Using mental simulation to improve the agent learning rate of large-scale multiagent transport simulations

Author(s):
Fourie, Pieter Jacobus; Illenberger, Johannes; Nagel, Kai

Publication Date:
2013

Permanent Link:
https://doi.org/10.3929/ethz-a-007340021

Using mental simulation to improve the agent learning rate of large-scale multiagent transport simulations

Date of submission: 2012-07-15

Pieter J. Fourie
Future Cities Laboratory, Singapore-ETH Centre, #06-01 CREATE Tower, Singapore, 138602
phone: +65 9116 8440
fax:
fourie@ivt.baug.ethz.ch

Johannes Illenberger
Transport systems planning and transport telematics, Institute for Land and Sea Transport Systems, TU Berlin Sekr. SG12, Salzufer 17-19, 10587 Berlin
phone: +49 30 314-78 793
fax:
illenberger@vsp.tu-berlin.de

Kai Nagel
Transport systems planning and transport telematics, Institute for Land and Sea Transport Systems, TU Berlin Sekr. SG12, Salzufer 17-19, 10587 Berlin
phone: +49 30 314-23 308
fax:
nagel@vsp.tu-berlin.de

Words: 5971 words + 6 figures = 7471 word equivalents

ABSTRACT

The Multi-Agent Transport Simulation toolkit, MATSim, employs a trial-and-error evolution-inspired approach, by executing and scoring agent day plans in a time-consuming traffic simulation. This paper introduces an information feedback loop to the MATSim framework that evaluates and improves plans before they are executed. We test the technique in an extensive scenario for Zurich, Switzerland, incorporating mode choice, road-pricing, secondary activity location choice, activity timing adjustment and dynamic routing. We find that the technique dramatically improves convergence rates for such complex, large-scale simulations, and fully exploits modern multi-core computer architectures. Its simple operational logic promises easy integration with all existing and upcoming MATSim functionality, and opens the door to more sophisticated approaches to large-scale, integrated transportation planning.

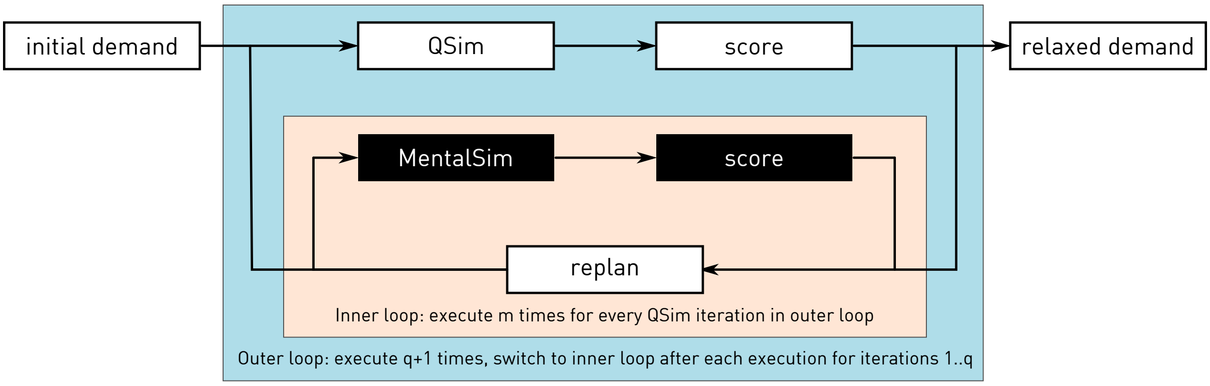


FIGURE 1 Illustration of the operational principle for mental simulation in the MATSim framework. The current framework is shown by the white boxes; the logic behind mental simulation is to introduce an extra feedback loop (inner loop).

INTRODUCTION

Humanity's dominance over nature is largely due to our ability to constantly adapt to challenges in our environment within a single lifetime, rather than across evolutionary time. Based on knowledge about what's happening in the world, we can evaluate our intended actions in our minds before executing them in the real world, thus avoiding the wasteful trial-and-error approach that characterizes evolutionary adaptation. In this paper, an analogy of this concept is applied to a co-evolutionary, multi-agent transport simulation system to design a flexible strategy for improving simulation times.

The Multi-Agent Transport Simulation toolkit, MATSim *(1)*, simulates agent learning by iteratively executing agent day plans of activities and connecting trips in a mobility simulation (currently a queue simulation, or QSim). After each iteration plans are mutated across a number of choice dimensions and poorly performing plans are discarded. Agent behavior therefore improves only across 'generations' through trial-and-error, analogous to evolutionary adaptation.

Mobility simulations are time-consuming, as the interactions of all agents participating in the transportation network are executed. Their performance also does not scale well with the parallel-processing capabilities offered by modern computer architectures. This is due to the synchronization required between computational threads when transferring agents between portions of the network handled by different computational cores. In fact, the complexity introduced by this synchronization makes the single-threaded version of QSim much more reliable and easy to maintain and debug.

However, despite being time-consuming, QSim matches the dynamics of the real transport system quite well, and produces various indexes of transport system performance, such as network link travel times and volumes through the course of the day, and waiting times at public transport stops. The method proposed in this paper uses this information to evaluate and adapt plans before execution, resulting in improved agent learning rates and system convergence time.

Multi-agent transport simulation

MATSim simulates the traffic produced in a transportation network by agents pursuing daily schedules of activities (plans) separated in time and space. Its principle of operation is shown by the white boxes in Figure 1. The system is fed with an initial demand of agent plans that are repeatedly executed in a traffic simulation (QSim). After each QSim run, plan performance is evaluated using a utility-based scoring function. Then, agent plans are mutated along a number of choice dimensions, such as activity start times and durations, route choice, trip transport mode, activity location choice, etc., to produce new plans for execution in the following traffic simulation. With increasing iterations, the number of plans in each agent's memory grows up to a limiting number, following which poorly performing plans are discarded. Consequently, the average score of plans improves with increasing iterations, until a steady state is reached where plan mutations produce only marginal changes in score.

Clearly, this approach is analogous to that of evolution by natural selection, where a genotype (plan) is expressed as a phenotype in the physical environment (agent in traffic) *(2, 3, 4)*. The success of the phenotype determines the longevity of genes in the genotype (combinations of plan elements, such as mode choice, activity timing and location, that become more-or-less stable features across generations).

Mutation approaches

In Figure 1, the 'replan' action represents the mutations producing evolutionary change. Replanning is done through the chaining of modules into strategies. An example strategy might be:

> Draw 10% of agents, [randomly select a previously executed plan from memory for each agent and make a copy of it], [adjust the start time and duration for each activity in the plan by a random number of seconds less than half an hour], [find the quickest network route between activities based on quarter-hourly travel times from the previous iteration], mark these plans as ready for execution.
> For all remaining agents, [select a previously executed plan from memory based on plan score], mark these plans as ready for execution.

In this example, each set of brackets denotes a replanning module. Some modules are merely plan selectors, and do not mutate plans. Other modules can be divided into *random-response* and *best-response* mutators. For the strategy set out above, the start time and duration adjustment module is random-response, while the router is a best-response replanning module, using a Dijkstra algorithm to find the lowest cost route through the network at a given time of day.

Best-response vs. random-response replanning

Best-response modules, though computationally burdensome, reduce total simulation time by exploiting traffic information from the previous iteration, while random-response modules rely on the trial-and-error of the evolutionary algorithm to produce better plans across many iterations.

More complex best-response modules have been developed that explore multiple dimensions of the agent decision space, in order to dramatically reduce the number of iterations until convergence (e.g. *5, 6, 7)*. In general, these modules apply a metaheuristic approach to explore the solution space, ideally evaluating solution quality using the same utility-based scoring function selected for the scoring step of the MATSim framework.

Such monolithic replanning modules have a number of disadvantages. Firstly, they are purpose-built; if a scenario element is not included in the module, its influence is not considered in the solution. For instance, suppose `modx`, a time-and-mode optimizing module, consistently finds that the best departure time for an agent is 7 am, by car, just when the congestion pricing starts on the highway connecting that agent to work. If `modx` does not consider road-pricing in its design, the resulting plan will be sub-optimal, as the router will, say, find a lower-cost but slower route to work for the given departure time. A more favorable possible alternative, e.g. departing earlier to avoid the road pricing, is unlikely to be found, as `modx` optimizes one sub-problem and the router another.

As the feature set of MATSim grows with time, these modules therefore become obsolete, and require significant re-design to remain relevant. Which brings us to the second problem; due to their design complexity, best-response replanning modules are harder to maintain and integrate with new functionalities than simple random-response modules.

MATSim is an open-source project, developed and maintained by an ever-changing, international group of volunteers, mainly researchers, who generally rely on public funding for projects. That funding tends to be channeled toward the development of new functionality. Maintenance and integration is therefore focused on core functionality relevant to a project; if the design of a module is not readily understandable, it is likely to remain obsolete or useable in only limited cases for extended periods of time.

A final, finer point of contention on the use of best-response replanning modules, is that the plans they produce do not reflect the lack of perfect knowledge and compromise that their real-world counterparts exhibit. They race toward the optimum in each iteration, quickly converging to a highly coordinated system, impervious to the churn from minor changes in behavior occurring day-to-day in the real transport system.

Research idea

To summarize; from a systems engineering point of view, simpler, random response modules are more favorable as they are easier to maintain and integrate. From a transport planning point of view, simpler modules can be chained together to solve more complex problems, albeit at the cost of longer simulation runs due to the expanding search space. Best-response modules use information feedback from the last simulation run to produce plans that are more likely to perform well in the mobility simulation. Plans produced by random-response modules are as likely to perform poorly as otherwise, and it's left to the evolutionary selection process to decide their fate.

The purpose of this investigation is simply to introduce an information feedback mechanism into the MATSim framework, where the quality of plans produced by the random response modules are evaluated before execution. Plan quality is evaluated by feeding expected travel times (calculated from system metrics produced in the previous simulation) and activity performance times into the same scoring function as is used for the mobility simulation. This process is similar to the 'mental simulation' humans perform when imagining a daily schedule being acted out, given their knowledge of the transport system and the demands of their activity commitments. Therefore we shall refer to this mechanism as mental simulation, and to the set of operational modules as MentalSim.

The expectation is not only that mental simulation should produce improved convergence rates for any combination of random-response modules, but that the relative simplicity of the operational principle should make it easy to maintain and integrate with new functionalities introduced to the MATSim system.

Related work

The idea of predicting the outcome of actions through learning and feedback between the mental and physical domains is not new to transport simulation *(8, 9)*. A multi-level feedback loop, using transport system metrics on one level to inform the location decisions of households and firms, and individual learning on the other as agents respond to resulting changes in demand patterns, has also been the subject of recent investigation *(10)*. Also, UrbanSim *(11)* can use so-called "skims" which means to use a previous output of the assignment model in order to avoid running it – this implies the assumption that travel speeds in the transport system remain the same over a couple of UrbanSim iterations. Indeed, using transport system metrics to inform agent learning is the basis of operation of the best-response replanning modules discussed above.

What makes the contribution unique in this case, however, is that the simplicity of the mental simulation approach should produce *extensible* accelerated agent learning. It should allow arbitrary combinations of current and future scenario elements and replanning modules to investigate complex transportation questions, such as 'what would the influence of a morning toll be on public transport ridership and household ride-sharing in a large, demographically diverse metropolitan area?'; and produce answers in short enough time spans as to prove useful to decision-makers. In addition, the present paper performs systematic investigations concerning the convergence gains with the new approach with respect to wall clock time.

The following section provides details on the design of the mechanism, given the framework of MATSim. This is followed by some initial results where the system was tested for speed and solution quality on a scenario for Zurich, Switzerland. Concluding remarks summarize what has been learned, and highlight further possible improvements.

METHOD

Figure 1 illustrates the principle behind the mental simulation idea. The system is fed with an initial demand of agent plans, which get executed in a queue simulation (QSim). Plans are scored and sent to the replanning modules. An inner loop is then executed for a number of iterations, where new plans are executed in MentalSim, scored, and sent for replanning. After, say, m such iterations, plans are selected again for execution in QSim, scored, and the inner loop repeats again for another m iterations. The outer loop repeats q times, then terminates with a final QSim and scoring step, leaving a relaxed demand.

MATSim events

In MATSim, the queue simulation produces time-stamped, atomic units of information called events, which describe what is happening to each agent at all times. Trawling through these events, it is possible to recontruct every agent's trajectory through the transportation system.

As the simulation proceeds, these events are sent to a number of event listeners. These are software modules that run in parallel to the simulation. Their purpose is two-fold: to aggregate and interpret events to calculate agent plan scores on the fly; and to build a dynamic record of transport system performance metrics, e.g. link travel times and volumes, public transport ridership, toll paid, emissions produced, etc.

In the default mode of operation, these metrics are used to route new plans through the network, then they are either written out to disk and reset before starting a new iteration, or kept for a number of iterations to perform averaging. This persistence of information capability is exploited by the MentalSim module.

MentalSim design

The MentalSim module is a bare-bone mobility simulator that reads through an agent plan and fires appropriate time-stamped events. What kind of events it generates is determined by the scoring function used for the simulation. The default scoring function, derived from Charypar and Nagel *(12)*, in its simplest form, rewards the performing of activities, and penalizes travel and arriving late for activities. It listens for actvity start and end events, as well as travel start and end events.

To illustrate its operation, suppose a simple home-work-home agent plan. MentalSim reads the home activity departure time from the plan and fires an activity end event with the departure time as its timestamp. It reads the mode of travel and fires a travel start event for that mode of travel, timestamped a second later. It then asks for the travel time between the origin and destination based on the route given in the plan from the appropriate data provider; suitably called a travel time calculator. This travel time is calculated from the recorded travel times during that time of day from the last QSim iteration. MentalSim adds this time to the last timestamp, and fires a travel end event and work activity start event. The process repeats for all activities and trips in the plan.

If a more complex scoring function is used, say one that considers road pricing, MentalSim will also fire link enter and leave events with the appropriate timestamps for all links in the agent's route. The toll costs will then be included if the agent traveled across tolled links or entered a road pricing cordon.

No interaction occurs between agents in MentalSim, so it can fully exploit modern multi-core computer architectures, as no synchronization between threads is required and access to data structures outside a MentalSim thread is read-only. Load balancing is simple; plans scheduled for execution are simply divided up between threads. Event processing is also completely parallelized, as are re-planning operations.

As there is no interaction between agents in mentalsim, it makes sense to only simulate newly generated plans, that do not have a score associated with them yet. This cuts down on the expected computational load even further, as each iteration only generates a small number of new plans, depending on the rate of replanning prescribed by the replanning strategy.

MentalSim simulation modes

The MentalSim module was designed to operate in two modes; *global* mode and *subset* mode.

Global mode

MentalSim simply replaces the regular QSim for m regular iterations. It thus operates on all newly generated plans passed to it by the replanning modules, with replanning applied across the entire agent population through random selection of agents at the end of each iteration.

Subset mode

Select agents at random, say 10% of the population, and only operate on this subset in the inner loop. Once the subset of agents has been selected, a single QSim-executed plan is randomly selected and copied for each one. The agents' QSim memory is then preserved, to be restored to them when they exit the inner loop. Each agent is then passed to the MentalSim inner loop with only the single copied plan in memory.

The agents then build up a series of MentalSim-executed plans until they reach their memory limit, following which they discard poorly performing plans until the maximum number of MentalSim iterations have been reached. The memory limit used in the inner loop can be set to a different size than that used in the outer loop, in order to expand the search space. Then, a single plan is selected for each agent and passed back from the MentalSim inner loop to the QSim outer loop. All other MentalSim plans except the selected plan are discarded, and each agent's stock of QSim executed plans is restored to memory. The selected MentalSim plan is then executed with all other agent plans in a QSim iteration.

The reasoning behind the two modes of operation is that agents might become 'delusional' from having their QSim-executed plans mixed up with MentalSim plans. QSim-executed plan scores reflect the 'real' performance of a plan, while MentalSim plan scores only reflect the expected performance of that plan. With the general mode of operation, high enough rates of replanning and a large number of MentalSim iterations in the inner loop will result in agents very quickly having their memory filled with plans that have only been executed in MentalSim.

For example, suppose a simulation where 24 MentalSim iterations are run for every QSim iteration, where the rate of replanning is 30% and where the maximum number of plans an agent can hold in memory is 5. From the binomial distribution, we know that the expected number of agents with less than 5 plans generated during the 24 MentalSim iterations comes to only 11.1%. This might hold serious consequences for the quality of our solution, as 'true' information on plan performance is lost.

The subset strategy would clearly prevent this situation, by discarding MentalSim plans and preserving the original QSim-executed plans of each agent selected for mental simulation. The only plans that persist in memory will be in the outer loop, and will always have been executed and scored in a QSim iteration.

EXPERIMENTAL SETUP

MentalSim performance was tested in a set of simulation experiments, with the following aim:

1. Test MentalSim behavior in terms of convergence rate when varying parameters such as replanning rates, QSim:MentalSim iteration ratio, and modes of operation;
2. Test MentalSim's computational performance and solution quality compared to a baseline, QSim-only simulation run of a large-scale scenario with a complex replanning strategy consisting of many modules.

Simulation scenario

We used the MATSim development scenario of Swiss car traffic crossing or operating within a 30km radius circle around Bellevue, Zurich, as used in many studies *(*see *6).* We use the same 10% sample from that study, as well as the facility information by the same authors. This scenario is supplemented with an arbitrary morning toll on all links exceeding a capacity of 4,000 vehicles per hour. The following re-planning modules were used in equal measure, with the total replanning rate (proportion of agents replanned) varied as part of the experimental setup:

1. activity start time and duration adjustment;
2. re-routing using travel times from the previous iteration;
3. subtour mode choice – switches the mode of transport of a randomly selected subtour to car/public transport given that, for this scenario, all agents have access to cars;

4. secondary activity location choice: shopping and leisure activities are switched to a randomly chosen location from a set of qualifying facilities;

Public transport is not explicitly simulated. Instead, trips using public transport are 'teleported' from origin to destination with a travel time that is twice that of the free speed shortest path *(13)*.

RESULTS

Characterizing solution state

MATSim employs stochasticity at various points in a simulation run, such as agent selection for different modes of replanning, plan selection for execution, and transition rules at intersections during a queue simulation. In order to make runs repeatable, a seed number is set for the Java random number generator at the beginning of a simulation run.

In our experiments, we used the same random seed for all simulation runs, except a baseline QSim-only run. Then, when comparing the solutions of two QSim-runs with the same parameters except random seed, we have an indication of the minimum deviation we can expect between any two runs of the same scenario.

The baseline against which simulation runs were compared was selected as the simulation state obtained by running the scenario for 101 iterations with QSim only, at an overall replanning rate of 30% per iteration, with a maximum agent memory of 5 plans per agent.

Three measures were used to characterize solution state for comparison against the baseline:

Average executed QSim score

We take the 101^{st} iteration score of 175.4 for the baseline run as a reference value. For all other runs, the first QSim iteration where the score was greater or equal to this value was selected and the rest of the measures were calculated.

Departure profile RMSD

Agent departures are compared at 5 minute intervals for the simulated day. We take the root mean square deviation (RMSD) from the baseline departures as an indication of how similar a simulation state is to the baseline in terms of activity timing.

Mode share

We also compare car mode share (number of car trips / total number of trips) for the large-scale scenario, as mode choice is one of the dimensions included in the replanning strategy.

The minimum value (e.g. *reference value*) for each measure is that of the case where only the random number seed differs from the baseline setup. We refer to this case as the *reference case*.

Varying QSim:MentalSim ratio

When keeping the replanning rate constant, we found that increasing the number of MentalSim iterations between QSim iterations increases the rate of convergence, as can be seen from Figure 2. In this figure, we compare the utility vs. number of QSim iterations for two QSim:MentalSim

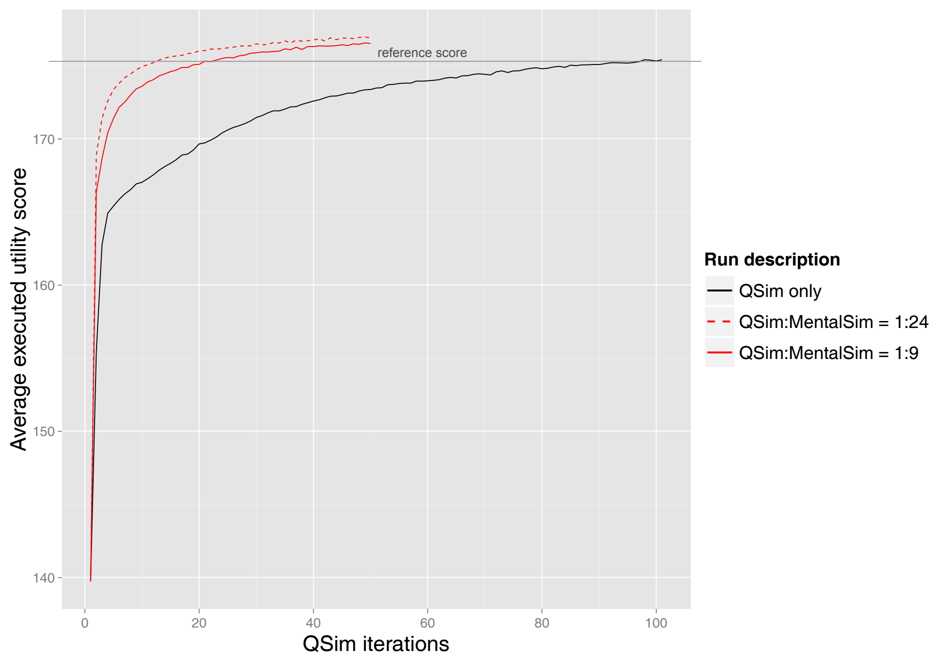


FIGURE 2 Average executed score versus QSim iterations for two ratios of QSim:MentalSim (red), compared with a reference QSim-only run.

ratios (red) against the reference case(black).

In general, for a given intermediate utility score, the number of QSim iterations required to achieve that score is inversely proportional to the total number of iterations executed during the simulation, e.g. QSim + MentalSim iterations. So, for the given comparison, we reach the utility of 175.4 in 101 QSim + 0 MentalSim = 101 iterations for the QSim only run, 24 QSim + (23×9) MentalSim = 231 iterations for the QSim:MentalSim = 1 : 9 run, and 14 QSim + (13×24) MentalSim = 326 iterations for the QSim:MentalSim = 1 : 24 run.

Global vs. subset mode

Score evolution

It was found that the executed score evolution over iterations initially proceeds slightly slower for subset mode than global mode, if the replanning rate and QSim:MentalSim ratio is held constant. Also, the spread of plan scores held in memory differs for the first few dozen iterations, as can be seen in Figure 3. The ribbons in that figure indicate the difference between the average best score and average worst score in agent memory. Subset mode maintains a larger diversity in plans for the first few dozen iterations, but then converges to the same spread as the global mode of operation.

The difference between the spread and rate of convergence for the two modes of operation was found to increase as the number of MentalSim iterations in the inner loop increases. This probably happens because QSim-executed scores are retained for the subset mode operation, while the global mode of operation does not treat QSim-executed plans differently from MentalSim-executed plans. In global mode, poorly performing plans are discarded the moment the agent's memory limit is reached, be that during a MentalSim or QSim iteration. In subset mode, the agents' sets of plans grow more gradually with increasing iterations, as the number of

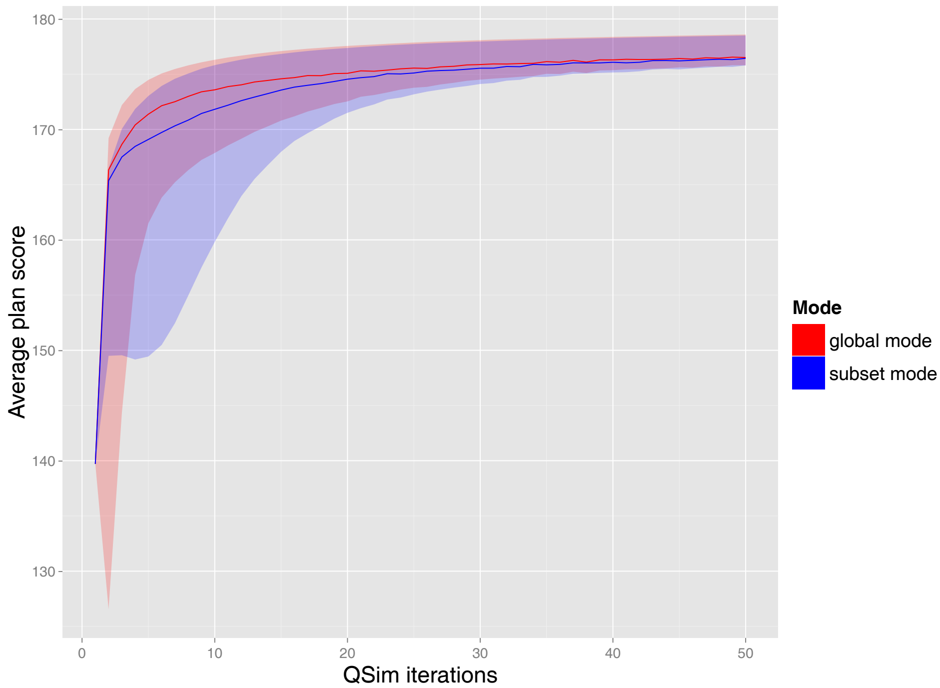
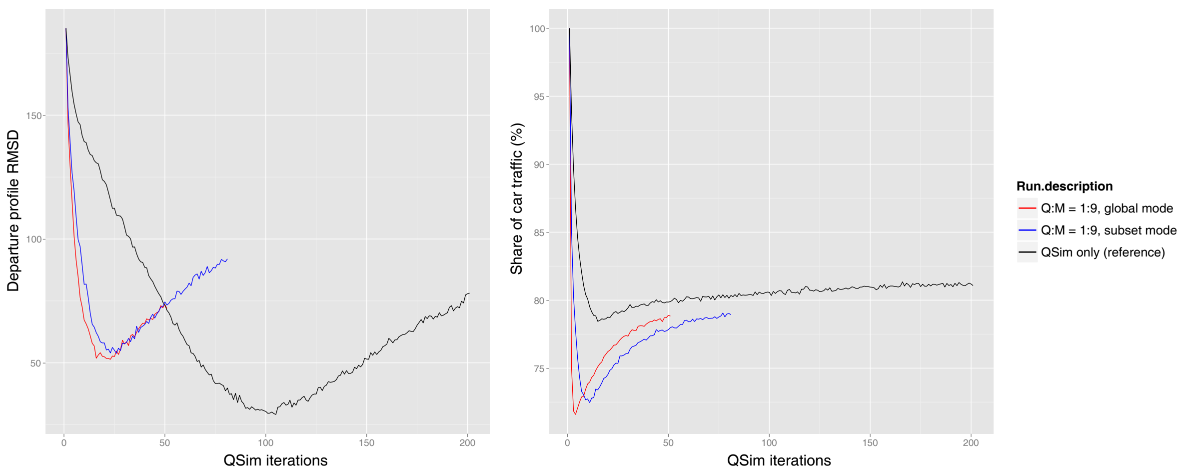


FIGURE 3 Comparison of average score evolution in the large-scale scenario for global mode (red) and subset mode (blue), QSim:MentalSim = 1:9 (0.3 replanning rate). Translucent ribbons indicate the spread of plan scores in agent's memory.



(a) Departure profile RMSD

(b) Car mode share

FIGURE 4 Departure profile RMSD and car mode share comparison for the two runs in Figure 3 against the reference run.

plans held in memory only increase with QSim iterations.

Solution state

Departure profile RMSD and mode share for both modes of operation are compared against the reference run in Figure 4. Here we can see that both modes of operation reach their minimum

RMSD value at the iteration where their score equals the reference score of 175.4. However both values are significantly larger than the minimum attained by the reference run at 101 iterations.

The reason for this difference is probably due to the different mode shares produced by the MentalSim runs when compared to the reference run (Figure 4b). The swing towards public transport is much larger for the MentalSim runs than for the reference run. The routing and travel time of public transport is independent of network conditions for our simulations, as public transport was not explicitly simulated in order to save simulation time. The mental simulation gives many more agents the chance to consider that during the initial iterations, with lots of car congestion, public transit is an attractive alternative. An agent's optimal departure time with public transit is, however, different from the same agent's optimal departure time with car. In the long run mode share seems to converge towards the reference value.

In a simpler experiment investigating this overshoot effect, it was found that executing a few QSim-only runs before switching on the inner loop reduces the effect. During the first 5-10 iterations (depending on the rate of re-routing) the largest congestion effects are rapidly reduced by the router, with a proportionately rapid improvement in utility. Then the influence of the router on utility diminishes, and the long process of improvement through random response and selection takes over (the flatter part of the utility curve).

Performance test

It was found that, even though the average QSim executed score improves faster with increasing iterations for all MentalSim-enabled runs, plotting these values versus wall clock time show a radically different indication of performance, as can be seen in Figure 5. It compares the influence of QSim:MentalSim ratio, number of computational cores and replanning rate on simulation (wall clock) time. Here it is clear that the MentalSim strategy is only effective as the number of cores committed to the simulation is increased.

Figure 6 shows the wall clock time it takes, with different set-ups, to reach a certain level of convergence, as described earlier. One notices that the computing (= wall clock) time for replanning scales inversely linear in the number of cores. That is, with an ever growing number of cores, that number will shrink more and more. This is due to the computational (and conceptual) decoupling of the replanning: every agent replans for herself. Second, one notices that replacing most of the regular QSim runs with MentalSim runs, as discussed in this paper, results in significantly reduced QSim contributions to the overall wall clock time, even if one counts in the additional time for the MentalSim and the additional overhead. At this point, it was possible to reduce the computing time by more than a factor of two, when comparing the 16 core results from the regular approach to the fastest version of using the 16 core machine with MentalSim.

An interesting result here is that lowering the replanning rate, while increasing the number of MentalSim iterations in the inner loop gives the best overall performance, with its most significant component being time spent on overhead operations. The reasons for this improved performance in comparison to the other 16 core MentalSim run will be explored in the next section.

DISCUSSION

The mental simulation approach was designed to be consistent with the pre-existing simulation logic of MATSim, and it appears to produce comparable results. In all cases, using the mental

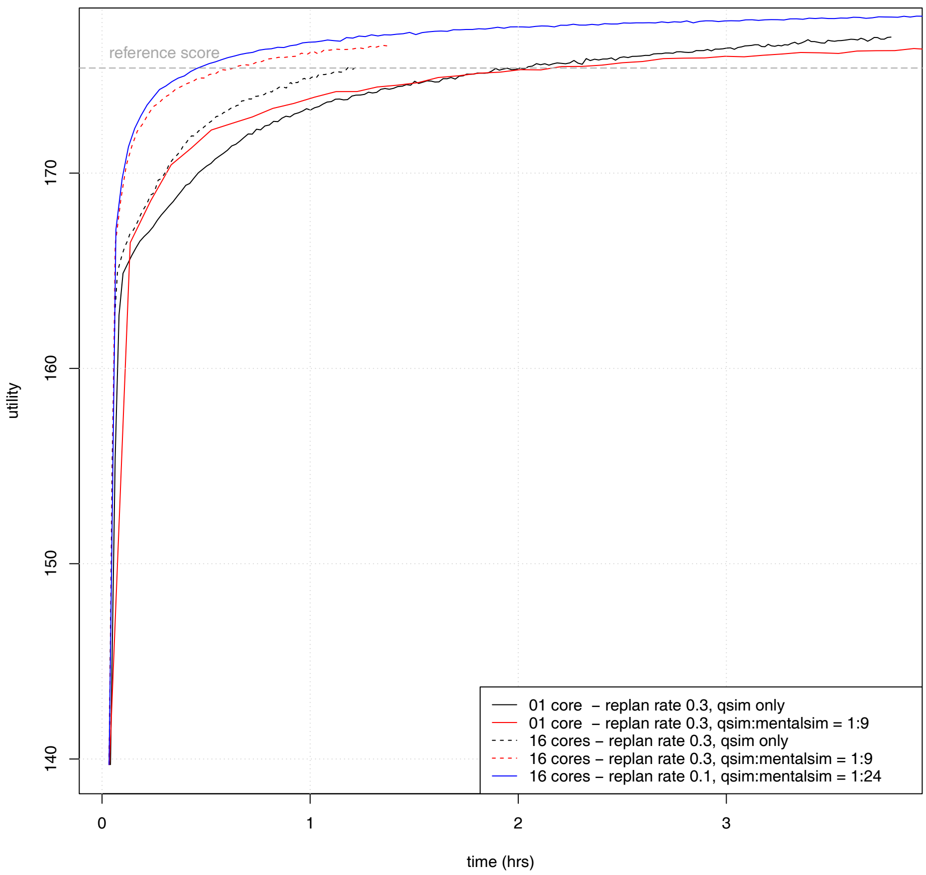


FIGURE 5 Score evolution vs time for large-scale scenario, comparing the influence of QSim:MentalSim ratio, number of computational cores and replanning rate (MentalSim module operating in global mode only).

simulation approach reduces the number of time-consuming QSim iterations required to achieve a given average plan score.

Global vs. subset mode

The reasoning behind subset mode was that agents could become 'delusional' by having plans with scores only from MentalSim. This 'delusion' was expected to manifest itself by too many of them changing their travel routes and timing, with no interaction to give congestion, with the result of just moving the congestion around in the network. This does not seem to be the case for the scenario used in these experiments.

An alternative reasoning might be that if MentalSim produces reasonably accurate events from good travel time information, and the number of MentalSim iterations in the inner loop are kept to a reasonable number for the given replanning rate, then global mode has the effect of keeping all plan scores in an agent's memory at likely values for the current QSim iteration, rather than some earlier simulation state.

For the subset strategy, the number of plans in an agents' memory grows with increasing QSim iterations at the same expected rate as for a QSim-only run with the same replanning rate.

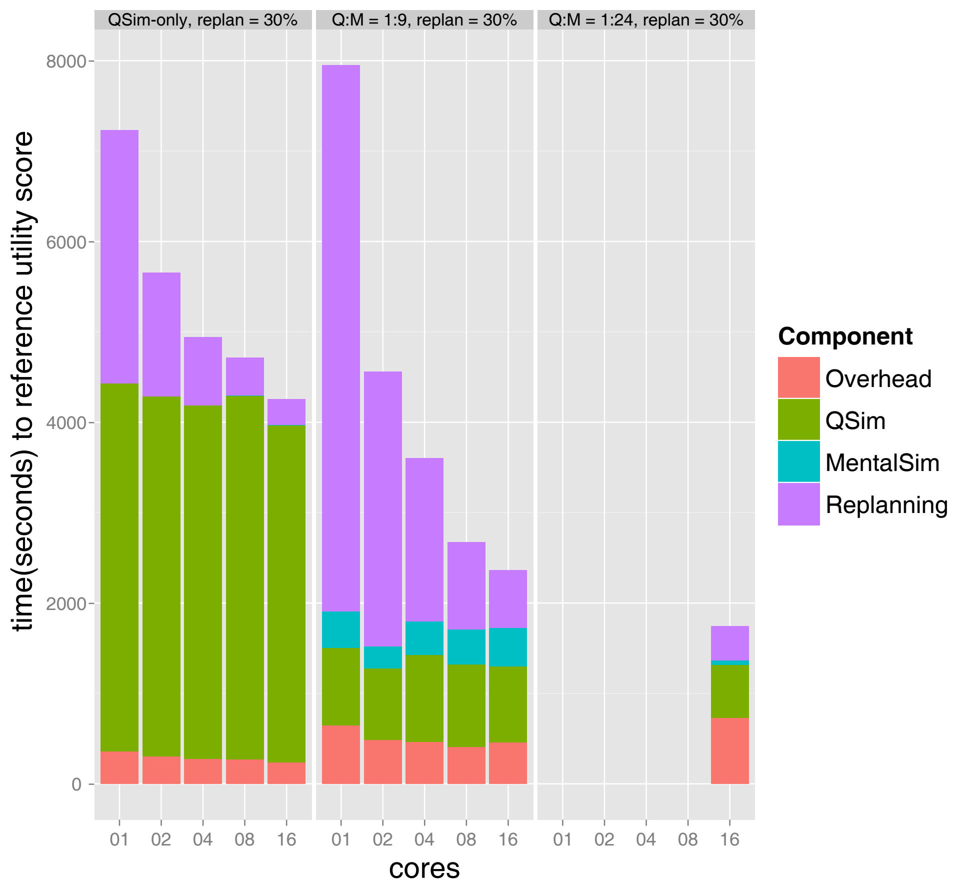


FIGURE 6 Computation time contributions vs number of cores for QSim only (0.3 replanning rate), QSim:MentalSim = 1:9 (0.3 replanning rate) and QSim:MentalSim = 1:24 (0.1 replanning rate) at the reference score (grey line in Figure 5).

The only difference is that the newly generated plans are expected to perform better than those generated by the QSim-only run, because they have been repeatedly adjusted and evaluated based on the last QSim's travel times. But, supposing an agent memory of five plans, it will still take any agent at least five QSim iterations to reach their memory limit. It also means that, at any time, at least one plan in the agent's memory will have been scored on information that is at least five QSim iterations old.

For global mode, agent memories grow much more rapidly with increasing QSim iterations, and a large proportion of their plans will have been evaluated using the most recent travel time information. Within a single generation of the outer loop, a large number of agents can be expected to have their memories filled. Consequently, for any given agent, it is unlikely that any given plan will have been scored on information that is more than one or two QSim iterations old.

As for 'delusion' — it appears that, as long as information is dependable, agent responses are, for most part, reasonable.

Performance

As expected, the mental simulation approach scales well with an increasing number of cores. Our experiments revealed that the interplay of replanning rate and number of MentalSim iterations in the inner loop have an important influence on convergence rate. Having a relatively low replanning rate with a higher number of MentalSim iterations in the inner loop produces the target score in less QSim iterations and less wallclock time.

At first glance, this is a surprising result, because the expected number of plans generated from one QSim iteration to the next is comparable for the two 16-core mental simulation runs in Figure 6. The first run has a replanning rate of 0.3 and QSim:MentalSim ratio of 1:9. Consequently, in 1+9 iterations, the expected number of new plans produced per agent comes to 3, with a standard deviation of 1.44. In comparison, the second run has a replanning rate of 0.1 and QSim:MentalSim ratio of 1:24, so in 1+24 iterations, it produces only 2.5 new plans per agent on average, with a standard deviation of 1.5.

The reason for the quicker convergence is probably due to the number of combinations of replanning modules that can act on any given plan in successive inner loop iterations for the second case. Even if any given combination has only a small chance of occurring; if it is favorable, it will be retained.

The expected value calculation also shows why the total replanning time of the second run is significantly less than the first: In total, it produces 16.7% less plans per outer loop cycle. It suffers, however, from an increased overhead due to a larger total number of iterations.

CONCLUSION AND OUTLOOK

The mental simulation approach should prove useful in reducing simulation times for most applications of MATSim. Its simple design should make it easy to maintain as MATSim functionality is extended. In this paper, it has been shown to work well with an extensive list of existing MATSim capabilities.

Reducing overhead

The next development step will be to integrate MentalSim into the core MATSim framework, and reduce the number of overhead operations occurring between MentalSim iterations. These operations include the calculation and writing out to disk of departure profiles, travel times and log files, and are a significant contributor to total wallclock time, limiting the improvement gains from paralellization.

Public transport

In this paper, public transport trips are not explicitly simulated in the QSim iterations, but instead teleported throught the network. Preliminary tests with mental simulation have shown promising results for scenarios that explicitly simulate public transport in the presence of private vehicle traffic *(*see *14)*, but further investigation is required.

Social network coordination and ride-sharing

The ultimate purpose of developing the mental simulation approach is to explore MATSim's capability to test integrated, complex scenarios. If solution spaces are huge if agents replan independently from each other, they become massively vast when one starts to consider the possibilities that open up when plans are coordinated within households and social networks. So far, only sub-problems of this integrated transport problem with social networks and ride sharing have been addressed in the MATSim development context, using best-response replanning modules employing complex metaheuristics. Mental simulation in combination with simple replanning modules will be investigated as an alternative to the best-response approach.

Parallel simulations

The present paper inserts the MentalSim so that it stays close to the pre-existing simulation logic. In global mode, the QSim is just replaced by the MentalSim; in subset mode, a sequence of MentalSim runs produce a single plan for selected agents, to be executed in the next QSim run. Even though performance gains are the result of the MentalSim module's capability to fully exploit parallel computation, the simulation logic is still serial.

Currently, the MATSim framework has all agent plans evolving from a single initial condition; the initial demand. The evolutionary logic might preclude certain plans from ever evolving. Consider for instance, an agent whose initial plan is close to a local optimum for being car-only. Assume that the global optimum for this agent is actually a public transport plan, with a markedly different temporal structure to that of the optimal car plan. A random-response switch to public transport for one or more trips produces worse performing plans given the car plan's temporal structure, and are quickly discarded as the agent's memory limit is reached. Consequently, the agent remains stuck at the local optimum.

Once the MentalSim capability is integrated into MATSim, however, this opens the door to more sophisticated approaches. For example, an agent could mentally optimize a public transit plan over many mental iterations and only then compare it to an already optimized car plan.

Also, such optimizations could run in parallel when computing resources are under-utilized during QSim runs (recall that the queue simulation is more reliable and easier to maintain in single-threaded mode).

Extending this idea, one could imagine a situation where several instances of an agent population are run in parallel, each with different initial conditions. Information on plan performance across different runs could then be monitored by a supervisory process; taking the best plans from these parallel threads and putting them together in a primary simulation.

ACKNOWLEDGEMENTS

The authors thank Dominik Grether from VSP, TU Berlin for his technical assistance during development of the MentalSim modules. This work is funded, in part, by a grant from the National Research Fund of Singapore.

REFERENCES

1. Balmer, M., M. Rieser, K. Meister, D. Charypar, N. Lefebvre and K. Nagel (2009) MATSim-T: architecture and simulation times, *Multi-Agent Systems for Traffic and Transportation Engineering*, 57–78.

2. Goldberg, D. (1989) *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley.

3. Hraber, P., T. Jones and S. Forrest (1994) The ecology of Echo, *Artificial Life*, **3** (3) 165–190.

4. Balmer, M. (2007) Travel demand modeling for multi-agent transport simulations: Algorithms and systems, Ph.D. Thesis, Swiss Federal Institute of Technology (ETH) Zürich, Switzerland.

5. Meister, K., M. Balmer, K. Axhausen and K. Nagel (2006) planomat: A comprehensive scheduler for a large-scale multi-agent transportation simulation, paper presented at the *6th Swiss Transport Research Conference*, Monte Verita, Ascona.

6. Horni, A., D. M. Scott, M. Balmer and K. W. Axhausen (2009) Location choice modeling for shopping and leisure activities with MATSim, *Transportation Research Record: Journal of the Transportation Research Board*, **2135** (-1) 87–95.

7. Dubernet, T. and K. Axhausen (2012) Including joint trips in a multi-agent transport simulation, paper presented at the *6th Swiss Transport Research Conference*, Monte Verita, Ascona.

8. Arentze, T. and H. Timmermans (2001) Inductive learning approach to evolutionary decision processes in activity-scheduling behavior: theory and numerical experiments, *Transportation Research Record: Journal of the Transportation Research Board*, **1752** (-1) 1–7.

9. Rieser, M., K. Nagel, U. Beuck, M. Balmer and J. Rümenapp (2007) Truly agent-oriented coupling of an activity-based demand generation with a multi-agent traffic simulation, *Transportation Research Record*, **2021**, 10–17.

10. Nicolai, T. W., L. Wang, K. Nagel and P. Waddell (2011) Coupling an urban simulation model with a travel model – A first sensitivity test, paper presented at the *Computers in Urban Planning and Urban Management (CUPUM)*, Lake Louise, Canada. Also VSP WP 11-07, see www.vsp.tu-berlin.de/publications.

11. Waddell, P., A. Borning, M. Noth, N. Freier, M. Becke and G. Ulfarsson (2003) Microsimulation of urban development and location choices: Design and implementation of UrbanSim, *Networks and Spatial Economics*, **3** (1) 43–67.

12. Charypar, D. and K. Nagel (2005) Generating complete all-day activity plans with genetic algorithms, *Transportation*, **32** (4) 369–397, ISSN 0049-4488.

13. Rieser, M., D. Grether and K. Nagel (2009) Adding mode choice to a multi-agent transport simulation, *Transportation Research Record: Travel Demand Forecasting 2009*, **2132**, 50–58.

14. Rieser, M. and K. Nagel (2009) Combined agent-based simulation of private car traffic and transit, paper presented at the *Proceedings of The 12th Conference of the International Association for Travel Behaviour Research (IATBR)*, Jaipur, India. Also VSP WP 09-11, see www.vsp.tu-berlin.de/publications.