Generation of Mathematical Programming Representation From Discrete-Event Simulation Models

ARTICLE HISTORY
Compiled January 3, 2021

ABSTRACT
This work proposes a mathematical programming representation (MPR) of discrete event simulation (DES) implemented with event-scheduling logic. The MPR of DES is the foundation of developing white-box simulation optimization algorithms. The decision variables of the MPR are the event occurring times and the history of the state changes of the system, and the constraints represent the system dynamics. The equivalence of the solution of the proposed mathematical programming model and the simulation run using the same sample paths is validated with several cases. The MRP generation has been coded in an automatic procedure that only requires the user to input the state variables and the events of the DES model.

1. Introduction

Discrete Event Simulation (DES) is one of the most used tool for performance evaluation of complex systems and, hence, simulation–optimization algorithms are widely used when performance evaluation has to be coupled with optimization, i.e., when the best system configuration, according to some criteria, has to be found meanwhile guaranteeing a given value of some performance measure. Most of the state–of–the–art simulation–optimization algorithms consider DES as a *black–box* function, and the structure of DES models has been seldom studied. On the contrary, a minority of the simulation–optimization literature explores the structure of the DES models, and such approaches are referred to as *white–box* simulation–optimization. Under the black–box setting, simulation–optimization algorithms work in an iterative way, alternating simulation and optimization procedures, thus possibly leading to computational inefficiency if the number of iterations and/or the computation time per iteration increases too much. The benefit of white–box simulation–optimization is the saving of simulation budget due to the fact that the optimization procedure is guided by the information contained in the structure of the DES model. However, the barrier to the use of white–box simulation–optimization is modeling DES as white–box, so that it eventually favors optimization.

This work proposes a procedure to establish a white–box simulation model, which is an equivalent Mathematical Programming Representation (MPR) model, based on the well-known event–scheduling logic (Law, 2014). The procedure is applicable to certain types of DES models, and the assumptions that the DES model should satisfy are also presented in this work.

Chan and Schruben (2008) proposed a modeling framework to translate a DES model into an MPR model in a general sense. Their modeling framework is based on the Event Relationship Graph (ERG) of the system dynamics. To derive the MPR, an

ERG of the discrete event system has to be constructed and expanded to an elementary ERG (EERG) model, and a procedure can be applied to translate the EERG model into an MPR. However, this procedure has some limitations. First, deriving an ERG is not an easy task, and the user has to pay quite much attention to detect all the event relationships and complete the triggering conditions between each pair of related events. The difficulty of developing ERG limits the wide spread of this procedure. Second, the modeling procedure is case–by–case depending on the event relationships, which means that the user has to analyze the event relationships one by one and identify which type of modeling, including the variables and constraints, he/she should apply for each event relationship. This is quite difficult, since EERG is an expansion of ERG; the resulting graph could be huge and writing down the complete MPR model could be even impossible.

This work proposes a procedure that does not need the ERG and can be used to automatically generate the MPR in a general–purpose programming language. Despite being different, the MPRs proposed in this work and Chan and Schruben (2008) lead to equivalent results, which, in turn, are both equivalent to a simulation realization. The produre can also be applied to devise MRP of DES with event cancellation, which was not considered in the work of Chan and Schruben (2008).

The benefit of developing an MPR might be not obvious (especially when there is already a DES model) due to the extremely high complexity of solving it. However, this work does not suggest to solve the MPR directly with state-of-the-art mathematical programming solvers. Instead, running the DES results in the optimal solution of MPR, and the MPR provides representation of the structure of the DES. With the vast theoretical and methodological results developed in the mathematical programming (MP) field, for instance sensitivity analysis as proposed by Chan and Schruben (2008), the MPR of a simulation model favors the optimal design and control of the discrete event systems.

Many works in the literature show the potentiality of this research direction. For instance, the gradient can be conveniently estimated from the simulation model, if the MPR is approximated into Linear Programming (LP) and the dual can be conveniently obtained (Chan & Schruben, 2008; Zhang, Matta, & Alfieri, 2020). Moreover, if some of the parameters in the MPR are changed to decision variables, the MPR becomes an integrated simulation–optimization model. Solving the integrated model provides the optimal solution of the optimization problem (Matta, 2008). MP–based algorithms, such as linear programming approximation (Alfieri & Matta, 2012), Benders decomposition (Weiss & Stolletz, 2015), column generation (Alfieri, Matta, & Pastore, 2020), have been applied to improve the efficiency of integrated MP model solution.

The application of MPR–based simulation–optimization approaches is usually found in operations management of manufacturing and service systems. The flexibility of DES for complex system evaluation and of MPR for modeling optimization problems, allowed the integrated simulation-optimization approach to be effectively applied to buffer allocation problems (Zhang, Pastore, Matta, & Alfieri, n.d.), even with complex blocking mechanisms (Pedrielli, Alfieri, & Matta, 2015) and to problems involving real value decision variables (Tan, 2015; Zhang & Matta, 2020). Before the above mentioned works were proposed, there were many state–of–the–art heuristic approaches addressing those problems, but without any guarantee of global or local optimality. Thus, the development of MPR–based simulation–optimization has made its contribution in the research area of manufacturing and service system optimization.

The contributions of this work are several and refers to different aspects. First, it

proposes an MPR of DES from event-scheduling execution logic, which is the foundation of many DES implementations. Thus, the procedure does not require much extra effort once one has an event-scheduling execution logic implemented. Second, it proposes the MPR of event cancellation, which has never been studied in the literature. Third, the vast literature in mathematical programming field can be applied to the resulting MPR, for instance, for gradient estimation. Last, the proposed MPR can be easily transformed into an MPR for optimizing the design parameters of the DES, which are common optimization problems in operations management field.

The rest of the paper is organized as follows. Section 2 describes the generation of the MPR of a DES model, including a brief introduction of event-scheduling algorithm, the assumptions for applying the proposed procedure, the modeling steps requiring user manually work, and the MPR itself automatically generated based on the model. Section 3 shows several examples of DES and the generated MPR, whose equivalence has been validated. Result discussion and conclusions are reported in Section 4 and 5, respectively.

2. MPR generation procedure

2.1. Event-scheduling execution logic of DES

The event-scheduling approach is the logic behind all the major DES software and used by practitioners when developing simulation codes with general purpose languages (Law, 2014). For sake of completeness, the logic is briefly described. The fundamental elements are the *system state* and the *events*. The system state $\mathbf{s}$ is a set of state variables s to describe the system at a particular time. An event ξ is composed by a collection of four objects, $(CS^{\xi}(\mathbf{s}),\ CC^{\xi}(\mathbf{s}),\ F^{\xi}(\mathbf{s}),\ T^{\xi})$, where:

$$\begin{aligned} CS^{\xi}(\mathbf{s}): &\quad \text{condition to schedule} \\ CC^{\xi}(\mathbf{s}): &\quad \text{condition to cancel} \\ F^{\xi}(\mathbf{s}): &\quad \text{state transition function} \\ T^{\xi}: &\quad \text{delay between schedule and occurrence.} \end{aligned}$$

Condition to schedule and to cancel an event are logic expressions based on the system state $\mathbf{s}$, and the output will be true if the system state allows to schedule or cancel executions. The occurrence of an event changes the system state, which is indicated by the state transition function. The input and output of $F^{\xi}(\mathbf{s})$ are the system state before and after event occurrence, respectively. There is a delay, zero or positive, between scheduling time and occurring time, and this delay is called *occurrence delay* and denoted as T^{ξ} in the rest of the work. In this work, T^{ξ} is considered as random variate. Generally, each event usually occurs multiple times during a simulation run, and the term *execution* is introduced to specify a certain occurrence of an event. An execution is composed by a set of four objects, (ξ, i, τ_0, τ_1), where ξ denotes the event type, i denotes the index of execution, τ_0 denotes the scheduling time and τ_1 denotes the occurring time.

Given a set of events $\mathbb{X}$, initial system state $\mathbf{s}_0$, and simulation length defined as the number of iterations and denoted by K, a DES model could be represented with the event-scheduling logic, as in Algorithm 1. The algorithm is initialized with simulation clock equal to zero, number of executions scheduled in the past of each event equal to zero, and empty future event list. The *future event list* is a list of executions, which

are scheduled but not occur yet. Each iteration is composed by three steps, which are scheduling new executions, canceling executions and performing an execution. First, for each event ξ, if the condition to schedule ξ is true, a new execution of ξ is created and added to the future event list, as in lines 3 to 9. Then, for each execution in the future event list, if the condition to cancel the event is true, the execution is removed from the event list, as in lines 12 to 16. Finally, the execution with the earliest occurring time is taken from the future event list, the system state is changed according to the state transition function of the event type, and simulation clock is set to the occurring time. If there are more than one execution with the same earliest occurring time, the algorithm can randomly pick one among them or according to a priority rule.

Algorithm 1 Event-scheduling algorithm.

Input:
Set of events $\mathbb{X}$.
Initial system state: $\mathbf{s}_0$.
Simulation length: number of iterations K.
Simulation clock: $clock = 0$.
Number of scheduled executions of event ξ: $i^{\xi} = 0$.
FutureEventList $= \{\}$.

1: **for** k=0 to K-1 **do**
2: **Schedule new executions**
3: **for** each event $\xi \in \mathbb{X}$ **do**
4: **if** $CS^{\xi}(\mathbf{s}_k)$ **then**
5: $i^{\xi} = i^{\xi} + 1$
6: Sample occurring delay $t^{\xi}_{i^{\xi}}$ from random variate T^{ξ}
7: Add execution $(\xi, i^{\xi}, clock, clock + t^{\xi}_{i^{\xi}})$ to *FutureEventList*.
8: **end if**
9: **end for**
10:
11: **Cancel executions**
12: **for** each execution *exe*$\in$*FutureEventList* **do**
13: **if** $CC^{exe.event}(\mathbf{s}_k)$ **then**
14: Remove *exe* from *FutureEventList*
15: **end if**
16: **end for**
17:
18: **Perform the earliest execution**
19: From *FutureEventList*, take execution *exe*, which is with the earliest occurring time.
20: Update system state: $\mathbf{s}_{k+1} \leftarrow F^{exe.event}(\mathbf{s}_k)$.
21: $clock = exe.occurring\_time$.
22: **end for**

In the following, a procedure to translate DES models into MPR models, based on the event-scheduling logic, is introduced. Before presenting the procedure, the assumptions that the DES model has to satisfy in order to have the procedure applicable, are described.

2.2. Assumptions

To apply the procedure proposed in this work, the following assumptions must be satisfied.

(1) State variables are integer.
(2) For all the events ξ, the *condition to schedule* and the *condition to cancel* are in the form $a^{\xi,s} \leq s \leq c^{\xi,s}$, where $a^{\xi,s}$ and $c^{\xi,s}$ are integers and represent, respectively, the lower bound and the upper bound of a range, for a certain state variables s. When multiple state variables are involved, they are combined using the logical operator "AND".
(3) The occurring delay of executions of the same event ξ are independent and identically distributed.
(4) When more than one executions in the future event list have the same occurring time, the sequence of performing the executions is immaterial, i.e., different sequences lead to the same new system state and the same new event list when the simulation clock is advanced.
(5) Only events with strictly positive occurrence delay can be canceled.

The first assumption requires the state variables to be integer. Integer variables widely exist in DES models, such as number of jobs in buffers, idle servers, and binary variables to model system behavior and control. A discrete state variable can be translated into an integer variable or a set of binary variables. For real-valued state variables, they can be approximately discretized. Thus, the first assumption is fairly general.

The second assumption is, instead, more strict. However, if a model does not satisfy this condition, one can consider to introduce extra binary variables to satisfy assumption (2). For instance, if the condition to schedule event ξ is $s \leq a^{\xi,s}\ OR\ s \geq c^{\xi,s}$, a binary variable $s^{'}$ can be introduced in the model, and $s^{'}$ is equal to one if and only if $s \leq a^{\xi,s}\ OR\ s \geq c^{\xi,s}$. Thus, violation of assumption (2) can be overcome, even though at the cost of an increase of the model complexity.

As for the third assumption, the delay represents usually service or inter-arrival time. For example, if an event represents the finishing of a job on a machine, it is scheduled (i.e., put in the list) when the job starts its processing, and it will be executed when the jobs will be finished, i.e., the time between scheduling and execution is the job processing time. If the condition is not met, i.e., if the occurrence delay is not iid, it can be splitted into several events so that each event has iid delay. For instance, if the distribution of service time depends on the job type, the event of *finishing* a job should be splitted such that the *finishing* of each job type is represented by one event.

The fourth assumption, in practice, says that the execution time is the only attribute of priority for the event executions in the future event list. If one would like to specify a different priority for events having the same execution time, he/she can specify the priority by adding the event with lower priority to the event list after the execution of the event with higher priority, which can be done by introducing extra binary variables.

The fifth assumption is an extension of the fourth one. Assumption (4) implies that only events with positive delay can be canceled. In fact, assume that there is a zero-delay event ξ with cancellation. After an execution $(\xi, i, \tau_0, \tau_0, false)$ is added to the future event list in iteration k. It is obvious that the condition to schedule and the condition to cancel an event is not identical, thus, either the new execution must occur immediately if there is no other executions in the future event list having the same occurring time τ_0 and event cancellation is not relevant, or there are such executions

and two cases can occur. In one case, execution $(\xi, i, \tau_0, \tau_0, false)$ is immediately executed in iteration $(k+1)$. In another case, if some other executions with occurring time equal to τ_0 is executed earlier than execution $(\xi, i, \tau_0, \tau_0, false)$, and the condition to cancel ξ is true at a certain time, then the execution $(\xi, i, \tau_0, \tau_0, false)$ is canceled and never occurs. The simulation realization in the latter case is different from the former one, and assumption (4) is violated. Thus, only positive-delay event can be canceled.

2.3. Mathematical programming generation procedure

To implement the event-schedule algorithm and generate MPR, events and state variables should be first defined. For instance, to simulate a G/G/m system, three events and two state variables are essential. The three events represent job arrival, service start and service finish, and are denoted by arr, ss and sf, respectively. The two state variables represent the number of jobs in the queue and the number of occupied servers, and are denoted by q and g, respectively. State variable q is non-negative integer, and g can be any integer between 0 and m, which is the number of servers.

To implement the event-scheduling logic in Algorithm 1, for each event ξ, the condition to schedule CS^{ξ}, the condition to cancel CC^{ξ}, the distribution of occurrence delay T^{ξ} and the state transition function must be provided. Besides, to derive the MPR with the proposed approach, some modifications must be made to the general event-scheduling algorithm in Algorithm 1, but the resulting algorithm will be equivalent. Each event is categorized as zero-delay or positive-delay, depending on whether the occurrence delay T^{ξ} is equal to zero with probability equal to one or not. The modifications differ for zero-delay and positive-delay events.

In Algorithm 1, there may be more than one execution of the event ξ in the future event list. The first modification is that the number of executions of any zero-delay event in the future event list at any time is no more than one. Multiple executions are equivalent to sequentially schedule a new execution after the previous one occurs.

The second modification is about the scheduling of positive-delay events. A *counting event* $\tilde{\xi}$ and a counting variable u^{ξ} are artificially created. The counting event $\tilde{\xi}$ is zero-delay, with the same scheduling condition as ξ. The simulation logic is then modified as follows. When the system state satisfies the condition to schedule ξ, an execution of event $\tilde{\xi}$, other than ξ, is added to the future event list. Only after event $\tilde{\xi}$ occurs, an execution of event ξ is added to the future event list. The counting variable u^{ξ} represents the number of executions of ξ in the future event list, and its value is incremented by one when counting event $\tilde{\xi}$ occurs and decremented by one when event ξ occurs.

The third modification is about event cancellation. A fifth object named *cancel* is introduced to an *execution*, which has a binary value indicating if the execution is canceled. It is initialized as *false* (i.e., zero), when an execution is created. Once the condition to cancel executions of event type ξ is true, the value of *cancel* of those executions is changed to *true*, (i.e., one), the counting variable u^{ξ} is set to zero, but the execution is not removed from the future event list. When a canceled execution is taken from the future event list, the system state will remain unchanged and the algorithm will continue to the next iteration.

Based on the assumptions and the three modifications above, compositions of zero-delay and positive-delay events can be modified accordingly. Zero-delay events are composed by a collection of two objects ($CS^{\xi}(\mathbf{s})$, $F^{\xi}(\mathbf{s})$), which are condition to schedule and state transition. Positive-delay events are composed by five objects (T^{ξ}, $\tilde{\xi}$, u^{ξ},

$CC^{\xi}(\mathbf{s})$, $F^{\xi}(\mathbf{s})$), which are occurrence delay, counting event, counting variable, condition to cancel and state transition, respectively.

The last modification is that besides integer K, which represents the total number of iterations, the number of executions of each event ξ, denoted by N^{ξ}, should also be provided. It is not necessary that N^{ξ} is exactly equal to the number of executions of event ξ in the simulation run, instead, it could be an upper bound. Generally speaking, N^{ξ} can be equal to K, but the smaller the value, the smaller the number of variables, thus, a simpler model will be consequently developed. Since N^{ξ} is not the exact number of executions of event ξ, when the simulation terminates, the future event list sometimes can be not empty, i.e., some executions are scheduled but never occur.

The modified event-scheduling logic is shown in Algorithm 2. The difference between the two algorithms are the following. In each iterations, only zero-delay events are scheduled by validating the condition to schedule, as in line 3. Positive-delay events are scheduled after the execution of its counting event is performed, as in lines 26 to 30. Canceled executions are not removed from the future event list, but the attribute *cancel* is set to true, and the counting variable is set zero, as in line 13. When performing an execution, only the transition function of uncanceled executions will be applied, as in lines 21 to 23.

To summarize, to implement the modified event-scheduling logic, zero-delay events, including counting events of positive-delay events, must be provided together with the condition to schedule and the transition function, and positive-delay events must be provided together with occurrence delay, counting event, counting variable, condition to cancel and state transition function. An example of G/G/m queue is shown in Table 1. Since event cancellation is not relevant to G/G/m queue, the condition to cancel is not showed. Event arr is a positive-delay event, as explained above. A counting event of arr, i.e., $a\tilde{r}r$, and a counting variable u^{arr} are defined. The condition to schedule arr is that u^{arr} is equal to zero, since it is a renewal process, i.e., each time a job arrives, one and only one new arrival can be scheduled. When arr is executed, one job arrives to the system, and queue level q is increased by one. Event ss is a zero-delay event. The condition to schedule ss is that there is one job waiting in the queue and one server available. Upon execution of ss, queue level is reduced by one, and the number of occupied servers is increased by one. Event fs is a positive-delay event, it is scheduled immediately after ss is executed, and the delay until its execution is equal to the service time, i.e., sampled from T^{sf}. Therefore, event ss can be regarded as the counting event of sf. Furthermore, state variable g is the counting variable of sf.

To derive the MPR of a simulation run of any system, one just needs state variables, zero-delay events and positive delay events as in Table 1, number of iterations K and number of executions N^{ξ}. Then, the MPR is derived as in next sections.

2.4. Mathematical programming model

The MPR represents the dynamics of the simulated system, equivalent to the event-scheduling algorithm, i.e., Algorithm 2. Specifically, execution scheduling times, execution occurring times and state variable changes during the simulation run can all be seen in the MPR as decision variables. The scheduling times and the occurring times of the execution of event type ξ indexed by i are denoted by $e_i^{\xi,0}$ and $e_i^{\xi,1}$, respectively. Variables $\mathcal{E}_k$ denotes the simulation clock at the beginning of iteration k in Algorithm 2. Variables $e_i^{\xi,0}$, $e_i^{\xi,1}$ and $\mathcal{E}_k$ are all real-valued and non-negative. Variable u_k^s is used

Algorithm 2 Modified event-scheduling logic.

for k=0 to K-1 **do**
  Schedule new zero-delay executions
  for each zero-delay event $\xi \in \mathbb{X}$ **do**
    if $CS^{\xi}(\mathbf{s}_k)$ & there is no execution of event ξ in *FutureEventList* **then**
      $i^{\xi} = i^{\xi} + 1$
      Add execution $(\xi, i^{\xi}$,clock,clock, $false)$ to *FutureEventList*.
    end if
  end for

  Cancel executions
  for each positive-delay execution $exe \in$ *FutureEventList* **do**
    if $CC^{\xi}(\mathbf{s}_k)$ **then**
      $exe.cancel$ = true, $u^{\xi} = 0$.
    end if
  end for

  Perform an execution
  From *FutureEventList*, take execution exe, which is with the earliest occurring time.
  $clock = exe.occurring\_time$.
  if $T^{\xi} > 0$ **then**
    if $exe.cancel$ is false **then**
      Update system state: $\mathbf{s}_{k+1} \leftarrow F^{\xi}(\mathbf{s}_k)$.
    end if
  else
    Update system state: $\mathbf{s}_{k+1} \leftarrow F^{\xi}(\mathbf{s}_k)$.
    Schedule new positive-delay execution
    if ξ is the counting event of $\xi^{'}$ **then**
      Sample $t_i^{\xi^{'}}$ from random variate $T^{\xi^{'}}$.
      Add execution $(\xi^{'}, i$,clock,clock+$t_i^{\xi^{'}}, false)$ to *FutureEventList*.
    end if
  end if
end for

State variables

s	Description	Initial value
g	Number of occupied servers	0
q	Number of jobs in the queue	0
u^{arr}	Number of executions with event type arr in the future event list	0

Zero-delay events

ξ	Description	Condition to schedule	State transition
$a\tilde{r}r$	Counting arrival	$u^{arr} \leq 0$	u^{arr}++
ss	Start	$1 \leq q, g \leq m-1$	g++,q--

Positive-delay events

ξ	Description	Delay	Counting event	Counting variable	State transition
arr	Arrival	T^{arr}	$a\tilde{r}r$	u^{arr}	q++,u^{arr}--
sf	Finish	T^{sf}	ss	g	g--

Table 1. Events to simulate G/G/m system.

to denote the value of state variable s at the beginning of iteration k in Algorithm 2. Variables u_k^s are integer according to assumption (1). Some binary variables are also used in the MPR, and they will be introduced in the following, during the explanation of the model.

The set of all events ξ is denoted by $\mathbb{X}$. $\mathbb{I}^{\xi}$ denotes the set $\{1, ..., N^{\xi}\}$, which is the number of executions of event ξ, and $\mathbb{K}$ denotes the set $\{0, ..., K\}$, which is the total number of executions in the simulation run. $\mathbb{S}$ denotes the set of all state variables. $\mathbb{S}^{\xi}$ and $\mathbb{S}^{\bar{\xi}}$ denote the set of state variables relevant to scheduling and cancellation conditions of event type ξ, respectively. $\mathbb{S}^{c}$ denotes the set of counting variables.

2.4.1. Event execution

The first group of mathematical relationships, denoted by group-A, are the constraints for performing one execution in one iteration of Algorithm 2. Binary variables $w_{i,k}^{\xi}$ are introduced to represent the i-th execution of event type ξ occurring in iteration $k-1$. The i-th execution of event ξ is performed in iteration $k-1$ if and only if $w_{i,k}^{\xi}$ is equal to one, and variable $e_{i,1}^{\xi}$ is binded to $\mathcal{E}_k$, as shown in constraints (A1) and (A2). Constraints (A3) state that in each iteration, one and only one execution occurs. Constraints (A4) state that an execution occurs at most once. Constraints (A5) imply that the simulation cannot be reversed from iteration $k-1$ to iteration k. Constraint (A6) implies that the simulation clock is initialized to zero. Constraints (A7) implies that the delay between the scheduling and occurrence of an execution is equal to a

sample from the random variate T^{ξ}.

$$e_i^{\xi,1} - \mathcal{E}_k \geq M(w_{i,k}^{\xi} - 1) \quad \forall\, \xi \in \mathbb{X}, i \in \mathbb{I}^{\xi}, k \in \mathbb{K} \quad (A1)$$

$$\mathcal{E}_k - e_i^{\xi,1} \geq M(w_{i,k}^{\xi} - 1) \quad \forall\, \xi \in \mathbb{X}, i \in \mathbb{I}^{\xi}, k \in \mathbb{K} \quad (A2)$$

$$\sum_{k \in \mathbb{K}} w_{i,k}^{\xi} \leq 1 \qquad \forall\, \xi \in \mathbb{X}, i \in \mathbb{I}^{\xi} \qquad (A3)$$

$$\sum_{\xi \in \mathbb{X}} \sum_{i \in \mathbb{I}^{\xi}} w_{i,k}^{\xi} = 1 \qquad \forall\, k \in \mathbb{K} \qquad (A4)$$

$$\mathcal{E}_k - \mathcal{E}_{k-1} \geq 0 \qquad \forall\, k \in \mathbb{K} \qquad (A5)$$

$$\mathcal{E}_0 = 0 \qquad (A6)$$

$$e_i^{\xi,1} - e_i^{\xi,0} = t_i^{\xi} \qquad \forall \xi \in \mathbb{X}, i \in \mathbb{I}^{\xi} \qquad (A7)$$

2.4.2. Constraints for scheduling new events

The second group of constraints, denoted by group B, state the scheduling of new executions. Binary variables $x_{i,k}^{\xi}$ are used to represent that an execution with event type ξ and index i is scheduled in iteration k if it is equal to one. Constraints (B1) and (B2) shows that the scheduling time of the execution is equal to the simulation clock when it is scheduled.

$$e_i^{\xi,0} - \mathcal{E}_k \geq M(x_{i,k}^{\xi} - 1) \quad \forall\, \xi \in \mathbb{X}, k \in \mathbb{K}, i \in \mathbb{I}^{\xi} \quad (B1)$$

$$\mathcal{E}_k - e_i^{\xi,0} \geq M(x_{i,k}^{\xi} - 1) \quad \forall\, \xi \in \mathbb{X}, k \in \mathbb{K}, i \in \mathbb{I}^{\xi} \quad (B2)$$

To schedule zero-delay events, the condition to schedule must be verified, as stated in lines 2 to 8 in Algorithm 2, and it is also shown by constraints (B3) to (B11). Binary variables z_k^{ξ} represents that the system state satisfies the condition to schedule event ξ and all the previously scheduled executions have been performed, i.e., there is no execution of ξ in the future event list. In this case, it is mandatory to schedule an event ξ in iteration k in Algorithm 2, if z_k^{ξ} is equal to one. Constraints (B3) and (B4) imply that if z_k^{ξ} is equal to one, the state variables satisfy the condition to schedule execution of event type ξ. Moreover, a set of binary variables $v_k^{\xi,s,0}$ and $v_k^{\xi,s,1}$ are used to verify if the condition to schedule ξ is false. Specifically, constraints (B5) state that if $v_k^{\xi,s,0}$ is equal to one, u_k^s will be smaller than $a^{\xi,s}$, and hence, the inequality $a^{\xi,s} \leq s$ is violated. Similar to constraints (B6), if $v_k^{\xi,s,1}$ is equal to one, $s \leq c^{\xi,s}$ is violated.

$$u_k^s - a^{\xi,s} \geq M(z_k^{\xi} - 1) \quad \forall\, \xi \in \mathbb{X}, k \in \mathbb{K}, s \in \mathbb{S}^{\xi},\ T^{\xi} = 0 \quad (B3)$$

$$c^{\xi,s} - u_k^s \geq M(z_k^{\xi} - 1) \quad \forall\, \xi \in \mathbb{X}, k \in \mathbb{K}, s \in \mathbb{S}^{\xi},\ T^{\xi} = 0 \quad (B4)$$

$$(a^{\xi,s} - 1) - u_k^s \geq M(v_k^{\xi,s,0} - 1) \quad \forall\, \xi \in \mathbb{X}, k \in \mathbb{K}, s \in \mathbb{S}^{\xi},\ T^{\xi} = 0 \quad (B5)$$

$$u_k^s - (c^{\xi,s} + 1) \geq M(v_k^{\xi,s,1} - 1) \quad \forall\, \xi \in \mathbb{X}, k \in \mathbb{K}, s \in \mathbb{S}^{\xi},\ T^{\xi} = 0 \quad (B6)$$

Binary variable u_k^{ξ} is introduced to verify if there is an execution of event type ξ in the future event list at the beginning of iteration k. u_0^{ξ} is initialized to zero as in constraints (B8), showing that there is no execution in the future event list. Constraints (B7) shows that u_k^{ξ} is turned to one if an execution of event type ξ is

scheduled and turned to zero if an execution of event type ξ is performed in iteration $(k-1)$. Constraints (B9) specify that z_k^ξ is equal to zero only if at least one of the above mentioned conditions is violated, i.e., either the condition to schedule is violated, or there is already an execution in the future event list.

$$u_k^\xi = u_{k-1}^\xi - \sum_{i \in \mathbb{I}^\xi} w_{i,k}^\xi + z_{k-1}^\xi \quad \forall\, \xi \in \mathbb{X}, k \in \mathbb{K},\ T^\xi = 0 \quad (B7)$$

$$u_0^\xi = 0 \qquad \forall\, \xi \in \mathbb{X},\ T^\xi = 0 \qquad (B8)$$

$$1 - z_k^\xi \leq \sum_{s \in \mathbb{S}^\xi} v_k^{\xi,s,0} + \sum_{s \in \mathbb{S}^\xi} v_k^{\xi,s,1} + u_k^\xi \quad \forall\, \xi \in \mathbb{X}, k \in \mathbb{K},\ T^\xi = 0 \quad (B9)$$

Constraints (B10) show that if z_k^ξ is equal to one, one execution of event type ξ is scheduled in iteration k. Constraints (B11) state that executions of event type ξ have different indices.

$$\sum_{i \in \mathbb{I}^\xi} x_{i,k}^\xi = z_k^\xi \quad \forall\, \xi \in \mathbb{X}, k \in \mathbb{K},\ T^\xi = 0 \quad (B10)$$

$$\sum_{k \in \mathbb{K}} x_{i,k}^\xi \leq 1 \quad \forall\, \xi \in \mathbb{X}, i \in \mathbb{I}^\xi,\ T^\xi = 0 \quad (B11)$$

To schedule positive-delay events, an execution of the counting event must be performed. Constraints (B12) show that one execution of positive-delay event ξ is scheduled after each execution of its counting event $\tilde{\xi}$.

$$x_{i,k}^\xi = w_{i,k}^{\tilde{\xi}} \quad \forall\, \xi \in \mathbb{X}, i \in \mathbb{I}^\xi, T^\xi > 0 \quad (B12)$$

2.4.3. Constraints for sequencing executions

The index i of executions represents the scheduling sequence, i.e., if an execution is scheduled in an earlier iteration, then its index will be smaller. Moreover, an execution must by performed after being scheduled. Group-C constraints force such sequences, which are relevant to all events. Constraints (C1) show that if the i-th execution of event ξ is not scheduled before simulation termination, then the $(i+1)$-th execution will not be scheduled. Constraints (C2) state that if the i-th execution of event ξ is not scheduled before simulation termination, then it will not be performed. Constraints (C3) state that an execution cannot be performed before being scheduled, unless it remains in the future event list at the end of the simulation run. Constraints (C4) depict that the $(i+1)$-th execution of event ξ must be scheduled after the i-th execution, unless the $(i+1)$-th execution is not scheduled before simulation termination.

$$\sum_{k \in \mathbb{K}} x_{i+1,k}^\xi - \sum_{k \in \mathbb{K}} x_{i,k}^\xi \leq 0 \quad \forall\, \xi \in \mathbb{X}, i \in \mathbb{I}^\xi \quad (C1)$$

$$\sum_{k \in \mathbb{K}} w_{i,k}^\xi - \sum_{k \in \mathbb{K}} x_{i,k}^\xi \leq 0 \quad \forall\, \xi \in \mathbb{X}, i \in \mathbb{I}^\xi \quad (C2)$$

$$\sum_{k \in \mathbb{K}} k w_{i,k}^\xi - \sum_{k \in \mathbb{K}} k x_{i,k}^\xi \geq 1 + M(\sum_{k \in \mathbb{K}} w_{i,k}^\xi - 1) \quad \forall\, \xi \in \mathbb{X}, i \in \mathbb{I}^\xi \quad (C3)$$

$$\sum_{k \in \mathbb{K}} k x_{i+1,k}^\xi - \sum_{k \in \mathbb{K}} k x_{i,k}^\xi \geq 1 + M(\sum_{k \in \mathbb{K}} x_{i+1,k}^\xi - 1) \quad \forall\, \xi \in \mathbb{X}, i \in \mathbb{I}^\xi \quad (C4)$$

2.4.4. Constraints for event cancellation

The fourth group of constraints, denoted by group-D, state that executions of event type ξ in the future event list are canceled if the cancellation condition is true. Similar to constraints (B3) to (B6), constraint (D1) to (D4) show that if binary variables $z_k^{\bar{\xi}}$ are equal to one, then the cancellation condition of event ξ is true at the beginning of iteration k, where binary variables $z_k^{\bar{\xi}}$, $v_k^{\bar{\xi},s,0}$ and $v_k^{\bar{\xi},s,1}$ are the counter parts of z_k^{ξ}, $v_k^{\xi,s,0}$ and $v_k^{\xi,s,1}$, but for event cancellation rather than event scheduling. Constraints (D5) show that $z_k^{\bar{\xi}}$ can be equal to zero only if the cancellation condition is false.

$$u_k^s - a^{\bar{\xi},s} \geq M(z_k^{\bar{\xi}} - 1) \quad \forall\, \xi \in \mathbb{X}, k \in \mathbb{K}, s \in \mathbb{S}^{\bar{\xi}} \quad (D1)$$

$$c^{\bar{\xi},s} - u_k^s \geq M(z_k^{\bar{\xi}} - 1) \quad \forall\, \xi \in \mathbb{X}, k \in \mathbb{K}, s \in \mathbb{S}^{\bar{\xi}} \quad (D2)$$

$$(a^{\bar{\xi},s} - 1) - u_k^s \geq M(v_k^{\bar{\xi},s,0} - 1) \quad \forall\, \xi \in \mathbb{X}, k \in \mathbb{K}, s \in \mathbb{S}^{\bar{\xi}} \quad (D3)$$

$$u_k^s - (c^{\bar{\xi},s} + 1) \geq M(v_k^{\bar{\xi},s,1} - 1) \quad \forall\, \xi \in \mathbb{X}, k \in \mathbb{K}, s \in \mathbb{S}^{\bar{\xi}} \quad (D4)$$

$$1 - z_k^{\bar{\xi}} \leq \sum_{s \in \mathbb{S}^{\bar{\xi}}} v_k^{\bar{\xi},s,0} + \sum_{s \in \mathbb{S}^{\bar{\xi}}} v_k^{\bar{\xi},s,1} \quad \forall\, \xi \in \mathbb{X}, k \in \mathbb{K} \quad (D5)$$

The i-th execution of event type ξ is canceled if there exists an iteration k where the cancellation condition is true, and it is scheduled before and performed after iteration k. The i-th execution of event type ξ is scheduled in iteration $k_i^{\xi,0}$ and performed in iteration $(k_i^{\xi,1} - 1)$, where both $k_i^{\xi,0}$ and $k_i^{\xi,1}$ are integer variables. The value of $k_i^{\xi,0}$ and $k_i^{\xi,1}$ are calculated as in constraints (D6) and (D7). The i-th execution of event type ξ will be canceled if there exists k between $k_i^{\xi,0}$ and $k_i^{\xi,1}$ such that $z_k^{\bar{\xi}}$ is equal to one. Binary variables $\theta_{i,k}^{\xi}$ equal to one are used to represent the existence of such a k, which is guaranteed by constraints (D8) and (D9).

$$k_i^{\xi,1} = \sum_{k \in \mathbb{K}} k w_{k,i}^{\xi} \quad \forall\, \xi \in \mathbb{X}, i \in \mathbb{I}^{\xi} \quad (D6)$$

$$k_i^{\xi,0} = \sum_{k \in \mathbb{K}} k x_{k,i}^{\xi} \quad \forall\, \xi \in \mathbb{X}, i \in \mathbb{I}^{\xi} \quad (D7)$$

$$k z_k^{\bar{\xi}} - k_i^{\xi,0} - 1 \geq M(\theta_{i,k}^{\xi} - 1) \quad \forall\, \xi \in \mathbb{X}, i \in \mathbb{I}^{\xi}, k \in \mathbb{K} \quad (D8)$$

$$k_i^{\xi,1} - 1 - k z_k^{\bar{\xi}} \geq M(\theta_{i,k}^{\xi} - 1) \quad \forall\, \xi \in \mathbb{X}, i \in \mathbb{I}^{\xi}, k \in \mathbb{K} \quad (D9)$$

Also, binary variables $\phi_{i,k}^{\xi,0}$ and $\phi_{i,k}^{\xi,1}$ and constraints (D12) are used to avoid not canceling the executions when the condition to cancellation is true. Specifically, variables $\phi_{i,k}^{\xi,0}$ and $\phi_{i,k}^{\xi,1}$ equal to one represent if the condition to cancel is true in iteration k but it is earlier than the scheduling of e_i^{ξ} or later than performing e_i^{ξ}, respectively, as stated by constraints (D10) and (D11). Constraints (D12) states that $\theta_{i,k}^{\xi}$ is equal to

zero only if at least one between $\phi_{i,k}^{\xi,0}$ and $\phi_{i,k}^{\xi,1}$ is equal to one.

$$k_i^{\xi,0} - kz_k^{\bar{\xi}} \geq M(\phi_{i,k}^{\xi,0} - 1) \quad \forall\, \xi \in \mathbb{X}, i \in \mathbb{I}^{\xi}, k \in \mathbb{K} \quad (D10)$$

$$kz_k^{\bar{\xi}} - k_i^{\xi,1} \geq M(\phi_{i,k}^{\xi,1} - 1) \quad \forall\, \xi \in \mathbb{X}, i \in \mathbb{I}^{\xi}, k \in \mathbb{K} \quad (D11)$$

$$1 - \theta_{i,k}^{\xi} \leq \phi_{i,k}^{\xi,0} + \phi_{i,k}^{\xi,1} \quad \forall\, \xi \in \mathbb{X}, i \in \mathbb{I}^{\xi}, k \in \mathbb{K} \quad (D12)$$

As introduced in section 2.4.1, binary variable $w_{i,k}^{\xi}$ equal to one represents that the i-th execution of event ξ is performed in iteration k. Binary variable $\gamma_{i,k}^{\xi}$ is now introduced to indicate if that execution is canceled. $\gamma_{i,k}^{\xi}$ equal to one indicates that the i-th execution of event ξ is performed in iteration $(k-1)$ and the state transition function is applied. Specifically, if $w_{i,k}^{\xi}$ is equal to one, and it is not canceled, i.e., $\sum_{k' \in \mathbb{K}} \theta_{i,k'}^{\xi}$ is equal to zero, then $\gamma_{i,k}^{\xi}$ is equal to one. Otherwise, if $w_{i,k}^{\xi}$ is equal to zero, or $\sum_{k' \in \mathbb{K}} \theta_{i,k'}^{\xi}$ is greater than one, that execution is canceled, and $\gamma_{i,k}^{\xi}$ is equal to zero. Those relationships are stated by constraints (D13) and (D14).

$$\gamma_{i,k}^{\xi} \geq w_{i,k}^{\xi} - \sum_{k' \in \mathbb{K}} \theta_{i,k'}^{\xi} \quad \forall\, \xi \in \mathbb{X}, i \in \mathbb{I}^{\xi}, k \in \mathbb{K} \quad (D13)$$

$$w_{i,k}^{\xi} - \sum_{k' \in \mathbb{K}} \theta_{i,k'}^{\xi} - 1 \geq M(\gamma_{i,k}^{\xi} - 1) \quad \forall\, \xi \in \mathbb{X}, i \in \mathbb{I}^{\xi}, k \in \mathbb{K} \quad (D14)$$

2.4.5. Constraints for state evolution

The value of state variables u^s are changed from u_k^s to u_{k+1}^s in iteration k, as stated in lines 22 and 25 in Algorithm 2. Constraints (E1) represent the evolution of the state variables if it is not a counting variable. Specifically, if an execution of event type ξ is performed and not canceled in iteration k, the state variable s is changed by function $F^{\xi,s}(u_k^s)$. Constraints (E2) to (E4) show the evolution of the counting variable u^{ξ} of positive-delay events ξ. If $z_k^{\bar{\xi}}$, representing event cancellation, is equal to one, i.e., executions of event ξ are canceled in iteration k, u^{ξ} is set to zero, as in (E2). Otherwise, it is increased by one if a counting event $\tilde{\xi}$ is executed and decreased by one if event ξ itself is executed.

$$u_{k+1}^s = \sum_{\xi \in \mathbb{X}} \sum_{i \in \mathbb{I}^{\xi}} \gamma_{i,k+1}^{\xi} F^{\xi,s}(u_k^s) \qquad \forall\, s \in \mathbb{S}/\mathbb{S}^c, k \in \mathbb{K} \qquad (E1)$$

$$u_{k+1}^{\xi} \leq M(1 - z_k^{\bar{\xi}}) \quad \forall\, \xi \in \mathbb{X}, t^{\xi} > 0, k \in \mathbb{K} \quad (E2)$$

$$u_{k+1}^{\xi} \leq u_k^{\xi} - \sum_{i \in \mathbb{I}^{\xi}} \gamma_{i,k+1}^{\xi} + \sum_{i \in \mathbb{I}^{\xi}} \gamma_{i,k+1}^{\tilde{\xi}} + M z_k^{\bar{\xi}} \quad \forall\, \xi \in \mathbb{X}, t^{\xi} > 0, k \in \mathbb{K} \quad (E3)$$

$$u_{k+1}^{\xi} \geq u_k^{\xi} - \sum_{i \in \mathbb{I}^{\xi}} \gamma_{i,k+1}^{\xi} + \sum_{i \in \mathbb{I}^{\xi}} \gamma_{i,k+1}^{\tilde{\xi}} - M z_k^{\bar{\xi}} \quad \forall\, \xi \in \mathbb{X}, t^{\xi} > 0, k \in \mathbb{K} \quad (E4)$$

If all the events change the state variables with a fixed increment or decrement equal to $\Delta^{\xi,s}$, constraints (E1) will be changed to (E5), which are linear constraints,

and the MPR becomes a MILP.

$$u_{k+1}^{s} = u_{k}^{s} + \sum_{\xi \in \mathbb{X}} \sum_{i \in \mathbb{I}^{\xi}} \gamma_{i,k+1}^{\xi} \Delta^{\xi,s} \quad \forall\ s \in \mathbb{S}, k \in \mathbb{K} \quad (E5)$$

If event ξ cannot be canceled, variables $\gamma_{i,k}^{\xi}$ and group-D constraints are not introduced, and the variables $\gamma_{i,k}^{\xi}$ are replaced by $w_{i,k}^{\xi}$ in the constraints (E1) and (E5). Constraints (E2) to (E4) are replaced by (E6).

$$u_{k+1}^{\xi} = u_{k}^{\xi} - \sum_{i \in \mathbb{I}^{\xi}} w_{i,k+1}^{\xi} + \sum_{i \in \mathbb{I}^{\tilde{\xi}}} w_{i,k+1}^{\tilde{\xi}} \quad \forall\ k \in \mathbb{K} \quad (E6)$$

The initial value of each state variable s should be given, denoted by s_0, which is shown with constraints (E7).

$$u_{0}^{s} = s_0 \quad \forall\ s \in \mathbb{S} \quad (E7)$$

2.4.6. Objective function

With the constraints defined in the previous sections, there is a unique feasible solution in terms of event scheduling, execution times, history of simulation clock, $e_i^{\xi,0}$, $e_i^{\xi,1}$ and $\mathcal{E}_k$. Thus, the objective function can be any function of those variables, for instance, average of system time or waiting time in queueing systems. Multiple feasible solutions may appear in terms of binary variables, since the sequence of executions with identical execution time is not uniquely defined. However, this will not impact event execution time thanks to assumption (4).

The flexibility of the objective function definition is a main difference between the formulation proposed by Chan and Schruben (2008) and this work, since the objective function of MPR in Chan and Schruben (2008) can be only the sum of all execution times.

3. Applications

In this section, several examples are presented, including a G/G/m queue, a merge queueing system composed by three single server stations, and a single server queue with failure as an example of event cancellation. The equivalence of MPR solution and history of a simulation run has been validated with K equal to 20 for 100 independent replicates.

3.1. G/G/m queue

The first example is a G/G/m queue. Table 1 shows the state variables and events composing the DES model, and the detailed explanation of the state variables and events can be found in section 2.3. The MPR generated by the approach proposed in this work is as follows.

Group-A, group-C and (B1) (B2) constraints are identical for all systems and all

event type as presented in section 2.4.1, 2.4.2 and 2.4.3.

$$(A1)-(A7),(B1),(B2),(C1)-(C4) \qquad \forall \xi \in \{arr, a\tilde{r}r, ss, sf\}$$

Events $a\tilde{r}r$ and ss are zero-delay, so constraints (B3) to (B11) are applied. Events arr and sf are positive-delay, so constraints (B13) are applied. Specifically, constraints (B7), (B8), (B10) and (B11) are identical for all systems, and hence, for sake of simplicity, they are not expanded.

$$(B7),(B8),(B10),(B11) \qquad \forall \xi \in \{a\tilde{r}r, ss\}$$

Constraints (1) are constraints (B4) of the counting event of arrival, stating that if a counting event of arrival is scheduled, i.e., $z_k^{a\tilde{r}r}$ is equal to one, there must be no execution of arrival in the future event list. Constraints (2) imply that if $v_k^{a\tilde{r}r,arr,1}$ is equal to one, there must be no execution of event arr in the future event list, hence, at most one between $z_k^{a\tilde{r}r}$ and $v_k^{a\tilde{r}r,arr,1}$ can be equal to one. Constraints (3) are constraints (B9) of event $a\tilde{r}r$, and imply that if event $a\tilde{r}r$ is not scheduled, it is either because there is already an execution of arr or an execution of $a\tilde{r}r$ in the future event list.

$$1 - u_k^{arr} \geq z_k^{a\tilde{r}r} \qquad \forall k \in \mathbb{K} \tag{1}$$

$$u_k^{arr} \geq v_k^{a\tilde{r}r,arr,1} \qquad \forall k \in \mathbb{K} \tag{2}$$

$$1 - z_k^{a\tilde{r}r} \leq v_k^{a\tilde{r}r,arr,1} + u_k^{a\tilde{r}r} \qquad \forall k \in \mathbb{K} \tag{3}$$

Constraints (4) are constraints (B4) of event ss concerning state variable q, stating that if an execution of ss is scheduled, i.e., z_k^{ss} is equal to one, there must be at least one job waiting in the queue. Constraints (5) imply that if $v_k^{ss,q,0}$ is equal to one, there must be no job waiting in the queue, hence, at most one between z_k^{ss} and $v_k^{ss,q,0}$ can be equal to one. Constraints (6) are constraints (B4) of event ss concerning state variable g, stating that if an execution of ss is scheduled, there must be at least one server available. Constraints (7) imply that if $v_k^{ss,g,1}$ is equal to one, all the servers are occupied, hence, at most one between z_k^{ss} and $v_k^{ss,g,1}$ can be equal to one. Constraints (8) are constraints (B9) of event ss, and imply that if event ss is not scheduled, it is either because there is no job in the queue, or no server is available, or there is already an execution of ss in the future event list.

$$u_k^q \geq z_k^{ss} \qquad \forall k \in \mathbb{K} \tag{4}$$

$$-u_k^q \geq K(v_k^{ss,q,0} - 1) \qquad \forall k \in \mathbb{K} \tag{5}$$

$$m - u_k^g \geq z_k^{ss} \qquad \forall k \in \mathbb{K} \tag{6}$$

$$u_k^q \geq m v_k^{ss,g,1} \qquad \forall k \in \mathbb{K} \tag{7}$$

$$1 - z_k^{ss} \leq v_k^{ss,q,0} + v_k^{ss,g,1} + u_k^{ss} \qquad \forall k \in \mathbb{K} \tag{8}$$

Since arrival event arr and finish event sf are positive-delay, constraints (B12) should be applied, as it can be seen in constraints (9) and (10). An execution of arrival event is scheduled if a counting event $a\tilde{r}r$ is executed. An execution of finish

event is scheduled if a counting event ss is executed.

$$x_{i,k}^{arr} = w_{i,k}^{a\tilde{r}r} \qquad \forall i \in \mathbb{I}, k \in \mathbb{K} \tag{9}$$

$$x_{i,k}^{sf} = w_{i,k}^{ss} \qquad \forall i \in \mathbb{I}, k \in \mathbb{K} \tag{10}$$

Constraints (11) show the evolution of state variable u^{arr}, i.e., the number of execution of arr in the future event list. It is incremented by one if the counting event $a\tilde{r}r$ is executed, and decremented by one if the arrival event arr is executed, i.e., a job arrives. Constraints (12) show the evolution of state variable q, i.e., the queue level. It is incremented by one if a job arrives, and decremented by one if a job starts to be served. Constraints (13) show the evolution of state variable g, i.e., number of occupied servers. It is incremented by one if a job starts to be served, and decremented by one if a job is released. The rest of the model indicates the initialization and range of state variables.

$$u_k^{arr} = u_{k-1}^{arr} - \sum_{i \in \mathbb{I}} w_{i,k}^{arr} + \sum_{i \in \mathbb{I}} w_{i,k}^{a\tilde{r}r} \qquad \forall k \in \mathbb{K} \tag{11}$$

$$u_k^{q} = u_{k-1}^{q} - \sum_{i \in \mathbb{I}} w_{i,k}^{ss} + \sum_{i \in \mathbb{I}} w_{i,k}^{arr} \qquad \forall k \in \mathbb{K} \tag{12}$$

$$u_k^{g} = u_{k-1}^{g} - \sum_{i \in \mathbb{I}} w_{i,k}^{fs} + \sum_{i \in \mathbb{I}} w_{i,k}^{ss} \qquad \forall k \in \mathbb{K} \tag{13}$$

$$u_0^{arr} = u_0^{q} = u_0^{g} = 0$$

$$u_k^{arr} \in \{0,1\},\ u_k^{g} \in \{0,...,m\},\ u_k^{q} \in \mathbb{N}$$

Table 2 shows a simulation run of a G/G/m queue including 10 executions. In iteration zero, the system is in initial state, and this state satisfies the condition to schedule the counting event of arrival, as in lines 2 to 8 in Algorithm 2. In the solution of MPR, the scheduling of $a\tilde{r}r$ is represented by $z_0^{a\tilde{r}r}$ and $x_{1,0}^{a\tilde{r}r}$ equal to 1, and its occurring time is equal to the simulation clock, which is equal to 0. The execution $(a\tilde{r}r, 1, e_1^{a\tilde{r}r,0}, e_1^{a\tilde{r}r,1})$, which is denoted by $e_1^{a\tilde{r}r}$ in the table for saving the space, is then performed since it is the earliest execution (line 18 in Algorithm 2), which is also indicated by $w_{1,1}^{a\tilde{r}r}$ equal to one, and simulation clock in the next iteration, i.e., iteration 1, is equal to the occurring time $e_{1,1}^{a\tilde{r}r}$ in the solution of MPR. After that the positive-delay event arr is then scheduled (lines 26 to 30 in Algorithm 2), which can also be seen from the MPR solution, since $x_{1,1}^{arr}$ is equal to one, the scheduling time of $e_{1,0}^{arr}$ is equal to the current simulation clock time and the occurrence time is 2.3 time unit later than the scheduling time, where 2.3 is sampled from the distribution of inter-arrival time T^{arr}. At the end of the iteration, the system state is update according to the state transition function of event $a\tilde{r}r$ (line 25 in Algorithm 2), which in the MPR solution is indicated by an increment equal to one of state variable u^{arr}. This procedure is repeated in the next iterations, and the equivalence of DES procedure and the solution of MPR can be seen from Table 2.

Table 2. Simulation run and MPR solution of G/G/m queue.

k	*clock*	New zero-delay executions	Future event list	Perform execution	New positive-delay execution	System state
0	0	$e_1^{a\tilde{r}r} = (a\tilde{r}r, 1, 0, 0)$	$\{e_1^{a\tilde{r}r}\}$	$e_1^{a\tilde{r}r}$	$e_1^{arr} = (arr, 1, 0, 2.3)$	$(u^{arr}, q, g) = (1, 0, 0)$
	$\mathcal{E}_0 = 0$	$z_0^{a\tilde{r}r} = 1, x_{1,0}^{a\tilde{r}r} = 1, e_{1,1}^{a\tilde{r}r} = 0$		$w_{1,1}^{a\tilde{r}r} = 1, \mathcal{E}_1 = e_{1,1}^{a\tilde{r}r}$	$x_{1,1}^{arr} = 1, e_{1,1}^{arr} = 0, e_{1,1}^{arr} = 2.3$	$u_1^{arr} = 1, u_1^q = 0, u_1^g = 0$
1	0		$\{e_1^{arr}\}$	e_1^{arr}		$(u^{arr}, q, g) = (0, 1, 0)$
	$\mathcal{E}_1 = 0$			$w_{1,2}^{arr} = 1, \mathcal{E}_2 = e_{1,1}^{arr}$		$u_2^{arr} = 0, u_2^q = 1, u_2^g = 0$
2	2.3	$e_2^{a\tilde{r}r} = (a\tilde{r}r, 2, 2.3, 2.3)$ $e_1^{ss} = (ss, 1, 2.3, 2.3)$	$\{e_2^{a\tilde{r}r}, e_1^{ss}\}$	$e_2^{a\tilde{r}r}$	$e_2^{arr} = (arr, 2, 2.3, 11.1)$	$(u^{arr}, q, g) = (1, 1, 0)$
	$\mathcal{E}_2 = 2.3$	$z_2^{a\tilde{r}r} = 1, x_{2,2}^{a\tilde{r}r} = 1, e_{2,1}^{a\tilde{r}r} = 2.3$ $z_2^{ss} = 1, x_{1,2}^{ss} = 1, e_{1,1}^{ss} = 2.3$		$w_{2,3}^{a\tilde{r}r} = 1, \mathcal{E}_3 = e_{2,1}^{a\tilde{r}r}$	$x_{2,3}^{arr} = 1, e_{2,0}^{arr} = 2.3, e_{2,1}^{arr} = 11.1$	$u_3^{arr} = 1, u_3^q = 1, u_3^g = 0$
3	2.3		$\{e_1^{ss}, e_2^{arr}\}$	e_1^{ss}	$e_1^{sf} = (sf, 1, 2.3, 6.0)$	$(u^{arr}, q, g) = (1, 0, 1)$
	$\mathcal{E}_3 = 2.3$			$w_{1,4}^{ss} = 1, \mathcal{E}_4 = e_{1,1}^{ss}$	$x_{1,4}^{sf} = 1, e_{1,0}^{sf} = 2.3, e_{1,1}^{sf} = 6.0$	$u_4^{arr} = 1, u_4^q = 0, u_4^g = 1$
4	2.3		$\{e_1^{sf}, e_2^{arr}\}$	e_1^{sf}		$(u^{arr}, q, g) = (1, 0, 0)$
	$\mathcal{E}_4 = 2.3$			$w_{1,5}^{sf} = 1, \mathcal{E}_5 = e_{1,1}^{sf}$		$u_5^{arr} = 1, u_5^q = 0, u_5^g = 0$
5	6.0		$\{e_2^{arr}\}$	e_2^{arr}		$(u^{arr}, q, g) = (0, 1, 0)$
	$\mathcal{E}_5 = 2.3$			$w_{2,6}^{arr} = 1, \mathcal{E}_6 = e_{2,1}^{arr}$		$u_6^{arr} = 0, u_6^q = 1, u_6^g = 0$
6	11.1	$e_3^{a\tilde{r}r} = (a\tilde{r}r, 3, 11.1, 11.1)$ $e_2^{ss} = (ss, 2, 11.1, 11.1)$	$\{e_3^{a\tilde{r}r}, e_2^{ss}\}$	$e_3^{a\tilde{r}r}$	$e_3^{arr} = (arr, 3, 11.1, 12.1)$	$(u^{arr}, q, g) = (1, 1, 0)$
	$\mathcal{E}_6 = 11.1$	$z_6^{a\tilde{r}r} = 1, x_{3,6}^{a\tilde{r}r} = 1, e_{3,1}^{a\tilde{r}r} = 11.1$ $z_6^{ss} = 1, x_{2,6}^{ss} = 1, e_{2,1}^{ss} = 11.1$		$w_{3,7}^{a\tilde{r}r} = 1, \mathcal{E}_7 = e_{3,1}^{a\tilde{r}r}$	$x_{3,7}^{arr} = 1, e_{3,0}^{arr} = 11.1, e_{3,1}^{arr} = 12.1$	$u_7^{arr} = 1, u_7^q = 1, u_7^g = 0$
7	11.1		$\{e_2^{ss}, e_3^{arr}\}$	e_2^{ss}	$e_2^{sf} = (sf, 2, 11.1, 21.8)$	$(u^{arr}, q, g) = (1, 0, 1)$
	$\mathcal{E}_7 = 11.1$			$w_{2,8}^{ss} = 1, \mathcal{E}_8 = e_{2,1}^{ss}$	$x_{2,8}^{sf} = 1, e_{2,0}^{sf} = 11.1, e_{1,1}^{sf} = 21.8$	$u_8^{arr} = 1, u_8^q = 0, u_8^g = 1$
8	11.1		$\{e_3^{arr}, e_2^{sf}\}$	e_3^{arr}		$(u^{arr}, q, g) = (0, 1, 1)$
	$\mathcal{E}_8 = 11.1$			$w_{3,9}^{arr} = 1, \mathcal{E}_9 = e_{3,1}^{arr}$		$u_9^{arr} = 0, u_9^q = 1, u_9^g = 1$
9	12.1	$e_4^{a\tilde{r}r} = (a\tilde{r}r, 4, 12.1, 12.1)$ $e_3^{ss} = (ss, 3, 12.1, 12.1)$	$\{e_4^{a\tilde{r}r}, e_3^{ss}, e_2^{sf}\}$	$e_4^{a\tilde{r}r}$	$e_4^{arr} = (arr, 4, 12.1, 17.3)$	$(u^{arr}, q, g) = (1, 1, 1)$
	$\mathcal{E}_9 = 12.1$	$z_9^{a\tilde{r}r} = 1, x_{4,9}^{a\tilde{r}r} = 1, e_{4,1}^{a\tilde{r}r} = 12.1$ $z_9^{ss} = 1, x_{3,9}^{ss} = 1, e_{3,1}^{ss} = 12.1$		$w_{4,10}^{a\tilde{r}r} = 1, \mathcal{E}_{10} = e_{4,1}^{a\tilde{r}r}$	$x_{4,10}^{arr} = 1, e_{4,0}^{arr} = 12.1, e_{4,1}^{arr} = 17.3$	$u_{10}^{arr} = 1, u_{10}^q = 1, u_{10}^g = 1$

3.2. Single-server merge

A queueing system composed of three servers in a merge architecture (Figure 1) is presented in this section. Jobs enter the system at server 1 or server 2, and the buffers in front of the two servers have infinite capacity. It is assumed that all the jobs arrive at time zero. After processing a job, server 1 or 2 can release the job to the buffer with capacity Q in front of server 3, denoted as buffer 3, if the buffer is not full. The blocking policy is blocking-after-service. If there is only one space available in the buffer, and both servers 1 and 2 are holding a finished job, server 1 will release the job. Server 3 can take jobs from buffer 3 and release the job immediately after the process is finished.

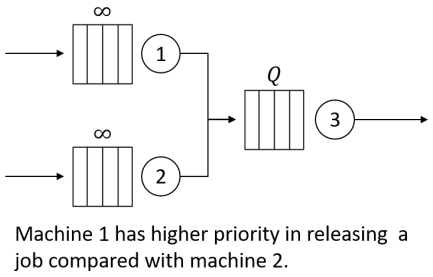


Figure 1. Example: single server merge.

State variables have to be defined in the first place, as in Table 3. Binary variable g^j for $j = 1, 2, 3$ is equal to one if server j is processing a job, otherwise it is equal to zero. Binary variable b^j for $j = 1, 2$ is equal to one if server j is holding a finished job, otherwise it is equal to zero. State variable b^3 is not defined, since the job will be released immediately after its processing. Integer variable q represents the number of jobs waiting in buffer 3.

The events composing the DES model are then defined as in Table 3. Event (ss, j) for $j = 1, 2, 3$ represents the starting of service of server j. If server j is idle, i.e., neither working nor holding a finished part, (ss, j) will be scheduled to execute immediately. After an (ss, j) is executed, server j becomes active, i.e., state variable g^j is incremented by one. Besides, if server 3 starts a job, the buffer level q is decreased by one. Event (d, j) for $j = 1, 2$ represents the departure of a job from server j. A $(d, 1)$ can be executed if server 1 is holding a finished job and there is at least one space available in buffer 3. A $(d, 2)$ can be executed immediately if server 2 is holding a finished job, server 1 is not holding a job and there is at least one space available in buffer 3. After executing (d, j), the server is not holding any job, hence, state variable b^j becomes zero, and the buffer level q is increased by one. Both (ss, j) and (d, j) can be executed immediately once scheduled, so they are zero-delay. Event (sf, j) represents that server j finishes a job, and it is scheduled immediately after (ss, j) is executed, and the occurrence delay is equal to the service time, i.e., sampled from T^j. Thus, (ss, j) is a counting event of (sf, j). After event (sf, j) is executed, server j is no longer working, thus, g^j is decreased by one. Thus, state variable g^j is a counting variable of (sf, j), since its value is incremented by one if (ss, j) is executed, and decremented by one if (sf, j) is executed. Specially for $j = 1, 2$, after a job is finished, the server holds a finished job, and state variable b^j is increased by one.

The MPR proposed in this work is as follows.

Group-A, group-C and (B1) (B2) constraints are identical for all systems and all

State variables

s	Description	Initial value
g^j	Equal to 1 if server j is working, otherwise equal to 0.	0
b^j	Equal to 1 if server j is blocked, otherwise equal to 0.	0
q	Number of jobs in the queue.	0

Zero-delay events

ξ	Description	Condition to schedule	State transition
$(ss,1)$	Start m1	$g^1 \leq 0$ & $b^1 \leq 0$	g^1++
$(d,1)$	Depart m1	$b^1 \geq 1$ & $q \leq Q-1$	b^1--, q++
$(ss,2)$	Start m2	$g^2 \leq 0$ & $b^2 \leq 0$	g^2++
$(d,2)$	Depart m2	$b^2 \geq 1$ & $q \leq Q-1$ & $b^1 \leq 0$	b^2--, q++
$(ss,3)$	Start m3	$g^3 \leq 0$ & $q \geq 1$	g^3++, q--

Positive-delay events

ξ	Description	Delay	Counting event	Counting variable	State transition
$(sf,1)$	Finish m1	$T^{sf,1}$	$(ss,1)$	g^1	g^1--, b^1++
$(sf,2)$	Finish m2	$T^{sf,2}$	$(ss,2)$	g^2	g^2--, b^2++
$(sf,3)$	Finish m3	$T^{sf,3}$	$(ss,3)$	g^3	g^3--

Table 3. Events to simulate Single-server merge system.

event type as presented in section 2.4.1, 2.4.2 and 2.4.3.

$$(A1)-(A7),(B1),(B2),(C1)-(C4) \qquad \begin{array}{l} \forall \xi \in \{(ss,1),(ss,2),(ss,3),(d,1),(d,2)\} \\ \forall \xi \in \{(sf,1),(sf,2),(sf,3)\} \end{array}$$

Events (ss,j) for $j=1,2,3$ and (d,j) for $j=1,2$ are all zero-delay, constraints (B3) to (B11) should be applied. Specifically, constraints (B7), (B8), (B10) and (B11) are identical for all systems, and for sake of simplicity, they are not expanded.

$$(B7),(B8),(B10),(B11) \qquad \forall \xi \in \{(ss,1),(ss,2),(ss,3),(d,1),(d,2)\}$$

Constraints (14) and (16), referring to constraints (B4), imply that if an execution of (ss,j) is scheduled, machine j is not occupied by a job under processing or finished. Constraints (15) and (17), referring to constraints (B6), imply that if variable $v_k^{ss,j,g^j,1}$ or $v_k^{ss,j,b^j,1}$ is equal to one, machine j is occupied by a job under processing or a finished job. Constraints (18), referring to constraints (B9), state that if execution of event $(ss,1)$ or $(ss,2)$ is not scheduled, it is either because the machine is occupied,

or there is already an execution in the future event list.

$$-u_k^{g^j} \geq z_k^{ss,j} - 1 \qquad \forall\ j = 1,2, k \in \mathbb{K} \tag{14}$$
$$u_k^{g^j} \geq v_k^{ss,j,g^j,1} \qquad \forall\ j = 1,2, k \in \mathbb{K} \tag{15}$$
$$-u_k^{b^j} \geq z_k^{ss,j} - 1 \qquad \forall\ j = 1,2, k \in \mathbb{K} \tag{16}$$
$$u_k^{b^j} \geq v_k^{ss,j,b^j,1} \qquad \forall\ j = 1,2, k \in \mathbb{K} \tag{17}$$
$$1 - z_k^{ss,j} \leq v_k^{ss,j,g^j,1} + v_k^{ss,j,b^j,1} + u_k^{ss,j} \qquad \forall\ j = 1,2, k \in \mathbb{K} \tag{18}$$

Similarly, constraints (19) to (23) refer to constraints (B3) to (B6) and (B9) of event $(ss, 3)$, stating that an execution of event $(ss, 3)$ can be scheduled if and only if machine 3 is not occupied, there is a job waiting in buffer 3, there is no executions of $(ss, 3)$ in the future event list.

$$-u_k^{g^3} \geq z_k^{ss,3} - 1 \qquad \forall\ k \in \mathbb{K} \tag{19}$$
$$u_k^{g^3} \geq v_k^{ss,3,g^3,1} \qquad \forall\ k \in \mathbb{K} \tag{20}$$
$$u_k^{q} \geq z_k^{ss,3} \qquad \forall\ k \in \mathbb{K} \tag{21}$$
$$-u_k^{q} \geq Q(v_k^{ss,3,q,0} - 1) \qquad \forall\ k \in \mathbb{K} \tag{22}$$
$$1 - z_k^{ss,3} \leq v_k^{ss,3,g^3,1} + v_k^{ss,3,q,0} + u_k^{ss,3} \qquad \forall\ k \in \mathbb{K} \tag{23}$$

Constraints (24) to (28) refer to constraints (B3) to (B6) and (B9) of event $(d, 1)$, implying that an execution of event $(d, 1)$ can be scheduled if and only if there is a finished job in machine 1, there is space available in buffer 3, there is no executions of $(d, 1)$ in the future event list.

$$u_k^{b^1} \geq z_k^{d,1} \qquad \forall\ k \in \mathbb{K} \tag{24}$$
$$-u_k^{b^1} \geq v_k^{d,1,b^1,0} - 1 \qquad \forall\ k \in \mathbb{K} \tag{25}$$
$$Q - u_k^{q} \geq z_k^{d,1} \qquad \forall\ k \in \mathbb{K} \tag{26}$$
$$u_k^{q} \geq Q v_k^{d,1,q,1} \qquad \forall\ k \in \mathbb{K} \tag{27}$$
$$1 - z_k^{d,1} \leq v_k^{d,1,b^1,0} + v_k^{d,1,q,1} + u_k^{d,1} \qquad \forall\ k \in \mathbb{K} \tag{28}$$

Constraints (29) to (35) refer to constraints (B3) to (B6) and (B9) of event $(d, 2)$, depicting that an execution of event $(d, 2)$ can be scheduled if and only if there is a finished job in machine 2, there is space available in buffer 3, there is no finished job waiting in machine 1, there is no executions of $(d, 2)$ in the future event list.

$$u_k^{b^2} \geq z_k^{d,2} \qquad \forall\ k \in \mathbb{K} \tag{29}$$
$$-u_k^{b^2} \geq v_k^{d,2,b^2,0} - 1 \qquad \forall\ k \in \mathbb{K} \tag{30}$$
$$Q - u_k^{q} \geq z_k^{d,2} \qquad \forall\ k \in \mathbb{K} \tag{31}$$
$$u_k^{q} \geq Q v_k^{d,2,q,1} \qquad \forall\ k \in \mathbb{K} \tag{32}$$
$$-u_k^{b^1} \geq z_k^{d,2} - 1 \qquad \forall\ k \in \mathbb{K} \tag{33}$$
$$u_k^{b^1} \geq v_k^{d,2,b^1,1} \qquad \forall\ k \in \mathbb{K} \tag{34}$$
$$1 - z_k^{d,2} \leq v_k^{d,2,b^2,0} + v_k^{d,2,q,1} + v_k^{d,2,b^1,1} + u_k^{d,2} \qquad \forall\ k \in \mathbb{K} \tag{35}$$

Since events (sf, j) are positive-delay, constraints (B12) should be applied, as in constraints (36). A finishing event of machine j can be scheduled after a starting event is executed.

$$x_{i,k}^{sf,j} = w_{i,k}^{ss,j} \qquad \forall\ j = 1,2,3, i \in \mathbb{I}^j, k \in \mathbb{K} \tag{36}$$

Constraints (37) to (39) indicate the evolution of state variables g^j, b^j and q, respectively. g^j is increased by one if machine j starts a job and decreased by one if it finished a job. b^j is increased by one if machine j finishes a job, and decreased by one if it releases a job. q is increased by one if machine 1 or 2 releases a job, and decreased by one if machine 3 seizes a job from buffer 3. The rest of the model indicates the initialization and range of state variables.

$$u_k^{g^j} = u_{k-1}^{g^j} + \sum_{i \in \mathbb{I}^j} w_{i,k}^{ss,j} - \sum_{i \in \mathbb{I}^j} w_{i,k}^{sf,j} \qquad \forall\ j = 1,2,3, k \in \mathbb{K} \tag{37}$$

$$u_k^{b^j} = u_{k-1}^{b^j} + \sum_{i \in \mathbb{I}^j} w_{i,k}^{sf,j} - \sum_{i \in \mathbb{I}^j} w_{i,k}^{d,j} \qquad \forall\ j = 1,2, k \in \mathbb{K} \tag{38}$$

$$u_k^q = u_{k-1}^q + \sum_{j=1,2} \sum_{i \in \mathbb{I}^j} w_{i,k}^{d,j} - \sum_{i \in \mathbb{I}^3} w_{i,k}^{ss,3} \qquad \forall\ k \in \mathbb{K} \tag{39}$$

$$u_0^{g^1} = u_0^{g^2} = u_0^{g^3} = u_0^{b^1} = u_0^{b^2} = u_0^q = 0$$

$$u_k^{g^1}, u_k^{g^2}, u_k^{g^3}, u_k^{b^1}, u_k^{b^2} \in \{0,1\},\ u_k^q \in \{0, ..., Q\} \qquad \forall\ k \in \mathbb{K}$$

3.3. G/G/1 with failure

A G/G/1 queueing system with unreliable server is studied in this section. The server has two states, up and down, respectively. When the server is in up state, it can process a job. When the server turns to down state from up state, the repair starts immediately, and if the server is processing a job, the job is discarded. After the repair finishes, the server turns to up state, and restarts to process new jobs. The repair time follows a general distribution. The server will then keep in up state for a certain time called up time, following a general distribution, and will fail again. The failure is time-dependent. Jobs arrive at the system following a general arrival process and enter the queue in front of the server. When the server is in up state and idle, the server can take a job from the queue and start the service. After the service, a job can be immediately released from the system.

As in Table 4, the state variables include integer variable q representing the number of jobs in the queue, binary variable g indicating whether the server is working or not, binary variable h indicating whether the server is in down state or not.

The events composing the DES model are then defined as in Table 4. Event arr indicates that a job arrives at the system, and it is scheduled after a previous arrival, with a delay equal to the inter-arrival time. A counting event of arr, i.e., $a\tilde{r}r$, and a counting variable u^{arr} should be defined. The condition to schedule an arrival is that there is no execution of arr in the future event list. When $a\tilde{r}r$ is executed, the counting variable u^{arr}, i.e., the number of executions of arr in the future event list, will be increased by one. When arr is executed, the number of jobs in the queue will be increased by one, and the counting variable u^{arr} will be decreased by one. Event ss represents that the server starts to process a job. If and only if there is at least one job in the queue, the server is in up state and idle, event ss is scheduled. When

event ss is executed, the number of jobs in the queue is reduced by one, and the server starts to work, i.e., state variable g becomes one. Event sf represents that a server finishes a job, and one execution of it is scheduled each time a job is started, and the occurrence delay is equal to the service time of the job. Event ss is the counting event, and state variable g is the counting variable of sf. After sf is executed, the server becomes idle. Once the server is in down state, job under process is discarded, i.e., scheduled execution of sf is canceled. Event fl represents that the state of the server becomes down, and it is scheduled after the previous repair finishes with delay equal to the up time. Since it is positive-delay, a counting event $\tilde{fl}$ should be added, and also a counting variable u^{fl}. The condition to schedule an $\tilde{fl}$ is that the server is in up state, i.e., h equal to zero, and there is no execution of fl in the future event list, i.e., u^{fl} smaller than or equal to zero. When $\tilde{fl}$ is executed, the number of executions of fl is increased by one. When fl is executed, the server is in down state, it cannot be working, i.e., g equal to zero, and its number of executions in the future event list is reduced by one. Event srp represents start of repair, and it is scheduled if the server is in down state and not under repair, and executed with no delay. When it is executed, the server is under repair. Event frp states the finish of repair, and it is scheduled after event srp is executed, hence, srp is the counting event, and state variable u^{frp} is introduced as the counting variable. After it is executed, the server becomes up, and no longer under repair.

The MPR proposed in this work is as follows.

Constraints (A1)-(A7), (C1)-(C4) and (B1) (B2) are applied to all events. Events $a\tilde{r}r$, ss, $\tilde{fl}$ and srp are all zero-delay, constraints (B3) to (B11) should be applied, and constraints (B12) should be applied to events arr, sf, fl and frp. Those constraints have been explained in the previous two examples, and then this example will not expand them.

$$\begin{aligned} (A1)-(A7),(B1),(B2),(C1)-(C4) \quad & \forall \xi \in \{a\tilde{r}r, ss, \tilde{fl}, srp\} \\ & \forall \xi \in \{arr, sf, fl, frp\} \\ (B3)-(B11) \quad & \forall \xi \in \{a\tilde{r}r, ss, \tilde{fl}, srp\} \\ (B12) \quad & \forall \xi \in \{arr, sf, fl, frp\} \end{aligned}$$

Constraints (40) to (53) are group-D constraints for canceling event sf. Binary variable $z_k^{\bar{sf}}$ is equal to one if executions of sf are removed from the future event list, otherwise the execution is not canceled. Specifically, constraints (40) and (42) show that if sf is canceled, state variable h must be greater than or equal to one, and there is at least one execution of sf in the future event list. Constraints (41) indicate that if $v^{\bar{sf},h,0}$ is equal to one, the machine is in up state. Constraints (43) indicate that if $v^{\bar{sf},g,0}$ is equal to one, there is no executions of sf in the future event list. Constraints (44) show that if executions of sf are not canceled, i.e., $z_k^{\bar{sf}}$ is equal to zero, either the

State variables

s	Description	Initial value
g	Equal to 1 if server is working on a job, otherwise equal to 0.	0
h	Equal to 1 if server fails, otherwise equal to 0.	0
q	Number of jobs in the queue.	0
u^{frp}	Number of executions with event type frp in the future event list.	0
u^{arr}	Number of executions with event type arr in the future event list.	0
u^{fl}	Number of executions with event type fl in the future event list.	0

Zero-delay events

ξ	Description	Condition to schedule	State transition
$a\tilde{r}r$	Counting arrival	$u^{arr} \leq 0$	${u^{arr}}_{++}$
ss	Start	$1 \leq q,\ g \leq 0,\ h \leq 0$	g++, q--
$\tilde{fl}$	Counting failure	$h \leq 0,\ u^{fl} \leq 0$	${u^{fl}}_{++}$
srp	Start to repair	$h \geq 1,\ u^{frp} \leq 0$	${u^{frp}}_{++}$

Positive-delay events

ξ	Description	Delay	Counting event	Counting variable	State transition	Condition to cancel
arr	Arrival	T^{arr}	$a\tilde{r}r$	u^{arr}	${u^{arr}}$--, q++	
sf	Finish	T^{sf}	ss	g	g--	$h \geq 1$, $g \geq 1$
fl	Failure	T^{fl}	$\tilde{fl}$	u^{fl}	h++, ${u^{fl}}$--	
frp	Finish of repair	T^{rp}	srp	u^{frp}	h--, ${u^{frp}}$--	

Table 4. Events to simulate G/G/1 with failure.

machine in in up state or there is no execution in the event list.

$$u_k^h \geq z_k^{\bar{sf}} \qquad \forall\ k \in \mathbb{K} \tag{40}$$

$$-u_k^h \geq v_k^{\bar{sf},h,0} - 1 \qquad \forall\ k \in \mathbb{K} \tag{41}$$

$$u_{k-1}^g \geq z_k^{\bar{sf}} \qquad \forall\ k \in \mathbb{K} \tag{42}$$

$$-u_{k-1}^g \geq v_k^{\bar{sf},g,0} - 1 \qquad \forall\ k \in \mathbb{K} \tag{43}$$

$$1 - z_k^{\bar{sf}} \leq v_k^{\bar{sf},s,0} + v_k^{\bar{sf},g,0} \qquad \forall\ k \in \mathbb{K} \tag{44}$$

Integer variables $k_i^{sf,0}$ and $k_i^{sf,1}$ denote the iteration index when the i-th execution of sf is scheduled and executed, as in constraints (45) and (46).

$$k_i^{f,1} = \sum_{k \in \mathbb{K}} k w_{k,i}^{sf} \qquad \forall\ i \in \mathbb{I}^{\xi} \tag{45}$$

$$k_i^{f,0} = \sum_{k \in \mathbb{K}} k x_{k,i}^{sf} \qquad \forall\ i \in \mathbb{I}^{\xi} \tag{46}$$

Binary variable $\theta_{i,k}^{sf}$ equal to one represents that i-th execution of event sf is canceled in iteration k. Constraints (47) and (48) state that if the i-th execution of sf is canceled in iteration k, then it must be scheduled before iteration k and executed after iteration k. Binary variable $\phi_{i,k}^{sf,0}$ equal to one indicates that the condition to cancel sf is not true in iteration k or the i-th execution of sf is scheduled after iteration k, as in constraints (49). Binary variable $\phi_{i,k}^{sf,1}$ equal to one indicates that the i-th execution of sf is executed before iteration k, as in constraints (50). Constraints (51) state that if execution i of sf is not canceled in iteration k, it is either because the condition to cancel is false, or it is executed before or scheduled after iteration k.

$$kz_k^{\bar{sf}} - k_i^{sf,0} - 1 \geq K(\theta_{i,k}^{sf} - 1) \qquad \forall\ i \in \mathbb{I}^{\xi}, k \in \mathbb{K} \tag{47}$$

$$k_i^{sf,1} - 1 - kz_k^{\bar{sf}} \geq k(\theta_{i,k}^{sf} - 1) \qquad \forall\ i \in \mathbb{I}^{\xi}, k \in \mathbb{K} \tag{48}$$

$$k_i^{sf,0} - kz_k^{\bar{sf}} \geq k(\phi_{i,k}^{sf,0} - 1) \qquad \forall\ i \in \mathbb{I}^{\xi}, k \in \mathbb{K} \tag{49}$$

$$kz_k^{\bar{sf}} - k_i^{sf,1} \geq K(\phi_{i,k}^{sf,1} - 1) \qquad \forall\ i \in \mathbb{I}^{\xi}, k \in \mathbb{K} \tag{50}$$

$$1 - \theta_{i,k}^{sf} \leq \phi_{i,k}^{sf,0} + \phi_{i,k}^{sf,1} \qquad \forall\ i \in \mathbb{I}^{\xi}, k \in \mathbb{K} \tag{51}$$

Binary variable $\gamma_{i,k}^{sf}$ equal to one represents that the i-th execution of event sf is performed in iteration k, and it is not canceled, as in constraints (53). Constraints (52) state that if $\gamma_{i,k}^{sf}$ is equal to zero, it is either because the execution is not performed in iteration k or it is canceled.

$$\gamma_{i,k}^{sf} \geq w_{i,k}^{sf} - \sum_{k' \in \mathbb{K}} \theta_{i,k'}^{sf} \qquad \forall\ i \in \mathbb{I}^{\xi}, k \in \mathbb{K} \tag{52}$$

$$w_{i,k}^{sf} - \sum_{k' \in \mathbb{K}} \theta_{i,k'}^{sf} - 1 \geq (N^{fl} + 1)(\gamma_{i,k}^{sf} - 1) \qquad \forall\ i \in \mathbb{I}^{\xi}, k \in \mathbb{K} \tag{53}$$

Constraints (54) to (60) depict the evolution of state variables. Constraints (54) to (57), referring to constraints (E5) and (E6), are similar to the two examples previously discussed, state variables u^{arr}, q, u^{fl} and u^{frp} are all changed by events without cancellation. Constraints (58) state that if event sf is canceled in iteration k, the number of executions of sf in the future event list will be reduced to zero. Constraints (59) and (60) show that if sf is not canceled in iteration k, the state variable g will be increased by one if ss is executed, and decreased by one if sf is executed without being canceled. The rest of the model indicates the initialization and range of state

variables.

$$u_k^{arr} = u_{k-1}^{arr} + \sum_{i \in \mathbb{I}^j} w_{i,k}^{\tilde{arr}} - \sum_{i \in \mathbb{I}^j} w_{i,k}^{arr} \qquad \forall\ k \in \mathbb{K} \tag{54}$$

$$u_k^{q} = u_{k-1}^{q} + \sum_{i \in \mathbb{I}^j} w_{i,k}^{arr} - \sum_{i \in \mathbb{I}^j} w_{i,k}^{ss} \qquad \forall\ k \in \mathbb{K} \tag{55}$$

$$u_k^{fl} = u_{k-1}^{fl} + \sum_{i \in \mathbb{I}^j} w_{i,k}^{\tilde{fl}} - \sum_{i \in \mathbb{I}^j} w_{i,k}^{fl} \qquad \forall\ k \in \mathbb{K} \tag{56}$$

$$u_k^{frp} = u_{k-1}^{frp} + \sum_{i \in \mathbb{I}^j} w_{i,k}^{srp} - \sum_{i \in \mathbb{I}^j} w_{i,k}^{frp} \qquad \forall\ k \in \mathbb{K} \tag{57}$$

$$u_k^{g} \leq 1 - z_k^{\bar{sf}} \qquad \forall\ k \in \mathbb{K} \tag{58}$$

$$u_k^{g} \leq u_{k-1}^{g} - \sum_{i \in \mathbb{I}} \gamma_{i,k}^{sf} + \sum_{i \in \mathbb{I}} w_{i,k}^{ss} + z_k^{\bar{sf}} \qquad \forall\ k \in \mathbb{K} \tag{59}$$

$$u_k^{g} \geq u_{k-1}^{g} - \sum_{i \in \mathbb{I}} \gamma_{i,k}^{sf} + \sum_{i \in \mathbb{I}} w_{i,k}^{ss} - z_k^{\bar{sf}} \qquad \forall\ k \in \mathbb{K} \tag{60}$$

$$u_0^{arr} = u_0^{fl} = u_0^{r} = u_0^{g} = u_0^{q} = 0$$

$$u_k^{arr}, u_k^{fl}, u_k^{frp}, u_k^{g} \in \{0,1\},\ u_k^{q} \in \{0, ..., Q\}$$

4. Discussion

Given the set of state variables, the set of zero-delay events, the sets of positive-delay events, as in Tables 1, 3, 4, and also K and N^{ξ}, the total number of iterations and the maximum number of executions of each event respectively, the implementation of the proposed approach is trivial. Appendix I provides the pseudo code of the proposed approach. As it can be seen, the algorithm is composed of several for-loops only. The java source code of an automatic MPR generation procedure can be found online at https://github.com/myrazhang/MRPofDES.

The MPR proposed in this work differs from the state-of-the-art MPR (Chan & Schruben, 2008) in the objective function. In the state-of-the-art formulation, the constraints represent the event triggering relationships, i.e., the execution time of the triggered event must be later than that of the triggering event with a delay, which is a sample from a predefined random variate. Hence, the objective function must be the minimization of the execution time of all the events, so that all the events are executed as soon as possible. In this work, the constraints represent that once the condition to schedule or cancel an event is true, the event must be scheduled or canceled, i.e., the logic that events must be executed as soon as possible is forced by constraints. Thus, the objective function can be defined in a more flexible way.

The proposed MPR presents only the realization of a sample path of DES, since it takes the samples of random variates as the coefficients of constraints. However, expanding the MPR from single-sample-path model into a multiple-sample-path model can be achieved by adding one more subscripts to each decision variables, and the resulting MPR is separable since there is not constraints linking variables from different sample paths, i.e., the sample paths are independent from each other.

As it can be seen, the proposed MPR contains both integer variables and real-valued variables. Furthermore, if the steady state performance from simulation model is of interest, the simulation length, i.e., the number of event executions, is usually

long. Thus, the MPR has many variables and constraints. Generally speaking, it is reasonable to use the MPR as representation instead of solving it as an MPR because the computational complexity is extremely high.

An application of the MPR is the calculation of sub-gradient from a simulation run, based on the sensitivity analysis of the approximate linear programming (LP) of the MPR. The procedure is as follows. The DES is first simulated with an event-scheduling execution logic, and the values of all decision variables of the MPR can be obtained during the simulation run. The approximate LP is formulated by replacing all the decision variables with a big-M multiplier, i.e., all variables except the ones representing event scheduling times, execution times, and state variables, with the value of all the decision variables with a big-M multiplier obtained from the simulation, and relaxing the integrality of the remaining variables. By solving the dual problem of the approximate LP, the gradient of the objective function can be obtained on some coefficients, for instance, the thresholds $a^{\xi,s}$ and $c^{\xi,s}$ of the ranges that compose the conditions to schedule or cancel events, which are part of the conditions to schedule events.

Another way to make use of the MPR is to derive an MPR of a simulation optimization problem. In the applications in manufacturing and service operations, the decision variables of an optimization problem are usually the thresholds $a^{\xi,s}$ and $c^{\xi,s}$ of the ranges that compose the conditions to schedule or cancel events, which are part of the conditions to schedule events. So, to transform the MPR of simulation into MPR of simulation optimization, one just needs to specify that certain thresholds are decision variables instead of coefficients. Then, the system performance could be in the objective function or in a constraint stating that it has to achieve a target. The convenience of transforming the proposed MPR into an MPR of simulation optimization represents another contribution of this work. However, the resulting MPR still has computational issue, which leads to our future research interest.

5. Conclusion

This work proposes an MPR of a sample path of a DES model that satisfies certain assumptions, based on the widely-applied event-scheduling execution logic of DES. In the MPR, the decision variables include event scheduling times, execution times, and system state after each event execution, and the constraints represent that events are scheduled or canceled when the system state satisfies the condition to schedule or cancel. Furthermore, the MPR also takes the samples of random variates as the coefficients of constraints, where the random variates are usually the time delay between the occurring time and scheduling time of an event execution. The equivalence of the MPR and the simulation run implemented with an event-scheduling algorithm has been validated using several cases. Future work will be dedicated to developing algorithm to solve optimization problems based on the MPR.

Appendix I: implementation pseudo code

Algorithm 3 Implementation pseudo code.

Input:
Set of events $\mathbb{X}$.
Initial system state: $\mathbf{s}_0$.
Simulation length: number of iterations K.
Execution number of event ξ: N^{ξ}

Initialization: (A6) (E7)
for $\xi \in \mathbb{X}$ **do**
  for $i = 1$ *to* N^{ξ} **do**
    for $k = 1$ *to* K **do**
      (A1) (A2) (B1) (B2)
    end for
    (A3) (A7) (C2) (C3)
  end for
  for $i = 1$ *to* $N^{\xi} - 1$ **do**
    (C1) (C4)
  end for
end for
for $k = 1$ *to* K **do**
  (A4) (A5)
end for
for each zero-delay event $\xi \in \mathbb{X}$ **do**
  for $k = 0$ *to* $K - 1$ **do**
    for $s \in \mathbb{S}^{\xi}$ **do**
      (B3) (B4) (B5) (B6)
    end for
    (B7) (B9) (B10)
  end for
  (B8)
  for $i = 1$ *to* N^{ξ} **do**
    (B11)
  end for
end for
for each positive-delay event $\xi \in \mathbb{X}$ **do**
  for $k = 0$ *to* $K - 1$ **do**
    for $i = 1$ *to* N^{ξ} **do**
      (B12) (D6) (D7) (D8) (D9) (D10) (D11) (D12) (D13) (D14)
    end for
    for $s \in \mathbb{S}^{\bar{\xi}}$ **do**
      (D1) (D2) (D3) (D4)
    end for
    (D5)
  end for
end for
for $k = 0$ *to* $K - 1$ **do**
  for $s \in \mathbb{S}/\mathbb{S}^{c}$ **do**
    (E1)

42: **end for**
43: **for** $s \in \mathbb{S}^c$ **do**
44: (E2) (E3) (E4)
45: **end for**
46: **end for**

Appendix I: G/G/m of Chan & Schruben's model

The MPR model proposed in Chan and Schruben (2008) is as follows:

$$\min\{\sum_{i=1}^{N}(e^a_{i,1} + e^r_{i,1} + e^f_{i,1})\}$$

$$e^a_{i,1} - e^a_{i-1,1} = t^a_i \qquad \forall\ i \tag{61}$$

$$e^f_{j,1} - e^r_{i,1} \geq t^f_i - M(1 - \sum_{l=max\{i-m+1,1\}}^{j} y^f_{i,l}) \qquad \forall\ i,j \tag{62}$$

$$e^r_{i,1} - e^a_{i,1} \geq 0 \qquad \forall\ i \tag{63}$$

$$e^r_{i,1} - e^f_{i-m,1} \geq 0 \qquad \forall\ i \tag{64}$$

$$\sum_{l=max\{i-m+1,1\}}^{n} y^f_{i,l} = 1 \qquad \forall i \tag{65}$$

$$\sum_{i=1}^{min\{j+m-1,n\}} y^f_{i,l} = 1 \qquad \forall j \tag{66}$$

To explain the MPR model, the ERG of G/G/1 queue is first shown in Figure 2. Each triggering relationship in the ERG, i.e., two connected nodes is represented by one constraint. Constraints (61) to (64) state the relationships of $e^a \rightarrow e^a$, $e^s \rightarrow e^f$, $e^a \rightarrow e^s$ and $e^f \rightarrow e^s$, respectively. The objective function is the sum of all the event execution time. As can be seen, the model is a linear programming.

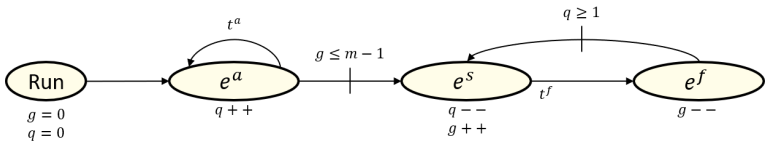


Figure 2. ERG of G/G/1 queue.

Appendix II: single-server merge of Chan & Schruben's model

The MPR model proposed in Chan and Schruben (2008) is as follows:

$$\min\{\sum_{i=1}^{N_1}(e_i^{s,1}+e_i^{f,1}+e_i^{d,1})+\sum_{i=1}^{N_2}(e_i^{s,2}+e_i^{f,2}+e_i^{d,2})+\sum_i e_i^{d,2,1}+$$
$$\sum_{i=1}^{N_1+N_2}(e_i^{s,3}+e_i^{d,3}-e_i^{w:(d,1),(d,2)})+\sum_{i=1}^{N_1+N_2}\sum_{k=1}^{N_1}e_{i,k,i-k}^{p:(d,1),(d,2)}\}$$

$$e_i^{s,1}-e_{i-1}^{d,1}=0 \tag{67}$$
$$e_i^{f,1}-e_i^{s,1}=t_i^1 \tag{68}$$
$$e_i^{s,2}-e_{i-1}^{d,2}=0 \tag{69}$$
$$e_i^{f,2}-e_i^{s,2}=t_i^2 \tag{70}$$
$$e_i^{d,3}-e_i^{s,3}=t_i^3 \tag{71}$$
$$e_i^{d,1}-e_i^{f,1}\geq 0 \tag{72}$$
$$e_i^{s,3}-e_{i-1}^{d,3}\geq 0 \tag{73}$$

$$e_i^{d,2}-e_j^{d,2,1}\geq M(\sigma_{(d,2,1):j,(d,2):i}-1) \tag{74}$$
$$e_i^{d,2}-e_j^{d,2,1}\geq m(1-\sigma_{(d,2,1):j,(d,2):i}) \tag{75}$$
$$e_j^{d,2,1}\geq e_k^{d,1}-M(1-\zeta_{(d,1):k,(d,2,1):j}) \tag{76}$$
$$e_k^{d,1}\geq e_j^{d,2,1}+m\zeta_{(d,1):k,(d,2,1):j}+(1-\zeta_{(d,1):k,(d,2,1):j})\varepsilon \tag{77}$$
$$e_{k+1}^{f,1}\geq e_j^{d,2,1}-M(1-\eta_{(d,2,1):j,(f,1):k+1}) \tag{78}$$
$$e_j^{d,2,1}\geq e_{k+1}^{f,1}+m\eta_{(d,2,1):j,(f,1):k+1} \tag{79}$$
$$\gamma_{(d,1):k,(d,2,1):j}-(\zeta_{(d,1):k,(d,2,1):j}+\eta_{(d,2,1):j,(f,1):k+1})+1\geq 0 \tag{80}$$
$$2\gamma_{(d,1):k,(d,2,1):j}-(\zeta_{(d,1):k,(d,2,1):j}+\eta_{(d,2,1):j,(f,1):k+1})\leq 0 \tag{81}$$
$$\sum_k \gamma_{(d,1):k,(d,2,1):j}-\sum_i \sigma_{(d,2,1):j,(d,2):i}\geq 0 \tag{82}$$
$$n\sum_i \sigma_{(d,2,1):j,(d,2):i}-\sum_k \gamma_{(d,1):k,(d,2,1):j}\geq 0 \tag{83}$$
$$\sum_i \sigma_{(d,2,1):j,(d,2):i}\leq 1 \tag{84}$$
$$\sum_j \sigma_{(d,2,1):j,(d,2):i}\leq 1 \tag{85}$$
$$\sum_{j=i}^{n}\sigma_{(d,2,1):j,(d,2):i}\geq \sum_{j=i+1}^{n}\sigma_{(d,2,1):j,(d,2):i+1} \tag{86}$$
$$\sum_{p=j+1}^{n}\sum_{q=1}^{i-1}\sigma_{(d,2,1):p,(d,2):q}\geq min\{i-1,n-j\}(1-\sigma_{(d,2,1):j,(d,2):i}) \tag{87}$$

$$e_i^{d,2,1} - e_j^{s,3} \geq M(\sigma_{(s,3):j,(d,2,1):i} - 1) \tag{88}$$

$$e_i^{d,2,1} - e_j^{s,3} \geq m(1 - \sigma_{(s,3):j,(d,2,1):i}) \tag{89}$$

$$e_j^{s,3} \geq e_k^{f,2} - M(1 - \zeta_{(f,2):k,(s,3):j}) \tag{90}$$

$$e_k^{f,2} \geq e_j^{s,3} + m\zeta_{(f,2):k,(s,3):j} + (1 - \zeta_{(f,2):k,(s,3):j})\varepsilon \tag{91}$$

$$e_{k+1}^{d,2} \geq e_j^{s,3} - M(1 - \eta_{(s,3):j,(d,2):k+1}) \tag{92}$$

$$e_j^{s,3} \geq e_{k+1}^{d,2} + m\eta_{(s,3):j,(d,2):k+1} \tag{93}$$

$$\gamma_{(f,2):k,(s,3):j} - (\zeta_{(f,2):k,(s,3):j} + \eta_{(s,3):j,(d,2):k+1}) + 1 \geq 0 \tag{94}$$

$$2\gamma_{(f,2):k,(s,3):j} - (\zeta_{(f,2):k,(s,3):j} + \eta_{(s,3):j,(d,2):k+1}) \leq 0 \tag{95}$$

$$\sum_k \gamma_{(f,2):k,(s,3):j} - \sum_i \sigma_{(s,3):j,(d,2,1):i} \geq 0 \tag{96}$$

$$n\sum_i \sigma_{(s,3):j,(d,2,1):i} - \sum_k \gamma_{(f,2):k,(s,3):j} \geq 0 \tag{97}$$

$$\sum_i \sigma_{(s,3):j,(d,2,1):i} \leq 1 \tag{98}$$

$$\sum_j \sigma_{(s,3):j,(d,2,1):i} \leq 1 \tag{99}$$

$$\sum_{j=i}^{n} \sigma_{(s,3):j,(d,2,1):i} \geq \sum_{j=i+1}^{n} \sigma_{(s,3):j,(d,2,1):i+1} \tag{100}$$

$$\sum_{p=j+1}^{n} \sum_{q=1}^{i-1} \sigma_{(s,3):p,(d,2,1):q} \geq min\{i-1, n-j\}(1 - \sigma_{(s,3):j,(d,2,1):i}) \tag{101}$$

$$e_i^{w:(d,1),(d,2)} \leq e_{i,k,i-k}^{p:(d,1),(d,2)} \tag{102}$$

$$e_k^{d,1} \geq e_{i-k}^{d,2} - M(1 - \alpha_{i,k,i-k}^{(d,1),(d,2)}) \tag{103}$$

$$e_{i-k}^{d,2} \geq e_{i-k}^{d,1} + m\alpha_{i,k,i-k}^{(d,1),(d,2)} + (1 - \alpha_{i,k,i-k}^{(d,1),(d,2)})\varepsilon \tag{104}$$

$$e_{i,k,i-k}^{p:(d,1),(d,2)} \geq e_k^{d,1} - M(1 - \alpha_{i,k,i-k}^{(d,1),(d,2)}) \tag{105}$$

$$e_k^{d,1} \geq e_{i,k,i-k}^{p:(d,1),(d,2)} + m(1 - \alpha_{i,k,i-k}^{(d,1),(d,2)}) \tag{106}$$

$$e_{i,k,i-k}^{p:(d,1),(d,2)} \geq e_k^{d,2} - M\alpha_{i,k,i-k}^{(d,1),(d,2)} \tag{107}$$

$$e_k^{d,2} \geq e_{i,k,i-k}^{p:(d,1),(d,2)} + m\alpha_{i,k,i-k}^{(d,1),(d,2)} \tag{108}$$

$$e_i^{d,2,1} - e_j^{f,2} \geq M(\sigma_{(f,2):j,(d,2,1):i} - 1) \tag{109}$$

$$e_i^{d,2,1} - e_j^{f,2} \geq m(1 - \sigma_{(f,2):j,(d,2,1):i}) \tag{110}$$

$$e_j^{f,2} \geq e_k^{s,3} - M(1 - \zeta_{(s,3):k,(f,2):j}) \tag{111}$$

$$e_k^{s,3} \geq e_j^{f,2} + m\zeta_{(s,3):k,(f,2):j} + (1 - \zeta_{(s,3):k,(f,2):j})\varepsilon \tag{112}$$

$$e_{k+Q}^{w:(d,1),(d,2)} \geq e_j^{f,2} - M(1 - \eta_{(f,2):j,(w:(d,1),(d,2)):k+Q}) \tag{113}$$

$$e_j^{f,2} \geq e_{k+Q}^{w:(d,1),(d,2)} + m\eta_{(f,2):j,(w:(d,1),(d,2)):k+Q} \tag{114}$$

$$\gamma_{(s,3):k,(f,2):j} - (\zeta_{(s,3):k,(f,2):j} + \eta_{(f,2):j,(w:(d,1),(d,2)):k+Q}) + 1 \geq 0 \tag{115}$$

$$2\gamma_{(s,3):k,(f,2):j} - (\zeta_{(s,3):k,(f,2):j} + \eta_{(f,2):j,(w:(d,1),(d,2)):k+Q}) \leq 0 \tag{116}$$

$$\sum_k \gamma_{(s,3):k,(f,2):j} - \sum_i \sigma_{(f,2):j,(d,2,1):i} \geq 0 \tag{117}$$

$$n\sum_i \sigma_{(f,2):j,(d,2,1):i} - \sum_k \gamma_{(s,3):k,(f,2):j} \geq 0 \tag{118}$$

$$\sum_i \sigma_{(f,2):j,(d,2,1):i} \leq 1 \tag{119}$$

$$\sum_j \sigma_{(f,2):j,(d,2,1):i} \leq 1 \tag{120}$$

$$\sum_{j=i}^{n} \sigma_{(f,2):j,(d,2,1):i} \geq \sum_{j=i+1}^{n} \sigma_{(f,2):j,(d,2,1):i+1} \tag{121}$$

$$\sum_{p=j+1}^{n} \sum_{q=1}^{i-1} \sigma_{(f,2):p,(d,2,1):q} \geq min\{i-1, n-j\}(1 - \sigma_{(f,2):j,(d,2,1):i}) \tag{122}$$

$$e_i^{s,3} \geq e_i^{w:(d,1),(d,2)} \tag{123}$$

$$e_i^{d,1} - e_j^{f,1} \geq M(\sigma_{(f,1):j,(d,1):i} - 1) \tag{124}$$

$$e_i^{d,1} - e_j^{f,1} \geq m(1 - \sigma_{(f,1):j,(d,1):i}) \tag{125}$$

$$e_j^{f,1} \geq e_k^{s,3} - M(1 - \zeta_{(s,3):k,(f,1):j}) \tag{126}$$

$$e_k^{s,3} \geq e_j^{f,1} + m\zeta_{(s,3):k,(f,1):j} + (1 - \zeta_{(s,3):k,(f,1):j})\varepsilon \tag{127}$$

$$e_{k+Q}^{d,2} \geq e_j^{f,1} - M(1 - \eta_{(f,1):j,(d,2):k+Q}) \tag{128}$$

$$e_j^{f,1} \geq e_{k+Q}^{d,2} + m\eta_{(f,1):j,(d,2):k+Q} \tag{129}$$

$$\gamma_{(s,3):k,(f,1):j} - (\zeta_{(s,3):k,(f,1):j} + \eta_{(f,1):j,(d,2):k+Q}) + 1 \geq 0 \tag{130}$$

$$2\gamma_{(s,3):k,(f,1):j} - (\zeta_{(s,3):k,(f,1):j} + \eta_{(f,1):j,(d,2):k+Q}) \leq 0 \tag{131}$$

$$\sum_k \gamma_{(s,3):k,(f,1):j} - \sum_i \sigma_{(f,1):j,(d,1):i} \geq 0 \tag{132}$$

$$n\sum_i \sigma_{(f,1):j,(d,1):i} - \sum_k \gamma_{(s,3):k,(f,1):j} \geq 0 \tag{133}$$

$$\sum_i \sigma_{(f,1):j,(d,1):i} \leq 1 \tag{134}$$

$$\sum_j \sigma_{(f,1):j,(d,1):i} \leq 1 \tag{135}$$

$$\sum_{j=i}^{n} \sigma_{(f,1):j,(d,1):i} \geq \sum_{j=i+1}^{n} \sigma_{(f,1):j,(d,1):i+1} \tag{136}$$

$$\sum_{p=j+1}^{n} \sum_{q=1}^{i-1} \sigma_{(f,1):p,(d,1):q} \geq min\{i-1, n-j\}(1 - \sigma_{(f,1):j,(d,1):i}) \tag{137}$$

To explain the MPR model, the ERG of the merge system is first shown in Figure 3. Each triggering relationship in the ERG is represented by one or more constraints. Since the modeling framework requires that there is at most one condition on each arc, event $e^{d,2,1}$ is introduced to split the composed conditions from $e^{f,2}$ and $e^{s,3}$ to $e^{d,3}$. The state variables m^1 and m^2 are also varied, and they are equal to one if the server is blocked. The new definition is equivalent to the one we used to derived the model proposed in this work, but simplify the model under the framework of Chan and Schruben (2008). Constraints (67) to (72) state the triggering relationship of $e^{d,1} \rightarrow e^{s,1}$, $e^{s,1} \rightarrow e^{f,1}$, $e^{d,2} \rightarrow e^{s,2}$, $e^{s,1} \rightarrow e^{f,1}$, $e^{s,3} \rightarrow e^{d,3}$ and $e^{s,3} \rightarrow e^{d,1}$, respectively. Constraints (73) show the triggering relationship between $e^{d,1} \rightarrow e^{s,3}$ and also $e^{d,2} \rightarrow e^{s,3}$. Constraints (74) to (87) imply the triggering relationship from $e^{d,2,1}$ to $e^{d,2}$. Constraints (88) to (101) imply the triggering relationship from $e^{d,3}$ to $e^{d,2,1}$. Constrains (102) to (108) imply the convolution of $e^{d,1}$ and $e^{d,2}$. The real-valued variables $e_i^{w:(d,1).(d,2)}$ represent the i-th execution of $e^{d,1}$ or $e^{d,2}$, and $e_{i,k,i-k}^{p:(d,1),(d,2)}$ are equal to the maximum between $e_k^{d,1}$ and $e_{i-k}^{d,2}$. Constrains (109) to (122) imply the triggering relationship from $e^{f,2}$ to $e^{d,2,1}$. Constraints (123) show the triggering relationship from $e^{d,3}$ to $e^{s,3}$. Constraints (124) to (137) show the triggering relationship from $e^{f,1}$ to $e^{d,1}$. The variables $\zeta,\ \sigma,\ \eta,\ \gamma,\ \alpha$ are all binary, and the notations are the same as in Chan and Schruben (2008). The objective function is also defined as proposed in Chan and Schruben (2008), i.e., minimizing the sum of all the event execution times, minimizing

the real-valued variables bounded from below (i.e., $e^{p:(d,1),(d,2)}$), and maximizing the real-valued variables bounded from above (i.e., $e^{w:(d,1),(d,2)}$).

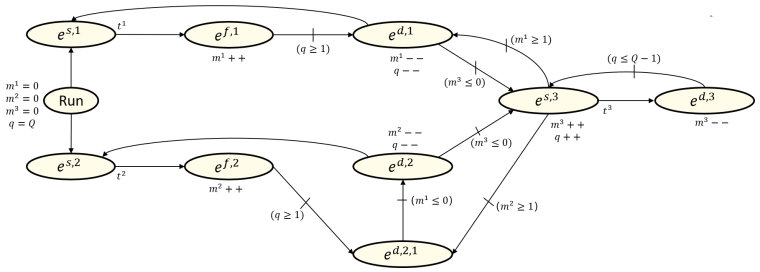


Figure 3. ERG of merge queueing system.

It can be seen that the model proposed in Chan and Schruben (2008) requires to derive different constraints from each arcs according to the condition on the arc and to the state changes of several events. Furthermore, the constraints bound the event execution time from below, which indicates that the event *can* be executed, and the objective function drives the events or each to be executed as soon as possible. However, the DES model *must* execute the events once the conditions are true, which cannot be guaranteed with the model. To guarantee the equivalence between the MPR and the simulation implementation, the multiplier of each term of the objective function has to be carefully chosen. Using the model proposed in this work, the objective function can be arbitrary, i.e., any performance indicator can be the objective function.

(s,S) policy

Zero-delay events

Variable	Event	Condition to schedule	State change
$e^{\tilde{ca}}$	Count customer arrival	$u^{ca} \leq 0$	u^{ca}++
e^{rr}	Require replenishment	$u^{ra} \leq 0$ & $q \leq s$	u^{ra}++
e^{cl}	Customer loss	$a \geq 1$ & $q \leq 0$	a--
e^{co}	Customer with order	$a \geq 1$ & $q \geq 1$	a--, q--

Positive-delay events

Variable	Event	Delay	$e^{\tilde{\xi}}$	u^{ξ}	β^{ξ}	State change
e^{ca}	Customer order arrival	T^{ca}	$e^{\tilde{ca}}$	u^{ca}	1	u^{ca}--, a++
e^{ra}	Replenishment arrival	T^{ra}	e^{rr}	u^{ra}	1	u^{ra}--, $q = q + S - s$

Table 5. Events to simulate (s,S) policy.

6. Draft

6.1. Condition based maintenance

State variables b: number of backorders q: finished goods in queue m: machine working/idle $fl \in \{0, 1, 2\}$: failure level $r1$: first level repair $r2$: second level repair

Events

Zero-delay events

Variable	Event	Condition to schedule	State change	N^ξ
$e^{a\tilde{r}r}$	Count arrival	$u^{arr} \leq 0$	u^{arr}++	$N^1 + N^2$
e^{ol}	Order leaves	$b \geq 1,\ q \geq 1$	b--, q--	N^1
e^{oa}	Abandoned	$b \geq B,\ so \geq 1$	so--	$N^2 - B$
e^{oe}	Order enters	$b \leq B - 1,\ so \geq 1$	so--, b++	$N^1 + B$
e^{s1}	Start	$q \leq Q - 1,\ m \leq 0,\ fl \leq 1$ $r1 \leq 0,\ (fl \leq 0\ or\ q \leq Q^r - 1)$	m++	??
e^{sr1}	Start repair 1	$q \geq Q^r,\ m \leq 0$ $1 \leq fl \leq 1,\ r1 \leq 0$	$r1$++	N^4
e^{sr2}	Start repair 2	$fl \geq 2,\ r2 \leq 0$	$r2$++	$N^3 - N^4$
$e^{f\tilde{l}1}$	Count failure 1	$fl \leq 0,\ u^{fl1} \leq 0$	u^{fl1}++	N^3
$e^{f\tilde{l}2}$	Count failure 2	$1 \leq fl \leq 1,\ u^{fl2} \leq 0$ $r1 \leq 0$	u^{fl2}++	N^3

Positive-delay events

Variable	Event	Delay	$e^{\tilde{\xi}}$	u^ξ	β^ξ	State change	N^ξ	Condition to cancel
e^{arr}	Arrival	T^{arr}	$e^{a\tilde{r}r}$	u^{arr}	1	u^{arr}--, so++	$N^1 + N^2$	
e^f	Finish	T^f	e^s	m	1	m--, q++, c^q++	??	$u^{fl2} \geq 1, m$
e^{fr1}	Finish repair 1	T^{fr1}	e^{sr1}	$r1$	1	fl--, $r1$--	N^4	
e^{fr2}	Finish repair 2	T^{fr2}	e^{sr2}	$r2$	1	$fl = fl - 2$, $r2$--	$N^3 - N^4$	
e^{fl1}	Failure 1	T^{fl1}	$e^{f\tilde{l}1}$	u^{fl1}	1	fl++, u^{fl1}--	N^3	
e^{fl2}	Failure 2	T^{fl2}	$e^{f\tilde{l}2}$	u^{fl2}	1	fl++, u^{fl2}--	N^3	$r1 \geq 1$, u^{fl2}

Table 6. Events to simulate G/G/1 with failure.

MP model of single server merge model:

$$\min \sum_k \mathcal{E}_k \tag{138}$$

$$e_i^{(\xi,j),1} - \mathcal{E}_k \geq M(w_{i,k}^{\xi,j} - 1) \qquad \xi \in \{s,f,d\}, j \in \{1,2,3\}, \forall i,k \tag{139}$$

$$\mathcal{E}_k - e_i^{(\xi,j),1} \geq M(w_{i,k}^{\xi,j} - 1) \qquad \xi \in \{s,f,d\}, j \in \{1,2,3\}, \forall i,k \tag{140}$$

$$\sum_k w_{i,k}^{\xi,j} = 1 \qquad \forall\ \xi \in \{s,f,d\}, j \in \{1,2,3\}, i \tag{141}$$

$$\sum_{(\xi,j),i} w_{i,k}^{\xi,j} = 1 \qquad \forall\ k \tag{142}$$

$$\sum_k k w_{i+1,k}^{\xi,j} - \sum_k k w_{i,k}^{\xi,j} \geq 1 \qquad \forall\ \xi \in \{s,f,d\}, j \in \{1,2,3\}, i \tag{143}$$

$$e_i^{s,j,1} - e_i^{s,j,0} \geq 0 \qquad j = 1,2,3, \forall\ i \tag{144}$$

$$e_i^{f,j,1} - e_i^{f,j,0} \geq t_i^j \qquad j = 1,2, \forall\ i \tag{145}$$

$$e_i^{d,j,1} - e_i^{d,j,0} \geq 0 \qquad j = 1,2, \forall\ i \tag{146}$$

$$e_i^{d,3,1} - e_i^{d,3,0} \geq t_i^3 \qquad \forall\ i \tag{147}$$

$$e_i^{\xi,j,0} - \mathcal{E}_k \geq M(x_{i,k}^{\xi,j} - 1) \qquad \forall\ \xi \in \{s,f,d\}, j = 1,2,3,k,i \tag{148}$$

$$\mathcal{E}_k - e_i^{\xi,j,0} \geq M(x_{i,k}^{\xi,j} - 1) \qquad \forall\ \xi \in \{s,f,d\}, j = 1,2,3,k,i \tag{149}$$

$$m_k^j = m_{k-1}^j + \sum_{i=1}^{N^j} (w_{i,k}^{s,j} + w_{i,k}^{f,j} - 2w_{i,k}^{d,j}) \qquad j = 1,2, \forall\ k \tag{150}$$

$$m_k^3 = m_{k-1}^3 + \sum_{i=1}^{N^3} (w_{i,k}^{s,3} - w_{i,k}^{d,3}) \qquad \forall\ k \tag{151}$$

$$q_k = q_{k-1} + \sum_{i=1}^{N^3} w_{i,k}^{s,3} - \sum_{i=1}^{N^1} w_{i,k}^{d,1} - \sum_{i=1}^{N^2} w_{i,k}^{d,2} \tag{152}$$

$$m_k^j \geq M(z_k^{s,j} - 1) \qquad j = 1,2,3, \forall\ k \tag{153}$$

$$1 - m_k^j \geq M(z_k^{f,j} - 1) \qquad j = 1,2, \forall\ k \tag{154}$$

$$m_k^j - 1 \geq M(z_k^{f,j} - 1) \qquad j = 1,2, \forall\ k \tag{155}$$

$$m_k^j - 2 \geq M(z_k^{d,j} - 1) \qquad j = 1,2, \forall\ k \tag{156}$$

$$q_k - 1 \geq M(z_k^{d,j} - 1) \qquad j = 1,2, \forall\ k \tag{157}$$

$$1 - m_k^1 \geq M(z_k^{d,2} - 1) \qquad \forall\ k \tag{158}$$

$$m_k^3 - 1 \geq M(z_k^{d,3} - 1) \qquad \forall\ k \tag{159}$$

$$(Q-1) - q_k \geq M(z_k^{s,3} - 1) \qquad \forall\ k \tag{160}$$

$$\sum_k x_{i,k}^{\xi,j} = 1 \qquad \forall\ \xi \in \{s,f,d\}, j = 1,2,3, \forall\ i,k \tag{161}$$

$$\sum_{i=1}^{N^j} x_{i,k}^{\xi,j} \leq z_k^{\xi,j} \qquad \forall\ \xi \in \{s,f,d\}, j = 1,2,3,k \tag{162}$$

$$\sum_k k x_{i+1,k}^{\xi,j} - \sum_k k x_{i,k}^{\xi,j} \geq 0 \qquad \forall\ \xi \in \{s,f,d\}, j = 1,2,3,i \tag{163}$$

7. Resource allocation problem of queueing systems

7.1. Mathematical programming representation of simulation model

We study only the system that the occurrence of an event will lead to the increment or decrement of one unit of the state variables. A simulation model is called a *natural* simulator if the following assumptions all hold:

(1) An event e^{ξ} can be triggered if the state variables $\mathbf{s}$ satisfy specific conditions *at that time*, regardless of the history of the state or event occurrence, and the condition is not changed along time, i.e., condition is static. It could be possible to define more state variables in case of history dependence and variant triggering conditions.
(2) Natural triggering relationship: if and only if e^{ξ} is an s-increment event, e^{ξ} triggers an s-decrement event, vice versa.
(3) Natural triggering condition: the condition for triggering an e^{ξ} is that each state variable s must within its predefined domain, i.e., $\mathbf{l} \leq \mathbf{s} \leq \mathbf{u}$, regardless of event type ξ.
(4) For all e^{ξ}, the number of execution N^{ξ} is known before simulation, and the simulation terminate when all types of events have been triggered for that number.

Assumptions for a variable x to be resource-type:

(1) For all e^{ξ}, $\mathbf{u}$ is monotonically increasing on x, and $\mathbf{l}$ is monotonically decreasing on x.

Formulate the MP model of simulation:

$e_i^{\xi} \geq 0$	time of the i-th occurrence of event e^{ξ}.
$\tau_l^{s+} \geq 0$	time of the l-th occurrence of events that increments state variable s.
$\tau_l^{s-} \geq 0$	time of the l-th occurrence of events that decrements state variable s.
$x_{i,l}^{\xi,s+} \in \{0,1\}$	equal to 1 if e_i^{ξ} is the l-th increment of s.
$x_{i,l}^{\xi,s-} \in \{0,1\}$	equal to 1 if e_i^{ξ} is the l-th decrement of s.

The MP model of simulation is

$$\min\{\textstyle\sum_{\xi,i} e_i^{\xi}\} \tag{164}$$

$$s.t. \tag{165}$$

$$\tau_l^{s+} - \tau_{l+s_0-u_s}^{s-} \geq 0 \qquad \forall\ s \tag{166}$$

$$\tau_l^{s-} - \tau_{l-s_0+l_s}^{s+} \geq 0 \qquad \forall\ s \tag{167}$$

$$\tau_l^{s+} - e_i^{\xi} \geq M(x_{i,l}^{\xi,s+} - 1) \quad \forall\ s\ and\ e^{\xi}\ with\ increment\ of\ s \tag{168}$$

$$\tau_l^{s-} - e_i^{\xi} \geq M(x_{i,l}^{\xi,s-} - 1) \quad \forall\ s\ and\ e^{\xi}\ with\ decrement\ of\ s \tag{169}$$

$$e_i^{\xi} - \tau_l^{s+} \geq M(x_{i,l}^{\xi,s+} - 1) \quad \forall\ s\ and\ e^{\xi}\ with\ increment\ of\ s \tag{170}$$

$$e_i^{\xi} - \tau_l^{s-} \geq M(x_{i,l}^{\xi,s-} - 1) \quad \forall\ s\ and\ e^{\xi}\ with\ decrement\ of\ s \tag{171}$$

$$\textstyle\sum_{\xi,i} x_{i,l}^{\xi,s+} = 1 \qquad \forall\ s,l \tag{172}$$

$$\textstyle\sum_{\xi,i} x_{i,l}^{\xi,s-} = 1 \qquad \forall\ s,l \tag{173}$$

$$\textstyle\sum_{s,l} x_{i,l}^{\xi,s+} = 1 \qquad \forall\ \xi,i \tag{174}$$

$$\textstyle\sum_{s,l} x_{i,l}^{\xi,s-} = 1 \qquad \forall\ \xi,i \tag{175}$$

$$\tag{176}$$

7.2. Mathematical programming representation of simulation model - V2

Revise event-based simulation algorithm.

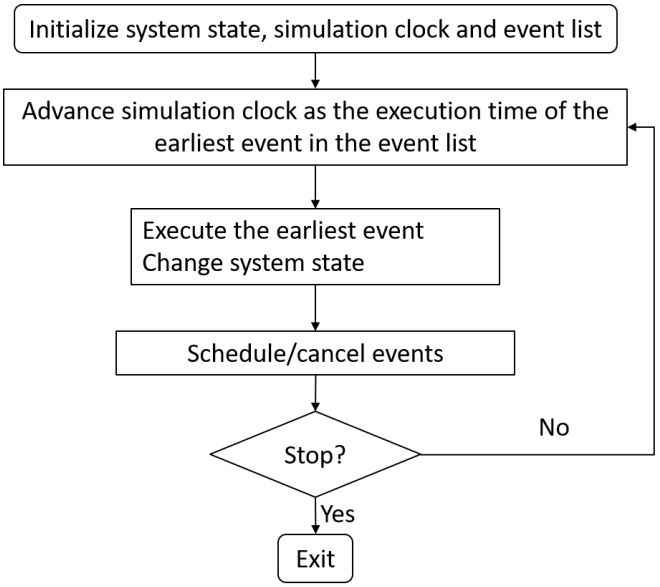


Figure 4. Event-based simulation algorithm.

An equivalent mathematical programming model exists if the following assumptions are satisfied:

(1) State variables are integer.
(2) For all event e^{ξ}, the *scheduling conditions* are in the form of $a^{\xi,s} \leq s \leq c^{\xi,s}$ combined with logic operator "AND", where s is a state variable, and $a^{\xi,s}$ and $c^{\xi,s}$ are lower and upper bounds.
(3) The scheduling conditions is independent of the history and not changed along time. (It could be possible to define more state variables in case of history dependence and time-variant scheduling conditions.)
(4) An event execution of e^{ξ} leads to integer increment or decrement equal to $\Delta^{\xi,s}$ of certain state variables s, and $\Delta^{\xi,s}$ is not changed along time.
(5) The delay between scheduling and execution time of an event e^{ξ}, denoted by t^{ξ}, is random variate. They can be generated independently from the simulation run. (*This point is different from ERG. In ERG, the delay is dependent on the edge, i.e, a couple of events, but I consider delay dependent on a single event.*)
(6) For all events e^{ξ}, the number of executions $\mathbb{I}^{\xi}$ is known before simulation.

Preparation Event e^{ξ} is expanded into a series of events $e^{\xi,0}$, $e^{\xi,1}$, ..., $e^{\xi,\Delta^{\xi}}$, where Δ^{ξ} is equal to the maximum among $\Delta^{\xi,s}$ for all $s \in \Theta^{\xi}$. The expansion is conducted as follows. First, event $e^{\xi,0}$ is executed as soon as all the scheduling conditions are satisfied, and the state variables $s \in \Theta^{\xi}$ are not changed. Then, event $e^{\xi,1}$ is executed after t^{ξ} time unit after an execution of $e^{\xi,0}$. For all $s \in \Theta^{\xi}$, if $\Delta^{\xi,s} \geq \delta$, $e^{\xi,\delta}$ will increase or decrease s by one, for all $\delta = 1, ..., \Delta^{\xi}$. The i-th execution of event $e^{\xi,\delta}$ for $\delta = 1, ..., \Delta^{\xi}$ are simultaneous.

Constraints (A) The constraints below imply that event $e^{\xi,1}$ is scheduled to exe-

e^{ξ}	event of type ξ
s	state variable
$\mathbb{S}$	set of all state variables
$\mathbb{S}^{\xi}$	set of state variables whose value is conditioned for scheduling event e^{ξ}.
$\Theta^{\xi+}$	the set of state variables that event e^{ξ} will increase its value.
$\Theta^{\xi-}$	the set of state variables that event e^{ξ} will decrease its value.
Θ^{ξ}	$\Theta^{\xi+} \cap \Theta^{\xi-}$
E^{s+}	set of events whose execution increases the value of state variable s.
E^{s-}	set of events whose execution decreases the value of state variable s.
$\Delta^{\xi,s}$	increment or decrement of state variable s when event e^{ξ} is executed.
I^{ξ}	total number of executions of event e^{ξ}
L^{s+}	total number of times that state variable s is increased.
L^{s-}	total number of times that state variable s is decreased.
t^{ξ}	delay between scheduling and execution of event e^{ξ}.
t_i^{ξ}	delay between i-th scheduling and its execution of event e^{ξ}.

Table 7. Notations

$e_i^{\xi,\delta} \geq 0$	time of i-th execution of event $e^{\xi,\delta}$
$\tau_l^{s+} \geq 0$	time when state variable s is increased for the l-th time.
$\tau_l^{s-} \geq 0$	time when state variable s is decreased for the l-th time.
$x_{i,i'}^{\xi} \in \{0,1\}$	equal to 1 if the $i^{'}$ execution of event e^{ξ} is the i-th scheduled one.
$y_{i,l}^{\xi,\delta,s+} \in \{0,1\}$	equal to 1 if the i-th execution of event e^{ξ} is the l-th time that state variable s in incre
$y_{i,l}^{\xi,\delta,s-} \in \{0,1\}$	equal to 1 if the i-th execution of event e^{ξ} is the l-th time that state variable s in decr
$z_{i,l}^{\xi,s+} \in \{0,1\}$	equal to 1 if the i-th scheduling of event e^{ξ} is later than τ_l^{s+}.

Table 8. Decision variables

cute with a delay t^{ξ}, after an execution of $e^{\xi,0}$.

$$e_{i^{'}}^{\xi,1} - e_i^{\xi,0} \geq t_i^{\xi} + M(x_{i,i^{'}}^{\xi} - 1) \qquad \forall \xi, i, i^{'} = 1, ..., I^{\xi} \tag{177}$$

It should be noticed that, if multiple executions of the same event e^{ξ} are allow to exist in the future event list simultaneously, the execution of $e^{\xi,1}$ scheduled by the i-th execution of $e^{\xi,0}$ may be not the i-th execution of $e^{\xi,1}$. Thus, binary variables $x_{i,i^{'}}^{\xi}$ are introduced, and it is equal to one if the $i^{'}$ execution of event $e^{\xi,1}$ is scheduled by the i-th execution of event $e^{\xi,0}$. Since each execution of $e^{\xi,0}$ can schedule one and only one execution of $e^{\xi,1}$, the following constraints should also be satisfied:

$$\sum_{i=1}^{N^{\xi}} x_{i,i^{'}}^{\xi} = 1 \qquad \forall\ \xi, i^{'} = 1, ..., I^{\xi} \tag{178}$$

$$\sum_{i^{'}=1}^{N^{\xi}} x_{i,i^{'}}^{\xi} = 1 \qquad \forall\ \xi, i = 1, ..., I^{\xi} \tag{179}$$

If up to α^ξ multiple executions of event e^ξ are allowed, the following constraints can be added:

$$e_i^{\xi,0} - e_{i-\alpha^\xi}^{\xi,1} \geq 0 \qquad \forall\ \xi,\ i = \alpha^\xi + 1, ..., I^\xi \tag{180}$$

$$\sum_{i'=i+\alpha^\xi}^{I^\xi} x_{i,i'}^\xi = 0 \qquad \forall\ \xi,\ i = 1, ..., I^\xi - \alpha^\xi \tag{181}$$

$$\sum_{i=1}^{i'-\alpha^\xi} x_{i,i'}^\xi = 0 \qquad \forall\ \xi,\ i' = \alpha^\xi + 1, ..., I^\xi \tag{182}$$

If α^ξ is equal to one, the constraints (A) are reduced to:

$$e_i^{\xi,1} - e_i^{\xi,0} \geq t_i^\xi \qquad \forall \xi, i = 1, ..., I^\xi \tag{183}$$

$$e_i^{\xi,0} - e_{i-1}^{\xi,1} \geq 0 \qquad \forall\ \xi,\ i = 2, ..., I^\xi \tag{184}$$

Constraints (B) Binding $e_i^{\xi,\delta}$ and $\tau_l^{s+},\ \tau_l^{s-}$:

$$\tau_l^{s+} - e_i^{\xi,\delta} \geq M(y_{i,l}^{\xi,\delta,s+} - 1) \qquad \forall\ s \in \mathbb{S}, e^\xi \in E^{s+}, \delta = 1, .., \Delta^{\xi,s}, i = 1, .., I^\xi, l = 1, .., L^{s+} \tag{185}$$

$$e_i^{\xi,\delta} - \tau_l^{s+} \geq M(y_{i,l}^{\xi,\delta,s+} - 1) \qquad \forall\ s \in \mathbb{S}, e^\xi \in E^{s+}, \delta = 1, .., \Delta^{\xi,s}, i = 1, .., I^\xi, l = 1, .., L^{s+} \tag{186}$$

$$\tau_l^{s-} - e_i^{\xi,\delta} \geq M(y_{i,l}^{\xi,\delta,s-} - 1) \qquad \forall\ s \in \mathbb{S}, e^\xi \in E^{s-}, \delta = 1, .., \Delta_s^\xi, i = 1, .., I^\xi, l = 1, .., L^{s-} \tag{187}$$

$$e_i^{\xi,\delta} - \tau_l^{s-} \geq M(y_{i,l}^{\xi,\delta,s-} - 1) \qquad \forall\ s \in \mathbb{S}, e^\xi \in E^{s-}, \delta = 1, .., \Delta_s^\xi, i = 1, .., I^\xi, l = 1, .., L^{s-} \tag{188}$$

$$\sum_{\substack{\xi: e^\xi \in E^{s+} \\ i=1,...,I^\xi \\ \Delta=1,...,\Delta_s^\xi}} y_{i,l}^{\xi,\delta,s+} = 1 \qquad \forall\ s \in \mathbb{S}, l = 1, ..., L^{s+} \tag{189}$$

$$\sum_{l=1,...,L^{s+}} y_{i,l}^{\xi,\delta,s+} = 1 \qquad \forall\ \xi, s \in \Theta^{\xi+}, i = 1, ..., I^\xi, \delta = 1, ..., \Delta_s^\xi \tag{190}$$

$$\sum_{\substack{\xi: e^\xi \in E^{s-} \\ i=1,...,I^\xi \\ \Delta=1,...,\Delta_s^\xi}} y_{i,l}^{\xi,\delta,s-} = 1 \qquad \forall\ s \in \mathbb{S}, l = 1, ..., L^{s-} \tag{191}$$

$$\sum_{l=1,...,L^{s-}} y_{i,l}^{\xi,\delta,s-} = 1 \qquad \forall\ \xi, s \in \Theta^{\xi-}, i = 1, ..., I^\xi, \delta = 1, ..., \Delta_s^\xi \tag{192}$$

$$\tag{193}$$

Binary variables $y_{i,l}^{\xi,\delta,s+} \in \{0, 1\}$ are equal to one if the i-th execution of event $e^{\xi,\delta}$ is the l-th time that state variable s in increased. Since events $e^{\xi,1}, ..., e^{\xi,\Delta_s^\xi}$ are expanded from one event e^ξ, and they are executed simultaneously, the following constraints are also added:

$$e_i^{\xi,\delta} = e_i^{\xi,1} \qquad \forall\ \xi,\ i = 1, ..., I^\xi, \delta = 1, ..., \Delta^\xi \tag{194}$$

$$y_{i,l+\delta-1}^{\xi,\delta,s+} = y_{i,l}^{\xi,1,s+} \qquad \forall\ \xi,\ s \in \Theta^{\xi+},\ \delta = 1, ..., \Delta_s^\xi,\ i = 1, ..., I^\xi,\ l = 1, ..., L^{s+} - \Delta_s^\xi + 1 \tag{195}$$

$$y_{i,l+\delta-1}^{\xi,\delta,s-} = y_{i,l}^{\xi,1,s-} \qquad \forall\ \xi,\ s \in \Theta^{\xi-},\ \delta = 1, ..., \Delta_s^\xi,\ i = 1, ..., I^\xi,\ l = 1, ..., L^{s-} - \Delta_s^\xi + 1 \tag{196}$$

Constraints (C) To trigger event $e^{\xi,0}$, the conditions $b_s^\xi \leq s \leq c_s^\xi$ for all state variable $s \in \mathbb{S}^\xi$ should be satisfied. $s \in \mathbb{S}^\xi$ can be categorized into one of the following three situations:

- event e^ξ does not change the value of s, i.e., $s \notin \Theta^\xi$.
- event e^ξ increases the value of s, i.e., $s \in \Theta^{\xi+}$.
- event e^ξ decreases the value of s, i.e., $s \in \Theta^{\xi-}$.

If event e^ξ does not change the value of s, or if it is executed after being scheduled with positive delay, i.e., $s \notin \Theta^\xi$ or $t^\xi > 0$, the following constraints are applied:

$$e_i^{\xi,0} - \tau_l^{s+} \leq M z_{i,l}^{\xi,s+} \quad \forall\ \xi, i = 1, ..., I^\xi,\ s \in \mathbb{S}^\xi, l = 1, ..., L^{s+} \tag{197}$$
$$e_i^{\xi,0} - \tau_l^{s-} \leq M z_{i,l}^{\xi,s-} \quad \forall\ \xi,\ i = 1, ..., I^\xi,\ s \in \mathbb{S}^\xi, l = 1, ..., L^{s-} \tag{198}$$
$$e_i^{\xi,0} - \tau_l^{s-} \geq -M \hat{z}_{i,l}^{\xi,s-} \quad \forall\ \xi,\ i = 1, ..., I^\xi,\ s \in \mathbb{S}^\xi, l = 1, ..., L^{s-} \tag{199}$$
$$e_i^{\xi,0} - \tau_l^{s+} \geq -M \hat{z}_{i,l}^{\xi,s+} \quad \forall\ \xi,\ i = 1, ..., I^\xi,\ s \in \mathbb{S}^\xi, l = 1, ..., L^{s+} \tag{200}$$
$$z_{i,l}^{\xi,s+} + \hat{z}_{i,s_0+l-b_s^\xi}^{\xi,s-} \leq 1 \qquad ?? \tag{201}$$
$$z_{i,l}^{\xi,s-} + \hat{z}_{i,-s_0+l+a_s^\xi}^{\xi,s+} \leq 1 \qquad ?? \tag{202}$$
$$\tag{203}$$

$$e_i^{\xi,0} - \tau_l^{s+} \geq M(z_{i,l}^{\xi,s+} - 1) \quad \forall\ \xi, i = 1, ..., I^\xi,\ s \in \mathbb{S}^\xi, l = 1, ..., L^{s+} \tag{204}$$
$$\tau_l^{s+} - e_i^{\xi,0} > -M z_{i,l}^{\xi,s+} \quad \forall\ \xi,\ i = 1, ..., I^\xi,\ s \in \mathbb{S}^\xi, l = 1, ..., L^{s+} \tag{205}$$
$$e_i^{\xi,0} - \tau_l^{s-} \geq M(z_{i,l}^{\xi,s-} - 1) \quad \forall\ \xi,\ i = 1, ..., I^\xi,\ s \in \mathbb{S}^\xi, l = 1, ..., L^{s-} \tag{206}$$
$$\tau_l^{s-} - e_i^{\xi,0} > -M z_{i,l}^{\xi,s-} \quad \forall\ \xi,\ i = 1, ..., I^\xi,\ s \in \mathbb{S}^\xi, l = 1, ..., L^{s-} \tag{207}$$
$$\tag{208}$$

If $e_i^{\xi,0}$ is executed after $\tau_l^{s+}(\tau_l^{s-})$, $z_{i,l}^{\xi,s+}(z_{i,l}^{\xi,s-})$ is equal to one. If $e_i^{\xi,0}$ is executed before $\tau_l^{s+}(\tau_l^{s-})$, $\hat{z}_{i,l}^{\xi,s+}(\hat{z}_{i,l}^{\xi,s-})$ is equal to one.

If event e^ξ increases the value of s, and it is executed immediately when scheduled, i.e., $s \in \Theta^{\xi+}$ and $t^\xi = 0$, the following constraint should be applied:

$$e^{\xi,0} - \tau_{s_0+l-1-b}^{s-} \geq M(y_{i,l}^{\xi,1,s+} - 1) \tag{209}$$
$$e^{\xi,0} - \tau_{s_0+l-1-a}^{s-} \leq M(1 - y_{i,l}^{\xi,1,s+}) \tag{210}$$

If event e^ξ decreases the value of s, and it is executed immediately when scheduled, i.e., $s \in \Theta^{\xi-}$ and $t^\xi = 0$, the following constraint should be applied:

$$e^{\xi,0} - \tau_{-s_0+l-1+a}^{s+} \geq M(y_{i,l}^{\xi,1,s-} - 1) \tag{211}$$
$$e^{\xi,0} - \tau_{-s_0+l-1+b}^{s+} \leq M(1 - y_{i,l}^{\xi,1,s-}) \tag{212}$$

7.3. Mathematical programming representation of simulation model - V3

Event-scheduling algorithm for DES.

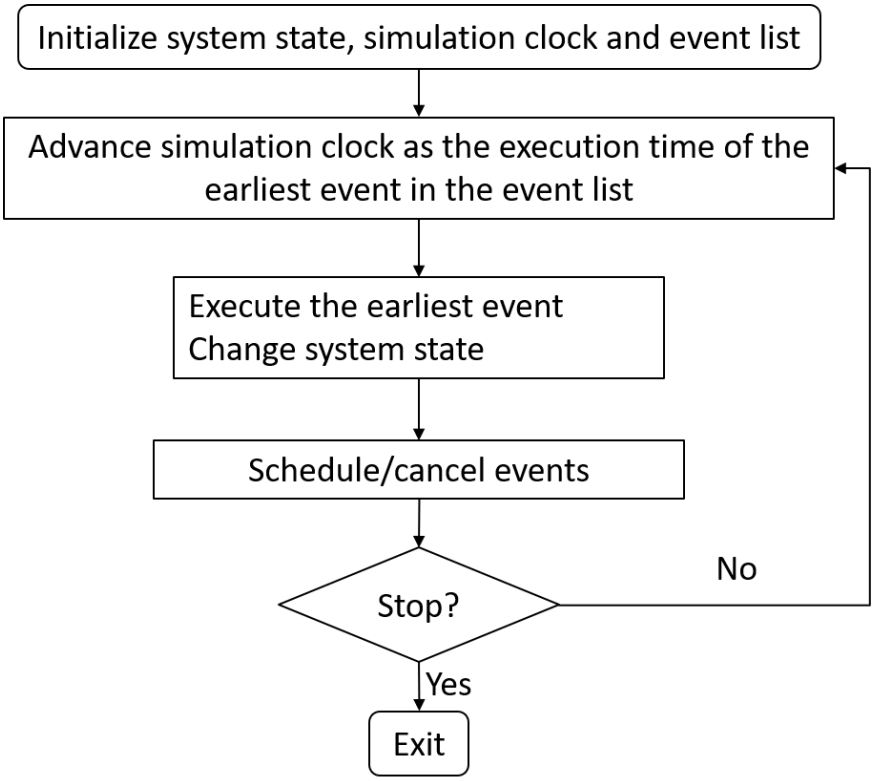


Figure 5. Event-based simulation algorithm.

An equivalent mathematical programming model exists if the following assumptions are satisfied:

(1) For all event e^{ξ}, the *scheduling conditions* are in the form of $b_s^{\xi} \leq s \leq c_s^{\xi}$ combined with logic operator "AND", where s is a state variable, and b_s^{ξ} and c_s^{ξ} are lower and upper bounds.
(2) The scheduling conditions is independent of the history and not changed along time. (It could be possible to define more state variables in case of history dependence and time-variant scheduling conditions.)
(3) An event execution of e^{ξ} leads to (integer) increment or decrement equal to Δ_s^{ξ} of certain state variables s, and Δ_s^{ξ} is not changed along time. (A direct evaluation can be modeled in this way.)
(4) The delay between scheduling and execution time of an event e^{ξ}, denoted by t^{ξ}, is random variate. They can be generated independently from the simulation run. (*This point is different from ERG. In ERG, the delay is dependent on the edge, i.e, a couple of events, but I consider delay dependent on a single event.*)
(5) For all events e^{ξ}, the number of executions N^{ξ} is known before simulation.

$e_i^{\xi,0} \geq 0$	i=1,...,I^ξ	the i-th scheduling time of event e^ξ.
$e_i^{\xi,1} \geq 0$	i=1,...,I^ξ	the i-th execution time of event e^ξ.
$\mathcal{E}_k \geq 0$	k=0,...,$\mathbb{K}$	time of the k-th execution of any events.
$u_k^s \in \mathbb{Z}$	k=0,...,$\mathbb{K}$	value of state variable s just after the k-th event.
$w_{i,k}^\xi \in \{0,1\}$	k=1,...,$\mathbb{K}$	binding $e_i^{\xi,1}$ and $\mathcal{E}_k$.
$x_{i,k}^\xi \in \{0,1\}$	k=0,...,$\mathbb{K}$	equal to one if $\mathcal{E}_k$ schedules $e^{\xi,0}$.
$y_{i,i'}^\xi \in \{0,1\}$		binding $e_i^{\xi,0}$ and $e_{i'}^{\xi,1}$ in case of overtaking.
$z_k^\xi \in \{0,1\}$	k=0,...,$\mathbb{K}$	equal to one if the condition for scheduling e^ξ is true right after $\mathcal{E}_k$.
$v_k^{\xi,s,0} \in \{0,1\}$	k=0,...,$\mathbb{K}$	equal to one if $s_k \leq a^{\xi,s} - 1$
$v_k^{\xi,s,1} \in \{0,1\}$	k=0,...,$\mathbb{K}$	equal to one if $s_k \geq b^{\xi,s} - 1$
$r_k^\xi \in \mathbb{Z}$	k=0,...,$\mathbb{K}$	number of existing parallel executions of $e_i^{\xi,1}$ after $\mathcal{E}_k$ before scheduling.
$n_k^\xi \in \mathbb{Z}$	k=0,...,$\mathbb{K}$	number of scheduled executions of $e_i^{\xi,1}$ after $\mathcal{E}_k$ before scheduling.

Table 9. Notation

Constraints (A): binding $e_i^{\xi,1}$ and $\mathcal{E}_k$:

$$e_i^{\xi,1} - \mathcal{E}_k \geq M(w_{i,k}^\xi - 1) \quad A1 \quad \forall\ \xi, i, k \tag{213}$$

$$\mathcal{E}_k - e_i^{\xi,1} \geq M(w_{i,k}^\xi - 1) \quad A2 \quad \forall\ \xi, i, k \tag{214}$$

$$\sum_k w_{i,k}^\xi = 1 \quad A3 \quad \forall\ \xi, i \tag{215}$$

$$\sum_{\xi,i} w_{i,k}^\xi = 1 \quad A4 \quad \forall\ k \tag{216}$$

$$\sum_k k w_{i+1,k}^\xi - \sum_k k w_{i,k}^\xi \geq 1 \quad A5 \quad \forall\ \xi, i \tag{217}$$

Constraints (B): binding $e_i^{\xi,0}$ and $e_{i'}^{\xi,1}$, where α^ξ is the maximal number of executions existing simultaneously in the event list:

$$e_{i'}^{\xi,1} - e_i^{\xi,0} \geq t_i^\xi + M(y_{i,i'}^\xi - 1) \quad B1 \quad \forall \xi, i, i' = 1, ..., N^\xi \tag{218}$$

$$e_i^{\xi,0} - e_{i'}^{\xi,1} \geq -t_i^\xi + M(y_{i,i'}^\xi - 1) \quad B2 \quad \forall \xi, i, i' = 1, ..., N^\xi \tag{219}$$

$$\sum_{i=1}^{N^\xi} y_{i,i'}^\xi = 1 \quad B3 \quad \forall\ \xi, i' = 1, ..., N^\xi \tag{220}$$

$$\sum_{i'=1}^{N^\xi} y_{i,i'}^\xi = 1 \quad B4 \quad \forall\ \xi, i = 1, ..., N^\xi \tag{221}$$

$$\sum_{i'=i+\alpha^\xi}^{N^\xi} y_{i,i'}^\xi = 0 \quad B5 \quad \forall\ \xi,\ i = 1, ..., N^\xi - \alpha^\xi \tag{222}$$

$$\sum_{i=1}^{i'-\alpha^\xi} y_{i,i'}^\xi = 0 \quad B6 \quad \forall\ \xi,\ i' = \alpha^\xi + 1, ..., N^\xi \tag{223}$$

If $\alpha^\xi = 1$, variables $y^\xi_{i,i'}$ are redundant and constraints (B) are reduced to:

$$e_i^{\xi,1} - e_i^{\xi,0} = t_i^\xi \quad B1 \quad \forall \xi, i = 1, ..., N^\xi \tag{224}$$

Number of executions of event e^ξ waiting in the event list can be a state variable n^ξ, and one condition for scheduling an e^ξ is $n^\xi \leq \alpha^\xi$. Thus, it can be managed as a generic scheduling condition.

Constraints (C): event e^ξ can be scheduled right after $\mathcal{E}_k$ if all state variables s satisfies condition $a_s^\xi \leq s_k \leq b_s^\xi$.

$$e_i^{\xi,0} - \mathcal{E}_k \geq M(x_{i,k}^\xi - 1) \quad C1 \quad \forall\ \xi, k, i \tag{225}$$

$$\mathcal{E}_k - e_i^{\xi,0} \geq M(x_{i,k}^\xi - 1) \quad C2 \quad \forall\ \xi, k, i \tag{226}$$

$$\sum_k x_{i,k}^\xi = 1 \quad C3 \quad \forall\ \xi, i \tag{227}$$

$$b^{\xi,s} - u_k^s \geq M(z_k^\xi - 1) \quad C4 \quad \forall\ \xi, k, s \tag{228}$$

$$u_k^s - a^{\xi,s} \geq M(z_k^\xi - 1) \quad C5 \quad \forall\ \xi, k, s \tag{229}$$

$$u_k^s - (b^{\xi,s} + 1) \geq M(v_k^{\xi,s,1} - 1) \quad C6 \quad \forall\ \xi, k, s \tag{230}$$

$$(a^{\xi,s} - 1) - u_k^s \geq M(v_k^{\xi,s,0} - 1) \quad C7 \quad \forall\ \xi, k, s \tag{231}$$

$$1 - z_k^\xi \leq \sum_{s \in \mathbb{S}^\xi} v_k^{\xi,s,0} + \sum_{s \in \mathbb{S}^\xi} v_k^{\xi,s,1} + v_k^{\xi,r} + v_k^{\xi,N} \quad C8 \quad \forall\ \xi, k \tag{232}$$

$$\sum_{i=1}^{N^\xi} x_{i,k}^\xi = z_k^\xi \quad C9 \quad \forall\ \xi, k \tag{233}$$

$$\sum_k k x_{i+1,k}^\xi - \sum_k k x_{i,k}^\xi \geq 1 \quad C10 \quad \forall\ \xi, i \tag{234}$$

Constraints (D): evolution of state variables

$$u_k^s = u_{k-1}^s + \sum_\xi \sum_{i=1}^{N^\xi} w_{i,k}^\xi \Delta^{\xi,s} \quad D1 \quad \forall\ s, k \tag{235}$$

$$r_k^\xi = r_{k-1}^\xi + X_{k-1}^\xi - \sum_i w_{i,k}^\xi \quad D2 \quad \forall\ \xi, k \tag{236}$$

$$R^\xi - r_k^\xi \geq z_k^\xi \quad D3 \quad \forall\ \xi, k \tag{237}$$

$$r_k^\xi \geq R^\xi v_k^{\xi,r} \quad D4 \quad \forall\ \xi, k \tag{238}$$

$$n_k^\xi = n_{k-1}^\xi + X_{k-1}^\xi \quad D5 \quad \forall\ \xi, k \tag{239}$$

$$N^\xi - n_k^\xi \geq z_k^\xi \quad D6 \quad \forall\ \xi, k \tag{240}$$

$$n_k^\xi \geq N^\xi v_k^{\xi,N} \quad D7 \quad \forall\ \xi, k \tag{241}$$

Constraints (E): others

$$\mathcal{E}_0 = 0 \quad E1 \tag{242}$$

$$\mathcal{E}_k - \mathcal{E}_{k-1} \geq 0 \quad E1 \quad \forall\ k \tag{243}$$

Objective function: with the constraints above, there is a unique solution in terms of event occurring times (solution of the binary variables could be multiple in case of multiple simultaneous events). Thus, the objective can be any function of event occurring time. I tried minimize/maximize the sum of $\mathcal{E}_k$, and they give the same solution.

Conditions for a variable x to be *resource-type* are not valid any more.

(1) $\forall\ \xi$ and s, upper bound c_s^ξ is monotonically increasing on x.
(2) $\forall\ \xi$ and s, lower bound b_s^ξ is monotonically decreasing on x.

The reason is that increasing c_s^ξ or decreasing b_s^ξ will tighten constraints C6 and C7. To be simple, we consider b only.

$$b - u_k^s \geq M(z_k^\xi - 1) \quad C4 - b \quad \forall\ \xi, k, s \tag{244}$$
$$u_k^s - (b+1) \geq M(v_k^{\xi,s,1} - 1) \quad C6 - b \quad \forall\ \xi, k, s \tag{245}$$

(when $u = b + 1$, event e^ξ cannot be scheduled.)

If b is increased to $b + 1$:

$$(b+1) - u_k^s \geq M(z_k^\xi - 1) \quad C4 - (b+1) \quad \forall\ \xi, k, s \tag{246}$$
$$u_k^s - (b+2) \geq M(v_k^{\xi,s,1} - 1) \quad C6 - (b+1) \quad \forall\ \xi, k, s \tag{247}$$

(when $u = b + 1$, event e^ξ must be scheduled.)

A group of relaxed constraints are:

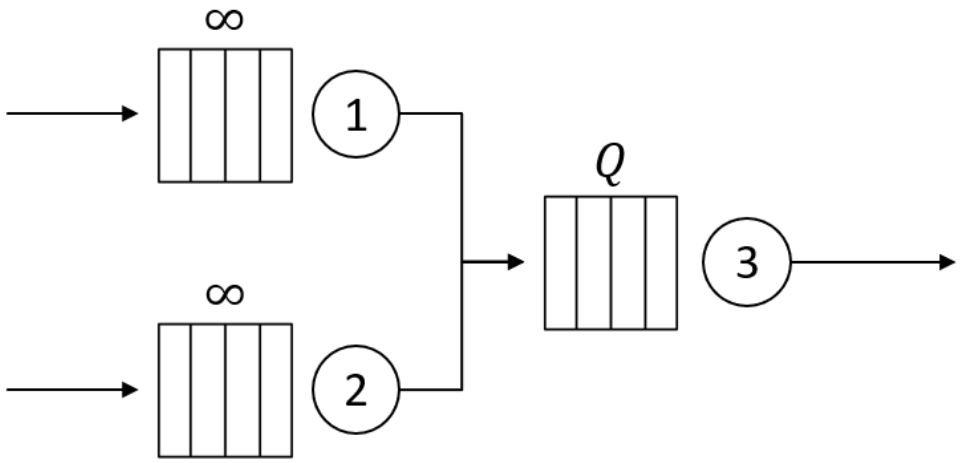
$$(b+1) - u_k^s \geq M(z_k^\xi - 1) \quad C4 - (b+1) \quad \forall\ \xi, k, s \tag{248}$$
$$u_k^s - (b+1) \geq M(v_k^{\xi,s,1} - 1) \quad C6 - (b) \quad \forall\ \xi, k, s \tag{249}$$

(when $u = b + 1$, event e^ξ can be scheduled or not.)

Todo:

(1) What kind of performance indicators can be used? (Regular function of time, in scheduling area. Weighted sum, maximum. Refer to book on scheduling.)

7.4. Merge



Machine 1 has higher priority in releasing a job compared with machine 2.

Figure 6. Example: merge.

Variable	Event	Condition to schedule	Delay	# executions	State change
$e^{s,1}$	Start m1	$m^1 \leq 0$	0	1	m^1++
$e^{f,1}$	Finish m1	$1 \leq m^1 \leq 1$	t^1	1	m^1++
$e^{d,1}$	Depart m1	$m^1 \geq 2\ AND$ $q \geq 1$	0	1	$m^1 = m^1 - 2,\ q--$
$e^{s,2}$	Start m2	$m^2 \leq 0$	0	1	m^2++
$e^{f,2}$	Finish m2	$1 \leq m^2 \leq 1$	t^2	1	m^2++
$e^{d,2}$	Depart m2	$m^2 \geq 2\ AND$ $q \geq 1\ AND$ $m^1 \leq 1$	0	1	$m^2 = m^2 - 2,\ q--$
$e^{s,3}$	Start m3	$m^3 \leq 0\ AND$ $q \leq Q-1$	0	1	$m^3++,\ q++$
$e^{d,3}$	Depart m3	$m^3 \geq 1$	t^3	1	m^3--

Table 10. Merge-S3M111

MP model:

$$\min \sum_k \mathcal{E}_k \tag{250}$$

$$e_i^{(\xi,j),1} - \mathcal{E}_k \geq M(w_{i,k}^{\xi,j} - 1) \quad \xi \in \{s,f,d\}, j \in \{1,2,3\}, \forall i,k \tag{251}$$

$$\mathcal{E}_k - e_i^{(\xi,j),1} \geq M(w_{i,k}^{\xi,j} - 1) \quad \xi \in \{s,f,d\}, j \in \{1,2,3\}, \forall i,k \tag{252}$$

$$\sum_k w_{i,k}^{\xi,j} = 1 \quad \forall\ \xi \in \{s,f,d\}, j \in \{1,2,3\}, i \tag{253}$$

$$\sum_{(\xi,j),i} w_{i,k}^{\xi,j} = 1 \quad \forall\ k \tag{254}$$

$$\sum_k k w_{i+1,k}^{\xi,j} - \sum_k k w_{i,k}^{\xi,j} \geq 1 \quad \forall\ \xi \in \{s,f,d\}, j \in \{1,2,3\}, i \tag{255}$$

$$e_i^{s,j,1} - e_i^{s,j,0} \geq 0 \quad j = 1,2,3, \forall\ i \tag{256}$$

$$e_i^{f,j,1} - e_i^{f,j,0} \geq t_i^j \quad j = 1,2, \forall\ i \tag{257}$$

$$e_i^{d,j,1} - e_i^{d,j,0} \geq 0 \quad j = 1,2, \forall\ i \tag{258}$$

$$e_i^{d,3,1} - e_i^{d,3,0} \geq t_i^3 \quad \forall\ i \tag{259}$$

$$e_i^{\xi,j,0} - \mathcal{E}_k \geq M(x_{i,k}^{\xi,j} - 1) \quad \forall\ \xi \in \{s,f,d\}, j = 1,2,3,k,i \tag{260}$$

$$\mathcal{E}_k - e_i^{\xi,j,0} \geq M(x_{i,k}^{\xi,j} - 1) \quad \forall\ \xi \in \{s,f,d\}, j = 1,2,3,k,i \tag{261}$$

$$m_k^j = m_{k-1}^j + \sum_{i=1}^{N^j} (w_{i,k}^{s,j} + w_{i,k}^{f,j} - 2w_{i,k}^{d,j}) \quad j = 1,2, \forall\ k \tag{262}$$

$$m_k^3 = m_{k-1}^3 + \sum_{i=1}^{N^3} (w_{i,k}^{s,3} - w_{i,k}^{d,3}) \quad \forall\ k \tag{263}$$

$$q_k = q_{k-1} + \sum_{i=1}^{N^3} w_{i,k}^{s,3} - \sum_{i=1}^{N^1} w_{i,k}^{d,1} - \sum_{i=1}^{N^2} w_{i,k}^{d,2} \tag{264}$$

$$m_k^j \geq M(z_k^{s,j} - 1) \quad j = 1,2,3, \forall\ k \tag{265}$$

$$1 - m_k^j \geq M(z_k^{f,j} - 1) \quad j = 1,2, \forall\ k \tag{266}$$

$$m_k^j - 1 \geq M(z_k^{f,j} - 1) \quad j = 1,2, \forall\ k \tag{267}$$

$$m_k^j - 2 \geq M(z_k^{d,j} - 1) \quad j = 1,2, \forall\ k \tag{268}$$

$$q_k - 1 \geq M(z_k^{d,j} - 1) \quad j = 1,2, \forall\ k \tag{269}$$

$$1 - m_k^1 \geq M(z_k^{d,2} - 1) \quad \forall\ k \tag{270}$$

7.5. Merge - 2 machines in station 3

Variable	Value	Initialization	Description
e^1	0,1,2	2	number of empty machines in station 1
f^1	0,1,2	0	number of finished jobs in station 1
e^2	0,1	1	number of empty machines in station 2
f^2	0,1	0	number of finished jobs in station 2
e^3	0,1	1	number of empty machines in station 3
q	0,...,Q	Q	number of available spaces in queue

Table 11. State variables: Merge-S3M211

Variable	Event	Condition to schedule	Delay	# executions	State change
ai $e^{f,1}$	Finish m1	$1 \leq e^1 \leq 2$	t^1	2	f^1++
$e^{d,1}$	Depart m1	$1 \leq f^1 \leq 2$ AND $1 \leq q \leq Q$	0	1	$e^1++,\ f^1--,\ q--$
$e^{s,2}$	Start m2	$1 \leq e^2 \leq 1$	0	1	e^2--
$e^{f,2}$	Finish m2	$1 \leq e^2 \leq 1$	t^2	1	f^2++
$e^{d,2}$	Depart m2	$1 \leq f^2 \leq 1$ AND $1 \leq q \leq Q$ AND $0 \leq f^1 \leq 0$	0	1	$f^2--, e^2++,\ q--$
$e^{s,3}$	Start m3	$1 \leq e^3 \leq 1$ AND $0 \leq q \leq Q-1$	0	1	$e^3--,\ q++$
$e^{d,3}$	Depart m3	$1 \leq e^3 \leq 1$ AND	t^3	1	e^3++
		$0 \leq q \leq Q-1$			

Table 12. Events: Merge-S3M211

The complete algorithm for solving RAP–PC is summarized in Algorithm 1. The resource capacities are initialized to the lower bound. The searching region of RAP–PC–MIP is initialized to $\mathbb{X}$, and the lower and upper bounds of the objective function, C^L and C^U, respectively, are set considering the upper bound and lower bound of the capacity of each resource. Lines 7 to 11 show that approximate cuts are generated and used in the model when infeasible solutions are found. Once a feasible solution is found, the upper bound C^U, which is also the incumbent solution, can be updated after comparing the value of the found feasible solution and that of the current incumbent. Then, all the currently used approximate cuts are replaced by exact cuts of the DIS. If there are only exact cuts in RAP–PC–MIP, the solution is the new lower bound C^L. The algorithm terminates when the gap between the upper bound and lower bound is within a tolerance or the time limit is exceeded.

11. Numerical analysis

11.1. Multiple-server merge

A multiple-server merge queueing system is shown in Figure 7. The multiple-server merge queue is a generalization of the system presented in Section 4.1, where the number of parallel servers in station 1, 2 and 3 is equal to s^1, s^2 and s^3, respectively. The state variables are changed accordingly. To describe the state of multiple-server station j, two state variables g^j and h^j are needed to represent the number of idle servers and the number of finished jobs of each the station. The state variable q is used to represent the number of available space in buffer 3.

The events composing the DES model are shown in Table 13. The start event of station 1 and 2 is scheduled when there are at least one empty server, and their execution decreases the number of empty servers by one. The scheduling condition of finish event $e^{f,j}$ is the same as $e^{s,j}$, but its execution will increase the number of finished jobs by one. Event $e^{f,j}$ is a multi-execution event with positive delays, an event to count the number of executions of it should be introduced. However, the event $e^{s,j}$ plays that role and the number of executions in the event list is equal to $(s^j - g^j)$. Similarly with single-server system, the departure of station 1 requires that there is at least one finished job in the station and there is at least one space available in buffer 3, but the departure of station 2 also requires that there is no finished job in station

Algorithm 4 MIP–based algorithm.

Input:

Lower bound $\mathbf{a} = [a_1, ..., a_J]$ and upper bound $\mathbf{b} = [b_1, ..., b_J]$ of resource capacity $\mathbf{x}$, such that $a_j \leq x_j \leq b_j \ \forall \ j = 1, ..., J$.

Tolerance of optimality gap ε_{opt}.

Optional input: time limit of the algorithm T_{lim}.

Ensure:

Sample-path global optimal $\mathbf{x}^*$.

Initialize system with lower bound $\mathbf{x} \leftarrow \mathbf{a}$
Initialize incumbent with upper bound $\mathbf{x}^* \leftarrow \mathbf{b}$.
Initialize lower bound of the objective $C^L \leftarrow \mathbf{c}^T\mathbf{a}$.
Initialize upper bound of the objective $C^U \leftarrow \mathbf{c}^T\mathbf{b}$.
Add initial constraints which defines $\mathbb{X}$ to the RAP–PC–MIP.
while $C^U - C^L > \varepsilon_{opt}$ and T_{lim} is not exceeded. **do**
  while There exists at least one violated performance constraint **do**
    Generate one approximate cut $CA(\bar{\mathbf{x}}, l)$ for each violated constraints l and add all the generated cuts to the RAP–PC–MIP.
    $\bar{\mathbf{x}} \leftarrow$ solution of the RAP–PC–MIP.
    Simulate the system of $\bar{\mathbf{x}}$.
  end while
  Update upper bound and incumbent $C^U \leftarrow \mathbf{c}^T\bar{\mathbf{x}}, \ \mathbf{x}^* \leftarrow \bar{\mathbf{x}}$ if $\mathbf{c}^T\bar{\mathbf{x}} < C^U$.
  if There exist approximate cuts in RAP–PC–MIP **then**
    For all the currently used approximate cuts $CA(\bar{\mathbf{x}}^r, l)$, find dominating infeasible solution $\bar{\mathbf{x}}_d(\bar{\mathbf{x}}^r)$ and replace approximate cuts $CA(\bar{\mathbf{x}}^r, l)$ by exact cuts $CE(\bar{\mathbf{x}}_d(\bar{\mathbf{x}}^r), l)$ of the DIS.
    $\bar{\mathbf{x}} \leftarrow$ solution of the RAP–PC–MIP.
    Simulate the system of $\bar{\mathbf{x}}$.
    Update lower bound $C^L \leftarrow \max\{\mathbf{c}^T\bar{\mathbf{x}}, \ C^L\}$.
  end if
end while

1. The departure of station 1 and 2 will increase the number of empty servers by one, decrease the number of finished jobs by one and decrease the number of available space by one. As for station 3, the start and departure event can be scheduled if there is at least one empty server and one job in buffer 3. Thus, $e^{s,3}$ is used to count the number executions in the event list of $e^{d,3}$. Execution of $e^{s,3}$ will decrease the number of empty server by one and increase the available space in buffer 3 by one.

12. Conclusion

References

Alfieri, A., & Matta, A. (2012). Mathematical programming formulations for approximate simulation of multistage production systems. *European Journal of Operational Research*, *219*(3), 773–783.

Alfieri, A., Matta, A., & Pastore, E. (2020). The time buffer approximated buffer allocation problem: A row–column generation approach. *Computers & Operations Research*, *115*, 104835.

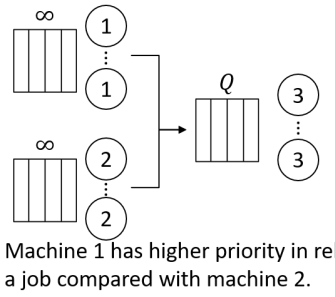


Figure 7. Example: multi-server merge.

Variable	Event	Condition to schedule	Delay	β^{ξ}	State change
$e^{s,1}$	Start 1	$1 \leq g^1$	0	1	$g^1 - -$
$e^{f,1}$	Finish 1	$1 \leq g^1$	t^1	s^1	$h^1 + +$
$e^{d,1}$	Depart 1	$1 \leq h^1$& $q \geq 1$	0	1	$g^1 + +, h^1 - -,\ q - -$
$e^{s,2}$	Start 2	$1 \leq g^2$	0	1	$g^2 - -$
$e^{f,2}$	Finish 2	$1 \leq g^2$	t^2	s^2	$h^2 + +$
$e^{d,2}$	Depart 2	$1 \leq h^2$ & $q \geq 1$ & $h^1 \leq 0$	0	1	$g^2 + +, h^2 - -,\ q - -$
$e^{s,3}$	Start 3	$1 \leq g^3$ & $q \leq Q - 1$	0	1	$g^3 - -,\ q + +$
$e^{d,3}$	Depart 3	$1 \leq g^3$ & $q \leq Q - 1$	t^3	s^3	$g^3 + +$

Table 13. Events to simulate a multi-server merge queueing system.

Chan, W. K., & Schruben, L. (2008). Optimization models of discrete-event system dynamics. *Operations Research*, *56*(5), 1218–1237.

Law, A. M. (2014). *Simulation modeling and analysis* (Vol. 5). McGraw-Hill New York.

Matta, A. (2008). Simulation optimization with mathematical programming representation of discrete event systems. In *2008 winter simulation conference* (pp. 1393–1400).

Pedrielli, G., Alfieri, A., & Matta, A. (2015). Integrated simulation–optimisation of pull control systems. *International Journal of Production Research*, *53*(14), 4317–4336.

Tan, B. (2015). Mathematical programming representations of the dynamics of continuous-flow production systems. *IIE Transactions*, *47*(2), 173–189.

Weiss, S., & Stolletz, R. (2015). Buffer allocation in stochastic flow lines via sample-based optimization with initial bounds. *OR spectrum*, *37*(4), 869–902.

Zhang, M., & Matta, A. (2020). Models and algorithms for throughput improvement problem of serial production lines via downtime reduction. *IISE Transactions*, 1–15.

Zhang, M., Matta, A., & Alfieri, A. (2020). Simulation optimization with mathematical programming representation of discrete event systems. In *2020 winter simulation conference* (p. accepted).

Zhang, M., Pastore, E., Matta, A., & Alfieri, A. (n.d.). Buffer allocation problem in production flow lines: a new benders decomposition based exact solution approach. *under review*.