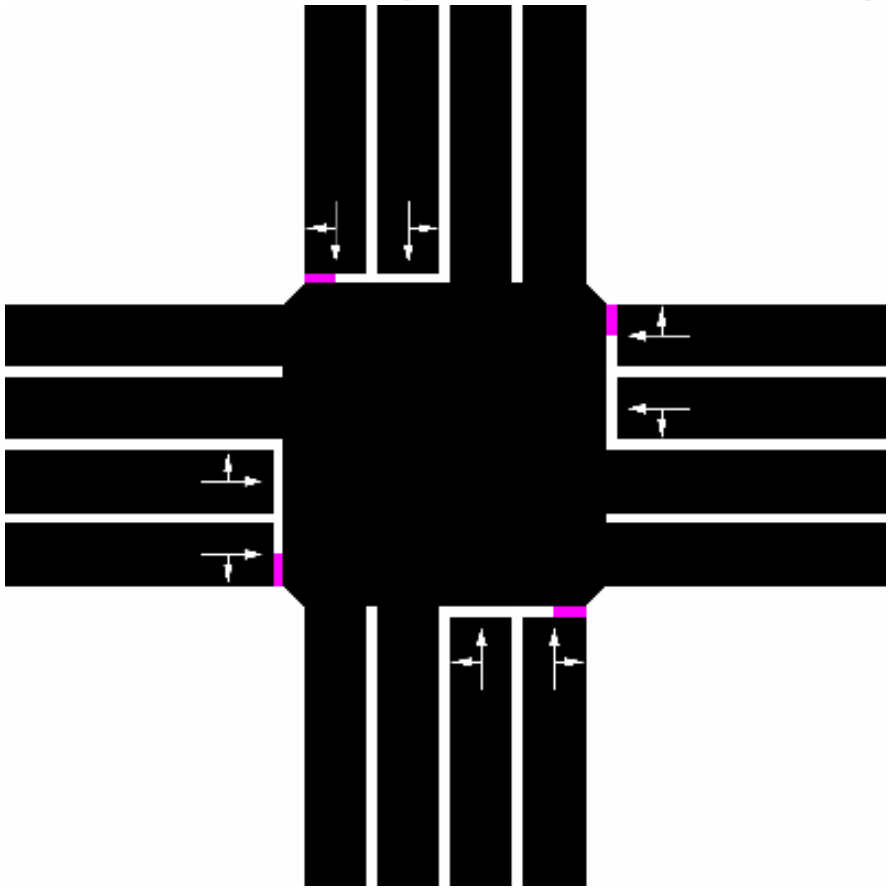
Computing lane-to-lane connections - Overview

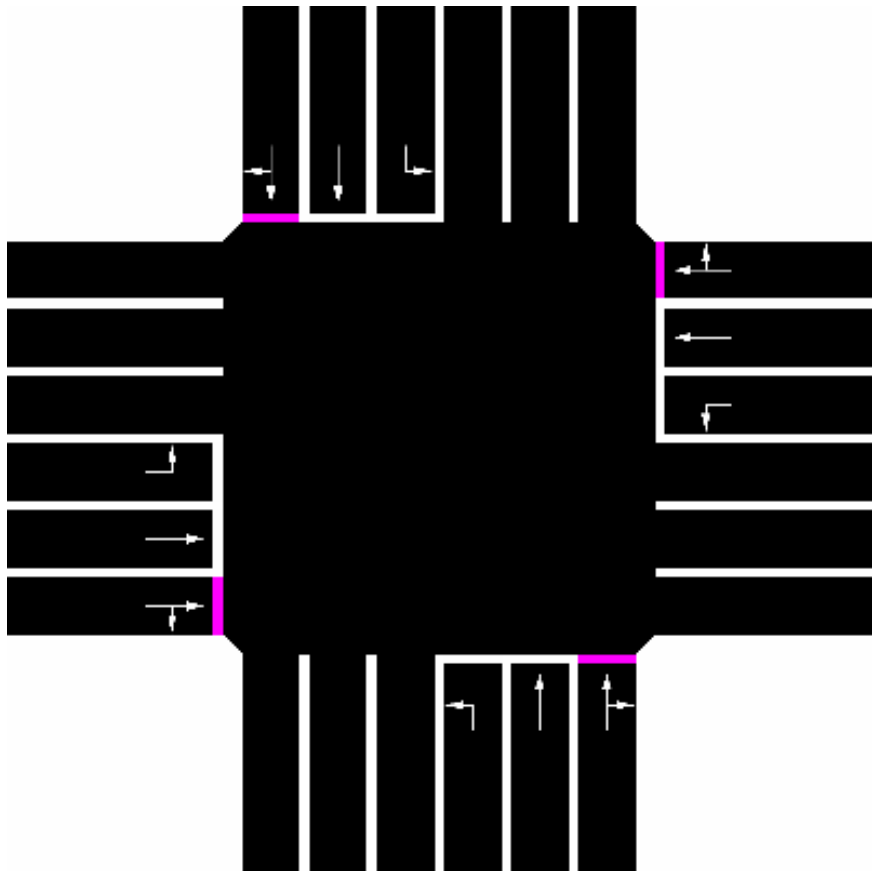
Steps:

1. for each edge: compute turnaround edges
2. for each node: sort each node's edges
3. for each node: compute each node's type
4. **for each node: set edge priorities**
5. for each edge: compute edge-to-edge connections
6. **for each edge: compute lanes-to-edge connections**
7. for each node: compute lane-to-lane connections
8. for each edge: recheck lanes
9. for each edge: append turnarounds

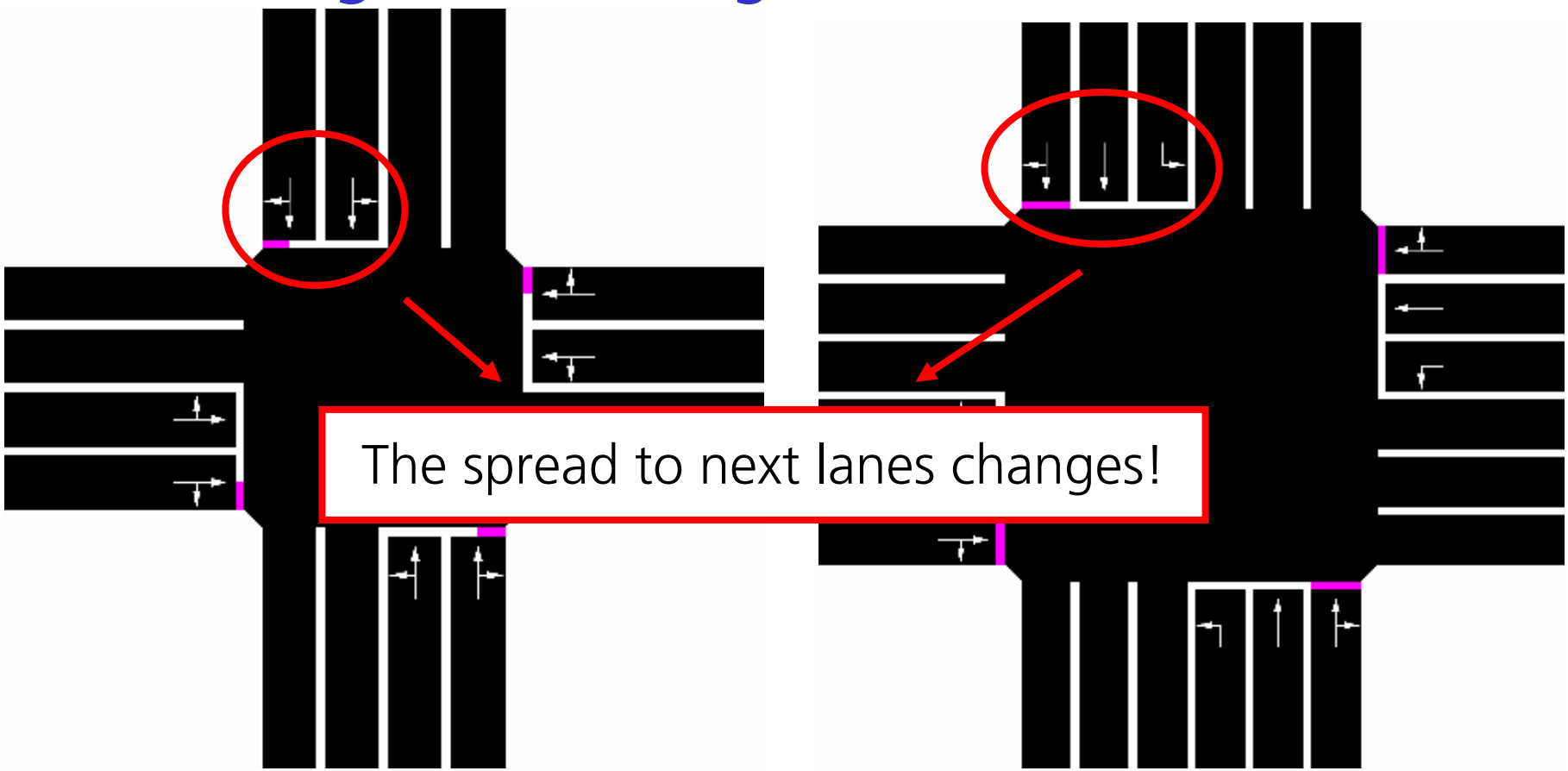
... quite many; we will not present them all herein.
A complete description may be found in the publication and the source.

Determining lanes-to-edge connections





Determining lanes-to-edge connections



Determining lanes-to-edge connections

Solution is using heuristics in step 6:

- get the list of connected edges beside the turnaround
- sort them by their angle
- for each edge in this list, compute its priority for the current edge:
  (priority = (connected edge's junction_priority + 1) * 2)
  - if one of the lower priorised outgoing roads goes to the right:
    - divide his importance by 2 as vehicles using it can leave the junction faster
  - if there are no major roads at this junctions:
    - multiply the outgoing road that goes straight by 2, making it more important then the others
  - compute the number of lanes that shall approach each of the connected edges:
  - sum up all priorities
  - for each outgoing (connected) edge:
    - number of lanes to use to reach this edge = this edge's priority for the current edge / priority sum
    - if number > number of current edge's lanes:
      - number = number of current edge's lanes

Determining lanes-to-edge connections

Solution is using heuristics in step 6:

- get the list of connected edges beside the turnaround
- sort them by their angle
- for each edge in this list, compute its priority for the current edge:
  (priority = (connected edge's junction_priority + 1) * 2)
  - if one of the lower priorised outgoing roads goes to the right:
    - divide his importance by 2 as vehicles using it can leave the junction faster
  - if there are no major roads at this junctions:
    - multiply the outgoing road that goes straight by 2, making it more important then the others
  - compute the number of lanes that shall approach each of the connected edges:
  - sum up all priorities
  - for each outgoing (connected) edge:
    - number of lanes to use to reach this edge = this edge's priority for the current edge / priority sum
    - if number > number of current edge's lanes:
      - number = number of current edge's lanes

Determining lanes-to-edge connections

What does it mean?

- **All edges get a weight in dependence to their „priority“ (major roads get a higher)**
- **The right turn gets only half priority if the destination is not a major road**

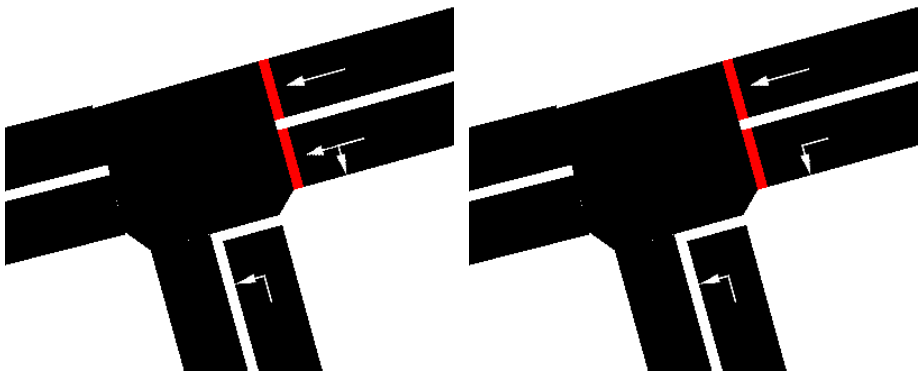
  Reason: right-turning vehicles move faster than left-turning (because they do not have to wait for vehicles coming from the opposite direction)

Validation

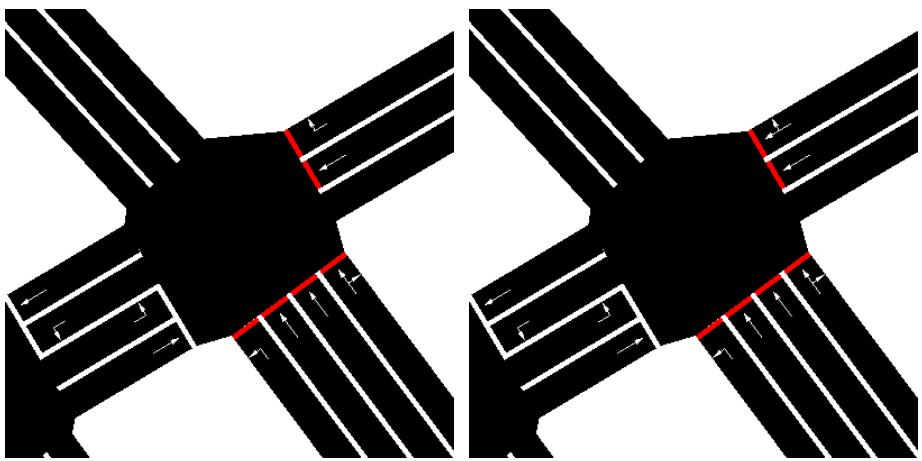
A validation has been done for the OIS-Scenario:

4 of 177 junctions were not proper, yielding in a accuracy of ~98%

Validation - falsely computed Junctions



Reason: unknown flow
(many more vehicles drive left)



Reason: unknown continuation

Conclusion

- The algorithm seems to be useful for most cases;
- In some certain cases, the computed information still has to be edited by hand
- But: several heuristics are used, which
  - Should be verified against reality more deeply
  - Which should be grounded in theory
- Next Steps:
  - Further validation
  - Validation for networks lying within other regions of the world
  - Guessing of traffic light positions, highway on-/off-ramps (in work)

SUMO Project Details

Participants:

- Institute of Traffic Research / DLR
- Zentrum für angewandte Informatik, Köln

current version:	**Version 0.8.2.4**
free download:	**http://sumo.sourceforge.net**
contact:	**Daniel.Krajzewicz@dlr.de**