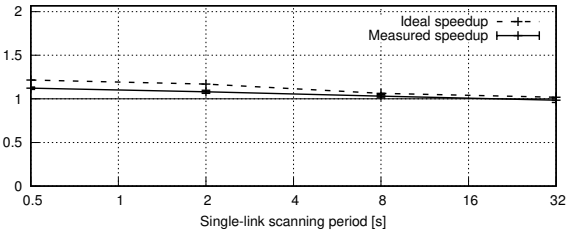
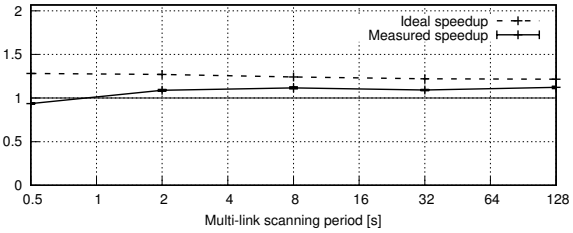
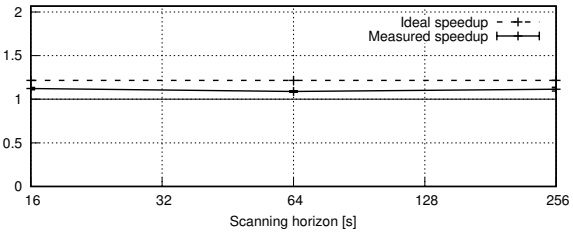


(a) Multi-link scanning period 0.5s, scanning horizon 16s.



(b) Single-link scanning period 0.5s, scanning horizon 16s.



(c) Single-link scanning period 0.5s, multi-link period 128s.

Figure 9: Ideal and measured speedup with 10 000 agents in the grid scenario.

Table 2: Performance in the Singapore scenario with scanning parameters configured using auto-tuning.

Peak agent count	6 000	19 500
Time-stepped execution time [s]	214.3 ± 3.4	789.1 ± 5.9
Steps skipped [%]	22.1 ± 0.1	10.3 ± 0.1
Steps skipped per fast-forwarding	85.45 ± 0.16	81.4 ± 0.15
Scanning overhead [s]	2.5 ± 0.0	7.4 ± 0.1
Fast-forwarding overhead [s]	2.0 ± 0.0	4.9 ± 0.0
Speedup	1.22 ± 0.03	1.05 ± 0.01

5 DISCUSSION

In this section, we discuss limitations and potential enhancements of the proposed fast-forwarding approach.

Applicability to more complex models and other domains: in the present paper, we assume that the fast-forward and scanning functions return a single agent state and time, implying that the routes of the agents (i.e., the sequences of edges) will not change after scanning has been performed. Depending on the considered simulation model, agents may change their routes on interaction with other agents or even spontaneously. For instance, an agent may re-route when it detects traffic congestion. To consider such models, we could either terminate the scanning process at the first point in time when a change is possible, or determine occupancy intervals for all possible branches. Both approaches may substantially reduce the opportunities for fast-forwarding compared to what has been shown in the simulation scenarios of the present paper. If routing decisions are made stochastically, pre-sampling from the pseudo-random number stream may still enable prediction of the agents' routes. In the extreme case of entirely unpredictable routes, fast-forwarding would be limited to disjoint areas reachable by agents according to their maximum velocity.

From the problem analysis in Section 3.1, we can infer that the applicability of our approach to other types of time-stepped agent-based simulations depends on the specific models used. The approach is applicable to models that allow for the prediction of future agent states and to scenarios where independent agent updates occur. For instance, in crowd simulations using predefined routes on a two-dimensional simulation space, the path taken by isolated agents may be fully predictable. The performance benefits of the approach depend on the computational cost of the fast-forward function relative to time-stepped updates as well as the costs for determining independence intervals.

Deviations compared to time-stepped execution: As discussed in Section 3 and evaluated in Section 4.1, state updates performed using the fast-forward function deviate slightly from those performed using iterative time steps. Although the fast-forward function can be argued to be closer to the behavior specified in IDM and the deviations are low, two undesirable properties emerge: first, as with any state alteration in an agent-based simulation, deviations may propagate through the road network. Second, the deviations depend on the parametrization of the fast-forwarding approach, i.e., on the frequency of scanning and on the scanning horizon. Thus, the approach interlinks the *execution* of the simulation and the observed *behavior* of the simulation. Ideally, to allow modelers to clearly identify cause-and-effect relationships when modifying the model or scenario, these two aspects should be decoupled.

Influence on statistics collection and visualization: Typically, in an agent-based simulation, statistics are gathered by periodically aggregating over the states of the agents in the simulation, e.g., over the velocities of the vehicles on a road network. Usually, the period of aggregation is a multiple of the time step size. If updates are performed asynchronously according to the proposed approach, agents may be fast-forwarded beyond the point when statistics are to be collected. A simple solution is to limit the scanning horizon to the next statistics gathering time. Visualization tools could apply interpolation to approximate agent states at time steps that have been skipped through fast-forwarding.

Further opportunities for fast-forwarding: the proposed multi-link scanning approach operates on the granularity of edges in a road network. Thus, if the modeled road network has a high percentage of edges that are substantially longer than the sensing range of vehicles, many opportunities for fast-forwarding are not exploited. By identifying independence intervals on a finer granularity, additional fast-forwarding opportunities could be unlocked. However, more complex scanning may incur an increase in overhead. Further, for efficiency, we limited fast-forwarding to the first agent that senses a graph edge. Fast-forwarding of multiple

agents occupying the same edge at disjoint time intervals within the scanning horizon may enable further speedup in some scenarios. Further, in road traffic simulations that consider traffic lights, trivial opportunities for fast-forwarding may be given for vehicles stopped in front of traffic lights. Such vehicles can be fast-forwarded to the next state change of the traffic lights. Finally, we only consider skipping opportunities for individual agents. An increase in fast-forwarding opportunities may be possible if the fast-forward function can be extended to clusters of agents.

Controlling overhead: a number of ways present themselves to control the scanning overhead and thus to enable more complex scanning procedures: first, the proposed algorithm is parametrized with the scanning periods and scanning horizon. Our evaluation showed that the optimal values for the parameters depend strongly on the considered scenario. Thus, we applied a simple auto-tuning scheme to adapt the scanning parameters according to the traffic conditions of the Singapore scenario.

Second, in addition to the variation of congestion across model time, congestion also typically shows variations across the simulated space. For instance, during peak hours a highly congested speedway will provide few opportunities for fast-forwarding, in contrast to sparsely populated roads in residential areas. In such situations, to avoid unnecessary computations, scanning could be restricted to areas outside congested areas. However, further considerations are then required to maintain correctness. Since vehicles may enter or exit congested areas within the considered scanning horizon, excluding agent interactions would require a safety margin around these areas, which could be defined based on static information such as speed limits.

Finally, the scanning operation may be offloaded to a separate processor. Within the accuracy allowed by the time step size, previously identified occupancy intervals may be outpaced by the simulation's progress, but not invalidated. Thus, after scanning, fast-forwarding could be applied to all agents that have not yet progressed beyond the target time. Further, during scanning, the scanning function is evaluated a number of times for each relevant vehicles independently, providing ample opportunities for parallelization, e.g., on graphics processing units.

6 RELATED WORK

In this section, we give an overview of previous work focusing on identifying and exploiting independence between state updates for parallelization of discrete-event simulations and for accelerating sequential and parallel time-stepped agent-based simulations. Further, we discuss hybrid modeling and simulation approaches that execute parts of a simulation microscopically, while applying less detailed and therefore less computationally intensive models for parts of the simulation where full accuracy is not required.

6.1 Exploiting independent state updates

The approach proposed in the present paper bears some similarities with methods from the field of parallel and distributed simulation, which is concerned with the execution of individual simulation runs on a set of inter-connected processing elements [8]. To reduce the cost of synchronization between processing elements, methods have been proposed to exploit *lookahead*, i.e., the difference in model time between an event's creation and execution time [7]. If a lower bound on the lookahead can be determined either prior to the simulation or at runtime [18, 24], intervals in model time can be identified during which processing elements can compute independently. Some previous works consider the minimum model time required for a sequence of events to propagate to a remote processing element [3, 4, 19, 20, 23, 27, 31]. Similarly to our approach, intervals of independence are derived according to the topology of the modeled system. However, instead of exploiting the identified independence for parallel execution, in our work, we accelerate sequential simulations by performing independent agent state updates using a computationally inexpensive fast-forward function. A further similarity exists to optimistically synchronized parallel and distributed simulations [6, 26], where some computations are performed speculatively and rolled back when a violation of the simulation correctness is detected. In our approach, the identification of occupancy intervals can be seen as speculative state updates under the assumption of independence among agents. When independence between the agent updates cannot be guaranteed, the results are discarded.

Some previous works have considered ways of accelerating time-stepped agent-based simulations by identifying independent state updates among agents: Scheutz et al. [13, 28, 29] apply a *translation function* that reflects the furthest possible amount of movement of an agent to determine an *event horizon* in model time. By identifying non-overlapping areas among multiple agents' event horizons, time intervals of mutually independent updates can be identified. Now, in the context of sequential agent-based simulations, agent updates can be prioritized to achieve the simulation's termination criterion with the minimum number of state updates. For instance, if the focus of the simulation study is on a particular agent, only the state updates directly or indirectly affecting this agent must be performed. In distributed agent-based simulations, idle times due to data dependencies can be reduced by prioritizing agent updates according to the data dependencies across processing elements. In contrast to our work, runtime reductions are achieved through changes in the ordering of agent updates, not through accelerating the state updates themselves. Since road traffic simulations are typically executed until all agents have reached a certain point in model time, the approach by Scheutz et al. would not accelerate such simulations.

Buss et al. [2] proposed a discrete-event modeling approach for scenarios involving movement and sensing. Instead of explicitly updating an entity's location over a sequence of time steps, events are scheduled at points in model time where changes in movement occur. However, determining suitable event scheduling times for sets of interacting vehicles may incur substantial overhead. Thus, in contrast to the purely discrete-event approach proposed by Buss et al., our proposed fast-forwarding approach maintains a time-stepped execution for all agents currently involved in an interaction. Further, while the work by Buss et al. and another work with a similar focus by Meyer [22] share with ours the general idea of avoiding explicit intermediate state changes, the main challenge lies in determining the points in model time when interactions between entities may occur and in determining the new agent state. In the present paper, we address these challenges in the context of microscopic road traffic simulations.

Less closely related to our approach is the concept of simulation cloning [14]. In this approach, the total execution time of a set of simulation runs is reduced by computing only the divergent state updates across multiple runs. For instance, if the behavior of only a single agent is modified across multiple runs, state changes of other agents that are unaffected by this agent are not recomputed [25]. Similarly, in updatable and exact-differential simulation [5, 12], intermediate events of an initial full simulation run are stored. Subsequent simulation runs branch off from this initial simulation run, reusing stored events that are unaffected by the branching. As in these approaches, fast-forwarding exploits the independence between state updates to accelerate simulations. However, instead of avoiding recomputation, the fast-forwarding approach proposed in the present paper avoids computation of some updates entirely.

Finally, the term "fast-forwarding" was used previously in other contexts where existing information is exploited to advance a simulated entity in model time. In the updatable simulations proposed by Ferenci et al. [5], some repeated event executions can be avoided, thus "fast-forwarding" the corresponding simulated entity. Mauve et al. [21] use the term "fast forward" to describe the re-execution of events after a rollback in the context of optimistic synchronization for distributed virtual environments.

6.2 Hybrid traffic simulation

In hybrid traffic simulation [1], microscopic models are combined with mesoscopic or macroscopic models to balance simulation fidelity and performance. Spatial or temporal segments of the simulation are selected in which a reduction in modeling detail and accuracy is acceptable. In these segments, vehicles are considered in aggregate, e.g., as sets of tasks in a queuing network or in terms of fluid dynamics. As a consequence, it is not always possible to study an individual vehicle across its entire route. The fast-forwarding approach proposed in the present paper bears a superficial similarity with hybrid traffic simulation in its reliance on an analytical solution for some of the state updates instead of a purely time-stepped execution. However, fast-forwarding is applied only if it is ensured that within the accuracy allowed by the simulation's time step size, the simulation results are unaffected. Since fast-forwarding does not consider agents in aggregate, each vehicle's progress on its route can still be studied individually.

7 CONCLUSIONS AND OUTLOOK

We propose an approach to accelerate microscopic traffic simulation by identifying intervals of independent state updates and performing such independent updates using a computationally inexpensive fast-forward function. The approach maintains the microscopic nature of the simulation. We derived a fast-forward function for the well-known Intelligent Driver Model. We evaluated the approach for a synthetic scenario and the road network of the city of Singapore. Our validation shows that the deviation from a purely time-stepped execution is marginal. The performance benefit of the approach depends strongly on the level of agent density in the scenarios. For scenarios with sparse traffic, a speedup of up to 2.8 was achieved, whereas with dense traffic, the reduced amount of opportunities for fast-forwarding allowed for only limited performance gains. One avenue for future work lies in identifying further opportunities for fast-forwarding, e.g., by increasing the spatial resolution of the approach, or by considering clusters of cars jointly. Further, methods to control the overhead of the approach could improve performance. For instance, avoiding attempts for fast-forwarding in congested areas of the road network could reduce unnecessary computations, while requiring further considerations to maintain correctness. Offloading the computational overhead to a separate processing element could hide some of the overhead. Finally, the fast-forwarding approach could be extended to models with more complex agent movement behaviors such as crowd models.

8 ACKNOWLEDGMENTS

This work was financially supported by the Singapore National Research Foundation under its Campus for Research Excellence And Technological Enterprise (CREATE) programme.

REFERENCES

[1] Wilco Burghout, Haris Koutsopoulos, and Ingmar Andreasson. 2005. Hybrid Mesoscopic-Microscopic Traffic Simulation. *Transportation Research Record: Journal of the Transportation Research Board* 1934 (2005), 218–255.
[2] Arnold H Buss and Paul J Sánchez. 2005. Simple Movement and Detection in Discrete Event Simulation. In *Proceedings of the Winter Simulation Conference*. Winter Simulation Conference, 992–1000.
[3] Moo-Kyoung Chung and Chong-Min Kyung. 2006. Improving Lookahead in Parallel Multiprocessor Simulation Using Dynamic Execution Path Prediction. In *Proceedings of the Workshop on Principles of Advanced and Distributed Simulation*. IEEE, 11–18.
[4] Ewa Deelman, Rajive Bagrodia, Rizos Sakellariou, and Vikram Adve. 2001. Improving Lookahead in Parallel Discrete Event Simulations of Large-scale Applications Using Compiler Analysis. In *Proceedings of the Workshop on Parallel and Distributed Simulation*. IEEE Computer Society, Washington, DC, USA, 5–13. http://dl.acm.org/citation.cfm?id=375658.375659
[5] Steve L Ferenci, Richard M Fujimoto, Mostafa H Ammar, Kalyan Perumalla, and George F Riley. 2002. Updateable Simulation of Communication Networks. In *Proceedings of the Workshop on Parallel and Distributed Simulation*. IEEE Computer Society, 107–114.
[6] Richard Fujimoto. 2015. Parallel and Distributed Simulation. In *Proceedings of the Winter Simulation Conference*. IEEE Press, 45–59.
[7] R. M. Fujimoto. 1988. Lookahead in Parallel Discrete Event Simulation. *Proceedings of the International Conference on Parallel Processing, Vol. 3* (1988), 34–41.
[8] Richard M Fujimoto. 2000. *Parallel and Distributed Simulation Systems*. Wiley New York.
[9] Walter Gander. 1985. On Halley's Iteration Method. *The American Mathematical Monthly* 92, 2 (1985), 131–134.
[10] Peter G Gipps. 1981. A Behavioural Car-following Model for Computer Simulation. *Transportation Research Part B: Methodological* 15, 2 (1981), 105–111.
[11] Peter G Gipps. 1986. A Model for the Structure of Lane-changing Decisions. *Transportation Research Part B: Methodological* 20, 5 (1986), 403–414.
[12] Masatoshi Hanai, Toyotaro Suzumura, Georgios Theodoropoulos, and Kalyan S Perumalla. 2015. Exact-differential Large-scale Traffic Simulation. In *Proceedings of the Conference on Principles of Advanced Discrete Simulation*. ACM, 271–280.
[13] Jack Harris and Matthias Scheutz. 2012. New Advances in Asynchronous Agent-based Scheduling. In *Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications*. WorldComp, 1.
[14] Maria Hybinette and Richard M Fujimoto. 2001. Cloning Parallel Simulations. *ACM Transactions on Modeling and Computer Simulation* 11, 4 (2001), 378–407.
[15] Arne Kesting, Martin Treiber, and Dirk Helbing. 2007. General Lane-changing Model MOBIL for Car-following Models. *Transportation Research Record: Journal of the Transportation Research Board* 1999 (2007), 86–94.
[16] Arne Kesting, Martin Treiber, and Dirk Helbing. 2008. Agents for Traffic Simulation. In *Multi-Agent Systems Simulation and Applications*, Adelinde M. Uhrmacher and Danny Weyns (Eds.). CRC Press, Chapter 11, 325–356. http://arxiv.org/abs/0805.0300
[17] Michael Lees, Brian Logan, and Rob Minson. 2005. Modelling environments for distributed simulation. In *Environments for Multi-Agent Systems*. 150–167. http://www.springerlink.com/index/81g6x6elxhx9tbnq.pdf
[18] Y.-B. Lin and E.D. Lazowska. 1990. Exploiting Lookahead in Parallel Simulation. *IEEE Transactions on Parallel and Distributed Systems* 1, 4 (1990), 457–469. https://doi.org/10.1109/71.80174
[19] Jason Liu and David M Nicol. 2002. Lookahead Revisited in Wireless Network Simulations. In *Proceedings of the Workshop on Parallel and Distributed Simulation*. IEEE Computer Society, 79–88.

[20] Boris D. Lubachevsky. 1989. Efficient Distributed Event-Driven Simulations of Multiple-Loop Networks. *Commun. ACM* 32, 1 (1989), 111–123.
[21] Martin Mauve, Jürgen Vogel, Volker Hilt, and Wolfgang Effelsberg. 2004. Local-lag and Timewarp: Providing Consistency for Replicated Continuous Applications. *IEEE Transactions on Multimedia* 6, 1 (2004), 47–57.
[22] Ruth Meyer. 2014. Event-Driven Multi-Agent Simulation. In *International Workshop on Multi-Agent Systems and Agent-Based Simulation*. Springer, 3–16.
[23] Richard A Meyer and Rajive L Bagrodia. 1999. Path Lookahead: a Data Flow View of PDES Models. In *Proceedings of the Workshop on Parallel and Distributed Simulation*. IEEE, 12–19.
[24] D.M. Nicol and J.H. Saltz. 1988. Dynamic Remapping of Parallel Computations with Varying Resource Demands. *IEEE Trans. Comput.* 37, 9 (1988), 1073–1087. https://doi.org/10.1109/12.2258
[25] Philip Pecher, Michael Hunter, and Richard Fujimoto. 2015. Efficient Execution of Replicated Transportation Simulations with Uncertain Vehicle Trajectories. *Procedia Computer Science* 51 (2015), 2638–2647.
[26] Kalyan S Perumalla, Mohammed M Olama, and Srikanth B Yoginath. 2016. Model-Based Dynamic Control of Speculative Forays in Parallel Computation. *Electronic Notes in Theoretical Computer Science* 327 (2016), 93–107.
[27] Patrick Peschlow, Andreas Voss, and Peter Martini. 2009. Good News for Parallel Wireless Network Simulations. In *Proceedings of the International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*. ACM, 134–142.
[28] Matthias Scheutz and Jack Harris. 2010. Adaptive Scheduling Algorithms for the Dynamic Distribution and Parallel Execution of Spatial Agent-based Models. *Parallel and Distributed Computational Intelligence* 269 (2010), 207–233.
[29] Matthias Scheutz and Paul Schermerhorn. 2006. Adaptive Algorithms for The Dynamic Distribution and Parallel Execution of Agent-based Models. *J. Parallel and Distrib. Comput.* 66, 8 (2006), 1037–1051. https://doi.org/10.1016/j.jpdc.2005.09.004
[30] Martin Treiber, Ansgar Hennecke, and Dirk Helbing. 2000. Congested Traffic States in Empirical Observations and Microscopic Simulations. *Physical Review E* 62, 2 (February 2000), 1805–1824.
[31] Jun Wang, Zhenjiang Dong, Sudhakar Yalamanchili, and George Riley. 2013. Optimizing Parallel Simulation of Multicore Systems Using Domain-Specific Knowledge. In *Proceedings of the Conference on Principles of Advanced Discrete Simulation*. ACM, 127–136.
[32] Yadong Xu, Wentong Cai, Heiko Aydt, Michael Lees, and Daniel Zehe. 2017. Relaxing Synchronization in Parallel Agent-based Road Traffic Simulation. *ACM Transactions on Modeling and Computer Simulation - Special Issue on PADS 2015* 27, 2 (2017), 14:1–24.
[33] Daniel Zehe, Suraj Nair, Alois Knoll, and David Eckhoff. 2017. Towards CityMoS: A Coupled City-Scale Mobility Simulation Framework. In *5th GI/ITG KuVS Fachgespräch Inter-Vehicle Communication*. FAU Erlangen-Nuremberg.