Increased convergence rates in multiagent transport simulations with pseudosimulation

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

Publication Date:
2013

Permanent Link:
https://doi.org/10.3929/ethz-b-000070377

Originally published in:
Transportation Research Record 2343, http://doi.org/10.3141/2343-09

Rights / License:
In Copyright - Non-Commercial Use Permitted
Increased Convergence Rates in Multiagent Transport Simulations with Pseudosimulation

Pieter J. Fourie, Johannes Illenberger, and Kai Nagel

A multimodeling approach to large-scale, activity-based, multiagent simulation of travel demand is introduced. MATSIM is a fully activity-based transport simulation. Its greatest current performance limitation is the network loading simulation, currently a queue simulation (QSim). QSim is iteratively executed for the entire agent population for evaluating the effects of random mutations on the activity plans of a fraction of the population. After each QSim, poorly performing plans are discarded, good plans are kept, and the agents slowly learn what works best for their individual activity needs. In the application presented, the system periodically replaces QSim for a number of iterations with a simplified pseudosimulation that runs approximately two orders of magnitude faster. The pseudosimulation uses travel time information from the preceding QSim iteration to estimate how well an agent day plan might perform. Repeated iterations of the pseudosimulation produce better-performing plans in a short time. These plans are passed to the QSim for updating of network travel time information, and the process repeats. The technique is tested in a scenario for Zurich, Switzerland, and incorporates mode choice, road pricing, secondary activity location choice, activity timing adjustment, and dynamic routing. The technique dramatically improves convergence rates for such complex, large-scale simulations and fully exploits modern multicore computer architectures.

Transportation is mostly a derived demand, produced by the need for individuals to pursue schedules of activities, separated in time and space. Individuals can adjust their activity schedules to various degrees in response to experienced transport system performance, for example, by changing the timing and location of activities, changing routes and modes of travel, avoiding toll, or sharing rides. Individuals do not have complete freedom: they are constrained by demographics, material means, household, work and other commitments, and physical needs (1). No closed-form analytical solution exists that captures the full complexity and dynamics of the transportation system and the travelers that participate and interact within it. Therefore, simulation-based techniques are becoming increasingly prominent as an alternative to aggregate, analytical methods (2–5).

The multiagent transport simulation toolkit (MATSIM) is a fully activity-based transport demand modeling system (6). It produces a full description of transport demand in the form of a day plan of activities and connecting trips for every individual (agent) in a large-scale urban scenario and of the resulting congestion patterns and network performance measures. It simulates the interaction between supply and demand by iteratively executing agent day plans in a mobility simulation, currently a queue simulation (QSim), also called network loading. Executed plans are scored, being rewarded for time spent at activities and penalized for time spent traveling or arriving late for activities. Plans are replicated and mutated across a number of choice dimensions, and poorly performing plans are discarded. Agent behavior therefore adapts to transport system performance across generations (iterations) through trial and error, analogous to evolutionary adaptation (7, 8). The overall approach is thus one of coevolution (9–11).

Mobility simulations are time-consuming, as the interactions of all agents participating in the transportation network are executed for every second in a 24-h simulated day. Plan mutation is comparatively fast (if mutation is simple and random), even when mutation occurs across many dimensions. However, as the number of choice dimensions in the scenario increases, the number of iterations and thus the number of mobility simulation runs required to explore the solution space increases. Conversely, the impact of random changes to day plans on the agents’ and thus the transport system’s performance rapidly diminishes with increasing iterations; therefore much time is spent on mobility simulation with diminishing returns for the rate of system evolution.

Compounding the problem is that it is relatively more challenging to gain performance from modern multicore computer architectures in the case of the mobility simulator design versus that of plan mutation algorithms. The synchronization required between computer threads in the mobility simulation typically produces diminishing returns in performance with increasing computation cores (12).

There is a growing need to integrate existing and emerging model capabilities such as within-household interaction and coordination (13, 14, 4), ridesharing (15), social network interaction (16, 17), complex mode chaining, dynamic multimodal pricing (18), public transportation, secondary activity location selection (19, 20), spatially distributed parking capacity (21), and multiday, need-based activity modeling (22), which will produce a fully integrated activity-based transport model. As agent choice dimensions and constraints increase, the model solution space explodes in size and complexity. Consequently, the MATSIM solution process is expected to require a dramatic increase in the number of iterations so that it can effectively explore the high-dimensional solution space. The efficiency of the
solution process therefore needs improvement that preserves the flexibility designed into the current framework.

This paper introduces a flexible surrogate model for the MATSIM framework that will increase the rate of system evolution. Multi-modeling techniques are frequently used in simulation-based optimization, in which estimation of a simplified model of the system is based on a sample of simulated observations. The simplified surrogate model ideally takes a deterministic form that is computationally cheap to evaluate. A comprehensive review of surrogate-based techniques is available elsewhere (23).

In the presented application, the system periodically replaces the current QSim for a number of iterations with a simplified pseudosimulation (PSim) that runs approximately two orders of magnitude faster. PSim uses simulated link travel time observations generated in the preceding QSim iteration to produce an estimate of how well an agent day plan might perform, which allows the existing model framework to select and improve plans before executing them in a full queue simulation.

This paper investigates the multimodeling approach for performance, compatibility, and simulation state in comparison with the standard approach. The approach was applied to a large-scale scenario for Zurich, Switzerland, consisting of 67,239 agents traveling in a network of 60,518 links with a dynamic road-pricing scheme, allowing agents to simultaneously adjust mode choice, location choice for discretionary activities, activity timing, and travel route.

Application of multimodeling techniques in transport simulation has been sparse. Other than certain plan mutation modules developed for MATSIM, which can be seen as surrogate models informed by the queue simulation (24), only a single application, developed by Osorio and Bierlaire, is a good example of substituting full simulation with a surrogate model (25, 26). Thus, the work presented here is an informative addition to the dynamic traffic assignment literature and is intended to spark interest in other applications of a similar approach.

**LITERATURE REVIEW**

**Performance Challenge**

The biggest obstacle to further acceleration of iterated transportation simulations is network loading simulation. Computational performance improvements largely are made through multiple central processing units or multiple cores (computational nodes). Although the remaining tasks of a single iteration are straightforwardly distributed across multiple computational nodes, this is not true for network loading. The reason is that the physical system is tightly integrated: if a vehicle reacts to another vehicle in a typical reaction time of 1 s, neighboring simulation items should not go out of synchronization by more than 1 s. In this situation, spatial decomposition minimizes interactions most (27) and may allow somewhat longer synchronization delays when network links are sufficiently long (28). However, parallel implementations of network loading are difficult to keep stable in terms of software engineering, and making them more stable affects performance (12). The queue simulation used in MATSIM is no exception.

**Multiagent 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 QSim network loading. After each QSim run, plan performance is evaluated with a utility-based scoring function. Then, a number of agent plans are replicated and mutated along several choice dimensions, such as activity start times and durations, route choice, trip transport mode, and activity location choice, to produce new plans for execution in the following QSim iteration. With increasing iterations, the number of plans in each agent’s memory grows 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 in which plan mutations produce only marginal changes in score.

This approach is analogous to that of evolution by natural selection, in which a genotype (plan) is expressed as a phenotype in the physical environment (agent in traffic) (7, 10, 29). 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). Because individual plan evolution takes place in an ecology of the concurrent evolution of other agent plans, the overall approach is one of coevolution (9–11).
Mutation Approaches

In Figure 1, the replan action represents the mutations producing evolutionary change over repeated MATSIM iterations. Replanning is done through the chaining of replanning modules into strategies. An example strategy is as follows:

- 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 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, the start time and duration adjustment module is random response, and the router is a best-response replanning module that uses a Dijkstra algorithm to find the lowest-cost route through the network at a given time of day.

Best-Response Versus Random-Response Replanning

Best-response modules, although computationally burdensome, reduce total simulation time by exploiting traffic information from the previous iteration to produce a near-optimal solution to the mutation they are supposed to effect. In the preceding example, the Dijkstra router uses link travel times from the previous network loading to calculate an optimal lowest-cost route for each set of origin and destination points in the agent’s plan. In contrast, random-response modules rely on trial and error by the evolutionary algorithm to produce better plans across many iterations and do not guarantee any improvement in plan fitness.

More complex best-response modules have been developed that explore multiple dimensions of the agent decision space to dramatically reduce the number of iterations until convergence (24, 30, 31). Such monolithic replanning modules have a number of disadvantages. They are purpose built: if a scenario element is not included in the module, its influence is not considered in the solution. For instance, modX, a time-and-mode optimizing module, does not consider road pricing in its design. It consistently finds that the best departure time for an agent is 7:00 a.m. by car, just when the congestion pricing starts on the highway connecting that agent to work. The module then passes this plan to the Dijkstra router, which does consider road pricing in its design. Given the agent departure time, the router gives the plan a low-cost but slower route, avoiding a toll on the highway, and the agent is penalized for arriving late at work. A more favorable alternative plan (e.g., departing earlier to avoid road pricing) is unlikely to be found, because modX optimizes one subproblem and the router another.

As the feature set of MATSIM grows with time, these modules therefore become obsolete and require significant redesign for incorporation of new features and for remaining relevant. However, because of design complexity, best-response replanning modules are more difficult to maintain and integrate with new functionalities than are simple random-response modules.

Simulation-Based Optimization with Surrogate Models

It appears that only one true multimodeling approach, in which detailed simulation is used to estimate a simplified surrogate model, has been applied to traffic simulation. Osorio and Bierlaire combined the output from an AIMSUN dynamic traffic microsimulator with a surrogate model that analytically captures stationary queue distributions (26). They used this approach to perform simulation-based optimization of signaling plans in a congested network (25).

Their approach differs in two respects from the one presented here. First, their method does not use an agent-based paradigm. Second, they use information from the microsimulation to arrive at an analytical description of the network. In this paper, information from the queue simulation is used to create a simple lookup table of travel times through the course of the day for every link of the network. The system then uses this information to evaluate and adapt plans before execution in the queue simulation. It therefore relies on the same mechanism of learning through feedback that forms the basis of the MATSIM coevolutionary logic.

Feedback and Learning

The idea of predicting the outcome of actions through learning and feedback between the mental and physical domains is not new to transport simulation (32, 33). A multilevel feedback loop that uses 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 been the subject of recent investigation (34). Also, UrbanSim can use so-called skims, which allows use of a previous output of the assignment model to avoid running it (35); this implies that travel speeds in the transport system remain the same over a couple of UrbanSim iterations.

DESIGN

Figure 1 illustrates the principle behind the multimodeling approach. The system is fed with an initial demand of agent plans, which are executed in QSim. Plans are scored and sent to the replanning modules. An inner loop then runs for several iterations, in which new plans are executed in PSim, scored, and sent for replanning. After, say, p such iterations, plans are selected again for execution in QSim, scored, and the inner loop repeats again for another p iterations. The outer loop repeats q times, then terminates with a final QSim and scoring step, leaving a relaxed demand.

MATSIM Events

In MATSIM, QSim writes out time-stamped, atomic units of information called events, which describe what is happening to each agent at all times. By trawling through these events, it is possible to reconstruct every agent’s trajectory through the transportation system and the time they spent at various activity locations. For example, an agent travels from home to work in a small network. Her event stream might be as follows:

<event=actEnd time=21600 person=1 link=1 actType=home />
<event=departure time=21600 person=1 link=1 legMode=car />
<event=wait2link time=21609 person=1 link=1 vehicle=1 />
The XML code shows the simulation time in seconds for each event. This agent (where ID = 1) therefore ends the home activity at 6:00 a.m., departs by car (vehicle ID = 1), then enters and leaves a number of links in the network to arrive at work at 06:20:46. The agent departs from work at the scheduled time of 5:00 p.m., as specified in her day activity plan, and continues home. Each link traversed is identified explicitly by a link ID. The time taken to traverse a link is generated by queue simulation dynamics (36) and is therefore a stochastic, emergent property of the simulation.

The default scoring function, derived from Charypar and Nagel (37), in its simplest form rewards the performance of activities and penalizes travel and arriving late for activities. During the scoring step in Figure 1, the scoring module evaluates the timing of each agent’s activity start and end events, as well as travel start and end events, to derive the total time spent at each activity, time spent travelling, and so forth. It does not care where the event stream comes from, as long as it is properly formed and is chronological for each agent. It therefore allows for a simulation module other than QSim to feed it with an event stream to score.

**PSim Operation**

From the QSim event stream, the travel time for each agent on each link during the course of the simulated day can be deduced. Therefore the simulated day can be sliced into arbitrary time intervals of, say, 15 min each, the average travel time calculated for each link during every interval, and these values stored in a lookup table. A replanning module now produces a new plan for the agent in the example in which she leaves home a little later or takes a different route to work. The PSim module constructs an event stream that represents her expected experience in the transport system by reading the appropriate times from the lookup table for each link in her route at each relevant time interval. It passes this event stream to the scoring module, which produces an expected score for the new plan and keeps the scored plan in the agent’s memory. After the process is repeated several times, the agent’s memory limit is reached, and the poorest performing plan is discarded at the end of each iteration.

The agent is now learning not from the full stochastic queue simulation but from a simplified representation of it; consequently PSim is a surrogate model for QSim. After several iterations, the agents are passed back to QSim so the plan performance can be evaluated, an updated lookup table of travel times is produced, and the process repeats. No physical interaction occurs between agents in PSim, so it can fully exploit modern multicore computer architectures because no synchronization between threads is required and access to data structures outside a PSim 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 replanning operations.

QSim always requires the full set of agent plans as travel times emerge from their interaction. Because there is no interaction between agents in PSim, simulating only newly generated plans that do not yet have an associated score makes sense. This cuts down on the expected computational load even further, as every iteration generates only a small number of new plans, depending on the rate of replanning prescribed by the replanning strategy.

**EXPERIMENTAL SETUP**

The multimodeling approach was tested for compatibility, computational performance, and simulation state through comparison of its results for a large-scale simulation scenario against those produced by a baseline simulation run that uses the default, QSim-only approach.

The simulation used the MATSIM development scenario of Swiss car traffic crossing or operating within a 30-km radius circle around Bellevue, Zurich, as used in the secondary activity location choice study of Horni et al. (20). The scenario, developed by Balmer et al. (38) and later updated and further documented (39, 40), is regularly used as a benchmark in MATSIM investigations. The same 10% sample from the original study, as well as the same network representation and facility information, were used for the simulation. The scenario contained 67,239 agents traveling in a network of 60,518 links and a total of 1,697,196 activity facilities. An arbitrary morning toll was introduced on all links exceeding a capacity of 4,000 vehicles per hour.

The following replanning modules were used in equal measure, and the total replanning rate (proportion of agents replanned) was varied as part of the experimental setup:

1. Activity start time and duration adjustment;
2. Rerouting with travel times from the previous iteration;
3. Subtour mode choice, which switches the mode of transport of a randomly selected subtour to car or public transport given that, for this scenario, all agents have access to cars; and
4. Secondary activity location choice, in which shopping and leisure activities are switched to a randomly chosen location from a set of qualifying facilities.

Public transport was not explicitly simulated because this capability would require a full public transport schedule of vehicle departure times and a full set of public transport lines and routes. Instead, trips that use public transport are “teleported” during the simulation from origin to destination with a travel time that is twice that of the free-speed shortest path through the network (41).

**RESULTS**

**Defining a Simulation State for Comparison**

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. So that runs will be repeatable, a seed number is set for the Java random number generator at the beginning of a simulation run.

The same random seed was used for all simulation runs except a baseline QSim-only run. The baseline for comparison was taken as
the simulation state obtained by running this scenario for 101 iterations with QSim only, at an overall replanning rate of 30% per iteration, with a maximum agent memory of five plans per agent. At this point, the average executed plan score for the simulation was 175.4.

Another QSim-only run was performed exactly as the baseline run, but with a different random seed. Its simulation state at the 101st iteration represents the minimum deviation from the baseline that can be expected, and thus it is called the reference run. For all other experimental runs, the first QSim iteration where the average executed plan score was greater than or equal to 175.4 was selected, and simulation state was compared against the baseline.

**Varying QSim:PSim Ratio**

With the replanning rate kept constant, increasing the number of PSim iterations between QSim iterations increased the rate of convergence, as shown in Figure 2. The figure compares the utility versus number of QSim iterations for two QSim:PSim ratios against the reference run. In general, for a given intermediate utility score, the number of QSim iterations required to achieve that score is approximately inversely proportional to the total number of iterations executed during the simulation, for example, QSim + PSim iterations.

**Performance Test**

Figure 3 compares the influence of QSim:PSim ratio, number of computational cores, and replanning rate on simulation (wall clock) time. The multimodeling strategy is effective only as the number of cores committed to the simulation is increased. Figure 4 shows the wall clock time needed, with different setups, to reach the reference average plan score. 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 increasingly shrink. This is because of the computational (and conceptual) decoupling of the replanning: every agent replans for herself. Next, replacing most of the regular QSim runs with PSim runs results in significantly reduced QSim contributions to the overall wall clock time, even if one counts in the additional time for the PSim and the additional overhead. At this point, it was possible to reduce the computing time by more than a factor of two in the comparison of the 16 core results of the default approach with the fastest version by using the 16-core machine with the multimodeling approach.

Lowering the replanning rate while increasing the number of PSim iterations in the inner loop gives the best overall performance. 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 multimodeling runs in Figure 4. The first run has a replanning rate of 0.3 and QSim:PSim ratio of 1:9. Consequently, in 1 + 9 iterations, the expected number of new plans produced per agent comes to three, with a standard deviation of 1.44. In comparison, the second run has a replanning rate of 0.1 and QSim:PSim 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 quicker convergence is probably caused by the larger 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
FIGURE 3  Score evolution versus time for large-scale scenario, comparing influence of QSim:PSim ratio, number of computational cores, and replanning rate.

FIGURE 4  Computation time contributions versus number of cores for QSim only (0.3 replanning rate), QSim:PSim = 1:9 (0.3 replanning rate), and QSim:PSim = 1:24 (0.1 replanning rate) at reference score (gray line in Figure 3).
produces 16.7% fewer plans per outer loop cycle. It suffers, however, from increased overhead because of a larger total number of iterations.

**Simulation State Comparison**

Figure 5 shows a comparison of three runs against the baseline for a number of measures of simulation state. The QSim-only reference run is shown in black. Two multimodeling simulations were run, one with an overall replanning rate of 0.3 and QSim:PSim ratio of 1:9 (dashed line), the other with an overall replanning rate of 0.1, QSim:PSim ratio of 1:24, and a subtour mode choice replanning rate set to half that of other replanning modules in the strategy. This deviation in strategy is explained in the mode share section. Dots on the lines show the first QSim iteration in which the average executed plan score is greater than or equal to the baseline value of 175.4.

**Departure Profile Root Mean Square Deviation**

The numbers of agent departures are compared at 5-min intervals for the simulated day. The root mean square deviation (RMSD) is taken from the baseline departures as an indication of how similar a simulation state is to the baseline for activity timing. These values are plotted against the number of QSim iterations in Figure 5a. All three runs reach a minimum value around their baseline score iteration, then depart from this value with increasing QSim iterations. This likely
happens because activity timing is mutated with a random-response module, and many iterations are needed to stabilize.

**Mode Share**

Car mode share is compared against the baseline in Figure 5b. At the same replanning rate, the swing toward public transport is much larger for a multimodeling run than for the reference run. The routing and travel time of public transport are independent of network conditions for the simulations, as public transport was not explicitly simulated. Repeated PSim iterations give many more agents the chance to consider that during the initial iterations, with lots of car congestion, public transit is an attractive alternative. Because of this, the alternative strategy was tested with a lower rate of mode switching, as described earlier in this subsection. This run minimizes the initial swing to public transport. It also shows a continuing increase in car mode share with increasing QSim iterations, but at a much higher rate than that shown by the reference run. As peak spreading continues with increasing iterations, the car becomes a viable alternative again.

**Daily Link Volume RMSD**

The daily volume of car traffic traversing every link in the network is compared with the volumes produced by the baseline run. The RMSD from the baseline link volumes is taken as an indication of how similar car traffic volumes are to the baseline (Figure 5c). This measure does not show a minimum at the reference score iteration for any of the runs and takes longer to reach a minimum. Although the value at the reference score iteration is approximately twice that of the reference case, it is still relatively small in absolute value. In the long run, this measure appears to approach values comparable to the reference case.

**Difference in Agent’s Total Travel Time**

The event stream is processed so the total travel time experienced by each agent can be compared with those produced by the baseline run. The difference for each agent is compared for the two runs, and the percentage of agents that experienced a difference below 5 min and 1 min, respectively, is taken.

Table 1 compares the agent’s total travel time difference for the three runs at the reference score iteration, along with the other measures of solution state discussed earlier. $\Delta_{\text{traveltime}} \leq 5 \text{ min}$ and $\Delta_{\text{traveltime}} \leq 1 \text{ min}$ denote the percentage of agents with a total travel time difference (from the baseline) less than 5 min and 1 min, respectively; sharecar denotes car mode share. The magnitudes for $\Delta_{\text{traveltime}}$ between the three cases are comparable; at least 74% of agents have a total travel time that lies within 5 min of that experienced in the baseline run.

**CONCLUSION**

The multimodeling approach was designed to be consistent with the preexisting simulation logic of MATSIM, and it appears to produce comparable results. In all cases, use of the new approach reduces the number of time-consuming QSim iterations needed to achieve a given average plan score. The approach should be useful for reducing simulation times for most applications of MATSIM. In this paper, it was shown to work well with an extensive list of existing MATSIM capabilities, and its simple design should make it easy to maintain because MATSIM functionality is extended.

The multimodeling approach produces markedly different simulation states for the same average utility score compared with the reference QSim-only run. The new approach produces comparable link volumes and experienced travel times. The difference in mode share highlights the effect of ignoring congestion effects when transit travel times are calculated during initial iterations. An extended PSim that takes actual transit performance into account (42) to calculate transit travel times would be an improvement to the current strategy of congestion-independent teleportation and is the subject of further research.

What emerges from the study is the clear need for consideration of the relative contribution of each replanning module to the simulation state, because average utility on its own is not a complete indication of what is happening in the simulation. Furthermore, the study shows that as replanning strategies become more complex, and various modules with different levels of impact on simulation state are chained together, many more iterations are needed to achieve a stable simulation state. Substituting PSim for QSim in many of these iterations will help achieve this state in reasonable time.

**ACKNOWLEDGMENTS**

The authors thank the TRB reviewers for their input. The authors also thank the developers of the Zurich scenario for making the data available: Michael Balmer and Marcel Rieser of Senozon A.G., and Kay Axhausen, Francesco Ciari, Andreas Horni, Konrad Meister, Basil Vitins, and Rashid Waraich of the Institute of Transport Planning and Systems, Eidgenössische Technische Hochschule Zürich. The authors thank Gunnar Flötteröd of the Royal Institute of Technology, Stockholm, for insightful feedback and thank Dominik Grether of Transport Systems Planning and Transport Telematics (VSP), Technische Universität Berlin, for technical assistance during development of the PSim module. The authors thank Kay Axhausen for discussions and for review of drafts of the manuscript. This work was funded in part by a grant from the National Research Fund of Singapore.

<table>
<thead>
<tr>
<th>Run Description</th>
<th>QSim Iteration</th>
<th>Departure RMSD</th>
<th>Link Volume RMSD</th>
<th>$\Delta_{\text{traveltime}} \leq 5 \text{ min} (%)$</th>
<th>$\Delta_{\text{traveltime}} \leq 1 \text{ min} (%)$</th>
<th>sharecar (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reference</td>
<td>101</td>
<td>32.56</td>
<td>5.67</td>
<td>77.1</td>
<td>66.4</td>
<td>80.7</td>
</tr>
<tr>
<td>0.3 Q:P = 1:9</td>
<td>20</td>
<td>50.85</td>
<td>24.53</td>
<td>74.6</td>
<td>65.0</td>
<td>76.2</td>
</tr>
<tr>
<td>0.1 Q:P = 1:24</td>
<td>13</td>
<td>56.42</td>
<td>30.00</td>
<td>76.1</td>
<td>66.7</td>
<td>79.99</td>
</tr>
</tbody>
</table>
REFERENCES


