1’000’000 Personen unterwegs: Eine Mikrosimulation der gesamten Schweiz

Kai Nagel

Department of Computer Science, ETH Zurich

CH-8092 Zurich, Switzerland

http://sim.inf.ethz.ch/

What do we want?

- Determine **system response to policy measures**
  - Measures may be ***time-dependent*** (e.g. ITS).
  - Want to use individual behavioral rules to describe traveler reactions. $\Rightarrow$ ***disaggregated travelers***
  - Want realistic virtual sensor data. $\Rightarrow$ ***disaggregated traffic***
- Enable **unconventional analysis** (how happy are people? how many destinations did they reach? how many others did they annoy on the way? ...)
  $\Rightarrow$ ***disaggregated***

$\Rightarrow$ want ***time-dependent*** and ***disaggregated***

Outline

- From 4-step process to **agents**
- **Mobility simulation**
- **Strategy generation**
- **Relaxation/Adaptation/Feedback/Learning**
- **Scenarios:** Gotthard, CH6-9, equil-net-acts
- **Future**

From 4-step process to agents

Traditional method: 4-step process

E.g. EMME/2, VISUM, POLYDROM.

Trip gen., Trip distrib., mode choice, route assignment.

Major **advantage** of 4-step process:

> ***Route assignment (= 4th step) has unique solution***

(in terms of link volumes; under some conditions).

$\Rightarrow$ **Any *correct computation will yield same result.***

Simplifies analysis enormously.

From 4-step proc. to agents

MAJOR shortcoming of 4-step process:

No dependence on time-of-day.

E.g.:

- No evaluation of time-dependent ICT capabilities.
- No peak-hour spreading; no scheduling reaction at all.
- In general: Use of behavioral rules not possib./plausib.
- Computation of emissions difficult to impossible.

Not known how to change this within 4-step without losing main advantage (mathematically proven uniqueness).

Micro-simulation

Alternative to 4-step process: **Micro-simulation**.

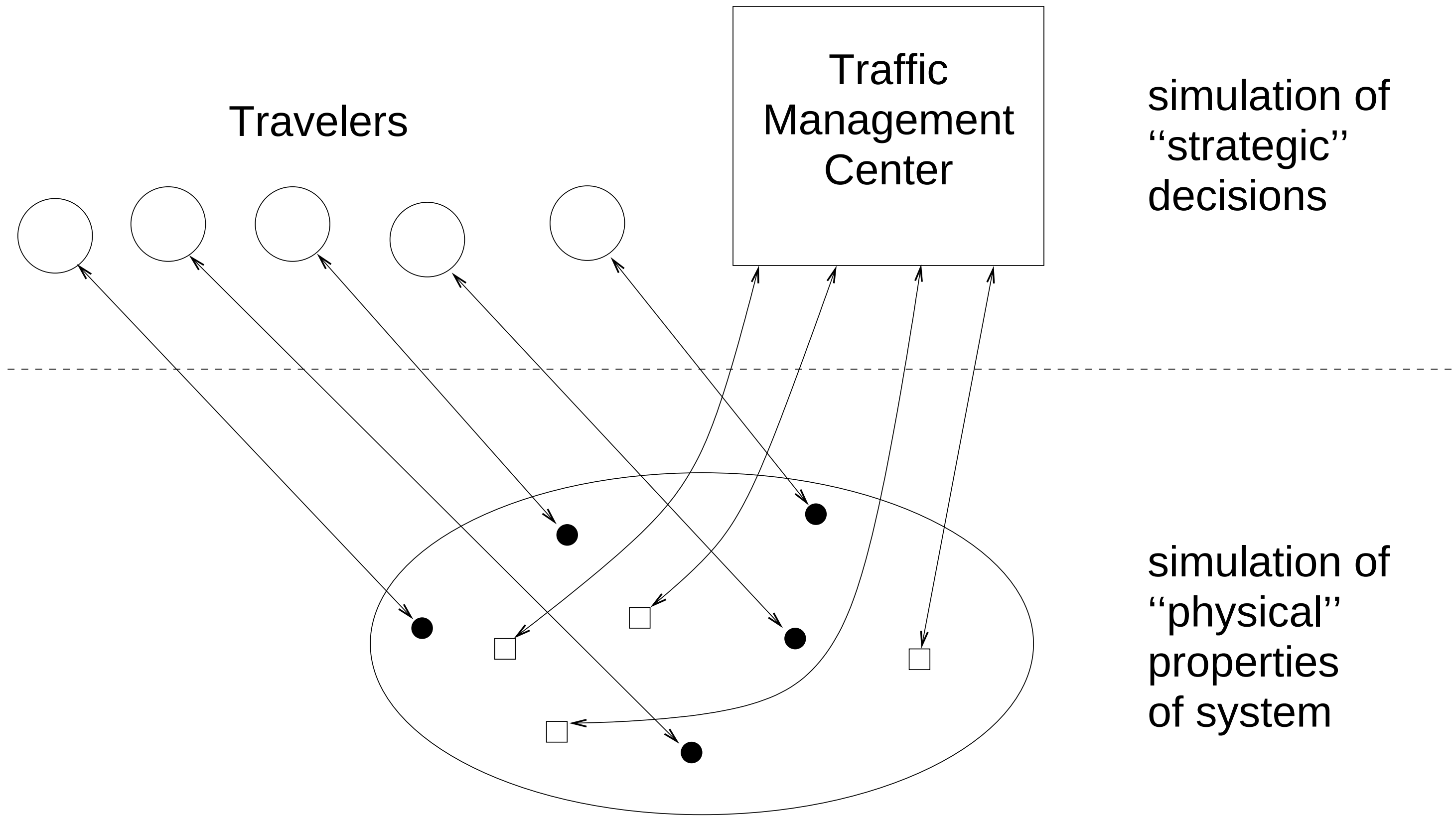
Micro-simulation: *Everything* (travelers, vehicles, traffic lights, etc.) can be individually resolved ...

... in principle. :–)

In practice, limits of

- coding,
- knowledge,
- data needs.

Physical vs. strategical level



Different focuses:

- Strategical level: psychology, sociology, AI
- Physical level: engineering, physics

Physical simulation ($=$ mobility simulation)

Traffic micro-simulation

Can do realistic traffic micro-simulations:

Even more realistic: vissim, paramics, mitsim, aimsun, ...
[[alps!!]] Christian Gloor, Duncan Cavens, Eckart Lange.

Traffic micro-simulation, ctd

Sometimes, “very realistic” is too slow. Then use simulations which have less detailed dynamics:

E.g.: dynamit, dynasmart, dynemo, netcell, queue sim.

With such a simulation: How much computing time to simulate 24 hrs of car traffic in all of CH?

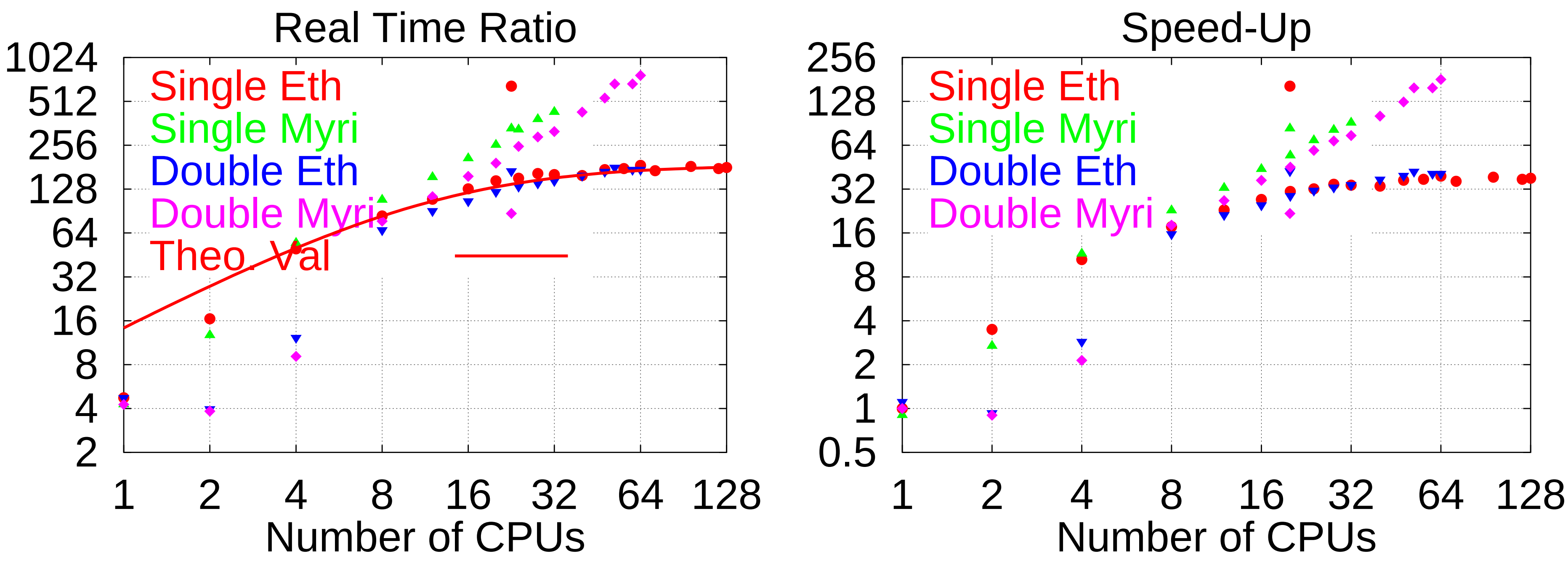
2 minutes

Queue sim. plus jam spillback (www.matsim.org); 64 Pentium CPUs with Myrinet communication; excl. output.

Makes agent-based approach to large-scale land-use planning possible.

Computational speed all of CH

(Real Time Ratio: How much faster than reality.)



- With Ethernet, saturates at approx $RTR = 170$.
  Indep of system size!!
- Super-linear speed-up
- Work by Nurhan Cetin.

Mobility simulation, status

Queue simulation with above computational speed implemented ...

... and seems to work well.

Note: Entirely based on data from static assignment, i.e. data that is usually already available.

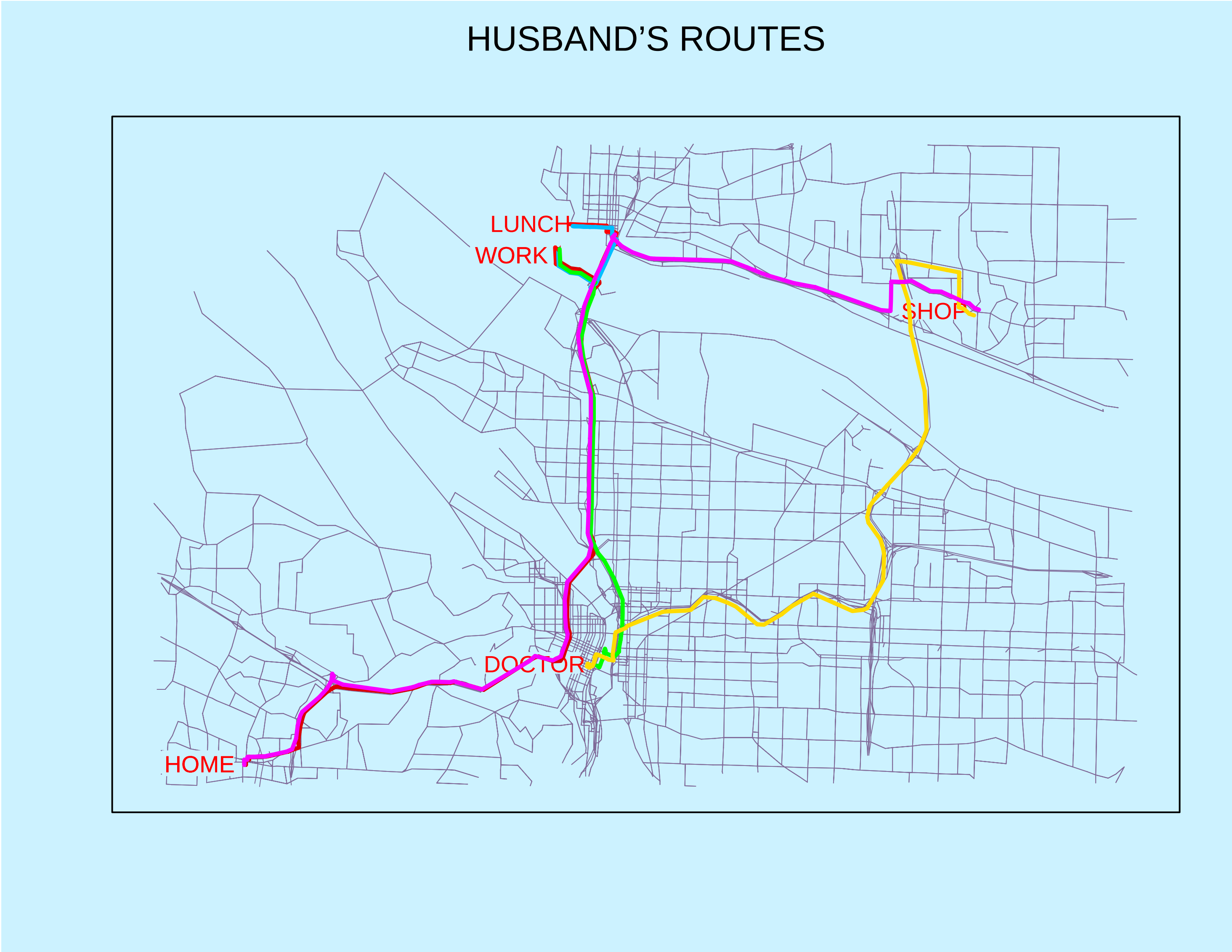
Strategy generation

Demand/Multi-agent design

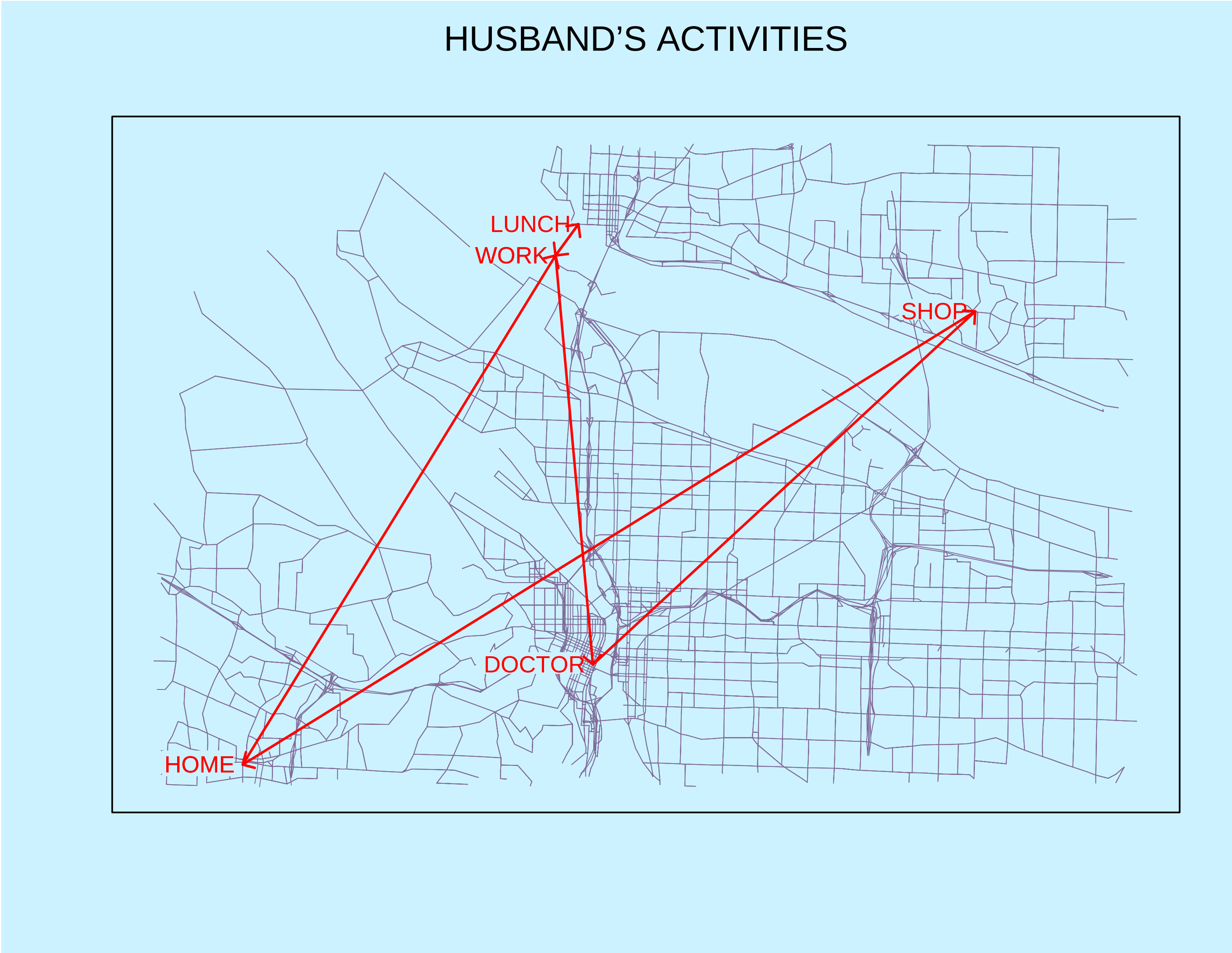
Strategical layer (demand generation) as discussed earlier

Also here do ***everything*** on the level of **individual people/agents**.

E.g. route choice



E.g. acts pattern/location choice



Daily plans in computer

Something like:

```
<person id="13" income="50kEuro/yr" >
    <plan score="-561">
        <act type="h" location="..." end_time="06:00" />
        <leg num="0" mode="car" expected_trav_time="00:15:04">
                <route>2 7 12</route>
        </leg>
        <act type="w" location="..." dur="08:00" />
        <leg num="1" mode="car" expected_trav_time="00:39:04">
                <route>13 14 15 1</route>
        </leg>
        <act type="h" location="..." link="1" />
    </plan>
    <plan score="-463">
        ...
    </plan>
</person>
```

for each person and each trip in the simulation.

Feedback/Learning

Learning/Adaptation

Real-world travelers learn, e.g.: If travel takes too long, ...

- ... try other route/mode/departure time.
- ... drop trips.
- ... find other job or other home.
- Etc.

Basic agent-based learning

1. All agents compute **initial plan** (strategy layer).
2. **Mobility simulation** is executed with those plans.
3. Some agents (e.g. 10%) find **new plans** (e.g. fastest path based on last iteration).
4. Goto 2.

Need some stopping criterion ...

Improved agent learning: Agent database

1. All agents compute **initial plan**.
2. **Mobility simulation** is executed with those plans ***and performance of each individual plan is recorded.***

   (score, fitness, utility, prospect theory, ...)
3. Some agents (e.g. 10%) find and select **new plans** ...

   ... ***but keep old plans in memory.***
4. Other agents **select between existing plans** according to performance ...

   ... or (w/ small proba) make random choice to re-evaluate.
5. Goto 2.

Intuition for agdb

AgentID	PlanID	Score	Description
20	1	123.4	lv home 8am and wlk to bus; take bus 6 ...
	2	133.7	lv home 8am w/ car; ...
	...	...	...
23	...	...	...

Advantages of agent database

- Considerably more robust: Many new plans can be bad and it still works.
- Scoring can be (somewhat) indep from how plan is constructed $\Rightarrow$ more consistent (see later).
- Each agent could start Genetic Algorithm on plans it knows.

Work by Bryan Raney.

Some conceptual issues w/ lrn/rpln

- 1-agent learning (e.g. new arrival to city)
- N-agent learning (all agents new)
- Day-to-day vs within-day replanning

1-agent learning

As said above:

- Make initial plan (acts pattern/locs/times, mode/route).
- Try out.
- Modify some or all or the plan. Maybe remember old plan.
- Re-try.
- Etc.

Artificial Intelligence, Maschine Learning, Complex Adaptive Systems

($=$ there is some technology around that can be explored)

N-agent learning

All agents learn simultaneously $\longrightarrow$ dynamics of the coevolutionary learning system.

If **deterministic**:

- Goes to an attractor (fixed point, periodic, chaotic).
- If “learning $=$ improving”: Attractive fixed point $\Rightarrow$ Nash Equilibrium ($=$ traditional solution).

If **stochastic**:

- Goes (normally) to stationary state space density (Markov).
- Can however be stuck in sub-regions of state space for arbitrarily long times (broken ergodicity).

Day-to-day vs within-day

Day-to-day: Every agent pre-plans whole day; whole day is executed; some agents change plan (over night); whole day is executed; etc.

- Modules can be coupled via files. External modules easy to integrate.
- Consistent with N-agent learning theo.
- Not very realistic.

Within-day: Agent is able to change plan *within* day.

- More realistic.
- Harder to implement (ext. modules; parallel comput.).
- More difficult to fit into theory [[ask]].

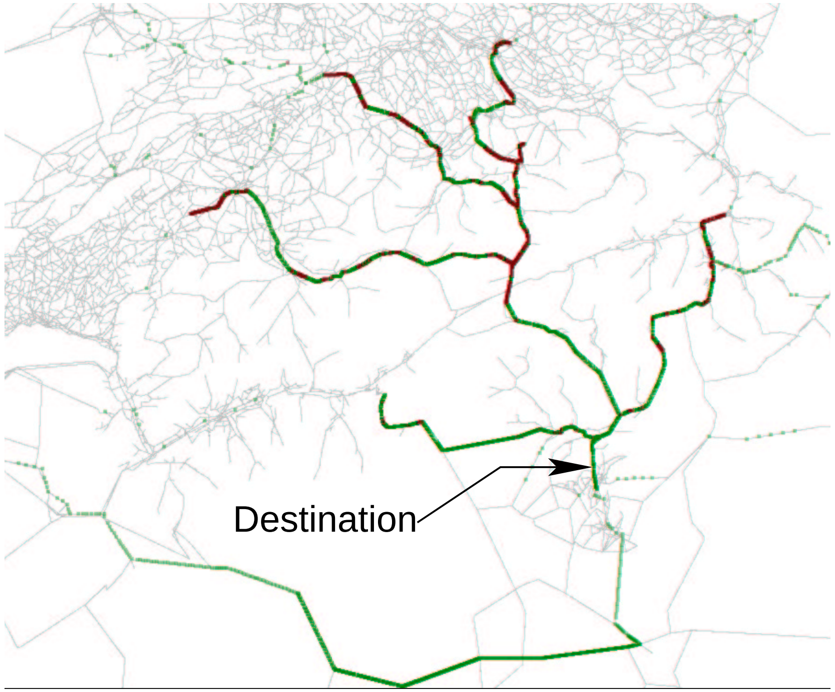
Scenario 1: "Gotthard"

(Agent-based dynamic traffic assignment (DTA) test scenario)

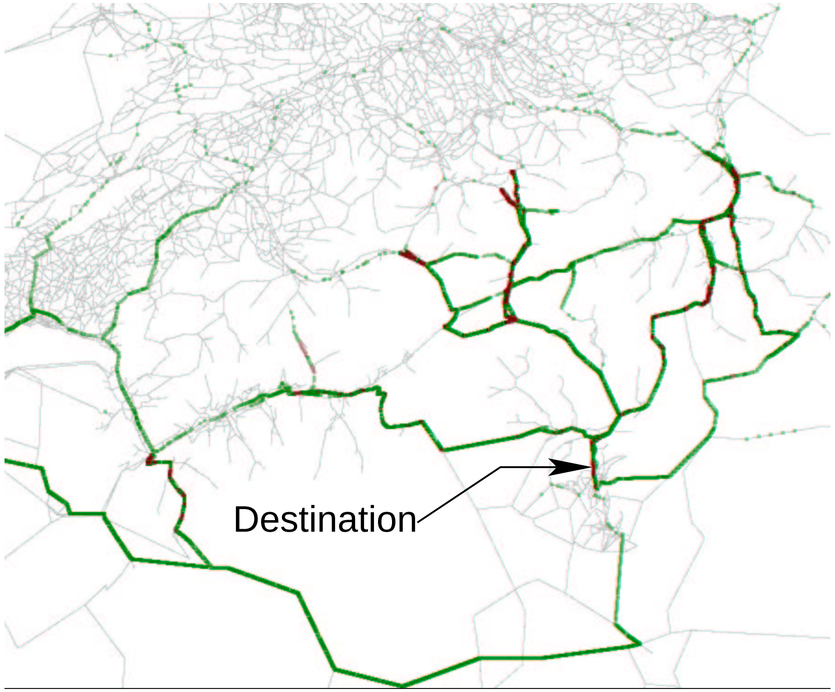
Gotthard scenario description

- **Network** with 20 000 links (major streets only).
- **Demand** ... 50 000 travelers all over Switzerland, starting between 6am and 7am, destination Lugano.
- **New plans:** Routes only, fastest path based on last iteration.
- Agdb as explained above; choice between old routes $e^{-\beta T_i}$.
- 50 iterations.
- **Mobility simulation** queue simulation as mentioned above.

Gotthard result



Everybody on route which would be fastest in empty system.



“Wider” spread of traffic.

Gotthard scenario, summary

- Routes “spread out” during iteration.
- Traffic jams into Lugano reasonably well equilibrated (not shown).
- The first publicly available version of TRANSIMS failed that test.

Note: Smaller test would be sufficient, see later (acts times).

Scenario 2: All of Switzerland

(Agent-based dynamic traffic assignment for real-world case)

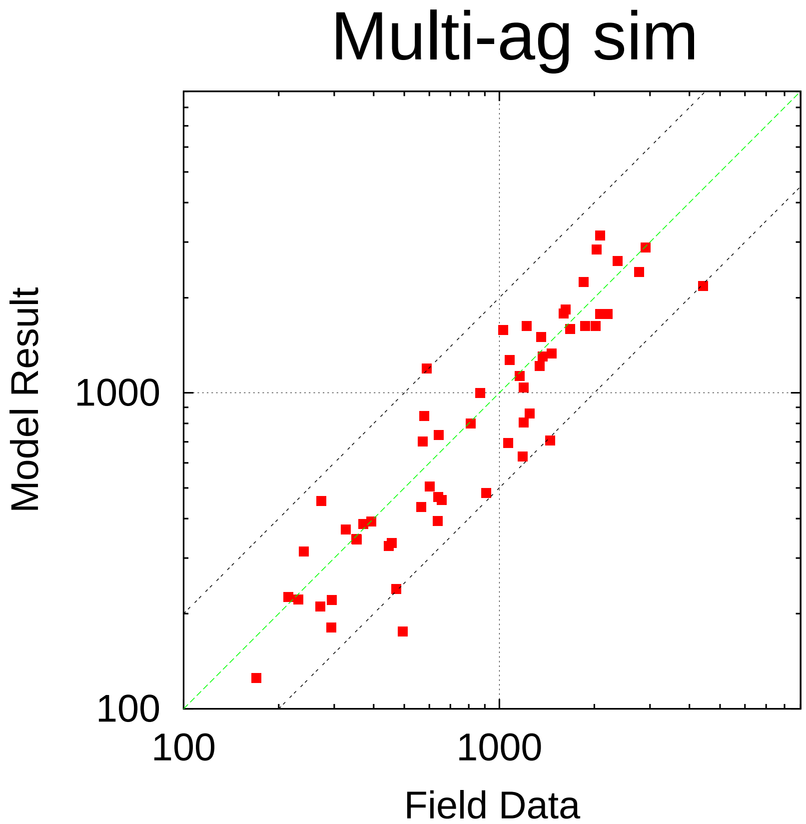
All-of-CH scenario description

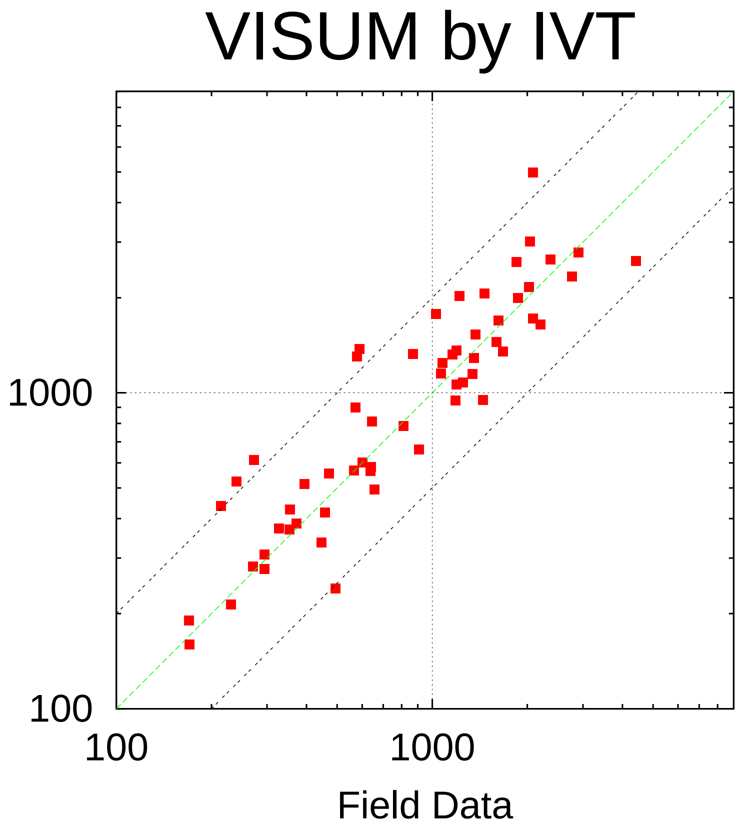
- **Network** with 20 000 links (same as above).
- **Demand** ... time-dependent origin-destination matrices. (Coming from IVT/Vrtic.)

  Disaggregated into individual trips
- **New plans:** Routes only, fastest path based on last iteration.
- Agdb as explained above; choice between old routes $e^{-\beta T_i}$.
- 50 iterations.
- **Mobility simulation** queue simulation as mentioned above.

[[vis ch6-9]]

Validation (7am to 8am, volumes)





	Multi-ag:.	VISUM:.
Mean Rel. Bias:	−5.3%	+16.3%
Mean Rel. Error:	25.4%	30.4%

All of CH, summary

- Can **replace route assignment step** from 4-step process without detoriation in quality.

  $\Rightarrow$ this is now time-dependent and agent-based (remember introduction)

- Can do this for **usefully large scenarios**.

- Activity-based demand generation ... see next.

Scenario 3: equil-net-acts

(Test scenario for activity time choice)

!!Preliminary!!

Scenario description

- **Network** small test network. **[[vis]]**
- **Demand** hwh pattern with same h and same w location for everybody.
- **New act times** constructed by genetic algorithm such that 24-hour utility is maximized. In principle:
  - $\beta_t \, t_{opt} \ln t / t_0$ utl for durations
  - Linear disutilities for travel, wait, late arrival, early departure, etc. – *Use travel times from last iteration.*
  - ***Trip chains!!***
  - Work by David Charypar.
- **New route** fastest path of last iteration.
- **Mobility simulation** queue sim.
- 150 iterations.

equil-net-acts result

[[show]]

Important:

- Each agent uses several full 24-hour dayplans.
- Successively improves them.
- Main problem is to implement this such that it works for large scale scenarios (10 mio agents).

Remark about departure time choice

We are modeling time choice for ***whole dayplans***.

In my opinion, if you really want to understand what's going on, this is the best approach ("gradient"–??).

Examples:

- Penalty of being late totally different for person with **additional acts at end of day** (e.g. kiga, shopping, theater) than for person without.
- Also totally different for person who has **time window for work** start when compared to person w/o time window.
- The **"forces" to being late** depend on what you do before (going to mtg in London from Birmingham).

Dept time choice, ctd

One way out: Have **subclasses** for all these people, and estimate separate coefficients α, β, γ.

However, does not solve core of problem. E.g. assume new kiga at work. Then have to estimate new model for “people w/ kiga at work” vs “people w/o”.

$\Rightarrow$ Our proposal: Construct dayplan and utilities from more atomic contributions.

(David Charypar)

Future

Equil-net-acts $\longrightarrow$ ch-net-acts

Status: Soon (Bryan Raney, Michael Balmer).

Higher resolution Zurich

[[show]]

Status: In progress (Bryan Raney, Michael Balmer).

Syn pop and full act-based demand generation

Also look at effects Glatttalbahn.

Status: In progress (Martin Frick (IVT), Thomas Bernard (IVT), Fabrice Marchal (CoLab), Bryan Raney (SIM), Michael Balmer (SIM))

Within-day replanning

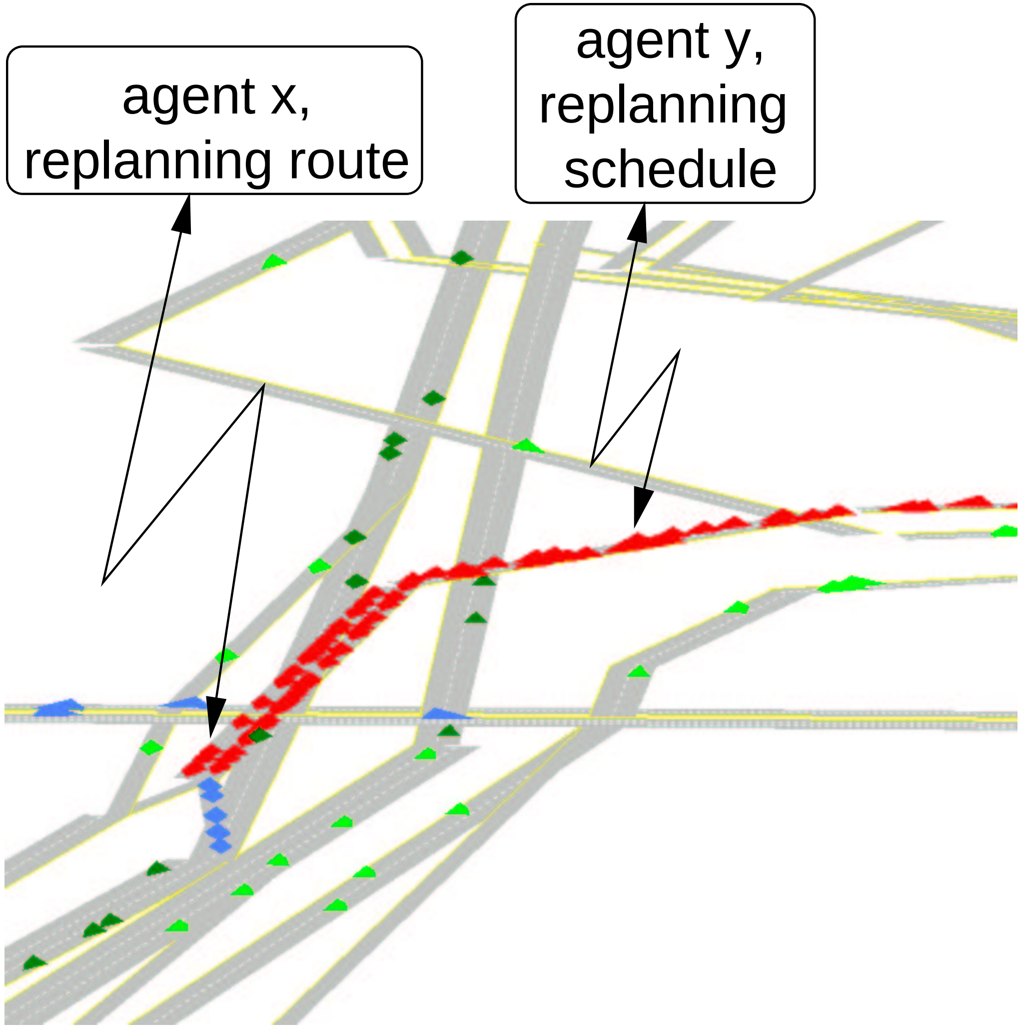
What: Agents should be able to modify plans not only over night, but during the day.

Problems: Standard subroutine calls easy to implement, but destroy computational performance, and are difficult when modules are developed with different programming languages on different platforms.

Our approach: Use messages between mob sim and strat layer.

Status: Works for “hiking in the alps”; does not (yet??) work for traffic simulations.

Messages, intuition



Summary

Summary

Truly agent-based

10 mio agents

Fairly realistic

Activity-based demand generation in progress ...

Will become useful for land use research

Acknowledgments

Nurhan Cetin – parallel computing

Bryan Raney – agent-based learning; all-of-CH

Christian Gloor – message-based simulation architecture; hiking in the Alps

Duncan Cavens, Eckart Lange – graphics; hiking in the Alps

Fabrice Marchal (postdoc) – help with data

Kay Axhausen, Martin Frick, Michael Bernard – real world data; behavioral rules

Condition-action-pairs

Game theo for chess: For every configuration (state, condition), give move (response, action).

Q-learning for chess: Learn book of condition-action-pairs via iteration. (Slow, but conceptually possible.)

Alternative: Compute action only when condition is met.

In same way, could construct “tree” of conditional behavior for traveler.

Ultimatum game

Unfortunately, "best action" when computed before the game not always "best action" when computed during game.

Example (ultimatum game, Stackelberg game): Cold war:

- U.S. decides *before* the game that it will retaliate to nuclear attack. Decision can*not* be changed during game.
  Result: Russians do not attack.
- Decision *can* be changed during game. That is, after Russian attack U.S. finds that retaliation makes own situation even worse.
  Result: Russians do attack, U.S. does not retaliate.

Well known $\rightarrow$ U.S. attempted to make retaliation automatic.

Ultimatum game in traffic/economics

Same situation in traffic: Traffic management center vs traveler adaptation.

Same situation in economics: Price setting vs consumer adaptation.