Routing in Dallas - Fort Worth

Overview

- Simulation Model
- Iterative Planning Process
  - Improvement during Iteration
  - Specific Comparison to TRANSIMS
  - Selection of Routes for Re-planning
  - Route-Loss
- Online Routing (Demo)

Routing in Dallas - Fort Worth

links

- asymetric CA model
- individual speed limit
- uniform number of lanes (no turning pockets)

intersections

- blocks to slow down and stop vehicles
- move to destination link if first site is vacant
- preserve order of incoming vehicles

traffic lights

- activate/deactivate incoming link
- an incoming link has one phase length for all outgoing links
- phase length weighted with the number of outgoing lanes

Routing in Dallas - Fort Worth

Implementation

- Geometric distribution of network on CPUs
- Links are split at the center as boundaries
- Message passing with PVM or MPI
- Static load-balancing using average execution times of previous iteration as link costs

Computational Speed

- Real-Time-Ratio is approx. 15 with 8 CPUs on Gershwin
- Clipping time 8 minutes
- Simulation Time 30-40 minutes
- Planning time 10-15 minutes
- Overall time per iteration: 48-63 minutes

Iterative Routing

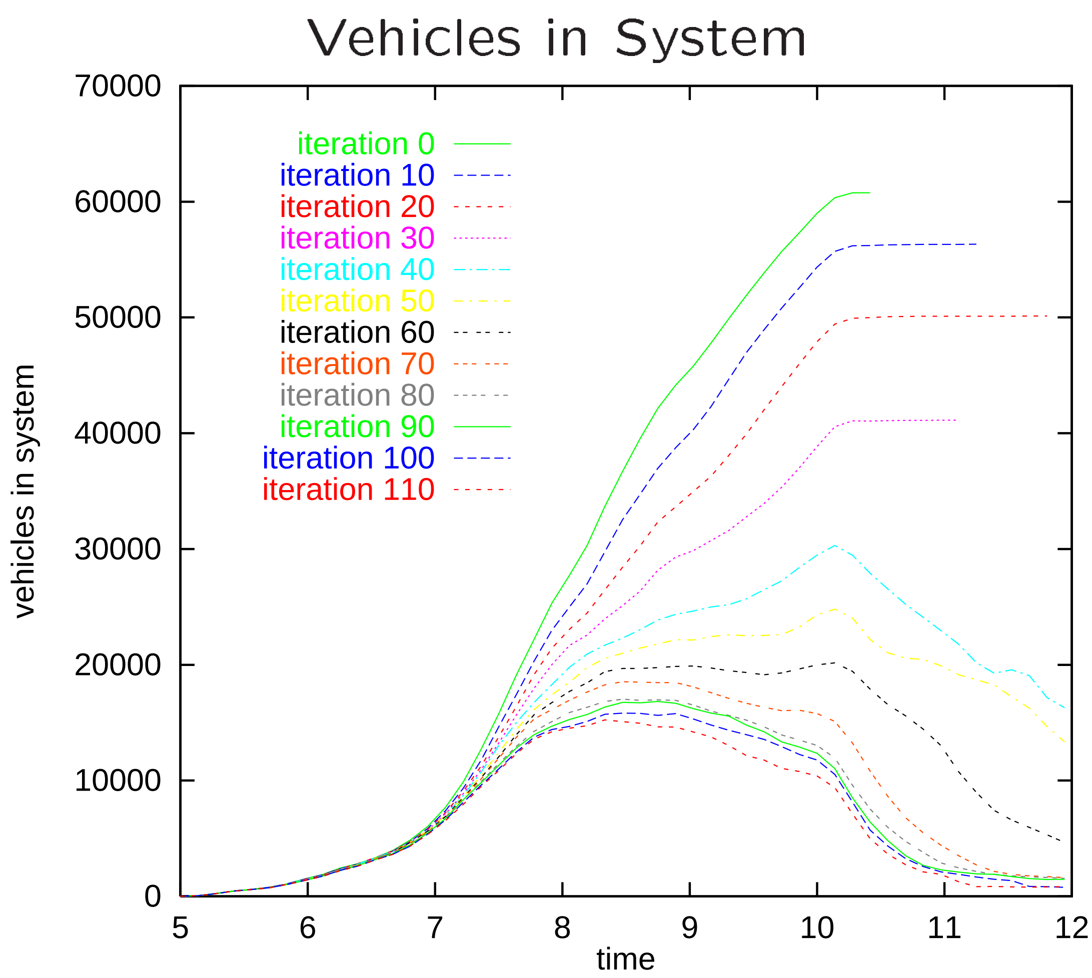
Basic Algorithm

- Start with an initial route-set,
- (*) Clip routes to study-area using free-flow velocities,
- Run micro-simulation ($\rightarrow$ link travel times),
- Re-plan fraction p_0 of routes using link travel times,
- Go to (*) if neccessary.

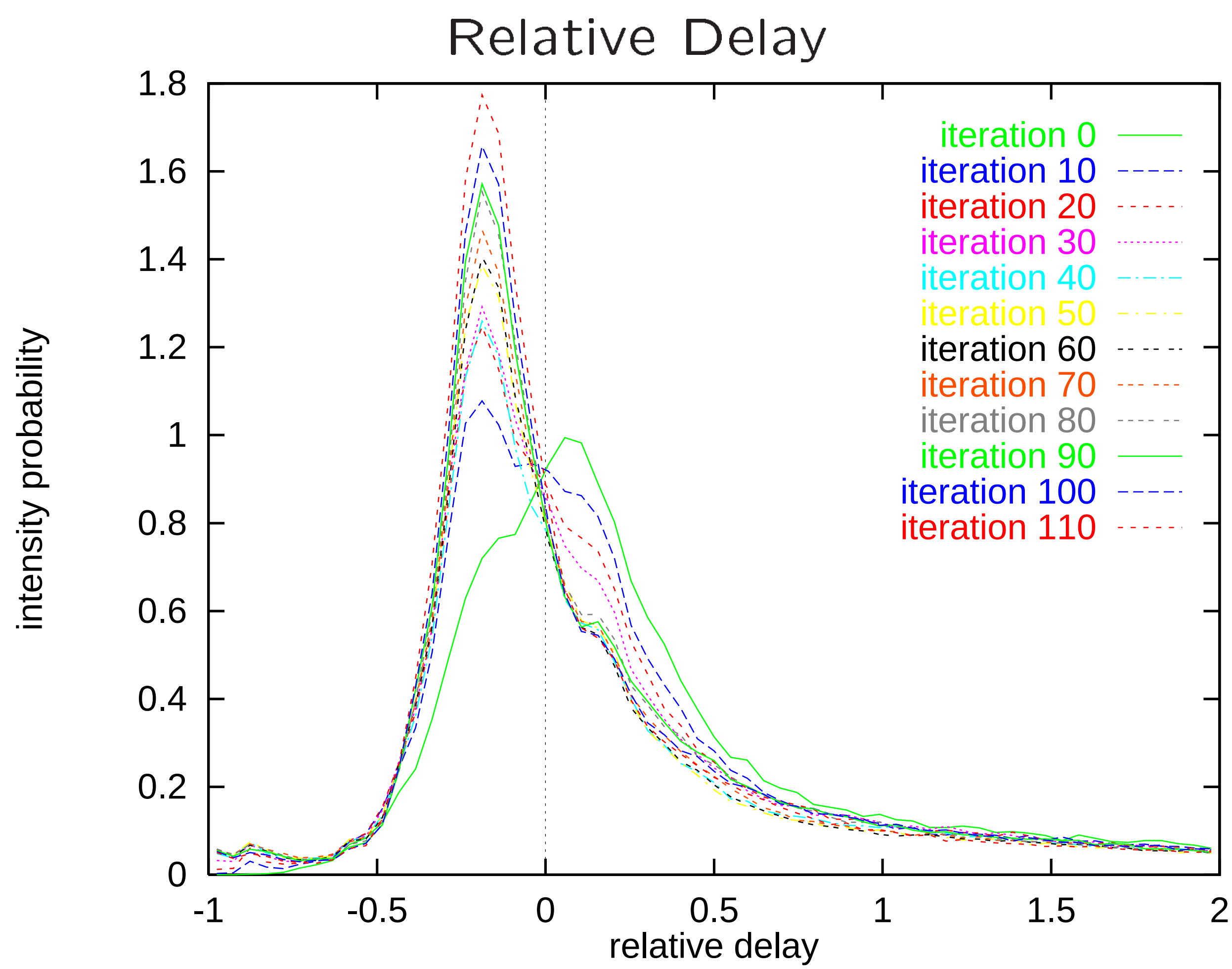
Current Questions (Not answered by this talk):

- Which initial route-set?
- What re-planning fraction?
- What subset of routes is to be re-planned?
- How many iterations?

Iterative Routing



Iterative Routing



Iterative Routing

Observations:

- situation in studyarea improves because
  - routes are distributed better (grid-locks disappear)
  - routes 'leave' study-area ($\rightarrow$ level-0 correction)
- some links are still heavily grid-locked at 10:00 am ($\rightarrow$ selection of routes for re-planning)
- about 10.000 vehicles are still in queues at noon ($\rightarrow$ queue feedback)

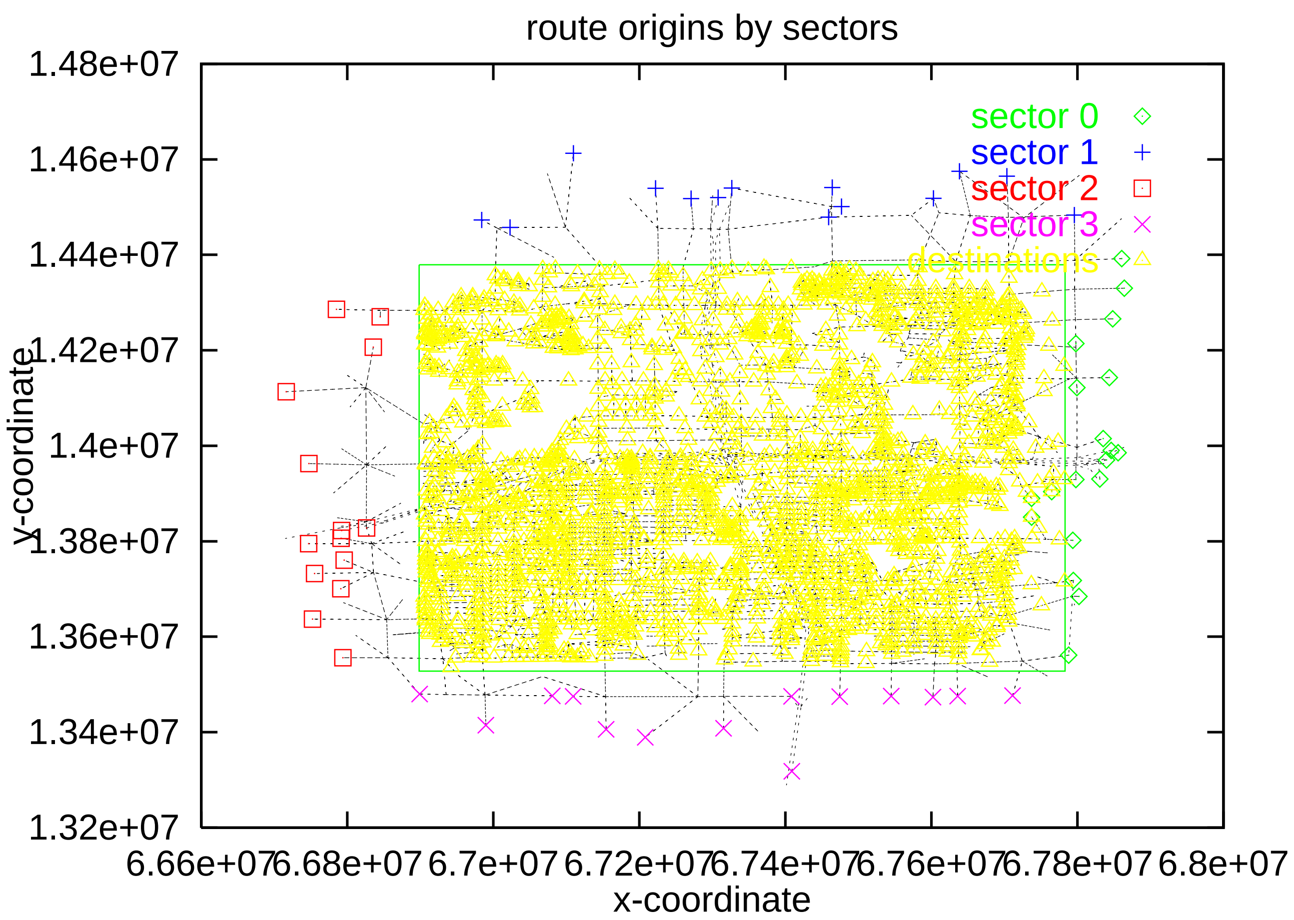
Iterative Routing

Specific Comparison with TRANSIMS-14 Run

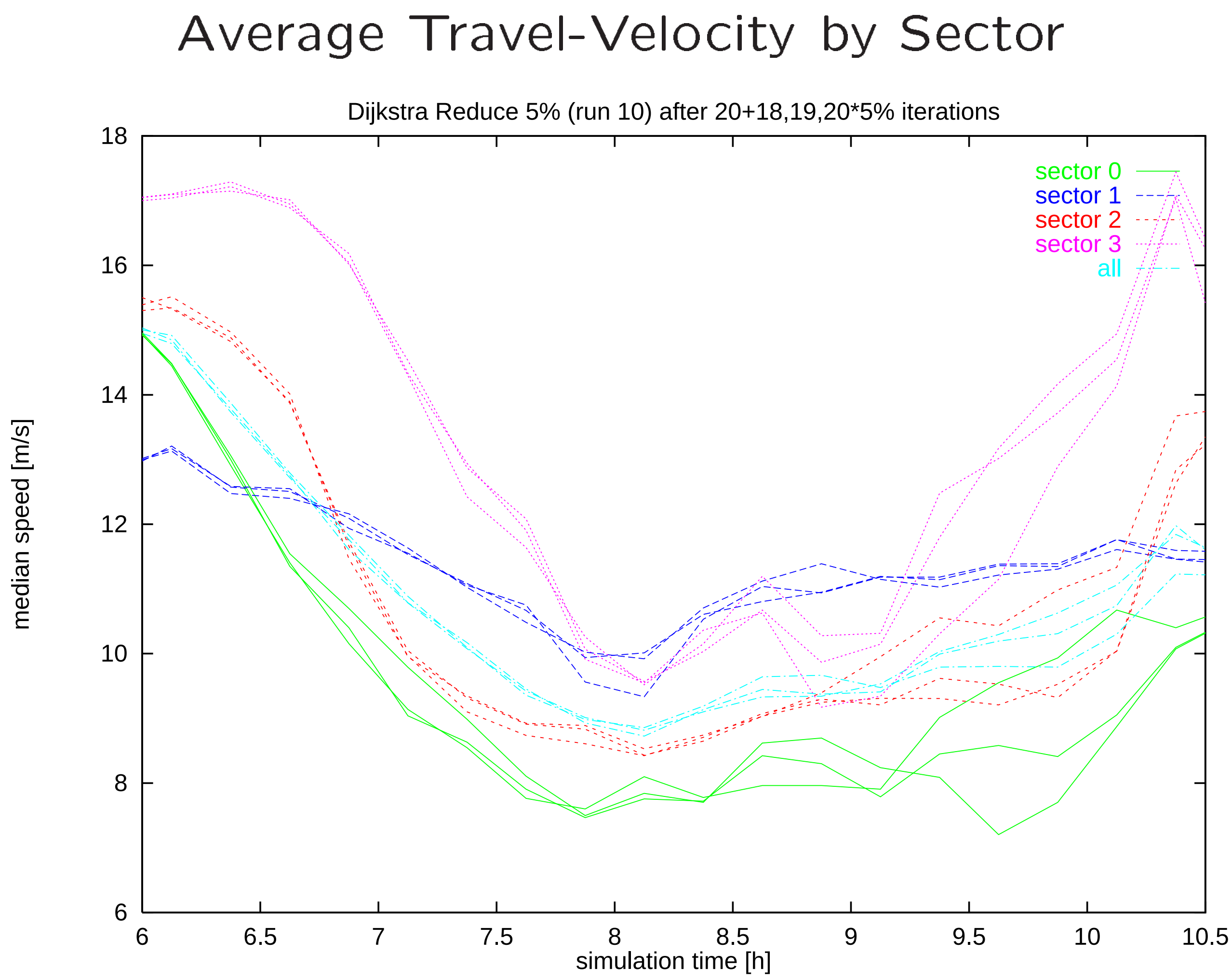
- Look at all routes with destinations inside the study-area.
- For each route store/compute:
  - the source location S where the route enters,
  - the destination location D,
  - the travel-time T from S to D,
  - the angle between S and the center of the area,
  - the average travel-velocity using T and the Eucledian distance between S and D.
- Average over all routes / all routes from each sector (north, south, east, west)

Iterative Routing

Route Origins and Destinations by Sector



Iterative Routing

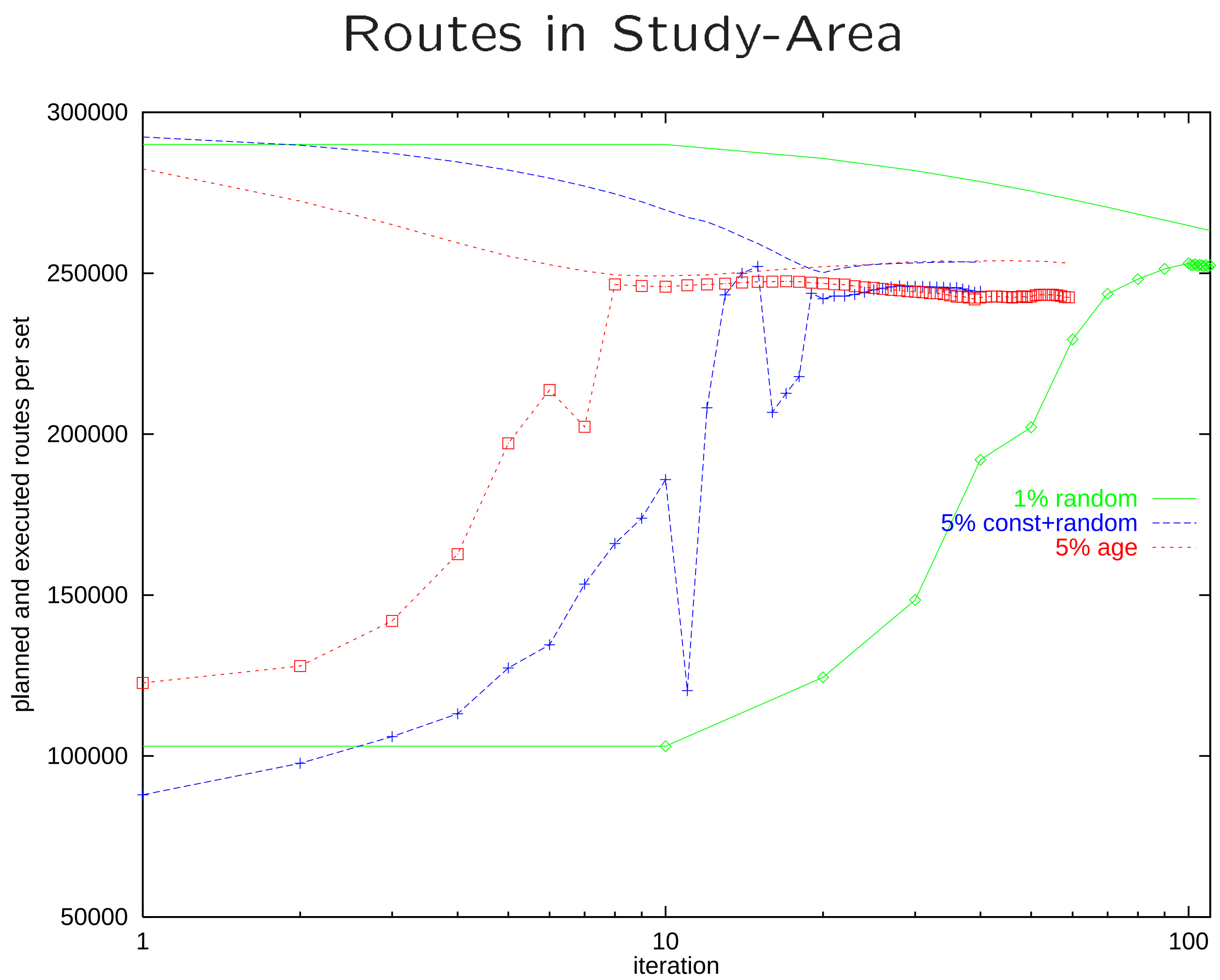


Iterative Routing

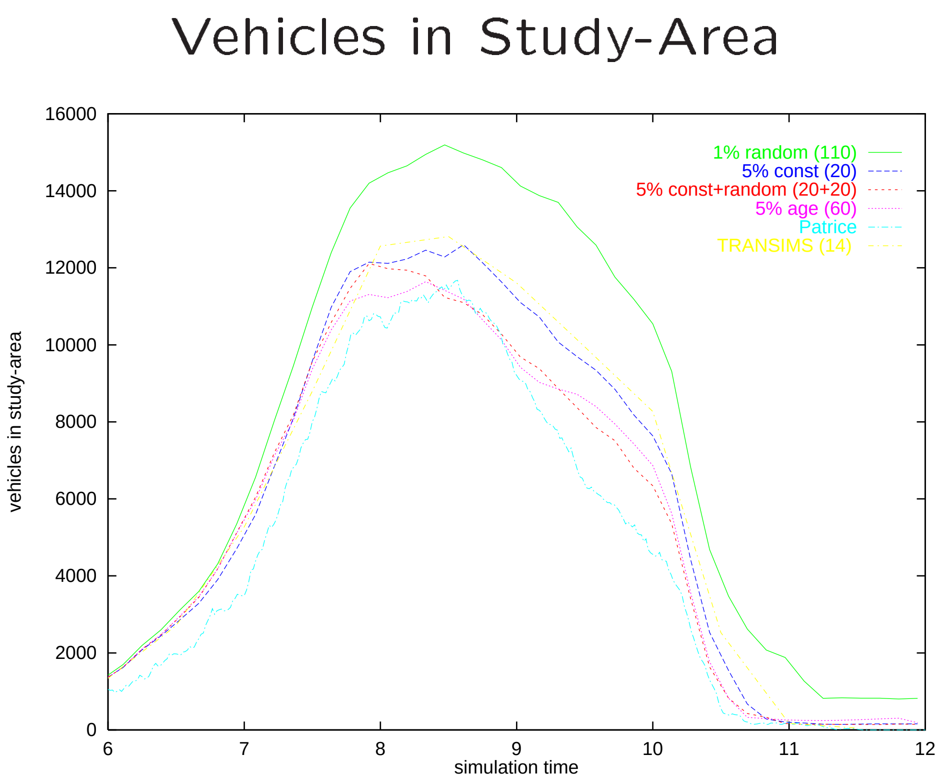
Selection of Routes for Re-planning

- random: $p(a) = p_0$
- constant: replan constant number of original plan-set
- age dependant: $p(a) = qa$

Iterative Routing

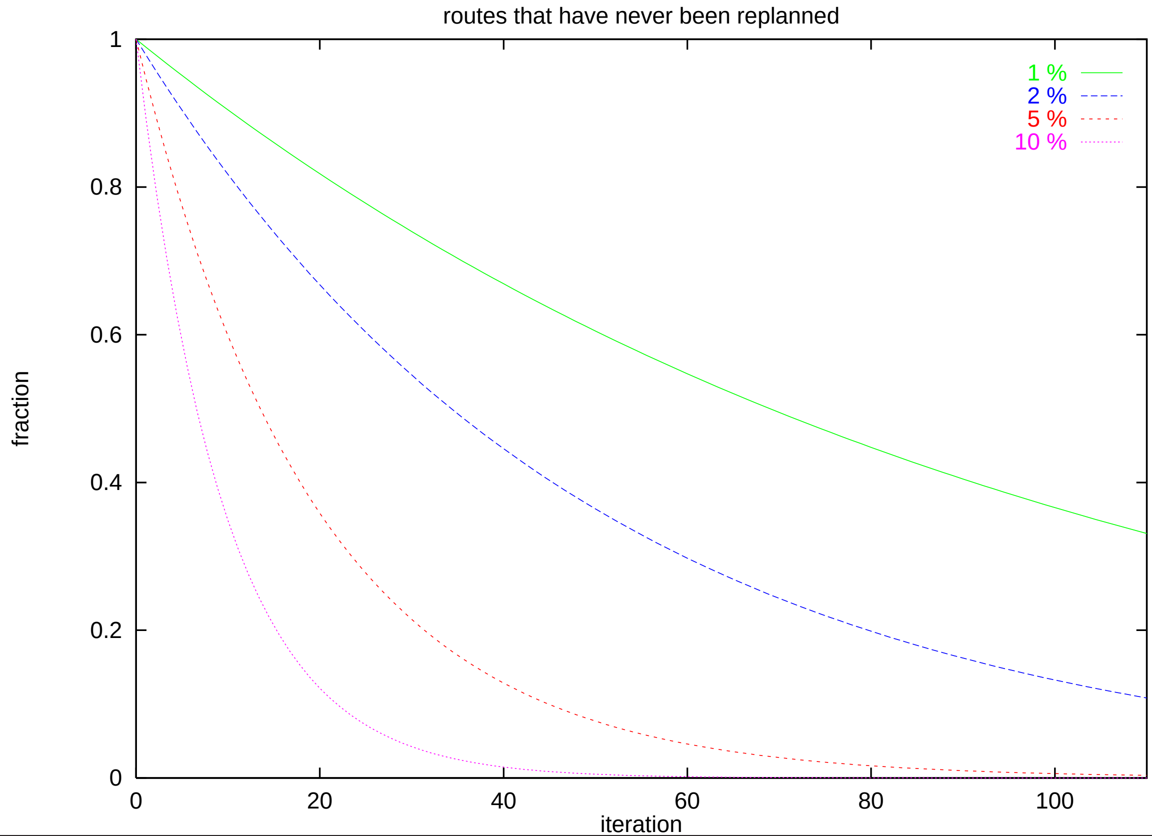


Iterative Routing



Iterative Routing

Problem with random selection: large probability not to be re-planned for small re-planning fraction p_0 even after many iterations n:



Iterative Routing

Age-dependant Re-planning

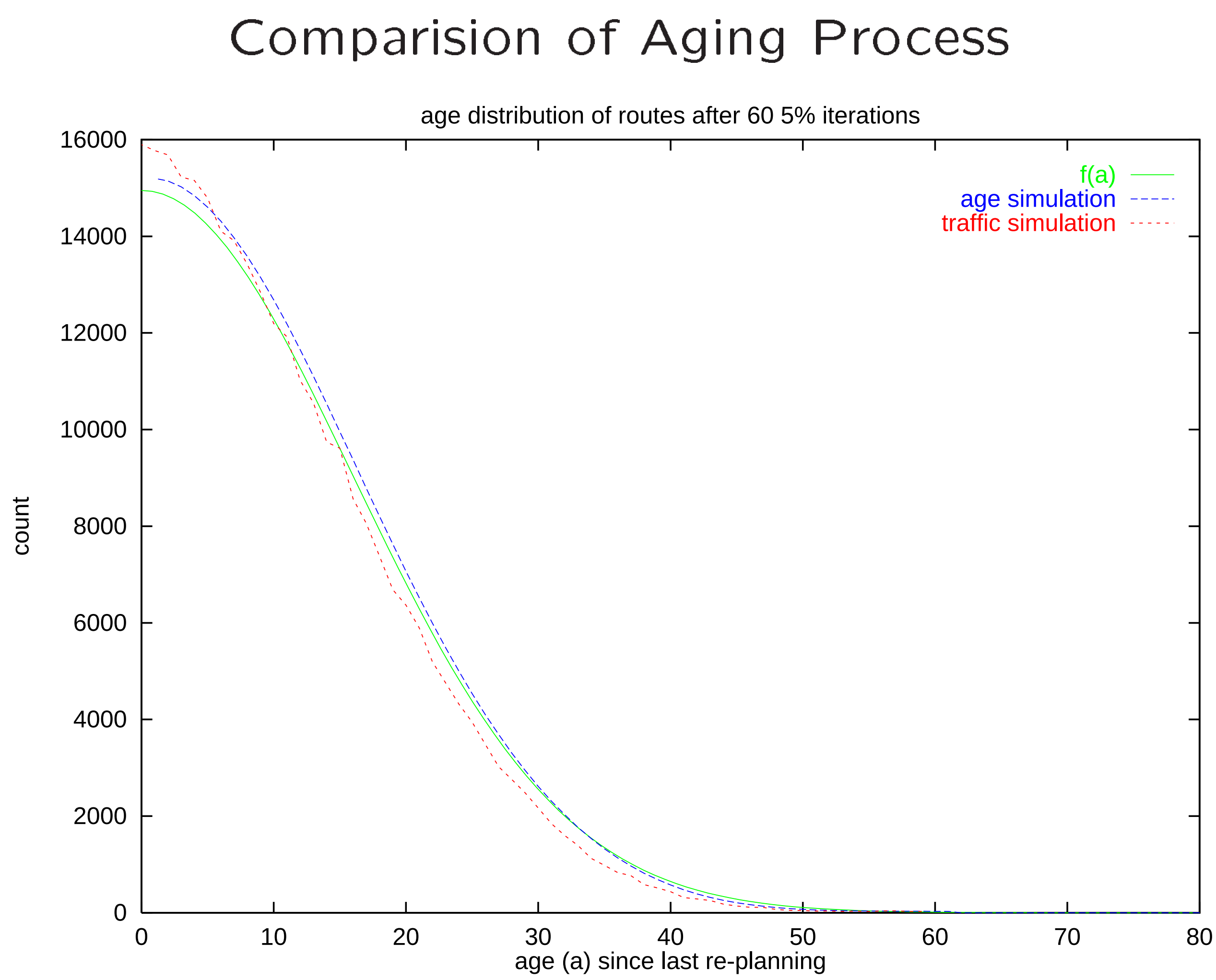
Idea: old plans are to be planned sooner: $p(a) = qa$

This leads to an ansatz for the age distribution:

$$f(a + \Delta a) = f(a) - p(a)f(a)\Delta a$$

$$f(a) = p_0 e^{-\pi/4 p_0^2 a^2} \qquad \text{normal distribution}$$

Iterative Routing



Iterative Routing

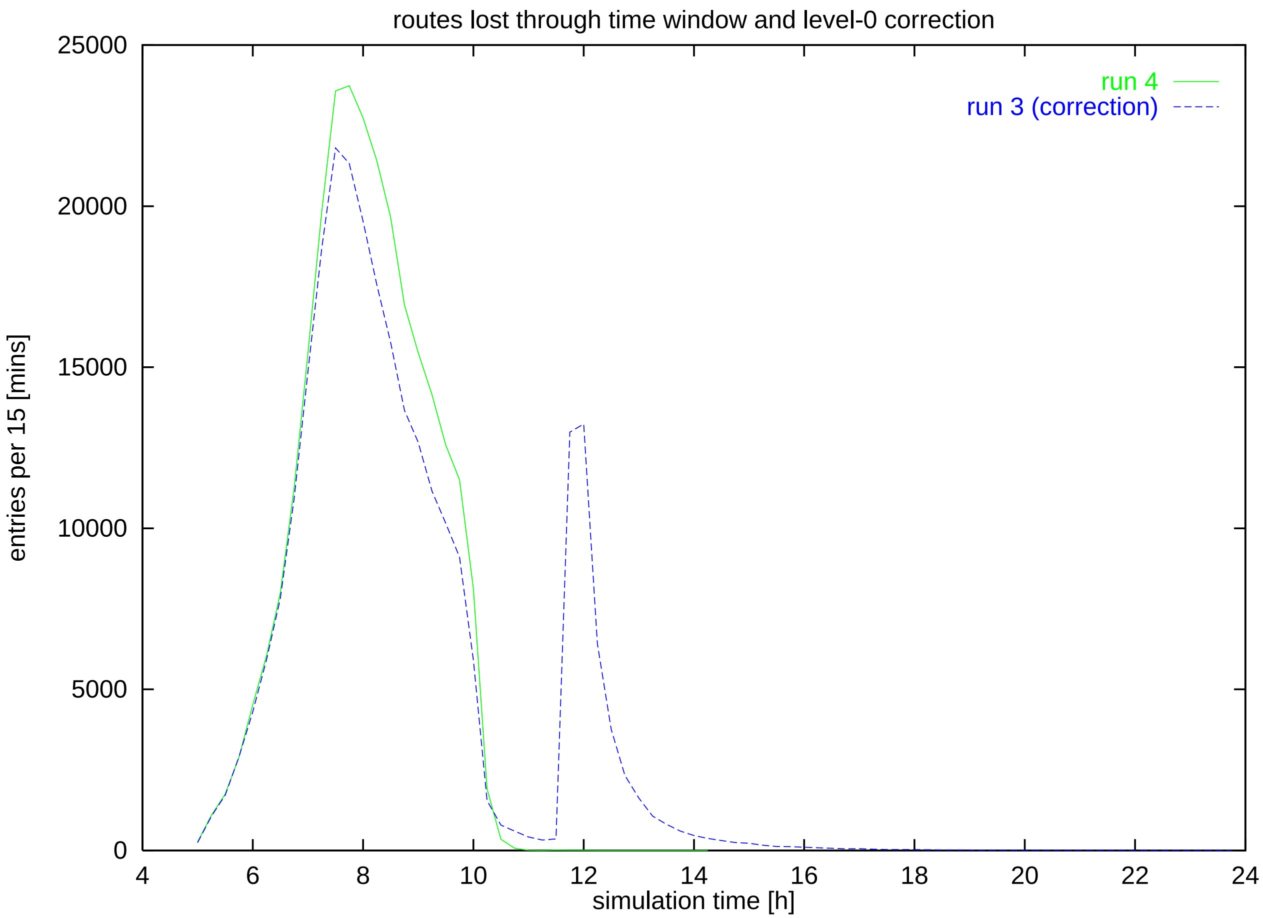
Route Loss: Is it OK? Too many routes anyway?

If it is not, how to prevent it?

Idea: make links ouside the study area look similar (= as bad) as those inside the area ("level-0 correction").
good: reduces loss of plans avoiding the study-area "space-wise"
bad: spreads entry times over a time period $\rightarrow$ loss of plans through time-window
Possible Fix: setting arrival-time instead of departure-time.

Iterative Routing

Routes Lost Through Level-0 Correction



Online Routing

Basic Algorithm

- Collect every 10 time-steps
- Average and broadcast every 120 time-steps
- Select fraction (10%) of travellers by route-ID to have access to route-guidance
- Compare current travel time estimate to the one computed by the planner
- If relative delay $>$ threshold (0.5) $\rightarrow$ compute shortest path with Dijkstra based upon current link travel times
- If relative improvement $>$ threshold (0.2) $\rightarrow$ update route

Online Routing

Comparison between static and 10%-online routing