Proceedings of the 2018 Winter Simulation Conference
M. Rabe, A. A. Juan, N. Mustafee, A. Skoogh, S. Jain, and B. Johansson, eds.

MODEL PREEMPTION BASED ON DYNAMIC ANALYSIS OF SIMULATION DATA TO ACCELERATE TRAFFIC LIGHT TIMING OPTIMISATION

Philipp Andelfinger
Sajeev Udayakumar
Wentong Cai

TUMCREATE Ltd and
Nanyang Technological University
1 Create Way
Singapore, 138602, SINGAPORE

David Eckhoff
Alois Knoll

TUMCREATE Ltd and
Technical University of Munich
1 Create Way
Singapore, 138602, SINGAPORE

ABSTRACT

Since simulation-based optimisation typically requires large numbers of runs to identify sufficiently good solutions, the costs in terms of time and hardware can be enormous. To avoid unnecessary simulation runs, surrogate models can be applied, which estimate the simulation output under a given parameter combination. Model preemption is a related technique that dynamically analyses the simulation state at runtime to identify runs unlikely to result in a high-quality solution and terminates such runs early. However, existing work on model preemption relies on model-specific termination rules. In this paper, we describe an architecture for simulation-based optimisation using model preemption based on estimations of the simulation output. In a case study, the approach is applied to the optimisation of traffic light timings in a traffic simulation. We show that within a given time and hardware budget, model preemption enables the identification of higher-quality solutions than those found through traditional simulation-based optimisation.

1 INTRODUCTION

Many complex engineering and design tasks can be formulated in terms of simulation-based optimisation problems. In simulation-based optimisation, the value of the objective function is an output statistic of a simulation run at a given input parameter combination. As in general optimisation, the goal is to identify a parameter combination that minimises or maximises the objective function. Simulation-based optimisation has been applied to simulations in domains such as environmental engineering (Asadzadeh et al. 2014; Matott et al. 2012), road traffic (Vlahogianni 2015), and manufacturing (Frank et al. 2013).

Simulation-based optimisation is often associated with enormous computational costs as the number of simulations runs required to explore a given parameter space can be large. To achieve a good result within a given time and hardware budget, search methods such as gradient descent or simulated annealing can be applied (Carson and Maria 1997). A number of search methods rely on surrogate models, which approximate the objective function based on previous evaluations at different points in the parameter space. Examples of surrogate models include regression models and neural networks. In simulation-based optimisation, a computationally inexpensive surrogate model can help avoid a large portion of the simulation runs required to identify a high-quality parameter combination (Yi et al. 2017).

Model preemption is a related technique, which terminates – or *preempts* – simulation runs that are expected to yield an output that is not beneficial towards the optimisation goal. In contrast to traditional approaches based on surrogate models, model preemption relies on dynamic data from partial simulation runs to decide whether a run should be terminated early, i.e., the termination decision is based on an analysis of the simulation state at runtime. Existing works on model preemption focus on model-specific

978-1-5386-6572-5/18/$31.00 ©2018 IEEE

termination rules. While the existing approaches guarantee that preempting a simulation run does not affect the overall optimisation outcome, both the opportunities for computational savings and the applicability to certain combinations of optimisation mechanisms and objective functions are limited by this constraint.

We propose an architecture and approach for model preemption that does not rely on model-specific termination rules. Instead, the termination decision is made by estimating the simulation output based on the simulation state at runtime. Similarly to surrogate modelling, an inexpensive approximation replaces the remaining portion of the simulation run. In an early case study, the proposed approach is applied to the optimisation of traffic light timings in a road traffic simulation, evaluating the tradeoff between the estimation accuracy and the runtime savings. This tradeoff is explored further on the example of three common benchmark functions for optimisation algorithms. Finally, we discuss opportunities for further runtime reductions by a more sophisticated analysis of the simulation state information.

The remainder of the paper is structured as follows: In Section 2, we briefly introduce simulation-based optimisation and discuss related work in surrogate modelling and model preemption. A generic architecture for model preemption is proposed in Section 3. In Section 4, a case study is presented to investigate the benefits and limitations of the approach in the context of road traffic simulation, followed by Section 5, where we assess the effects of preemption on the optimisation progress. Remaining challenges and future work are discussed in Section 6. In Section 7, conclusions and avenues for future work are discussed.

2 RELATED WORK

In the past decades, there has been broad and intensive research into methods for accelerating simulation-based optimisation techniques. In this section, we sketch the general approach and give an overview of works on data farming, where simulation runs are steered to maximise the knowledge gain. Finally, we cover existing works on model preemption and discuss how these works relate to ours.

2.1 Simulation-based Optimisation

In simulation-based optimisation, a simulation model of a system under study is evaluated at different points in the input parameter space in order to minimise or maximise an output statistic (Figueira and Almada-Lobo 2014). In the simplest case, each evaluation involves one simulation run, while in stochastic simulations, multiple runs with different random seed may be performed to reduce the uncertainty in the results. Since the response surface of the simulation model is not known a priori, the parameter space is explored heuristically starting from an initial parameter combination. Due to the enormous number of possible parameter combinations of typical simulation models, simulation-based optimisation methods usually do not guarantee that a global or even local optimum is returned once a convergence criterion has been satisfied or the time and hardware budget has been exhausted.

Heuristics such as gradient descent, simulated annealing or genetic algorithms are commonly applied to select the next parameter combination based on the output of the previous simulation runs. Frequently, an approximation of the response surface is generated to steer the parameter space exploration towards the most promising areas of the parameter space. Such *surrogate models*, commonly based on regression approaches such as kriging or artificial neural networks, mimic the behaviour of the simulation model given a point in the parameter space (Carson and Maria 1997).

2.2 Data Farming

Data farming is a broad term for approaches that systematically gather simulation output data at different parameter combinations to generate “big picture” knowledge of the response surface of the considered system (Brandstein and Horne 1998). While the term was originally coined in the context of military simulations, data farming has been applied in a range of domains such as manufacturing (Feldkamp et al. 2015) and molecular dynamics (Król et al. 2014).

Feldkamp et al. propose simulation-based knowledge discovery methods where machine learning and visualisation methods are applied to the collected simulation outputs to steer parameter space explorations towards areas of interest (Feldkamp et al. 2015; Feldkamp et al. 2017). After clustering the output statistics, the output statistics of runs within each cluster are analysed to understand the overall system behaviour. In contrast to model preemption approaches, simulation outputs are gathered after execution of the simulation. Further, the goal of the analysis is to guide future parameter selections, instead of reducing the time taken to evaluate a given parameter combination.

2.3 Model Preemption

A number of existing works considered the early termination of simulation runs that are not expected to be beneficial towards the progress of simulation-based optimisations. In such *model preemption* approaches, simulation state information is analysed at runtime to decide whether the simulation should be terminated. A simulation-based optimisation system that uses model preemption can be viewed as a dynamic data-driven application system (DDDAS): in DDDAS, the application behaviour is adapted according to dynamic data injected at runtime, while at the same time the application may steer the data gathering process (Darema 2004). With model preemption, runtime simulation state information is provided to the optimisation mechanism in order to dynamically allocate resources to current or subsequent simulation runs, and to steer the parameter selection. This way, information from the current partial runs is fed into the subsequent runs.

The existing work on model preemption focuses on termination criteria that are guaranteed not to affect the outcome of the optimisation process. Razavi et al. analyse opportunities and challenges for model preemption motivated by the large cost of calibrating simulation models in the context of environmental simulation models (Razavi et al. 2010). In optimisation-based calibration, the objective function is the error between the data and the simulation results. Their objective functions is the cumulative sum of squared errors of the simulation results compared to the empirical data, up to the given point in simulation time. Since this function increases monotonically throughout the simulation time, a threshold can be defined beyond which the simulation run cannot affect the trajectory of the parameter space exploration. Further, they assess different optimisation mechanisms with respect to their suitability for model preemption. Across a number of case studies and calibration techniques, computational savings from 19% to 97% are observed. Similarly, Shafii et al. achieved savings of 8% to 35% for the calibration of a simulation model of hydrological systems without affecting the calibration outcome (Shafii et al. 2015). Matott et al. achieved savings of 33% to 67% in the context of simulations of the treatment of harmful waste at landfills (Matott et al. 2012). They applied a particle swarm optimiser to steer the design space exploration.

Joslin et al. apply an approach similar to model preemption to design space exploration for civil engineering problems using finite-element analysis (Joslin et al. 2006). Their problem allows the *quality* and the *validity* of a given design to be evaluated separately. Since evaluating the quality is inexpensive, the validity of the design is only evaluated for high-quality designs. Differently from model preemption, their approach does not involve the analysis of partial simulation outputs at runtime. However, similarly to our work, a dynamic threshold for avoiding the expensive evaluation of a design's validity is applied. Thus, the threshold must be chosen so that computational savings are achieved while still arriving at a high-quality optimisation outcome.

In the present paper, we propose a general architecture for model preemption based on an analysis of the runtime simulation state. We forego the constraint of maintaining the same optimisation outcome as without preemption and evaluate the effect of different termination criteria on the computational savings and the quality of the optimisation progress within a given time and hardware budget. Ideally, the savings in runtime through preempting low-quality runs will allow the optimisation to progress further towards an optimum than when executing all simulations runs in full.

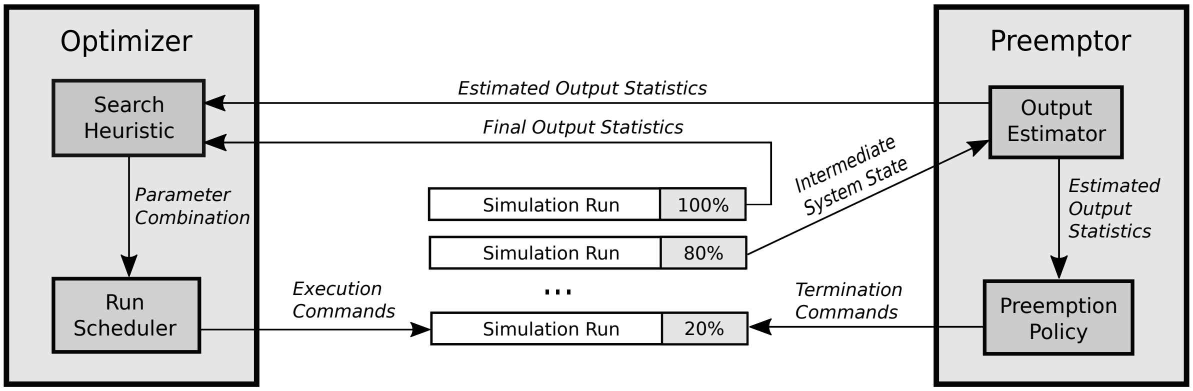


Figure 1: Proposed architecture for simulation-based optimisation using model preemption.

3 METHODOLOGY

In the following, we describe an architecture for model preemption that is targeted to be general enough to support a range of different simulation models, optimisation mechanisms, and objective functions. As shown in Figure 1, the main components of our proposed architecture are the *optimiser* and the *preemptor*. The left-hand side of the figure depicts a traditional simulation-based optimisation loop: a search heuristic such as gradient descent, simulated annealing or a genetic algorithm provides parameter combinations to a run scheduler, which executes simulation runs on the available hardware. The model preemption is performed by the preemptor component, which handles two tasks: the estimation of the simulation output based on intermediate simulation state information, and the application of the preemption policy.

3.1 Estimator

At a certain point during a simulation run (the *preemption point*), the preemptor is provided with the intermediate simulation state, on the basis of which the final simulation output is estimated. A type of surrogate model is applied to perform the estimation. However, in contrast to traditional surrogate models in simulation-based optimisation, the input is not the current simulation input parameter combination, but state information from a partial simulation run. When disregarding probabilistic model elements and assuming a static estimator for simplicity, we can express the estimation more formally as follows: given the input parameters $x_1,\dots,x_n$, let $f(x1,\dots,x_n)$ be the simulation *output*, and let $s(t,x_1,\dots,x_n)$ be the simulation *state* at simulation time t. A traditional surrogate model estimates the simulation output based on the input parameter combination, i.e., it is a function g with $g(x_1,\dots,x_n) = f(x_1,\dots,x_n) + r_1(x_1,\dots,x_n)$, with a residual r_1. In contrast, the estimator used in model preemption relies on the state of a partial simulation run and can be represented as a function h with $h(s(t,x_1,\dots x_n)) = f(x_1,\dots,x_n) + r_2(t,x_1,\dots,x_n)$. The residual r_2 is a function of the simulation time t and is expected to decrease with increasing t as the final output becomes progressively easier to predict.

Before an estimation can be performed, the surrogate model must be created. In this training phase, after a portion of a simulation run has been executed, the state information is provided to the estimator as training data. The simulation run is then continued and the final outcome after termination is also provided to the estimator. Based on a number of pairs of state information and final simulation outputs, the surrogate model is generated. Care must be taken to avoid both under- and overfitting of the surrogate model to the completed runs. Underfitting may occur if the surrogate model used is not suitable to model the relationships between state information and final outputs. While this cause may be avoided by choosing a more sophisticated surrogate model, underfitting may also be a result of a lack of training data. If the relationship between the intermediate simulation state and the final simulation output is complex, a large number of simulation runs may be required to reliably predict simulation outputs, potentially outweighing the runtime gains through the subsequent early termination of simulation runs. Conversely, overfitting may lead to preemption of high-quality runs: initially, the surrogate model is likely to be trained using simulation runs at portions of the parameter space that lead to low-quality outputs. Since the relationships used for estimation may evolve once higher-quality solutions are explored, there may be a need to initiate another training phase. One way of evaluating whether the surrogate model is still sufficiently accurate is by observing the quality of the estimation for runs that are executed in full, i.e., not preempted.

There is a tradeoff between the runtime cost of the training and estimation on one hand, and the potential for savings whenever a simulation run is preempted on the other hand. The cost of the estimation depends on the type of surrogate model used. Further, substantial overhead may be incurred if disk input and output is required to provide the surrogate model with the simulation state information used for estimation. This may occur if the estimation is performed based on large portions of the simulation state. An automated or manual sensitivity analysis may help to reduce the costs for training the model and for extraction of simulation state information.

A similar tradeoff exists when selecting the preemption point: at an early preemption point, estimations are likely to be inaccurate. However, the savings through early termination of a simulation run are high. By shifting the preemption point towards the end of the simulation, the savings are reduced, while the simulation output can be estimated more and more accurately. If the preemption point is set to the end of the simulation, the approach is reduced to traditional simulation-based optimisation. Similarly, a preemption point immediately at the start of the simulation run reduces the approach to traditional surrogate modelling, where the simulation output is estimated directly from the input parameter combination.

3.2 Preemption Policy

The second task of the preemptor is to apply the preemption policy, i.e., the criterion according to which runs are terminated. The selection of the preemption policy substantially affects both the computational savings and the quality of the solutions found during the optimisation process. Thus, as discussed in Section 2, the preemption policy has been a key consideration in existing works on model preemption.

A simple preemption policy is given by a fixed threshold with respect to statistics gathered from the partial simulation run's state. However, a fixed threshold is likely to only apply to early stages of the optimisation process: since the observed values of the objective function are likely to slowly evolve in the direction of the optimum, a fixed threshold may only allow for preemption in the early stages of the optimisation process. Thus, a dynamic threshold based on the current optimisation progress may be more appropriate. For instance, simulation runs may be terminated if the estimated output exceeds a constant factor of the best objective function value found up to that point in time, or if the current run is estimated to be of lower quality than a certain percentile of the previous runs. The effects of different combinations of optimisation mechanisms and preemption policies when targeting an unchanged outcome of the optimisation process are assessed by Razavi et al. (Razavi et al. 2010). In the present work, we propose a model preemption process that may affect the trajectory of the parameter space exploration, but aims at improving the quality of the solutions found within a given time budget. Since inaccurate estimations may lead to promising areas of the response surface not being explored, a tradeoff between estimation quality and runtime savings must be considered. In the case study presented in the following section, we evaluate the balance between the computational savings and the optimisation progress so that a better solution is identified within the constraints of a given time and hardware budget. The tradeoff is further investigated in Section 5.

4 CASE STUDY: TRAFFIC LIGHT TIMING OPTIMISATION

We performed an experiment with model preemption in a road traffic simulation. Our aim is to optimise traffic light timings to minimise the average travel time of vehicles passing through a road network. The scenario covers part of the Boon Lay area in the city of Singapore simulated using CityMoS, a city-scale microscopic traffic simulator (Zehe et al. 2017).

4.1 Experimental Setup

We extended CityMoS with a framework for optimisation and model preemption according to the architecture described in Section 3. Similarly to some existing works on traffic light timing optimisation (Sánchez et al. 2008; Sánchez et al. 2010; Teo et al. 2010), a genetic algorithm is used as the search heuristic. Our implementation is based on the GAlib genetic algorithm library (http://lancet.mit.edu/ga/). The parameters to be optimised are the durations of each of the traffic light phases for a small road network covering two intersections. There are 16 parameters, each represented by a 16 bit floating point number. The genetic algorithm was configured to apply mutation and crossover directly to the binary representation of the parameters, limiting the range of each parameter to millisecond values in the interval $[0, 20000]$.

After preliminary experiments, we configured the search heuristic and preemptor parameters so that both the benefits and the limitations of the approach can be shown. The genetic algorithm was parametrised with a population size of 20, a mutation probability of 0.01, and a crossover probability of 0.3. Of the 20 chromosomes in each generation, the 5 chromosomes with the best objective function value are carried over into the next generation. Each optimisation process was terminated after one hour wall-clock time. Each full simulation run took about 14 seconds wall-clock time to simulate two hours of model time, during which 1500 vehicles pass the network.

Once a configurable percentage of the vehicles have reached their destination, we evaluate whether the current run should be preempted. The experiments were conducted with preemption points after 50% and 80% of trips. According to our measurements, preempting a run at these two preemption points saved about 45% and 60% of the total runtime, respectively.

The estimator was implemented as follows: at each preemption point, a linear regression model is generated that estimates the average travel time at the end of the simulation from the average travel time of the vehicles that have already finished their trips. The regression is computed over the results of a configurable number of past full simulation runs. The estimations in our experiments are based on the past 40 runs. This estimator requires only simple statistical information from the simulator. However, in future work, a more sophisticated analysis may provide more timely and more accurate preemption information (cf. Section 6). As the preemption policy, the current simulation run is terminated if the estimated average travel time is higher than the median of the results from the past 40 full simulation runs. Each experiment was executed on a single core of a machine equipped with an Intel Core i5-7400 CPU and 16 GB of RAM.

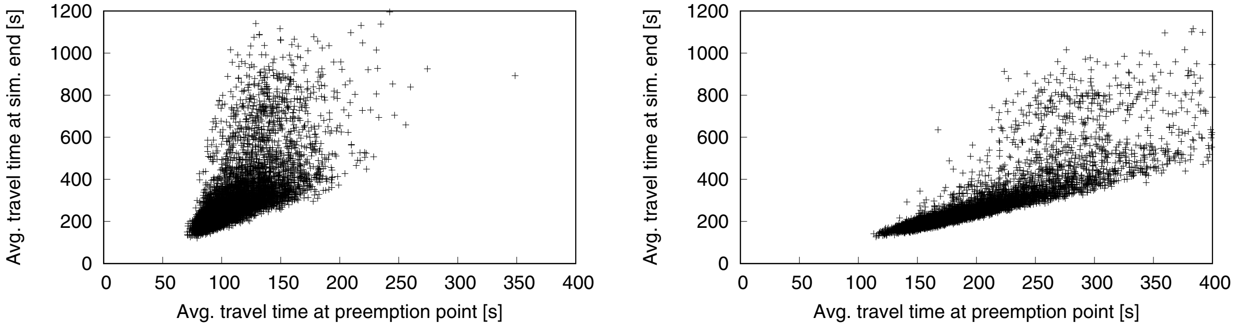
The main question we aim to answer with our experiment is whether in our model preemption setup, the increase in the number of runs per unit wall-clock time can outweigh the effect of discarding some high-quality parameter combinations due to inaccurate estimations of the final simulation result.

4.2 Results

We evaluate two aspects of the measurement results: first, the correlation between the data used for estimation and the final simulation output is investigated. A reasonable level of estimation accuracy must be achieved to avoid discarding high-quality parameter combinations. Second, we compare the optimisation progress in terms of the average travel times achieved with the parameter combinations selected by the genetic algorithm.

Figure 2 shows scatter plots of the average travel times at the preemption point and at the end of the respective simulation run. The preemption point is set to the time when 50% or 80% vehicles have finished their trips. We can see that while there is substantial variance in both cases, at 80% of trips a reasonable linear correlation can be observed (Pearson coefficient: 0.854). Clearly, the linear correlation becomes stronger if the preemption is shifted towards the end of the simulation, which, however, reduces the runtime savings through preempting a run.

Figure 3 shows an example of the simulation progress with and without preemption. With preemption enabled, runs that are estimated to be above the median of the past 40 full runs are terminated, allowing a larger number of parameter combinations to be evaluated within the time budget.



(a) After 50% of trips. Pearson coefficient: 0.675.

(b) After 80% of trips. Pearson coefficient: 0.854.

Figure 2: Scatter plot of the average travel times at the preemption point and at the end of the simulation. With a preemption point closer to the end of the simulation, the linear correlation becomes more pronounced.

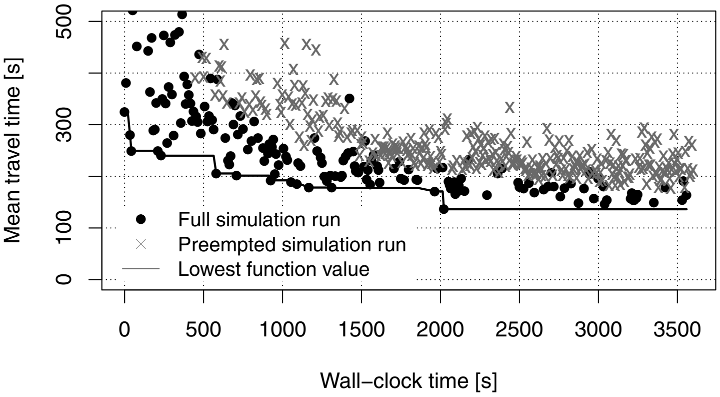


Figure 3: Example of the progress of an optimisation process using model preemption for the case study scenario. The black line indicates the lowest travel time identified up to the given time, black points indicate the output of an individual simulation run. A grey x indicates the estimated output of a preempted run.

In Figure 4, we plot the lowest average travel time achieved using model preemption and without preemption. The reported times include the overhead required to update the linear estimator used for prediction of the simulation output. The results shown are averages across at least 448 optimisation processes of one hour each. Without model preemption, an average of 407.3 runs were performed during the optimisation process. With preemption points at 50% and 80%, 632.2 and 479.7 runs were executed, which is an increase of 55.5% and 17.7%, respectively. With model preemption after 80% of trips, slightly better solutions could be identified, while preemption at 50% produced lower-quality solutions. Table 1 shows the confidence intervals at different points in time to evaluate the significance of the results. Since the differences in the solution quality are slight, we consider these results only a preliminary indication that runtime savings can be achieved, even though the confidence intervals after 60 minutes are non-overlapping. When increasing the confidence level to 99%, the confidence intervals for the non-preemptive execution and preemption 80% are $[151.9, 154.5]$, $[157.6, 163.6]$, and $[149.3, 151.8]$. Future work will consider a more sophisticated estimation of the simulation output to show a more pronounced benefit of the approach.

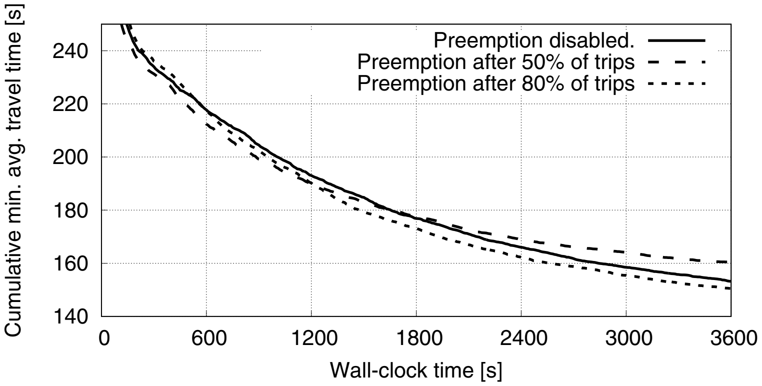


Figure 4: Optimisation progress for the case study scenario with and without model preemption. The curves indicate the lowest travel time identified up to the given point in wall-clock time, averaged across at least 448 repetitions of the optimisation process of one hour wall-clock time each.

Table 1: Lowest objective function value found in the traffic light timing optimisation up to the given wall-clock time with 95% confidence intervals.

Wall-clock time [min]	Preemption disabled	Preemption after 50% of trips	Preemption after 80% of trips
15	204.5 ± 1.7	199.2 ± 1.7	202.6 ± 1.6
30	176.9 ± 1.3	177.5 ± 2.0	173.1 ± 1.2
45	161.8 ± 1.1	166.2 ± 2.2	158.8 ± 1.0
60	153.2 ± 1.0	160.5 ± 2.3	150.5 ± 0.9

5 EFFECT OF ESTIMATION ERROR

As shown in the previous section, one of the key challenges when applying model preemption as described in the present paper is selecting a suitable preemption point. If preemption is considered too early throughout a simulation run, the estimation of the final simulation output may be inaccurate and high-quality solutions can be discarded. If preemption is considered too late, the runtime savings may be small.

In the following, the effect of the estimation error at different levels of runtime savings on the solution quality is assessed. To show results where the response surface is well-known, for this experiment we replace the simulation by common benchmark objective functions. As in the case study, a genetic algorithm is employed to find the minimum function value by varying two input parameters x and y. The following three non-linear objective functions are considered:

- Booth (Mehmani et al. 2012): $f(x,y) = (x+2y-7)^2 + (2x+y-5)^2$
- Himmelblau (Himmelblau 1972): $f(x,y) = (x^2+y-11)^2 + (x+y^2-7)^2$
- Rosenbrock (Rosenbrock 1960): $f_{a,b}(x,y) = (a-x)^2 + b(y-x^2)^2$ with $a = 1$ and $b = 100$

All of the three functions have minimum values of 0. To identify solutions of reasonable quality within the experiment time frame, the parameter space explored by the genetic algorithm was constrained as follows:

- Booth: $x \in [0,2], y \in [0,6]$
- Himmelblau: $x \in [-10,10], y \in [-10,10]$
- Rosenbrock: $x \in [0,2], y \in [0,2]$

Since the amount of computation per function evaluation is too small to apply preemption, we only imitate the effects of model preemption. The inaccuracy in the estimation of the simulation output is imitated by adding Gaussian random noise to the function value. Instead of returning the exact value of the function,

Table 2: Optimisation results for the benchmark functions, depending on the parameter c_1, which scales the standard deviation of the estimation error, and the relative runtime savings c_2 per preemption. The lowest identified function values are set in boldface.

c_1: *Estimation error*	c_2: *Runtime savings*	Booth [$\times 10^{-4}$]	Himmelblau	Rosenbrock [$\times 10^{-3}$]
0	0	4.00	0.063	7.14
0.1	0.1	3.08	0.048	6.20
0.1	0.25	1.89	0.031	5.29
0.1	0.5	**0.61**	**0.013**	**3.28**
1	0.1	3.23	0.055	6.36
1	0.25	2.27	0.038	5.70
1	0.5	1.02	0.022	4.13
10	0.1	5.00	0.068	6.32
10	0.25	3.54	0.056	5.92
10	0.5	2.16	0.035	5.10

when a function evaluation is "preempted", a Gaussian random variate is returned with a mean at the objective function value v, and a standard deviation of $v \times c_1$, with a configurable parameter $c_1 \in \{0.1, 1, 10\}$ to simulate different degrees of estimation error.

Analogously to the case study in the previous section, runs are preempted if the estimation is above the median of the past 40 full runs. In these cases, the noisy function value is returned. Otherwise, the exact objective function value is returned, imitating the case where the simulation run was completed and the exact result is available. Initially, the optimisation process is configured to have a budget of 1000 function evaluations. To imitate the opportunity to use the runtime savings to perform additional simulation runs within the given time budget, each preemption adds a configurable number of $c_2 \in \{0.1, 0.25, 0.5\}$ function evaluations to the budget.

Table 2 shows averages and 95% confidence intervals of the highest-quality objective function values identified before termination. We averaged across 10 000 optimisation processes per parameter combination of c_1 and c_2. All 95% confidence intervals were smaller than 13% of the respective mean value. In most cases, the solution quality increases through model preemption. Even with large amounts of added noise, the best identified solution is close to the base case without model preemption. The preemption increased the total number of function evaluations by around 4% (Booth, Himmelblau, and Rosenbrock, $c_1 = 1.0, c_2 = 0.1$) to about 67% (Rosenbrock, $c_1 = 1, c_2 = 0.5$).

To study the case of an asymmetric distribution of the estimation error, the experiment was repeated with noise generated according to a log-normal distribution, which is highly skewed when the standard deviations is large. Overall, we observed a similar trend as with the Gaussian noise. For all three objective functions, the final objective function values deviated at most 31% from the results under Gaussian noise.

From the results, we conclude that with the considered response surfaces, the optimisation progress may be accelerated by model preemption even if substantial estimation errors occur. However, this may be specific to optimisation using genetic algorithms, which are based on random mutation and combination of parameter combinations. More targeted search heuristics such as gradient descent may be misled more strongly by the estimation error, so that promising areas of the response surface may not be explored. In Section 6, potential extensions to the approach to maintain high-quality estimations are discussed.

6 DISCUSSION

A challenge when applying the model preemption approach is given by the substantial number of configurable parameters that may affect the optimisation progress in non-obvious ways. Aspects such as the type of model chosen to estimate the simulation output, the duration and frequency of estimator training periods, the preemption point, and the termination policy may need to be adapted to the model and scenario at hand based on domain knowledge or trial optimisation runs. A tradeoff must be considered when tuning the preemption policy: if the tolerance for low-quality runs is too high, barely any run will be terminated and the benefit of the approach is low. On the other hand, if the tolerance is too low, simulation runs that would positively affect the optimisation progress are preempted. Generally, the preemption policy and the preemption point should be chosen in tandem: towards the end of the simulation, it will typically be possible to estimate the simulation output at reasonable accuracy. In such cases, the closer the preemption point is towards the end of the simulation run, the lower the tolerance for low-quality runs can be set without discarding important simulation runs. Furthermore, for each full simulation run, the estimation quality can be evaluated to either adapt the strictness of the preemption policy or to shift the preemption point until a sufficient accuracy can be achieved.

To deploy the approach targeting an existing simulation system, the simulator must support the extraction of state information at runtime. Such facilities could range from simple aggregation mechanisms to gather simulation statistics, to full access to the simulation state. Simulators that support state-saving, e.g., for optimistic parallel and distributed simulation, are already equipped to support the analysis. If large amounts of data must be analysed, frameworks such as Apache Spark (http://spark.apache.org) could be employed, which enable parallelised data analysis across multiple nodes. Alternatively, full simulation runs may be performed until a sufficient level of confidence has been achieved to decide that the run will not be beneficial, after which no further averaging runs are performed for the current parameter combination.

If the considered simulation model is probabilistic, multiple runs will typically be performed for each simulation run to reduce the uncertainty in the results. When using model preemption, some of the results over which the averaging is applied could be terminated. In effect, since for the preempted runs only an estimation of the simulation output will be returned to the optimiser, the uncertainty for runs that are expected to be of low quality is higher than for other runs, which agrees with the goal of the model preemption approach. However, it may be more beneficial to entirely terminate the evaluation of the current parameter combination once one run has been preempted. Alternatively, the averaging for the current parameter combination may be terminated entirely once a sufficient confidence level has been achieved.

The preemption policy could be adapted to terminate even high-quality runs if there is a sufficient confidence in the estimation accuracy. However, since often, the input parameter combination of a high-quality run is in the proximity of other high-quality solutions, a high level of confidence is required to not mislead the search heuristic.

In our case study, the estimation and preemption decision was based on the same state information as the one used in the objective function. A key avenue for further exploration of the idea of model preemption is to automatically identify model-specific properties of the simulation state that allow for early termination without significantly affecting the optimisation progress. For instance, from the distance of a certain percentage of vehicles from their destination at the preemption point it may be inferred that the average travel time will not be low enough to benefit the optimisation progress. In future work, we aim to extract such inference rules for different scenarios and objective functions.

7 CONCLUSIONS

We presented an architecture for model preemption based on an analysis of dynamic simulation data to reduce the time required to achieve a given solution quality in simulation-based optimisation. Based on an estimation of the final simulation output, unpromising simulation runs are terminated early. The results of a case study of traffic light optimisation using microscopic traffic simulation indicate that the model preemption approach is in fact able to help identify higher-quality solutions within a given time budget. However, tradeoffs must be considered in the parametrisation of the approach to successfully accelerate the optimisation progress. In future work, we plan to explore the use of machine learning approaches to increase the estimation accuracy and to allow for earlier termination of unpromising runs.

ACKNOWLEDGEMENT

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

REFERENCES

Asadzadeh, M., S. Razavi, B. A. Tolson, and D. Fay. 2014. “Pre-emption Strategies for Efficient Multi-objective Optimization: Application to the Development of Lake Superior Regulation Plan”. *Environmental Modelling & Software* 54:128–141.

Brandstein, A., and G. Horne. 1998. “Data Farming: A Meta-Technique for Research in the 21st Century”. *Maneuver Warfare Science* 1998:93–99.

Carson, Y., and A. Maria. 1997. “Simulation Optimization: Methods and Applications”. In *Proceedings of the 1997 Winter Simulation Conference*, edited by S. Andradottir et al., 118–124. Piscataway, New Jersey: IEEE.

Darema, F. 2004. “Dynamic Data Driven Applications Systems: A New Paradigm for Application Simulations and Measurements”. In *International Conference on Computational Science*, June 7th–9th, Krakow, Poland, 662–669.

Feldkamp, N., S. Bergmann, and S. Strassburger. 2015. “Knowledge Discovery in Manufacturing Simulations”. In *Proceedings of the Conference on Principles of Advanced Discrete Simulation*, June 10th–12th, London, UK, 3–12.

Feldkamp, N., S. Bergmann, and S. Strassburger. 2017. “Online Analysis of Simulation Data with Stream-based Data Mining”. In *Proceedings of the Conference on Principles of Advanced Discrete Simulation*, May 24th–26th, Singapore, Singapore, 241–248.

Figueira, G., and B. Almada-Lobo. 2014. “Hybrid Simulation-Optimization Methods: A Taxonomy and Discussion”. *Simulation Modelling Practice and Theory* 46:118–134.

Frank, M., C. Laroque, and T. Uhlig. 2013. “Reducing Computation Time in Simulation-based Optimization of Manufacturing Systems”. In *Proceedings of the Winter Simulation Conference*, edited by R. Pasupathy et al., 2710–2721. Piscataway, New Jersey: IEEE.

Himmelblau, D. M. 1972. *Applied Nonlinear Programming*. New York City, New York: McGraw-Hill.

Joslin, D., J. Dragovich, H. Vo, and J. Terada. 2006. “Opportunistic Fitness Evaluation in a Genetic Algorithm for Civil Engineering Design Optimization”. In *Congress on Evolutionary Computation*, July 16th–21st, Vancouver, Canada, 2904–2911.

Król, D., M. Orzechowski, J. Kitowski, C. Niethammer, A. Sulisto, and A. Wafai. 2014. “A Cloud-based Data Farming Platform for Molecular Dynamics Simulations”. In *Proceedings of the International Conference on Utility and Cloud Computing*, December 8th–11th, London, UK, 579–584.

Matott, L. S., B. A. Tolson, and M. Asadzadeh. 2012. “A Benchmarking Framework for Simulation-based Optimization of Environmental Models”. *Environmental Modelling & Software* 35:19–30.

Mehmani, A., J. Zhang, S. Chowdhury, and A. Messac. 2012. “Surrogate-based Design Optimization with Adaptive Sequential Sampling”. In *Proceedings of the Structures, Structural Dynamics and Materials Conference*, April 23rd–26th, Honolulu, HI, 1–14.

Razavi, S., B. A. Tolson, L. S. Matott, N. R. Thomson, A. MacLean, and F. R. Seglenieks. 2010. “Reducing the Computational Cost of Automatic Calibration through Model Preemption”. *Water Resources Research* 46(11):1–17.

Rosenbrock, H. 1960. “An Automatic Method for Finding the Greatest or Least Value of a Function”. *The Computer Journal* 3(3):175–184.

Sánchez, J., M. Galán, and E. Rubio. 2008. “Applying a Traffic Lights Evolutionary Optimization Technique to a Real Case: “Las Ramblas” Area in Santa Cruz de Tenerife”. *IEEE Transactions on Evolutionary Computation* 12(1):25–40.

Sánchez, J., M. Galán, and E. Rubio. 2010. "Traffic Signal Optimization in "La Almozara" District in Saragossa Under Congestion Conditions, Using Genetic Algorithms, Traffic Microsimulation, and Cluster Computing". *IEEE Transactions on Intelligent Transportation Systems* 11(1):132–141.

Shafii, M., B. Tolson, and S. Matott. 2015. "Improving the Efficiency of Monte Carlo Bayesian Calibration of Hydrologic Models via Model Pre-emption". *Journal of Hydroinformatics* 17:763–770.

Teo, K. T. K., W. Y. Kow, and Y. Chin. 2010. "Optimization of Traffic Flow within an Urban Traffic Light Intersection with Genetic Algorithm". In *International Conference on Computational Intelligence, Modelling and Simulation*, September 28th–30th, Tuban, Indonesia, 172–177.

Vlahogianni, E. I. 2015. "Optimization of Traffic Forecasting: Intelligent Surrogate Modelling". *Transportation Research Part C: Emerging Technologies* 55:14–23.

Yi, W., J. Zhong, S. Tan, W. Cai, and N. Hu. 2017. "Surrogate Assisted Calibration Framework for Crowd Model Calibration". In *Proceedings of the 2017 Winter Simulation Conference*, edited by W. K. V. Chan et al., 1216–1227. Piscataway, New Jersey: IEEE.

Zehe, D., S. Nair, A. Knoll, and D. Eckhoff. 2017. "Towards CityMoS: A Coupled City-Scale Mobility Simulation Framework". In *GI/ITG KuVS Fachgespräch Inter-Vehicle Communication*, 6th–7th April, Erlangen, Germany, 26–28.

AUTHOR BIOGRAPHIES

PHILIPP ANDELFINGER is a postdoctoral research fellow at TUMCREATE and Nanyang Technological University (NTU), Singapore in the group of Prof Wentong Cai. He received his diploma and Ph.D. in Computer Science in 2011 and 2016 from Karlsruhe Institute of Technology (KIT), Germany. His research focuses on parallel and distributed simulation in heterogeneous hardware environments, agent-based simulation, and network simulation. His email address is pandelfinger@ntu.edu.sg.

SAJEEV UDAYAKUMAR is a research associate at TUMCREATE and NTU in the group of Prof Wentong Cai. He received a Bachelor's degree in Electronics and Telecommunications Engineering from University of Moratuwa, Sri Lanka and an M.Sc. in Information Systems from NTU. His email address is saje0004@e.ntu.edu.sg.

DAVID ECKHOFF is a postdoctoral research fellow and principal investigator at TUMCREATE, Singapore. David received his Ph.D. degree in Engineering and his M.Sc. degree in Computer Science from the University of Erlangen in 2016 and 2009, respectively. His research interests include privacy protection and smart cities with a particular focus on modelling and simulation. His email address is david.eckhoff@tum-create.edu.sg.

WENTONG CAI is a Professor in the School of Computer Engineering at Nanyang Technological University, Singapore. He received his Ph.D. in Computer Science from University of Exeter (UK) in 1991. His expertise is mainly in the areas of Modeling and Simulation and Parallel and Distributed Computing. He has published extensively in these areas and has received a number of best paper awards at the international conferences for his research in parallel and distributed simulation. He is an associate editor of the ACM Transactions on Modeling and Computer Simulation (TOMACS) and an editor of the Future Generation Computer Systems (FGCS). His email address is aswtcai@ntu.edu.sg.

ALOIS KNOLL received his diploma (M.Sc.) degree in Electrical/Communications Engineering from the University of Stuttgart and his PhD degree in Computer Science from the Technical University of Berlin. He was a full professor at the University of Bielefeld until 2001. Since autumn 2001 he has been a professor of Computer Science at the Technische Universität München (TUM). Between April 2004 and March 2006 he was Executive Director of the Institute of Computer Science at TUM. His email address is knoll@in.tum.de.