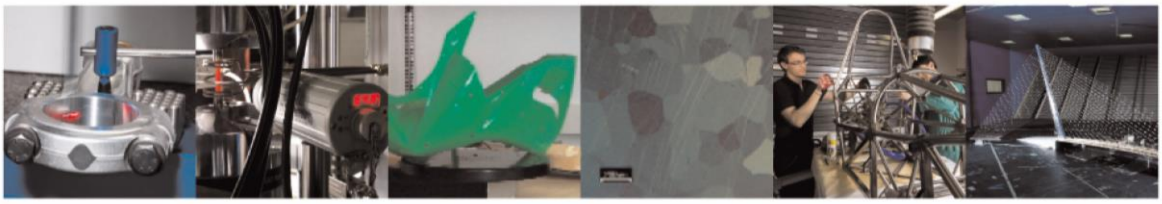
POLITECNICO MILANO 1863

DIPARTIMENTO DI MECCANICA



Design and control of manufacturing systems: a discrete event optimisation methodology

Pedrielli, G.; Matta, A.; Alfieri, A.; Zhang, M.

This is an Accepted Manuscript of an article published by Taylor & Francis in INTERNATIONAL JOURNAL OF PRODUCTION RESEARCH on 20 Dec 2017, available online: http://dx.doi.org/10.1080/00207543.2017.1412532

Design and Control of Manufacturing Systems: a Discrete Event Optimization Methodology

Giulia Pedrielli

School of Computing, Informatics, and Decision Systems Engineering

Arizona State University.

email address: Giulia.Pedrielli@asu.edu

Andrea Matta

Department of Mechanical Engineering
Politecnico di Milano

Corresponding author

Via La Masa 1, 20156 Milano, Italy

email address: andrea.matta@polimi.it

phone: +390223998530

Arianna Alfieri

Department of Management and Production Engineering
Politecnico di Torino

email address: arianna.alfieri@polito.it

Mengyi Zhang

School of Mechanical Engineering
Shanghai Jiao Tong University

email address: myrazhang@foxmail.com

Abstract

Simulation–optimization has gained a great attention due to its success in the design of complex manufacturing systems. In this paper, we look at manufacturing as a special class of queueing systems and propose the Discrete Event Optimization (DEO) methodology, which provides a formal way to develop integrated mathematical models for the simultaneous simulation and optimization. In the case the obtained model is a mixed integer linear programming model, the methodology provides a formal way to generate approximations of them. The analytical properties of DEO models are analyzed for the first time in the framework of sample path optimization and mathematical programming. The methodology represents a reference for the use of mathematical programming as a way to model simulation–optimization for queueing systems. The applicability of the DEO methodology to complex problems is showed using the task and buffer allocation problem in a production line.

Mathematical Programming, Manufacturing Systems, Simulation, Optimization, Queueing Systems

1 Introduction

Simulation–optimization has received an important attention as an effective technique to improve the performance of manufacturing systems. In fact, for this particular category of systems, discrete event models are usually adopted to reproduce their dynamics.

In this paper, we look into the general queueing systems with specific examples in the manufacturing domain. In this area, simulation–optimization is typically carried out through iterative procedures alternating two separated modules: an optimization module for the choice of the best system configuration (e.g., the number of machines at each stage of a transfer manufacturing line, the storage capacity between workstations) and a simulation module for the evaluation of the system performance. In such framework, the simulation module, mostly considered as a black-box, takes as input the solution of the optimization module and generates the performance of the system as output (Law, 2007). The output of the simulation module is then used as input for the optimization module, which is solved again. This iterative procedure continues until the optimal solution is found or a predefined stopping condition is satisfied (Kleijnen, 2008; Spall, 2003). The simulation module is usually a discrete event simulator, whereas several methods are available in the literature for the optimization module (Fu et al., 2015; Healy and Schruben, 1991; Jin and Schmeiser, 2003). Examples are response surface methodology (Myers et al., 2009), stochastic approximation (Kushner and Yin, 1997), ranking and selection (Boesel et al., 2003), meta–heuristics (Hong and Nelson, 2006), random search (Andrandóttir, 2005; Zabinsky, 2009) and mathematical programming (Fu et al., 2005).

In the area of computing, several modeling techniques are used for simulation and property verification such as Petri Nets, Finite State Automata (FSA), and Event Relationship Graphs (ERGs). In particular, ERGs are a general language for modeling and simulation of Discrete Event Systems (DESs) (Schruben, 1983). They have been successfully applied for the evaluation of the DESs performance (Savage et al., 2005; Chan, 2005; Liu et al., 2012; Buss and Sanchez, 2002) and showed to over perform other formalisms in terms of modeling power and generality (Cassandras and Lafortune, 2008; Cao and Zhang, 2008).

Chan and Schruben (2008) provided a general scheme to translate ERG simulation models into a set of equivalent mathematical programming models. This represented a breakthrough in the understanding of the deep relationship between simulation and optimization and boosted the research in the area of simulation–optimization. Matta (2008) and Alfieri and Matta (2012b) extended the work in Chan and Schruben (2008) to the simulation–optimization models of multi–server tandem flow lines. An integrated model is generated for the purpose of solving a simulation–optimization problem for queueing systems. In fact, if mathematical programming is adopted to solve the optimization model and the simulation model is also generated as a mathematical program, the simulation becomes strongly related to the optimization of the system. The basic idea is that, if a set of constraints exists to model the system dynamics, it is then possible to embed them within an optimization model. This means that the feasibility of the solution given by the optimizer does not need to be checked by an external simulator. In other words, we optimize the system while simulating it.

Mathematical programming for simultaneous simulation and optimization establishes an alternative paradigm where the evaluation of the candidate solution is intertwined with the search of the optimum. However, despite the above listed positive aspects of the integrated simulation–optimization approach, when the optimization is taken into account, mathematical models can become Mixed Integer Linear Programming (MILP) models and the related computational burden has to be considered.

Alfieri and Matta (2012b) then proposed the concept of *time buffer* (TB), introduced by Matta (2008), to approximate the MILP models into linear programs (LP) for specific problems. In general, time buffer is a continuous variable used to approximate the corresponding integer variable defined in the original (MILP) model. This substitution prevents the optimization model from becoming MILP, although introducing an approximation. To approximate a discrete variable with a time buffer, it has to be possible to formally describe its effect on the events characterizing the system dynamics. This holds when the system behavior can be formulated as a set of max-plus type equations (Baccelli et al. (1992), Buzacott and Shantikumar (1993)).

Pedrielli et al. (2015b) proposed a first generalization for the cases provided in (Alfieri and Matta, 2012b,a; Pedrielli et al., 2015a). Other researchers have recently used integrated mathematical programming models to deal with various simulation–optimization problems ((Helber et al., 2011a,c,b; Weiss and Stolletz, 2015; Tan, 2015; Stolletz and Weiss, 2013; Schwarz and Stolletz, 2013; Weiss and Stolletz, 2013; Weiss et al., 2017)), however, no formal properties of the methodology have been investigated so far.

The main objective of this paper is to present a methodology for integrated simulation–optimization of manufacturing systems that can be represented as queueing systems. We refer to this methodology as Discrete Event Optimization (DEO), due to the fact that events are interpreted as decision variables in a unique mathematical program, which integrates system performance evaluation and optimization. Specifically, this paper provides the foundations to develop integrated simulation–optimization of DESs, thus extending the work of Chan and Schruben (2008) in the scope of optimization, while providing deeper theoretical support with respect to (Alfieri and Matta, 2012b,a; Pedrielli et al., 2015a).

Another objective of this work is to provide theoretical structure to integrated mathematical programming models for simulation–optimization. Mathematical programming is exploited to derive second order properties (convexity and monotonicity) of both objective function and constraints. These properties allow the analysis of the asymptotic behavior of the algorithm used for solving the simulation–optimization problem in the framework of Sample Path Optimization (SPO).

Such a general framework also contributes to the current approaches in simulation–optimization along three directions:

- Exploration vs. exploitation dilemma is "solved by construction": a single mathematical model taking care of optimization while considering the system dynamics allows to solve the problem within the mathematical programming framework. Existence and uniqueness of the global optimum relate back to the properties of the objective function and of the feasible region;

- No independent and expensive simulation runs are required to evaluate, separately, objective function and constraints. In fact, most of the simulation–optimization techniques need independent simulations to estimate the different stochastic function describing the optimization problem. Differently, the mathematical programming model exploits the dependency of the stochastic functions;

- A mathematical program reveals structures characterizing the problem graph (e.g., modularity, separation, symmetry), which can be used for making the solution approach quicker. Sensitivity analysis can also be exploited to compute gradients serving an external search procedure.

For all these reasons, DEO represents an important contribution in providing a remarkably different perspective that leads to the solution of several problems characterizing most of the known techniques. Researchers and practitioners can use DEO to generate formal representations of simulation–optimization problems and develop *ah hoc* efficient algorithms on the top of them. Besides this, the same framework can be used as support to already existing gradient based searching techniques.

The paper is organized as follows. Section 2 revises the main concepts of the Event Relationship Graphs (ERGs). DEO is presented in section 3. Section 4 analyses the properties of DEO models. The applicability of the DEO framework to a complex realistic problem is proposed in Section 5. Section 6 concludes the paper.

2 Event Relationship Graphs

Event Relationship Graphs (ERGs) are possibly one of the most effective languages to model the dynamics of discrete event systems (Schruben, 1983; Law, 2007; Savage et al., 2005; Chan and Schruben, 2006; Matta et al., 2014).

The vertices of an ERG represent the state changes that take place when a particular type of event occurs. The directed arcs of the graph represent the relationships between pairs of events. The state changes associated with each event vertex appear in braces. Labels on directed arcs representing all the dynamic and logical relationships between the events specify the conditions and time delays between the occurrences of events. Following the definition presented in Askin and Standridge (1993), we can refer to Figure 1 and define a generic arc in an ERG:

> *After event A occurs, if condition i_{AB} is then true, event B will immediately been scheduled to occur t_{AB} time units into the future.*

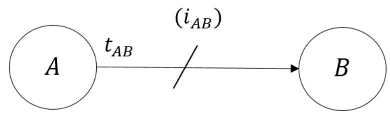


Figure 1: Basic Element of an event relationship graph (ERG) (Chan and Schruben (2006))

Figure 2 shows the ERG representation of a queueing system with R_0 identical parallel servers, which process jobs in batches of b units.

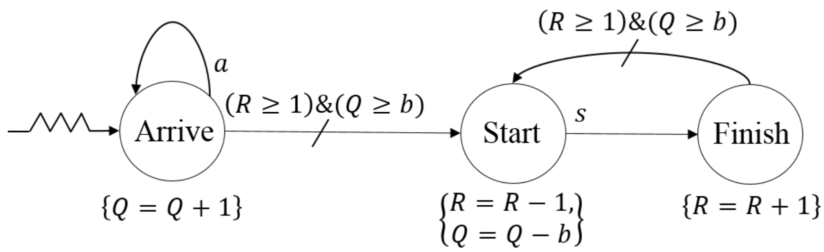


Figure 2: ERG for a batch-processing queue with parallel resources (Chan and Schruben (2008)).

In Figure 2, the state is described by two integers: R, representing the number of idle resources at each point in time, and Q, which refers to the number of jobs currently waiting in line for service. The input data for simulating the ERG are the (random) customer inter–arrival times, a, and the (random) service times, s.
The queue in Figure 2 is assumed to be initially empty with all the servers idle. Therefore, the initial value of the number of available resources, R, is the total number of parallel servers in the system, R_0, and Q is initially set equal to zero, i.e., the initial state of the queue $Q_0 = 0$. Initially scheduled events are indicated by broken arrows as in Law (2007). The only event initially scheduled here is the first job arriving at time zero. The "&" is used here to denote the Boolean AND operator. A formal demonstration of the translation of an ERG into an optimization model is the main contribution of Chan and Schruben (2008).

The mathematical model corresponding to the simulation of the system in Figure 2 is provided in the following (Chan and Schruben, 2008), for the particular case in which only one server is in the system, i.e., $R_0 = 1$ and the batch size is equal to 1, i.e., $b = 1$:

$$
\begin{aligned}
P : \min \qquad & \mathcal{H} = \sum_{i=1}^{n} (A_i + S_i + F_i) \\
& \text{s.to} \\
& A_{i+1} = A_i + a_{i+1} \;\; i = 1, \ldots, n-1 \\
& F_i = S_i + s_i \;\; i = 1, \ldots, n \\
& S_i \geq A_i \;\; i = 1, \ldots, n \\
& S_i \geq F_{i-1} \;\; i = 2, \ldots, n \\
& A_1 = 0 \qquad A_i, S_i, F_i \;\; \text{n.s.r.} \; \forall i
\end{aligned}
$$

where A_i, S_i, F_i represent for each simulated customer i the arrival time, the start time of service and the finish time of service, respectively. An ERG, when observed from a simulation point of view, is a graphical representation of the underlying event scheduling process. Instead, when an ERG is considered from a mathematical programming perspective, the unconditional arcs in the ERG impose equality constraints, and the conditional arcs in the ERG impose greater than or equal to constraints on event occurrence times. From this point of view, an ERG is a set of constraints on the event occurrence times. Each equality or inequality constraint directly impacts at most on the two event types connected by the arc, with a right hand side for an equality constraint being the time delay for the arc. Simulating an ERG, with a sequence of generated random variates as input data, involves executing all the event instances as soon as they are feasible. This is equivalent to the objective of minimizing the times of all the event instances subject to the constraints imposed by the arcs in the ERG.

ERGs are independent from any specific representation of a discrete event model. They define a discrete event system model at a more fundamental level than any commercial simulation language. In fact, they express the relationships between the different event functions in the simulation code. Process interaction and activity scanning paradigms, which are more typical in commercial simulation codes, can be translated into their basic ERGs (Schruben, 2010).

3 Methodology

The DEO methodology is presented in this section. Section 3.1 describes the class of queueing systems addressed in this paper, providing the main notation and the terminology adopted. Section 3.2 presents the mathematical programs, which extend those in (Chan and Schruben, 2008) to include also the optimization component.

3.1 Notation and terminology

The DESs we consider are queueing networks with the set of servers $\mathbb{J} = \{0, \dots, J+1\}$ and the set of possible transaction routes for customer i ($i \in \mathbb{N},\ \mathbb{N} = \{1, \dots, n\}$) between servers, represented by $\mathbb{Q}_i = \{(j, j') | j, j' \in \mathbb{J}\}$, $\forall i$. For each pair (j, j'), the connection between j and j' belongs to $\mathbb{Q}_i$ if and only if customer i can directly flow from server j to j'. The *source*, represented by index $j = 0$, is the server having no predecessor. The *sink* is, instead, the server having no successor and it is indexed by $J + 1$. The source represents an infinite external arrival stream of customers, whereas the sink is the output gate through which customers are released from the network. Queues may have either finite or infinite capacity. We consider a general setting in which no explicit condition has to be imposed over the system layout, i.e., server j is not the $j-$th on which each customer is served, i.e., j simply represents the server label. Analogously, customer i is not the $i-$th in the sequence.

Let E^{ξ}_{ij} and e^{ξ}_{ij} denote the events occurring in the system and their occurrence times, respectively, where $\xi \in \mathbb{T}$ is the event type (e.g., arrival, start of service, etc.), and the pair (i, j) indicates the customer i and the server j the event refers to. We assume that customer i at server j undergoes a service activity that is temporally defined by a start event E^s_{ij} occurring at time e^s_{ij} and a completion event E^f_{ij} occurring at time e^f_{ij}; the duration of the service process is $t^{fs}_{ij,ij}$.

In the case of stochastic DES, $t^{\xi\xi'}_{ij,i'j'}$ may represent the duration of a general activity, where ξ and ξ' generalize the start and finish events and indexes $i'j'$ represent any entity–server pair, and may follow some known statistical distributions.

The set of all the possible events is identified by $\mathbb{W}$. The flow of each entity i is determined by the occurrence of a set of events $\mathbb{W}^i = \left\{E^{\xi}_{ij}, \xi \in \mathbb{T},\ j|(j,j') \text{ or } (j',j) \in \mathbb{Q}_i\right\},\ i \in \mathbb{N}$. Each event E^{ξ}_{ij} in the set $\mathbb{W}^i$ has a set $\mathbb{W}^{I\xi}_{ij}$ of input events, i.e., *triggering events*, and a set $\mathbb{W}^{O\xi}_{ij}$ of output events, i.e., *triggered events*. Notice that elements in the sets $\mathbb{W}^{I\xi}_{ij}$ and $\mathbb{W}^{O\xi}_{ij}$ might not be in the set $\mathbb{W}^i$.

3.2 Integrated models for simulation–optimization

The mathematical formulation for integrated simulation–optimization problems can be generated by recognizing the existence of two main ways, in which events can constrain each other, i.e., delay and control. Hence, two main categories of events can be defined: (1) *natural events*, and (2) *control events*. While natural events are only responsible for delay–type connections, control events generate control–type connections. Control events are grouped in the set $\mathbb{W}^C$, while natural events are in the set $\mathbb{W}^N$, and we have $\mathbb{W}^N \cap \mathbb{W}^C = \{0\}$.

Delay–type connections In this case, events E^{ξ}_{ij} and $E^{\xi'}_{ij'}$ have to happen with a certain relative delay $t^{\xi\xi'}_{ij,ij'}$. Such a relationship between events is translated into a dynamics constraint of the type:

$$e^{\xi}_{ij} - e^{\xi'}_{ij'} \geq t^{\xi\xi'}_{ij,ij'} \tag{1}$$

It is important to highlight that the customer sequence $\{i\}$ is here meant to represent the order in which customers are served by the system and not the arrival order. However, in the case a sequencing decision is taken, then $\{i\}$ is not a–priori determined and control constraints of the type (2) need to be generated for the integer case (although $\kappa^{\xi\xi'}_{ij,i'j'}$ is a parameter in this case). All the delay–type constraints are linear in nature. Indeed, $t^{\xi\xi'}_{ij,ij'}$ might even be boolean, but it represents an input parameter (i.e., it is known) and, as such, does not compromise the linearity of the relationship.

Control–type connections Control constraints are generated from all the event pairs where the delay between a natural event and a control event, $e^{\xi}_{ij} - e^{\xi'}_{i'j'}$, is controlled through a decision variable, which can be either continuous, $s^{\xi\xi'}_{ij,i'j'}$, or binary, $\kappa^{\xi\xi'}_{ij,i'j'}$. We obtain the following constraints:

$$
\begin{aligned}
e^{\xi}_{ij} - e^{\xi'}_{i'j'} + s^{\xi\xi'}_{ij,i'j'} \geq 0 \qquad & \forall (\xi, i, j) \in \mathbb{W}^N,\ \left(\xi', i', j'\right) \in \mathbb{W}^C \cap \mathbb{W}^{I\xi}_{ij}, \\
e^{\xi}_{ij} - e^{\xi'}_{i'j'} - M \cdot \kappa^{\xi\xi'}_{ij,i'j'} \geq -M \qquad & \forall (\xi, i, j) \in \mathbb{W}^N,\ \left(\xi', i', j'\right) \in \mathbb{W}^C \cap \mathbb{W}^{I\xi}_{ij},
\end{aligned}
\tag{2}
$$

3.2.1 General formulation

According to the previous considerations, resulting in the constraints (1)–(2), the optimization of a system corresponds to the search of the best set of control events $\mathbb{W}^C = \left\{E^{\xi}_{ij}\right\}$, as well as the values of $s^{\xi\xi'}_{ij,i'j'}$ and $\kappa^{\xi\xi'}_{ij,i'j'}$ in constraints (2) such that the resulting event occurrence times $\left\{e^{\xi}_{ij}\right\}$ satisfy some target performance. We can now formulate the DEO simulation–optimization problem. In the following model, for the sake of simplifying the expressions, we will characterize each event with the triple (ξ, i, j) instead of referring to the event E^{ξ}_{ij}. The general formulation results:

$$
P : \min \quad \mathcal{H} = \sum_{(\xi,i,j)\in\mathbb{W}} \alpha^{\xi}_{ij} e^{\xi}_{ij} +
$$

$$
+ \sum_{(\xi,i,j)\in\mathbb{W}^N} \sum_{(\xi',i',j')\in\mathbb{W}^C\cap\mathbb{W}^{I\xi}_{ij}} \left(\beta^{\xi\xi'}_{ij,i'j'} s^{\xi\xi'}_{ij,i'j'} + \gamma^{\xi\xi'}_{ij,i'j'} \kappa^{\xi\xi'}_{ij,i'j'}\right) + \vartheta \cdot \varepsilon \tag{3}
$$

$$
\sum_{(\xi,i,j)\in\mathbb{W}^C} g^{\xi}_{ij}\left(e^{\xi}_{ij}\right) - \varepsilon \leq \mu^* \tag{4}
$$

$$
e^{\xi}_{ij} - e^{\xi'}_{ij'} \geq t^{\xi\xi'}_{ij,ij'} \qquad \forall (\xi, i, j) \in \mathbb{W}, \left(\xi', i, j'\right) \in \mathbb{W}^{I\xi}_{ij} \tag{5}
$$

$$
e^{\xi}_{ij} - e^{\xi'}_{i'j'} + s^{\xi\xi'}_{ij,i'j'} \geq 0 \qquad \forall (\xi, i, j) \in \mathbb{W}^N,\ \left(\xi', i', j'\right) \in \mathbb{W}^C \cap \mathbb{W}^{I\xi}_{ij}, \tag{6}
$$

$$
e^{\xi}_{ij} - e^{\xi'}_{i'j'} - M \cdot \kappa^{\xi\xi'}_{ij,i'j'} \geq -M \qquad \forall (\xi, i, j) \in \mathbb{W}^N,\ \left(\xi', i', j'\right) \in \mathbb{W}^C \cap \mathbb{W}^{I\xi}_{ij}, \tag{7}
$$

Equation (3) is the objective function, having as decision variables the event times e^{ξ}_{ij}, and the control parameters $s^{\xi\xi'}_{ij,i'j'}$ and $\kappa^{\xi\xi'}_{ij,i'j'}$. Function (3) can consider a single or multiple objectives depending on the values of the input parameters α^{ξ}_{ij}, $\beta^{\xi\xi'}_{ij,i'j'}$ and $\gamma^{\xi\xi'}_{ij,i'j'}$. The term $\vartheta \cdot \varepsilon$ serves the purpose to penalize finite sample path solutions that do not meet the desired performance (i.e., violate the constraint (4) if the decision variable ε is not considered). This penalization approach has an impact on the implementation of the algorithm to solve the problem but not on the asymptotic properties. Equation (4) is the performance constraint, where μ^* is the target performance and g is any function of the control event times.

The delay–type connections are translated into constraints (5), generated following the scheme in (1). According to these constraints, the service sequence for customer i is defined by the precedence relationships between the events E^{ξ}_{ij} and $E^{\xi'}_{ij'}$, occurring at time e^{ξ}_{ij} and $e^{\xi'}_{ij'}$, respectively. Parameters $t^{\xi\xi'}_{ij,ij'}$ form the collection of realizations of the random variables characterizing the queueing system (e.g., arrival times and service times).

As stated in the previous sections, we assume to know the probabilistic characterization of the input stochastic processes. Hence, we can generate $t_{ij,ij'}^{\xi\xi'}$ as realizations of known random variables.

In constraints (6)–(7), generated as those in (2), variables $e_{i,j}^{\xi}$ and $e_{i'j'}^{\xi'}$ represent the time occurrences of two events, relating customer i' on server j' and customer i on server j, which are linked by a control. If the relationship between the two event times is boolean (7), the constraint function has the form $(1 - \kappa_{ij,i'j'}^{\xi\xi'}) \cdot M$, where $\kappa_{ij,i'j'}^{\xi\xi'}$ is a binary decision variable and M is a large number. Instead, in case of continuous relationship, inequality (6) is function of the continuous variable $s_{ij,'i'j'}^{\xi\xi'}$.

The main feature of DEO models is that they are based on events rather than on states, which generally grow faster than events. Notice that when $\beta_{ij,i'j'}^{\xi\xi'} = 0$ and $\gamma_{ij,i'j'}^{\xi\xi'} = 0 \;\; \forall \left(\xi\xi'\right), \left(ij, i'j'\right)$ in equation (3), $\left\{s_{ij,i'j'}^{\xi\xi'}\right\}$ and $\left\{\kappa_{ij,i'j'}^{\xi\xi'}\right\}$ are input parameters, and the performance constraint is not present (and, consequently, $\varepsilon = 0$), we are solving a pure simulation problem as in the representation of Chan and Schruben (2008). Hence, the only decision variables are the event times e_{ij}^{ξ} and we refer to the function $\mathcal{H}$ as χ. Instead, we will refer to function (3) as $\mathcal{S}$ when $\alpha_{ij}^{\xi} = 0$.

Furthermore, since $i = 1, \ldots, n$, the model size is a function of n, i.e., the number of simulated customers. In fact, as the simulation length increases (i.e., n increases), also the number of decision variables and constraints increases. In particular, we can notice a quadratic growth of the model in the number of considered customers.

Example To show how the DEO framework can be used to derive integrated mathematical programming models, we refer to a simple example of simulation–optimization problem. Let us consider the optimization problem of choosing the service rate in a GI/G/1 queue to balance the cost of increasing the service rate with the benefit of reducing the steady-state expected waiting time. The ERG for this problem is the same as in Figure 2 with $b = 1, R = 1$. Three event types characterize the system: arrival (corresponding to the event time e_i^a), start of the process (corresponding to the event time e_i^s) and the departure event (corresponding to the event time e_i^d). We can derive the following mathematical programming model considering the relationships between events:

$$P : \min \quad \mathcal{H} = c^{sd} \cdot s^{sd}$$

$$e_i^a - e_{i-1}^a \geq t_{a,i} \;\; i = 2, \ldots, n \tag{8}$$

$$e_i^d - e_i^s - s^{sd} \cdot \left(F^{-1}(u_i)\right) \geq 0 \;\; \forall i \tag{9}$$

$$e_{i+1}^s - e_i^d \geq 0 \;\; i = 1, \ldots, n-1 \tag{10}$$

$$\sum_i \left(e_i^s - e_i^a\right) \leq n \cdot \mu^* \tag{11}$$

With reference to the general formulation, we minimize the continuous decision variable s^{sd}, which corresponds to $s_{ij,i'j'}^{\xi\xi'}$ in the general problem P. In particular, the superscript sd corresponds to the two event types *start* and *departure*. The subscript of the decision variable should be $\left(ij, i'j'\right)$. However, since the expected processing time is the same for every job and there is only one stage in the system, we avoid the notation $s^{sd}_{i1,i1}$ and we refer to the decision variable as s^{sd}. In the model, c^{sd} plays the role of $\beta^{\xi\xi'}_{ij,i'j'}$. The same simplification in notation, as highlighted for the decision variable, applies to the rest of the model.

The first three sets of constraints simply describe the dynamics of the GI/G/1 queue, that is, the inter–arrival time $t_{a,i}$ between the arrival of customers $i-1$ and i (constraints (8)), the service time $s^{sd} \cdot \left(F^{-1}(u_i)\right)$ elapsing between the start of the service of customer i and its departure from the queue (constraints (9)), and the precedence between customer i and $i+1$, i.e., customer $i+1$ can start service only after the departure of customer i (constraints (10)). In constraints (9), the third term represents the variability in the service time, seen as the product between the average service time and a random variable generated according to the distribution of the processing time F (the term u_i is a random number used for the generation).

Finally, constraint (11) represents the upper bound on the total (and then average, since the number of customers is a–priori defined) waiting time.

The model is a linear programming model, thus it can be easily solved. The important aspect is that this simple example shows how the integration of simulation and optimization can be achieved.

4 Solution Methodology

In this section, we provide a generalization of the approximation scheme provided in Alfieri and Matta (2012b) in order to define the continuous counterpart of the model in equations (3)–(7). We also extend the authors' results in terms of structural relationship between the approximate and exact formulations. Such approximation mechanism serves the purpose of decreasing the complexity and the computational effort required to solve the sample path optimization problem when the variables of type $\kappa^{\xi\xi'}_{ij,i'j'}$ are present.

Finally, we analyze the properties of the continuous models (being them exact or approximate models), in terms of asymptotic convergence, when the number of simulated entities (e.g., customers, parts, etc.) becomes large in the setting of sample path optimization.

4.1 Approximation mechanisms

When problem P in (3)–(7) is a MILP, it can be approximated to an LP by using the construct of *time buffers* proposed in Alfieri and Matta (2012b).

As a result of such approximation, the objective function becomes:

$$\mathcal{S} = \sum_{(\xi,i,j)\in\mathbb{W}^N} \sum_{(\xi',i',j')\in\mathbb{W}^C\cap\mathbb{W}^{I\xi}_{ij}} \beta^{\xi\xi'}_{ij,i'j'} s^{\xi\xi'}_{ij,i'j'} + \vartheta\cdot\varepsilon$$

where $\beta^{\xi\xi'}_{ij,i'j'}$, i.e., the cost related to the continuous decision variables, is assumed given as input.

Given the generic event types ξ, ξ', the customers pair $\left(i, i'\right)$ and the servers j, j', if the events E^{ξ}_{ij} and $E^{\xi'}_{i'j'}$ are involved in constraints of type (7), each integer variable $\kappa^{\xi\xi'}_{ij,i'j'}$ can be replaced by the continuous counterpart $s^{\xi\xi'}_{ij,i'j'}$. It is important to notice that $s^{\xi\xi'}_{ij,i'j'}$ does not represent a pure relaxation of the binary variable $\kappa^{\xi\xi'}_{ij,i'j'}$ as $s^{\xi\xi'}_{ij,i'j'}$ might take any positive value and it is not limited in the interval $[0,1]$. Formally, the following constraint:

$$e^{\xi}_{ij} - e^{\xi'}_{i'j'} - M \cdot \kappa^{\xi\xi'}_{ij,i'j'} \geq -M \qquad \forall (\xi, i, j) \in \mathbb{W}^N,\ \left(\xi', i', j'\right) \in \mathbb{W}^C \cap \mathbb{W}^{I\xi}_{ij} \tag{12}$$

is approximated by:

$$e^{\xi}_{ij} - e^{\xi'}_{i'j'} + s^{\xi\xi'}_{ij,i'j'} \geq 0 \qquad \forall (\xi, i, j) \in \mathbb{W}^N,\ \left(\xi', i', j'\right) \in \mathbb{W}^C \cap \mathbb{W}^{I\xi}_{ij} \tag{13}$$

This approximation is structural and general because it operates on the connection between two events independently from their nature. As a consequence, DEO approximate models can be developed for any application field or problem that fits within its framework.

4.2 Formal relationship between exact and approximate DEO models

When performing the aforementioned approximation, it is important to understand the relationship between the approximate and the exact solutions. In order to derive such formal relationship, we refer to the simulation version of problem P, i.e., $\beta^{\xi\xi'}_{ij,i'j'} = \gamma^{\xi\xi'}_{ij,i'j'} = \vartheta = 0$ and $\alpha^{\xi}_{ij} = 1$.

We define the *bounding set* as follows:

Definition 1. *Let $\mathbb{K}^*$ be the optimal solution of the exact optimization model, i.e., the collection of the optimal values for the binary decision variables $\left\{\kappa^{*,\xi\xi'}_{ij,i'j'}\right\}$. The set*

$$\mathbb{B} = \left\{ \left(\kappa^{l,\xi\xi'}_{ij,i'j'}, \kappa^{u,\xi\xi'}_{ij,i'j'}\right) : \kappa^{l,\xi\xi'}_{ij,i'j'} \leq \kappa^{*,\xi\xi'}_{ij,i'j'} \leq \kappa^{u,\xi\xi'}_{ij,i'j'} \right\}$$

is the bounding set and characterizes the relationship between the exact and the approximate problem.

The bounding set is derived by analyzing inequalities (12) and their continuous counterpart (13). The following theorem states the main result:

Theorem 1. *Given a pair of events $\left\{(\xi, i, j), \left(\xi', i', j'\right)\right\} : \left(\xi', i', j'\right) \in \mathbb{W}^{I\xi}_{ij}$, and the event times $\tilde{e}^{\xi}_{ij}$ and e^{ξ}_{ij} resulting from the approximate simulation model and from the exact simulation model when $\kappa^{\xi\xi'}_{ij,i'j'} = 1$, respectively, if the following condition holds, for each pair of events:*

$$e^{\xi}_{ij} \leq \tilde{e}^{\xi}_{ij} - s^{\xi\xi'}_{ij,i'j'} \tag{14}$$

then the objective function value from the approximated model $\tilde{\chi}$ is smaller than the one from the exact model, i.e., the approximate system performs better than the exact one. Therefore, $\kappa^{\xi\xi'}_{ij,i'j'}$ is the lower bound on the solution, $\kappa^{l,\xi\xi'}_{ij,i'j'}$. If, on the other hand, the following condition holds, for *each pair of events:*

$$e_{ij}^{\xi} \geq \tilde{e}_{ij}^{\xi} - s_{ij,i'j'}^{\xi\xi'} \tag{15}$$

then the objective function value from the approximate model $\tilde{\chi}$ *is larger than the one from the exact model, i.e., the approximate system performs worse than the exact one. Therefore,* $\kappa_{ij,i'j'}^{\xi\xi'}$ *is the upper bound on the solution,* $\kappa_{ij,i'j'}^{u,\xi\xi'}$*.*

Proof. See Appendix. □

4.3 Asymptotic properties of continuous DEO models in the sample path optimization setting

In this section, we will focus on continuous simulation–optimization models, where $\alpha_{ij}^{\xi} = 0$ for all $(\xi, i, j) \in \mathbb{W}$, $\gamma_{ij,i'j'}^{\xi\xi'} = 0$ for all $(\xi, i, j) \in \mathbb{W}^N, (\xi', i', j') \in \left(\mathbb{W}^C \cup \mathbb{W}_{ij}^{I\xi}\right)$, and only continuous decision variables $s_{ij,i'j'}^{\xi\xi'}$ for all $(\xi, i, j) \in \mathbb{W}^N, (\xi', i', j') \in \left(\mathbb{W}^C \cup \mathbb{W}_{ij}^{I\xi}\right)$ are defined. These models can either refer to exact problems in which decision variables are continuous or to approximate problems devised as shown in section 4.1. Moreover, we will assume that the objective function is simply the sum of all the variables, i.e., $\beta_{ij,i'j'}^{\xi\xi'} = 1 \, \forall (\xi, i, j) \in \mathbb{W}^N, (\xi', i', j') \in \left(\mathbb{W}^C \cup \mathbb{W}_{ij}^{I\xi}\right)$.

Since the models are solved according to the sample path optimization approach, we use the results in Robinson (1996), which provides the asymptotic characterization of sample path optimization algorithms. The analysis is made of three main parts: 1) analysis of the second order properties in the context of simulation and optimization; 2) analysis of the constraints of the integrated model P; 3) application of the results in Robinson (1996) to the integrated simulation–optimization model. The second order properties of the considered optimization models and related simulation models guarantee the regularity conditions at the basis of constraint classification and existence results. The second part of the analysis is required since Robinson (1996) does not consider stochastically constrained problems. Then, once the first two parts are characterized, we can apply the main results in Robinson (1996) and prove the convergence in our setting.

4.3.1 Preliminaries

In order to simplify the notation, we will use the matrix forms to represent the models instead of the extended notation used in section 3.2. In order to make the theoretical presentation simpler, we will separate the optimization and the simulation model that were presented in their general form in section 3.2.1.

In the following, let $\mathbf{F}_n \subset \mathbb{X}_n \times \mathbb{R}_+^{||\mathbb{T}|| \times n} \times \mathbb{R}_+$ be the *feasible* region for the approximate optimization problem (for a finite sample path of size n), where $\mathbb{X}_n$ is the domain for the continuous decision variables $\mathbf{s}$, $\mathbb{R}_+^{||\mathbb{T}|| \times n}$ is the domain for the event times and $\mathbb{R}_+^n$ the domain for the ε variable. Since the main results will be related to the behavior of the continuous solution $\mathbf{s}$ rather than the event times $\mathbf{e}$, it is useful to define the projection of the feasible region $\mathbf{F}_n$ onto the decision variable space $\mathbb{X}_n$. We will refer to this set, representing the sample path–feasible solutions, as Σ_n. Finally, in order to highlight the role of the sample size n, we will refer to the continuous decision variables and to the event times as $\mathbf{s}_n$ and $\mathbf{e}_n$ respectively.

The primal (on the left) and the dual (on the right) integrated simulation–optimization models, in their matrix forms, are the following:

$$
\begin{array}{rlrl}
\min & \mathcal{S}_n\left[(\mathbf{b}_i(\tau_i))_{i=1}^m\right] = \mathbf{1}'\mathbf{s}_n + \vartheta \cdot \varepsilon & \max & \mathbf{b}^1(\tau)\mathbf{u}_{\mathcal{D}} + \mathbf{b}^2(\tau)\mathbf{u}_{\mathcal{P}} - \mu^* \cdot \nu \quad (16) \\
& \text{s.t.} & & \\
& \mathbf{A}^{\mathcal{D}}\mathbf{e}_n \geq \mathbf{b}^1(\tau) & & \mathbf{A}^{\mathcal{D}'}\mathbf{u}_{\mathcal{D}} \leq \mathbf{0} \quad (17) \\
& \mathbf{A}^{\mathcal{P}}\left[\mathbf{e}_n|\mathbf{s}_n\right] \geq \mathbf{b}^2(\tau) & & \mathbf{A}^{\mathcal{P}'}\mathbf{u}_{\mathcal{P}} \leq \mathbf{1} \quad (18) \\
& \varepsilon - \sum_{\nu \in \mathbb{W}^C} \mathbf{g}(\mathbf{e}_n) \geq -\mu^* & & \nu \leq \vartheta \\
& \mathbf{e}_n \geq 0,\ \varepsilon \geq 0,\ \mathbf{s}_n \in \mathbb{X}_n & & \mathbf{u} \geq 0,\ \nu \geq 0
\end{array}
$$

The vector $[e_n|\mathbf{s}_n]'$, where $[\cdot|\cdot]'$ is the row vector obtained by the concatenation of two column vectors, contains the decision variables $\mathbf{s}$ and the event times $\mathbf{e}$ that, together with the ε variable, represent the variables of the primal model, while $\mathbf{u}' = [\mathbf{u}_{\mathcal{D}}|\mathbf{u}_{\mathcal{P}}]$ and ν represent the dual variables. The matrix $\mathbf{A} = \left[\mathbf{A}^{\mathcal{D}}|\mathbf{A}^{\mathcal{P}}\right]$ is an $l \times m$-dimensional matrix, where l represents the number of constraints, not including the performance constraint(s) in equation (4), and m the number of decision variables. According to the definitions provided in section 3, constraints (17) and (18) are the same as (5) and (6), respectively.

The m-dimensional vector of the right hand side $\mathbf{b} = \left[\mathbf{b}^1|\mathbf{b}^2\right] = \{b_1, b_2, \ldots, b_m\}$ consists of the realizations of the random variables, B, which are assumed to follow univariate distributions, i.e., $B \sim \mathcal{V}^B(\tau)$, where τ refers to the parametrization of the considered distribution. The link between the realizations and the parameters of the sampling distribution is made explicit through the notation $\mathbf{b}(\tau)$.

The objective of the primal problem, $\mathcal{S}_n(\mathbf{s}_n(\mathbf{b}(\tau), \mu^*), \varepsilon(\mathbf{b}(\tau), \mu^*))$, is a function of $\mathbf{s}_n$, which is itself a function of the right hand side $\mathbf{b}$ parametrized over τ.

The same modeling approach can be applied to the simulation model.

$$
\begin{array}{rlrl}
\min & \chi\left[(\mathbf{s}_{n,i}(\tau_i), \mathbf{b}_i(\tau_i))_{i=1}^m\right] = \mathbf{1}'e_n & \max & \mathbf{b}^1(\tau)\mathbf{u}_{\mathcal{D}} + \mathbf{b}^2(\tau)\mathbf{u}_{\mathcal{P}} \\
& \text{s.t.} & & \\
& \mathbf{A}^{\mathcal{D}}\mathbf{e}_n \geq \mathbf{b}^1(\tau) & & \mathbf{A}^{\mathcal{D}'}\mathbf{u}_{\mathcal{D}} \leq \mathbf{0} \\
& \mathbf{A}^{\mathcal{P}}\mathbf{e}_n \geq \left[\mathbf{b}^2(\tau)|\mathbf{s}_n\right] & & \mathbf{A}^{\mathcal{P}'}\mathbf{u}_{\mathcal{P}} \leq \mathbf{1} \\
& e_n \geq 0 & & \mathbf{u} \geq 0
\end{array}
$$

The model is the same as the optimization one, with two main differences: 1) no performance constraint is present and 2) $\mathbf{s}$ are known parameters, as $\mathbf{b}$, and not decision variables. The only primal decision variables are the event times $\mathbf{e}$.

When interested in studying the behavior of function χ with respect to only $\mathbf{s}_n$ or $\mathbf{b}$, we will use $\chi(\mathbf{s}_n, \cdot)$ ($\mathbf{s}_n$ fixed, $\mathbf{b}$ variable) and $\chi(\cdot, \mathbf{b})$ ($\mathbf{s}_n$ variable, $\mathbf{b}$ fixed), respectively.
The formulation just presented allows to explicitly indicate the set of control constraints (equations (6)), having the decision variables $\mathbf{s}$ as right hand side, and the natural dynamic constraints (equations (5)), not containing it. Again, the dependency of both $\mathbf{b}$ and $\mathbf{s}_n$ on the parameter τ is made explicit through the notation $\mathbf{b}(\tau)$ and $\mathbf{s}_n(\tau)$.

In order to characterize the sample path problem, we will rely on the mathematical programming framework and make use of the definitions in (Yao and Shanthikumar, 1991) of *SICX(sp)* and *SDCX(sp)* representing monotone convexity notions in the sample path sense. In particular, they refer to stochastic increasing and convex, and stochastic decreasing and convex in the sample path (*sp*) sense, respectively.

4.3.2 Second order properties

We first prove monotonicity and convexity of the objective function with respect to the stochastic right hand side in the approximate optimization model. The presented analysis is based on the works of Yao and Shanthikumar (1991) and Shaked and Shanthikumar (1988), although the reference framework of these contributions is not the Mathematical Programming Representation (MPR).
A contribution within the MPR framework is given in Chan (2005), where the second order properties of the finishing times in multi–server flow lines are proved. However, the models proposed in Chan (2005) are in the scope of simulation.

In this section, we will adopt the notation $\mathcal{S}_n(\mathbf{b})$ when it is important to emphasize the dependency of the objective from the right hand side $\mathbf{b}$, whereas the notation $\mathcal{S}_n(\mathbf{s}_n)$ will be used to stress the relationship between the objective function and the continuous variable solution $\mathbf{s}_n$.
The following property characterizes the relationship between the objective function $\mathcal{S}_n$ and τ parameterizing the distribution of B.

Property 1. *(Second order properties for $\mathcal{S}_n$) If $\{\mathbf{b}_i(\tau_i)\}$ are m families of random variables and $\{\mathbf{b}_i(\tau_i)\} \in$ SICX(sp), then $\mathcal{S}_n\left[(\mathbf{b}_i(\tau_i))_{i=1}^m\right]$ is SICX(sp) in $(\tau_i)_{i=1}^m$.*

Proof. See Appendix. □

Moreover, the optimization model is a minimization problem and the performance constraint is a $\leq$–type constraint; hence, as the right hand side increases, the objective function cannot increase. This proves that the objective function $\mathcal{S}_n^*(\mathbf{b})$ is decreasing in the target performance μ^*.

Exploiting the matrix formulation of the primal and dual models provided at the beginning of the section, we can further characterize the function χ and the average system performance $\hat{\Lambda}$.

Property 2. *(Second order properties for χ) If $\{\mathbf{s}_i(\tau_{n,i})\}$ are m families of random variables and $\{\mathbf{s}_{n,i}(\tau_i)\} \in$ SICX(sp), then $\chi\left[(\mathbf{s}_{n,i}(\tau_i), \cdot)_{i=1}^m\right]$ is SICX(sp) in $(\tau_i)_{i=1}^m$. If $\{\mathbf{b}_i(\tau_i)\}$ are m families of random variables and $\{\mathbf{b}_i(\tau_i)\} \in$ SICX(sp), then $\chi\left[(\cdot, \mathbf{b}_i(\tau_i))_{i=1}^m\right]$ is SICX(sp) in $(\tau_i)_{i=1}^m$.*

Proof. The proof is the same as Property 1 and, therefore, it is not reported. □

Corollary 1. *Let $\Lambda(\mathbf{s}, B)$ be the expected value of the target performance μ defined as:*

$$\Lambda(\mathbf{s}, B) \triangleq E_B[\mu], \tag{19}$$

and let $\hat{\Lambda}(\mathbf{s}_n, \mathbf{b})$ be its estimator:

$$\hat{\Lambda}(\mathbf{s}_n, \mathbf{b}) \triangleq \frac{1}{n}\sum_{i=1}^{n}\frac{g(e_{iJ})}{i}. \tag{20}$$

The average performance $\left\{\hat{\Lambda}(\mathbf{s}_n, \mathbf{b})\right\} \in$ SICX(sp) in the service times $\mathbf{b}$ and $\left\{\hat{\Lambda}(\mathbf{s}_n, \mathbf{b})\right\} \in$ SDCX(sp) in the continuous variables $\mathbf{s}_n$.

Proof. See Appendix. □

4.3.3 Constraints characterization

In the following, we will use the notation $\mathbf{s}_n^*$ and $\mathcal{S}_n^*$ to refer to the optimal solution and to the objective function of the finite sample path problem (keeping the same notation adopted in the previous sections). The infinite sample path solution will be denoted as $\mathbf{s}$ or $\mathbf{s}_\infty$, whereas the objective function will be referred to as $\mathcal{S}$ or $\mathcal{S}_\infty$, alternatively.

Let $\lambda(\mathbf{s}, B)$ be the difference between the expected value of the actual performance and the target performance:

$$\lambda(\mathbf{s}, B) \triangleq \Lambda(\mathbf{s}, B) - \mu^*, \tag{21}$$

and let $\hat{\lambda}(\mathbf{s}, \mathbf{b})$ be its estimator:

$$\hat{\lambda}(\mathbf{s}_n, \mathbf{b}) \triangleq \hat{\Lambda}(\mathbf{s}_n, \mathbf{b}) - \mu^*. \tag{22}$$

The estimates defined in (20) and (22) are sample averages. The expected values (19) and (21) are functions of the continuous variables $\mathbf{s}$ and of the collection of random variables B.
We will denote the expected values with $\Lambda(\mathbf{s}, B)$ and $\lambda(\mathbf{s}, B)$ when we want to stress that the property described is related to the considered random variables (i.e., the distribution taken into account), whereas the notation $\Lambda(\mathbf{s}, \cdot)$ and $\lambda(\mathbf{s}, \cdot)$ will be adopted when the property is independent from the distributions.

Using the previously introduced notation, the general formulation in (4)–(6), can be rewritten as:

$$\begin{aligned}\min_{s,e,\varepsilon\in\Sigma} \mathcal{S}_n(\mathbf{b})\\ \text{s.t.}\\ (5)--(6)\\ \hat{\lambda}(\mathbf{s}_n,\mathbf{b}) \leq \varepsilon\end{aligned} \tag{23}$$

Assumption 1. *The system under analysis is stationary and the target performance μ^* is such that $\mu^* \geq \mu_{min}$, being μ_{min} the best performance that can be reached by the system in a steady state.*

We first characterize the asymptotic properties of the performance function estimate (constraint (4)) to the expected performance value. Adopting the same arguments of Assumption 3.1 in (Haskell et al., 2012), we prove the following lemma.

Lemma 1. *The following holds for the approximate optimization model:*

(a) *$\Lambda(\mathbf{s}, B)$ is Lipschitz continuous in the domain of $\mathbf{s}$ for P_B almost all $\tau \in \mathbb{R}^{||\tau||}$, where P_B refers to the density implied by the collection of random variables $\{B\}$. Then there exists a function $\Pi : \mathbb{R}^{||\tau||} \to \mathbb{R}$ such that*

$$||\Lambda(\mathbf{s}_1, B) - \Lambda(\mathbf{s}_2, B)|| \leq \Pi(\tau)||\mathbf{s}_1 - \mathbf{s}_2||,$$

for P_B almost all $\tau \in \mathbb{R}^{||\tau||}$ and such $\Pi(\tau)$ is integrable.

(b) *The moment generating function of $\Pi(\tau)$, denoted as $M_{\Pi(\tau)}(l)$, is finite for all the l in a neighborhood of 0.*

Proof. See Appendix. □

Note that $\Pi(\tau)$ is a stochastic function since it is related to the distance between random variables $||\mathbf{s}_1 - -\mathbf{s}_2||$.

4.3.4 Asymptotic convergence

In the context of stochastic optimization, epi–convergence is one of the most used concepts in the analysis of the asymptotic properties (Shapiro, 2003; Lachou, 1998; Attouch, 1991; King and Wets, 1991; Kall, 1986; Rockafellar and Wets, 1998; Topsoe, 2006; Wets, 1991).
We prove epi–convergence for the optimization objective function using the result that if the objective function $\mathcal{S}_n^*$ epi–converges to $\mathcal{S}^*{}_\infty$, then the optimal solution $\mathbf{s}_n^*$ converges to $\mathbf{s}_\infty^*$ (Robinson, 1987; Rockafellar and Wets, 1998; Robinson, 1996).

In the following, we will use the notation $\mathbf{s}$ and $\mathcal{S}$ to indicate the infinite sample path solution and the objective function, respectively, while $\mathbf{s}_n$ and $\mathcal{S}_n$ will indicate the optimal solution and the objective function of the finite sample path case, as in the previous sections.

The next lemma shows that the sample average estimator is an unbiased estimator of the performance expected value and, as a result, the sample path–constraint set converges to the true constraints set (Shapiro, 2003).

Lemma 2. $\sup\left\{|\lambda(\mathbf{s},B)-\hat{\lambda}(\mathbf{s},B)|\colon \mathbf{s}\in\Sigma\right\}\rightarrow 0$ *as* $n\rightarrow\infty$ *almost surely. As a result,* $\hat{\lambda}(\mathbf{s})$ *converges to* $\lambda(\mathbf{s})$ *uniformly on* Σ *almost surely.*

Proof. See Appendix. □

In order to apply the fundamental results in (Robinson, 1996), we need to show that Assumptions A–B in Definitions 2.2 and 2.3 in (Robinson, 1996) hold and this is proved by Lemma 3 and Lemma 4.

Lemma 3 (Assumption A, Definition 2.2 in Robinson (1996), page 517)**.** *Function* $\mathcal{S}_n$ *satisfies the following conditions: (a) for each* $1\le n<\infty$, $\mathcal{S}_n$ *is lower semi–continuous. (b)* $\mathcal{S}_n\rightarrow_{epi}\mathcal{S}$*, i.e., it epi–converges to* $\mathcal{S}$*.*

Proof. See Appendix. □

Lemma 4 (Assumption B, Definition 2.3 in Robinson (1996), page 517)**.** $\mathcal{S}$ *is proper and the set of minimizers of the finite sample path optimization problem* $\mathbb{S}_n^*=\{\mathbf{s}_n\in\Sigma_n|\mathcal{S}_n=\mathcal{S}_n^*\}$ *is not empty and it is compact.*

Proof. See Appendix. □

Having analyzed the constraints, using Lemma 4, we can now characterize the relationship between $\hat{\lambda}(\cdot,\cdot)$ and $\lambda(\cdot,\cdot)$. First, we separately analyze $\hat{\lambda}(\cdot,\cdot)$ and $\lambda(\cdot,\cdot)$, in order to verify their properties. We then study the "distance" between the two functions as the sample path increases.

Proposition 1. *Let the function* $\pi=E\left[\Pi(\tau)\right]$ *be the expectation of* $\Pi(\tau)$ *and* π_n *be the sample average approximation of* $E\left[\Pi(\tau)\right]$*:* $\pi_n\triangleq\frac{1}{n}\sum_{j=1}^{n}\Pi_j(\tau)$*.*

If function $\lambda(\mathbf{s})$ *is bounded on* Σ*,* $\hat{\lambda}(\mathbf{s}_n,\cdot)$ *is* P_B*–almost surely bounded on* Σ*.*

If function $\lambda(\mathbf{s},\cdot)$ *is Lipschitz continuous on* Σ*,* $\hat{\lambda}(\mathbf{s}_n,\cdot)$ *is* P_B*–almost surely Lipschitz continuous on* Σ*.*

Proof. See Appendix. □

Theorem 2 (Convergence of $\mathcal{S}$, from Theorem 3.2 in Robinson (1996), page 519)**.** *The minimizer* $\mathbf{s}_n^*$ *epi–converges to the infinite sample path solution* $\mathbf{s}^*$ *with probability 1. The related objective function value* $\mathcal{S}_n^*$ *converges uniformly to the infinite sample path objective function value* $\mathcal{S}^*$*.*

Proof. See Appendix. □

5 Application

5.1 Problem description and DEO mathematical modeling

In this section, we apply DEO to an industrial relevant problem, the joint Task and Buffer Allocation Problem (TBAP) in a production line. This problem has two components: the Task Allocation Problem (TAP) and the Buffer Allocation Problem (BAP). All the parts need to be processed following a given sequence of tasks. The TAP aims at assigning all the tasks to the minimum number of stations while guaranteeing a target maximum average cycle time (ϑ^*). In the TAP, the task assignment is constrained by the manufacturing sequence, which means that the $(l+1)-th$ task can only be processed after the $l-th$ task. The processing times are assumed stochastic. Specifically, the average processing times of tasks (τ_l) are given as parameters, the average processing time μ_j at station j is equal to the sum of the processing times of the tasks assigned to it. Processing time t_{ij} of part i at station j follows a given stochastic distribution.

To improve the performance of stochastic production lines, buffer spaces are usually allocated between stations, and the BAP solves for the optimal allocation of buffer space in the manufacturing line. The stations and the buffers compose a flow line as shown in Figure 3. We also assume that the first station is never starved and the last station is never blocked, i.e., the arrival buffer and the departure buffer are infinite.

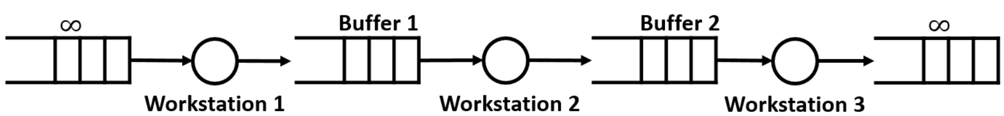


Figure 3: Example of open flow line with 3 workstations (Zhang et al. (2016))

Using the parameters and the variables reported in the following, the DEO model of the TBAP can be devised.

Parameters

C_m: station unit cost

C_b: buffer space unit cost

U_m: upper bound on the station number

U_b: upper bound on the single buffer space

T: number of tasks

n: number of simulated parts

d: warm-up length in terms of number of parts

τ_l: average processing time of task l

z_{ij}: random number for part i at station j

e_i^a: arrival time of part i

ϑ^*: target cycle time

Decision variables

$\mu_j \in R$: average processing time at station j

$m_j \in \{0,1\}$: station allocation variables. If some tasks are assigned to station j, $m_j = 1$; otherwise, $m_j = 0$

$y_{lj} \in \{0,1\}$: task assignment variables. If task l is assigned to station j, $y_{lj} = 1$; otherwise, $y_{lj} = 0$

$x_{jk} \in \{0,1\}$: buffer allocation variables. If the capacity of the j-th buffer is $k-1$, $x_{jk} = 1$; otherwise, $x_{jk} = 0$

$t_{ij} \in R$: processing time of part i at station j

$e^f_{ij} \in R$: finishing time of part i at station j

Model

The integrated mathematical programming model is as follows.

$$\min \quad C_m \sum_{j=1}^{U_m} m_j + C_b \sum_{j=1}^{U_m} \sum_{k=1}^{U_b} k x_{jk}$$

s.t.

$$\sum_{j=1}^{U_m} y_{lj} = 1 \qquad l = 1, ..., T \tag{24}$$

$$\sum_{l=1}^{T} \tau_l y_{lj} = \mu_j \qquad j = 1, ..., U_m \tag{25}$$

$$\sum_{j=1}^{U_m} j(y_{lj} - y_{l+1,j}) \leq 0 \qquad l = 1, ..., T \tag{26}$$

$$\sum_{k=1}^{U_b} x_{jk} = 1 \qquad j = 1, ..., U_m - 1 \tag{27}$$

$$m_j \geq y_{lj} \qquad j = 1, 2, ..., U_m, l = 1, ..., T \tag{28}$$

$$m_{j-1} \geq m_j \qquad j = 2, ..., U_m \tag{29}$$

$$t_{ij} = F^{-1}(\mu_j, z_{ij}) \qquad j = 1, ..., U_m, i = 1, ..., n \tag{30}$$

$$e^f_{i1} - e^a_i \geq t_{i1} \qquad i = 1, ..., n \tag{31}$$

$$e^f_{i+1,j} - e^f_{ij} \geq t_{i+1,j} \qquad j = 1, ..., U_m, i = 1, ..., n-1 \tag{32}$$

$$e^f_{i,j+1} - e^f_{ij} \geq t_{i,j+1} \qquad j = 1, ..., U_m - 1, i = 1, ..., n \tag{33}$$

$$e^f_{i+k,j} - e^f_{i,j+1} \geq t_{i+k,j} - (1 - x_{jk})M \qquad j = 1, ..., U_m - 1, k = 1, ..., U_b, i = 1, ..., n-k \tag{34}$$

$$\frac{e^f_{n,U_M} - e^f_{d,U_M}}{n-d} \leq \vartheta^* \tag{35}$$

The objective function is the minimization of the cost of stations and the cost of buffer spaces. Constraints (24) to (29) are the optimization constraints. Specifically, constraints (24) state that each task is assigned to one station. Constraints (25) define the average processing time at a station as the sum of the average processing times of the tasks assigned to the station. Constraints (26) force the $l - th$ task to be assigned before the $(l+1) - th$ task. Constraints (27) state that each buffer can be assigned a single capacity. Constraints (28) prevent a station to be allocated to the line if no task has been assigned to it. Constraints (29) impose that all the stations allocated are in the first part of the line.

Constraints (30) deal with the random generation of part processing times t_{ij}, which are a function of the average processing time and of the random numbers z_{ij}. As the value of the decision variable μ_j changes, the generated processing times t_{ij} are modified accordingly.

Constraints (31)–(34) describe the production process. Constraints (31) state that the i-th part arrives at the line at time e_i^a. Constraints (32) state that one station cannot process more than one part at the same time. Constraints (33) impose that a part cannot be processed by more than one station at the same time. Constraints (34) state that if the capacity of buffer j is equal to $k-1$ (which means $x_{jk} = 1$), part $i+k$ cannot enter station j before part i leaves station $(j+1)$. Constraint (35) is the performance constraint, bounding from above the average system cycle time (i.e., the mean cycle time cannot be larger than ϑ^*).

The number of binary variables and the number of continuous variables in the model are $U_m U_b + n \cdot U_m$ and $2n \cdot U_m + U_m$, respectively. The number of constraints containing binary variables is $n \cdot U_m U_b + n^2$, and the number of continuous constraints is $3n \cdot U_m$. Therefore, when designing long production lines or when considering long simulations, the computational complexity can be very high.

By solving this MILP model, the global optimum can be obtained using a single–replication experiment under the DEO framework. However, because of the high complexity of solving the exact model, an approximate LP model can be devised by: 1) considering the processing time at all the stations to be continuously and arbitrarily assigned, and 2) replacing the buffer allocation problem by the time buffer allocation proposed in section 4.1. In order to solve the approximate model we need to introduce the following parameters and variables:

Parameters

$\tilde{C}_m$: adjusted station unit cost (approximate model)
$\tilde{C}_b$: adjusted time buffer unit cost (approximate model)

Decision Variables

$s_{jk} \in R$: time buffer variables (approximate model).

$$\min \quad \tilde{C}_m \sum_{j=1}^{U_m} j\mu_j + \tilde{C}_b \sum_{j=1}^{U_m-1} \sum_{k=1}^{U_b} k s_{jk}$$

s.t.

$$\sum_{j=1}^{U_m} \mu_j = \sum_{l=1}^{T} \tau_l \tag{36}$$

$$e^f_{i+k,j} - e^f_{i,j+1} \geq t_{i+k,j} - s_{ik} \qquad j = 1, ..., U_m - 1, k = 1, ..., U_b, i = 1, ..., n - k \tag{37}$$

(30), (31), (32), (33), (35)

The objective function is the approximate minimization of the station number and the time buffer capacity. Constraint (36) states that the sum of the average processing times of all the stations has to be equal to the sum of the processing times of all the tasks. Constraints (37) correspond to constraints (34) under the time buffer approximation.

5.2 DEO based solution algorithm

The LP model just described is far less complex than the exact MILP model. However, numerical tests have shown that simply solving it (e.g., with ILOG CPLEX) is not really effective mainly because the task assignment is not determined, since the assignment variables y_{lj} are no longer included. For this reason, a math-heuristic algorithm is proposed to solve the problem more effectively.

Using the DEO methodology, it is possible to develop a specific algorithm to approximately solve the TBAP. The algorithm is a math-heuristic that decomposes the problem into three different problems, each one solved using a mathematical programming model. Specifically, it solves the LP approximate model, a deterministic TAP and the approximate BAP in three steps to approximately find out the optimal system configuration. In the first step, the approximate LP model is solved to find the number of stations m^*, which is determined based upon the optimal service rate μ^*_j. Specifically, m^* is equal to the smallest value of j for which $\mu^*_j > 0$. In the second step, the deterministic TAP with m^* stations is solved to find the average processing times μ_j of stations that minimize the cycle time. In the third step, given μ_j for each j, the DEO model of the BAP with time buffer approximation is solved.

The cycle time constraint of the BAP may be feasible or infeasible depending on the number of stations m^*. A tuning rule and an iterative procedure including the second and the third steps are applied to find out the minimum station number that makes the TBAP feasible. If, after the second and the third steps are performed, the BAP model is infeasible, the value of m^* will be increased by one, and the deterministic TAP and the approximate BAP models are solved again. This iterative scheme continues until the BAP model is feasible, and m^* is the minimum station number. On the contrary, if the BAP is feasible after the second and the third steps are performed, m^* is reduced by one, and the two models (TAP and BAP) are solved again, until the BAP model is infeasible, and $m^* + 1$ is the minimum station number. The complete solution procedure is shown in Figure 4.

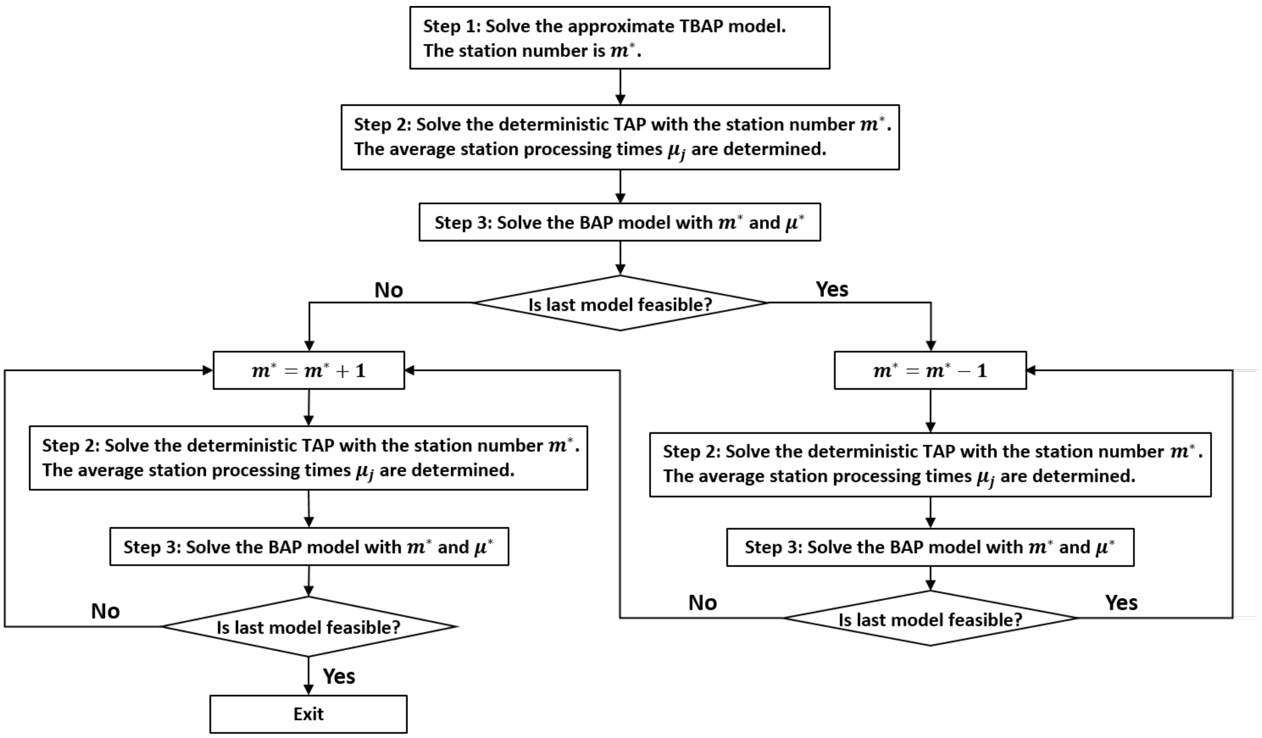


Figure 4: Math-heuristic algorithm.

In the first step, the approximate TBAP model is the one reported at the end of section 5.1, i.e.,

$$\min \quad \tilde{C}_m \sum_{j=1}^{U_m} j\mu_j + \tilde{C}_b \sum_{j=1}^{U_m-1} \sum_{k=1}^{U_b} k s_{jk}$$

$$\text{s.t.}$$

(30), (31), (32), (33), (35), (36), (37) In the second step, the model of the deterministic TAP that is solved is the following:

$$\min \quad CT$$

$$\text{s.t.}$$

$$\sum_{j=1}^{m^*} y_{lj} = 1 \qquad l = 1, ..., T \tag{38}$$

$$\sum_{l=1}^{T} \tau_l y_{lj} \leq CT \qquad j = 1, ..., m^* \tag{39}$$

$$\sum_{j=1}^{m^*} j(y_{lj} - y_{l+1,j}) \leq 0 \qquad l = 1, ..., T-1 \tag{40}$$

where CT is a positive continuous decision variable. In the third step, the used DEO model of the BAP with time buffer approximation is:

$$\min \sum_{j=1}^{m^*-1} \sum_{k=1}^{U_b} ks_{jk}$$

$$\text{s.t.}$$

$$(30), (31), (32), (33), (35), (37)$$

5.3 Numerical analysis

Ten TBAP instances with different task times have been defined, in which $T = 50$ tasks are to be assigned. The upper bound of station number U_m has been set to 15, and the upper bound of single buffer space U_b to 20. The station processing times t_{ij} follow $beta(2, 2)$ distribution in the interval $(0,\ 2\mu_j)$; specifically, processing time in equation (30) is generated using the formula $t_{i,j} = 2\mu_j \cdot z_{i,j}$, where $z_{i,j}$ is randomly generated from $beta(2, 2)$ on the interval $[0, 1]$. Each instance has been replicated ten times, i.e., ten sample paths have been used for the random numbers z_{ij}. The station unit cost both in the exact and in the approximate model, i.e., C_m and $\tilde{C}_m$, is equal to 1000, while the buffer space unit cost C_b and $\tilde{C}_b$ is equal to 1. The number of parts in simulation n is equal to 5000.

The solution found by our procedure has been verified by running independent experiments with Arena simulation models, where the replication length is 100 000, and the number of parts of warm-up is equal to 500, identified with the Welch's approach.

Table 1 reports the results for the ten instances, averaged on the ten replications. The first column refers to the TBAP instance (the task times are different in each instance). The second, third and fourth columns refer to the station number, average processing time of stations and stage buffer spaces, respectively. The last column reports the average cycle time found simulating the solution with Arena. From Table 1, it can be noticed that the cycle time constraints $\vartheta^* \leq 1000$ is always satisfied (the values in the last column are all smaller than 1000).

Table 1: Solution of the TBAPs provided by the math-heuristic on average.

Instance	Station number	Station average processing time	Buffer Allocation	Cycle time
1	9	824 677 815 854 860 817 839 792 798	2 2 3 3 3 3 2 2	957±3
2	7	743 587 816 799 892 853 895	2 2 2 3 3 3	971±3
3	9	777 785 945 839 780 648 935 920 862	3 3 4 3 3 3 5 4	982±3
4	8	770 940 932 830 793 837 786 815	3 5 4 3 3 3 2	981±3
5	8	787 888 790 898 865 838 804 908	3 3 3 3 3 3 3	975±3
6	7	801 803 786 867 851 834 885	2 2 3 3 3 3	960±3
7	8	700 885 853 915 890 905 855 835	2 3 4 4 4 4 3	980±3
8	8	825 683 885 748 854 910 915 774	2 2 3 3 4 4 3	973±3
9	10	715 803 804 784 845 889 899 918 660 794	2 2 3 3 3 4 4 3 2	975±3
10	8	896 795 881 816 808 757 818 870	3 3 3 3 2 2 3	961±3

When different sample paths are used, the station number and the task allocation is always the same for each instance, but the buffer allocation can slightly vary. This issue is shown in Table 2, which reports in detail the ten replications (i.e, the ten sample paths) related to instance 4.

Table 2: Solution of the instance 4 provided by the math-heuristic in detail for each of the 10 sample paths.

Station number	Total buffer space	Station average processing time	Buffer Allocation
8	23	770 940 932 830 793 837 786 815	3 5 4 3 3 3 2
8	24	770 940 932 830 793 837 786 815	3 5 4 3 3 3 3
8	23	770 940 932 830 793 837 786 815	3 5 4 3 3 3 2
8	24	770 940 932 830 793 837 786 815	3 5 4 3 3 3 3
8	22	770 940 932 830 793 837 786 815	3 4 4 3 3 3 2
8	23	770 940 932 830 793 837 786 815	3 4 4 3 3 3 3
8	23	770 940 932 830 793 837 786 815	3 5 4 3 3 3 2
8	22	770 940 932 830 793 837 786 815	3 4 4 3 3 3 2
8	24	770 940 932 830 793 837 786 815	3 5 4 3 3 3 3
8	23	770 940 932 830 793 837 786 815	3 5 4 3 3 3 2

The same TBAP is also solved using OptQuest in Arena by running 10000 iterations, where each iteration executes one replication. Each simulation has a length of 5000 parts and a warm–up of 500 parts. Figure 5 shows the total costs of the solutions (i.e., the assembly lines) found by the math-heuristic and by OptQuest.

The results of the experiments show that the proposed math-heuristic algorithm based on DEO is both efficient and accurate. In fact, as reported in Figure 5, the cost of the solution provided by the heuristic is always lower than that of the solution provided by OptQuest. Indeed, in 8 cases out of 10, OptQuest finds a solution that has one or two stations more than the solution achieved by using the math-heuristic. Furthermore, for instance 9, Optquest cannot find a feasible solution. Finally, the average computation time of the math-heuristic algorithm is around 10 minutes for instance, while the average instance time of OptQuest is around 20 hours.

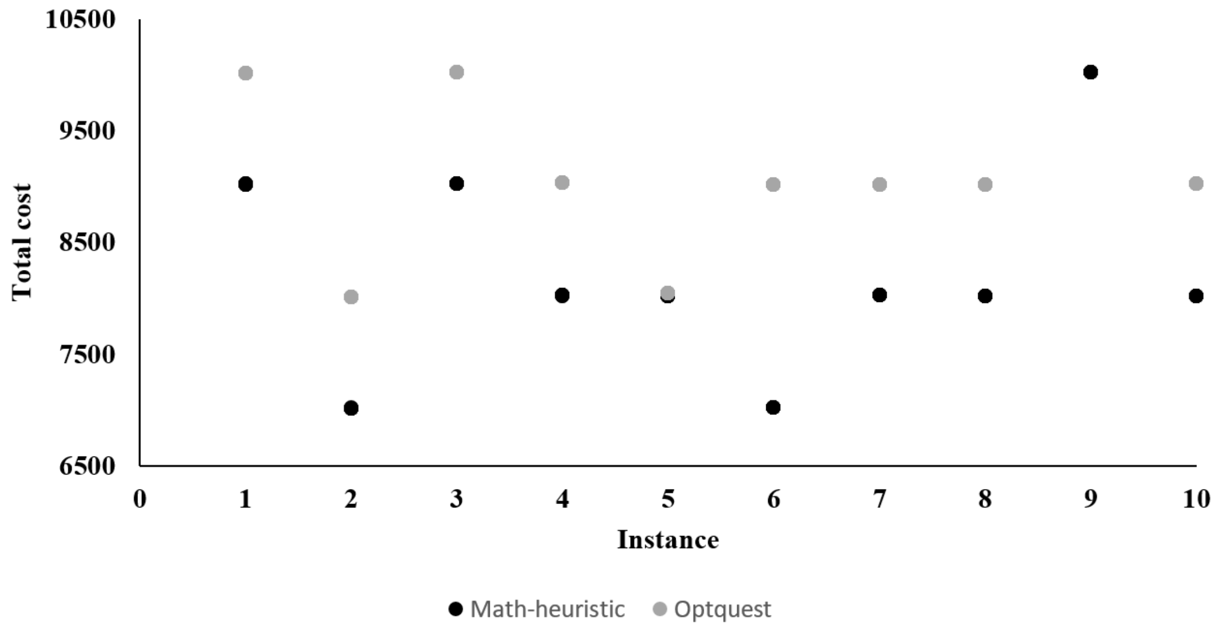


Figure 5: Total cost comparison between Optquest and the proposed math-heuristic.

6 Conclusion

In this paper, we have presented the Discrete Event Optimization (DEO) methodology for the integrated modeling and solution of simulation–optimization problems. We have shown the general formulation for DEO models for the integrated simulation–optimization of queueing systems. Asymptotic properties of DEO in the case of continuous variable models have been provided under the framework of sample path optimization.

DEO has been applied to model the TBAP and develop a solution algorithm to show its flexibility and applicability to a complex problem.

The ability to derive white box models that integrate simulation and optimization for manufacturing systems represents a fundamental step towards the boost of optimization techniques based on the theory of sample path optimization and mathematical programming. Future research can exploit the structure of DEO models to build analytical representations of simulation–optimization problems, which are generally developed using black box simulation models. Such analytical representations can be used to introduce new kinds of approximations and algorithms to make more efficient the search for optimal solutions.

The authors are currently working on the investigation of fast algorithms to efficiently solve large classes of queueing simulation–optimization problems, relying on the network properties of the dual formulation of the DEO model. The finite time performance is also under study by exploiting again the mathematical programming setting and interpreting the regret as the pricing function in a column generation approach.

Acknowledgement

This work is funded by the National Nature Science Foundation of China (61473188 "A Simulation Optimization Framework for Design and Control of Complex Production System"). Andrea Matta was a full professor at Shanghai Jiao Tong University, and he is the principal investigator of the project.

References

Alfieri, A., A. Matta. 2012a. Mathematical programming representation of pull controlled single-product serial manufacturing systems. *Journal of Intelligent Manufacturing* **23**(1) 23–35. doi: 10.1007/s10845-009-0371-x. URL `http://dx.doi.org/10.1007/s10845-009-0371-x`.

Alfieri, Arianna, Andrea Matta. 2012b. Mathematical programming formulations for approximate simulation of multistage production systems. *European Journal of Operational Research* **219**(3) 773 – 783. doi:http://dx.doi.org/10.1016/j.ejor.2011.12.044. URL `http://www.sciencedirect.com/science/article/pii/S0377221712000306`.

Andradóttir, S. 2005. *An overview of simulation optimization via random search*, chap. 21 - Handbooks in Operations Research and Management Science: Simulation. Elsevier.

Askin, Ronald G, Charles R Standridge. 1993. *Modeling and analysis of manufacturing systems*. John Wiley & Sons Inc.

Attouch, H. 1984. *Variational Convergence of Functions and Operators*. Pitman, London.

Attouch, H. 1991. Quantitative stability of variational systems i: The epigraphical distance. *Transactions American Society* **328** 695–729.

Baccelli, F., G. Cohen, G.J. Olsder, J.P. Quadrat. 1992. *Synchronization and linearity : an algebra for discrete event systems*. Wiley.

Beer, G. 1993. *Topologies on Closed and Closed Convex Sets*. Kluwer Academic Publishers, Dordrecht.

Boesel, J. N., L. Barry, S.H. Kim. 2003. Using ranking and selection to "clean up" after simulation optimization. *Operations Research* **51** 814–825. doi:10.1287/opre.51.5.814.16751.

Buss, Arnold H., Paul J. Sanchez. 2002. Modeling very large scale systems: Building complex models with legos (listener event graph objects). *Proceedings of the 34th Conference on Winter Simulation: Exploring New Frontiers*. WSC '02, Winter Simulation Conference, 732–737. URL `http://dl.acm.org/citation.cfm?id=1030453.1030555`.

Buzacott, J.A., J.G. Shantikumar. 1993. *Stochastic Models of Manufacturing Systems*. Prentice–Hall.

Cao, Xi-Ren, Junyu Zhang. 2008. Event-based optimization of markov systems. *Automatic Control, IEEE Transactions on* **53**(4) 1076–1082. doi:10.1109/TAC.2008.919557.

Cassandras, Christos G, Stephane Lafortune. 2008. *Introduction to discrete event systems*. Springer.

Chan, W. K. 2005. Mathematical programming representations of discrete-event system dynamics. Ph.D. thesis, IEOR University of California, Berkeley.

Chan, W. K., L. W. Schruben. 2006. Response gradient estimation using mathematical programming models of discrete–event system sample paths. L. F. Perrone, F. P. Wieland, J. Liu, B. G. Lawson, D. M. Nicol, R. M. Fujimoto, eds., *Proceedings of the 2006 Winter Simulation Conference*. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc., 272–278.

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

Fu, M., F.G. Glover, J. April. 2005. Simulation optimization: a review, new developments, and applications. *Proceedings of the 2005 Winter Simulation Conference*.

Fu, Michael, et al. 2015. *Handbook of simulation optimization*, vol. 216. Springer.

Haskell, William B, J George Shanthikumar, Z Max Shen. 2012. Aspects of increasing concave stochastic order constrained optimization. Tech. rep., Technical report.

Healy, K., L. W. Schruben. 1991. Retrospective simulation response optimization. *Proceedings of the 23rd conference on Winter simulation*.

Helber, S., K. Schimmelpfeng, R. Stolletz. 2011a. Setting inventory levels of conwip flow lines via linear programming. *BuR Business Research Journal* **4**(1) 98–115.

Helber, S., K. Schimmelpfeng, R. Stolletz, S. Lagershausen. 2011b. Using linear programming to analyze and optimize stochastic flow lines. *Annals of Operations Research* **182**(1) 193–211. doi:10.1007/s10479-010-0692-3.

Helber, Stefan, Katja Schimmelpfeng, Raik Stolletz, Svenja Lagershausen. 2011c. Using linear programming to analyze and optimize stochastic flow lines. *Annals of Operations Research* **182**(1) 193–211.

Hong, L. J., B. L. Nelson. 2006. Discrete optimization via simulation using compass. *Operations Research* **54**(1) 115–129.

Jin, J., B. Schmeiser. 2003. Simulation-based retrospective optimization of stochastic systems: a family of algorithms. *Proceedings of the 2003 Winter Simulation Conference*.

Kall, Peter. 1986. Approximation to optimization problems: An elementary review. *Mathematics of Operations Research* **11**(1) pp. 9–18.

King, A.J., R. J. B. Wets. 1991. Epi-consistency of convex stochastic programs. *Stochastics and Stochastics Reports* **34** 259–260.

Kleijnen, J. P. C. 2008. *Design and analysis of simulation experiments*, *International Series in Operations Research & Management Science*, vol. 111. Springer.

Kushner, H.J., G. Yin. 1997. *Stochastic approximation algorithms and applications*. Springer.

Lachou, P. 1998. *Evaluation of the epiconvergence*. Institut fr Mathematik der Humboldt-Universitt zu Berlin.

Law, A.M. 2007. *Simulation Modeling and Analysis*. 4th ed. McGraw-Hill.

Liu, Ying, Hui Zhang, Chunping Li, Roger Jianxin Jiao. 2012. Workflow simulation for operational decision support using event graph through process mining. *Decision Support Systems* **52**(3) 685 – 697. doi:http://dx.doi.org/10.1016/j.dss.2011.11.003. URL `http://www.sciencedirect.com/science/article/pii/S0167923611002077`.

Matta, A. 2008. Simulation optimization with mathematical programming representation of discrete event systems. *Proceedings of the 2008 Winter Simulation Conference*.

Matta, Andrea, Giulia Pedrielli, Arianna Alfieri. 2014. Event relationship graph lite: Event based modeling for simulation-optimization of control policies in discrete event systems. *Winter Simulation Conference (WSC), 2014*. IEEE.

Myers, R.H., D.C. Montgomery, C.M. Anderson-Cook. 2009. *Response surface methodology: process and product optimization using designed experiments*. Wiley Series in Probability and Statistics, Wiley.

Pedrielli, Giulia, Arianna Alfieri, Andrea Matta. 2015a. Integrated simulation–optimisation of pull control systems. *International Journal of Production Research* **53**(14) 4317–4336.

Pedrielli, Giulia, Andrea Matta, Arianna Alfieri. 2015b. Discrete event optimization: Single-run integrated simulation-optimization using mathematical programming. *Winter Simulation Conference (WSC), 2015*. IEEE, 3557–3568.

Robinson, M., S. 1987. Local epicontinuity and local optimization. *Mathematical Programming* **37** 208–222.

Robinson, S.M. 1996. Analysis of sample–path optimization. *Mathematics of Operations Research* **21** 513–528.

Rockafellar, R. T., R. J.-B. Wets. 1998. *Variational Analysis*. Springer, New York.

Salinetti, G., R. J.B. Wets. 1979. On the convergence of sequence of convex sets in finite dimensions. *SIAM Review* **21** 18–33.

Savage, E. L., L. W. Schruben, E. Ycesan. 2005. On the generality of event-graph models. *INFORMS Journal on Computing* **17**(1) 3–9. doi:10.1287/ijoc.1030.0053.

Schruben, L. W. 1983. Simulation modeling with event graphs. *Communications of the ACM* **26**(11) 957 – 963.

Schruben, Lee. 2010. Simulation modeling for analysis. *ACM Transactions on Modeling and Computer Simulation (TOMACS)* **20**(1) 1–22.

Schwarz, J.A., R. Stolletz. 2013. A sampling approach for the analysis of time-dependent stochastic flow lines. *Proceedings of the 2013 Conference on Stochastic Models of Manufacturing and Service Operations*. Kloster Seeon, Germany.

Shaked, M., J.G. Shanthikumar. 2007. *Stochastic orders*. Springer series in statistics, Springer. URL `http://books.google.com/books?id=rPiToBK2rwwC`.

Shaked, Moshe, J. George Shanthikumar. 1988. Stochastic convexity and its applications. *Advances in Applied Probability* **20**(2) pp. 427–446.

Shapiro, Alexander. 2003. Monte carlo sampling methods. *Handbooks in operations research and management science* **10** 353–425.

Spall, J. C. 2003. *Introduction to Stochastic Search and Optimization: Estimation, Simulation, and Control*. Wiley.

Stolletz, R., S. Weiss. 2013. Buffer allocation using exact linear programming formulations and sampling approaches. *Preprints of the 2013 IFAC Conference on Manufacturing Modelling, Management, and Control*. Saint Petersburg, Russia.

Tan, Bari. 2015. Mathematical programming representations of the dynamics of continuous-flow production systems. *IIE Transactions* **47**(2) 173–189. doi:10.1080/0740817X.2014.892232. URL `http://dx.doi.org/10.1080/0740817X.2014.892232`.

Topsoe, Flemming. 2006. *Topology and measure*, vol. 133. Springer.

Weiss, S, Matta A, Stolletz R. 2017. Optimization of buffer allocations in flow lines with limited supply. *IISE Transactions* **0**(0) 1–12. doi:10.1080/24725854.2017.1328751. URL `http://dx.doi.org/10.1080/24725854.2017.1328751`.

Weiss, S., R. Stolletz. 2013. First results on: A benders decomposition approach for the optimization of flow lines with stochastic processing times. *Proceedings of the 2013 Conference on Stochastic Models of Manufacturing and Service Operations*. Kloster Seeon, Germany.

Weiss, Sophie, Raik Stolletz. 2015. Buffer allocation in stochastic flow lines via sample-based optimization with initial bounds. *OR Spectrum* 1–34doi:10.1007/s00291-015-0393-z. URL `http://dx.doi.org/10.1007/s00291-015-0393-z`.

Wets, R. 1991. Constrained estimation consistency and asymptotics. *Applied Stochastic Models Data Analysis* **7** 17–32.

Yao, D., J. George Shanthikumar. 1991. Strong stochastic convexity: Closure properties and applications. *Journal of Applied Probability* **28**(1) pp. 131–145.

Zabinsky, Zelda B. 2009. Random search algorithms. *Wiley Encyclopedia of Operations Research and Management Science* .

Zhang, Mengyi, Andrea Matta, Giulia Pedrielli. 2016. Discrete event optimization: workstation and buffer allocation problem in manufacturing flow lines. *Winter Simulation Conference (WSC), 2016*. IEEE, 2879–2890.

7 Appendix

7.1 Proof of Theorem 1

Proof. Considering constraints of the type (12) and (13), we can derive the following inequalities:

$$
\begin{aligned}
e^{\xi}_{i,j} &\geq e^{\xi'}_{i'j'} \\
\tilde{e}^{\xi}_{ij} &\geq \tilde{e}^{\xi'}_{i'j'} - s^{\xi\xi'}_{ij,i'j'}
\end{aligned} \tag{41}
$$

It is apparent that if $e^{\xi'}_{i'j'} \geq \tilde{e}^{\xi'}_{i'j'} - s^{\xi\xi'}_{ij,i'j'}$, then $e^{\xi}_{ij} \geq \tilde{e}^{\xi}_{ij}$, i.e., the exact system with $\kappa^{\xi\xi'}_{ij,i'j'} = 1$ is less efficient than the approximate counterpart with $s^{\xi\xi'}_{ij,i'j'}$. Therefore, $\kappa^{\xi\xi'}_{ij,i'j'}$ is a lowerbound. The upperbound is derived in a similar way, verifying if $e^{\xi'}_{i'j'} \leq \tilde{e}^{\xi'}_{i'j'} - s^{\xi\xi'}_{ij,i'j'}$. □

7.2 Proof of Property 1: Second Order Properties for $\mathcal{S}_n$

Proof. To prove the increasing property, two vectors, $\mathbf{b}_{(1)}$ and $\mathbf{b}_{(2)}$, are constructed in a way such that $\mathbf{b}_{(1)} \leq \mathbf{b}_{(2)}$ in each component. Then, at the optimum, $\mathcal{S}_n(\mathbf{b})$ satisfies:

$$
\mathcal{S}^*_n(\mathbf{b}_{(1)}) = \mathbf{b}'_{(1)}\mathbf{u}^{(1)*} - \mu^*\nu^{(1)*} \leq \mathbf{b}'_{(2)}\mathbf{u}^{(1)*} - \mu^*\nu^{(1)*} \leq \mathbf{b}'_{(2)}\mathbf{u}^{(2)*} - \mu^*\nu^{(2)*} = \mathcal{S}^*_n(\mathbf{b}_{(2)}),
$$

where $\left(\mathbf{u}^{(1)*}, \nu^{(1)*}\right)$ and $\left(\mathbf{u}^{(2)*}, \nu^{(2)*}\right)$ are optimal solutions of the dual models having as right hand side $\mathbf{b}_{(1)}$ and $\mathbf{b}_{(2)}$, respectively. Furthermore, $\mathbf{u}^{(\cdot)*}$ is the concatenation of $\mathbf{u}^{(\cdot)*}_{\mathcal{D}}$ and $\mathbf{u}^{(\cdot)*}_{\mathcal{P}}$, and $\mathbf{b}_{(\cdot)}$ is the concatenation of $\mathbf{b}^1_{(\cdot)}$ and $\mathbf{b}^2_{(\cdot)}$.
The first inequality is justified by the fact that the solution $\left(\mathbf{u}^{(1)*}, \nu^{(1)*}\right)$ is dual feasible under the increased right hand side, and the second inequality derives from standard LP duality theory.

To prove that the objective function is convex, take $0 \leq \beta \leq 1$ and let $\left(\mathbf{u}^{(1)*}, \nu^{(1)*}\right)$, $\left(\mathbf{u}^{(2)*}, \nu^{(2)*}\right)$ and $\left(\mathbf{u}^{(3)*}, \nu^{(3)*}\right)$ be the optimal solutions of the dual of the approximate optimization model when the right hand side vectors are $\beta\mathbf{b}_{(1)} + (1-\beta)\mathbf{b}_{(2)}$, $\mathbf{b}_{(1)}$ and $\mathbf{b}_{(2)}$, respectively. Then we have that:

$$
\begin{aligned}
\mathcal{S}^*_n(\beta\mathbf{b}_{(1)} + (1-\beta)\mathbf{b}_{(2)}) &= (\beta\mathbf{b}'_{(1)} + (1-\beta)\mathbf{b}'_{(2)})\mathbf{u}^{(1)*} - \mu^*\nu^{(1)*} \\
&= \beta\mathbf{b}'_{(1)}\mathbf{u}^{(1)*} + (1-\beta)\mathbf{b}'_{(2)}\mathbf{u}^{(1)*} - \mu^*\nu^{(1)*} \\
&\leq \beta\left(\mathbf{b}'_{(1)}\mathbf{u}^{(2)*} - \mu^*\nu^{(2)*}\right) + (1-\beta)\left(\mathbf{b}'_{(2)}\mathbf{u}^{(3)*} - \mu^*\nu^{(3)*}\right) \\
&= \beta\mathcal{S}^*_n(\mathbf{b}_{(1)}) + (1-\beta)\mathcal{S}^*_n(\mathbf{b}_{(2)}).
\end{aligned}
$$

The first equality comes from the fundamental relationship between the primal and the dual at the optimum. The second is a simple algebraic manipulation, whereas the inequality follows from the definition of $\left(\mathbf{u}^{(1)*}, \nu^{(1)*}\right)$, $\left(\mathbf{u}^{(2)*}, \nu^{(2)*}\right)$ and $\left(\mathbf{u}^{(3)*}, \nu^{(3)*}\right)$. The last step comes from the fundamental relationship between the primal and the dual at the optimum.
Therefore, $\mathcal{S}_n(\mathbf{b})$ is an increasing and convex function of τ.

To prove *SICX(sp)*, we use the monotonicity and convexity just proved and we consider the

SICX property owned by the right hand side. This part of the proof is based on Shaked and Shanthikumar (1988) (Proposition 3.2). In particular, let τ_i (with $i = 1, 2, 3, 4$) be four scalars such that:

$$\tau_1 \leq \tau_2 \leq \tau_3 \leq \tau_4$$
$$\tau_2 + \tau_3 = \tau_1 + \tau_4.$$

We used four values since this is the minimum collection size to define all the possible orderings. Since $\{\mathbf{b}(\tau)\} \in$ *SICX(sp)*, following the approach in Shaked and Shanthikumar (2007), we can prove the desired property generating four random variables $\hat{\mathbf{b}}^{(i)} =_{st} \mathbf{b}(\tau_i)$ such that:

$$\hat{\mathbf{b}}_{(2)} + \hat{\mathbf{b}}_{(3)} \leq \hat{\mathbf{b}}_{(4)} + \hat{\mathbf{b}}_{(1)} \qquad \textit{almost surely},$$
$$\left[\hat{\mathbf{b}}_{(1)}, \hat{\mathbf{b}}_{(2)}, \hat{\mathbf{b}}_{(3)}\right] \leq \hat{\mathbf{b}}_{(4)} \qquad \textit{almost surely}.$$

where $\left[\hat{\mathbf{b}}_{(1)}, \hat{\mathbf{b}}_{(2)}, \hat{\mathbf{b}}_{(3)}\right] \leq \hat{\mathbf{b}}_{(4)}$ states that $\hat{\mathbf{b}}_{(1)}, \hat{\mathbf{b}}_{(2)}, \hat{\mathbf{b}}_{(3)}$ are all smaller than $\hat{\mathbf{b}}_{(4)}$. The monotonicity and convexity of $\mathcal{S}(\mathbf{b})$ imply that

$$\begin{aligned}\mathcal{S}_n^*(\hat{\mathbf{b}}_{(2)}) - \mathcal{S}_n^*(\hat{\mathbf{b}}_{(1)}) &\leq \mathcal{S}_n^*(\hat{\mathbf{b}}_{(4)}) - \mathcal{S}_n^*\left(\hat{\mathbf{b}}_{(4)} + \hat{\mathbf{b}}_{(1)} - \hat{\mathbf{b}}_{(2)}\right) \\ &\leq \mathcal{S}_n^*(\hat{\mathbf{b}}_{(4)}) - \mathcal{S}_n^*(\hat{\mathbf{b}}_{(3)});\end{aligned}$$

therefore, there exist four random variables $\hat{S}_i =_{st} \mathcal{S}(B)$ (where $=_{st}$ stands for *equivalent in a stochastic sense* and B is the collection of the random variables) on a common probability space, such that:

$$\hat{S}^*_{n,2} + \hat{S}^*_{n,3} \leq \hat{S}^*_{n,1} + \hat{S}^*_{n,4} \quad \textit{almost surely}\,.$$
$$\left[\hat{S}^*_{n,1}, \hat{S}^*_{n,2}, \hat{S}^*_{n,3}\right] \leq \hat{S}^*_{n,4}, \quad \textit{almost surely},$$

This proves that $\mathcal{S}_n\left[(\mathbf{b}_i\,(\tau_i))_{i=1}^m\right]$ is *SICX(sp)* in $(\tau_i)_{i=1}^m$ (Yao and Shanthikumar, 1991). □

7.3 Proof of Corollary 1

Proof. The average performance estimate $\frac{1}{n}\sum_{i=1}^{n}\frac{g(e_{iJ})}{i}$ is obtained from $\chi^*(\mathbf{s}, \mathbf{b}) = \sum_{i=1}^n \sum_{j=1}^J g(e_{ij})$ by convex operations.

Since χ^* is *SICX(sp)* in the service times and the *SICX(sp)* property is closed with respect to monotonic convex operations (Yao and Shanthikumar, 1991), then the average performance is *SICX(sp)* with respect to the service times.

Since χ^* is *SDCX(sp)* in the continuous variables $\mathbf{s}$ and the *SDCX(sp)* property is closed with respect to monotonic convex operations (Yao and Shanthikumar, 1991), then the average performance is *SDCX(sp)* with respect to the continuous variables $\mathbf{s}$. □

7.4 Proof of Lemma 1

Proof. Corollary 1 proves that $\Lambda(\mathbf{s}, B)$ is convex in the parameters τ of distribution $\mathcal{V}^B$ of B, hence it is always possible to define function Π. In addition, if the system is stationary and $\mu^* \geq \mu_{min}$, then $\Lambda(\mathbf{s}, B) < \infty$ *with probability 1*, hence function Π is integrable. Corollary 1 also guarantees that, if the difference $||\mathbf{s}_1 - \mathbf{s}_2||$ is finite, the difference $||\Lambda(\mathbf{s}_1, B) - \Lambda(\mathbf{s}_2, B)||$ is also finite.
For convexity, finiteness and compactness of the set Σ (section 4.3.1; notice that we dropped the subscript n since we are referring to the *infinite sample path* set), the moment generating functions of $\Pi(\tau)$ is finite in a neighborhood of 0 (Beer, 1993; Salinetti and Wets, 1979). □

7.5 Proof of Lemma 2

Proof. Let n^* be a constant whose value is a function of $\mathbf{s}$, and let ε be an arbitrarily small value such that:

$$|\hat{\lambda}(\mathbf{s}, \cdot) - \lambda(\mathbf{s}, \cdot)| < \varepsilon/3,$$

for all the sample paths having a size $n \geq n^*(\mathbf{s})$. Functions $\left\{\hat{\lambda}(\mathbf{s}_n, \cdot)\right\}$ are decreasing convex in $\mathbf{s}_n$ and increasing convex in $\mathbf{b}$ (Corollary 1), then Lipschitz continuity holds P_B–almost surely. We can define two Lipschitz constants, δ_1 and δ_2, which depend on B, but do not depend on $\mathbf{s}$, such that, taking a configuration $\mathbf{s}'$,

$$||\mathbf{s} - \mathbf{s}'||_\Sigma < \delta_1 \text{ and } ||\mathbf{s} - \mathbf{s}'||_\Sigma < \delta_2,$$

implies, respectively

$$|\hat{\lambda}(\mathbf{s}, \cdot) - \hat{\lambda}(\mathbf{s}', \cdot)| < \varepsilon/3. \text{ and } |\lambda(\mathbf{s}, \cdot) - \lambda(\mathbf{s}', \cdot)| < \varepsilon/3.$$

Let $\delta^* = \min\{\delta_1, \delta_2\}$ and $\mathcal{D}_\delta(\mathbf{s})$ be a set of measure δ. Given the compactness of Σ, we can take a finite subcover of all $\mathcal{D}$-sets defined as

$$\{\mathcal{D}_\delta(\mathbf{s}_d) : d = 1, \ldots, D\}.$$

Let $n^* \geq \max\{n^*(\mathbf{s}_d) : d = 1, \ldots, D\}$ and, for every $\mathbf{s} \in \Sigma$, let $\mathbf{s}_{d^*}$ be such that $\mathbf{s} \in \mathcal{D}_{\delta^*}(\mathbf{s}_{d^*})$. It follows that

$$\begin{aligned}
|\hat{\lambda}(\mathbf{s}, \cdot) - \lambda(\mathbf{s}, \cdot)| &\leq \\
\leq |\hat{\lambda}(\mathbf{s}, \cdot) - \hat{\lambda}(\mathbf{s}_{d^*}, \cdot) + \hat{\lambda}(\mathbf{s}_{d^*}, \cdot) - \lambda(\mathbf{s}_{d^*}, \cdot) + \lambda(\mathbf{s}_{d^*}, \cdot) - \lambda(\mathbf{s}, \cdot)| &\leq \\
\leq |\hat{\lambda}(\mathbf{s}, \cdot) - \hat{\lambda}(\mathbf{s}_{d^*}, \cdot)| + |\hat{\lambda}(\mathbf{s}_{d^*}, \cdot) - \lambda(\mathbf{s}_{d^*}, \cdot)| + |\lambda(\mathbf{s}_{d^*}, \cdot) - \lambda(\mathbf{s}, \cdot)| &< \\
< \varepsilon/3 + \varepsilon/3 + \varepsilon/3 &= \varepsilon
\end{aligned}$$

The uniform convergence is then proved as a direct consequence:

$$|\hat{\lambda}_n(\mathbf{s}, \cdot) - \lambda(\mathbf{s}, \cdot)| \leq \sup\left\{|\hat{\lambda}_n(\mathbf{s}, \cdot) - \lambda(\mathbf{s}, \cdot)|\right\} \to 0.$$

□

7.6 Proof of Lemma 3 (Assumption A)

Proof. Function $\mathcal{S}_n$ is lower semi continuous *iff* the related epigraph, $E_{\mathcal{S}_n}$, is closed (Attouch, 1984). The epigraph of the function $\mathcal{S}_n$ represents the set of solutions having a value of the objective function smaller than some predefined $\phi > \mathcal{S}_n^*$, i.e., $E_{\mathcal{S}_n} = \{(\mathbf{s}_n, \phi) : \mathcal{S}_n(\mathbf{s}_n) \leq \phi\}$. In particular, let $\mathbf{s}_n$ be a feasible solution and ϕ be a real positive value. Function $\mathcal{S}_n : \Sigma_n \to \mathbb{R}_+$ is linear in $\mathbf{s}_n$ and it is defined over the compact set Σ_n. As a result, there exists an arbitrarily small $\delta(\phi) \in \mathbb{R}$ such that the neighborhood of a (feasible) solution, $\mathbf{s}_n + \delta$, is outside the epigraph $E_{\mathcal{S}_n}$; thus $E_{\mathcal{S}_n}$ is a closed set. This proves that $\mathcal{S}_n$ is lower semi continuous.

Function $\mathcal{S}_n \to_{epi} \mathcal{S}$ if, in addition to lower–semi continuity, it uniformly converges to the infinite sample path function $\mathcal{S}$ on compact subsets of the function domain Σ. Since the random variables B are such that $P(B \geq \infty) = 0$, then the objective function is finite $\mathcal{S} \leq \infty$ a.s., hence the function is proper. In addition, the objective function is linear in $\mathbf{s}$, convex in the service times (Property 1) and it is defined over a compact set Σ_n. Since, from Lemma 2, we know that $\Sigma_n \to \Sigma$ uniformly, we have uniform convergence. □

7.7 Proof of Lemma 4 (Assumption B)

Proof. The problem in (16) always admits a feasible solution for construction. Let μ_{min} be the best performance that can be reached by the system in a steady state. If the system is stationary, a finite μ_{min} exists. As a result, the problem solution set is non–empty. Under Assumption 1, as the sample path size goes to ∞, the optimal solution is characterized by $\varepsilon^* = 0$. □

7.8 Proof of Proposition 1

Proof. Let $\mathbf{s}_0 \in \Sigma$ be a continuous solution. Lemma 1 leads to the following chain of inequalities for the infinite sample path problem:

$$\begin{aligned} |\lambda(\mathbf{s}, \mathbf{B})| &\leq |\lambda(\mathbf{s}_0, \mathbf{B})| + \Pi(\tau)||\mathbf{s} - \mathbf{s}_0|| \\ &\leq |\lambda(\mathbf{s}_0, \mathbf{B})| + \Pi(\tau) \max_{\mathbf{s}, \mathbf{s}_0 \in \Sigma} ||\mathbf{s} - \mathbf{s}_0||. \end{aligned} \tag{42}$$

Taking the expectations, we obtain the following inequalities

$$\begin{aligned} E[\lambda(\mathbf{s}, B)] \leq &\quad E[|\lambda(\mathbf{s}_0, B)| + \Pi(\tau)||\mathbf{s} - \mathbf{s}_0||] \\ 0 \leq &\quad |\lambda(\mathbf{s}_0, B)| + E[\Pi] \max_{\mathbf{s}_1, \mathbf{s}_2 \in \Sigma} ||\mathbf{s}_1 - \mathbf{s}_2|| \quad = \\ = &\quad |\lambda(\mathbf{s}_0, B)| + \pi \max_{\mathbf{s}_1, \mathbf{s}_2 \in \Sigma} ||\mathbf{s}_1 - \mathbf{s}_2|| \quad < \infty \end{aligned} \tag{43}$$

The inequalities $|\lambda(s_0, B)| < \infty$ and $\Pi < \infty$ hold because of Lemma 1, whereas $\max_{\mathbf{s}_1, \mathbf{s}_2 \in \Sigma} ||\mathbf{s}_1 - \mathbf{s}_2||$ (the *deviation* of the set Σ) is finite because the set is compact. This proves that equation (43) holds, i.e., $|\lambda(s_0, B)|$ and π are both finite.

The chain of equalities $E[\lambda(\mathbf{s}, B)] = E[\varepsilon] = 0$ is guaranteed by any target performance satisfying $\mu^* \geq \mu_{min}$. In fact, from Lemma 4, as $n \to \infty$, we are guaranteed that the solution set is not empty and $\varepsilon^* = 0$ exists.

The same result can be proved for function $\hat{\lambda}_n(\mathbf{s}, \mathbf{b})$. We can rewrite the chain of inequalities for the finite sample path problem as follows:

$$\begin{aligned} E\left[\hat{\lambda}(\mathbf{s}_n, \mathbf{b})\right] \leq & \quad E\left[|\hat{\lambda}(\mathbf{s}_{n,0}, \mathbf{b})| + \Pi(\tau)||\mathbf{s}_n - \mathbf{s}_{n,0}||\right] \\ \alpha \leq & \quad E\left[|\hat{\lambda}(\mathbf{s}_0, \mathbf{b})|\right] + E\left[\hat{\Pi}\right] \max_{\mathbf{s}_{n,1}, \mathbf{s}_{n,2} \in \Sigma} ||\mathbf{s}_{n,1} - \mathbf{s}_{n,2}|| \quad = \\ = & \quad E\left[|\hat{\lambda}(\mathbf{s}_{n,0}, \mathbf{b})|\right] + \hat{\pi} \max_{\mathbf{s}_{n,1}, \mathbf{s}_{n,2} \in \Sigma} ||\mathbf{s}_{n,1} - \mathbf{s}_{n,2}|| \quad < \infty \end{aligned}$$

Note that, in the second inequality, we have α instead of 0. In fact, in the finite sample path case, we cannot guarantee that $\hat{\varepsilon}^* = 0$, i.e., in general, $E\left[\hat{\lambda}(\mathbf{s}, \mathbf{b})\right] = E[\hat{\varepsilon}] = 0$ does not hold.

Lipschitz continuity holds as a consequence of Corollary 1. In fact, function $\lambda(\mathbf{s}, B)$ is increasing convex in the service times, realization of the random variables in B, and it is decreasing convex in the multidimensional array $\mathbf{s}$. □

7.9 Proof of Theorem 2

Proof. From Lemma 4, the optimal solution $\mathbf{s}^*$ to the infinite–sample path optimization problem exists and it is finite. The ε-problem is developed in a way that a minimum to the sample path problem always exists, i.e., for each n, the sample–path solution $\mathbf{s}_n$ is such that $s_n \in \mathbb{S}_n^*$. As $n \to \infty$ this solution converges to a limiting value $\mathbf{s}$.

Let $\mathcal{S}_n^*$ be the value of the objective function at the optimum: $\mathcal{S}_n^* = \inf \mathcal{S}_n$. Function $\mathcal{S}_n$ satisfies epi–convergence (Lemma 3 and 4). Given Lemma 2, we can use epi–convergence to prove that $\mathbb{S}_n^*$ converges to $\mathbb{S}^*$, i.e., $\mathbf{s}_n^*$ converges to $\mathbf{s}^*$.

Let Γ be the set defined as $\Gamma = \{\mathbf{b} : \sup\{b_l\} = \infty\}$ (note that Γ has measure 0 under Assumption 1). For every realization $\mathbf{b}$ of the service times such that $\mathbf{b} \notin \Gamma$, the following holds (this result is in Theorem 3.2 in (Robinson, 1996), page 519):

(a) $\mathcal{S}^* \leq \limsup \mathcal{S}_n^*$;

(b) if $\mathbf{s}_n$ is a sequence converging to $\mathbf{s}$ and if, for each n, $\mathbf{s}_n \in \mathbb{S}_n^*$, then $\mathbf{s} \in \mathbb{S}^*$.

Epi–convergence results in $\mathbf{s} \in \mathbb{S}^*$, hence the sample path optimal solution converges to the infinite sample path optimal solution. □