# NESTED ON-LINE SIMULATION — A M/M/1-QUEUEING EXAMPLE

Thomas Bessey
Helena Szczerbicka
Simulation & Modeling, Department of Computer Science, University of Hannover
Welfengarten 1, 30167 Hannover, Germany
{tby,hsz}@sim.uni-hannover.de

**KEYWORDS**
On-Line Simulation, Queueing Discipline, Nesting, Recursion

## ABSTRACT

For a M/M/1-queueing system, we study a novel queueing discipline based on on-line simulation which seeks to optimize future performance of the system by present decisions. Every arriving job has a due date which should not be exceeded by its actual completion time; in case of such excess, an undesired tardiness of the job is observed. The performance of the system is measured in terms of the overall mean tardiness. Basically, the queueing discipline used is Earliest Due Date. In addition, for every arriving job, its completion time is estimated by means of simulation; the job is rejected to be processed if its estimated completion time exceeds its due date (otherwise it is enqueued according to Earliest Due Date). Since future jobs may be enqueued in front of jobs already waiting in the queue (due to them having earlier due dates), the simulation should include the arrival process of the system for accurate estimations.

However, simulation of the system, particularly of its arrival process, introduces nesting of on-line simulation in itself due to recursive calls. In previous work, it turned out that nesting may have a significant impact on the performance resulting from operation of a queueing discipline based on on-line simulation. In this paper, we compare the performance of the system as it is controlled by on-line simulation with and without inclusion of the arrival process as well as with and without nesting of on-line simulation. It turns out that the results differ significantly not with respect to the overall mean tardiness (as in previous work) but with respect to the percentage of rejected jobs, which may have a considerable impact on overall customer satisfaction in real-world scenarios.

## INTRODUCTION

Today, semiconductor manufacturing becomes more and more customer-driven, i.e., instead of production "to the stock", chips are manufactured "just in time" (JIT) of customer demands. For such systems, optimality of their operation largely depends on their ability to respond rapidly to customer demands as they change over time (dynamic demands). In this context, novel approaches for real-time scheduling of jobs as well as injection of jobs into the system have been proposed, e.g. [7, 5, 8, 11] and [10], respectively. Such approaches apply simulation in order to simulate scheduling or injection alternatives and to select the alternative which leads to best estimated performance based on the current state of the system; as the simulation is executed during actual system operation and initialized to the current state, it is referred to as "on-line simulation" [4].

### On-Line Simulation

The idea is to optimize the system's performance online by repeatedly adjusting the system's parameters properly. This is referred to as adaptive control. In general, adaptive control is either reactive or proactive. The former type is characterized by adjusting the system's parameters only after a considerable performance drop is observed.

Proactive adaptive control tries to adjust the system's parameters before a performance drop is to occur, in order to avoid this drop. To this end, the system's future evolution is assessed in advance repeatedly. The instants of time at which the assessment of the further evolution and adjustment of the parameters are done are called decision points. The assessment is done as follows: First, the system's current state is copied to several identical system models. For each of these models, certain values of the system's parameters are set, according to some appropriate policies that alternatively could control the system. Once the initialization is done, the models are analyzed in order to assess the future evolution

*Proceedings 16th European Simulation Symposium*
*György Lipovszki, István Molnár © SCS Press, 2004*
*ISBN 1-56555-286-5(book) / ISBN 1-84233-106-x(CD)*

under each policy. As the system under control typically is complex, simulation is the only feasible analysis method. This method is referred to as on-line simulation. With the results, the policy that leads to the optimal future performance of the system under control is chosen to be implemented next, that is, the system is controlled by the chosen policy until the next decision point. This process is referred to as decision making.

There are several problems encountered with this approach, such as setting of the decision points, repeated validation of the system model (the search space for the parameters may vary over time) and proper analysis of the simulation results [1]. As simulation runs consume much time, the number and the length of the simulation runs become crucial. Since the system under control continues to evolve while the next policy is sought by on-line simulation, further problems arise [1]. For a detailed discussion of on-line simulation and associated proactive adaptive control, see [4].

For some applications of this approach for traffic systems and (flexible) manufacturing systems, see [6] and [11, 5, 9], respectively. However, these applications are by far not suitable for widely adoption to the real world; they merely employ classical off-line simulation techniques for on-line usage. By now, no strict theoretical research has been done.

In this work, we focus on the application of on-line simulation to the control of injection of jobs into the system.

We have introduced on-line simulation as a means for evaluation of different alternative system configurations that may be operated for the time between two subsequent decision points in response to certain characteristics of the environment in order to avoid performance drops. There is some more specific usage of on-line simulation that has attracted attention recently: Novel approaches based on on-line simulation are suggested for the adaptive control of arrival processes into a (sub)system. For example, the injection of jobs (release of work) into a manufacturing system could be controlled in an adaptive manner in that jobs may be re-ordered or re-scheduled (e.g., grouped in batches) or even rejected in order to optimize overall job processing with respect to certain performance measures such as minimal tardiness. On-line simulation is used to simulate injection alternatives. The key difference of this approach to what we introduced before is that on-line simulation is executed for every job arriving at the system in order to make a decision (regarding ordering, scheduling and acceptance/rejection). Instead of choosing between different configurations (injection policies) that can alternatively be operated for some time in order to make decisions regarding injection (such as First Come First Served, Earliest Due Date, Least Slack First, Shortest Job First etc.), on-line simulation is used for every job individually. Thus, the resulting injection policy itself is continuously adapting to the job injection process (it is the only configuration of the system for its entire operational lifetime). This way, it is expected to get even better results than when applying the "classical" approach since subsequent decision points have minimal distance regarding the external cause of state changings of the system (as there is not any arrival event between two subsequent decision points).

In [10], simulation-based job acceptance/rejection is presented as a novel approach for performance optimization (of a manufacturing system). However, the cited work is based on considerable simplifying assumptions such as exclusion of any arrival process from on-line simulation, questioning its applicability.

As inclusion of the arrival process leads to nesting of on-line simulation in itself due to recursive calls (the arrival process in turn is controlled by on-line simulation) which have to be terminated at some depth, the impact of different depths on the success of the approach certainly is of great interest.

**This Work**

For a M/M/1-queueing system, we study a novel queueing discipline based on on-line simulation which seeks to optimize future performance of the system by present decisions. As in the case of semiconductor manufacturing, every arriving job has a due date which should not be exceeded by its actual completion time; in case of such excess, an undesired tardiness of the job is observed. The performance of the system is measured in terms of the overall mean tardiness $\bar{T}$. This measure is of major importance in real-world scenarios as it has direct impact on satisfaction of customer needs resulting from the JIT-paradigm. Basically, the queueing discipline used is Earliest Due Date (EDD). The basic idea of EDD is to order jobs in the queue according to their due dates; it gives jobs having earlier due dates higher priority than jobs having later due dates, potentially reducing overall mean tardiness. In addition, for every arriving job, its completion time is estimated by means of simulation; the job is rejected to be processed if its estimated completion time exceeds its due date (otherwise it is enqueued according to Earliest Due Date). Since future jobs may be enqueued in front of jobs already waiting in the queue (due to them having earlier due dates), the simulation should

include the arrival process of the system for accurate estimations.

Of course, a trade-off between mean tardiness according to all jobs and rejection of single jobs in order to optimize overall customer satisfaction should be found, which is out of scope of this paper. However, we show that the percentage of rejected jobs differs significantly depending on whether the arrival process is included in the on-line simulation or not and whether the on-line simulation is nested or not.

This study is intended as providing further insight into the application of on-line simulation to the control of injection of jobs; it is a supplement of previous work [2].

The remainder of this paper is organized as follows: In the next section, the system together with the novel queueing discipline based on on-line simulation is introduced in detail; following this, some experimental results according to performance of the system as it operates under the queueing discipline with and without inclusion of the arrival process as well as with and without nesting of on-line simulation are given. Finally, in the last sections, we outline future work and give an overall conclusion, respectively.

## THE SYSTEM

The system is an open queueing system with one single job arrival process and one single resource or server for the processing of the jobs. Whenever the resource is busy processing a job, arriving jobs wait in an unlimited queue (if not rejected); each job visits the resource at most once, i.e., there is not any loop nor any preemption. Both the interarrival times and the processing times are assumed to be exponentially distributed, i.e. the processes in question are Poissonian. The queueing system described is referred to as M/M/1 [3]. With $\lambda$ being the arrival rate and $\mu$ being the service rate ($\lambda, \mu > 0$), the system is stable, i.e., it reaches a steady state, if and only if $\lambda < \mu$.

We consider $K = 3$ job classes that may differ in their mean processing times $\mu_k^{-1}$ $(k = 1, 2, \ldots, K)$ as well as in their mean relative due dates, where we define the mean relative due date of a job class as a multiple of its mean processing time, denoted by the factor $f_k$. The due date of an actual job is its arrival time advanced by the relative due date of its class. The weight $w_k$ of job class $k$ is defined as the probability that an arriving job is of class $k$, i.e. $\sum_{k=1}^{K} w_k = 1$. This way, the arrival process with rate $\lambda$ is logically split into $K$ subprocesses with rates $\lambda_k$ respectively; it can be shown that these subprocesses are in turn Poissonian, with rates $\lambda_k = w_k \lambda$ [3].

Table 1: Parameter Values of this Study

| $\mathbf{k}$ | $\mathbf{w_k}$ | $\boldsymbol{\lambda_k^{-1}}$ | $\boldsymbol{\mu_k^{-1}}$ | $\mathbf{f_k}$ |
|---|---|---|---|---|
| **1** | 0.75 | $0.75^{-1}$ | 0.90 | 11 |
| **2** | 0.20 | $0.20^{-1}$ | 0.95 | 13 |
| **3** | 0.05 | $0.05^{-1}$ | 1.50 | 15 |

The parameter values used in this study are given in table 1, where the mean interarrival time over all job classes is $\lambda^{-1} = \left( \sum_{k=1}^{K} \lambda_k \right)^{-1} = \underline{1}$. The system is stable regardless of the queueing discipline used, as the mean processing time over all job classes is $\mu^{-1} = \sum_{k=1}^{K} w_k \mu_k^{-1} = \underline{0.94}$. The rather high utilization of the system, which is $\sigma = \lambda/\mu = \underline{0.94}$, provides a sufficiently filled system for significant observations.

### On-Line Simulation (OSIM)

For every job, the following algorithm is applied:

Copy current system state $q \rightarrow$ queue $q'$
Insert job into $q'$ according to EDD
Simulate system with initial state $q'$
$\rightarrow$ Let $c$ be completion time of job

The simulation introduces parameters such as number and length of replications, confidence level, relative error and warmup period. Their values will be given when presenting the results as these are dependent from that values. In any case, the simulation is executed fast enough to avoid actual job arrivals during simulation, so jobs are never rejected because of inavailability of the queueing discipline OSIM being busy.

If the length of replications is not large enough, a job might not be completed during simulation; in such case, $c := \infty$.

The job is rejected to be processed if and only if $c > Due$, where $Due$ is the due date of the job.

With the given algorithm, OSIM may reject jobs for optimization purposes, as opposed to other queueing disciplines such as EDD. More specifically, OSIM prevents any estimated tardiness of newly arriving jobs. However, as a job is waiting in the queue, it may still experience tardiness due to enqueueing of new jobs in front of it, so the policy **does not** aim at

estimated performance $\tilde{T} = 0$ (optimality). While a modified algorithm could easily dequeue any waiting job predicted to experience tardiness (as estimated by simulation), such approach of dropping jobs initially accepted is of no general use in real-world scenarios as it is likely to reduce customer satisfaction.

The simulation based queueing discipline introduces another parameter, which denotes the queueing discipline assumed during simulation. As job arrivals occur during simulation, they have to be enqueued according to some discipline, just like in case of the real system. (Though simulated in this study, we may refer to the actually operated system as *real system*.) This way, we may even nest OSIM in itself. The Java code we have written for this study allows us in a very simple way to do such nesting up to any depth (for statistical correctness, with separate random number streams for any arrival process and any service process). The key question is whether and to what depth such nesting may sufficiently improve performance of the system. There is an intuitive idea of such improvement by nesting as performance estimation by on-line simulation should be much better when assuming the actual queueing discipline (OSIM) of the real system instead of some approximation (such as EDD, which is without any job rejection). The nesting of OSIM is terminated at some depth by assuming First Come, First Served (FCFS) as queueing discipline of the innermost OSIM; this way, the arrival process is effectively excluded from on-line simulation at this depth.

The major issue will be to treat nesting of on-line simulation analytically or numerically and to find that way applicable approaches for the computation of feasible and reasonable nesting depths before actual operation. The need for such approaches becomes evident facing the complexity of nested OSIM with respect to execution time, which increases exponentially according to the nesting depth $d$. Clearly, we seek for a trade-off between the expected performance gain, as achieved by application of nested OSIM, and the effort for executing such nesting.

For real-world complex systems, execution of nested on-line simulation is likely to become a tremendous task even for $d = 2$, questioning the purpose of any research on nesting. However, in any case where on-line simulation may be executed sufficiently fast (by means of existing or future speed-up techniques), the issue of nesting as well as the need for a solution becomes inevitable.

Table 2: Results of OSIM without Inclusion of Arrival Process

| $\mathbf{r_1}$ | $\mathbf{r_2}$ | $\mathbf{r_3}$ | $\mathbf{\bar{T}_1}$ | $\mathbf{\bar{T}_2}$ | $\mathbf{\bar{T}_3}$ | $\mathbf{\bar{T}}$ |
|---|---|---|---|---|---|---|
| 0.06 | 0.02 | 0.0 | 0.3 | 0.3 | 0.6 | 0.3 |

## EXPERIMENTAL RESULTS

We study the results observed with and without inclusion of the arrival process in the on-line simulation as well as with and without nesting of on-line simulation in order to discuss the impact of nesting on the results.

For an analytical treatment of OSIM, we need the distribution of the response time, as any response time less than the relative due date has to be excluded from consideration; however, there is no analytical solution yet even for the M/M/1-system, so we use simulation instead.

We verified our simulator of the real system with the help of some simple analytical results according to FCFS; as OSIM in turn is implemented using the very same simulator, all simulation results are considered as correct.

In any case, simulation of the real system is executed at confidence level 99% with a relative error limited to 10% (with respect to the performance measure). Its simulated operational lifetime is always large enough (fix at 10,000) to reach steady state, with the warmup period being 10% of that lifetime.

### EDD

Using EDD without any job rejection (without any on-line simulation) as queueing discipline, we get a performance of about 7.3.

### OSIM without Inclusion of Arrival Process

As any experiment involving OSIM takes much time to be completed, we limit the number of replications of any on-line simulation to 10; however, given the simplicity of the system, this limitation is not crucial to the results. The length of every replication of the on-line simulation is fix at 100; as the traces show, larger lengths are unnecessary.

The results are given in table 2, where $r_k$ is the percentage of rejected jobs of class $k$ related to all arriving jobs of that class.

**Result:** OSIM leads to dramatically improved performance as compared to EDD, with an overall im-

Table 3: Results of OSIM with Inclusion of Arrival Process Assuming EDD

| $r_1$ | $r_2$ | $r_3$ | $\bar{T}_1$ | $\bar{T}_2$ | $\bar{T}_3$ | $\bar{T}$ |
|---|---|---|---|---|---|---|
| 0.04 | 0.07 | 0.12 | 0.3 | 0.3 | 0.4 | 0.3 |

provement of about 96%.

In order to ensure that the improved performance is not just a consequence of random (uniform) rejection of jobs, we compare the results of OSIM with those of a modified version of EDD which is enhanced with uniform job rejection, where the probabilities of rejection are given by the percentages of rejection as observed when operating OSIM. The modified EDD results in a performance of about 2.3; we still note an improvement of performance when operating OSIM of about 87%.

### OSIM with Inclusion of Arrival Process

As discussed earlier, inclusion of the arrival process requires that the on-line simulation assumes some queueing discipline in order to enqueue the arriving jobs virtually. We present some results observed when assuming EDD as well as OSIM without inclusion of the arrival process, i.e., OSIM nested once.

#### ... Assuming EDD

The results are given in table 3.

**Result:** Interestingly, the performance remains unchanged (as compared to OSIM without inclusion of the arrival process), while the percentages of job rejection now are significantly higher (except for the main job class); in particular, while jobs of the third class are never rejected in the former case, we now note a considerable percentage of rejection, resulting in a reduced tardiness of that job class. While this has no effect on the overall tardiness (since jobs of the third class are rare), the result shows a significant difference between exclusion and inclusion of the arrival process with respect to job rejection, which can be crucial in real-world scenarios.

#### ... Assuming OSIM without Inclusion of Arrival Process

The results are given in table 4.

**Result:** In comparison to the former case of assuming EDD, the percentages of job rejection now are significantly lower among all job classes, resulting in a tardiness being reduced by 33%. The result shows a significant difference between assuming EDD (which is just an approximation for the actual queueing discipline) and assuming OSIM without inclusion of the arrival process (which is also just an approximation, however a better one).

Table 4: Results of OSIM with Inclusion of Arrival Process Assuming OSIM without Inclusion of Arrival Process

| $r_1$ | $r_2$ | $r_3$ | $\bar{T}_1$ | $\bar{T}_2$ | $\bar{T}_3$ | $\bar{T}$ |
|---|---|---|---|---|---|---|
| 0.03 | 0.03 | 0.01 | 0.2 | 0.2 | 0.2 | 0.2 |

### Preliminary Conclusion

The above results show that the overall tardiness depends not only on the percentages of job rejection but also on **which** jobs are rejected; OSIM nested once leads to a reduced tardiness while providing still fairly low percentages of job rejection.

## FUTURE WORK

We seek to find applicable approaches for the computation of feasible and reasonable nesting depths before actual operation. Promising solutions include analytical treatment of nesting by means of abstract models, numerical approximation by means of iteration schemes as well as metamodeling by means of self-referenced neural networks. We are currently working on these issues.

## CONCLUSION

On-line simulation is a promising approach for novel job acceptance/rejection policies; however, its success largely depends on the accuracy of approximation of the actual policy by the policy assumed during on-line simulation. Nesting of on-line simulation in itself due to recursive calls can dramatically improve accuracy, as we have shown in previous work. More generally speaking, nesting has a significant impact on the operation of a simulation-based policy, depending on the definition of performance measures and decision criteria, as we have illustrated in this work by means of a simple yet surprising example. However, as the effort for executing nesting of on-line simulation becomes tremendous for real-world complex systems, solutions must be found providing either feasible and reasonable bounds for the nesting depth or sufficient metamodels. This is considered to be a major challenge.

## REFERENCES


[1] T. Bessey. On-line simulation: Towards new statistical approaches. In *Proc. Summer Computer Simulation Conference (SCSC)*, 2003.

[2] T. Bessey. Application of on-line simulation to M/M/1-priority queueing. In *Proc. Summer Computer Simulation Conference (SCSC)*, 2004.

[3] G. Bolch, S. Greiner, H. d. Meer, and K. S. Trivedi. *Queueing Networks and Markov Chains*. Wiley, 1998.

[4] W. J. Davis. On-line simulation: Need and evolving research requirements. In J. Banks, editor, *Handbook of simulation*, chapter 13. Wiley, New York, 1998.

[5] G. R. Drake and J. S. Smith. Simulation system for real-time planning, scheduling, and control. In *Proc. Winter Simulation Conference (WSC)*, 1996.

[6] J. Esser, L. Neubert, J. Wahle, and M. Schreckenberg. Microscopic online simulation of urban traffic. In *Proc. 14th International Symposium on Transportation and Traffic Theory*, 1999.

[7] C. M. Harmonosky. Implementation issues using simulation for real-time scheduling, control, and monitoring. In *Proc. Winter Simulation Conference (WSC)*, 1990.

[8] C. M. Harmonosky, R. H. Farr, and M.-C. Ni. Selective rerouting using simulated steady state system data. In *Proc. Winter Simulation Conference (WSC)*, 1997.

[9] S. Manivannan and J. Banks. Real-time control of a manufacturing cell using knowledge-based simulation. In *Proc. Winter Simulation Conference (WSC)*, 1991.

[10] A. Nandi and P. Rogers. Simulation-based order acceptance in make-to-order manufacturing systems. In *Proc. Summer Computer Simulation Conference (SCSC)*, 2003.

[11] A. I. Sivakumar. Optimization of cycle time & utilization in semiconductor test manufacturing using simulation based, on-line near-real-time scheduling system. In *Proc. Winter Simulation Conference (WSC)*, 1999.